ダウンロード・データあります

第2章 グラフを制するものは解析を制す！

お勧め2…可視化ライブラリ Matplotlib

辰岡 鉄郎

Matplotlibとは

● Pythonの定番可視化ライブラリ

データを人が理解しやすい形式で表現するデータの可視化（Visualization）は，データ解析の成功の鍵を握ると言っても過言ではありません．Matplotlibは，グラフ描画やアニメーション生成，インタラクティブなグラフ・ウィンドウ（グラフの拡大縮小や移動などがマウス操作で可能）などを提供する，Pythonのデータ可視化ライブラリです．

● グラフ機能のモジュール…pyplot

matplotlib.pyplotは，グラフ機能に関連するモジュールです．線のスタイル，ラベルのフォント，軸の設定などをきめ細かく制御でき，2D，3Dの多彩なグラフの種類，さまざまな形式での保存が可能です．seabornなどのライブラリを用いて，さらに洗練されたデザインのグラフを描画できます．

Matplotlibの基本

● Matplotlibの書式には2種類のスタイルがある

Matplotlibでグラフを描画するには，次の2つのスタイルがあります．

- Object-oriented（OO）スタイル（OO-style）

Figure，Axesオブジェクトを生成しax.plot()など，オブジェクトのメソッドで記述するスタイルです．

- pyplotスタイル（pyplot-style）

plt.plot()など，pyplot関数で記述するスタイルです．

● グラフ描画の具体例

具体例を見てみましょう．下記は，いずれも図1のグラフを描画するコードです．

- 共通（ライブラリのインポート，データ生成）

```
import matplotlib.pyplot as plt
import numpy as np  #データ生成用
x = np.linspace(0, 2*np.pi, 100)
```

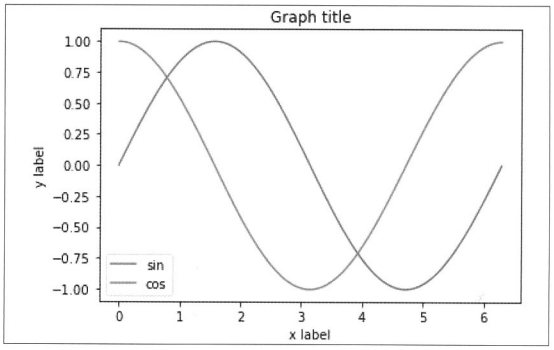


図1　グラフ描画の例

- OO-styleの例（ax.～()などの形）

```
fig, ax = plt.subplots()
                          # オブジェクトの生成
ax.plot(x, np.sin(x), label='sin')
                            # 正弦波の描画
ax.plot(x, np.cos(x), label='cos')
                              # 余弦波の描画
ax.set_xlabel('x label')
                          # x軸ラベルを表示
ax.set_ylabel('y label')
                          # y軸ラベルを表示
ax.set_title('Graph title')
                            # タイトルを表示
ax.legend()   # 凡例を表示
plt.show()
```

- pyplot-styleの例（plt.～()の形）

```
plt.figure()
plt.plot(x, np.sin(x), label='sin')
plt.plot(x, np.cos(x), label='cos')
plt.xlabel('x label')
plt.ylabel('y label')
plt.title('Graph title')
plt.legend()
plt.show()
```

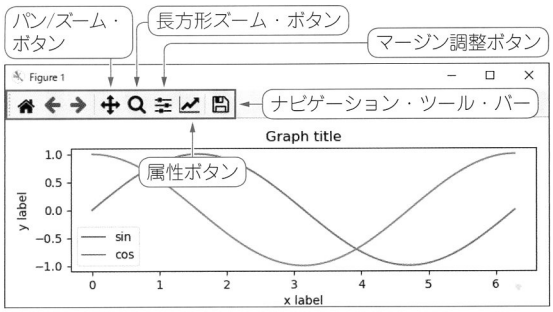


図2　Figureウィンドウ

軸の表示設定はAxesオブジェクトを参照するなど，細かな制御を行うにはOO-styleの方がなじみますが，pyplot-styleでもplt.gca()で途中からオブジェクトを取得することも可能なため，どちらを利用しても問題ありません．ライブラリのリファレンス・サイト(1)では，Jupyter Notebookなどで簡便に使うときはpyplot-style，再利用を意図した本格的なスクリプトや関数などではOO-styleが推奨されています．

● Matplotlibの用語

凡例をlegend，グラフ内の目盛線をgrid（グリッド）と呼ぶなど，関数名や引数名の多くは一般的な表現が使われていますが，表示領域と軸の用語は紛らわしいので整理しておきます．

- Figure（matplotlib.figure.Figureクラス）

図全体のオブジェクトです．1つ以上のグラフ・エリア（Axes）を持ちます．

- Axes（matplotlib.axes.Axesクラス）

直訳はaxis（軸）の複数形ですが，Matplotlibでは，グラフまたはプロット・エリアのオブジェクトです．

- Axis（matplotlib.axis.Axisクラス）

目盛と目盛ラベルを含む軸のオブジェクトです．

- Spine（matplotlib.spines.Spineクラス）

直訳では脊椎ですが，軸の枠線のことです．例として，グラフ・エリアの右の枠線を非表示にしたいときや，下の線を縦軸の0の位置に表示したいときは，Axesオブジェクトをaxとすると，

```
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_position('zero')
```

のように書くことができます．

● 大抵の折れ線グラフは虎の巻1つでOK

表1（稿末）の「Matplotlib 2Dグラフ虎の巻」は，主に折れ線グラフを作成する際に必要となる機能について一覧にまとめたものです．Matplotlibの機能は膨大なので，記載のない関数や引数は無数にありますが，よく使うと思われる項目については，取り得る設定値や初期値まで網羅しており，折れ線グラフの大抵の用途では，検索エンジンのお世話にならずに済むと思います．

● その他のグラフも関数名が分かれば検索が簡単

折れ線以外の2Dグラフについては，後半に関数名とサンプルのグラフを載せるに留めています．関数名だけでも分かれば，例えば，離散データのグラフで使われるplt.stem()関数を調べるのに「丸と棒のグラフ」などと調べてstem()にたどり着くステップが不要となり，すぐに使い方を調査することができます．ただし表にあるグラフは全てではなく，3Dグラフやアニメーションは，誌面の都合上全く触れられておりません．これらについては，他のリソースをご活用ください．

● 公式Cheatsheets（虎の巻）までの橋渡しに

Matplotlibには，公式のCheatsheets(2)も存在します．デザインも洗練されていて，一度理解した人には網羅性があり重宝するのですが，あまりに密度が濃い上，英語のみなので，情報に圧倒され初心者向きではないように感じます．for beginnersと書かれた資料も用意されていますが，こちらは逆に，簡素過ぎる印象を受けます．

本稿の虎の巻は，日本語で，一般的な機能を程良くまとめ，公式版を利用するまでの橋渡しとなる資料を目指しました．**表1**はExcelファイルとして本誌ウェブ・ページから入手できます．用途に応じて足りない部分はあるかもしれませんが，人は，よく使うグラフや機能は大体決まっています．不足分を書き込んだりしながら，自分専用の虎の巻を作ってみてください．

とっても便利なFigureウィンドウ

● ズームや移動が簡単だと解析がはかどる

図2は，Matplotlibでグラフを描画したときに表示されるFigureウィンドウです．画面上部に並んだアイコン（ナビゲーション・ツール・バー）から，ズームや移動などグラフを簡単に操作できます．データ・サイズが大きい場合など，拡大/縮小や移動機能は大変便利で，これだけでもMatplotlibを使う価値があります．

● ナビゲーション・ツール・バーのボタン

- ホーム/戻る/進むボタン

ホーム・ボタンで初期表示に戻る．
戻る/進むボタンで，1つ前/1つ後の操作に戻る．

- パン/ズーム・ボタン

グラフの移動（パン）と拡大/縮小（ズーム）．
左クリックで移動，右クリックでズーム．

matplotlib 2Dグラフ虎の巻（Ver. 3. 3. 4）

OO: Object-oriented style, PP: pyplot style
fig: Figureオブジェクト, ax: Axesオブジェクト

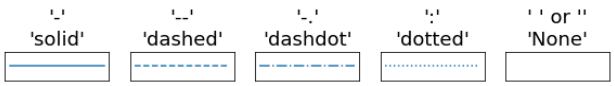
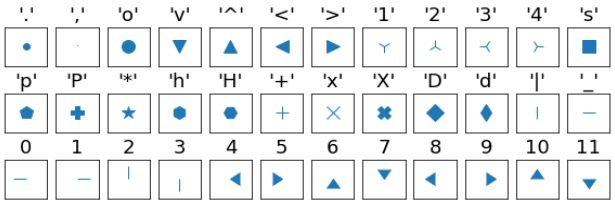
■ライブラリ

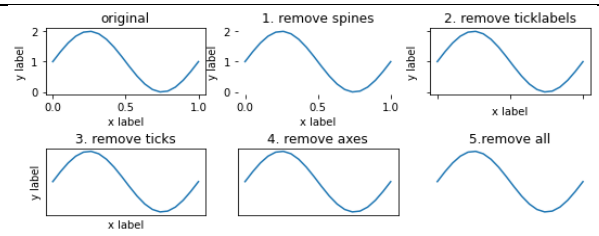
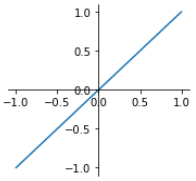
項目		スタイル / 書式または例	説明
ライブラリのインポート		import matplotlib.pyplot as plt	
Jupyter Notebookでの利用		%matplotlib inline	Jupyter Notebookで使用する際に必要

■グラフウィンドウ，画像ファイル出力

描画領域を生成	OO	fig = plt.figure()	グラフエリアの生成は「複数グラフの描画」の項参照
	PP	plt.figure()	
描画領域を生成（サイズ指定）		plt.figure(figsize=(width, height))	幅と高さをインチで指定（初期値: 6.4, 4.8，dpiの初期値: 100）
グラフエリアも同時に生成	OO	fig, ax = plt.subplots()	ウィンドウサイズ指定時は，fig, ax = plt.subplots(figsize=(width, height))
グラフエリアも同時に生成（複数）	OO	fig, axs = plt.subplots(2, 2)	2 x 2 の4つのグラフエリアを生成（「複数グラフの描画」の項参照）
Axesオブジェクトを取得	PP	ax = plt.gca()	axと置かずにplt.gca().tick_params()などの使い方も可
グラフの表示		plt.show()	
PNGファイル保存	OO	fig.savefig(filepath)	フォルダパスとファイル名の場合 os.path.join(dirname, filename) を利用
PNGファイル保存（dpi指定）	OO	fig.savefig(filepath, dpi=300)	plt.savefig()を使うとメモリが解放されないことがあるので fig.savefig()を使う
ファイル保存時のクローズ処理	OO	fig.clf(); plt.close()	メモリリーク防止のため fig.clf(), plt.close() を両方実行する

■折れ線グラフ

折れ線グラフ	OO	ax.plot([x,] y)	
	PP	plt.plot([x,] y)	
折れ線グラフ（フォーマット指定）		plt.plot([x,] y, 'o-r')	第2または第3引数として文字列でフォーマット指定（'[marker][line][color]'）
折れ線グラフ（オプション指定）		plt.plot([x,] y, option=value,…)	option=value の組み合わせは下記参照
ラベル		label='wave_label'	凡例表示の際に使用される（.legend()の中で指定することも可）
波形色		color='wave_color' （c='wave_color'） 例） ① color='C0' ② color='b' ③ color='navy' ④ color=(1,0,0) ⑤ color='#FF0000' ⑥ color='0.0' 補足）④，⑤では透明度も指定できる	①デフォルト色 'Cn'（C0～C9）  ②メジャー色 'x' (b, g, r, c, m, y, k, w)  ③登録色 'color name' (※1)  ④RGB指定 (R, G, B[, A])  ⑤RGB指定 '#RRGGBB[AA]'  ⑥グレースケール 'n.n' 
波形の透明度		alpha=0.5	アルファブレンド（0.0: 完全に透明 ～ 1.0:透明度ゼロ）
線の種類		linestyle='solid' （ls='solid'）	'-', '--', '-.', ':', ' ', '', 'solid', 'dashed', 'dashdot', 'dotted', 'None'より選択 
線幅		linewidth=1.5 （lw=1.5）	線幅のポイント数（初期値: 1.5）
マーカー		marker='o'	マーカー形状（' ', '', 'None': マーカーなし，',': ピクセル）（※2） 
マーカーサイズ		markersize=6 （ms=6）	マーカーサイズのポイント数（初期値: 6）
マーカー色		markerfacecolor='C0' （mfc='C0'）	指定方法は波形色と同様
マーカーの線幅		markeredgewidth=1.0 （mew=1.0）	マーカーの線幅のポイント数（初期値: 1.0）
マーカーの線色		markeredgecolor='C0' （mec='C0'）	指定方法は波形色と同様
zオーダー		zorder=1.0	値が小さい順に描画される（値が大きい方が前面に表示される）
グラフタイトル		plt.title('str', option=value,…)	OO-style: ax.set_title('str')
フォントサイズ		fontsize=16.0 （size=16.0）	フォントサイズを数値で指定または 'xx-small', 'x-small', 'small', 'medium', 'large', 'x-large', 'xx-large'（初期値: 'large'）
左寄せ，センタリング，右寄せ		loc='center'	文字列のアライメント（'center', 'left', 'right'）（初期値: 'center'）
オフセット		pad=6.0	グラフエリア上端からの距離をポイント数で指定（初期値: 6.0）
グラフ全体のタイトル	OO	fig.suptitle('str')	fontsize または font で文字サイズを指定可
	PP	plt.suptitle('str')	

軸の範囲	OO	x軸: ax.set_xlim(x1, x2) y軸: ax.set_ylim(x1, x2) 両軸: ax.axis([x1, x2, y1, y2])	axis()の引数はリストではなく ax.axis(xmin=x1, xmax=x2, ymin=y1, ymax=y2) としてもOK
	PP	x軸: plt.xlim(x1, x2) y軸: plt.ylim(y1, y2) 両軸: plt.axis([x1, x2, y1, y2])	
対数軸	OO	x軸: ax.xscale('log') y軸: ax.yscale('log')	ax.plot()の代わりに， x軸: ax.semilogx() y軸: ax.semilogy() 両対数: ax.loglog() もある
	PP	x軸: plt.xscale('log') y軸: plt.yscale('log')	
y/x アスペクト比	OO	ax.set_aspect(1.0)	アスペクト比1の場合は，aspect=1の代わりに'equal'を指定してもOK
	PP	plt.axis('scaled')	plt.axis()は'scaled','equal'でアスペクト比1になるが'equal'は長さも等しくなる
左右軸	OO	ax2 = ax.twinx()	twinx()で右側の軸を持つAxesオブジェクトを取得してax2に描画する
軸の非表示		下記参照	非表示になる項目は下図参照
1. 枠線の非表示	OO	_ = [s.set_visible(False) for s in ax.spines.values()]	
2. 目盛ラベルの非表示	OO	ax.axes.xaxis.set_ticklabels([]) ax.axes.yaxis.set_ticklabels([])	
3. 目盛の非表示	OO	ax.axes.xaxis.set_ticks([]) ax.axes.yaxis.set_ticks([])	
4. 枠線以外の非表示	OO	ax.axes.xaxis.set_visible(False) ax.axes.yaxis.set_visible(False)	
5. 全て非表示	OO	ax.axis('off')	
軸の位置	OO	ax.spines['left'].set_position(('data', 0)) spinesの[]内は， 'left', 'right', 'top', 'bottom' より選択	set_positionは，(タイプ, 量)のタプルまたは 'center': ('axes', 0.5)と等価 'zero': ('data', 0.0)と等価 タイプは下記より選択 'outward': データ領域からの距離をポイント数で指定（内側が負） 'axes': グラフエリアの座標軸（0〜1）で指定 'data': データの座標で指定 例） ax.spines['bottom'].set_position('zero') ax.spines['left'].set_position('zero') ax.spines['right'].set_visible(False) ax.spines['top'].set_visible(False) 
軸ラベル	OO	ax.set_xlabel('str') ax.set_ylabel('str')	引数 fontsize または font で文字サイズを指定可（初期値: 10.0） 引数 color で文字色を指定可（指定方法は波形色と同様）
	PP	plt.xlabel('str') plt.ylabel('str')	
目盛の値と目盛ラベル	OO	ax.set_xticklabels(ticks [, labels]) ax.set_yticklabels(ticks [, labels]) ax.set_xticks(ticks) ax.set_yticks(ticks)	ticks には，目盛を振る位置を指定（例: [1, 2, 3] ），空のリスト [] は目盛無し r分割: np.linspace(min, max, r+1)，d間隔: np.arange(min, max+d, d) labels には，目盛位置に表示する文字列を指定（ 例: ['1月', '2月', '3月'] ） 引数 fontsize または font で文字サイズを指定可（初期値: 10.0）
	PP	plt.xticks(ticks [, labels]) plt.yticks(ticks [, labels])	引数 color で文字色を指定可（指定方法は波形色と同様） 引数 rotation=45（角度），ha='right'（右端を目盛位置）で回転も可
目盛の属性	OO	ax.tick_params(option=value,…)	
対象軸		axis='x'	'x', 'y', 'both' より選択（初期値: 'both'）
対象目盛（主目盛，補助目盛）		which='major'	'major'（主目盛），'minor'（補助目盛），'both'より選択（初期値: 'major'）
目盛線の向き		direction='out'	'in'（内側），'out'（外側），'inout'（両側）より選択（初期値: 'out'）
目盛線の長さ		length=3.5	目盛線の長さをポイント数で指定（初期値: major: 3.5, minor: 2）
目盛線の太さ		width=0.8	目盛線の太さをポイント数で指定（初期値: major: 0.8, minor: 0.6）
目盛線の色		color='k'	指定方法は波形色と同様（初期値: 'black'）
目盛線から目盛ラベルまでの距離		pad=3.5	目盛ラベルまでの距離をポイント数で指定（初期値: major: 3.5, minor: 3.4）
フォントサイズ		labelsize='medium'	指定方法はグラフタイトルと同様（初期値: 'medium'）
目盛ラベルの色		labelcolor='black'	指定方法は波形色と同様（初期値: 'black'）
目盛ラベルの回転		labelrotation=0	目盛ラベルの角度を度数で指定（初期値: 0）
グリッド線	OO	ax.grid(option=value,…)	線の属性はLine2Dの設定が利用できる（linestyleまたはls，linewidthまたはlw，colorまたはcなど）
	PP	plt.grid(option=value,…)	
対象軸		axis='x'	'x', 'y', 'both' より選択（初期値: 'both'）
対象目盛（主目盛，補助目盛）		which='major'	'major'（主目盛），'minor'（補助目盛），'both'より選択（初期値: 'major'）
凡例	OO	ax.legend(option=value,…)	引数 labels に凡例の文字列リストを指定しても良いし，plot()でlabelを指定していれば不要
	PP	plt.legend(option=value,…)	
表示位置		loc='best'	'best':0, 'upper right':1, 'upper left':2, 'lower left':3, 'lower right':4, 'right':5, 'center left':6, 'center right':7, 'lower center':8, 'upper center':9, 'center':10 より文字列または番号で指定（初期値: 'best'） 左下隅の座標を2値のタプルで指定することも可（左下(0,0), 右上(1,1)） 負または1より大きい値を指定するとグラフエリア外にも表示できる
表示位置の座標指定		bbox_to_anchor=(0.0, 0.0)	locの座標指定は左下隅固定だがbbox_to_anchorを指定するとlocは凡例のどこを基準とするかを示しbbox_to_anchorで座標指定できる
凡例表示の列数		ncol=1	表示の列数を整数で指定（初期値: 1）
背景の透明度		framealpha=0.8	背景のアルファブレンド（初期値: 0.8）
表示マーカー数		numpoints=1	表示されるマーカーの数（初期値: 1）
凡例の枠内のマージン		borderpad=0.4	凡例内のマージンをフォントサイズ単位で指定（初期値: 0.4）
凡例の軸からのオフセット		borderaxespad=0.5	軸からの距離をフォントサイズ単位で指定（初期値: 0.5）
余白		plt.subplots_adjust(left=0.1, bottom=0.1, right=0.9, top=0.9)	グラフエリアの範囲を，左下を(0, 0)，右上を(1, 1)としたときの座標で指定
グラフエリアの背景色	OO	ax.set_facecolor('c')	色の指定は波形色と同様

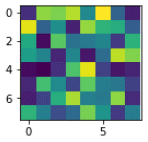
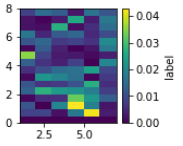
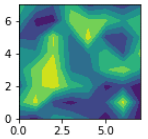
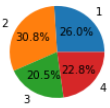
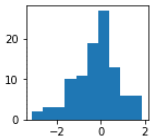
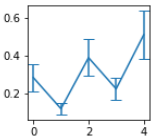
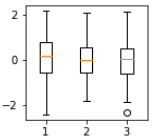
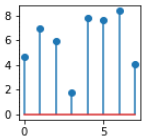
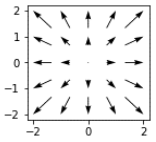
塗りつぶし	OO	ax.fill_between(x, y [, y2], option=value…)	関数を1つ指定した場合はx軸との間を塗りつぶし， 関数を2つ指定した場合は挟まれた区間を塗りつぶす 例） u = np.zeros(len(y1)) u[y2>y1]=y1[y2>y1]; u[y2<=y1]=y2[y2<=y1] ax.fill_between(x, u, fc='C2', alpha=0.5) ax.fill_between(x, y1, y2, where=(y2 > y1), fc='C0', alpha=0.5) ax.fill_between(x, y1, y2, where=(y2 <= y1), fc='C1', alpha=0.5)
	PP	plt.fill_between(x, y [, y2], option=value…)	
水平線の描画	OO	ax.axhline(y=y1, xmin=x1, xmax=x2, option=value…) ax.hlines([y1, …], xmin=x1, xmax=x2, option=value…)	ax.axhlineはxminとyminを省略すると画面全体に線が引かれる この範囲はグラフの表示範囲を変えてもグラフ領域の相対位置に保たれる hlinesは同じ範囲の水平線を複数本同時に描画できる グラフの表示範囲を変えても指定した座標の位置に保たれる
垂直線の描画	OO	ax.axvline(x=x1, ymin=y1, ymax=y2, option=value…) ax.vlines([x1, …], ymin=y1, ymax=y2, option=value…)	水平線と同様 （誌面の都合上省略しているが，水平線，垂直線ともに， PP（pyplot-style）では，「ax.」を「plt.」に置き換える）
文字の描画	OO	ax.text(x, y, str, option=value,…)	座標(x, y)に文字列strを描画する（引数に transform=ax.transAxes を付けると左下を(0, 0)，右上を(1.0, 1.0)としたグラフエリアでの相対座標で指定可） 数値valを小数点3桁で表示するときは 'test {:.3f}'.format(val)のように書く
	PP	plt.text(x, y, str, option=value,…)	
フォントサイズ		fontsize=16.0 （size=16.0）	フォントサイズを数値で指定または 'xx-small', 'x-small', 'small', 'medium', 'large', 'x-large', 'xx-large'（初期値: 'medium'）
横方向のアライメント		horizontalalignment='left' （ha='left'）	'center', 'left', 'right'より選択（初期値: 'left'） 'left'であれば文字列の左端が指定座標の位置に来るように表示される
縦方向のアライメント		verticalalignment='baseline' （va='baseline'）	'center', 'top', 'bottom', 'baseline', 'center_baseline'より選択（初期値: 'baseline'）'baseline'は文字の下端で'bottom'はマージンあり
テキストの角度		rotation=0.0	角度を度数または'vertical', 'horizontal'で指定（初期値: 0）
矢印の描画	OO	ax.arrow(x, y, dx, dy, option=value,…)	始点(x, y)から大きさ(dx, dy)の矢印を描画する 色の指定は， 単色の場合: color='b' とすれば良く，枠線と中身の色を分ける場合は， 線色: edgecolor(or ec)='r'，塗りつぶし色: facecolor(or fc)='y' のように書く
	PP	plt.arrow(x, y, dx, dy, option=value,…)	
線幅		width=0.1	矢印の線の太さを座標スケールで指定（初期値: 0.001）
矢尻の幅		head_width=0.3	矢尻の幅を座標スケールで指定（初期値: 3*width）
矢尻の長さ		head_length=0.3	矢尻の長さを座標スケールで指定（初期値: 1.5*head_width）
終点を矢尻の先端とするか否か		length_includes_head=True	矢印の大きさに矢尻の長さを含めるか（初期値: False）

■複数グラフの描画

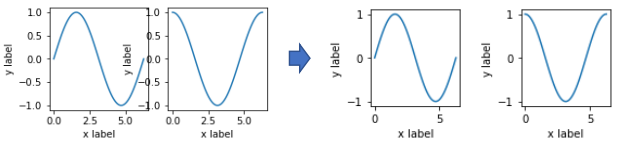
グラフエリアの生成	OO	fig, axs = plt.subplots(r, c) ax = fig.add_subplot(rcn) ax = fig.add_subplot(r, c, n)	r行 c列のn番目（左から右，上から下の順で1から開始）のグラフエリアを生成 fig, axs = plt.subplots(r, c)の場合は axs[m, n]でアクセス（rまたはcが1の場合は axs[n]） fig, (ax1, ax2) = plt.subplots(2, 1)としてもOK また，add_subplotでは軸の属性なども設定可 例） ax1 = fig.add_subplot(2, 1, 1) ax2 = fig.add_subplot(2, 2, 3) ax3 = fig.add_subplot(2, 2, 4) plt.tight_layout()
	PP	plt.subplot(rcn) plt.subplot(r, c, n)	
グリッド状レイアウト	OO	ax = plt.subplot2grid((r, c), (y, x), rowspan=m, colspan=n)	r行 c列の(y, x)の位置にサイズ(m, n)のグラフエリアを生成 例）上図と同じレイアウトを生成 ax1 = plt.subplot2grid((2, 2), (0, 0), colspan=2) ax2 = plt.subplot2grid((2, 2), (1, 0)) ax3 = plt.subplot2grid((2, 2), (1, 1)) plt.tight_layout()
	PP	plt.subplot2grid((r, c), (y, x), rowspan=m, colspan=n)	
座標指定	OO	ax = fig.add_axes([x, y, w, h]) 補足）座標(x, y)，大きさ(w, h)は描画領域の左下を(0, 0)，右上を(1, 1)とした座標系で指定する	グラフの左下を座標(x, y)として大きさ(w, h)のグラフエリアを生成 グラフの中に埋め込むこともできる 例） fig, ax = plt.subplots(figsize=(3,3)) ax.plot(t, dat); ax.set_ylim(-1.2, 3.2) ax1 = fig.add_axes([0.3, 0.6, 0.2, 0.2]) ax1.plot(t[0:25], dat[0:25]) ax2 = fig.add_axes([0.5, 0.6, 0.2, 0.2]) ax2.plot(t[100:125], dat[100:125]) ax2.tick_params(left=False, right=True, labelleft=False, labelright=True)
グラフエリア間の間隔		plt.subplots_adjust(wspace=0.2, hspace=0.2)	wspace: 左右の間隔（初期値: 0.2） hspace: 上下の間隔（初期値: 0.2）

■その他のグラフ

散布図	OO	ax.scatter(x, y, option=value,…)	主なオプション s: マーカーサイズ c: マーカー色 (color) marker: マーカー形状 label: ラベル zorder: zオーダー
	PP	plt.scatter(x, y, option=value,…)	
棒グラフ	OO	ax.bar(x, y, option=value,…)	主なオプション width: 棒の幅（初期値: 0.8） c: 棒の色 (color)，ec: 棒の枠線の色 (edgecolor) yerr: y軸方向のエラーバー ecolor: エラーバーの線色 capsize: エラーバー両端の線の長さをポイント数で指定
	PP	plt.bar(x, y, option=value,…)	

カラーマップ（画像）	OO	ax.imshow(z, option=value,…)	0.0-1.0または0-255の(M, N)2次元配列または， (M, N, 3)のRGBまたは(M, N, 4)のRGBAを引数に取る 主なオプション cmap: カラーマップ（初期値: 'viridis'）（※3） norm: 正規化オブジェクト interpolation: 補間法 
	PP	plt.imshow(z, option=value,…)	
カラーマップとカラーバー	OO	c = ax.pcolormesh(x, y, z, option=value,…) fig.colorbar(c, ax=ax)	pcolormesh(x, y, z)の形で軸のスケールを指定できる ピクセルのアスペクト比が1であればimshow()の方が高速 バーのラベルは .colorbar(c, ax=ax).set_label('label') 主なオプション vmax, vmin: カラーマップの値の範囲 shading: 塗りつぶしスタイル（初期値: 'flat'） 
	PP	plt.pcolormesh(x, y, z, option=value,…) plt.colorbar()	
等高線	OO	ax.contour(z, option=value,…) ax.contourf(z, option=value,…)	contour()は等高線 contourf()は線間を塗りつぶした色付きの等高線（右図） いずれもcontour(x, y, z)の形式で座標指定もできる contourの戻り値を引数にclabel()を実行してラベル表示も可 主なオプション levels: 等高線の段数 
	PP	plt.contour(z, option=value,…) plt.contourf(z, option=value,…)	
円グラフ	OO	ax.pie(x, option=value,…)	主なオプション explode: 外側に切り離して表示する場合のオフセット値を指定 labels: 各要素のラベル colors: 各要素の色 shadow: 影表示 startangle: 開始点の角度 
	PP	plt.pie(x, option=value,…)	
ヒストグラム	OO	ax.hist(x, option=value,…)	主なオプション bins: ヒストグラムの分割数 range: ヒストグラムの範囲（最小値，最大値を指定） rwidth: 棒の幅 color: 棒の色 label: ラベル 
	PP	plt.hist(x, option=value,…)	
エラーバー	OO	ax.errorbar(x, y, yerr, option=value,…)	errorbar(x, y, yerr, xerr)の形式でx方向のエラーバーも可 主なオプション fmt: 線の書式 elinewidth: エラーバーの線幅 ecolor: エラーバーの線色 capsize: エラーバー両端の線の長さをポイント数で指定 
	PP	plt.errorbar(x, y, yerr, option=value,…)	
箱ひげ図	OO	ax.boxplot(z, option=value,…)	主なオプション labels: ラベル notch: 箱の中央を凹ませる vert: Falseで横方向の箱ひげ図 flierprops: 外れ値のフォーマットを指定 showfliers: Falseで外れ値を非表示 
	PP	plt.boxplot(z, option=value,…)	
離散データ	OO	ax.stem(x, y, option=value,…)	主なオプション linefmt: 縦線のフォーマットを指定 markerfmt: マーカーのフォーマットを指定 basefmt: 基線のフォーマットを指定 label: ラベル use_line_collection: 警告が出る場合Trueに設定 
	PP	plt.stem(x, y, option=value,…)	
ベクトル場	OO	ax.quiver(u, v, option=value,…)	quiver(x, y, u, v)の形式で(x, y)座標の指定も可 主なオプション pivot: 矢印の配置方法（矢印の始点，終点，中央など） headwidth: 矢印の幅 headlength: 矢印の斜め線の長さ headaxislength: 矢印の軸の交点までの長さ 
	PP	plt.quiver(u, v, option=value,…)	

■レイアウト・デザインの調整

グラフマージンの自動調整	OO	fig.tight_layout()	グラフが重なってしまう場合や余っている場合，グラフの間隔を調整する （suptitleとtitleが重なる場合は，tight_layout(rect=[0,0,1,0.96])とする） 
	PP	plt.tight_layout()	
seabornによる自動スタイル設定		plt.style.use('seaborn-white')	seabornライブラリがインストールされている場合は，plt.style.use()などでseaborn のスタイルを利用すると洗練されたデザインのグラフが描画できる
スタイルシートの一時的な変更		plt.rcParams['font.size']= 18.0	plt.rcParams[]でスタイルシートの値を一時的に変更できる 左のコードではfont.sizeのデフォルト値10を18に設定している matplotlibrcファイルの場所は下記のコードにより確認できる import matplotlib as mpl print(mpl.matplotlib_fname())

■Tips

パラメータの一括設定	OO	ax.set(option=value,…)	ax.set_title('title_str')ではなく，ax.set(title='title_str',…)で一括で指定する方法 もある
	PP	plt.setp(obj, option=value,…)	
ギリシャ文字		π	ラベルなどの文字列中にギリシャ文字を含めるには$\…$を用いる

(※1) matplotlib.colors	色見本	https://matplotlib.org/stable/gallery/color/named_colors.html
(※2) matplotlib.markers	マーカー	https://matplotlib.org/stable/api/markers_api.html
(※3) Colormap reference	カラーマップ	https://matplotlib.org/stable/gallery/color/colormap_reference.html