

Matplotlib クイックリファレンス

図、軸、プロット、カスタマイズ

基本プロット

折れ線グラフ

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [2, 4, 1, 8, 3]
plt.plot(x, y)
plt.show()
```

クイックプロットショートカット

```
plt.plot(y) # x auto 0..N-1
plt.plot(x, y, "ro--") # red circles, dashed
plt.plot(x, y, "bs-") # blue squares, solid
```

フォーマット文字列コード

| | | | | |
|-----|------|------|-----|----------------|
| 'r' | 'g' | 'b' | 'k' | 赤、緑、青、黒 |
| 'o' | 's' | '^' | 'D' | 円、四角、三角、ダイヤモンド |
| '-' | '--' | '-.' | '.' | 実線、破線、一点鎖線、点線 |

サブプロット

Figure と Axes

```
fig, ax = plt.subplots()
ax.plot(x, y)
ax.set_title("Single Plot")
plt.show()
```

サブプロットのグリッド

```
fig, axes = plt.subplots(2, 2, figsize=(8, 6))
axes[0, 0].plot(x, y)
axes[0, 1].bar(x, y)
axes[1, 0].scatter(x, y)
fig.tight_layout()
```

軸の共有

```
fig, (ax1, ax2) = plt.subplots(1, 2,
                              sharey=True, figsize=(10, 4))
ax1.plot(x, y)
ax2.plot(x, y2)
```

ラベルとタイトル

軸ラベルとタイトル

```
plt.plot(x, y)
plt.xlabel("Time (s)")
plt.ylabel("Value")
plt.title("Sensor Reading")
plt.show()
```

OO スタイルのラベル

```
fig, ax = plt.subplots()
ax.plot(x, y)
ax.set_xlabel("X"); ax.set_ylabel("Y")
ax.set_title("My Plot")
```

注釈

```
ax.annotate("Peak", xy=(4, 8),
            xytext=(3, 9),
            arrowprops=dict(arrowstyle="->"))
```

カスタマイズ

色とスタイル

```
plt.plot(x, y, color="#FF5733",
         linewidth=2, linestyle="--")
plt.plot(x, y2, color="steelblue",
         marker="o", markersize=5)
```

Figure のサイズと DPI

```
fig, ax = plt.subplots(figsize=(10, 6), dpi=100)
plt.rcParams["figure.figsize"] = (8, 5)
```

スタイルシート

```
print(plt.style.available) # list all
plt.style.use("seaborn-v0_8")
plt.style.use("ggplot")
```

棒グラフとヒストグラム

棒グラフ

```
labels = ["A", "B", "C", "D"]
values = [23, 45, 12, 37]
plt.bar(labels, values, color="teal")
plt.show()
```

グループ化 / 積み上げ棒グラフ

```
import numpy as np
x = np.arange(4); w = 0.35
plt.bar(x - w/2, v1, w, label="2024")
plt.bar(x + w/2, v2, w, label="2025")
plt.xticks(x, labels)
```

ヒストグラム

```
data = np.random.randn(1000)
plt.hist(data, bins=30, edgecolor="black",
         alpha=0.7)
plt.show()
```

散布図と折れ線

散布図

```
plt.scatter(x, y, c="red", s=50,
           alpha=0.6, edgecolors="black")
plt.show()
```

カラーマップ付き散布図

```
sc = plt.scatter(x, y, c=values,
                 cmap="viridis", s=sizes)
plt.colorbar(sc, label="Intensity")
```

複数の折れ線

```
plt.plot(x, y1, label="Train")
plt.plot(x, y2, label="Validation")
plt.legend()
plt.show()
```

軸と目盛り

軸の範囲とスケール

```
ax.set_xlim(0, 10)
ax.set_ylim(-1, 1)
ax.set_xscale("log")
ax.set_yscale("log")
```

カスタム目盛り

```
ax.set_xticks([0, 1, 2, 3, 4])
ax.set_xticklabels(["Mon", "Tue", "Wed",
                  "Thu", "Fri"], rotation=45)
```

グリッド

```
ax.grid(True, linestyle="--", alpha=0.5)
ax.grid(axis="y") # horizontal only
```

凡例

凡例の追加

```
ax.plot(x, y, label="Series A")
ax.plot(x, y2, label="Series B")
ax.legend(loc="upper right")
```

凡例の配置

| | |
|-------------------------|------------------|
| 'best' | 自動で最適な位置 (デフォルト) |
| 'upper left' | 左上 |
| 'lower right' | 右下 |
| 'center' | 軸の中央 |
| 'bbox_to_anchor=(1, 1)' | 軸の外側に配置 |

凡例のカスタマイズ

```
ax.legend(fontsize=8, frameon=False,
         ncol=2, title="Legend")
```

保存

ファイルに保存

```
plt.savefig("plot.png", dpi=300,
         bbox_inches="tight")
plt.savefig("plot.pdf")
plt.savefig("plot.svg", transparent=True)
```

対応フォーマット

| | |
|------------|-------------------------|
| PNG | ラスター、Web / 画面に最適 |
| PDF | ベクター、印刷 / 論文に最適 |
| SVG | ベクター、Web でのスケーラブルな使用に最適 |
| EPS | ベクター、旧来の科学誌向け |

Figure オブジェクトから保存

```
fig, ax = plt.subplots()
ax.plot(x, y)
fig.savefig("output.png", dpi=150,
         facecolor="white")
```

よく使うパターン

双軸 (Y スケール 2 つ)

```
fig, ax1 = plt.subplots()
ax2 = ax1.twinx()
ax1.plot(x, temp, "r-", label="Temp")
ax2.plot(x, pressure, "b-", label="Pressure")
```

塗りつぶし

```
ax.fill_between(x, y_low, y_high,
               alpha=0.3, color="blue")
```

Matplotlib クイックリファレンス

imshow によるヒートマップ

```
data = np.random.rand(10, 10)
plt.imshow(data, cmap="hot",
            interpolation="nearest")
plt.colorbar()
```

円グラフ

```
plt.pie(sizes, labels=labels,
        autopct="%1.1f%%", startangle=90)
plt.axis("equal")
```