Generating and Plotting Sinusoidal Signals: Code and Explanation

Grok

May 13, 2025

1 Introduction

This document presents a Python script that generates sinusoidal signals using NumPy and visualizes them with Plotly. The script is explained line by line, detailing the purpose, functionality, and context of each component. The code and explanations are provided to clarify how the script generates and plots three sinusoidal waves with different frequencies and amplitudes.

2 Python Code

The complete Python script is shown below. It defines functions to generate sinusoidal signals and plot them interactively.

```
import plotly.graph_objects as go
import numpy as np

def generate_sinusoid(frequency, amplitude, duration,
    sample_rate=1000):
    """Generate a sinusoidal signal."""
    t = np.linspace(0, duration, int(sample_rate * duration),
        endpoint=False)
    signal = amplitude * np.sin(2 * np.pi * frequency * t)
    return t, signal

def plot_time_series(series_list, labels, title="Time Series"):
    """Plot multiple time series using Plotly."""
    fig = go.Figure()
    colors = ['blue', 'red', 'green']
    for i, (time, signal) in enumerate(series_list):
        fig.add_trace(
            go.Scatter(
                x=time,
                y=signal,
                name=labels[i],
                line=dict(color=colors[i % len(colors)]),
                mode='lines'
            )
        )
    fig.update_layout(title=title, xaxis_title="Time (s)",
        yaxis_title="Amplitude", showlegend=True)
    fig.show()

if __name__ == "__main__":
    # Generate three sinusoids
```

```
    series = [
        generate_sinusoid(frequency=2, amplitude=1, duration=2),
        generate_sinusoid(frequency=5, amplitude=0.5, duration=2),
        generate_sinusoid(frequency=3, amplitude=0.8, duration=2)
    ]
    labels = ["2 Hz, Amp 1", "5 Hz, Amp 0.5", "3 Hz, Amp 0.8"]

    # Plot
    plot_time_series(series, labels, title="Sinusoidal Waves")
```

Listing 1: Python script for generating and plotting sinusoidal signals

3 Detailed Explanation

Below is a line-by-line explanation of the code, organized by section for clarity. Each lines purpose, mechanics, and context are described.

3.1 Imports

- `import plotly.graph_objects as go`
  - **Purpose**: Imports the `graph_objects` module from Plotly, aliased as `go`.
  - **Details**: Plotly is a library for interactive visualizations. The `graph_objects` module provides a low-level interface for constructing plots, allowing control over traces (e.g., lines, scatter points) and layouts. The `go` alias simplifies referencing.
  - **Example Use**: `go.Scatter` creates line plots.
- `import numpy as np`
  - **Purpose**: Imports NumPy, aliased as `np`.
  - **Details**: NumPy is a library for numerical computations, ideal for arrays and mathematical functions. Its used for generating time points and sinusoids. The `np` alias is standard.
  - **Example Use**: `np.linspace` and `np.sin` are used.

3.2 Function: `generate_sinusoid`

- `def generate_sinusoid(frequency, amplitude, duration, sample_rate=1000):`
  - **Purpose**: Defines a function to generate a sinusoidal signal.
  - **Parameters**:
    - `frequency`: Frequency in Hertz (Hz), controlling cycles per second.
    - `amplitude`: Peak amplitude, setting signal height.
    - `duration`: Signal duration in seconds.
    - `sample_rate`: Samples per second (default 1000 Hz).
  - **Details**: Creates time and signal arrays for a sinusoid.
- `"""Generate a sinusoidal signal."""`
  - **Purpose**: Docstring describing the function.
  - **Details**: Concise documentation of the functions role.
- `t = np.linspace(0, duration, int(sample_rate * duration), endpoint=False)`

- **Purpose**: Creates a time array `t`.
- **Details**: `np.linspace` generates evenly spaced points from 0 to `duration`, excluding the endpoint. The number of points is `sample_rate * duration`. Example: For `duration=2`, `sample_rate=1000`, it creates 2000 points from 0 to 1.999.

- `signal = amplitude * np.sin(2 * np.pi * frequency * t)`
  - **Purpose**: Computes the sinusoidal signal.
  - **Details**: Uses the formula $A\sin(2\pi ft)$. `np.sin` computes sine for phase values, scaled by `amplitude`. Example: For `amplitude=1`, `frequency=2`, the signal oscillates between -1 and 1, 2 cycles/second.

- `return t, signal`
  - **Purpose**: Returns time and signal arrays.
  - **Details**: Returns a tuple for plotting or processing.

3.3 Function: `plot_time_series`

- `def plot_time_series(series_list, labels, title="Time Series"):`
  - **Purpose**: Plots multiple time series with Plotly.
  - **Parameters**:
    - `series_list`: List of `(time, signal)` tuples.
    - `labels`: List of legend labels.
    - `title`: Plot title (default Time Series).

- `"""Plot multiple time series using Plotly."""`
  - **Purpose**: Docstring for the function.

- `fig = go.Figure()`
  - **Purpose**: Creates a new Plotly figure.
  - **Details**: Initializes an empty figure for traces and layout.

- `colors = ['blue', 'red', 'green']`
  - **Purpose**: Defines colors for traces.
  - **Details**: Colors cycle if more series than colors.

- `for i, (time, signal) in enumerate(series_list):`
  - **Purpose**: Iterates over `series_list`, unpacking tuples.
  - **Details**: `enumerate` provides index and tuple for adding traces.

- `fig.add_trace(go.Scatter(x=time, y=signal, name=labels[i], line=dict(color=colors[i % len(colors)]), mode='lines'))`
  - **Purpose**: Adds a line plot trace.
  - **Details**: `go.Scatter` creates a line plot with `x=time`, `y=signal`, legend label `labels[i]`, and cycled color. `mode='lines'` ensures a continuous line.

- `fig.update_layout(title=title, xaxis_title="Time (s)", yaxis_title="Amplitude", showlegend=True)`

- **Purpose**: Configures figure layout.
- **Details**: Sets title, axis labels, and enables legend.

- `fig.show()`
  - **Purpose**: Displays the interactive figure.
  - **Details**: Renders the plot in a browser or notebook.

3.4 Main Block

- `if __name__ == "__main__":`
  - **Purpose**: Runs code if script is executed directly.
  - **Details**: Allows importing without running example code.
- `series = [generate_sinusoid(frequency=2, amplitude=1, duration=2), ...]`
  - **Purpose**: Generates three sinusoids.
  - **Details**: Creates signals: 2 Hz (amplitude 1), 5 Hz (amplitude 0.5), 3 Hz (amplitude 0.8), each 2 seconds, 2000 samples.
- `labels = ["2 Hz, Amp 1", "5 Hz, Amp 0.5", "3 Hz, Amp 0.8"]`
  - **Purpose**: Defines legend labels.
  - **Details**: Matches `series` order.
- `plot_time_series(series, labels, title="Sinusoidal Waves")`
  - **Purpose**: Plots the sinusoids.
  - **Details**: Produces an interactive plot with three waves.

4 Summary

The script defines:

- `generate_sinusoid`: Creates a sinusoidal signal with given parameters.
- `plot_time_series`: Plots multiple signals interactively.

The main block generates three sinusoids (2 Hz, 5 Hz, 3 Hz; amplitudes 1, 0.5, 0.8) and plots them in a figure with blue, red, and green lines, labeled axes, legend, and title Sinusoidal Waves. The output is an interactive Plotly figure.