Modeling the Motion of a Damped Pendulum

Sean MacBride, Mohammed Hayat

1 - Project Goals

In this project, a damped pendulum of constant mass was observed. Using a camera, the position of the pendulum bob was observed. Data on the position and velocity of the pendulum bob was gathered. Using this information, it is possible to graph the angle between the pendulum arm and its equilibrium position as a function of time. The differential equation used to describe the position of the pendulum is

$$\ddot{\theta} + b\left(l\dot{\theta}\right)^n + \omega_0^2 \sin\theta = 0 \tag{1.1}$$

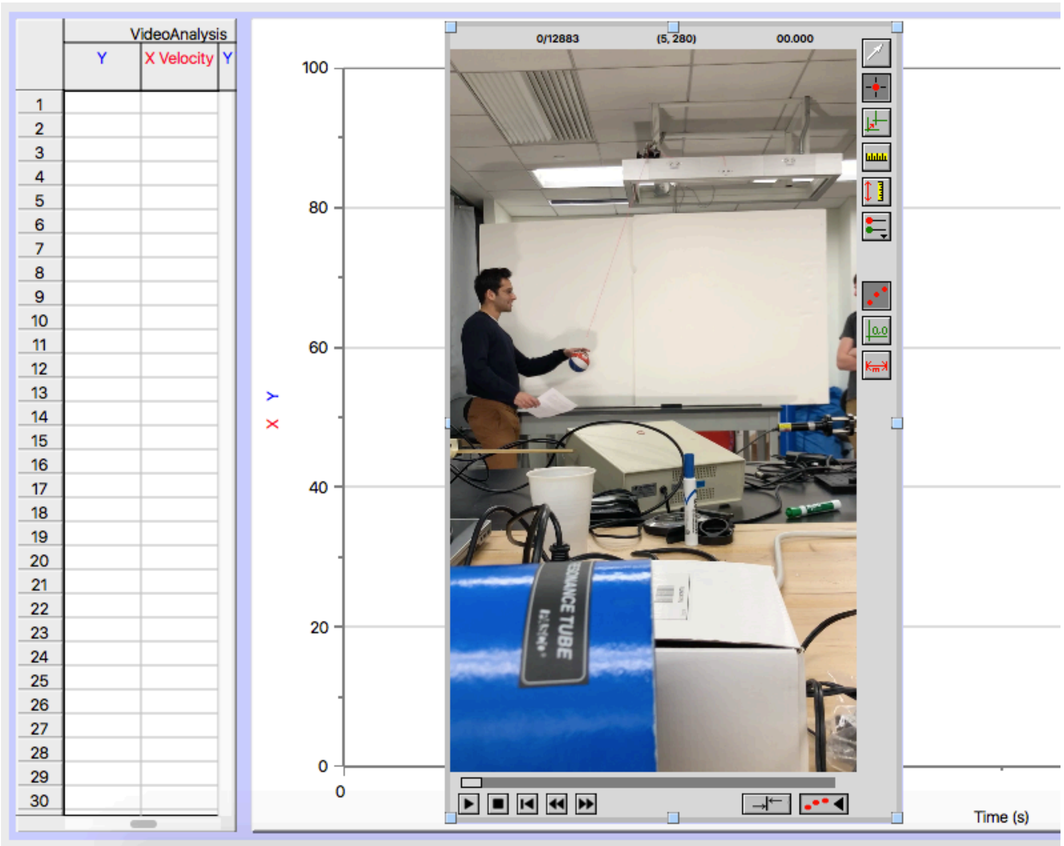
Solving this ODE numerically using the Euler forward method, it is possible to model the motion using specified damping parameters for *b* and *n*. This yields the angle theta between the equilibrium position and the pendulum arm in terms of time. Modeling theta as a function of time to match our experimental data is the main goal of this project.

2 - Experimental Apparatus

The experimental apparatus consisted of a simple pendulum design. The pivot point was located 1.15 meters above the pendulum mass. After two unsuccessful trials, it was ensured that the pendulum string was not getting caught on the apparatus holding the pivot point in place. The pendulum had a length of 1.15 meters. The point mass on the end of the string was a ball. The area behind the pendulum was illuminated by a lamp and a poster-board, to make data analysis easier to track. The motion of the pendulum bob was captured by a mobile phone placed so that it could capture the entire range of motion of the pendulum bob.

3 - Data Collection Procedure

The phone camera was placed to capture the entire motion of the pendulum bob. The camera observed the entire motion of the pendulum bob, including the equilibrium position of the pendulum. The video of the pendulum was then loaded into Logger Pro.



3.1 - Logger Pro being used to obtain the x and y coordinates of the pendulum bob

Using Logger Pro, the x and y positions can be plotted in pixels, with corresponding time values. The first data point was treated as time equals zero. This data was exported into a .csv file, and then extracted into appropriate lists in python.

---

4 - Data Analysis Procedure

Using the data gathered by Logger Pro, a list of x and y positions was compiled into python lists. Using python code and the matplotlib library, the experimental x and y positions of the pendulum bob were plotted. Law of cosines states

$$\theta(t) = \cos^{-1}\left(1 - \frac{(x(t) - x_0)^2 + (y(t) - y_0)^2}{2l^2}\right) \tag{4.1}$$

Where $x(t)$ is x position in pixels at a given time, $y(t)$ is y position in pixels at a given time, x_0 is the equilibrium position of x in pixels, y_0 is the equilibrium position of y in pixels, and l is the length of the bob in pixels.

Using data from the .csv file, the pivot position in pixels was determined to be

$$x = 177\ y = 562. \tag{4.2}$$

The equilibrium position of the bob in pixels was determined to be

$$x = 182\ y = 362 \tag{4.3}$$

The equation of the line joining the two points in pixels was determined to be

$$y = -40x + 7642 \tag{4.4}$$

Equation 4.1 for experimental theta as a function of time can therefore be rewritten as

$$\theta(t) = \cos^{-1}\left(1 - \frac{(x(t) - 182)^2 + (y(t) - 362)^2}{66284}\right) \tag{4.5}$$

The analytical equation of motion for a damped pendulum is

$$\ddot{\theta} + b\left(l\dot{\theta}\right)^n + \omega_0^2 \sin\theta = 0 \tag{1.1}$$

Where theta is the angle between the equilibrium position and the pendulum arm in terms of time, ω_0^2 is the period of the pendulum, b is the damping parameter, and n is the damping exponent. To solve this equation using the Euler forward method to numerically approximate and model the motion of the pendulum, it is required to do the Euler forward method on two equations:

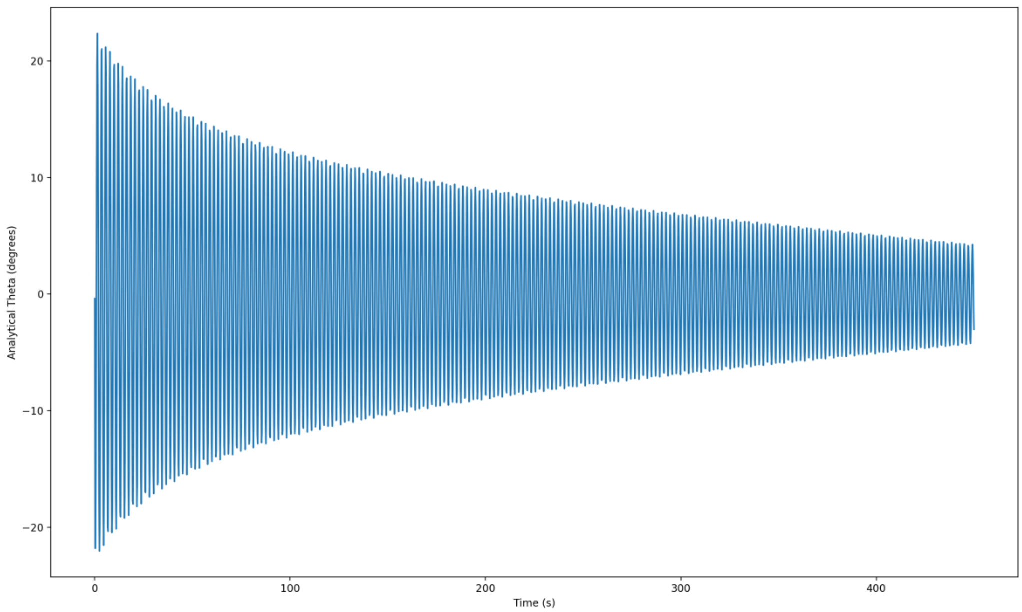
$$\dot{\theta}_{k+1} = \dot{\theta}_k + h\left(-bl^n\dot{\theta}_k^n - \omega_0^2\sin(\theta_k)\right) \tag{4.6}$$

$$\theta_{k+1} = \theta_k + h\left(\dot{\theta}_k\right) \tag{4.7}$$

The initial conditions of this equation can be obtained from seeing that $\dot{\theta}(0) = 0$ and that $\theta(0)$ can be obtained from the initial value of experimental data. $\theta(0)$ was found to experimentally be -0.38013 radians or -21.7798447 degrees.

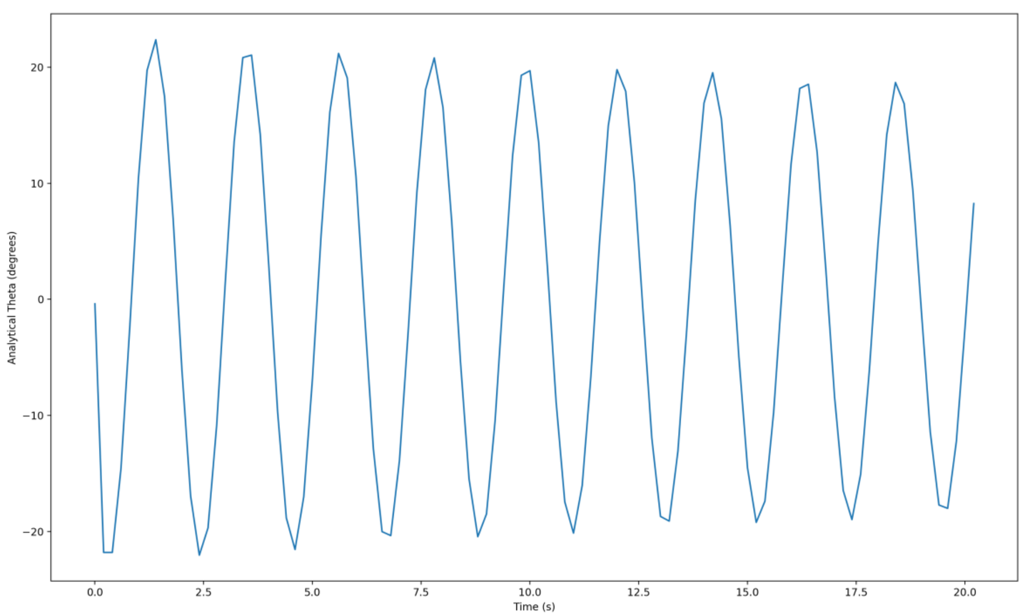
5 - Discussion

The graph of the analytical theta over the entire range of time values is as follows:



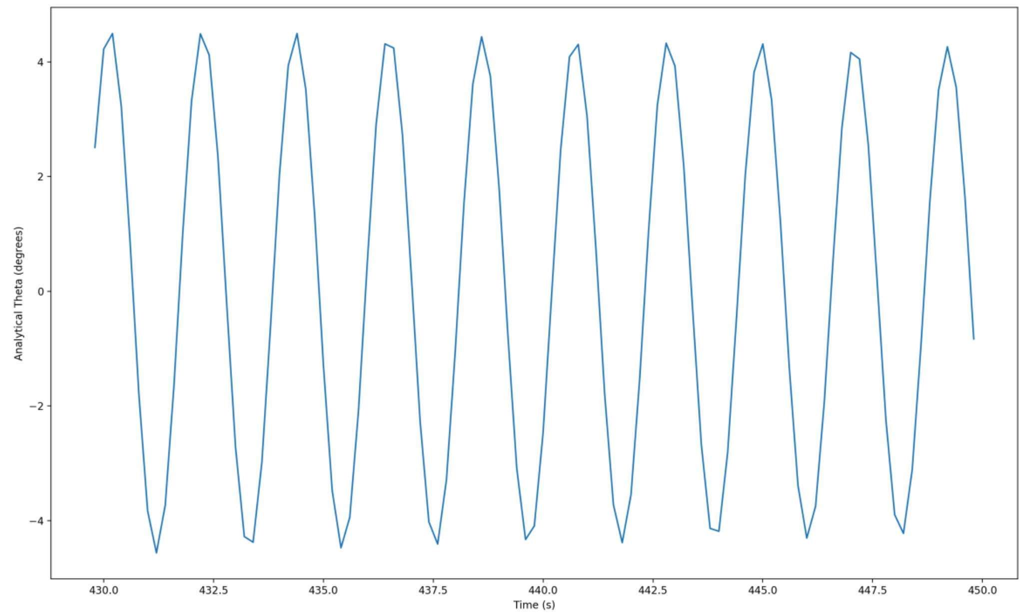
5.1 - The graph of the analytical theta

The graph of analytical theta over the first 20 seconds is as follows:



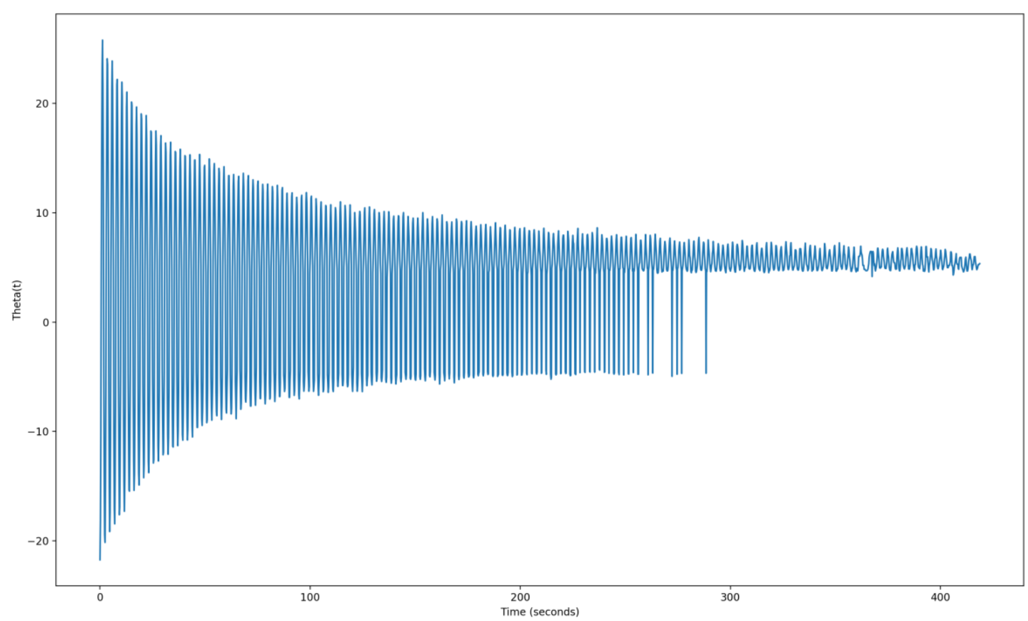
5.2 - The graph of the analytical theta

The graph of analytical theta over the last 20 seconds is as follows:



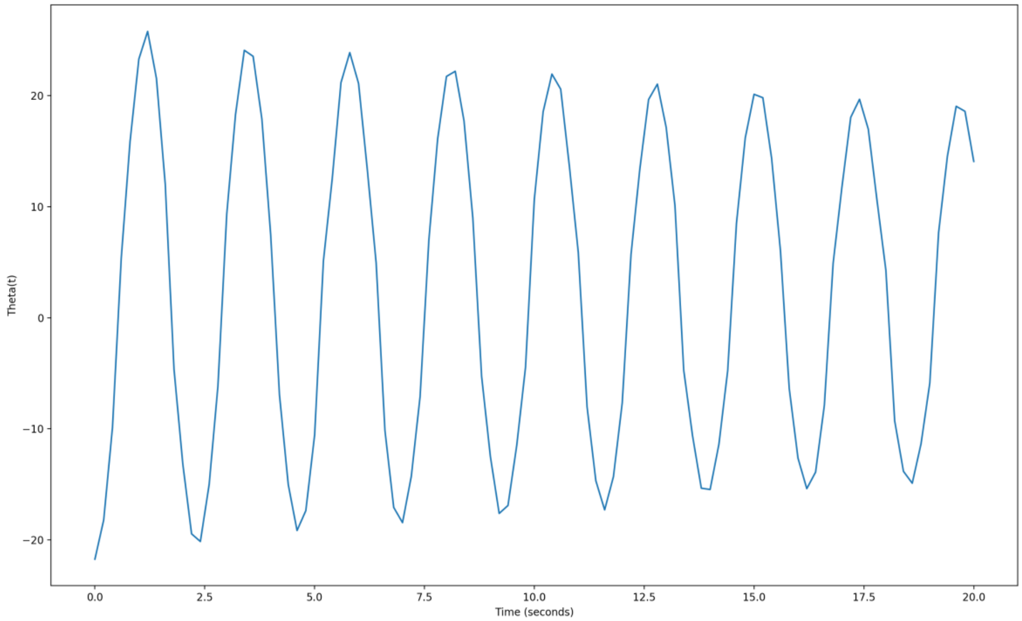
5.3 - The graph of the analytical theta

The graph of experimental theta over the full range of time values is as follows:



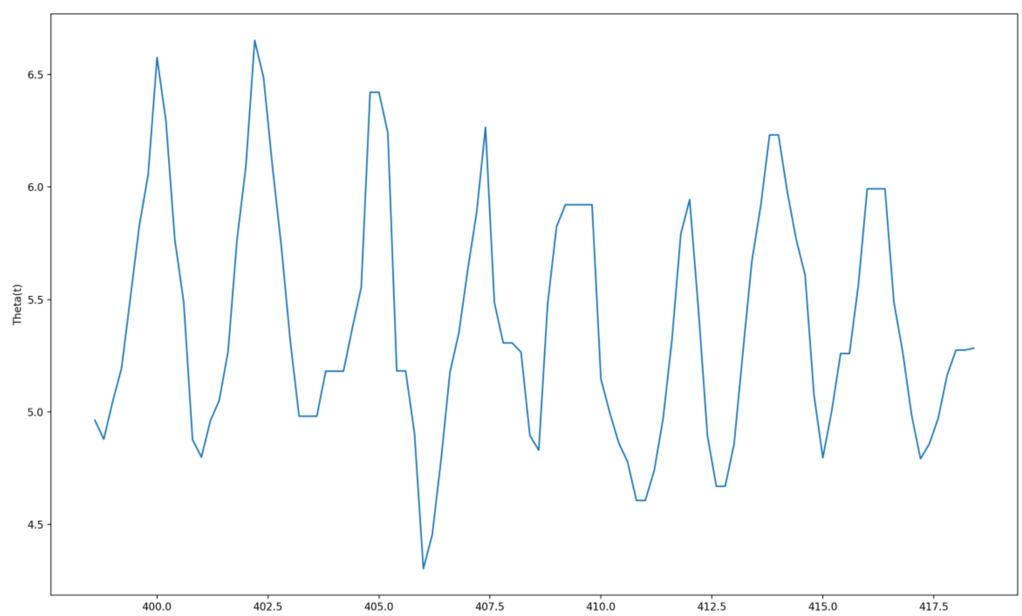
5.4 - The graph of the experimental theta

The graph of experimental theta over the first 20 seconds is as follows:



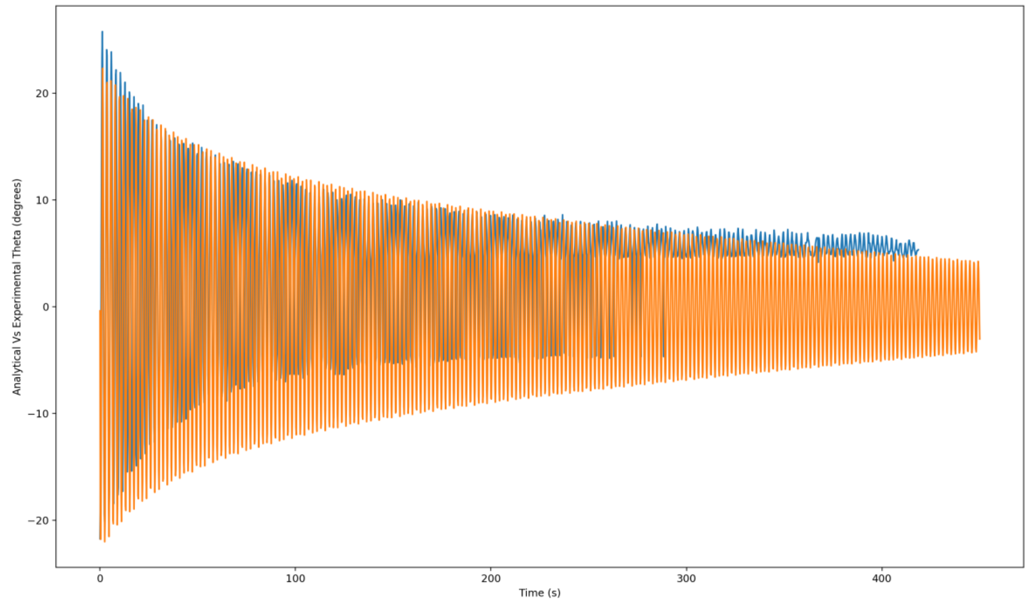
5.5 - The graph of the experimental theta

The graph of experimental theta over the last 20 seconds is as follows:



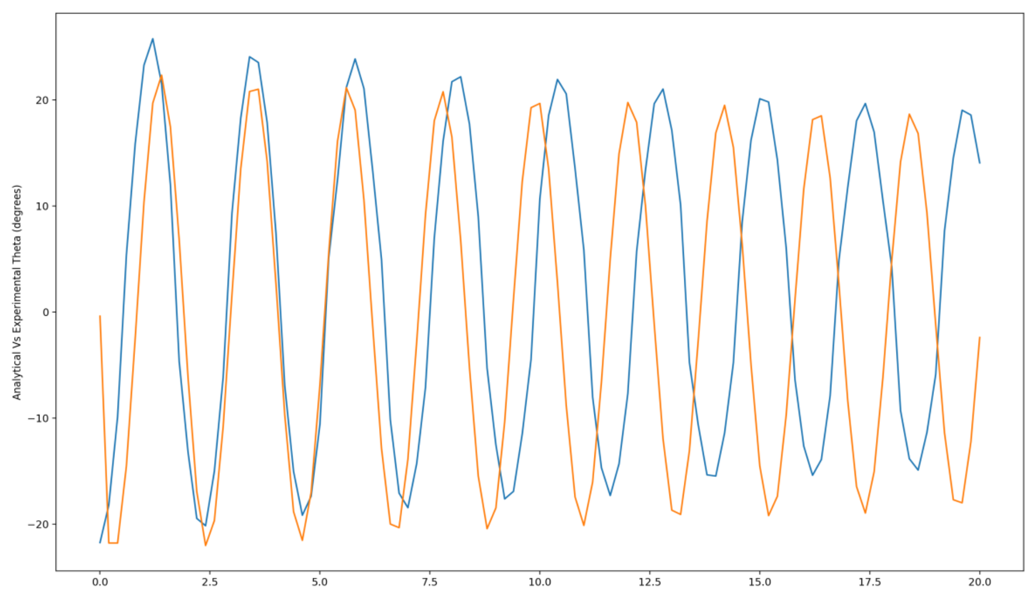
5.6 - The graph of the experimental theta

The graph of analytic theta (orange) and experimental theta (blue) over the full range of time values is as follows:



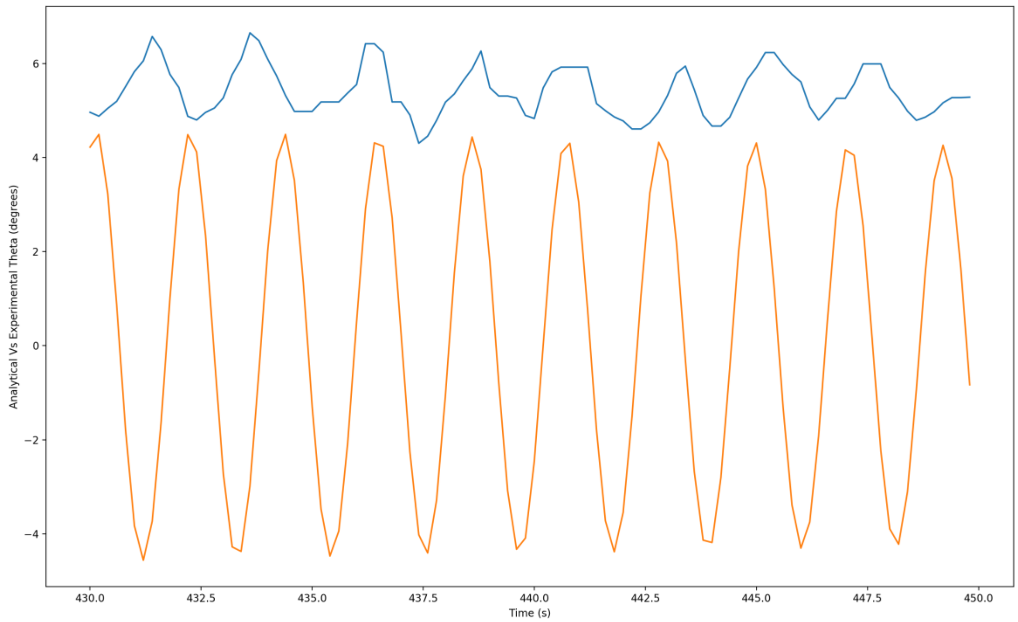
5.7 - The graph of analytic and experimental theta

The graph of analytic theta (orange) and experimental theta (blue) over the first 20 seconds is as follows:



5.8 - The graph of the analytical and experimental theta

The graph of analytic theta (orange) and experimental theta (blue) over the last 20 seconds is as follows:



5.9 - The graph of the analytical and experimental theta

The three graphs for analytical theta show a decaying sinusoidal curve, satisfying the general motion of a damped pendulum. The constants for the analytic model were $b = 1.479$, $h = 0.2$, $n = 0.995$, $l = 1.15$, and $g = 9.81976$, which is the local gravity in Norton. The experimental theta and analytical theta go in and out of phase ~35 seconds. The final 200 seconds of experimental data contain errors in majority of the data. This is likely due to mis-clicking on the data as theta values became smaller. The clicking error would be larger in the smaller theta ranges. The error is likely not a result of the code, as it plots the first 250 seconds of data accurately.

<u>6 - Conclusion</u>

The project was an excellent learning experience about the dampening effect in a pendulum. The data clearly depicts the decay of theta against time, which is caused by the dampening project. However, there were some notable challenges. The first clear challenge was the video footage. It was filmed by a smartphone camera, though these cameras are good enough for everyday use, a higher tech camera would be more suitable. The next challenge was the manual tracking of the pendulum bob in Logger Pro. This could have caused some inefficiencies in the tracking itself. This can cause theta and time graphs to have some distortion. Manual tracking could be the reason why the last 20 seconds of the experimental theta graph is disfigured. It could also be the reason why the last 30 percent of the experimental theta graph is only positive. For the future, two key things could improve this project. The first improvement being a higher quality camera to reduce any poor image quality and distortion. The second upgrade would be an automated tracking system. This system could be attached to the bob or be implemented in the tracking software (such as Logger Pro) itself, or an object tracking software. The project was a success and the result produced was expected.

---

7 - Bibliography

Maitra, Dipankar

"Alpha: Making the World's Knowledge Computable." Wolfram, Wolfram Alpha LLC, 26 Apr. 2018, 6:17, www.wolframalpha.com/input/?i=Gravity%2Bin%2BNorton%2C%2BMA.

---

8 – Appendix

1.1 – The analytical equation of motion for a damped pendulum with omega, theta, b (dampening parameter), n (dampening exponent), theta double dot.

3.1 – Image of Logger Pro software being used to track the coordinates of motion of the ball.

4.1 – The equation of theta(t) (in degrees) with y(t), y initial, x(t), x initial, l (length of rope) all in pixels

4.5 – Substituted version of 4.1

5.1 – Analytical theta degrees vs time in seconds.

5.2 – First 20 seconds of analytical theta.

5.3 – Last 20 seconds of analytical theta.

5.4 – Graph of experimental theta vs time.

5.5 – First 20 seconds of experimental theta.

5.6 – Last 20 seconds of experimental theta.

5.7 – Analytical theta vs Experimental theta in time.

5.8 – First 20 seconds of analytical vs experimental theta.

5.9 – Last 20 seconds of analytical vs experimental theta.

Code for experimental theta and graphs:

```
tlist = np.genfromtxt(ip, usecols=(0), delimiter=",", skip_header=1)
xposlist = np.genfromtxt(ip, usecols=(1), delimiter=",", skip_header=1)
yposlist = np.genfromtxt(ip, usecols=(2), delimiter=",", skip_header=1)
```

```
xvellist = np.genfromtxt(ip, usecols=(3), delimiter=",", skip_header=1)
yvellist = np.genfromtxt(ip, usecols=(4), delimiter=",", skip_header=1)

thetatlist=[]
for i in range(len(tlist)):
    thetaval=np.arccos((xposlist[i]-177)**2+(yposlist[i]-562)**2)/(79202)
    thetatlist.append(thetaval)
tlast20list=tlist[-102:len(tlist)-1]#last 20 second time stamps
tfirst20list=tlist[:102]#first 20 second time stamps
thetalast20list=thetatlist[-102:len(thetatlist)-1]
thetafirst20list=thetatlist[:102]

thetatlist=[]

for i in range(len(tlist)):
    theta = math.acos(1-(((xposlist[i]-182)**2 + (yposlist[i] - 362)**2)/(66284)))
    # theta = Cos^-1 (1 - (Xi - X0)^2 + (Yi -Y0)^2)/(2b^2)
    if (xposlist[i] < 182):
        theta = -theta
    theta = (theta*360)/(2*math.pi)
    thetatlist.append(theta)

tlast20list=tlist[-102:len(tlist)-1]#last 20 second time stamps
tfirst20list=tlist[:102]#first 20 second time stamps
thetalast20list=thetatlist[-102:len(thetatlist)-1]
thetafirst20list=thetatlist[:102]
def xygraph():
    plt.plot(xposlist,yposlist)
    plt.xlabel("X Position (Pixels)")
    plt.ylabel("Y Position (Pixels)")
    plt.show()
    input("Press <enter> to continue")
    plt.close()


def thetatall():
    plt.plot(tlist, thetatlist)
    plt.xlabel("Time (seconds)")
    plt.ylabel("Theta(t)")
    plt.show()
```

```
        input("Press <enter> to continue")
        plt.close()

    def thetalast20():
        plt.plot(tlast20list, thetalast20list)
        plt.xlabel("Time (seconds)")
        plt.ylabel("Theta(t)")
        plt.xim(400,420)
        plt.show()
        input("Press <enter> to continue")
        plt.close()

    def thetafirst20():
        plt.plot(tfirst20list, thetafirst20list)
        plt.xlabel("Time (seconds)")
        plt.ylabel("Theta(t)")
        plt.xim(0,20)
        plt.show()
        input("Press <enter> to continue")
        plt.close()
```

Code for Analytical theta and comparison graphs:

```
def thetagraph(tlist,ThetaList):
    plt.plot(tlist,ThetaList)
    plt.xlabel('Time (s)')
    plt.ylabel('Analytical Theta')
    plt.show()
    input("Press Enter to continue")
    plt.close()

def thetagraphFirst20(tlistF20,ThetaListF20):
    plt.plot(tlistF20,ThetaListF20)
    plt.xlabel('Time (s)')
    plt.ylabel('Analytical Theta')
    plt.show()
    input("Press Enter to continue")
    plt.close()

def thetagraphLast20(tlistL20,ThetaListL20):
```

```
        plt.plot(tlistL20,ThetaListL20)
        plt.xlabel('Time (s)')
        plt.ylabel('Analytical Theta')
        plt.show()
        input("Press Enter to continue")
        plt.close()

def thetaLastAnyVsExp(tlistL20,ThetaListL20, AT, Ath):
  plt.plot(tlistL20,ThetaListL20)
  plt.plot(AT,Ath)
  plt.xlabel('Time (s)')
  plt.ylabel('Analytical Vs Experimental Theta')
  plt.show()
  input("Press Enter to continue")
  plt.close()

def thetaFirstAnyVsExp(tlistF20,ThetaListF20, AT, Ath):
  plt.plot(tlistF20,ThetaListF20)
  plt.plot(AT,Ath)
  plt.xlabel('Time (s)')
  plt.ylabel('Analytical Vs Experimental Theta')
  plt.show()
  input("Press Enter to continue")
  plt.close()

def thetaAnyVsExp(tlist,Theta, AT, Ath):
  plt.plot(tlist,Theta)
  plt.plot(AT,Ath)
  plt.xlabel('Time (s)')
  plt.ylabel('Analytical Vs Experimental Theta')
  plt.show()
  input("Press Enter to continue")
  plt.close()

def main():

        told=0
        ThetaDotOld=0
        ThetaOld=-0.38013
        h=0.2#timestep, decide for yourself
```

```
b=1.189#play with the damping parameter
n=1.025#play with this too (check sheet)
l=1.15#length in pixels
g=9.81976 #local gravity in norton
omegaknot= 6.5465

Fdrag = 0
FdragList = []
FdragList.append(0)
tlist=[]
ThetaDotList=[]
ThetaList=[]

tlist.append(told)
ThetaDotList.append(ThetaDotOld)
ThetaList.append(ThetaOld)

for i in range(1,int(450/h)):
    Fdrag = ((-b) * (l**n) * abs(ThetaDotOld)**n)
    if (ThetaDotOld < 0):
        Fdrag = -Fdrag

    tnew=told+h
    ThetaDotNew= ThetaDotOld + (h*(Fdrag - omegaknot*math.sin(ThetaOld)))
    ThetaNew=ThetaOld + h*ThetaDotOld

    tlist.append(tnew)

    ThetaList.append(ThetaOld*(180/math.pi))
    ThetaDotList.append(ThetaDotNew)

    FdragList.append(Fdrag)

    told=tnew
    ThetaDotOld=ThetaDotNew
    ThetaOld=ThetaNew

#tlast20list=tlist[-102:len(tlist)-1]#last 20 second time stamps
#tfirst20list=tlist[:102]#first 20 second time stamps
#thetalast20list=Thetatlist[-102:len(thetatlist)-1]
```

```
    #thetafirst20list=thetatlist[:102]

    AtFirst20 = tlist[:102]
    AtLast20 = tlist[-102:len(tlist)-1]

    AThetaLast20 = ThetaList[-102:len(ThetaList)-1]
    AThetaFirst20 = ThetaList[:102]

    thetagraph(tlist,ThetaList)

    thetagraphFirst20(AtFirst20,AThetaFirst20)

    thetagraphLast20(AtLast20,AThetaLast20)

    thetaLastAnyVsExp(AtLast20,thetalast20list, AtLast20, AThetaLast20)

    thetaFirstAnyVsExp(AtFirst20,thetafirst20list, AtFirst20, AThetaFirst20)

    thetaAnyVsExp(tlist2,thetatlist,tlist,ThetaList)

main()
```