Sumer Malhotra
3035347457

PHYS2160 Final Report

Damped Forced Vibrations of a Spring Mass System

1. Introduction

This project explores uses Python as a tool in solving a second-order differential equation concerned with damping forced vibrations of a vertically suspended spring mass system.

2. Theory

We will iteratively build up the intuition that leads to the final differential equation to solve.

Consider an object of mass *m* suspended at the end of a vertical spring with spring constant *k*. By Hooke's Law, we know that if the object is stretched *x* units away from its equilibrium length, a restoring force = -*kx* is exerted by the spring on the object (depicted in figure 1). By Newton's second law of motion, we know force is equal to mass times acceleration. Combining all of this information, we end up with our preliminary differential equation:

$$m\frac{d^2x}{dt^2} = -kx \qquad \text{or} \qquad m\frac{d^2x}{dt^2} + kx = 0$$

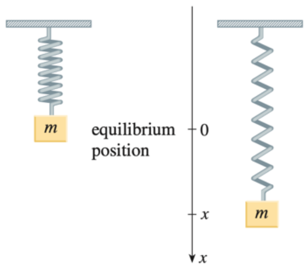


Figure 1

Next, we consider the medium through which the spring is moving. There is a damping force which acts upon the spring and object system as it vibrates in the medium. This damping force is provided by viscosity and is proportional to the velocity of the object, but in a direction opposite to it. We know that damping force = $-c\frac{dx}{dt}$ where *c* is a positive constant called the damping constant.

Now, by F = ma, we get restoring force + damping force = ma, which gives:

$$m \frac{d^2x}{dt^2} + c \frac{dx}{dt} + kx = 0$$

Finally, consider an external force F(t), then by Newton's second law we get mass times acceleration = restoring + damping + external forces. Accordingly, the equation becomes:

$$m \frac{d^2x}{dt^2} + c \frac{dx}{dt} + kx = F(t)$$

This external force is usually a periodic force function of the form: $F(t) = F_0 \cos \omega_0 t$

If w_o = w, then the applied frequency will equal the natural frequency and cause very large amplitudes. This phenomenon is called resonance.

3. Algorithm

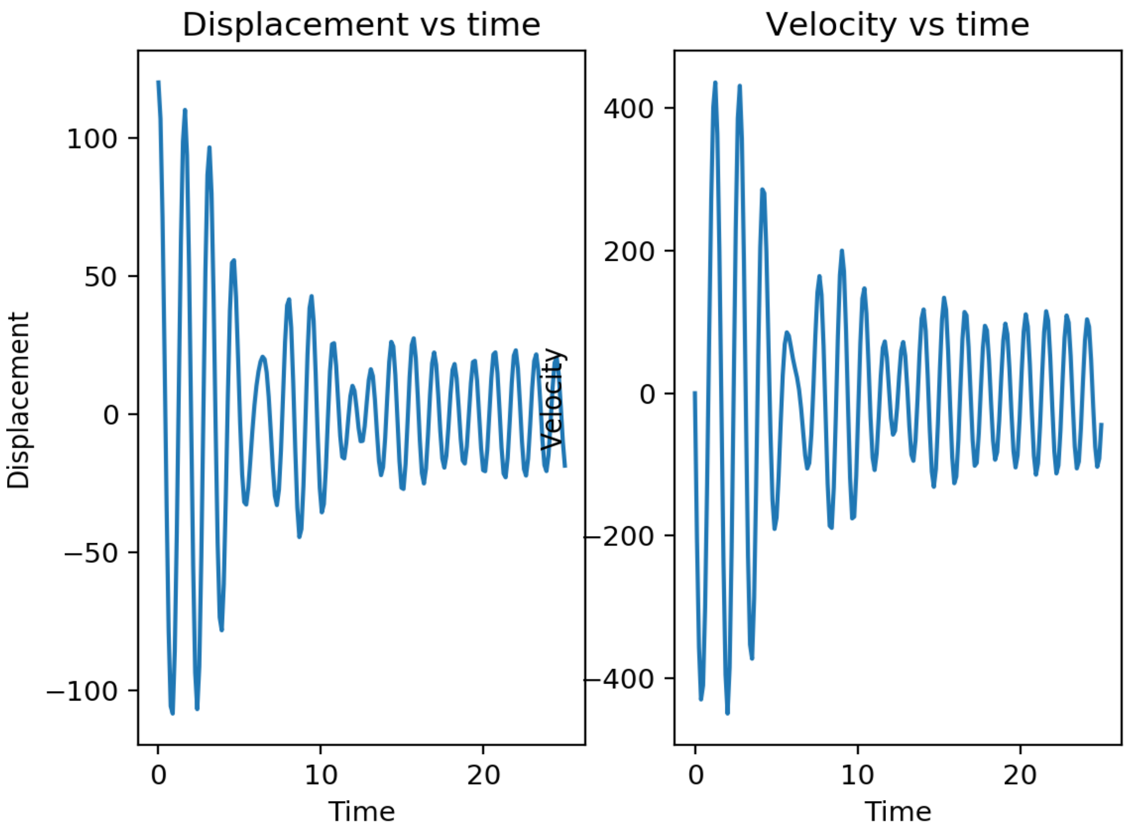
1. Start
2. Declare constants and starting values
3. Create a function which is used to integrate the second order differential equation
4. Integrate the function and store results in appropriate variables using scipy.integrate.odeint
5. Create a 3D animation of the block vibrating on a spring in air using vpython
6. Plot the displacement vs time, velocity vs time graphs using matplotlib
7. Stop

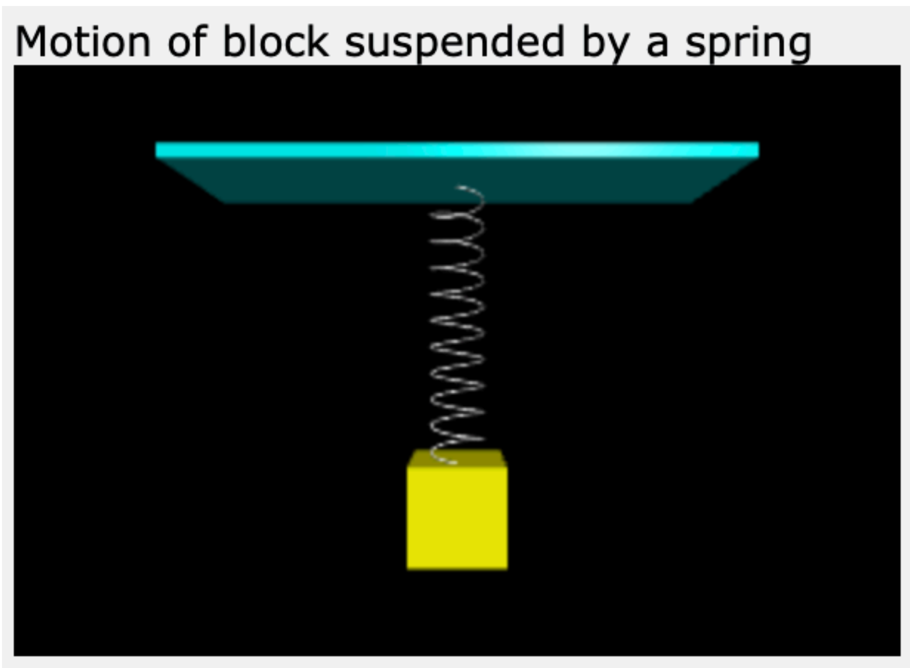
4. Program

```
import math
import vpython
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt

def dZ_dt(Z, t):
    return [Z[1], (F_0 * math.cos(omega_0 * t) - c * Z[1] - k * Z[0]) / m]

m = 0.5
c = 0.2
k = 8
F_0 = 100
omega_0 = 5
Z0 = [120, 0]
ts = np.linspace(0, 25, 200)
Zs = odeint(dZ_dt, Z0, ts)
ys = Zs[:,0]
zs = Zs[:, 1]

vpython.scene.height = 200
vpython.scene.width = 300
vpython.scene.title = "Motion of block suspended by a spring"
vpython.scene.center = vpython.vector(0,2,0)
ceiling = vpython.box(pos=vpython.vector(0,4.8,1), axis=vpython.vector(1,0,0),
                      length=8, width=2, height=0.2, color=vpython.color.cyan)
spring = vpython.helix(pos=vpython.vector(0,0.7,1), axis=vpython.vector(0,1,0),
            radius=0.4, length=4, coils=10)
block = vpython.box(pos=vpython.vector(0,0,1), axis=vpython.vector(1,0,0),
length=1.4, width=1.4, height=1.4, color=vpython.color.yellow)

for y in ys:
    vpython.rate(50)
    _y = y/100
    block.pos = vpython.vector(0,0 + _y,1)
    spring.pos = vpython.vector(0,0.7 + _y,1)

plt.subplot(1, 2, 1)
plt.plot(ts,ys)
plt.title("Displacement vs time")
plt.xlabel("Time")
plt.ylabel("Displacement")
plt.subplot(1, 2, 2)
plt.plot(ts,zs)
plt.title("Velocity vs time")
plt.xlabel("Time")
plt.ylabel("Velocity")
plt.show()
```

5. Results

In this section, snippets of the output (including displacement vs time, velocity vs time graph and screenshot of animation created using vpython module) will be posted.





6. Discussion

For the displacement vs time and velocity vs time graphs, we clearly notice a constant damping as time t increases. This has to do with the damping constant associated with the equation.

For the animation created for the motion of the block, the program iteratively updates the position of the block and spring, thereby showing the movement. Further to this, we can also appropriately modify the length of the spring to make the animation smoother.