

Table of Contents



- [1. Installation](#)
- [2. Import Matplotlib](#)
- [3. Basic Plot](#)
- [4. Plot Types](#)
 - [Line Plot:](#)
 - [Bar Chart:](#)
 - [Histogram:](#)
 - [Scatter Plot:](#)
 - [Pie Chart:](#)
- [5. Customizing Plots](#)
- [6. Subplots](#)
- [7. Adding Text and Annotations](#)
- [8. Saving Plots](#)
- [9. Working with Multiple Lines](#)
- [10. Common Customizations](#)
- [11. Histogram Example](#)
- [12. Pie Chart Example](#)
- [13. Advanced Plot – Multiple Axes](#)
- [14. Scatter Plot with Regression Line](#)
- [15. Common Matplotlib Commands](#)
 - [Example: Complete Dashboard](#)

1. Installation

```
pip install matplotlib
```

or for Jupyter Notebooks:

```
pip install notebook matplotlib
```

2. Import Matplotlib

```
import matplotlib.pyplot as plt
```

For inline plots in Jupyter Notebooks:

```
%matplotlib inline
```

3. Basic Plot

```
import matplotlib.pyplot as plt
```

```
x = [1, 2, 3, 4, 5]  
y = [10, 15, 25, 30, 40]
```

```
plt.plot(x, y)  
plt.title('Basic Line Plot')  
plt.xlabel('X Axis')  
plt.ylabel('Y Axis')  
plt.show()
```

4. Plot Types

Line Plot:

```
plt.plot(x, y)
```

Bar Chart:

```
plt.bar(x, y)
```

Histogram:

```
plt.hist(y, bins=5)
```

Scatter Plot:

```
plt.scatter(x, y)
```

Pie Chart:

```
plt.pie(y, labels=x)
```

5. Customizing Plots

- Change Line Style & Color:

```
plt.plot(x, y, linestyle='--', color='r', marker='o')
```

- Add Grid:

```
plt.grid(True)
```

- Add Legend:

```
plt.legend(['Data'])
```

- Adjust Axis Limits:

```
plt.xlim(0, 10)  
plt.ylim(0, 50)
```

- Figure Size:

```
plt.figure(figsize=(8, 5))
```

6. Subplots

```
plt.figure(figsize=(10, 5))
```

```
plt.subplot(1, 2, 1)  
plt.plot(x, y)
```

```
plt.title('Plot 1')

plt.subplot(1, 2, 2)
plt.bar(x, y)
plt.title('Plot 2')

plt.tight_layout()
plt.show()
```

7. Adding Text and Annotations

```
plt.plot(x, y)
plt.text(3, 25, 'Peak Value')
plt.annotate('Max Point', xy=(5, 40), xytext=(3, 30),
            arrowprops=dict(facecolor='blue', shrink=0.05))
```

8. Saving Plots

```
plt.savefig('plot.png')
```

- File Formats: PNG, PDF, SVG, JPG

9. Working with Multiple Lines

```
plt.plot(x, y, label='Line 1')
plt.plot(x, [15, 20, 35, 40, 50], label='Line 2')
plt.legend()
```

10. Common Customizations

- Line Width:

```
plt.plot(x, y, linewidth=2)
```

- Alpha (Transparency):

```
plt.bar(x, y, alpha=0.7)
```

- Color Maps (Heatmaps/Scatter):

```
plt.scatter(x, y, c=y, cmap='viridis')  
plt.colorbar()
```

11. Histogram Example

```
data = [5, 15, 25, 35, 45, 55, 65, 75]  
plt.hist(data, bins=5, edgecolor='black')  
plt.title('Histogram Example')  
plt.show()
```

12. Pie Chart Example

```
sizes = [30, 20, 40, 10]  
labels = ['A', 'B', 'C', 'D']  
  
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)  
plt.title('Pie Chart Example')  
plt.show()
```

13. Advanced Plot - Multiple Axes

```
fig, ax1 = plt.subplots()

ax2 = ax1.twinx()
ax1.plot(x, y, 'g-')
ax2.plot(x, [10, 20, 30, 40, 50], 'b--')

ax1.set_xlabel('X Data')
ax1.set_ylabel('Primary Axis', color='g')
ax2.set_ylabel('Secondary Axis', color='b')

plt.show()
```

14. Scatter Plot with Regression Line

```
import numpy as np

x = np.linspace(0, 10, 50)
y = 3 * x + np.random.randn(50)

plt.scatter(x, y)
plt.plot(x, 3 * x, color='red')
plt.title('Scatter Plot with Regression Line')
plt.show()
```

15. Common Matplotlib Commands

Command	Description
plt.plot()	Line plot
plt.bar()	Bar chart
plt.scatter()	Scatter plot
plt.hist()	Histogram

Command	Description
<code>plt.pie()</code>	Pie chart
<code>plt.xlabel()</code> / <code>plt.ylabel()</code>	Axis labels
<code>plt.title()</code>	Plot title
<code>plt.grid()</code>	Add grid to the plot
<code>plt.legend()</code>	Show legend
<code>plt.show()</code>	Display plot

Example: Complete Dashboard

```
plt.figure(figsize=(12, 6))
```

```
plt.subplot(2, 2, 1)  
plt.plot(x, y)  
plt.title('Line Plot')
```

```
plt.subplot(2, 2, 2)  
plt.bar(x, y)  
plt.title('Bar Chart')
```

```
plt.subplot(2, 2, 3)  
plt.scatter(x, y)  
plt.title('Scatter Plot')
```

```
plt.subplot(2, 2, 4)  
plt.hist(y, bins=5)  
plt.title('Histogram')
```

```
plt.tight_layout()  
plt.show()
```