

実験で使える！

Pythonを用いたデータ処理

～グラフや表を秒で作成してYouTubeを見る時間を増やそう！～

初学者向け（Pythonなにそれ！？って人歓迎）

7月2日（日）21:00～開講

時間：2時間程度の予定



~CONTENTS~

1

Numpy Pandasを駆使
して、
爆速でデータの計算を
する

2

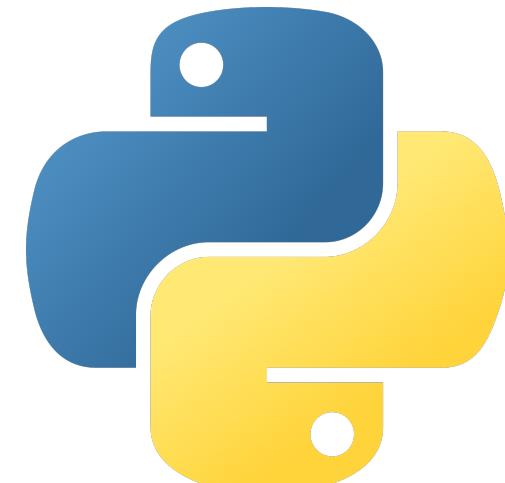
最小二乗法や不確かさ
の計算を関数化して、
一瞬で呼び出せる
ようにする

3

MatPlotLibを使って、
Tikzで使える
グラフを作成する

Pythonとはなんぞや？

- ・インタプリタ言語（比較的遅い）
- ・スクリプト言語（簡単でとっつきやすい）
- ・豊富なLibraries
- ・科学計算に強い（モノづくりも出来る）
- ・Data Scienceや機械学習にも
(Pandas, Numpy, OpenCV, Tensorflow etc...)



環境構築は済ませましたか？

- ・済んでいる人→ご自身の好きな環境でどうぞ
- ・済んでない人→ Google Colaboratoryを使おう

<https://colab.research.google.com/?hl=ja>

学び方

- ・プログラミング言語の学び方はとにかく「覚える」こと
→ 「調べる」こと！！！！
- ・「これ出来たら便利だな」は調べたら大抵やり方が載ってる
- ・「プログラマの精神」を理解する（30分かかる作業を3秒で終わらせるために3時間かけてコードを書く）
→ 焦らない / 使い回しが出来るように書く

おすすめのweb pages

- 公式チュートリアル

<https://docs.python.org/ja/3/tutorial/>

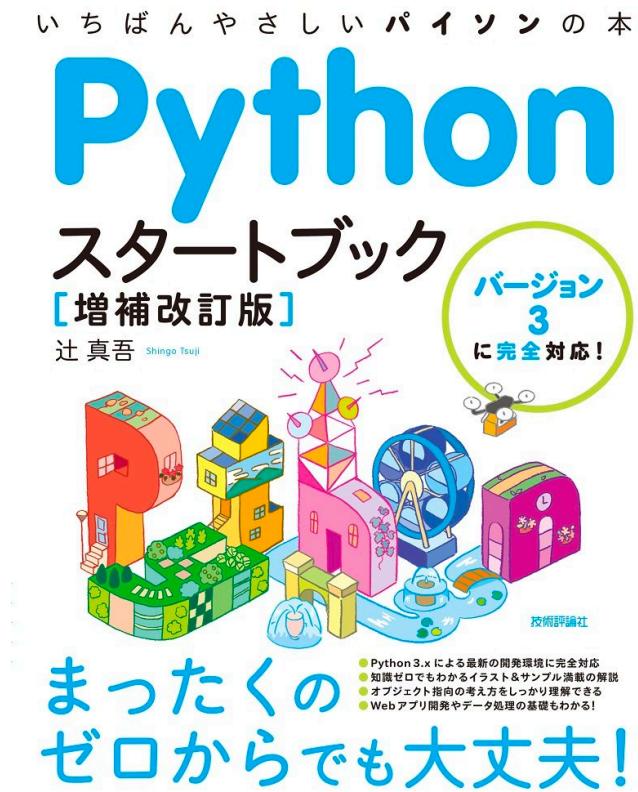
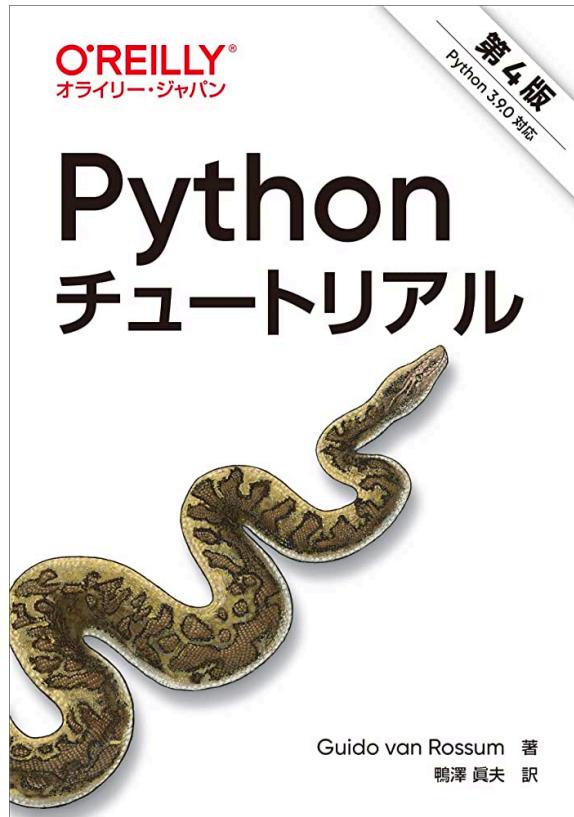
- Python japan (公式ではない) (こっちの方が易しいかも?)

<https://www.python.jp/train/index.html>

本で学ぶということ

- ・プログラミングの「基礎的な部分」は大して変わらないので、1冊くらいは本を持っていてもいいかも知れない
- ・結局はネットが**最強**.....
- ・O'Reillyは良書の宝庫（個人の意見です）

おすすめの本（文法書）



おすすめの本その2



Pythonの基本構文を学ぼう



Hello World

- programming界の「ごあいさつ」
- 文字列の出力

```
print("Hello World! !!")
```

参考までに.....

- C言語とJavaのHello World

```
#include <stdio.h>

int main(){
    printf("Hello
World!!!");
    return 0;
}
```

```
class Test {

    public static void main(String[] args) {
        System.out.println("Hello World!!!");
    }
}
```

1. 変数と演算

- 数学で使う $x, y \dots$ と同じ
- C言語やJavaだと「型」という概念が堅い（Pythonは緩い）
- 覚えるべき型：
整数→int, 小数→float, 文字列→str, 真偽値→bool
- 覚えるべき演算子：
 $+$ →足し算 $-$ →引き算 $*$ →掛け算 $/$ →割り算 $**$ →累乗
 $\%$ →あまり

1. 変数

- 試してみよう

```
a = 100
b = 40.5
c = "a"

# aを出力
print(a)

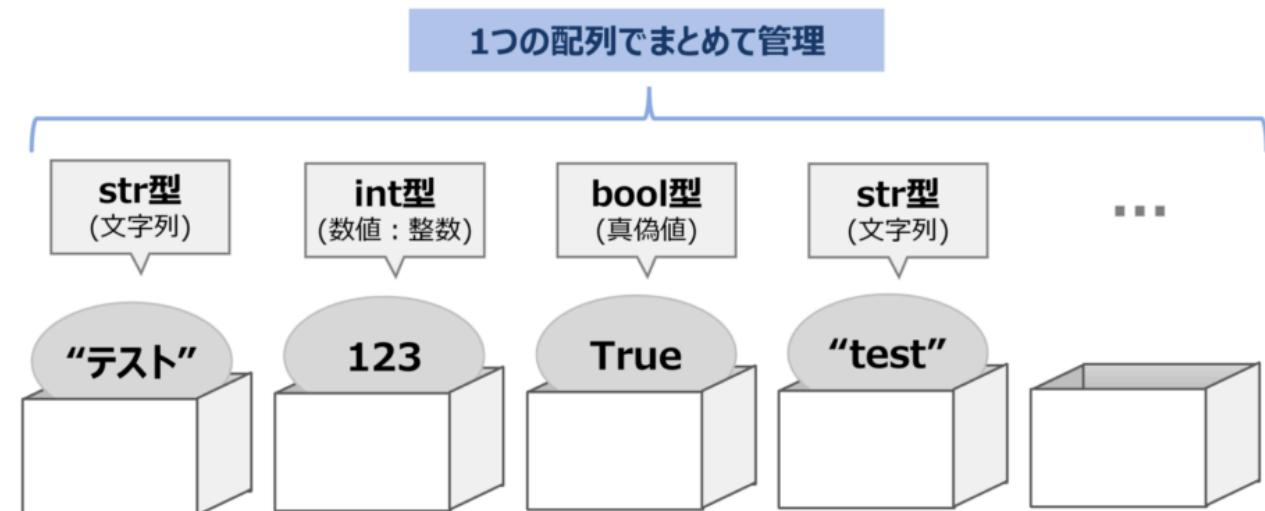
# 文字列とくっつける
print(f"aは{a}, bは{b}です")

# 割り算をやってみよう
print(a/b)

# 型を調べる
print(type(a))
print(type(b))
print(type(c))
```

2. 配列

- 数たちを入れる箱みたいなイメージ



2. 配列

- 試してみよう

```
# 普通の配列
a = [1,2,3,4,5,6]
print(f"aは {a}")
print(f"1番目の要素は {a[1]}")

# 二次元配列
b = [[1,2,3],[4,5,6]]
print(b)

# 足し算 (引き算, 掛け算, 割り算はない)
print(a + b)

# 連想配列
x = {"a":123, "b":345}
print(x["a"])
```

3. if文 (if, elif, else)

- 条件分岐
- 「もしも～なら」
- indent (press **Tab** or **Space**) を忘れずに
- elifやelseは別になくてもOK

3. if文 (if, elif, else)

- 試してみよう

```
x = int(input("入力してください"))

# if文による分岐
if(x == 1):
    print("Hello World!!!")
elif(x == 2):
    print("こんにちは！！！")
else:
    print("さようなら")
```

4. while文

- 条件分岐
- 「～な間」
- indent (press **Tab** or **Space**) を忘れずに
- 無限ループに注意！！！

4. while文

- 試してみよう

```
stop = False  
  
while(not stop) :  
    x = input("やめますか？")  
    if(x == "a"):  
        stop = True
```

5. for文

- ある作業を順番に行いたいときに使う
- indent (press **Tab** or **Space**) を忘れずに
- めっちゃ使うのでいろいろな書き方がある

for文

- 試してみよう

```
a = [3,1,4,5,2]
# 初期値

sum = 0
# aの要素すべてをsumに足し続ける

for i in a:
    sum += i
print(sum)

sum = 0
for i in range(10):
    sum += i
print(sum)
```

6. 関数

- 処理をまとめたいときにすごく便利
- 「return」で値を返す（返せる値は1つだけ！）
- 1本でだらだら書かずに、こまめに関数にまとめよう

試してみよう！

- 最小二乗法を関数で書いてみよう.

$$D = N \sum x_i^2 - \left(\sum x_i \right)^2$$

$$D_1 = \sum y_i \sum x_i^2 - \sum x_i \sum x_i y_i$$

$$D_2 = N \sum x_i y_i - \sum x_i \sum y_i$$

のとき,

$$y = \frac{D_1}{D} + \frac{D_2}{D} x$$

7. module (外部Library)

- Pythonの最大の強み (豊富なLibrary)
- 構造から理解して使うには「クラス」について学ぶとよい
- まずは他人の書いたコードのコピペからはじめよう

データの整理をしよう



csvファイルとは

- Comma-Separated Value (カンマで区切られた値)
- 最も簡単なデータ整理の方法
- データサイエンスではよく使われる
- Excelで編集可能 (拡張子をxlsx→csvにする)

numpy, pandasとは

- ・データ整理を使うライブラリ
- ・標準では入っていないので、pip installが必要