Trigonometry with Sine and Cosine Functions: Series 1

Understanding Transformations and Plotting in Python

Table of contents

1 Exercises

In this series of exercises, we will apply our knowledge of trigonometry and Python programming to work with sine and cosine functions, exploring their transformations using amplitude, frequency, and phase shift.

We will use Python libraries such as **NumPy** for numerical computations and **Matplotlib** for plotting graphs.

Let's get started!

1.1 Exercise 1: Defining a Sine Function with Transformations

Question

Create a Python function named `my_sin` that calculates the sine of a value `x` with a given `amplitude`, `frequency`, and `phase_shift`. Use the `NumPy` library to perform the calculations.

- Define the function `my_sin(x, amplitude, frequency, phase_shift)`.
- The function should return `y = amplitude * np.sin(frequency * x + phase_shift)`.

> *Example of defining a Python function:*
>
> ```
> def function_name(parameters):
>     # Function body
>     return result
> ```

```
# Define your my_sin function here
```

1.2 Exercise 2: Plotting a Transformed Sine Wave

Question

Using the `my_sin` function you just created, plot a sine wave with the following parameters:

- **Amplitude**: 1.5
- **Frequency**: 2
- **Phase Shift**: $\frac{\pi}{3}$

Instructions:

- Import necessary libraries (`numpy` and `matplotlib`).
- Create an array `x` from 0 to 2π with 1001 points.
- Calculate `y` using your `my_sin` function.
- Plot the resulting wave using `Matplotlib`.

> *Hint:* Use `np.linspace(0, 2 * np.pi, 1001)` to create the `x` array.
> *Example of plotting in Python:*
>
> ```
> import matplotlib.pyplot as plt
>
> plt.plot(x, y)
> plt.show()
> ```

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt

# Create x array
# x = ?

# Use your my_sin function to calculate y
# y = ?

# Plot the wave
# plt.plot(x, y)
# plt.show()
```

1.3 Exercise 3: Observing the Effect of Changing Parameters

Question

Experiment with changing the `amplitude`, `frequency`, and `phase_shift` values in your `my_sin` function. For each parameter, keep the other two constant and observe how the graph changes.

Instructions:

- Import necessary libraries (`numpy` and `matplotlib`).
- Create an array `x` from 0 to 2π with 1001 points.
- Set up three separate plots:
  1. **Amplitude Variation**:
     - Use amplitude values of 0.5, 1, 2.
     - Keep frequency = 1, phase_shift = 0.
  2. **Frequency Variation**:
     - Use frequency values of 0.5, 1, 2.
     - Keep amplitude = 1, phase_shift = 0.
  3. **Phase Shift Variation**:
     - Use phase_shift values of 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$.
     - Keep amplitude = 1, frequency = 1.
- For each plot, use a `for` loop to iterate over the varying parameter values.
- Plot each set of waves on the same graph for comparison.

Example of using a `for` loop in Python:

```
values = [value1, value2, value3]
for value in values:
    # Code to execute for each value
    result = some_function(value)
    # Print the result to the console
    print(result)
```

In this example, the `for` loop goes through each `value` in the `values` list and executes the code block indented under the `for` statement.

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt

# Create x array
x = np.linspace(0.0, 2 * np.pi, 1001)

# Amplitude Variation
amplitudes = [0.5, 1, 2]
for amplitude in amplitudes:
    # Calculate y using my_sin function
    # y = ?
    # Plot the wave
    # plt.plot(x, y, label=f'Amplitude = {amplitude}')
    pass  # Replace with your code

# Add title, labels, and legend
# plt.title('Effect of Changing Amplitude')
# plt.xlabel('x')
# plt.ylabel('y')
# plt.legend()
# plt.show()

# Repeat the above steps for Frequency Variation and Phase Shift Variation
```

1.4 Exercise 4: Combining Waves

Question

Create two waves and sum them together:

- **Wave A**: Sine wave with amplitude 1, frequency 1, phase shift 0.
- **Wave B**: Sine wave with amplitude 0.5, frequency 5, phase shift 0.

Instructions:

- Use your `my_sin` function to calculate `wave_a` and `wave_b`.
- Sum the two waves to get `wave_combined = wave_a + wave_b`.
- Plot all three waves (`wave_a`, `wave_b`, and `wave_combined`) on the same graph.
- Label each wave appropriately.

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt

# Create x array
x = np.linspace(0.0, 2 * np.pi, 1001)

# Calculate wave_a using my_sin function
# wave_a = ?

# Calculate wave_b using my_sin function
# wave_b = ?

# Sum the two waves
# wave_combined = ?

# Plot the waves
# plt.plot(x, wave_a, label='Wave A')
# plt.plot(x, wave_b, label='Wave B')
# plt.plot(x, wave_combined, linestyle='--', label='Combined Wave')
# plt.title('Combining Two Sine Waves')
# plt.xlabel('x')
# plt.ylabel('y')
# plt.legend()
# plt.show()
```

1.5 Exercise 5: Exploring the Effect of Phase Shift in Wave Combination

Question

Adjust the `phase_shift` of **Wave B** in Exercise 4 to $\frac{\pi}{2}$ and observe how the combined wave changes.

Instructions:

- Recalculate `wave_b` with `phase_shift = np.pi / 2`.
- Sum the new `wave_b` with `wave_a` to get `wave_combined_shifted = wave_a + wave_b`.
- Plot the new combined wave and compare it to the previous one.
- Discuss how the phase shift affects the resulting wave.

```
# Recalculate wave_b with phase shift of pi/2
# wave_b = ?

# Sum the new waves
# wave_combined_shifted = ?

# Plot the new combined wave
# plt.plot(x, wave_a, label='Wave A')
# plt.plot(x, wave_b, label='Wave B with Phase Shift $\\pi/2$')
# plt.plot(x, wave_combined_shifted,
#          linestyle='--', label='Combined Sine Wave with Phase Shift')
# plt.title('Effect of Phase Shift on Combined Sine Wave')
# plt.xlabel('x')
# plt.ylabel('y')
# plt.legend()
# plt.show()
```

1.6 Exercise 6: Researching Online References

Question

Find online resources that explain the `NumPy` and `Matplotlib` libraries and how they are used for mathematical computations and plotting in Python.

Instructions:

- Search for tutorials on `NumPy` and `Matplotlib`.
- List at least two resources for each library.
- Briefly describe what you can learn from each resource.

Write your answers below:

NumPy Resources:

1. **Resource Name**
    - *Link*:
    - *Description*:
2. **Resource Name**

- *Link*:
- *Description*:

Matplotlib Resources:

1. **Resource Name**
   - *Link*:
   - *Description*:
2. **Resource Name**
   - *Link*:
   - *Description*:

---

2 Conclusion

In this exercise series, we've applied our understanding of sine and cosine functions and their transformations in Python. By working through these problems, you've practiced defining functions, using `for` loops, plotting graphs, and exploring the effects of changing parameters on trigonometric functions.

Keep experimenting with different values and functions to deepen your understanding of trigonometry and programming in Python.

---

3 Additional Resources

- **Understanding Sine and Cosine Functions**: Khan Academy
- **Python Programming Basics**: Python Official Documentation
- **NumPy Library Guide**: NumPy User Guide
- **Matplotlib Plotting Guide**: Matplotlib Tutorials

---

4 Acknowledgments

We hope this series has helped you understand how to apply trigonometric concepts using Python. Keep practicing, and don't hesitate to explore more about programming and mathematics!