Rochester Institute of Technology

PHYS-377 **2020-2021 Spring**

Advanced Computational Physics

Chaotic Systems - Double Pendulum

Ramsey Phuc

May 04, 2021

1 Introduction and Background

1.1 Introduction

A dynamical system, in physics, typically describes the state of a particle that varies over time. One particular example of a dynamical system in physics is the double pendulum. A single pendulum consists of one object (a bob) that is connected to a fixed point by an non-stretchable string. A double pendulum is an extension of the single pendulum where you attach another bob at the end of the single pendulum. Unlike a single pendulum, the double pendulum has a strong sensitivity to its initial conditions, which causes the double pendulum to behave in a chaotic way.

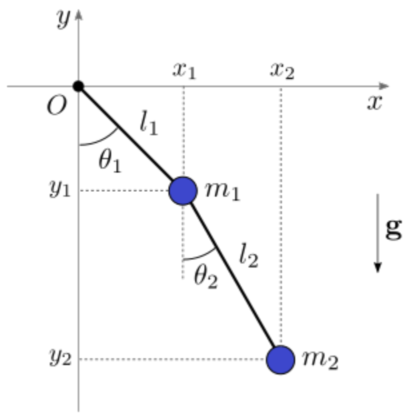

Figure 1: A visualization of a double pendulum

1.2 Goals

1. Derive the potential, kinetic, and total energies of a generic double pendulum system.
2. Derive the system of equations that represents the motion for a generic double pendulum system.
3. Solve the system of equations numerically using a custom ODE solver. This solver uses the Runge-Kutta Order 4 method.
4. Create an animation of the double pendulum along with some animated graphs. These animated graphs are meant to be in sync with the double pendulum animation.
5. Show that the Double Pendulum system is a chaotic system by showing the differences a small change in a initial condition can do.

1.3 Fundamental Physics

Figure 1 captures a frame of the double pendulum. Here we will set the initial anchor at the origin O. x_i represents the horizontal distance from the origin to bob i, y_i represents the vertical distance from the origin to bob i, θ_i represents the angle bob i makes between the anchor or the bob above it and the vertical.

We can derive the equations that are necessary to show the behavior of this chaotic system. Lets start by deriving the potential, kinetic, and the total energy of this system. From figure 1, we can determine the positions of each bob.

$$\begin{cases} x_1 = l_1 \sin(\theta_1) \\ y_1 = -l_1 \cos(\theta_1) \end{cases}, \quad \begin{cases} x_2 = l_1 \sin(\theta_1) + l_2 \sin(\theta_2) \\ y_2 = -l_1 \cos(\theta_1) - l_2 \cos(\theta_2) \end{cases}$$

Using these equations, we can determine the potential energy (PE) of the system. The potential energy of a double pendulum is:

$$\begin{aligned} PE &= m_1 g h_1 + m_2 g h_2 \\ &= m_1 g y_1 + m_2 g y_2 \\ &= -m_1 g l_1 \cos(\theta_1) - m_2 g l_1 \cos(\theta_1) - m_2 g l_2 \cos(\theta_2) \\ &\boxed{PE = -g l_1 [m_1 + m_2] \cos(\theta_1) - m_2 g l_2 \cos(\theta_2)} \end{aligned}$$

To determine the kinetic energy (KE) of this system, we will need the velocity of each bob. That can be done by first taking the derivative of each bobs' component:

$$\begin{cases} v_{x_1} = l_1 \dot{\theta}_1 \cos(\theta_1) \\ v_{y_1} = l_1 \dot{\theta}_1 \sin(\theta_1) \end{cases}, \quad \begin{cases} v_{x_2} = l_1 \dot{\theta}_1 \cos(\theta_1) + l_2 \dot{\theta}_2 \cos(\theta_2) \\ v_{y_2} = l_1 \dot{\theta}_1 \sin(\theta_1) + l_2 \dot{\theta}_2 \sin(\theta_2) \end{cases}$$

and then taking the sum of the square of each component:

$$v_n^2 = v_{x_n}^2 + v_{y_n}^2 \implies \begin{cases} v_1^2 = l_1^2 \dot{\theta}_1{}^2 \\ v_2^2 = l_1^2 \dot{\theta}_1{}^2 + l_2^2 \dot{\theta}_2{}^2 + 2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) \end{cases}$$

The kinetic energy of a double pendulum is:

$$\begin{aligned} KE &= \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 \\ &= \frac{1}{2} m_1 l_1^2 \dot{\theta}_1{}^2 + \frac{1}{2} m_2 \left(l_1^2 \dot{\theta}_1{}^2 + l_2^2 \dot{\theta}_2{}^2 + 2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) \right) \\ &\boxed{KE = \frac{[m_1 + m_2] l_1^2 \dot{\theta}_1{}^2}{2} + \frac{m_2 l_2^2 \dot{\theta}_2{}^2}{2} + m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2)} \end{aligned}$$

Thus the total energy (TE) of the double pendulum system is:

$$TE = PE + KE$$

$$\boxed{TE = -g l_1 [m_1 + m_2] \cos(\theta_1) - m_2 g l_2 \cos(\theta_2) + \frac{[m_1 + m_2] l_1^2 \dot{\theta}_1{}^2}{2} + \frac{m_2 l_2^2 \dot{\theta}_2{}^2}{2} + m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2)}$$

Now lets derive the equations that capture the motion of the system. To do this, we will need to determine the Lagrangian:

$$L = KE - PE$$
$$L = \frac{[m_1+m_2]l_1^2\dot{\theta_1}^2}{2} + \frac{m_2 l_2^2\dot{\theta_2}^2}{2} + m_2 l_1 l_2 \dot{\theta}_1\dot{\theta}_2\cos(\theta_1-\theta_2) + g l_1[m_1+m_2]\cos(\theta_1) + m_2 g l_2\cos(\theta_2)$$

For each angle, we will use the Euler-Lagrange equation. The Euler-Lagrange equation is:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial\dot{\theta}}\right) - \frac{\partial L}{\partial\theta} = 0$$

So for θ_1:

$$\begin{aligned}
0 &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial\dot{\theta}_1}\right) - \frac{\partial L}{\partial\theta_1} \\
&= \frac{\mathrm{d}}{\mathrm{d}t}\Big([m_1+m_2]l_1^2\dot{\theta}_1 + m_2 l_1 l_2\dot{\theta}_2\cos(\theta_1-\theta_2)\Big) - \Big(-m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) - g l_1[m_1+m_2]\sin(\theta_1)\Big) \\
&= [m_1+m_2]l_1^2\ddot{\theta}_1 + m_2 l_1 l_2\ddot{\theta}_2\cos(\theta_1-\theta_2) + m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) + g l_1[m_1+m_2]\sin(\theta_1) \\
0 &= [m_1+m_2]l_1\ddot{\theta}_1 + m_2 l_2\ddot{\theta}_2\cos(\theta_1-\theta_2) + m_2 l_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) + g[m_1+m_2]\sin(\theta_1)
\end{aligned}$$

and for θ_2:

$$\begin{aligned}
0 &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial\dot{\theta}_2}\right) - \frac{\partial L}{\partial\theta_2} \\
&= \frac{\mathrm{d}}{\mathrm{d}t}\Big(m_2 l_2^2\dot{\theta}_2 + m_2 l_1 l_2\dot{\theta}_1\cos(\theta_1-\theta_2)\Big) - \Big(m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) - m_2 g l_2\sin(\theta_2)\Big) \\
&= m_2 l_2^2\ddot{\theta}_2 + m_2 l_1 l_2\ddot{\theta}_1\cos(\theta_1-\theta_2) - m_2 l_1 l_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) + m_2 g l_2\sin(\theta_2) \\
0 &= l_2\ddot{\theta}_2 + l_1\ddot{\theta}_1\cos(\theta_1-\theta_2) - l_1\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) + g\sin(\theta_2)
\end{aligned}$$

So the equations that represent the motion of a double pendulum are:

$$\begin{cases}
[m_1+m_2]l_1\ddot{\theta}_1 + m_2 l_2\ddot{\theta}_2\cos(\theta_1-\theta_2) + m_2 l_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) + g[m_1+m_2]\sin(\theta_1) & = 0 \\
l_2\ddot{\theta}_2 + l_1\ddot{\theta}_1\cos(\theta_1-\theta_2) - l_1\dot{\theta}_1\dot{\theta}_2\sin(\theta_1-\theta_2) + g\sin(\theta_2) & = 0
\end{cases}$$

or alternatively:

$$\begin{cases}
\ddot{\theta}_1 = \dfrac{-g(2m_1+m_2)\sin(\theta_1) - m_2 g\sin(\theta_1-2\theta_2) - 2m_2\sin(\theta_1-\theta_2)[\dot{\theta}_2^2 l_2 + \dot{\theta}_1^2 l_1\cos(\theta_1-\theta_2)]}{l_1(2m_1+m_2-m_2\cos(2(\theta_1-\theta_2)))} \\
\ddot{\theta}_2 = \dfrac{2(\dot{\theta}_1^2 l_1(m_1+m_2) + (m_1+m_2)g\cos(\theta_1) + \dot{\theta}_2^2 l_2 m_2\cos(\theta_1-\theta_2))\sin(\theta_1-\theta_2)}{l_2(2m_1+m_2-m_2\cos(2(\theta_1-\theta_2)))}
\end{cases}$$

2 Program Manual

2.1 Summary of the program

This program only contains one file that does everything. The purpose of this program is to not only solve the equations of motion of the double pendulum system, but also generate an animation of a double pendulum system with different animated plots that are in sync with animation of the double pendulum system.

2.2 Installation requirements

For this program to work, you need to be able to successfully run a python file. For this file, it is best to have an IDE to not only edit the file, but also importing built-in packages that are necessary to run the program. Once you are able to import the necessary packages, you should be able to run the python file.

2.3 Details on the program

After the user has decided on the value of the initial conditions and the parameters, the user can run the program. Upon running the program, the first thing that the program will do is solve the system of equations mentioned in Section 1. Since there are differential equations of order greater than 1, we will break those equations down so we have a system of order 1 differential equations. The following is a part of the code that captures the system of order 1 differential equations:

```python
def df(F, t, params):
    # Unpack variables
    theta1, theta2, omega1, omega2 = F

    # Unpack parameters
    m1, m2, l1, l2, g = params

    # System of differential equations
    theta1_t = omega1
    theta2_t = omega2

    # Breaking down each component since the expression is too big
    A1 = -g * (2 * m1 + m2) * np.sin(theta1)
    A2 = - m2 * g * np.sin(theta1 - 2 * theta2)
    A3 = - 2 * m2 * np.sin(theta1 - theta2)
    A4 = (omega2 ** 2) * l2 + (omega1 ** 2) * l1 * np.cos(theta1 - theta2)
    B = l1 * (2 * m1 + m2 - m2 * np.cos(2 * (theta1 - theta2)))
    omega1_t = (A1 + A2 + A3 * A4) / B

    # Breaking down each component since the expression is too big
    C1 = 2 * np.sin(theta1 - theta2)
    C2 = (omega1 ** 2) * l1 * (m1 + m2)
    C3 = g * (m1 + m2) * np.cos(theta1)
    C4 = (omega2 ** 2) * l2 * m2 * np.cos(theta1 - theta2)
    D = l2 * (2 * m1 + m2 - m2 * np.cos(2 * (theta1 - theta2)))
    omega2_t = C1 * (C2 + C3 + C4) / D
    return np.array([theta1_t, theta2_t, omega1_t, omega2_t], dtype=float)
```

The method used to solve this system of differential equations is the Runge-Kutta order 4 method. The following is my implementation of the Runge-Kutta order 4:

```python
def rk4(df, r, ts, h, params):
    """
    Runge-Kutta 4 Method

    :param df: System of ODE(s) to solve
    :param r: Initial Condition vector
    :param ts: time
    :param h: Step Size
    :param params: Parameters/Constants in the differential equation
    :return: Approximated solution
    """
    # Initialize the approximated solution
    rs = [r]

    # For each x value
    for i in range(1, len(ts)):
        # Compute f1 = f(r(t), t)
        f1 = df(rs[i - 1], ts[i - 1], params)

        # Compute f2 = f(r(t) + (h/2)*f1, t + (h/2))
        f2 = df(rs[i - 1] + (h / 2) * f1, ts[i - 1] + (h / 2), params)

        # Compute f3 = f(r(t) + (h/2)*f2, t + (h/2))
        f3 = df(rs[i - 1] + (h / 2) * f2, ts[i - 1] + (h / 2), params)

        # Compute f4 = f(r(t) + h*f3, t + h)
        f4 = df(rs[i - 1] + h * f3, ts[i - 1] + h, params)

        # Compute r(t + h) = r(t) + (h/6)*(f1 + 2*f2 + 2*f3 + f4)
        r = rs[i - 1] + (h / 6) * (f1 + 2 * f2 + 2 * f3 + f4)

        # Record the approximated value
        rs.append(r)

    # Return rs as an array
    return get_data(rs)
```

In the above code, there is a `get_data()` function. This function is a custom function and the only purpose it serves in this code is to gather the data we want more easily. The input is an $n \times m$ array and it transposes that array. The following lines of code is the code for the `get_data()` function:

```python
def get_data(data):
    """
    Transposes the array
    """
    numRows, numCols = len(data), len(data[0])
    temp = np.empty((numCols, numRows), dtype=float)
    for r in range(0, numRows):
        for c in range(0, numCols):
            temp[c][r] = data[r][c]
    if len(temp) == 1:
        return temp[0]
    else:
        return temp
```

Once the system of differential equations is solved, the program will generate five plots based the solution to the system of differential equations. These five plots will be saved as images for the user's reference and they are plotting:

1. θ_2 vs θ_1
2. ω_2 vs ω_1
3. θ_1, θ_2 vs time
4. ω_1, ω_2 vs time
5. Potential, Kinetic, and Total Energies vs time

After the plots are generated and saved as images, the program will create an animation. Using the constants, parameters, and initial conditions that the user initialized as well as the solution to the system of differential equations, this animation will involve:

- A simulation of the double pendulum system
- θ_2 vs θ_1
- ω_2 vs ω_1
- θ_1, θ_2 vs time
- ω_1, ω_2 vs time
- Potential, Kinetic, and Total Energies vs time

3 Results and Analysis

3.1 First set of initial conditions

Consider the case with the following parameters and initial conditions:

$$\begin{cases} m_1 = 1 \text{ kg} \\ m_2 = 1 \text{ kg} \end{cases}, \begin{cases} l_1 = 1 \text{ m} \\ l_2 = 1 \text{ m} \end{cases}, \begin{cases} \theta_{1,0} = 45^\circ \\ \theta_{2,0} = 0^\circ \end{cases}, \begin{cases} \omega_{1,0} = 0^\circ/\text{s} \\ \omega_{2,0} = 0^\circ/\text{s} \end{cases}, \ 0 \le t \le 10$$

where each $\theta_{i,0}$ and $\omega_{i,0}$ are the initial starting angles and angular velocities for each bob. Then the program will generate the following plots:

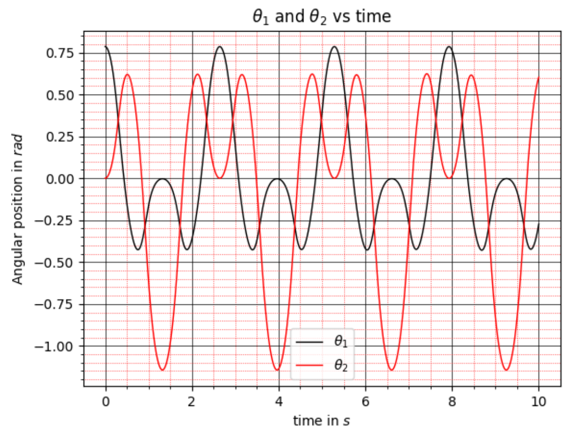

Figure 2: θ_1, θ_2 vs time

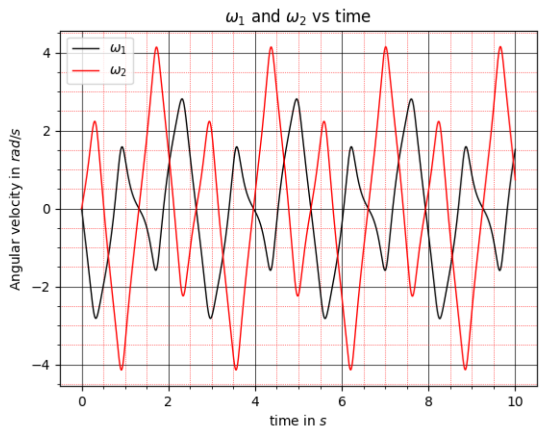

Figure 3: ω_1, ω_2 vs time

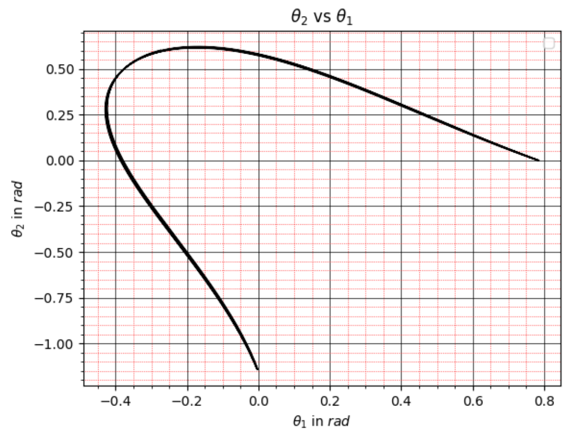

Figure 4: θ_2 vs θ_1

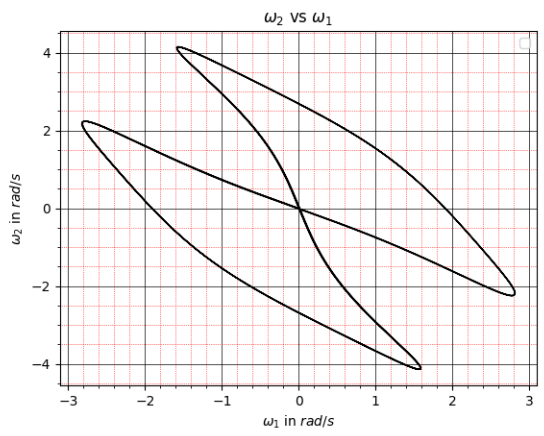

Figure 5: ω_2 vs ω_1

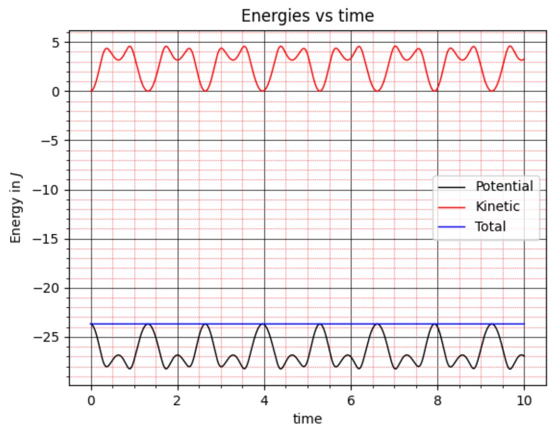

Figure 6: Potential, Kinetic, and Total Energies vs time

Figures 2 and 3 are time plots for the position and angular velocity of each bob in the double pendulum system respectively. For a time of 10 seconds, it is clear that the double pendulum system under this set of initial conditions shows a consistent periodic behavior. These plots show that the double pendulum system has a period about 2.6383 seconds. Figures 4 and 5 are phase plane plots for the position and angular velocity of each bob in the double pendulum system respectively. It is more apparent in the animation when running the code, but these phase plane plots show that there is a consistent pattern in how the system behaves. Finally, Figure 6 is a time plot that plots the Potential, Kinetic, and Total energies against time. Here we can see that not only the consistent periodic behavior exist, but every is conserved.

3.2 Slightly changing one initial condition

Lets use the same set up but change one of the initial conditions slightly, which is $\theta_{2,0} = 1°$.

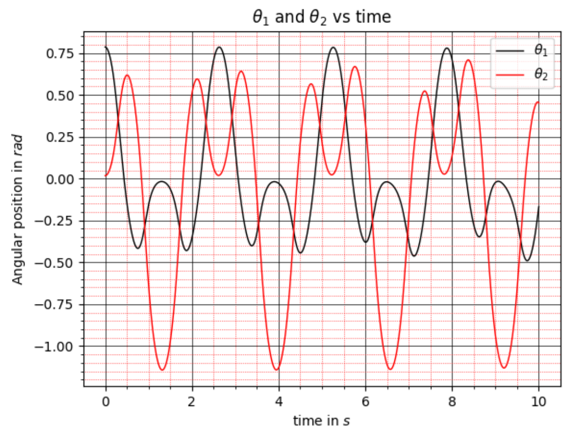

Figure 7: θ_1, θ_2 vs time

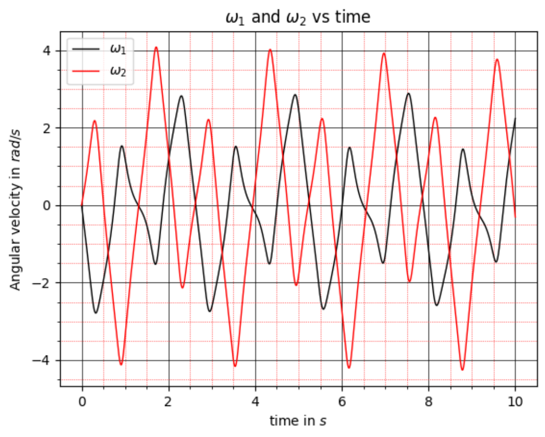

Figure 8: ω_1, ω_2 vs time

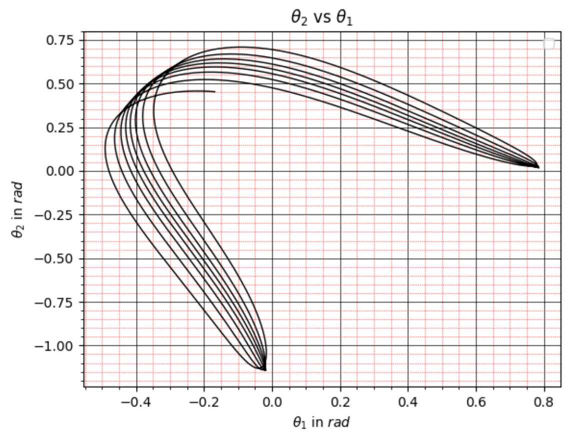

Figure 9: θ_2 vs θ_1

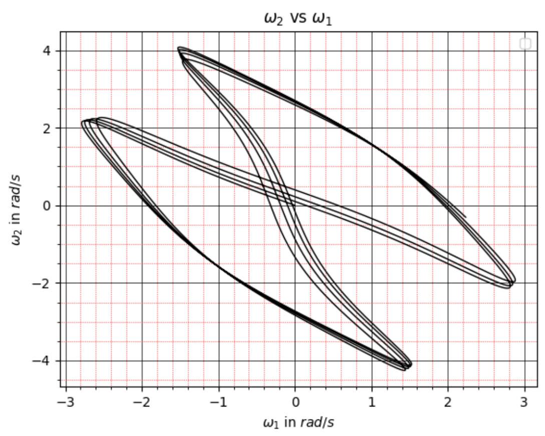

Figure 10: ω_2 vs ω_1

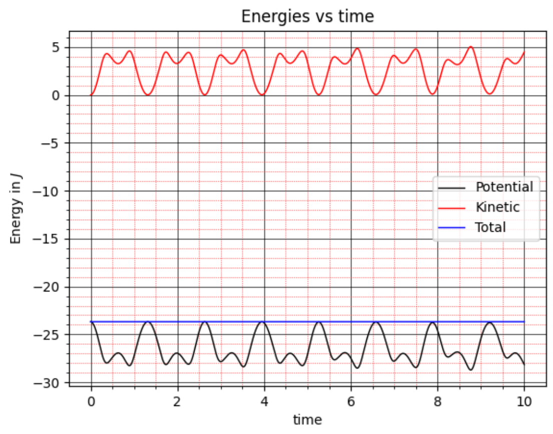

Figure 11: Potential, Kinetic, and Total Energies vs time

Compared to Figures 2 and 3, Figures 7 and 8 look very similar. However, we can see that the bobs' position and angular velocity starts to differ after each period in Figure 7 and 8 respectively. The system does show a periodic behavior, but each period is slightly different than the previous period. This is more apparent when we look at Figures 9 and 10. These figures show that the system starts to be inconsistent while maintaining its periodic behavior over time. Just by increasing the initial angle of bob 2 by 1°, the system starts to diverge from its initial period. This shows that the Double Pendulum System is a chaotic system. Figure 11 shows how the energy is conserved under these initial conditions. While energy is conserved, the amount of Potential and Kinetic energy differs as time goes on, which aligns with the fact that the system behaves in a chaotic way.

4 Summary

What we have done is derived the necessary equations that capture the behavior of thee Double Pendulum system using simple mathematics, energy laws, and Lagrangian Mechanics. Using the code provided, we were able to solve the system of equations that captures the motion of the Double Pendulum system. Then, we were able to show that the Double Pendulum system is a chaotic system by showing that the behavior will change from a slight change in initial conditions.

5 References

1. https://www.myphysicslab.com/pendulum/double-pendulum-en.html

2. *Computational Physics* by Mark Newman