Python Visualization Libraries Guide: Matplotlib and Seaborn

Table of Contents

1. Introduction
2. Matplotlib
   - Overview
   - Graph Types
3. Seaborn
   - Overview
   - Graph Types
4. Library Comparison

Introduction

- This guide provides a comprehensive overview of two popular Python visualization libraries: **Matplotlib** and **Seaborn**.
- We'll explore their features, capabilities, and provide practical examples for various graph types.
- To install these libraries:
  1. pip install matplotlib
  2. pip install seaborn

Matplotlib

Matplotlib Overview

Matplotlib is the foundational library for data visualization in Python. It provides a MATLAB-like interface and offers extensive control over every aspect of plots. Key features include:

- Low-level control over plot elements
- Publication-quality figures
- Wide range of plot types
- Extensive customization options
- Strong integration with NumPy and Pandas

Matplotlib Graph Types

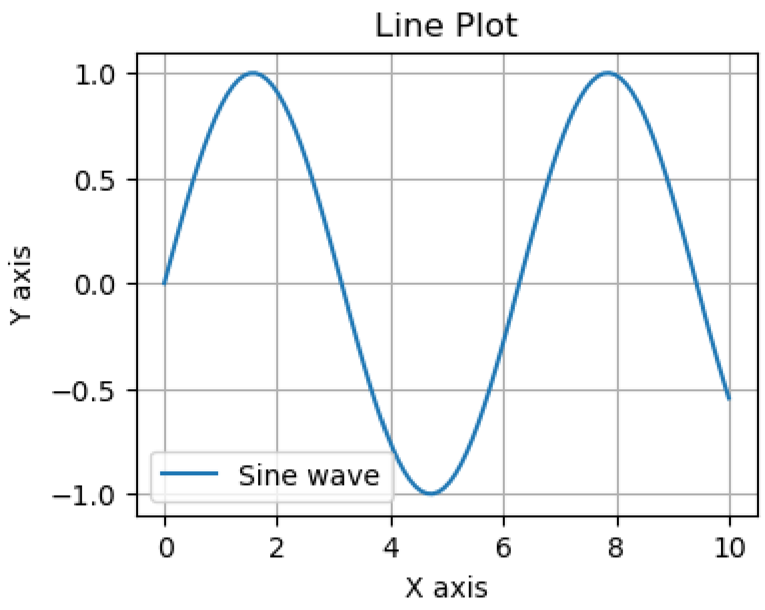
1. Line Plot

- Description: Line plots are perfect for showing trends over time or continuous data.
- Use Case: Tracking stock prices, temperature changes, or any time-series data.

In [12]:
```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.figure(figsize=(4, 3))
plt.plot(x, y, label='Sine wave')
plt.title('Line Plot')
plt.xlabel('X axis')
plt.ylabel('Y axis')
plt.legend()
plt.grid(True)
plt.show()
```



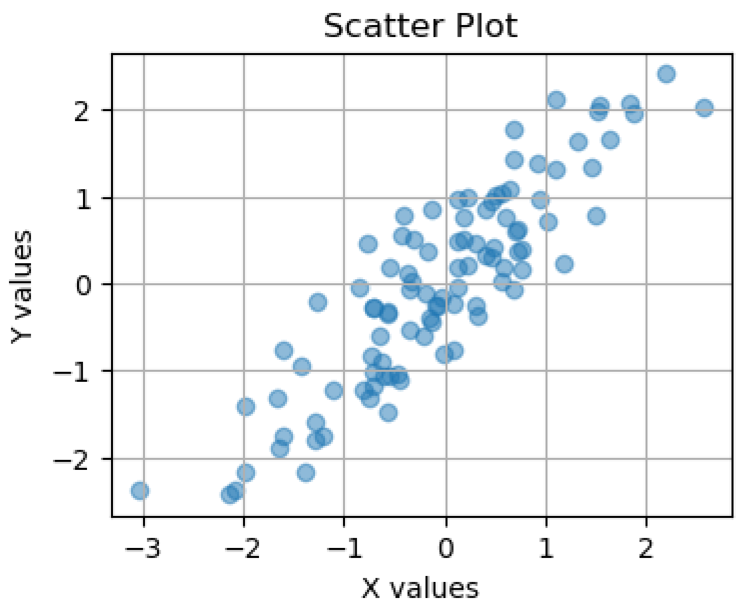
2. Scatter Plot

- Description: Scatter plots show the relationship between two variables.
- Use Case: Analyzing correlation between variables, such as height vs. weight.

In [13]:
```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(0, 1, 100)
y = x + np.random.normal(0, 0.5, 100)

plt.figure(figsize=(4, 3))
plt.scatter(x, y, alpha=0.5)
plt.title('Scatter Plot')
plt.xlabel('X values')
plt.ylabel('Y values')
plt.grid(True)
plt.show()
```



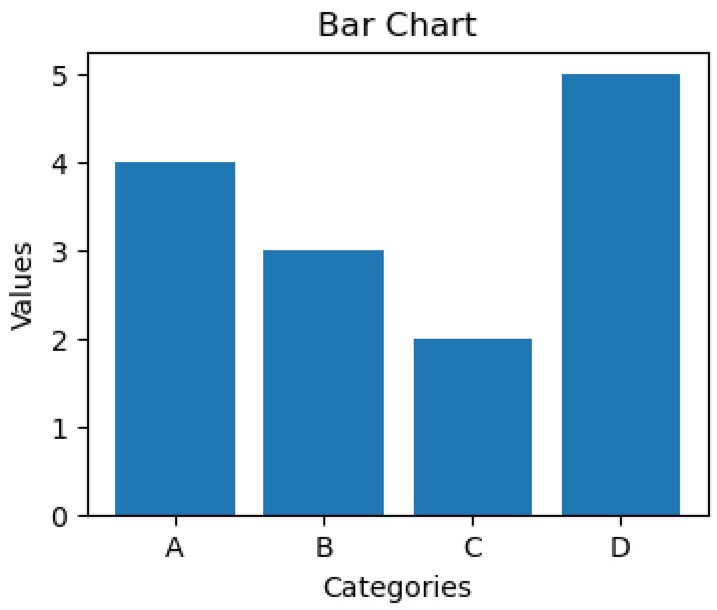
3. Bar Chart

- Description: Bar charts compare quantities across different categories.
- Use Case: Comparing sales across different products or regions.

In [14]:
```python
import matplotlib.pyplot as plt

categories = ['A', 'B', 'C', 'D']
values = [4, 3, 2, 5]

plt.figure(figsize=(4, 3))
plt.bar(categories, values)
plt.title('Bar Chart')
plt.xlabel('Categories')
plt.ylabel('Values')
plt.show()
```



Seaborn

Seaborn Overview

Seaborn is built on top of Matplotlib and provides a high-level interface for statistical graphics. Key features include:

- Beautiful default styles
- Built-in themes for professional-looking plots
- Integration with Pandas DataFrames
- Statistical estimation and visualization
- Complex visualizations with minimal code

Seaborn Graph Types

1. Distribution Plot

- Description: Visualizes the distribution of a dataset.
- Use Case: Analyzing the spread and shape of continuous data.

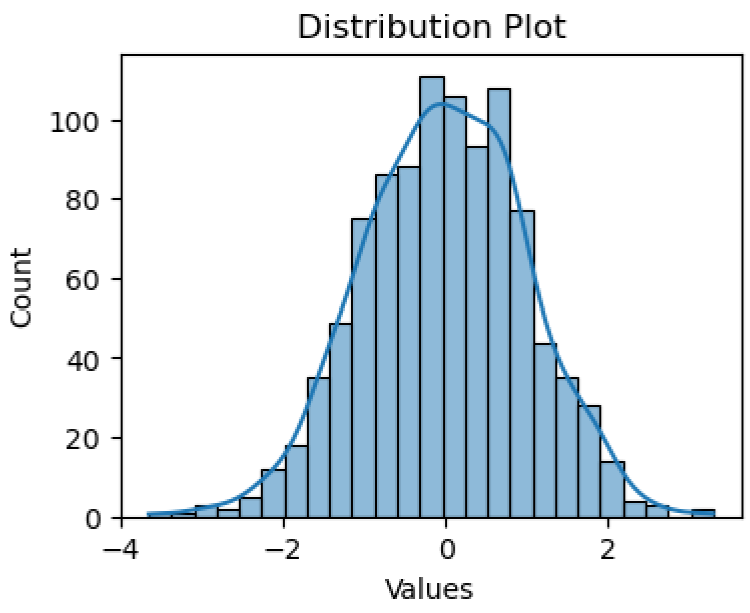
In [19]:

```
import seaborn as sns
import numpy as np

data = np.random.normal(0, 1, 1000)

plt.figure(figsize=(4, 3))
sns.histplot(data=data, kde=True)
plt.title('Distribution Plot')
plt.xlabel('Values')
plt.ylabel('Count')
plt.show()
```

```
C:\Users\Dhara\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



2. Box Plot

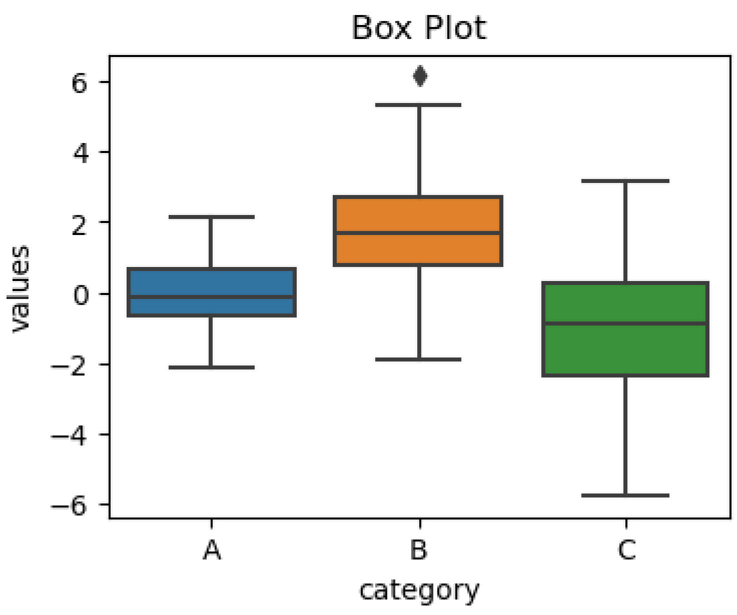
- Description: Shows the distribution of quantitative data across categories.
- Use Case: Comparing distributions and identifying outliers.

In [21]:

```
import seaborn as sns
import pandas as pd
import numpy as np

categories = ['A', 'B', 'C']
data = pd.DataFrame({
    'category': np.repeat(categories, 100),
    'values': np.concatenate([
        np.random.normal(0, 1, 100),
        np.random.normal(2, 1.5, 100),
        np.random.normal(-1, 2, 100)
    ])
})

plt.figure(figsize=(4, 3))
sns.boxplot(x='category', y='values', data=data)
plt.title('Box Plot')
plt.show()
```



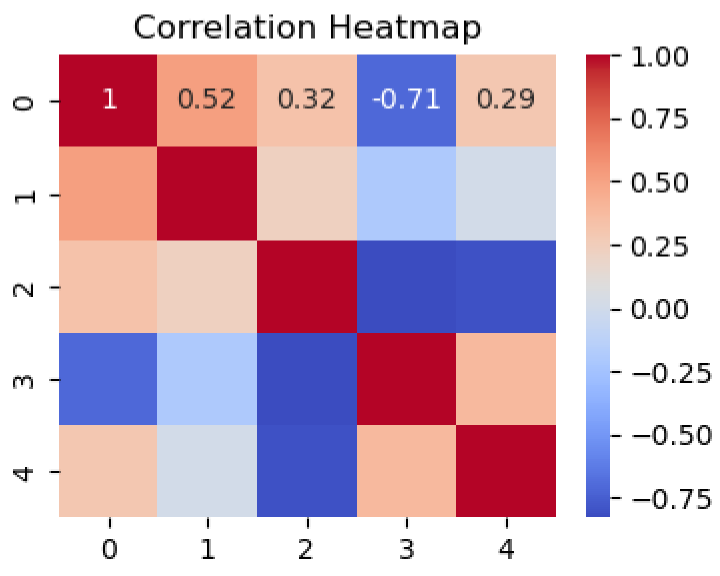
3. Heatmap

- Description: Displays matrix data as color-encoded cells.
- Use Case: Correlation matrices, confusion matrices, or any 2D matrix data.

In [22]:
```
import seaborn as sns
import numpy as np

data = np.random.rand(5, 5)
correlation_matrix = np.corrcoef(data)

plt.figure(figsize=(4, 3))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



Library Comparison

Matplotlib

Strengths:

- Complete control over plot elements
- Extensive customization options
- Large community and documentation
- Stable and mature library
- Great for publication-quality figures

Weaknesses:

- More verbose code required
- Default styles are basic
- Steeper learning curve
- Less intuitive for statistical plotting

Seaborn

Strengths:

- Beautiful default styles
- Easier to create complex statistical visualizations
- Excellent integration with Pandas
- Less code required for common plots
- Built-in themes and color palettes

Weaknesses:

- Less fine-grained control compared to Matplotlib
- Fewer plot types available
- Some advanced customizations require Matplotlib knowledge
- Can be slower with very large datasets

Performance Considerations

- Matplotlib is generally faster for simple plots
- Seaborn's statistical computations can slow down with large datasets
- Both libraries handle moderate-sized datasets well
- For very large datasets, consider using specialized libraries like Plotly or Bokeh

When to Use Each Library

- Use Matplotlib when:

    - You need complete control over plot elements
    - Creating publication-quality figures
    - Working with simple plots
    - Performance is critical

- Use Seaborn when:

    - Creating statistical visualizations
    - Working with Pandas DataFrames
    - Need attractive plots with minimal code
    - Don't need extensive customization

For more information, visit:

- Matplotlib: https://matplotlib.org/
- Seaborn: https://seaborn.pydata.org/