

**BACHELOR OF SCIENCE
HONOURS IN PHYSICS
DEPARTMENT OF SCIENCES
2024-2025**



**Project Reports
Modeling and Analysis of Simple, Damped,
Forced Harmonic Oscillators by
Python Programming
2024-2025**

Submitted By:

Aashish Tharu Gamuwa

Q.ID: 23600004

Roll No: 2303304005

Submitted To:

Dr. Rupali Mishra

Table of Contents

1. Introduction	3
2. Simple Harmonic Oscillator	3
2.1 Governing Differential Equation	3
2.2 General Solution and Interpretation	4
2.3 Phase Space Representation	4
2.4 Energy Analysis	4
3. Damped Harmonic Oscillator	5
1. Underdamped ($\zeta < 1$):	6
Critically Damped ($\zeta = 1$):	6
3. Overdamped ($\zeta > 1$):	6
4. Forced Harmonic Oscillator	7
4.1 Mathematical Foundation	7
4.2 General Solution	7
4.3 Amplitude of Steady-State Response	7
4.4 Resonance	8
4.5 Phase Angle (δ)	8
5. System Behavior Analysis	8
5.1 Free vs Damped Motion	8
5.2 Resonance in Forced Oscillations	9
5.3 Phase Plots and Amplitude Response	9
6. Numerical Modeling and Simulations	12
7. Discussion and Analysis	15
7.1 Simple Harmonic Oscillator	15
7.2 Damped Harmonic Oscillator	15
7.3. Forced Harmonic Oscillator	16
8. Conclusion	16

1. Introduction

Oscillatory systems are fundamental to understanding diverse natural and engineered phenomena. At the core lies **harmonic motion**, governed by a restoring force that seeks to return the system to equilibrium. This motion serves as an essential model for analyzing vibrations and wave-like behaviors across physics, engineering, and applied sciences.

This study explores three primary forms of harmonic motion: **simple harmonic motion (SHM)**, **damped harmonic motion**, and **forced harmonic motion**. SHM idealizes undisturbed, energy-conserving systems, while damping introduces realistic energy losses, leading to decaying amplitudes. Forced motion, driven by external periodic forces, reveals complex behaviors such as **resonance**.

Understanding these motions provides critical insights into system dynamics and underpins practical applications in areas ranging from mechanical design to signal processing. This paper offers a comparative theoretical framework for each oscillatory model.

2. Simple Harmonic Oscillator

The **Simple Harmonic Oscillator (SHO)** serves as the foundational model for understanding linear oscillatory systems. It is described by a linear second-order differential equation and exhibits sinusoidal behavior under idealized, non-dissipative conditions. Despite its simplicity, the SHO model provides profound insights into a wide range of physical systems, including mechanical vibrations, electrical circuits, molecular dynamics, and quantum harmonic oscillators.

2.1 Governing Differential Equation

Starting from Newton's Second Law:

$$F = ma = m \frac{d^2x}{dt^2}$$

and combining with Hooke's Law:

$$F = -kx$$

and combining with Hooke's Law:

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

This simplifies to the standard SHO differential equation:

$$\frac{d^2x}{dt^2} + \omega_0^2 x = 0$$

where $\omega_0 = \sqrt{\frac{k}{m}}$ is the **natural angular frequency** of the system.

2.2 General Solution and Interpretation

The solution to this second-order homogeneous linear differential equation is:

$$x(t) = C_1 \cos(\omega_0 t) + C_2 \sin(\omega_0 t)$$

or, using the amplitude-phase form:

$$x(t) = A \cos(\omega_0 t + \phi)$$

with constants A (amplitude) and ϕ (initial phase), determined by initial conditions

2.3 Phase Space Representation

In phase space (position-velocity plane), the SHO exhibits circular or elliptical trajectories:

$$x(t) = A \cos(\omega_0 t), \dot{x}(t) = -A\omega_0 \sin(\omega_0 t)$$

Thus, the parametric plot (\dot{x}, x) describes a closed curve:

$$\left(\frac{x}{A}\right)^2 + \left(\frac{\dot{x}}{A\omega_0}\right)^2 = 1$$

This represents **conservative dynamics**, where the total energy remains constant, and the motion is confined to a closed orbit.

2.4 Energy Analysis

The total mechanical energy of the SHO is the sum of its kinetic and potential energy:

Kinetic Energy (T): $T = \frac{1}{2} m \dot{x}^2$

Potential Energy (V): $V = \frac{1}{2} k x^2$

Total Energy: $E = T + V = \frac{1}{2} m \dot{x}^2 + \frac{1}{2} k x^2 = \frac{1}{2} k A^2$

This energy remains constant throughout the motion, reflecting the **absence of damping or external forces**.

2.5 Eigenmode and Natural Frequency

The SHO is a classic **eigenvalue problem**. Solving:

$$\frac{d^2x}{dt^2} + \omega_0^2 x = 0$$

is equivalent to solving for the system's **natural modes** of vibration. The solution implies a **single frequency** ω_0 , characteristic of linear, single-degree-of-freedom systems. In complex systems (e.g., multi-mass systems or strings), similar analysis leads to multiple eigenfrequencies and mode shapes.

2.7 Importance in Physics and Engineering

- In **quantum mechanics**, the quantum SHO forms the basis for understanding vibrational quantization.
- In **electrical engineering**, it models LC circuits, with analogs for mass and spring being inductance and capacitance.
- In **mechanical systems**, it underpins vibration isolation, resonance, and stability analysis.

3. Damped Harmonic Oscillator

A **Damped Harmonic Oscillator** includes energy loss mechanisms such as friction or air resistance, leading to gradually reduced amplitude over time. This is a more realistic model of physical oscillating systems.

Mathematical Foundation

The motion is described by the second-order linear differential equation:

$$d^2x/dt^2 + 2\beta dx/dt + \omega^2 x = 0$$

Where:

- $x(t)$ is the displacement,
- β is the damping coefficient ($\beta = c/2m$, with c being the damping constant),

- ω_0 is the undamped natural angular frequency ($\omega_0 = \sqrt{k/m}$).

Nature of the Solution

The behavior of the system depends on the damping ratio ζ (zeta), defined as:

$$\zeta = \beta / \omega_0$$

There are **three cases** based on the value of ζ :

1. Underdamped ($\zeta < 1$):

- Oscillatory motion with gradually decreasing amplitude.
- Solution: $x(t) = Ae^{(-\beta t)} \cos(\omega_d t + \varphi)$
- $\omega_d = \sqrt{(\omega_0^2 - \beta^2)}$ is the damped angular frequency.

Critically Damped ($\zeta = 1$):

- Fastest return to equilibrium without oscillating.
- Solution: $x(t) = (A + Bt)e^{(-\beta t)}$

3. Overdamped ($\zeta > 1$):

- No oscillations; system returns slowly to equilibrium.
- Solution: $x(t) = Ae^{(r_1 t)} + Be^{(r_2 t)}$, where r_1 and r_2 are real, negative, distinct roots.

Energy Considerations

In damped systems, mechanical energy is no longer conserved:

- The total energy decreases over time, typically exponentially.
- Dissipated energy is transformed into heat or other forms due to the damping force.

4. Forced Harmonic Oscillator

A **Forced Harmonic Oscillator** involves an external time-dependent driving force acting on the system, which can sustain or amplify oscillations even in the presence of damping. This model is vital for understanding resonance and many real-world applications such as electrical circuits, mechanical systems, and acoustic vibrations.

4.1 Mathematical Foundation

The motion is governed by the differential equation:

$$d^2x/dt^2 + 2\beta dx/dt + \omega_0^2 x = F_0/m * \cos(\omega t)$$

Where:

- F_0 is the amplitude of the external force,
- ω is the angular frequency of the driving force,
- m is the mass,
- β and ω_0 are as defined previously.

4.2 General Solution

The complete solution consists of two parts:

1. **Homogeneous solution (x_h)** – Represents transient response and decays over time due to damping.
2. **Particular solution (x_p)** – Represents the steady-state response due to the external force.

Eventually, the transient solution dies out, and the system exhibits **steady-state oscillations** at the driving frequency ω .

Steady-state solution has the form:

$$x(t) = A \cos(\omega t - \delta)$$

Where:

- A is the amplitude of forced oscillation,
- δ is the phase difference between the force and the response.

4.3 Amplitude of Steady-State Response

The amplitude A depends on ω , ω_0 , and β :

$$A = (F^0/m)/\sqrt{(\omega_0^2 - \omega^2)^2 + (2\beta\omega)^2}$$

This expression shows **resonant behavior**.

4.4 Resonance

Resonance occurs when the driving frequency ω is close to the natural frequency ω_0 :

- Maximum amplitude at or near $\omega = \omega_0$ (for small damping),
- With low damping, the system can respond with large amplitude,
- Sharpness of resonance peak depends on damping (lower β = sharper peak).

4.5 Phase Angle (δ)

The phase lag between the displacement and driving force is given by:

$$\tan(\delta) = (2\beta\omega)/(\omega_0^2 - \omega^2)$$

- At low frequencies: displacement is in phase with the force,
- At resonance: $\delta \approx 90^\circ$,
- At high frequencies: displacement lags behind the force by nearly 180° .

5. System Behavior Analysis

In this section, we analyze and interpret the behavior of different types of harmonic oscillators—simple, damped, and forced—by examining their response under various physical conditions. These behaviors are critical in understanding how real-world systems oscillate, respond to energy dissipation, and react to external periodic forces.

5.1 Free vs Damped Motion

Simple Harmonic Oscillator (Free Motion)

The free, undamped oscillator moves with constant amplitude and frequency, governed by:

$$x(t) = A\cos(\omega_0 t + \phi)$$

where A is the amplitude, ω_0 is the natural frequency, and ϕ is the phase constant. Energy is conserved, oscillating between kinetic and potential forms.

Damped Oscillator (Energy Dissipation)

In real systems, damping causes the amplitude to decay over time. The general solution for underdamped motion is:

$$x(t) = Ae^{-\gamma t} \cos(\omega_d t + \phi)$$

where γ is the damping coefficient and $\omega_d = \sqrt{\omega_0^2 - \gamma^2}$ is the damped natural frequency. Energy is dissipated as heat or friction, and motion eventually ceases unless externally driven.

5.2 Resonance in Forced Oscillations

When a periodic external force $F(t) = F_0 \cos$ is applied, the system's response depends on the driving frequency ω .

- **Below Resonance:** The oscillator follows the driving force with a small amplitude.
- **At Resonance:** When $\omega = \omega_0$, the system exhibits maximum amplitude response. If damping is small, this amplitude can become extremely large—a phenomenon known as **resonance**.
- **Above Resonance:** Amplitude decreases and the system lags behind the driving force significantly.

The steady-state solution for forced motion is:

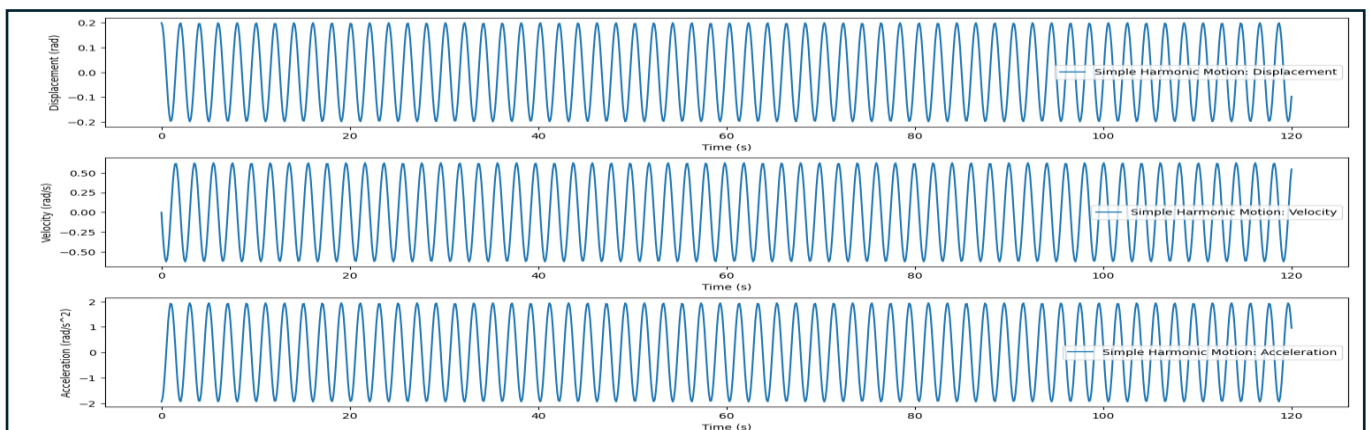
$$x(t) = A(\omega) \cos(\omega t - \delta)$$

where $A(\omega)$ is the frequency-dependent amplitude and δ is the phase lag. The amplitude peaks at resonance and drops off for higher and lower frequencies.

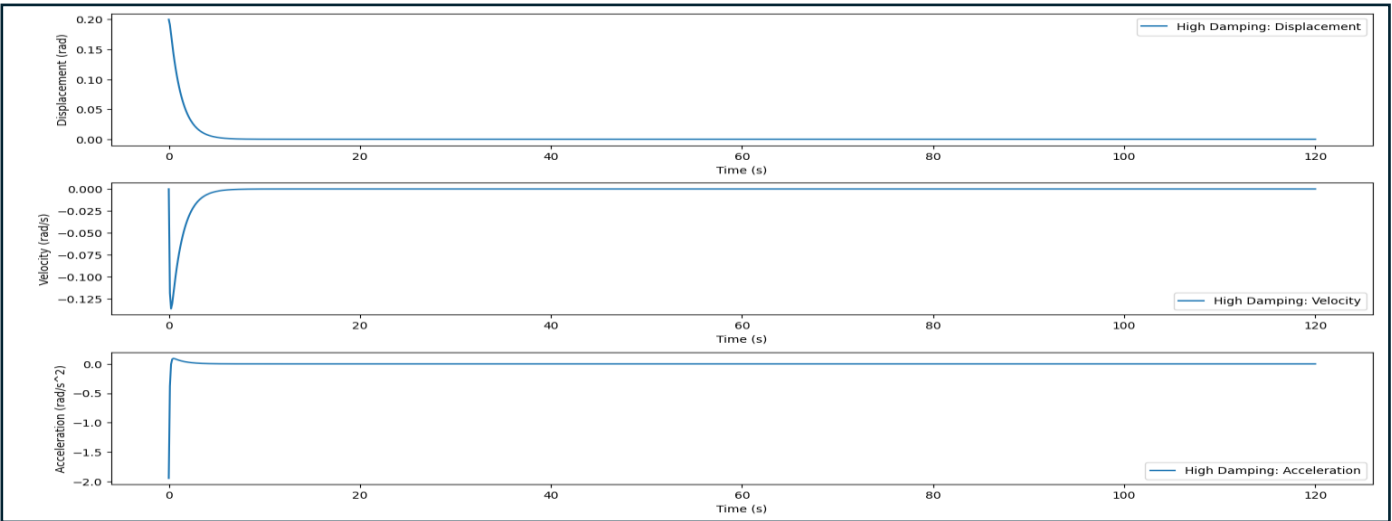
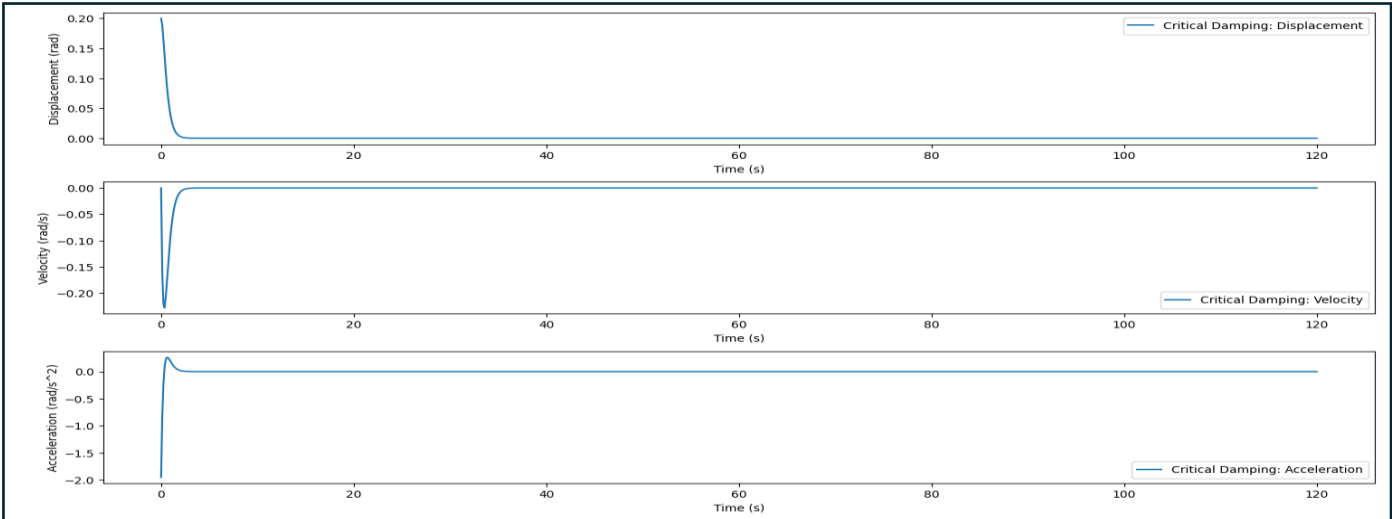
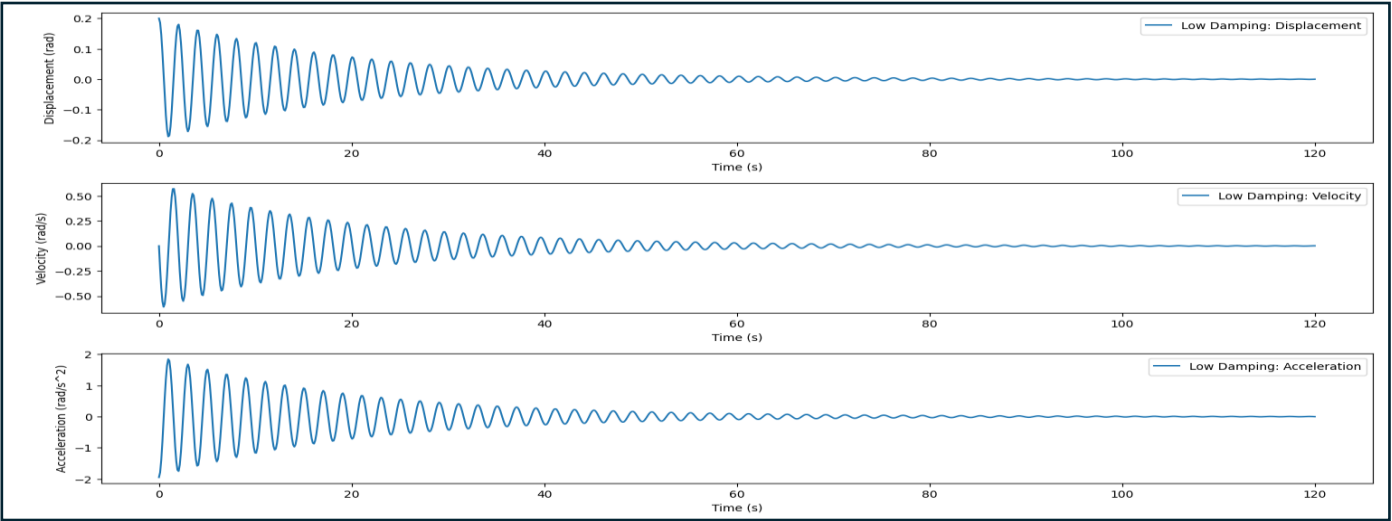
5.3 Phase Plots and Amplitude Response

Phase Plots: Visualize the state of the oscillator by plotting velocity versus displacement.

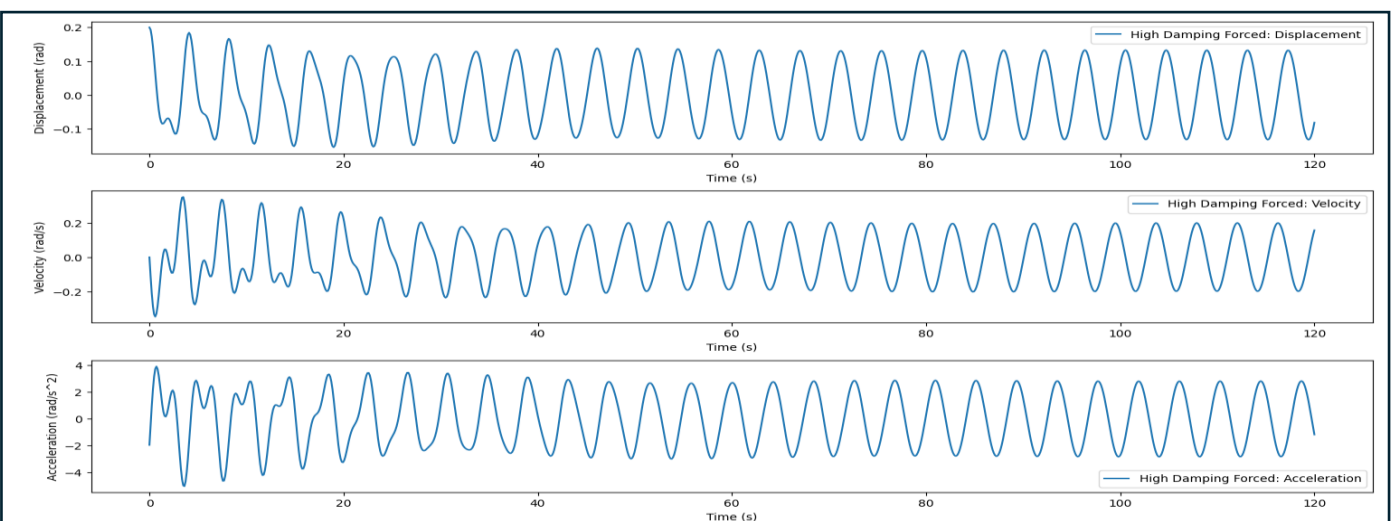
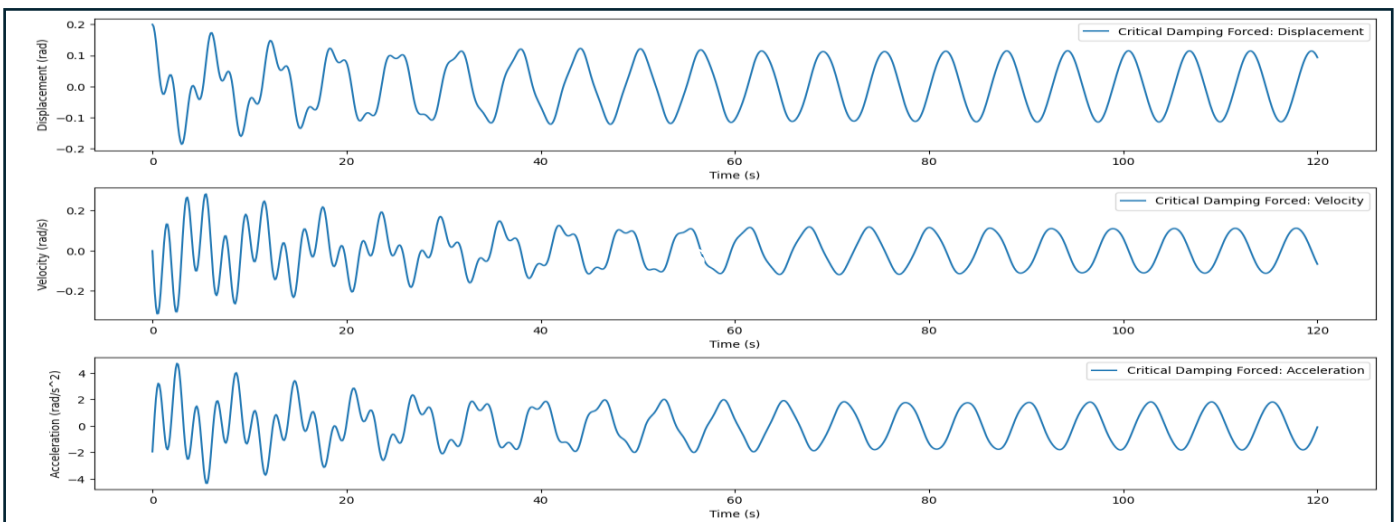
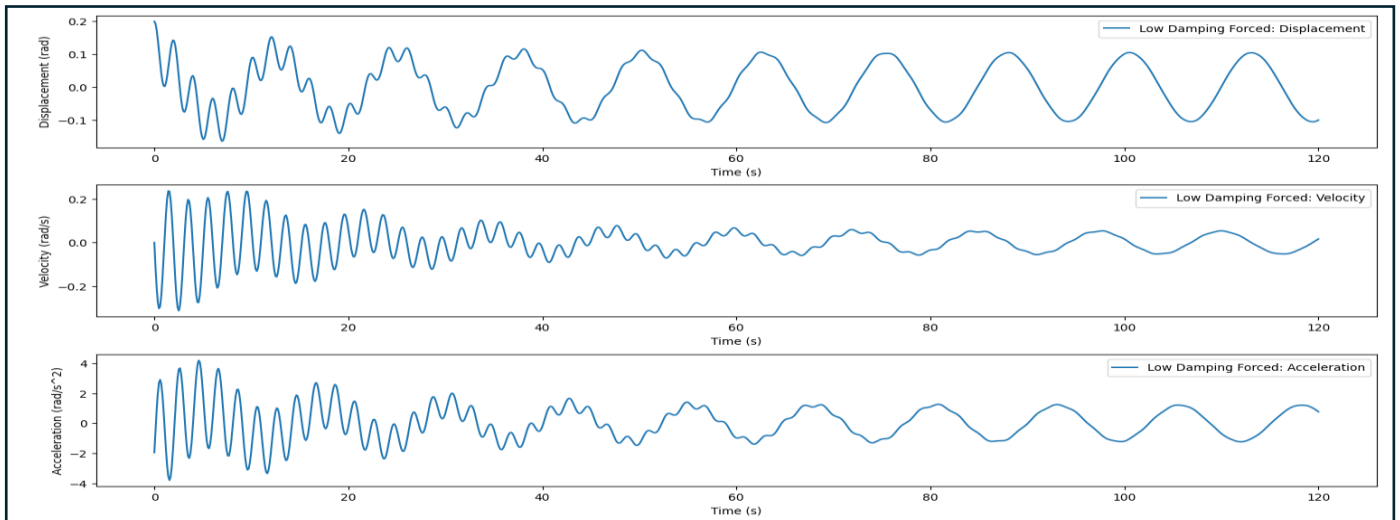
Simple Harmonic Oscillator



Damped Harmonic Oscillator



Forced Harmonic Oscillator



6. Numerical Modeling and Simulations

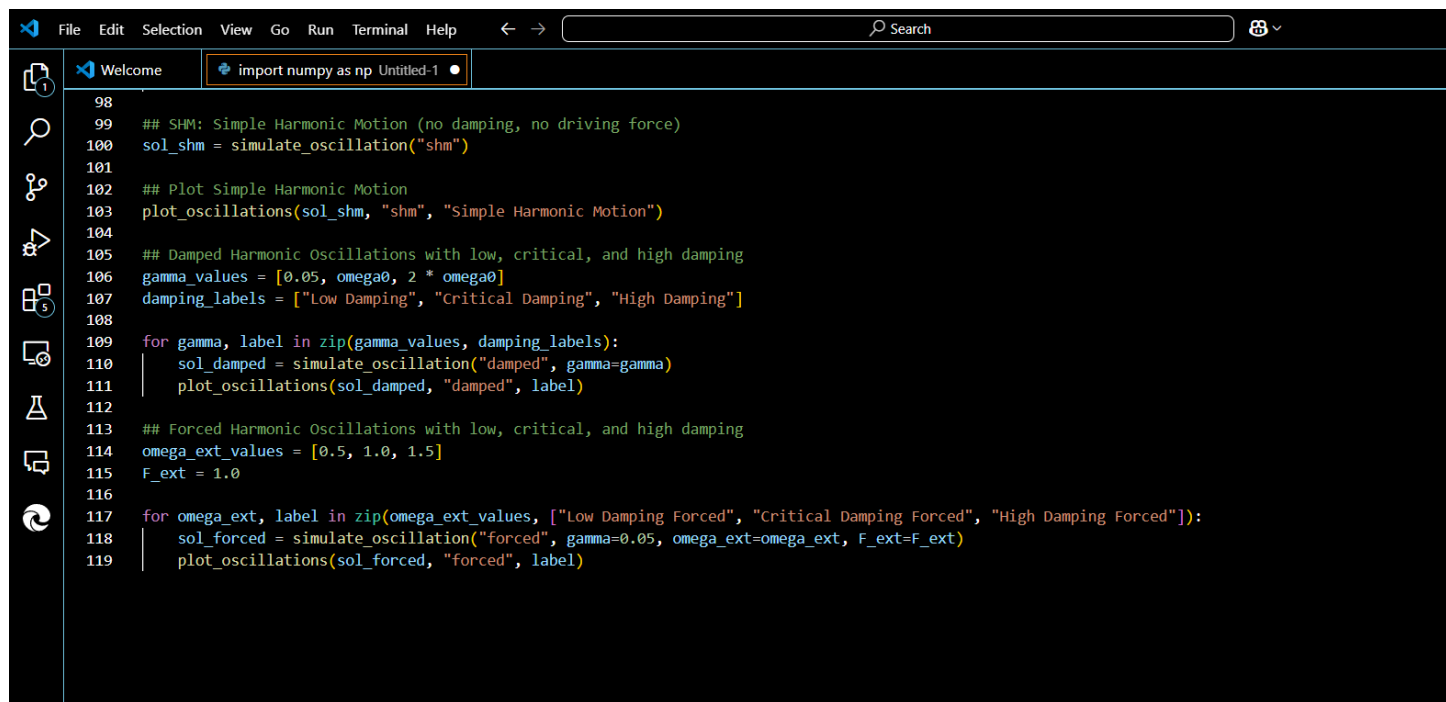
```
File Edit Selection View Go Run Terminal Help ← → Search

Welcome import numpy as np Untitled-1

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.integrate import odeint
4 from scipy.signal import find_peaks
5
6 ## Constants
7 g = 9.81 ## Acceleration due to gravity in m/s^2
8 L = 1.0 ## Length of the pendulum in meters
9 theta0 = 0.2 ## Initial angle in radians (small angle approximation)
10 omega0 = np.sqrt(g / L) ## Natural frequency of the pendulum (SHM)
11 time = np.linspace(0, 120, 1000) ## Time array for plotting
12
13 ## Function to compute the equations of motion for SHM and damped/forced oscillations
14 def equation_of_motion(y, t, gamma, omega_ext, F_ext, damping_type):
15     theta, omega = y
16     if damping_type == "damped":
17         ## Damped harmonic oscillator (under small damping conditions)
18         dydt = [omega, -2*gamma*omega - omega0**2 * np.sin(theta)]
19     elif damping_type == "forced":
20         ## Forced harmonic oscillator
21         dydt = [omega, -2*gamma*omega - omega0**2 * np.sin(theta) + F_ext * np.cos(omega_ext * t)]
22     else:
23         ## Simple Harmonic Oscillator (SHM)
24         dydt = [omega, -omega0**2 * np.sin(theta)]
25     return dydt
26
27 ## Function to simulate SHM, damped, and forced oscillations
28 def simulate_oscillation(damping_type, gamma=0, omega_ext=0, F_ext=0):
29     ## Initial conditions: [initial angle, initial angular velocity]
30     y0 = [theta0, 0.0]
31     ## Solve ODE for the given damping type
32     sol = odeint(equation_of_motion, y0, time, args=(gamma, omega_ext, F_ext, damping_type))
33     return sol
```

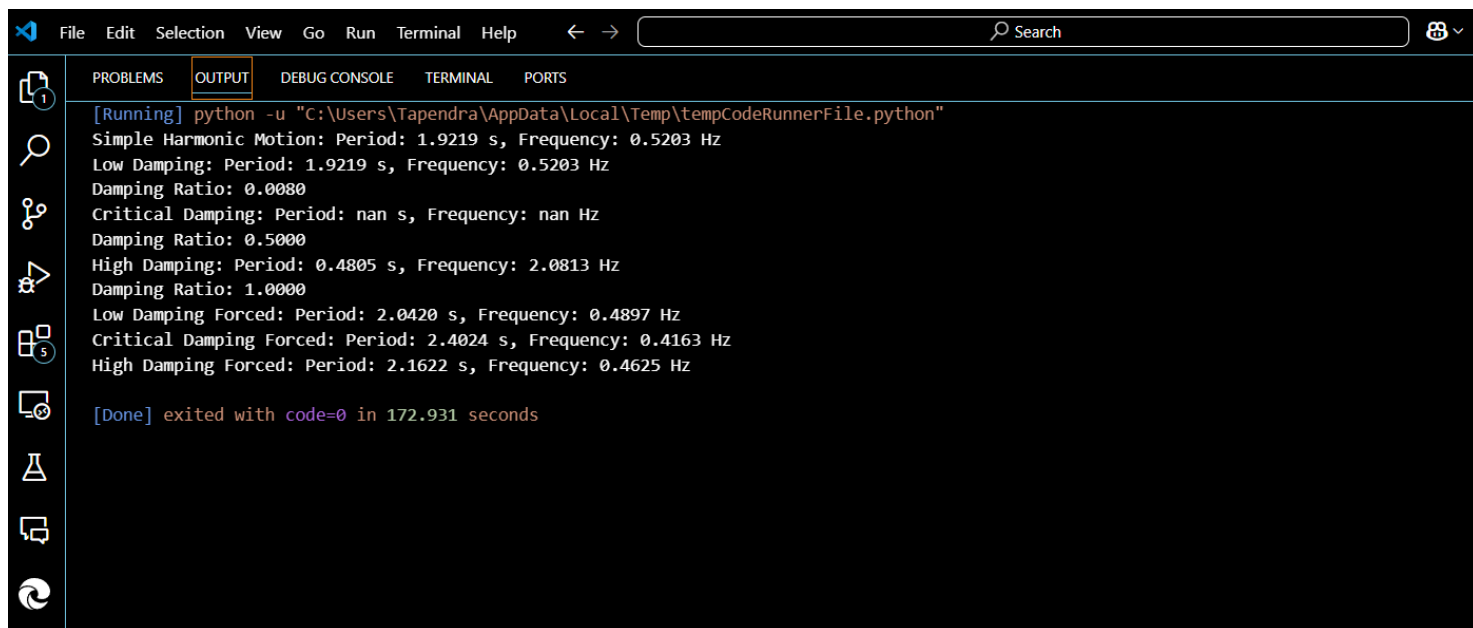
```
File Edit Selection View Go Run Terminal Help ← → Search
Welcome import numpy as np Untitled-1
34
35 ## Function to calculate period, frequency, and damping ratio
36 def calculate_parameters(time, theta, damping_type, gamma=None):
37     ## Find the peaks in the displacement (for oscillation period)
38     peaks, _ = find_peaks(theta)
39
40     if len(peaks) > 1:
41         ## Calculate the time period based on peak-to-peak distance
42         period = time[peaks[1]] - time[peaks[0]]
43         frequency = 1 / period
44     else:
45         ## If not enough peaks, return NaN (or we can return a fixed value like zero for no oscillations)
46         period = np.nan
47         frequency = np.nan
48
49     if damping_type == "damped" and gamma is not None:
50         ## Calculate the damping ratio for damped oscillations (using gamma)
51         damping_ratio = gamma / (2 * np.sqrt(g * L))
52     else:
53         damping_ratio = np.nan
54
55     return period, frequency, damping_ratio
56
57 ## Plotting function for displacement, velocity, and acceleration
58 def plot_oscillations(sol, damping_type, damping_label=""):
59     ## Extract solutions
60     theta = sol[:, 0]
61     omega = sol[:, 1]
62     acceleration = -omega**2 * np.sin(theta) if damping_type == "shm" else -omega**2 * np.sin(theta) - 2 * gamma * omega
63
64     ## Calculate parameters (period, frequency, damping_ratio)
65     period, frequency, damping_ratio = calculate_parameters(time, theta, damping_type, gamma if damping_type == "damped" else None)
66
```

```
File Edit Selection View Go Run Terminal Help ← → Search
Welcome import numpy as np Untitled-1
58 def plot_oscillations(sol, damping_type, damping_label=""):
66
67     ## Display calculated parameters
68     print(f"{damping_label}: Period: {period:.4f} s, Frequency: {frequency:.4f} Hz")
69     if not np.isnan(damping_ratio):
70         print(f"Damping Ratio: {damping_ratio:.4f}")
71
72     ## Plotting the results
73     plt.figure(figsize=(12, 8))
74
75     ## Displacement vs Time
76     plt.subplot(3, 1, 1)
77     plt.plot(time, theta, label=f"{damping_label}: Displacement")
78     plt.xlabel("Time (s)")
79     plt.ylabel("Displacement (rad)")
80     plt.legend(loc="best")
81
82     ## Velocity vs Time
83     plt.subplot(3, 1, 2)
84     plt.plot(time, omega, label=f"{damping_label}: Velocity")
85     plt.xlabel("Time (s)")
86     plt.ylabel("Velocity (rad/s)")
87     plt.legend(loc="best")
88
89     ## Acceleration vs Time
90     plt.subplot(3, 1, 3)
91     plt.plot(time, acceleration, label=f"{damping_label}: Acceleration")
92     plt.xlabel("Time (s)")
93     plt.ylabel("Acceleration (rad/s^2)")
94     plt.legend(loc="best")
95
96     plt.tight_layout()
97     plt.show()
```



The image shows a Visual Studio Code editor window with a dark theme. The menu bar at the top includes File, Edit, Selection, View, Go, Run, Terminal, and Help. A search bar is located on the right. The file explorer on the left shows a file named 'import numpy as np Untitled-1'. The main editor area contains Python code for simulating harmonic motion. The code includes comments and function calls for Simple Harmonic Motion (SHM), Damped Harmonic Oscillations, and Forced Harmonic Oscillations. The code is as follows:

```
98
99  ## SHM: Simple Harmonic Motion (no damping, no driving force)
100 sol_shm = simulate_oscillation("shm")
101
102  ## Plot Simple Harmonic Motion
103 plot_oscillations(sol_shm, "shm", "Simple Harmonic Motion")
104
105  ## Damped Harmonic Oscillations with low, critical, and high damping
106 gamma_values = [0.05, omega0, 2 * omega0]
107 damping_labels = ["Low Damping", "Critical Damping", "High Damping"]
108
109  for gamma, label in zip(gamma_values, damping_labels):
110      sol_damped = simulate_oscillation("damped", gamma=gamma)
111      plot_oscillations(sol_damped, "damped", label)
112
113  ## Forced Harmonic Oscillations with low, critical, and high damping
114  omega_ext_values = [0.5, 1.0, 1.5]
115  F_ext = 1.0
116
117  for omega_ext, label in zip(omega_ext_values, ["Low Damping Forced", "Critical Damping Forced", "High Damping Forced"]):
118      sol_forced = simulate_oscillation("forced", gamma=0.05, omega_ext=omega_ext, F_ext=F_ext)
119      plot_oscillations(sol_forced, "forced", label)
```



The image shows the same Visual Studio Code editor window, but with the 'OUTPUT' tab selected. The output shows the results of the simulations. The code is as follows:

```
[Running] python -u "C:\Users\Tapendra\AppData\Local\Temp\tempCodeRunnerFile.python"
Simple Harmonic Motion: Period: 1.9219 s, Frequency: 0.5203 Hz
Low Damping: Period: 1.9219 s, Frequency: 0.5203 Hz
Damping Ratio: 0.0080
Critical Damping: Period: nan s, Frequency: nan Hz
Damping Ratio: 0.5000
High Damping: Period: 0.4805 s, Frequency: 2.0813 Hz
Damping Ratio: 1.0000
Low Damping Forced: Period: 2.0420 s, Frequency: 0.4897 Hz
Critical Damping Forced: Period: 2.4024 s, Frequency: 0.4163 Hz
High Damping Forced: Period: 2.1622 s, Frequency: 0.4625 Hz

[Done] exited with code=0 in 172.931 seconds
```

7. Discussion and Analysis

The behavior of harmonic oscillators—whether simple, damped, or forced—encapsulates a wide array of physical systems ranging from mechanical vibrations to electrical circuits. The present study, through theoretical formulations and numerical simulations, explores these models in detail, offering insight into how different factors influence system dynamics.

7.1 Simple Harmonic Oscillator

In an undamped system, the motion is governed solely by the restoring force proportional to displacement, resulting in **simple harmonic motion (SHM)**. The solution to the differential equation $\frac{d^2x}{dt^2} + \omega_0^2x = 0$ yields sinusoidal functions, indicating perpetual oscillation at the natural frequency ω_0 . Key observations include:

- Constant amplitude and energy (ideal system).
- Motion is periodic and predictable.
- Amplitude is independent of mass or spring stiffness but entirely determined by initial conditions.

This idealized model serves as the foundation for understanding more complex oscillatory behavior.

7.2 Damped Harmonic Oscillator

Real-world systems experience energy loss due to resistive forces such as friction or air drag. The introduction of a damping term leads to the differential equation:

$$\frac{d^2x}{dt^2} + 2\beta \frac{dx}{dt} + \omega_0^2x = 0$$

Depending on the damping coefficient β , the system exhibits:

- **Underdamped motion:** Oscillations persist but decay exponentially.
- **Critically damped motion:** Fastest return to equilibrium without oscillation.
- **Overdamped motion:** Slow, non-oscillatory return to equilibrium.

From simulations, underdamped systems showed frequency reduction and amplitude decay. Critically and overdamped systems demonstrated how damping can stabilize a system rapidly but may delay responsiveness in overdamping.

7.3. Forced Harmonic Oscillator

When an external periodic force is applied, the system responds with a combination of natural and driven frequencies:

$$\frac{d^2x}{dt^2} + 2\beta \frac{dx}{dt} + \omega_0^2 x = F_0 \cos(\omega t)$$

The most significant outcome is **resonance**, which occurs when the driving frequency ω approaches the system's natural frequency ω_0 . This results in:

- Amplitude amplification (especially in low damping).
- Phase difference between driving force and displacement.
- Shift in resonance peak due to damping.

Simulation results clearly demonstrated resonance behavior, confirming theoretical predictions. With low damping, the amplitude response curve exhibited a sharp resonance peak, while increasing damping led to broader, flattened resonance curves.

Conclusion of Analysis

This analysis validates the classical models of oscillatory systems and demonstrates their practical significance. Damping and external forces have profound effects on system stability, response, and performance. These results have implications for designing real-world systems—from vehicle suspensions to electronic filters—where oscillatory behavior must be precisely controlled or exploited.

8. Conclusion

This project explored the dynamics of simple, damped, and forced harmonic oscillators through theoretical analysis and numerical modeling. The simple harmonic oscillator demonstrated ideal periodic motion with constant amplitude and energy. Introducing damping revealed how resistive forces dissipate energy, leading to amplitude decay and varied system responses based on damping strength. In the forced oscillator, resonance emerged as a critical phenomenon where driving frequency matched the system's natural frequency, significantly amplifying oscillations. Overall, this study highlights how oscillatory behavior is influenced by system parameters and external inputs, offering valuable insights into real-world applications where control of oscillations is essential.