
📊 Matplotlib Detailed Notes (Beginner to Intermediate)

✅ Getting Started

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

📈 Line Plot

➤ Basic Line Plot

Plots a line between x and y coordinates.

x = [1, 2, 3, 4]

y = [2, 4, 1, 3]

plt.plot(x, y)

plt.show()

➤ Line Plot from Pandas DataFrame

df = pd.DataFrame({'year': [2010, 2011, 2012], 'sales': [100, 120, 130]})

plt.plot(df['year'], df['sales'])

➤ Plot Multiple Lines

plt.plot(x, y, label="Line 1")

plt.plot(x, [i+1 for i in y], label="Line 2")

➤ Add Title and Axis Labels

plt.title("Sales Over Years")

plt.xlabel("Year")

plt.ylabel("Sales")

➤ Customize Line Style, Color, and Marker

plt.plot(x, y, color='green', linestyle='--', linewidth=2, marker='o', markersize=8)

- color: color name or hex code (e.g., '#ff5733')

- linestyle: solid ('-'), dashed ('--'), dotted (':')
- marker: 'o', '*', 's' (circle, star, square)
- markersize: size of the marker in points

➤ **Add Legend**

```
plt.legend(loc='upper left')  # locations: best, upper right, lower left, etc.
```

➤ **Set Axis Limits**

```
plt.xlim(0, 10)
plt.ylim(0, 20)
```

➤ **Enable Grid**

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

🔵 Scatter Plot

➤ **Basic Scatter Plot**

```
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]
plt.scatter(x, y)
```

➤ **Scatter from DataFrame**

```
plt.scatter(df["feature1"], df["feature2"])
```

➤ **Customize Marker, Size, Color**

```
plt.scatter(x, y, color='red', marker='^', s=100, label="Data Points")
```

- s: size of marker
- marker: symbol
- color: color of points

➤ **Scatter using plt.plot()**

```
plt.plot(x, y, 'o')  # same as scatter with circles
```

➤ **Difference: plt.plot() vs plt.scatter()**

- plot(): Connects points with lines, default for time-series or trends.
- scatter(): Plots unconnected dots; good for correlation/relationships.

📊 Bar Chart

➤ Vertical Bar Chart

```
labels = ['A', 'B', 'C']

values = [10, 5, 8]

plt.bar(labels, values)
```

➤ Horizontal Bar Chart

```
plt.barh(labels, values)
```

➤ Multiple Bars (Grouped Bar Chart)

```
x = np.arange(len(labels))

plt.bar(x - 0.2, [5, 6, 7], width=0.4, label="2020")

plt.bar(x + 0.2, [6, 7, 8], width=0.4, label="2021")

plt.xticks(x, labels)

plt.legend()
```

➤ Fix Overlapping Labels

```
plt.xticks(rotation=45)
```

➤ Stacked Bar Chart

```
bottom_vals = [5, 3, 4]

top_vals = [2, 4, 1]

plt.bar(labels, bottom_vals, label='Base')

plt.bar(labels, top_vals, bottom=bottom_vals, label='Stacked')
```

📉 Histogram

➤ Create a Histogram

```
data = [1, 2, 2, 3, 3, 3, 4, 4, 5]

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

- bins: number of intervals or groups
- Good for showing frequency distribution

➤ Histogram with Log Scale

```
plt.hist(data, log=True)
```

🥧 Pie Chart

➤ Basic Pie Chart

```
labels = ['Python', 'Java', 'C++']

sizes = [50, 30, 20]

plt.pie(sizes, labels=labels)
```

➤ Add Percentages and Colors

```
colors = ['#ff9999', '#66b3ff', '#99ff99']

plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%')
```

- autopct: format to display percentage values

➤ Add Explode and Shadow

```
explode = [0, 0.1, 0]  # Explode second slice

plt.pie(sizes, labels=labels, explode=explode, shadow=True, autopct='%1.1f%%')
```

🎨 Styling and Customization

➤ Change Plot Style

Use predefined styles:

```
plt.style.use('ggplot')     # 'seaborn', 'classic', 'bmh', 'fivethirtyeight', etc.
```

➤ Save Figure

```
plt.savefig("plot.png")     # Save as PNG

plt.savefig("plot.pdf")     # Save as PDF
```
