

# 情報処理概論

九州大学 工学部地球環境工学科 講義資料 担当:木村

## 07 グラフ描画／数値計算ライブラリ



九州大学  
KYUSHU UNIVERSITY

### 1) グラフ描画ライブラリ matplotlib

Pythonでグラフを描画したり、イメージを表示させたりする際に使用するライブラリ  
散布図・折れ線グラフ・棒グラフ・円グラフ・等高線図・3Dグラフ表示など

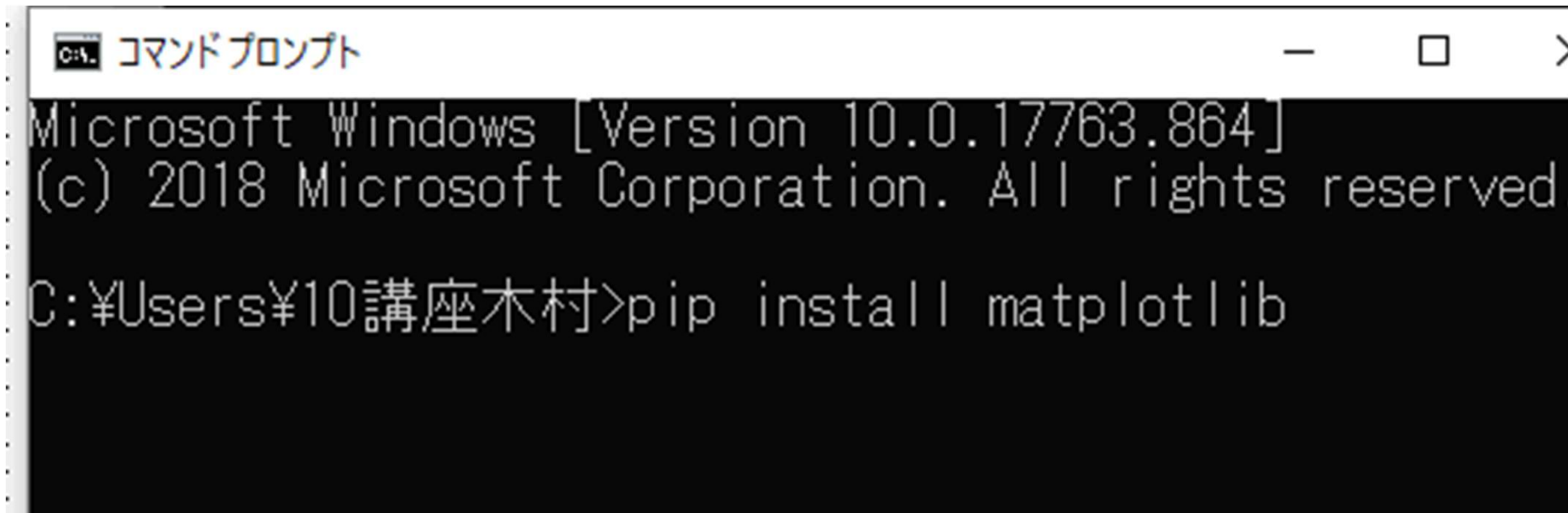
### 2) 数値計算ライブラリ numpy

ベクトルや行列などの計算

# グラフ描画ライブラリ matplotlib をインストールする

Windows10の場合: コマンドプロンプト上で

> pip install matplotlib  
を実行

A screenshot of a Windows Command Prompt window. The title bar reads 'コマンドプロンプト'. The window content shows the following text: 'Microsoft Windows [Version 10.0.17763.864]', '(c) 2018 Microsoft Corporation. All rights reserved.', and 'C:\Users\¥10講座木村>pip install matplotlib'.

```
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\¥10講座木村>pip install matplotlib
```

インストールがうまく行かない場合、  
インターネットを検索して解決してください

```
import matplotlib # グラフ描画ライブラリ
matplotlib.use('Agg') # グラフを画面に表示しないで画像ファイルとして保存する
import matplotlib.pyplot as plt
#----データを定義
num_list_x=[0,3,6,11,12,7, 2,1] # xデータ
num_list_y=[3,4,3, 4, 8,9,10,6] # yデータ
```

## 例1 散布図の画像生成

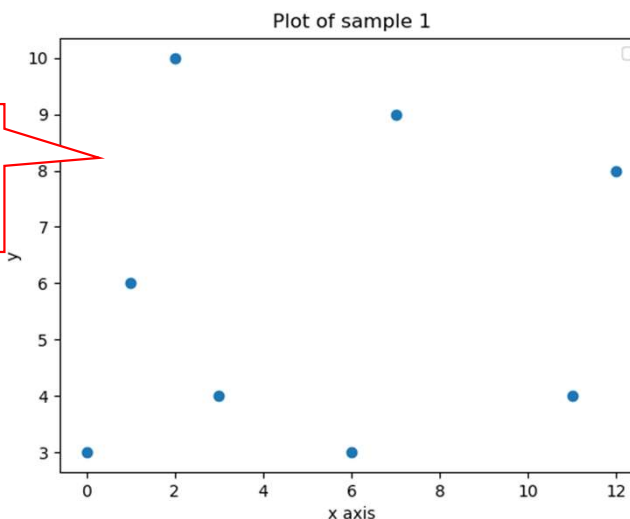
```
#---figure画面figの生成
fig = plt.figure()
#---figure画面へ埋め込まれるデータプロット領域axを生成
ax = fig.add_subplot(1, 1, 1) # fig.add_subplot(行,列,場所)を表します。
#---データプロット領域axにx軸のラベル設定
ax.set_xlabel('x axis')
#---データプロット領域axにy軸のラベル設定
ax.set_ylabel('y')
```

figure内部に  
subplotのaxを生成

```
#---データプロット領域axにデータの散布図描画
ax.scatter(num_list_x, num_list_y)
```

```
#---データプロット領域axに legend と title を表示
ax.legend(loc='best')
ax.set_title('Plot of sample 1')
#---描画したグラフを png 形式の画像ファイルとして保存
plt.savefig('figure.png') #【注意】予め matplotlib.use('Agg') を実行しておくこと
```

実行結果：  
figure.png



## 例2 折れ線グラフの画像生成

前スライドの以下の部分

```
#----データプロット領域axにデータの散布図描画  
ax.scatter(num_list_x, num_list_y)
```

ここを

```
ax.plot(num_list_x, num_list_y)
```

または

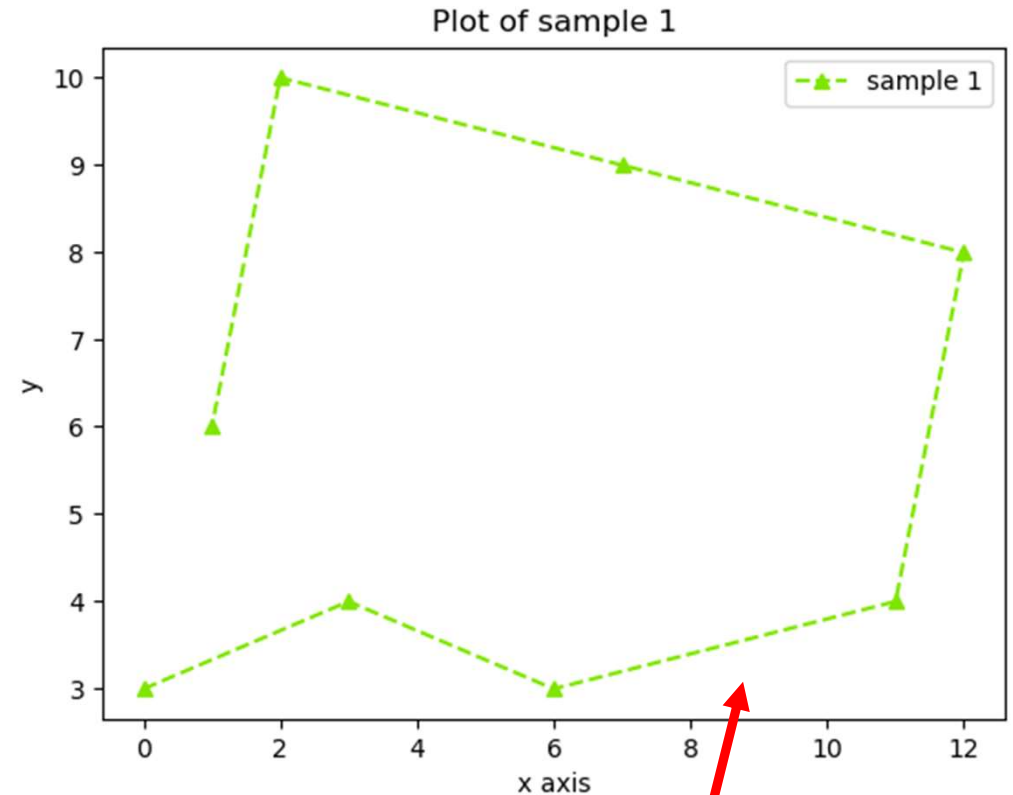
```
ax.plot(num_list_x, num_list_y, linestyle='-', color='b', label='sample 1', marker='x')
```

または

```
ax.plot(num_list_x, num_list_y, linestyle='--', color=[0.5,0.9,0], label='sample 1', marker='^')
```

へ変更

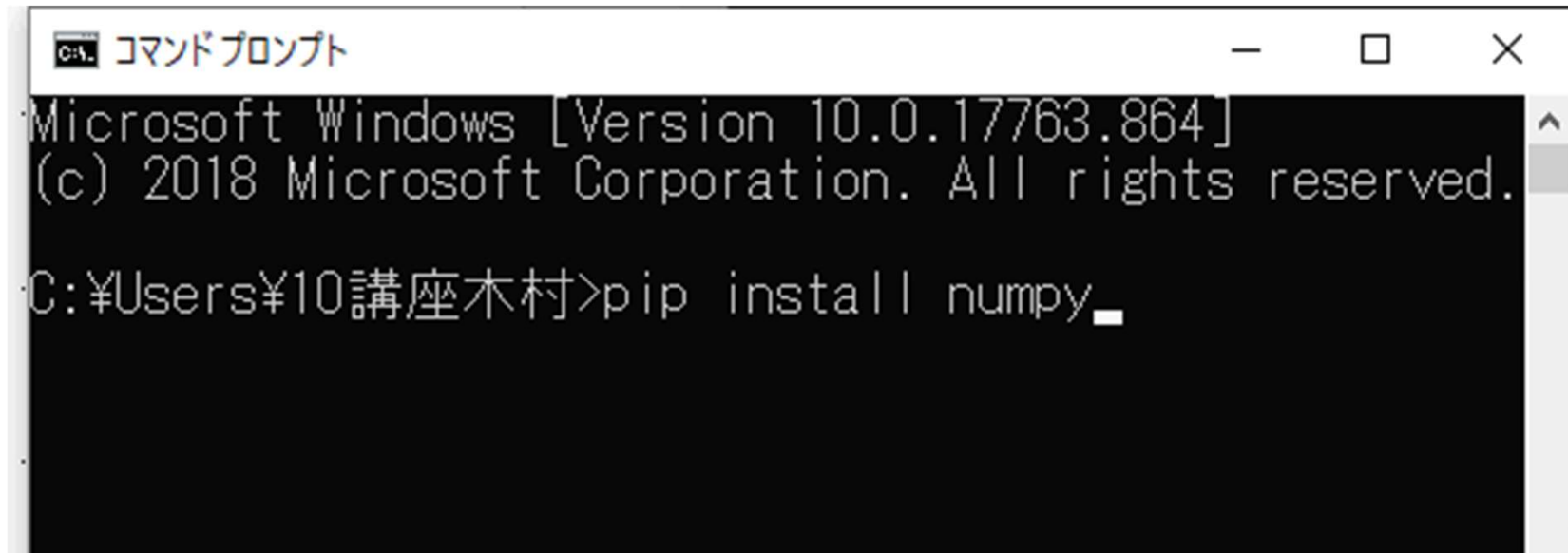
---



# 数値計算ライブラリ numpy をインストールする

Windows10の場合: コマンドプロンプト上で

> pip install numpy  
を実行

A screenshot of a Windows Command Prompt window. The title bar reads 'コマンドプロンプト'. The window content shows the following text: 'Microsoft Windows [Version 10.0.17763.864] (c) 2018 Microsoft Corporation. All rights reserved. C:¥Users¥10講座木村>pip install numpy\_'. The cursor is positioned at the end of the command line.

```
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:¥Users¥10講座木村>pip install numpy_
```

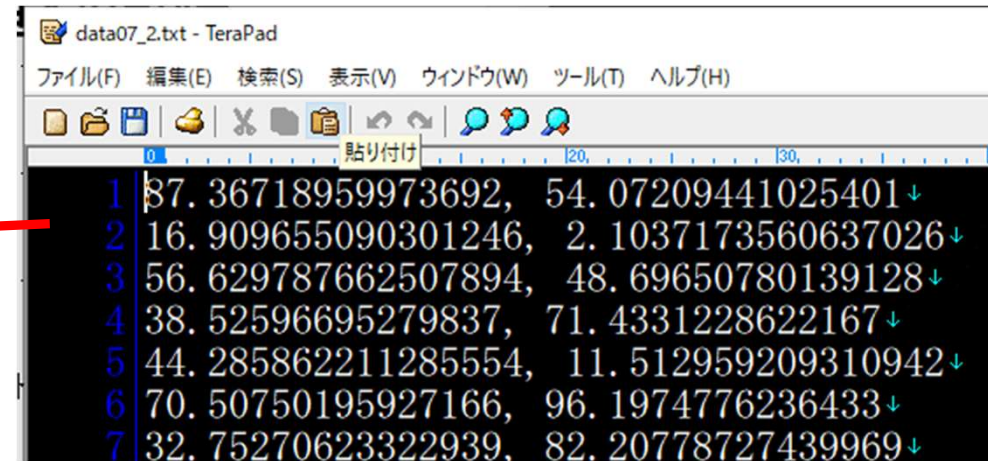
pipでインストールしたライブラリをアンインストールする場合は  
> pip uninstall numpy  
でOK

## 例3 単純回帰

```
import numpy as np #---数値計算ライブラリ
import matplotlib # グラフ描画ライブラリ
matplotlib.use('Agg') # グラフを画面に表示しないで画像ファイルとして保存する
import matplotlib.pyplot as plt
```

```
x_list=[] #数字のリストを作成 最初は空っぽ
y_list=[] #数字のリストを作成 最初は空っぽ
#-----データをファイルから読み込み
```

```
with open("data07_2.txt","r") as datafile:
    while True:
        lineStr = datafile.readline().strip()
        if lineStr=="":
            break
        # print( lineStr )
        moji_list=lineStr.split(sep=",") #----文字列lineStrをカンマで分割してリストmoji_list生成
        x_list.append( float(moji_list[0]) )
        y_list.append( float(moji_list[1]) )
```



```
data07_2.txt - TeraPad
ファイル(F) 編集(E) 検索(S) 表示(V) ウィンドウ(W) ツール(T) ヘルプ(H)
貼り付け
1 87.36718959973692, 54.07209441025401 ↓
2 16.909655090301246, 2.1037173560637026 ↓
3 56.629787662507894, 48.69650780139128 ↓
4 38.52596695279837, 71.4331228622167 ↓
5 44.285862211285554, 11.512959209310942 ↓
6 70.50750195927166, 96.1974776236433 ↓
7 32.75270623322939, 82.20778727439969 ↓
```

```
model = np.polyfit(x_list, y_list, 1)
modelfunc = np.poly1d( model ) # 回帰モデルから1次式を得る
```

np.polyfit(x,y,n): n次式で2変数の回帰分析

プログラムの続きは次スライド

## 前スライドのプログラムの続き

```
#---figure画面figの生成
fig = plt.figure()
#---figure画面へ埋め込まれるデータプロット領域axを生成
ax = fig.add_subplot(1, 1, 1) # fig.add_subplot(行,列,場所)を表します。

#---データプロット領域axにx軸のラベル設定
ax.set_xlabel('x axis')

#---データプロット領域axにy軸のラベル設定
ax.set_ylabel('y')

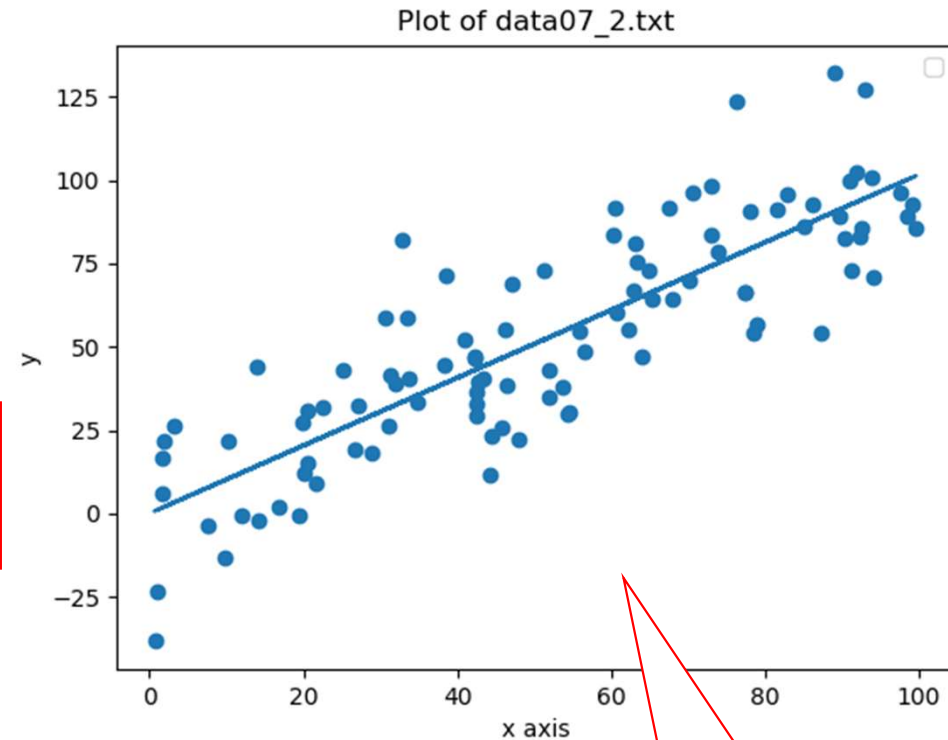
#---データプロット領域axにデータの散布図描画
ax.scatter( x_list, y_list )

#---データプロット領域axにデータのグラフ描画
ax.plot( x_list, modelfunc(x_list) )

#---データプロット領域axに legend と title を表示
ax.legend(loc='best')
ax.set_title('Plot of data07_2.txt')

#---描画したグラフを png 形式の画像ファイルとして保存
plt.savefig('figure07_2.png') # 【注意】予め matplotlib.use('Agg') を実行しておくこと
```

前スライドで  
計算した回帰  
モデル



実行結果:  
figure07\_2.png

# 多重回帰

観測値  $y$  を変数  $x_1, x_2, \dots, x_K$  を用いて以下の式で説明する:

$$y = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_K x_K + e$$

確率変動・誤差

このとき、 $n$  個の観測値  $(y_1, x_{11}, x_{12}, \dots, x_{1K}), (y_2, x_{21}, x_{22}, \dots, x_{2K}), (y_n, x_{n1}, x_{n2}, \dots, x_{nK})$  によって係数  $b_0, b_1, b_2, \dots, b_K$  の最小2乗推定量を求める。ここで、

$$\begin{array}{l}
 \text{目的変数} \\
 \text{行列} \\
 \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}
 \end{array}
 \quad
 \begin{array}{l}
 \text{説明変数} \\
 \text{行列} \\
 \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1K} \\ 1 & x_{21} & x_{22} & \dots & x_{2K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{nK} \end{bmatrix}
 \end{array}
 \quad
 \begin{array}{l}
 \mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_K \end{bmatrix} \\
 \text{回帰係数} \\
 \text{行列}
 \end{array}
 \quad
 \begin{array}{l}
 \text{誤差変数} \\
 \text{行列} \\
 \mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}
 \end{array}
 \quad \text{と表すと、}$$

$\mathbf{y} = \mathbf{Xb} + \mathbf{e}$       誤差変数行列  $\mathbf{e}$  の平方和  $\|\mathbf{e}\|^2$  を最小にする  $\mathbf{b}$  を求める

線形表現

→ 回帰推定(最小2乗法)    回帰モデル



# データから回帰モデルを得て何がうれしいか？ 回帰モデルによる推定

未知の説明変数(回帰変数)の値が  $(x_{q1}, x_{q2}, \dots, x_{qK})$  で与えられたときの

目的変数(被回帰変数)の値  $y_q$  をデータから**推定**できる！

$$y_q = b_0 + b_1 x_{q1} + b_2 x_{q2} + \dots + b_K x_{qK}$$

**推定値**

誤差eの項はゼロで計算

それでは、  
回帰係数行列

$$\mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_K \end{bmatrix}$$

をデータからどのように求めるか？

誤差ベクトル  $\mathbf{e}$  の平方和  $\|\mathbf{e}\|^2$  を最小にする  $\mathbf{b}$  を最尤推定値  $\hat{\mathbf{b}}$  と表すと、

単純回帰の場合と同様に、回帰係数の各要素で誤差ベクトルの平方和を偏微分し、これらが全てゼロとした連立方程式を立てて解くことにより、回帰係数ベクトルは以下の式で計算される：

$$\hat{\mathbf{b}} = \left( \mathbf{X}^{\text{Trans}} \mathbf{X} \right)^{-1} \mathbf{X}^{\text{Trans}} \mathbf{y}$$

ただし  $\mathbf{X}^{\text{Trans}}$  は  $\mathbf{X}$  の転置行列を表す。

$\mathbf{X}$  の擬似逆行列  $\mathbf{X}^+$


pseudo-inverse matrix

ただし  $\mathbf{X}$  は  $m$  行  $n$  列、 $m > n$

$\|\mathbf{e}\|^2$  の最小値  $S_e$  を残差平方和といい、

$$S_e = \mathbf{y}^{\text{Trans}} \left\{ \mathbf{I} - \mathbf{X} \left( \mathbf{X}^{\text{Trans}} \mathbf{X} \right)^{-1} \mathbf{X}^{\text{Trans}} \right\} \mathbf{y}$$

± $\sigma$  の範囲内に  
68.27% の  
データが存在

で与えられる。   $\sigma = \sqrt{\frac{S_e}{n}}$  より、回帰で推定する場合の精度が分かる

**【システム設計工学演習問題】 2019.12.6 学籍番号**

氏名

自動車の燃費について以下のようなデータがある。

「クラウン」から「ギャランΣ」までのデータを利用して車種「ルーチェ」の10モード走行性能  $y$  を予測するには、どのような計算をしたら良いか？ **表の数値を使って行列X,Yを作成**し、それらを使って具体的な計算方法を説明せよ。

車名	x1	x2	x3	X4	x5	x6	y
クラウン	1.360	4.778	2.4251	125	17.5	8.8	8.7
マークII	1.245	4.100	2.4082	125	17.5	8.8	9.5
カムリ	1.070	3.214	2.3575	120	17.6	8.7	10.6
ソアラ	1.235	4.100	2.3052	125	17.5	8.8	9.2
セドリック	1.420	4.625	2.4251	130	17.5	9.5	8.9
ローレル	1.175	3.889	2.3660	125	17.0	9.1	9.2
スカイライン	1.175	4.111	2.3198	125	17.0	9.1	9.2
レパード	1.220	3.900	2.2899	125	17.0	9.1	9.4
カペラ	1.030	3.450	2.3829	120	17.0	8.6	10.2
ギャランΣ	1.180	3.665	2.3645	110	16.7	8.5	10.6
ルーチェ	1.150	3.909	2.3829	120	17.0	8.6	?

x1:車体重量(1000kg), x2:減速比, x3:幅×高さ(m<sup>2</sup>), x4:最大出力(ps),  
x5:最大トルク(kgm), x6:圧縮比, y:10モード走行(km/l)

## 例4 多重回帰

```
import numpy as np #----数値計算ライブラリ
from numpy import linalg as LA #----行列計算モジュール

#-----データをリストへ格納して定義
# [1, 車体重量(1000kg),減速比,幅×高さ(m2),最大出力(ps),最大トルク(kgm),圧縮比]
# ^^最初の要素は1
data_x_list = [[ 1.0, 1.36 , 4.78 , 2.43 , 125 , 17.5 , 8.8 ]] # クラウン
data_x_list.append([ 1.0, 1.25 , 4.10 , 2.41 , 125 , 17.5 , 8.8 ]) # マークII
data_x_list.append([ 1.0, 1.07 , 3.21 , 2.36 , 120 , 17.6 , 8.7 ]) # カムリ
data_x_list.append([ 1.0, 1.24 , 4.10 , 2.31 , 125 , 17.5 , 8.8 ]) # ソアラ
data_x_list.append([ 1.0, 1.42 , 4.63 , 2.43 , 130 , 17.5 , 9.5 ]) # セドリック
data_x_list.append([ 1.0, 1.18 , 3.89 , 2.37 , 125 , 17.0 , 9.1 ]) # ローレル
data_x_list.append([ 1.0, 1.18 , 4.11 , 2.32 , 125 , 17.0 , 9.1 ]) # SKYLINE
data_x_list.append([ 1.0, 1.22 , 3.90 , 2.29 , 125 , 17.0 , 9.1 ]) # レパード
data_x_list.append([ 1.0, 1.03 , 3.45 , 2.38 , 120 , 17.0 , 8.6 ]) # カペラ
data_x_list.append([ 1.0, 1.18 , 3.67 , 2.36 , 110 , 16.7 , 8.5 ]) # ギャランΣ
#-----各車種の10モード走行燃費(km/L)
data_y_list = [[ 8.7],[9.5],[10.6],[9.2],[8.9],[9.2],[9.2],[9.4],[10.2],[10.6]]
#-----データのリストから行列を生成
matrix_x = np.matrix( data_x_list ) # 行列x生成
matrix_y = np.matrix( data_y_list ) # 行列y生成

print( matrix_x ) # 確認用
print( matrix_y ) # 確認用
```

プログラムの続きは次スライド

## 前スライドのプログラムの続き

$$\hat{\mathbf{b}} = \left( \mathbf{X}^{\text{Trans}} \mathbf{X} \right)^{-1} \mathbf{X}^{\text{Trans}} \mathbf{y}$$

```
#-----行列xとyから回帰パラメータbを求める
param_b = LA.inv(matrix_x.T @ matrix_x) @ matrix_x.T @ matrix_y
# 逆行列 転置行列 @は行列の積

#-----推定対象のルーチェの要目
query_x = np.matrix([[1.0, 1.15 , 3.91 , 2.38 , 120 , 17.0 , 8.6 ]])
print("query_x = ")
print( query_x )

#-----回帰パラメータbと推定対象の要目から、推定対象のyを推定
estimated_y = query_x @ param_b #---行列の積
print("estimated_y = ")
print( estimated_y ) # 推定値を画面に出力 (ちなみに観測データでは9.8)
```

# 課題：多重回帰プログラム

- 1) 例4のプログラムにコードを追加し、残差平方和 $Se$ とデータ個数から推定精度 $\sigma$ を計算して表示すること

## 【ヒント】

課題のプログラムでは、この値を数字ではなく、データがさらに追加されてもプログラムを変更せずに済むように他の行列(リスト)の要素の個数から自動的に与えるようにすること

- 1)  $N \times N$ の正方行列の単位行列は、`numpy.eye( N )` で与えられる。
- 2)  $x$  の平方根は `numpy.sqrt( x )` でも得られる

以上のプログラムを作成せよ。

プログラムの先頭行にコメント文で自分の氏名と学籍番号を入れておくこと

# ← この記号より左側の文字列はコメントになる

\* データファイルdata.txtの読み込みにおいて、文字コードに関するエラーが解消しない場合、データファイルの文字コードをSHIFT JISにして保存してみよ。

## まとめ

グラフ描画ライブラリ **matplotlib**

数値計算ライブラリ **numpy**

ベクトルや行列の計算／多項式近似

## 第7回 レポート課題提出方法

課題のプログラムを

下記の課題提出用フォルダへ、課題の番号と提出者が分かるようにファイル名を以下のようにしてアップロードせよ

第7回1TE19xxxZ名前.py

[https://share.iii.kyushu-u.ac.jp/public/IRbwAAVITI5A2X4BE45t6TqQIE0UQSQUI5Bap\\_kZ\\_sjy](https://share.iii.kyushu-u.ac.jp/public/IRbwAAVITI5A2X4BE45t6TqQIE0UQSQUI5Bap_kZ_sjy)

講義資料、および上記フォルダへのリンクは下記ホームページから

<http://sysplan.nams.kyushu-u.ac.jp/gen/edu/InfoProcess/2019/index.html>

プログラムがエラーで動かないとき

エラーメッセージをよく読もう

エラー箇所が行が表示されているはず(その1行前も怪しい)

文字が全角になっていないか確認すること

カッコ()やクォーテーション””が全角になっていないか？

空白部分に全角の空白が紛れ込んでいないか？

インデントは「文字数」でそろえないとエラーになる