Computational Physics with Python

Objective:

To explore laser physics concepts using Python simulations:

- Visualize **temporal pulse shapes**, **diffraction**, **interference**, and **pulse propagation**
- Gain hands-on practice with **basic coding and plotting**
- Use animations to understand **dynamic laser behaviour**

Activity 1 : Temporal Profile of a Laser Pulse

Simple Physics Problem

A pulsed laser has intensity that varies with time, often approximated by a Gaussian:

$$I(t) = I_0 \exp\left(-\frac{t^2}{\tau^2}\right)$$

where

$I(t)$ = intensity of laser pulse at time t
I_0 = peak (maximum) intensity at the center of the pulse
τ = pulse width
t = time (in second)

Python code. Save the program as exercise1.py

```python
import numpy as np
import matplotlib.pyplot as plt

I0 = 1.0
tau = 10e-12  # pulse width (10 ps)
t = np.linspace(-50e-12, 50e-12, 1000)

I = I0 * np.exp(-t**2 / tau**2)

plt.plot(t * 1e12, I)
plt.title("Gaussian Laser Pulse")
plt.xlabel("Time (ps)")
plt.ylabel("Intensity (a.u.)")
plt.grid(True)
plt.show()
```

Questions

1. What is the intensity at $t = 0$?
2. What happens when τ is changed to 20e12 ?
3. Estimate the **FWHM** of the pulse.
4. What happens if I_0 is doubled?

Activity 2 : Single-Slit Diffraction

Physics Background

In single-slit diffraction, the intensity distribution on a screen at distance L from the slit is given by:

$$I(\theta) = I_0 \left(\frac{sin(\beta)}{\beta}\right)^2$$

where

$$\beta = \frac{\pi a}{\lambda} \sin\theta$$

a = slit width
λ = wavelength of the laser light
θ = diffraction angle
I_0 = central maximum intensity

Python code. Save the program as exercise2.py

```python
import numpy as np
import matplotlib.pyplot as plt

wavelength = 500e-9   # 500 nm
a = 20e-6             # slit width (20 µm)
theta = np.linspace(-0.01, 0.01, 1000)
beta = (np.pi * a / wavelength) * np.sin(theta)

I = (np.sin(beta) / beta)**2
I[beta == 0] = 1.0

plt.plot(np.degrees(theta), I)
plt.title("Single-Slit Diffraction")
plt.xlabel("Angle (degrees)")
plt.ylabel("Intensity (a.u.)")
plt.grid(True)
plt.show()
```

Questions

1. How many bright fringes are visible?
2. What happens when you change a to 40e-6?
3. What is the angle of the first minimum

Activity 3 : Analysing Single-Slit Diffraction Patterns Using Python

Objective:

1. Use experimental data from a single-slit diffraction experiment
2. Apply the theoretical diffraction intensity equation
3. Plot experimental and theoretical intensity patterns
4. Determine the slit width by fitting the theory to the data

Using the same problem as in Activity 2, we shall compared the experimental data and apply the diffraction intensity equation into the data. The experiment data is given in the diffraction_data.csv where it contains:

x_mm: position on the screen (in mm)

intensity: measured intensity (arbitrary units)

Steps for Activity 3:

Python code. Save the program as exercise3.py

1. Load and Visualize Data

exercise3.py > ...

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Load data
data = pd.read_csv("diffraction_data.csv")
x_mm = data["x_mm"].values
intensity = data["intensity"].values

# Plot
plt.plot(x_mm, intensity, 'o', label='Experimental Data')
plt.xlabel("Position x (mm)")
plt.ylabel("Intensity (a.u.)")
plt.title("Diffraction Pattern")
plt.legend()
plt.grid(True)
plt.show()

```

2. Define Theoretical Diffraction Function

```python
def theoretical_intensity(x, a, L, wavelength):
    beta = (np.pi * a * x / L) / wavelength
    # Avoid divide by zero at center
    beta = np.where(beta == 0, 1e-10, beta)
    return (np.sin(beta)/beta)**2

```

3. Fit Theory to Data and Find Slit Width (see code next page)

```python
from scipy.optimize import curve_fit
# Convert to meters
x_m = x_mm * 1e-3
L = 1.0      # screen distance in meters
wavelength = 650e-9  # red laser in meters

# Define model for curve fitting
def model(x, a):
    return theoretical_intensity(x, a, L, wavelength)

# Fit
popt, _ = curve_fit(model, x_m, intensity, p0=[1e-4])  # initial guess for a
a_fit = popt[0]

# Plot fit
x_fit = np.linspace(min(x_m), max(x_m), 500)
y_fit = model(x_fit, a_fit)

plt.plot(x_mm, intensity, 'o', label='Experimental Data')
plt.plot(x_fit*1e3, y_fit, '-', label=f'Theory Fit (a = {a_fit*1e6:.1f} µm)')
plt.xlabel("Position x (mm)")
plt.ylabel("Intensity")
plt.legend()
plt.grid(True)
plt.title("Fit of Diffraction Pattern")
plt.show()
```

Activity 4 : Double-Slit Interference

Physics background

Light through two slits interferes:

$$I(\theta) = I_0\, cos^2\left(\frac{\pi d}{\lambda}\sin\theta\right)$$

$I(\theta)$ = intensity at angle θ
d = distance between two slits
λ = wavelength of the laser light
θ = diffraction angle

Python code. Save the program as exercise4.py

exercise4.py > ...

```python
import numpy as np
import matplotlib.pyplot as plt

wavelength = 500e-9
d = 100e-6
theta = np.linspace(-0.02, 0.02, 1000)

delta = (np.pi * d / wavelength) * np.sin(theta)
I = np.cos(delta)**2

plt.plot(np.degrees(theta), I)
plt.title("Double-Slit Interference")
plt.xlabel("Angle (degrees)")
plt.ylabel("Intensity (a.u.)")
plt.grid(True)
plt.show()
```

Questions:

1. Count the number of fringes in ±1°.
2. How does d = 50e-6 change fringe spacing?
3. What happens if wavelength = 700e-9?

Activity 5 : Animation – Propagation of a Laser Pulse

Physics background

A laser pulse travels through space like a moving Gaussian wave:

$$I(x,t) = I_0 \exp\left(-\frac{(x-vt)^2}{\tau^2}\right)$$

$I(x,t)$ = intensity at position x and time t

v = velocity of the pulse in moving space
x = spatial position
t = time
τ = d

Python code. Save the program as exercise5.py

exercise5.py > ...

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation

# Parameters
I0 = 1.0
tau = 1.0       # pulse width
v = 1.0         # speed of pulse
x = np.linspace(-10, 30, 500)

fig, ax = plt.subplots()
line, = ax.plot([], [], lw=2)
ax.set_xlim(-10, 30)
ax.set_ylim(0, 1.2)
ax.set_title("Laser Pulse Propagation")
ax.set_xlabel("x (position)")
ax.set_ylabel("Intensity")

def init():
    line.set_data([], [])
    return line,

def animate(t):
    I = I0 * np.exp(-((x - v * t)**2) / tau**2)
    line.set_data(x, I)
    return line,

frames = np.linspace(0, 20, 200)
ani = animation.FuncAnimation(fig, animate, frames=frames, init_func=init,
                              blit=True, interval=50)

plt.show()
```

Questions

1. How does changing τ affect the pulse width?
2. Try v = 2.0. What changes?
3. Modify the code to show two pulses moving toward each other. What happens?

End of page