Matplotlib Beginner Cheat Sheet

1. Import & Setup

The standard convention for importing Matplotlib's 'pyplot' module.

```
import matplotlib.pyplot as plt
import numpy as np # Often used for data generation
```

2. Anatomy of a Plot

Understanding the two core objects is key to using Matplotlib effectively.

- **Figure**: The top-level container for all plot elements (the entire window or page).
- **Axes**: An individual plot or graph. A Figure can contain one or more Axes. You use the Axes object to plot data and customize it.

3. The Core Plotting Workflow (OO Style)

The recommended way to create plots. This approach is explicit and more flexible, especially for multiple plots.

1. Create a Figure and one or more Axes with `plt.subplots()`.
2. Call plotting methods on the Axes object (e.g., `ax.plot()`).
3. Customize the plot using methods of the Axes object (e.g., `ax.set_title()`).
4. Display the plot with `plt.show()`.

```
# 1. Create a Figure and an Axes
fig, ax = plt.subplots()

# Create some data
x = np.linspace(0, 10, 100)
y = np.sin(x)

# 2. Plot data on the Axes
ax.plot(x, y)

# 3. Customize the plot
ax.set_title("A Simple Sine Wave")
ax.set_xlabel("x-axis")
ax.set_ylabel("y-axis")
ax.grid(True)

# 4. Show the plot
plt.show()
```

4. Common Plot Types

All of these are methods of the `Axes` object (`ax`).

Line Plot

Good for showing trends over continuous data.

```
x = np.linspace(0, 2, 100)
ax.plot(x, x, label='linear')
ax.plot(x, x**2, label='quadratic')
ax.plot(x, x**3, label='cubic')
```

Scatter Plot

Good for showing the relationship between two variables.

```
x = np.random.rand(50)
y = np.random.rand(50)
ax.scatter(x, y)
```

Bar Chart

Good for comparing quantities across categories.

```
categories = ['A', 'B', 'C']
values = [10, 25, 17]
ax.bar(categories, values) # Vertical
# ax.barh(categories, values) # Horizontal
```

Histogram

Good for showing the distribution of a single variable.

```
data = np.random.randn(1000)
ax.hist(data, bins=30)
```

Box Plot

Shows the distribution summary (min, max, median, quartiles).

```
data = [np.random.normal(0, std, 100) for std in
    range(1, 4)]
ax.boxplot(data, vert=True, patch_artist=True)
```

5. Customizing the Plot

Add labels, titles, and legends to make plots understandable.

```
# Add a title to the Axes
ax.set_title('My Plot Title', fontsize=16)

# Add axis labels
ax.set_xlabel('X-Axis Label')
ax.set_ylabel('Y-Axis Label')

# Set axis limits
ax.set_xlim([0, 10])
ax.set_ylim([-1, 1])

# Add a grid
ax.grid(True, linestyle='--', alpha=0.6)

# Add a legend
# Note: You must add a 'label' to your plot calls
ax.plot(x, y, label='My Data')
ax.legend(loc='upper right') # Best location
```

6. Styling Lines & Markers

Control the appearance of your plots with keyword arguments.

```
ax.plot(x, y,
    color='red',          # or '#FF0000' or 'r'
    linestyle='--',     # '--', '-', '-.', ':'
    linewidth=2,
    marker='o',           # 'o', '.', ',', 'x', '+'
    markersize=5,
    alpha=0.8,            # Transparency
    label='Styled Line')
```

Matplotlib Beginner Cheat Sheet

7. Multiple Plots (Subplots)

Create a grid of Axes within a single Figure.

```
# Create a 2x2 grid of subplots
fig, axs = plt.subplots(2, 2, figsize=(8, 6))

# axs is a 2D NumPy array; access each Axes by index
axs[0, 0].plot(x, y)
axs[0, 0].set_title('Top-Left')

axs[0, 1].scatter(np.random.rand(20), np.random.rand
    (20))
axs[0, 1].set_title('Top-Right')

axs[1, 0].hist(np.random.randn(100))
axs[1, 0].set_title('Bottom-Left')

axs[1, 1].bar(['A', 'B'], [5, 8])
axs[1, 1].set_title('Bottom-Right')

# Adjust layout to prevent labels from overlapping
plt.tight_layout()

plt.show()
```

8. Saving & Displaying

The final steps to output your work.

Displaying the Plot

Renders the figure in a window. Call this once at the end of your script.

```
plt.show()
```

Saving the Plot

Saves the figure to a file. **Call this before `plt.show()`.**

```
# Save the figure to a file
fig.savefig('my_plot.png') # .pdf, .svg, .jpg

# Save with higher resolution (dots per inch)
fig.savefig('my_plot_high_res.png', dpi=300)
```