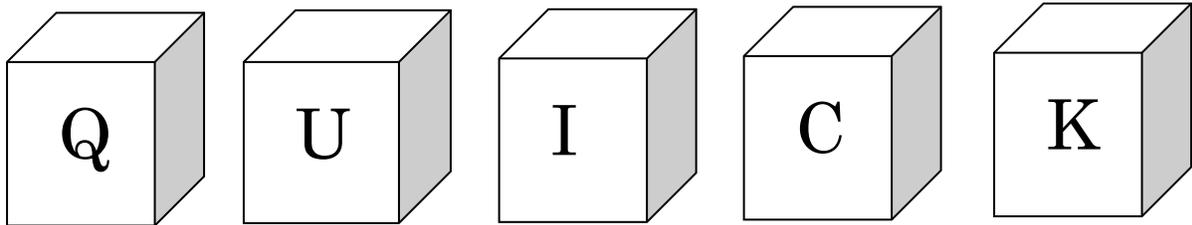


Quick Check

▶ Grammer & Usage



>> Python で学び直す数学

【関数とグラフ／微分と積分編】

～ matplotlib を使ってグラフを描画してみよう

仕事や趣味を通して Python のコードを書いている方であれば、「Jupyter Notebook を使ってグラフを描画」といってピンとくる方も多いと思いますが、実際に興味はあるけれど、どう使ってみればよいのかわからないという方も多いと思います。Python のライブラリの基本的な書き方を含め、学生時代に習った数学の問題を通して、グラフを描画するという形で演習をしていきたいと思います。

「数学的な問題を Python で簡単なスクリプトを作って動作を確認する」ことを通して、Python に触れる機会をつくっていきたいと考えています。Python に慣れるという点でも手を動かして考える機会にして頂ければ幸いです。

今回は、Python で学び直す数学【関数とグラフ／微分と積分編】の確認をしていきます。

○ グラフ描画～matplotlib の使い方

XX

数学の授業で、方程式を習った際に、方眼紙を用いて作図をした、という方もいらっしゃるかもしれません。Python では「matplotlib」というライブラリでグラフや図形を描画することができます。

実際に、今回のテーマである「関数とグラフ／微分積分」について、以下の例題の Python のコードを考えていきます。

- ・ 直交する直線
- ・ 連立方程式／2直線の交点
- ・ 関数と導関数
- ・ 接線
- ・ 定積分の計算

それでは、「Mapplotlib でグラフを描画」する方法からみていきましょう。

○ Matplotlib とは？

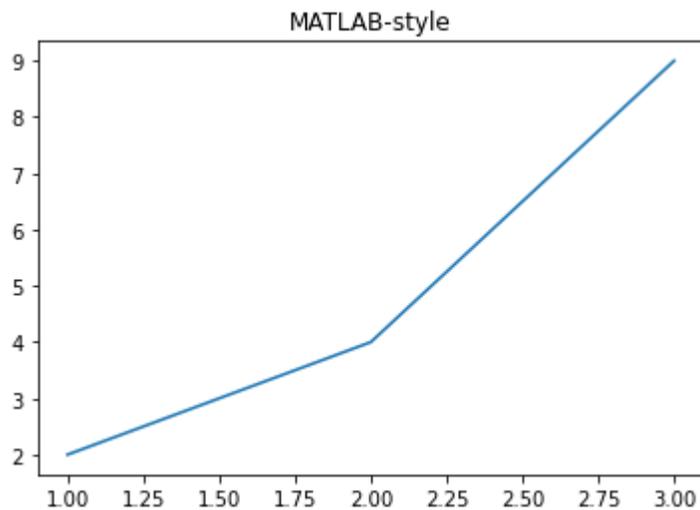
Python で主に2次元のグラフを描画するためのライブラリのこと。

Matplotlib を利用するには、以下のようにインポートを行います。As キーワードを使用して plt で呼び出せるようにします。

In

```
import matplotlib.pyplot as plt
```

まずは折れ線グラフの書き方から



上図の折れ線グラフは以下のコードを記述することで表現できます。

In

```
# データを用意
x = [1,2,3]
y = [2,4,9]

plt.plot(x,y)          # 折れ線グラフを描画
plt.title('MATLAB-style') # グラフにタイトルを設定

plt.show()            # グラフを表示
```

x 座標と y 座標を与えて `plot` 関数に引数として渡すことで、棒グラフを描画することができるのですね。

(x, y)が1つあれば点を描画でき、もう1つ(x2, y2)あれば、2点を結ぶ直線ができ、細かく座標を取っていけば、関数のグラフも描画できる、といった具合に応用して考えることができます。

○ 一次方程式

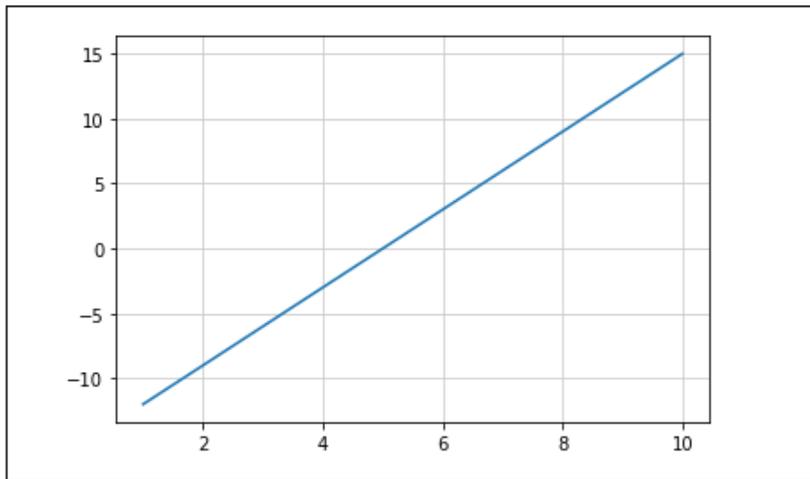
以下のような一次方程式のグラフを描画する例を考えてみましょう。

問題

matplotlib を使って、1 次方程式

$y = 3x - 15$ のグラフを描画してください。

ここでは、 x を 1~10 の整数値で取得し、対応する y の値でグラフを描画するものとする



x を 1~10 の整数値で取得する方法として、`range` 関数を利用することで開始値から終了までの連続した値を要素を生成します。

In

```
>>> range(5)          # range(stop)
0 1 2 3 4 5

>>> range(1,11)       # range(start, stop)
1 2 3 4 5 6 7 8 9 10

>>> range(1,11, 2)    # range(start, stop, step)
1 3 5 7 9
```

$y = 3x - 15$ のグラフ を描くには、上のように x の値 (リスト) を取得し、それに対応する y の値 (リスト) を `plot` 関数に引数として渡します。

```
%matplotlib inline
import matplotlib.pyplot as plt
# データ
x = list(range(1, 11)) # x の値 (1~10)
y = []
for i in range(10):
    y.append(3*x[i] - 15) # y= 3x-15
# グラフ
plt.plot(x, y)
plt.grid(color='0.8')
plt.show()
```

また、配列や行列を効率よく扱うことができるパッケージである「**numpy**」をインポートしてから利用することで、より簡易的に直線の方程式を「**y**に**x**を代入する式」として記述することができます。

xを1~10の整数値で取得する方法として、**arange**関数を利用することで開始値から終了までの連続した値を要素を生成します。

In

```
>>>import numpy as np

>>> np.arange(5)           # arange(stop)
0 1 2 3 4 5

>>> np.arange(1,11)       # arange(start, stop)
1 2 3 4 5 6 7 8 9 10

>>> np.arange(1,11, 2)    # arange(start, stop, step)
1 3 5 7 9
```

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np

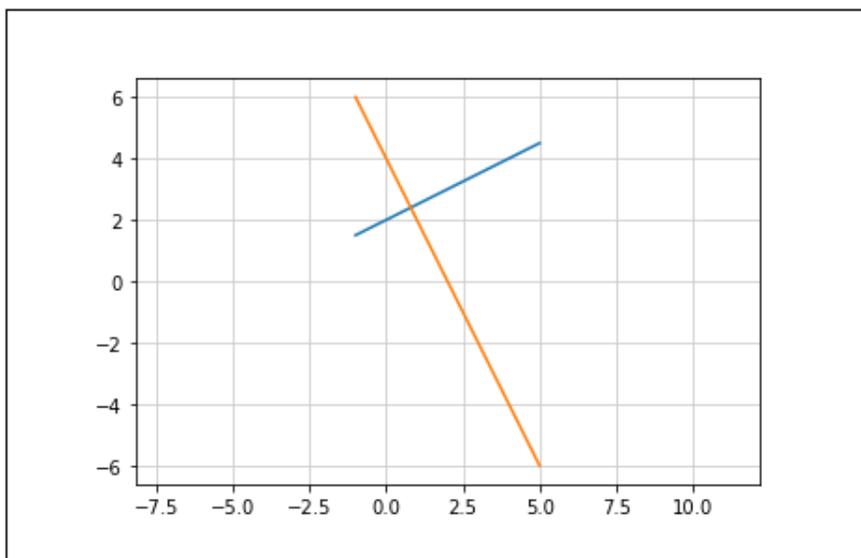
# データ
x = np.arange(1, 11) # x の値 (1~10)
y = 3*x + 15

# グラフ
plt.plot(x, y)
plt.grid(color='0.8')
plt.show()
```

⇒ ここで類題として、添付資料「直交する直線」シートにある演習問題を解いてみましょう

問題

matplotlibを使って、点(-1, 6)を通過して
直線 $y = 1/2 x + 2$ と直交する直線を描画してください。
ここでは、 x を-1~5の整数値で取得し、対応する y の値でグラフを描画するものとする



○ 連立方程式

Python のライブラリを使った連立方程式の解き方をみていきましょう。

具体的には、SymPy の中から Symbol クラスのオブジェクトを生成して、solve 関数を実行して求めます。例をみていきましょう。

問題

SymPy の中から、Symbol クラスを使って solve 関数をインポート(*)した上で次の連立方程式を解いてください。

$$\begin{aligned} a + b &= 3 \\ 5a + 2b &= 12 \end{aligned}$$

(*) にてインポート

```
from sympy import Symbol, solve
```

```
from sympy import Symbol, solve
# 式を定義
a = Symbol('a')      -①
b = Symbol('b')      -②
ex1 = a + b - 3
ex2 = 5*a + 2*b - 12
# 連立方程式を解く
print(solve((ex1, ex2)))
```

⇒ ここで類題として、添付資料「2直線の交点」シートにある演習問題を解いてみましょう

■ 2直線の交点

問題

SymPyの中から、Symbolクラスを使ってsolve関数をインポート(*)した上で次の2直線の交点(連立方程式の解)を求めてください。

$$y = -3x + 9$$

$$y = 1/2x + 2$$

(*)にてインポート

```
from sympy import Symbol, solve
```

matplotlibを使って、関数とグラフを描画する方法とPythonのライブラリを利用して直線の交点を求める方法について紹介しました。

これまでの知識をもとに、方程式のグラフを描画する以下の演習を解いてみましょう。

○ 関数と導関数

■ 関数と導関数のグラフ

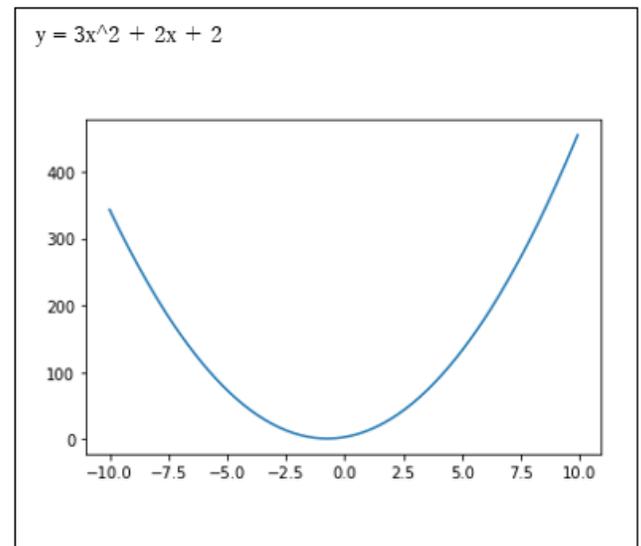
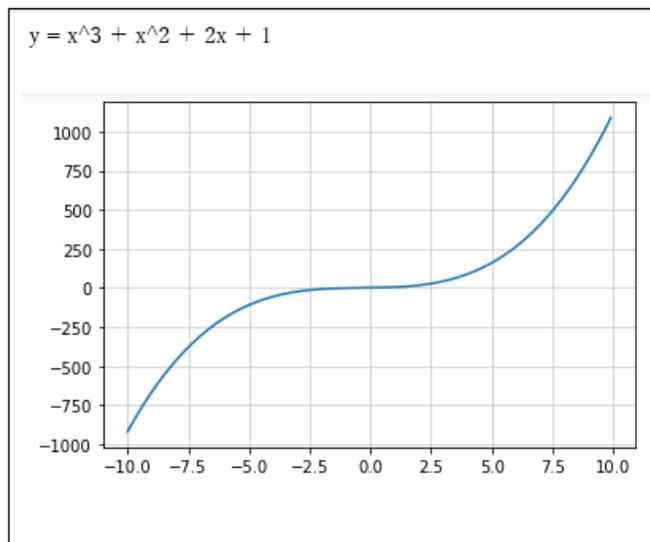
問題

matplotlibを使って、

$$y = x^3 + x^2 + 2x + 1$$

導関数 $y = 3x^2 + 2x + 2$ のグラフをそれぞれ描画してください。

ここでは、 x を-10~10で0.1の値間隔で取得し、対応する y の値でグラフを描画するものとする

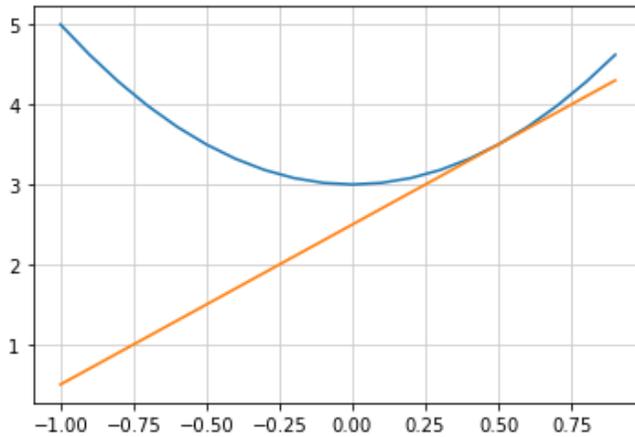


○ 接線を描画

問題

matplotlibを使って、
 $y = 2x^2 + 3$ の(0.5, 3.5)における接線を描画してください。

ここでは、 x を-1~1で0.1の値間隔で取得し、対応する y の値でグラフを描画するものとする



○ 定積分

問題

$y = x^2 + 3x + 2$ の方程式の
 x の値が-3から3までの面積を求めてください。

次のように積分の関数を定義して考えてみましょう。

```
>>> def F(x):  
    return 1/3 *x**3 + 3/ 2* x**2 + 2* x
```

=====
以前学校で学んできた内容をもとに Python でスクリプトを実行しながら確認できるのは面白いなど感じる方もいらっしゃるかもしれませんが、自分にできる範囲のものから少しずつ Python にも挑戦してみようかなと思っていただければ幸いです。

以上となります。

参考文献：

- ・谷尻かおり『文系プログラマーのための Python で学び直す高校数学』日経 BP 社（2021 年）