

Python Graph 学習ガイド

河合勝彦

名古屋市立大学大学院経済学研究科

2026 年 1 月 12 日

概要

本ガイドは、Python と Matplotlib を用いたグラフ作成を体系的に学ぶための学習資料である。リポジトリ構成、画像再生成フロー、代表的な可視化手法を整理し、数式の補足や学習課題、練習問題を通じて理解を深める。初級から中級の学習者が、実装とドキュメント運用を連携させながら学べるよう構成している。

キーワード: Matplotlib, NumPy, 可視化, プロット, 学習ガイド

目次

1	ガイドの目的と前提	2
2	リポジトリ構成の全体像	2
3	利用関数インデックス	3
3.1	Matplotlib (pyplot)	3
3.2	Matplotlib (Axes / 3D)	3
3.3	NumPy	3
4	画像生成の中心スクリプト: <code>generate_graphs.py</code>	3
5	基本 2D プロットの読み解き	4
6	画像表示と等高線プロット	5
7	ベクトル場と極座標プロット	6
8	3D サーフェスプロット	6
9	<code>images/</code> に保存される成果物	7

10	graph.md の構成と学び方	8
11	graph.html と index.html の役割	8
12	style.css のスタイル設計	8
13	package-lock.json とフロントエンド依存	9
14	再生成ワークフロー	9
15	次のステップ（発展）	9
16	練習問題	10
17	模範解答	11
18	参考サイト	11
19	付録: Python グラフ作成に役立つ LLM プロンプト集	11

1 ガイドの目的と前提

このガイドは、Matplotlib による代表的なグラフをまとめたリポジトリを、初級から中級の学習者が自力で理解・拡張できるように整理したものです。主な学習対象は、サンプル画像の自動生成、Markdown ドキュメントの構成、HTML への展開と見た目の調整です。

必要な前提は以下です。

- Python 3.9 以上の実行環境
- Matplotlib と NumPy の基礎知識
- HTML/CSS の基本的な読み書き

2 リポジトリ構成の全体像

このリポジトリは、ソースと成果物が同じ階層に揃うように設計されています。主なファイルと役割は以下のとおりです。

パス	概要
<code>generate_graphs.py</code>	PNG サンプル画像の一括生成スクリプト
<code>images/</code>	生成されたグラフ画像の保存先
<code>graph.md</code>	代表的なプロットとサンプルコードの一覧
<code>graph.html</code>	<code>graph.md</code> を HTML 化したドキュメント

<code>index.html</code>	ランディングページ
<code>style.css</code>	HTML ページ全体のスタイル
<code>package-lock.json</code>	Web 用依存関係のロックファイル

3 利用関数インデックス

主に使用する関数を用途別に整理します。

3.1 Matplotlib (pyplot)

- `plt.plot()`, `plt.scatter()`, `plt.bar()`, `plt.hist()`, `plt.boxplot()`, `plt.pie()`
- `plt.imshow()`, `plt.contour()`, `plt.contourf()`, `plt.colorbar()`
- `plt.quiver()`, `plt.subplot()`, `plt.subplots()`, `plt.legend()`
- `plt.savefig()`, `plt.show()`, `plt.tight_layout()`

3.2 Matplotlib (Axes / 3D)

- `ax.plot()`, `ax.scatter()`, `ax.bar()`, `ax.hist()`
- `ax.plot_surface()`, `ax.plot_wireframe()`
- `ax.contour()`, `ax.contourf()`, `ax.bar3d()`

3.3 NumPy

- `np.linspace()`, `np.meshgrid()`, `np.mgrid`
- `np.random.seed()`, `np.random.rand()`, `np.random.randn()`, `np.random.normal()`
- `np.exp()`, `np.sin()`, `np.sqrt()`, `np.sort()`
- `np.polyfit()`, `np.corrcoef()`

4 画像生成の中心スクリプト：generate_graphs.py

サンプル画像はすべて `generate_graphs.py` で生成されます。特徴は次のとおりです。

- 出力先は `IMAGE_DIR = "images"` で統一されています。
- 各プロットは関数ごとに分離され、再利用と学習がしやすい構成です。
- 乱数を使う箇所では `np.random.seed(0)` により再現性が確保されています。

主要なエントリポイントは次のとおりです。

```
if __name__ == "__main__":  
    main()
```

この入口から順に各プロット関数が呼ばれ、PNG が生成されます。ここでは「実行されたら main を呼ぶ」という最小の流れだけを示しており、実際の処理は main 内に集約されています。例えば、折れ線グラフの関数は次のような構成です。

```
def line_plot():  
    x = np.linspace(0, 10, 100)  
    y = np.sin(x)  
    plt.plot(x, y)  
    plt.title("Line Plot")  
    plt.savefig(f"{IMAGE_DIR}/line_plot.png")  
    plt.close()
```

ポイントは、1つの関数で「データ生成→描画→保存→クローズ」を完結させることです。これにより、関数を増やしても副作用が残りにくく、順番に呼ぶだけで一括生成できます。

5 基本 2D プロットの読み解き

基本的なグラフは、次の関数で生成されています。

- `line_plot()`：正弦波の折れ線グラフ
- `scatter_plot()`：色分け付き散布図
- `bar_chart()`：カテゴリ別の棒グラフ
- `histogram()`：分布を示すヒストグラム
- `box_plot()`：箱ひげ図による分布比較
- `pie_chart()`：構成比を示す円グラフ

それぞれの関数は、`plt.title()` でタイトルを付けた後に `plt.savefig()` で保存しています。学習の第一歩として、ラベルや色、数値の変更を試すと理解が深まります。例えば散布図では次のようにラベルとカラーマップが指定されています。

```
plt.scatter(x, y, c=x, cmap="viridis", marker="o")  
plt.xlabel("X")  
plt.ylabel("Y")  
plt.title("Scatter Plot")  
plt.colorbar()
```

ここでは `c=x` によって `x` の値を色に割り当てています。色が連続的に変化するので、分布の偏りや傾向が視覚的にわかりやすくなります。ラベルを日本語に変えたり、`cmap` を変更したりするだけで印象が大きく変わります。

■数式の視点 折れ線グラフの例では、 $y = \sin(x)$ を描いています。サイン波は周期 2π を持つため、 x の範囲を広げると振動回数が増えます。散布図では (x_i, y_i) の点集合を可視化しており、色 $c(x_i)$ を重ねることで第三の変数を表現しています。

■図ごとの学習課題（チェックリスト）

- 折れ線グラフ: $y = \sin(x)$ を $y = \sin(2x)$ に変え、周期の違いを観察する。期待される観察結果: 波の山と谷の間隔が半分になり、同じ範囲で振動回数が増える。
- 散布図: $c = y$ に変更して色づけの軸を比較する。期待される観察結果: 色の濃淡が y の増減に沿って変化し、縦方向の傾向が見えやすくなる。
- 棒グラフ: カテゴリ数を増やし、ラベルの重なりを解消する工夫を考える。期待される観察結果: ラベルが詰まり、回転や省略などの調整が必要だと分かる。
- ヒストグラム: `bins` を 10/50 に変え、分布の見え方を比較する。期待される観察結果: ビン数を増やすと細部が見えるがノイズも増える。
- 箱ひげ図: 各系列の平均を変えて、中央値と四分位範囲の変化を読む。期待される観察結果: 中央値が上下に移動し、分布の広がりは大きくは変わらない。
- 円グラフ: パーセンテージの合計が 100% になるように値を調整する。期待される観察結果: 各扇形の比率が合計 100% で閉じることを確認できる。

6 画像表示と等高線プロット

2D 配列を使った可視化は、画像処理やヒートマップの基礎になります。

- `image_display()`: 乱数画像の可視化とカラーバー
- `filled_contour_plot()`: ガウス状の面を等高線で可視化

この領域では、`imshow` の `cmap` や `contourf` の `levels` を変更して、表現の違いを確認するのが効果的です。以下は等高線の塗りつぶし例です。

```
plt.contourf(X, Y, Z, levels=20, cmap="coolwarm")
plt.colorbar()
plt.title("Filled Contour Plot")
```

等高線は等しい高さを連続的に結ぶ線で、`contourf` はその間を塗りつぶします。段数を増やすと滑らかな印象になり、少なくすると階調の変化が強調されます。

■数式の視点 ここで使っている面は $Z = \exp(-(X^2 + Y^2))$ で、原点を中心に同心円状に減衰するガウス型の面です。`exp` の性質により、原点から離れるほど値が急速に小さくなります。

■図ごとの学習課題（チェックリスト）

- 画像表示: `cmap` を `gray` に変更し、濃淡表現の違いを比較する。期待される観察結果: 色相がなくなり、明暗のみで構造が見える。
- 等高線: $Z = \sin(X) \cos(Y)$ に変更し、周期的な縞模様を確認する。期待される観察結果: 縞が格子状に広がり、周期性が明確になる。
- 等高線: `levels` を 5/40 に変更し、滑らかさを比較する。期待される観察結果: 5 では段差が強く、40 では滑らかなグラデーションになる。

7 ベクトル場と極座標プロット

動きや方向を表現する図は、数学・物理・データ可視化に応用できます。

- `quiver_plot()`: ベクトル場の矢印表現
- `polar_plot()`: 極座標で描く周期的な曲線

ベクトル場の式を変更し、矢印の密度や大きさがどう変わるか観察してみましょう。矢印プロットは次のように計算されたベクトルを可視化しています。

```
Y, X = np.mgrid[-3:3:100j, -3:3:100j]
U = -1 - X**2 + Y
V = 1 + X - Y**2
plt.quiver(X, Y, U, V)
```

ここでは U と V が矢印の向きと長さを決めます。式を一次式に変更するだけでも、流れの形が大きく変化します。

■数式の視点 ベクトル場は $\vec{v}(x, y) = (U(x, y), V(x, y))$ で表されます。現在の定義では二次式が入っており、原点近傍と周辺で向きが大きく変わる流れ場になります。

■図ごとの学習課題（チェックリスト）

- ベクトル場: $U = -Y, V = X$ に変更し、回転場を確認する。期待される観察結果: 原点周りの円運動を示す矢印が現れる。
- ベクトル場: 100j を 20j に変更し、矢印密度の違いを比較する。期待される観察結果: 矢印が粗くなり全体の流れが見やすくなる。
- 極座標: $r = 1 + \sin(4\theta)$ を $r = 1 + \sin(6\theta)$ に変更し、花卉数を観察する。期待される観察結果: 花卉の数が 4 から 6 が増える。

8 3D サーフェスプロット

`surface_plot()` では 3D の面を描画しています。`mpl_toolkits.mplot3d` を利用し、`plot_surface()` で高さの変化を表現します。

3D プロットは視点変更やカラーマップ変更で理解が深まります。必要に応じて `ax.view_init()` を追加し、角度の違いを確認してください。視点変更の例は次のとおりです。

```
ax.view_init(elev=30, azim=45)
```

第一引数が仰角、第二引数が方位角です。角度を変えることで凹凸の見え方が変わり、データの特徴を把握しやすくなります。

■数式の視点 3D 面では $Z = \sin(\sqrt{X^2 + Y^2})$ を使っています。 $r = \sqrt{X^2 + Y^2}$ は原点からの距離なので、円環状に波が広がる同心円パターンになります。

■図ごとの学習課題（チェックリスト）

- ☐ 3D 面: `sin` を `cos` に変え、山と谷の位置がどう変わるか比較する。期待される観察結果: 波の位相がずれ、山と谷の位置が入れ替わる。
- ☐ 3D 面: グリッドの分割数を 30/80 に変更し、滑らかさと計算量の違いを体感する。期待される観察結果: 分割数が増えるほど滑らかになるが処理が重くなる。
- ☐ 視点: `elev` と `azim` を変更し、凹凸が読み取りやすい角度を探す。期待される観察結果: 角度によって陰影の見え方が変わり、形状の理解が容易になる。

9 images/ に保存される成果物

`generate_graphs.py` の実行後、次の PNG が生成されます。

- `images/line_plot.png`
- `images/scatter_plot.png`
- `images/bar_chart.png`
- `images/histogram.png`
- `images/box_plot.png`
- `images/pie_chart.png`
- `images/image_display.png`
- `images/filled_contour_plot.png`
- `images/quiver_plot.png`
- `images/polar_plot.png`
- `images/3d_surface_plot.png`
- `images/regression_plot.png`
- `images/residuals_plot.png`
- `images/correlation_heatmap.png`

学習の中級ステップとして、`plt.savefig()` のファイル名を変更し、`graph.md` 側の参照も合わ

せて更新する練習をすると良いです。

10 graph.md の構成と学び方

graph.md はプロット一覧のカタログとして機能します。表形式でプロット種別と関数を整理し、後半にはサンプルコードと画像が並びます。統計分析向けの可視化として、回帰直線付き散布図、残差プロット、相関行列ヒートマップも追加しています。画像の埋め込みは Markdown で次のように記述します。

```
![Line Plot](images/line_plot.png)
```

角括弧内が代替テキスト、丸括弧内が画像パスです。ファイル名と画像が一致しない場合は表示されないため、生成スクリプトと合わせて管理します。

おすすめの学習手順は次のとおりです。

1. 表で目的のプロットを見つける
2. サンプルコードを読む
3. 該当する generate_graphs.py の関数を確認する
4. 実行して画像の変化を確認する

11 graph.html と index.html の役割

graph.html は graph.md を HTML 化した成果物で、style.css を読み込みます。index.html はトップページとして、ドキュメントへの導線や概要を提供します。

学習の観点では、graph.md を更新して graph.html を再生成し、ブラウザで見た目を確認する流れが重要です。

12 style.css のスタイル設計

style.css では、色やタイポグラフィ、レイアウトに関する変数が集中管理されています。以下のようなテーマ設計が特徴です。

- CSS 変数による配色管理
- ヒーローセクションやカード UI の専用スタイル
- ダークモード対応

中級者向けの練習として、--color_primary や --color_bg を変更し、全体の印象がどう変わるか観察してください。色設計は CSS 変数でまとめて調整できます。

```
:root {  
  --color-bg: #eef2ff;
```



```
--color-primary: #2563eb;  
}
```

:root に定義した変数は全体に適用されるため、変数を変えるとページ全体の配色が連動して更新されます。

13 package-lock.json とフロントエンド依存

package-lock.json は、HTML/CSS 周辺のビルドやツール実行時に使われる依存関係を固定します。直接の編集は避け、必要な場合は `npm install` で更新します。

学習の目的は、フロントエンドのビルドに依存関係があることを理解し、変更時にロックファイルを更新する意識を持つことです。

14 再生成ワークフロー

このリポジトリでは、次の順で作業するのが基本です。

1. Python 仮想環境を用意し、matplotlib と numpy をインストール
2. `python generate_graphs.py` で画像を再生成
3. `graph.md` を編集
4. `pandoc graph.md -s -c style.css -o graph.html` を実行
5. ブラウザで `index.html` と `graph.html` を確認

代表的なコマンドは以下です。

```
python generate_graphs.py  
pandoc graph.md -s -c style.css -o graph.html
```

前者で PNG を再生成し、後方で Markdown を HTML に変換します。実行後は `images/` と HTML の更新内容を確認します。

15 次のステップ（発展）

中級者向けの発展課題として、次のようなテーマがあります。

- 新しいプロット関数を `generate_graphs.py` に追加する
- `graph.md` の表に新しいカテゴリを追加する
- `style.css` のタイポグラフィを改良する
- `index.html` に新しいカードセクションを追加する

16 練習問題

1. [初級] `generate_graphs.py` で画像を保存する関数の共通パターンを簡潔に説明せよ。

2. [初級] 折れ線グラフで $y = \sin(2x)$ に変更したとき、周期はどう変化するか。

3. [初級] 散布図で $c=x$ と $c=y$ の違いを、可視化上の効果として述べよ。

4. [中級] ヒストグラムの `bins` を増やすと、どのような長所と短所があるか。

5. [中級] 等高線で $\exp(-(X^2 + Y^2))$ を使う理由を、形状の特徴として説明せよ。

6. [初級] ベクトル場を $U = -Y, V = X$ に変更すると、どのような流れが現れるか。

7. [中級] 3D サーフェスで $Z = \sin(\sqrt{X^2 + Y^2})$ の意味を、幾何学的に説明せよ。

8. [初級] `graph.md` の画像埋め込み記法の要素（角括弧と丸括弧の意味）を述べよ。

9. [初級] `style.css` の `:root` に変数を置く利点を 1 つ挙げよ。

10. [中級] `pandoc graph.md -s -c style.css -o graph.html` の各オプションの役割を説明せよ。

17 模範解答

1. [初級] データ生成→描画→保存→クローズを 1 関数内で完結させ、`main()` から順に呼び出して一括生成する。
2. [初級] 周期が半分になり、同じ範囲で振動回数が 2 倍になる。
3. [初級] `c=x` は横方向の値に応じて色が変わり、`c=y` は縦方向の値に応じて色が変わる。
4. [中級] 長所は分布の細部が見えること、短所はノイズが強調され滑らかさが失われること。
5. [中級] 原点を中心に同心円状に減衰する面となり、中心が高く周辺が低い形状になる。
6. [初級] 原点周りの回転場が現れ、円運動を示す矢印が並ぶ。
7. [中級] $r = \sqrt{X^2 + Y^2}$ が原点からの距離で、距離に応じて波が上下する同心円状の波面を表す。
8. [初級] 角括弧は代替テキスト、丸括弧は画像ファイルのパスを指定する。
9. [初級] 変数を一箇所で管理でき、配色変更が全体に一括で反映される。
10. [中級] `-s` はスタンドアロン化、`-c` は CSS の指定、`-o` は出力ファイル名の指定。

18 参考サイト

- Matplotlib User Guide: <https://matplotlib.org/stable/users/index.html>
- Matplotlib Tutorials: <https://matplotlib.org/stable/tutorials/index.html>
- NumPy User Guide: <https://numpy.org/doc/stable/user/index.html>
- Pandoc User Guide: <https://pandoc.org/MANUAL.html>
- MDN Web Docs (CSS): <https://developer.mozilla.org/docs/Web/CSS>

19 付録: Python グラフ作成に役立つ LLM プロンプト集

用途別に初級から少し高度なものまでをまとめています。必要に応じてデータや要件を差し替えて利用してください。

1. [初級] 折れ線グラフの基本テンプレート

Pythonで折れ線グラフを描く最小例をください。
NumPyでxを0~10まで100分割し、Matplotlibで`y=sin(x)`を描画。
保存ファイル名は`line_plot.png`、`plt.savefig`まで含めて。

2. [初級] 散布図の色分け
-

散布図でxとyを乱数で50点生成し、xの値で色分けしてください。
NumPyで乱数を生成し、Matplotlibで描画。カラーマップはviridis、カラーバーも表示。

3. [初級] 棒グラフのカテゴリ比較

カテゴリ["A", "B", "C", "D"]と値[10, 24, 36, 18]の棒グラフを作成。
Matplotlibで描画し、タイトルとy軸ラベルを追加して保存してください。

4. [初級] ヒストグラムの基本

NumPyで標準正規分布の乱数1000件を生成し、Matplotlibでヒストグラムを作成。
bins=30、灰色、枠線は黒にしてください。

5. [初級] 円グラフの比率表示

リンゴ/バナナ/さくらんぼ/デーツの比率を円グラフで表示。
Matplotlibでパーセンテージ表示(小数1桁)を付けてください。

6. [初級] 箱ひげ図の比較

NumPyで4系列の正規乱数データ(各100件)を作り、Matplotlibで箱ひげ図を描画。
系列ごとに平均をずらして差を見せてください。

7. [初級] 画像表示 (imshow)

NumPyで10x10の乱数配列を作成し、Matplotlibのimshowで表示。
カラーマップはviridis、タイトルも追加してください。

8. [初級] 等高線 (contourf) の基本

X, Yを-3~3のグリッドにし、 $Z = \exp(-(X^2 + Y^2))$ を描く。
Matplotlibのcontourfでlevels=20、cmap=coolwarm、カラーバー付きで。

9. [初級] ベクトル場 (quiver) の基本

X, YのグリッドからU, Vを計算しquiverで描画する例を示して。
NumPyで格子を作り、Matplotlibのquiverで描画。U=-1-X**2+Y, V=1+X-Y**2 を使う。

10. [初級] 極座標プロット

NumPyでthetaを0~ 2π で100分割し、Matplotlibの極座標で描画。
 $r = 1 + \sin(4 * \theta)$ を使ってください。

11. [初級] 3D サーフェスの最小例

X, Yを-5~5のグリッド、 $Z = \sin(\sqrt{X^2 + Y^2})$ の3D面を描画。
Matplotlibのmplot3dでplot_surfaceを使い、viridisカラーマップを使用。

12. [中級] 複数サブプロットの配置

2x2のサブプロットに折れ線/散布/棒/ヒストグラムを配置。
Matplotlibのsubplotsで作成し、各サブプロットにタイトルを付けてレイアウト調整もして。

13. [中級] エラーバー付き折れ線

NumPyでデータを生成し、Matplotlibの`errorbar`でエラーバー付き折れ線を描きたい。
yの平均と標準偏差から計算するコードを示して。

14. [中級] ログスケールの比較

同じデータを線形軸とログ軸で比較する図を作成。
Matplotlibで左に通常plot、右に`semilogy`で配置してください。

15. [中級] CDF の自作

乱数データからCDFを作るコードを示して。
NumPyの`np.sort`と累積確率を使い、Matplotlibで折れ線描画して保存。

16. [中級] カラーバーと凡例の併用

散布図にカラーバーと凡例の両方を付けたい。
Matplotlibでカラーバーは値の連続、凡例は系列名で分けて。

17. [中級] グラフのスタイル統一

Matplotlibのスタイルを一括で整える方法を教えて。
フォント、線幅、配色を`rcParams`で設定する例が欲しい。

18. [中級] 画像の保存解像度調整

PNGの解像度を高くして保存する方法を示して。
Matplotlibで`figsize`と`dpi`を指定し、保存時の注意点も一言添えて。

19. [中級] 日本語ラベルの注意点

Matplotlibで日本語ラベルを表示する際の注意点と対策を教えて。
フォント設定の例も簡潔に示して。

20. [中級] 再生成スクリプトの拡張

`generate_graphs.py`に新しいプロット関数を追加したい。
関数追加の手順と、`main`への登録例を示して。

21. [中級] 統計分析: 回帰直線と信頼区間

散布図に単回帰直線と95%信頼区間を重ねたい。
`statsmodels`または`scipy`で回帰を行い、Matplotlibで描画する例を示して。

22. [中級] 統計分析: 残差プロット

回帰分析の残差プロットを作成するコードをください。
`statsmodels`または`numpy`で回帰し、Matplotlibで上段に散布図+回帰線、下段に残差を配置したい。

23. [中級] 統計分析: 相関行列の可視化

PandasのDataFrameから相関行列を計算し、
Pandasで相関を算出し、Matplotlibでヒートマップ表示する方法を示して。

24. [中級] 統計分析: 分布比較の重ね描き

2つのデータ群の分布を同じヒストグラムに重ねたい。
NumPyでデータを用意し、Matplotlibで透明度を付けて比較できるようにして。

25. [中級] 統計分析: 箱ひげ図と平均値の併記

複数群の箱ひげ図に平均値マーカーを重ねたい。
NumPyで平均を計算し、Matplotlibで平均を点で表示し、凡例も付けて。