プログラミング応用 第3回

河瀬 康志

2018年6月25日

アウトライン

演習問題

問 1

'20180618' のような年月日の 8 桁からなる文字列 (yyyymmdd) を引数として，'2018/6/18' のような文字列 (yyyy/mm/dd) を出力する関数を作成せよ．逆に，年月日がそれぞれ整数 (year,month,day) として与えられた時，8 桁の文字列に変換する関数も作成せよ．($0 \leq \text{year} \leq 9999$ と仮定する)

問 2

`http://yambi.jp/lecture/advanced_programming2018/kokoro.txt` では，ルビが《》を用いて表されている．例えば，

私《わたくし》は

のように表記されている．kokoro.txt を読み込み，ルビを削除したファイル kokoro2.txt を出力するプログラムを作成せよ．

演習問題

問 3

ランダムに線分を何本かつくり，どれとも交差していない線分は黒で，
どれかと交差している線分は赤で表示せよ．

問 4

凸包のプログラムを，凸包の面積の表示もするように改造せよ．
面積の表示には `create_text` を用いよ．

問 5 (おまけ)

凸包の計算を 1 ステップずつ描画するようなアニメーションを作成せよ．

授業スケジュール

	日程	内容
第 1 回	6/11	ガイダンス・復習
第 2 回	6/18	文字列操作（文字列整形，パターンマッチ，正規表現） 平面幾何（線分の交差判定，点と直線の距離，凸包）
第 3 回	6/25	乱数（一様分布，正規分布への変換，乱数生成） 統計（データ処理，フィッティング）
第 4 回	7/2	計算量（オーダー表記） スタックとキュー（幅優先探索，深さ優先探索）
第 5 回	7/9	ソートアルゴリズム バックトラック（N クイーン問題，数独）
第 6 回	7/23	動的計画法（ナップサック問題） 最短経路探索（Warshall-Floyd, Bellman-Ford, Dijkstra）
第 7 回	7/30	巡回セールスマン問題
期末試験	8/6?	—

アウトライン

乱数とは

- 乱数とは：何の規則性も持たないような数列
- 確定的な計算では真の乱数を作ることはできない

乱数の作り方

- 乱数抽出：ランダムに見える現象から乱数を取り出す
  - キーボード入力のタイミング，マウスの動きなど
  - 遅い
- 疑似乱数：短い乱数から「ランダムに見える」長いビット列を作る
  - 元となる乱数を「疑似乱数の種」という
  - 高速

乱数抽出

os.urandom(n) で n バイトの乱数を取り出す
（ただし，情報が十分に溜まっていない場合は疑似乱数となる）

```
>>> import os
>>> list(os.urandom(4)) # 4 バイトからなるランダムな文字列
[64, 185, 152, 30]
```

random モジュールでは，この乱数を疑似乱数の種として使う
（環境によっては現在時刻が使われる）

疑似乱数

- Python では random モジュールを使うことで利用できる
- Mersenne Twister (MT19937) という生成方法を使っている
  - 1997 年に松本眞と西村拓士が発表
  - 周期は $2^{19937} - 1$ (メルセンヌ素数)

```
>>> import random
>>> random.random() # 値域 [0.0,1.0) のランダムな小数 (標準一様分布)
0.48559804457756095
>>> random.seed(10) # 疑似乱数の種を 10 に設定
>>> random.random(), random.random(), random.random()
(0.571402594689913 5, 0.4288890546751146, 0.5780913011344704)
>>> random.seed(10) # 疑似乱数の種を 10 に再設定
>>> random.random(), random.random(), random.random()
(0.571402594689913 5, 0.4288890546751146, 0.5780913011344704)
```

ランダム選択

```
>>> import random
>>> random.randrange(10) # 0,1,...,9 からランダム選択
3
>>> random.randrange(1,7) # 1,2,...,6 からランダム選択
2
>>> random.randint(1,6) # 1,2,...,6 からランダム選択
6
>>> random.choice(range(1,7)) # 1,2,...,6 からランダム選択
7
>>> random.choice('abcdefghijklmnopqrstuvwxyz') # ランダムな文字
'e'
>>> ''.join([random.choice('01') for i in range(10)]) # ランダム 01 列
'1001011101'
```

確率的条件分岐

確率 p で f(), 確率 $1-p$ で g() を実行

```
p = 0.7
if random.random()<p:
    f()
else:
    g()
```

確率 p で f(), 確率 q で g(), 確率 $1-p-q$ で h() を実行

```
p,q = 0.3,0.5
r = random.random()
if r<p:
    f()
elif r<p+q:
    g()
else:
    h()
```

シャッフル

リストのシャッフルとして正しいのは？

```
def shuffle0(l):
    random.shuffle(l)

def shuffle1(l):
    for i in range(len(l)):
        j = random.randrange(len(l))
        l[i],l[j]=l[j],l[i]

def shuffle2(l):
    for i in range(len(l)):
        j = random.randrange(len(l))
        k = random.randrange(len(l))
        l[j],l[k]=l[k],l[j]

def shuffle3(l):
    for i in range(len(l)):
        j = random.randrange(i,len(l))
        l[i],l[j]=l[j],l[i]
```

シャッフル

リストのシャッフルとして正しいのは？

```
def shuffle0(l):
    random.shuffle(l)

def shuffle1(l):
    for i in range(len(l)):
        j = random.randrange(len(l))
        l[i],l[j]=l[j],l[i]

def shuffle2(l):
    for i in range(len(l)):
        j = random.randrange(len(l))
        k = random.randrange(len(l))
        l[j],l[k]=l[k],l[j]

def shuffle3(l):
    for i in range(len(l)):
        j = random.randrange(i,len(l))
        l[i],l[j]=l[j],l[i]
```

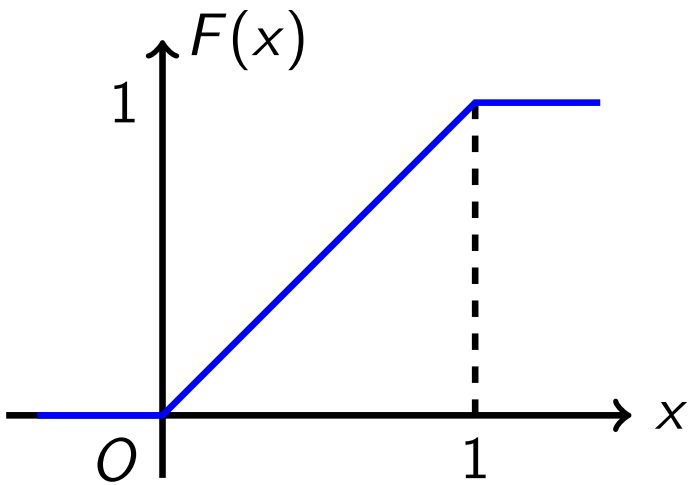
shuffle0 と shuffle3 とだけが正しい $\Rightarrow$ 演習問題

連続分布

累積分布関数

- $F(x) = P[X \leq x]$

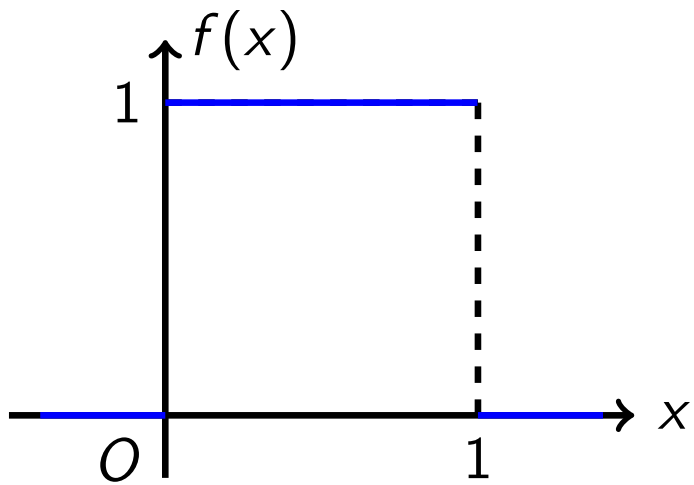
例：標準一様分布 $U(0,1)$



確率密度関数

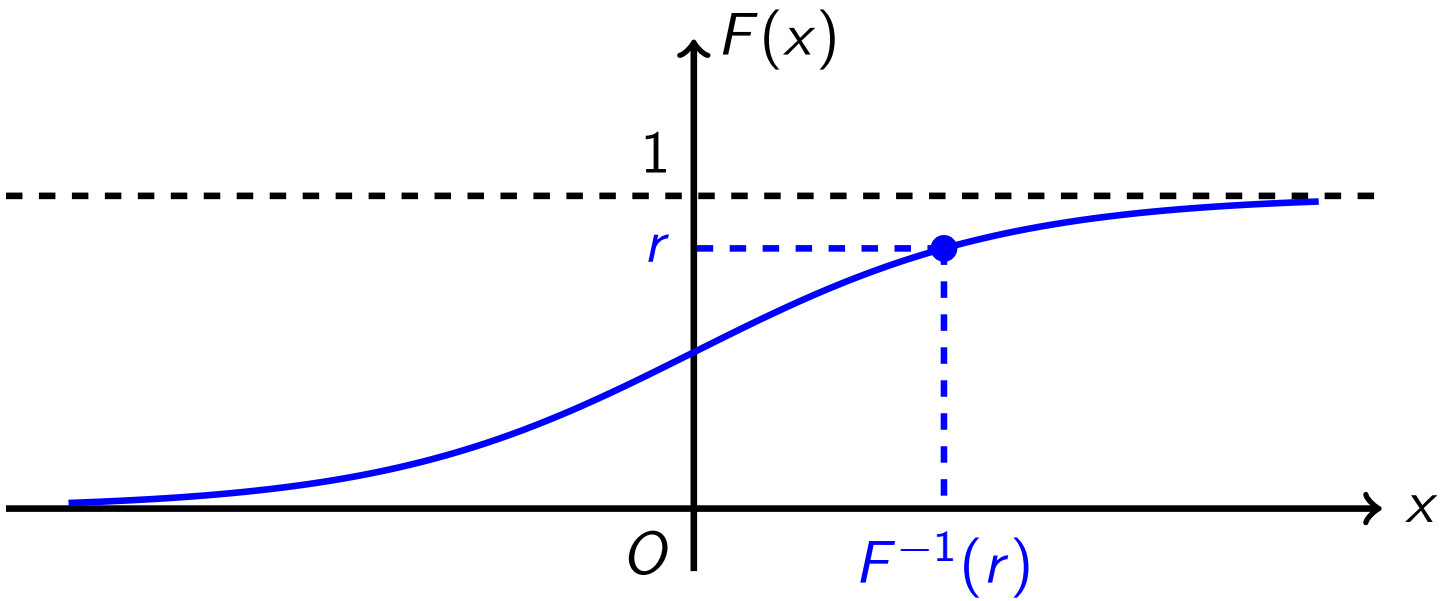
- $f(x) = dF(x)/dx$

例：標準一様分布 $U(0,1)$



標準一様分布からの変換

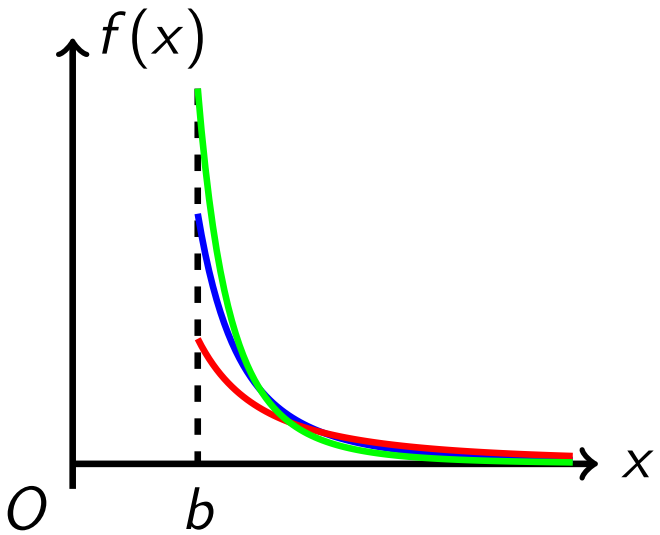
累積分布関数が $F(x)$ の分布である乱数は，
標準一様分布の乱数 r に対し，$F^{-1}(r)$ で得られる

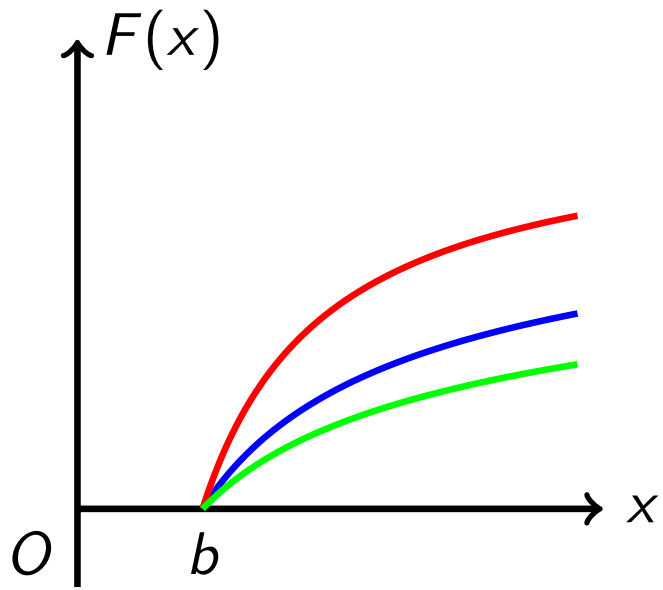


例：パレート分布

累積分布関数は a, b $(a > 0, b > 0)$ をパラメータとして

$$F(x) = \begin{cases} 1 - \left(\frac{b}{x}\right)^a & (x \geq b), \\ 0 & (x < b) \end{cases} \qquad F^{-1}(x) = \frac{b}{\sqrt[a]{1-x}}.$$





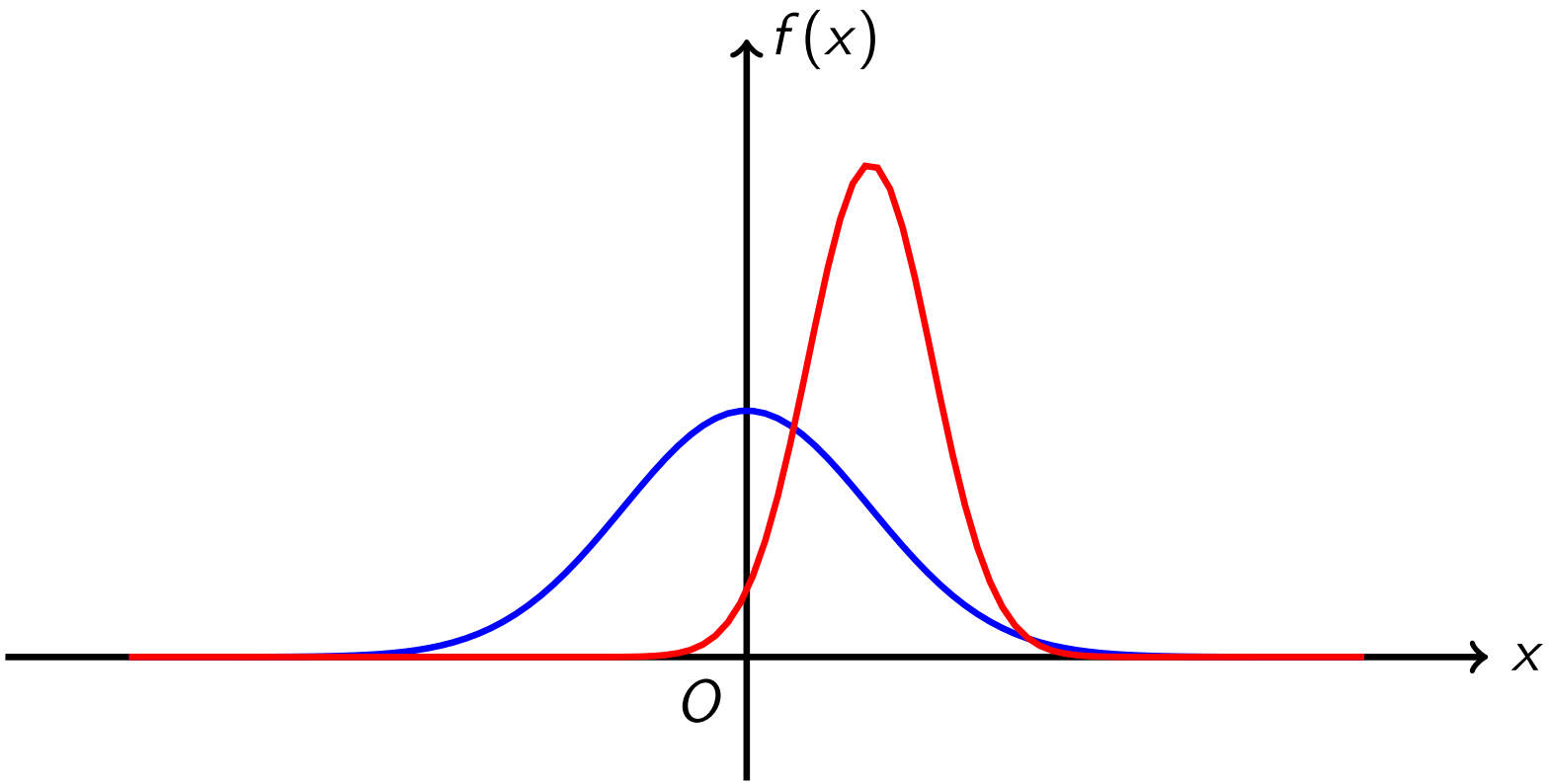

```
>>> import random
>>> a, b = 2.0, 1.0
>>> b/(1.0-random.random())**(1.0/a)
1.2182178643933748
```

正規分布（ガウス分布）

平均 μ, 分散 σ^2 とするとき，確率密度関数は

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

特に，平均 0, 分散 1 の正規分布を標準正規分布という



正規分布に従う乱数

```
>>> import random
>>> random.gauss(0,1) # 標準正規分布
1.584870567860332
>>> random.gauss(1,2) # 平均 1，標準偏差 2 の正規分布
4.548914172664087
```

Advanced Topic: 標準一様分布からの変換 (Box-Muller 変換)

X, Y を独立な標準一様分布とすると，

$$Z_1 = \sqrt{-2\log X}\cos(2\pi Y), \qquad Z_2 = \sqrt{-2\log X}\sin(2\pi Y)$$

は独立な標準正規分布となる

```
>>> import random
>>> import math
>>> x,y = random.random(),random.random()
>>> z1=(-2*math.log(x))**0.5 * math.cos(2*math.pi*y)
>>> z2=(-2*math.log(x))**0.5 * math.sin(2*math.pi*y)
```

Advanced Topic: モンテカルロ法

- 乱数を用いたシミュレーションや数値積分をする方法
- 例：円周率を求める
  - 正方形 $(0 \leq x, y \leq 1)$ の中に四分円 $(x, y \geq 0,\ x^2 + y^2 \leq 1)$ が内接
  - 正方形の中にランダムに点を打ったときに四分円に入る確率は $\pi/4$
  - n 回シミュレーションをして m 回入ったとするとき $4 \cdot m/n$ は π に近いはず

```
import random
m,n=0,10000
for i in range(n):
    if random.random()**2+random.random()**2<=1:
        m+=1
print(4.0*m/n) # 3.1668
```

アウトライン

基本的な統計量

```
>>> l = [27,31,53,49]
>>> max(l) # 最大値
53
>>> min(l) # 最小値
27
>>> average = float(sum(l))/len(l) # 平均
>>> average
40.0
>>> sum([(i-average)**2 for i in l])/len(l) # 分散
125.0
```

同じ要素の数え上げ

```
>>> import random
>>> l = [random.randint(1,10) for i in range(100)] # 1,..,10 の乱数 100 個
>>> c = {}
>>> for i in l: c[i]=c.get(i,0)+1 # 数え上げ
...
>>> c
{1: 7, 2: 14, 3: 8, 4: 7, 5: 7, 6: 8, 7: 15, 8: 11, 9: 9, 10: 14}
>>> max(c.keys(),key=c.get) # 最頻値を出力
7
>>> [k for (k,v) in c.items() if v==max(c.values())] # 複数の最頻値がある場合
[7]
>>> total = sum(c.values()) # 要素の個数
>>> t = 0 # prefix sum
>>> for i in sorted(c.keys()):
...     if t <= total/2 < t+c[i]: print(i) # 中央値
...     t += c[i]
...
6
```

numpy を使う場合

```
>>> import numpy as np
>>> l = [27,31,53,49]
>>> np.average(l) # 平均
40.0
>>> np.var(l)     # 分散
125.0
>>> np.median(l)  # 中央値
40.0
```

インストールされていない場合は，端末から以下を実行

```
conda install numpy
```

あるいは

```
python3 -m pip install numpy
```

CSV ファイル

- Comma-Separated Values (Character-Separated Values)
- いくつかの項目をカンマ (タブや半角空白) で区切ったテキスト
- Excel などの表計算ソフトでも扱える

sample.csv

```
first,last,gender,age,email
Theodore,Blake,Male,20,ecusamo@ehe.co.uk
Jimmy,Howell,Female,54,kawud@ke.io
Rachel,Fernandez,Male,39,eteune@jiiraomo.net
Cory,Webb,Male,20,jaag@izaobeama.com
Christian,Oliver,Female,30,petalif@vemus.com
Elva,Sims,Male,34,nubokhup@fazet.edu
Ryan,Briggs,Male,53,enivafju@mimvolu.gov
Amanda,Hernandez,Female,45,ro@fauz.com
Mathilda,Bradley,Female,40,veltagsus@cu.io
```

generated by http://www.convertcsv.com/generate-test-data.htm

CSV ファイルの読み込み

```
# 単純な方法
f=open('sample.csv','r')
table = [list(map(str.strip,line.split(','))) for line in f]
f.close()

# csv モジュールを用いる方法
import csv
f=open('sample.csv','r')
table2 = list(csv.reader(f))
f.close()
```

区切り文字を半角空白にしたい場合は `csv.reader(f,delimater=' ')`

CSV ファイルの書き込み

```
# 単純な方法
f=open('output.csv','w')
f.write(','.join(map(str,[1,2,3]))+'\n')
f.write(','.join(map(str,[4,5,6]))+'\n')
f.write(','.join(map(str,[7,8,9]))+'\n')
f.close()


# csv モジュールを用いる方法
import csv
f=open('output2.csv','w')
writer = csv.writer(f)
writer.writerow([1,2,3]) # 1 行を書き込み
writer.writerows([[4,5,6],[7,8,9]]) # 複数行を書き込み
f.close()
```

Advanced Topic: その他のファイル形式

よく使われるファイル形式として csv 以外にも下記のものがある

- XML (Extensible Markup Language)

```
<?xml version="1.0" encoding="utf-8" ?>
<list>
  <customer>
    <name>Theodore Blake</name>
    <age>20</age>
  </customer>
  <customer>
    <name>Jimmy Howell</name>
    <age>54</age>
  </customer>
</list>
```

- JSON (JavaScript Object Notation)

```
[
    {'name': 'Theodore Blake', 'age': 20},
    {'name': Jimmy Howell, 'age': 53}
]
```

matplotlib

- 2 次元グラフィックス用の Python パッケージ
- 詳細は `http://matplotlib.org/index.html`
- あるいは `http://www.scipy-lectures.org/intro/matplotlib/matplotlib.html`

インストールされていない場合は，端末から以下を実行

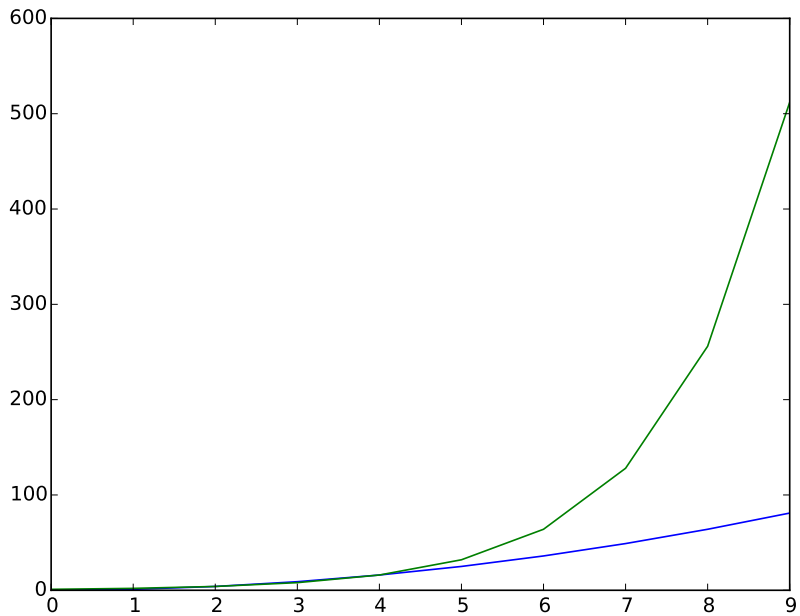
```
conda install matplotlib
```

あるいは

```
python3 -m pip install matplotlib
```

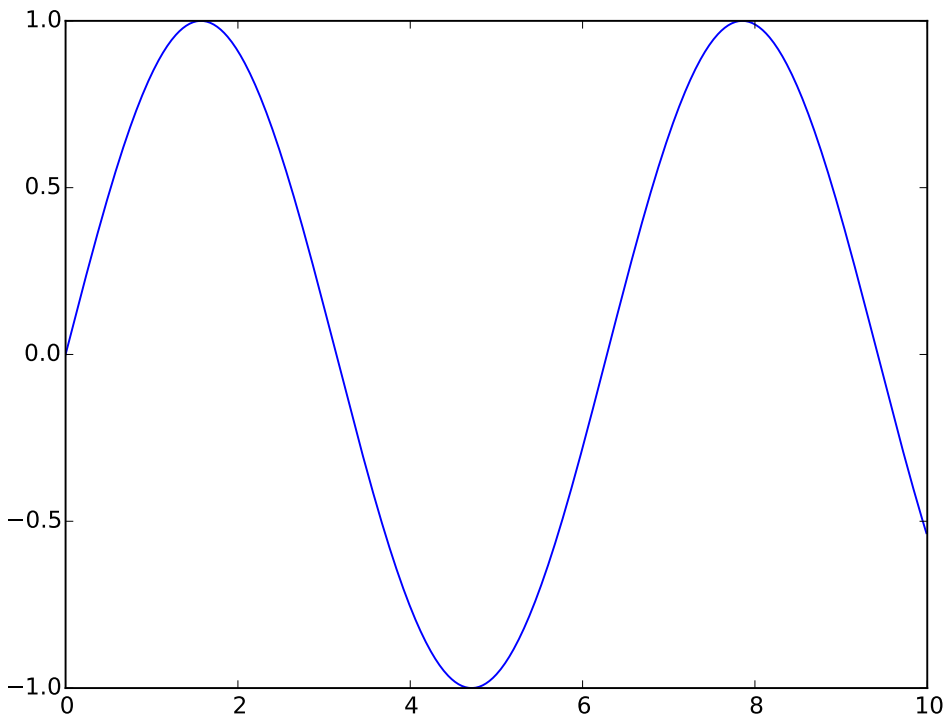
一次元リスト

```
import matplotlib.pyplot as plt
sq = [i**2 for i in range(10)]
exp = [2**i for i in range(10)]
plt.plot(sq)
plt.plot(exp)
plt.savefig('plot.pdf',format='pdf') # 保存
plt.show() # 表示
```



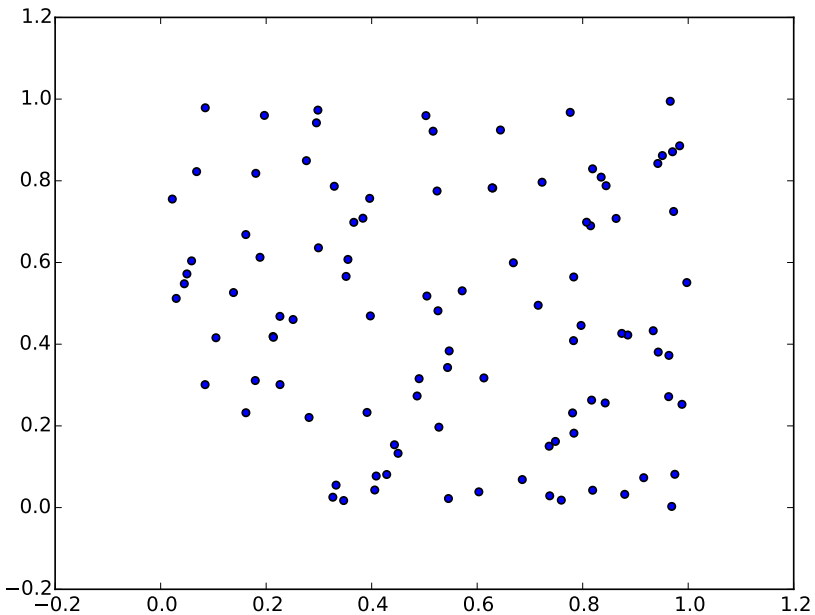
折れ線グラフ

```
import matplotlib.pyplot as plt
import math
xs = [x/100.0 for x in range(1000)]
ys = [math.sin(x) for x in xs]
plt.plot(xs,ys)
plt.show()
```



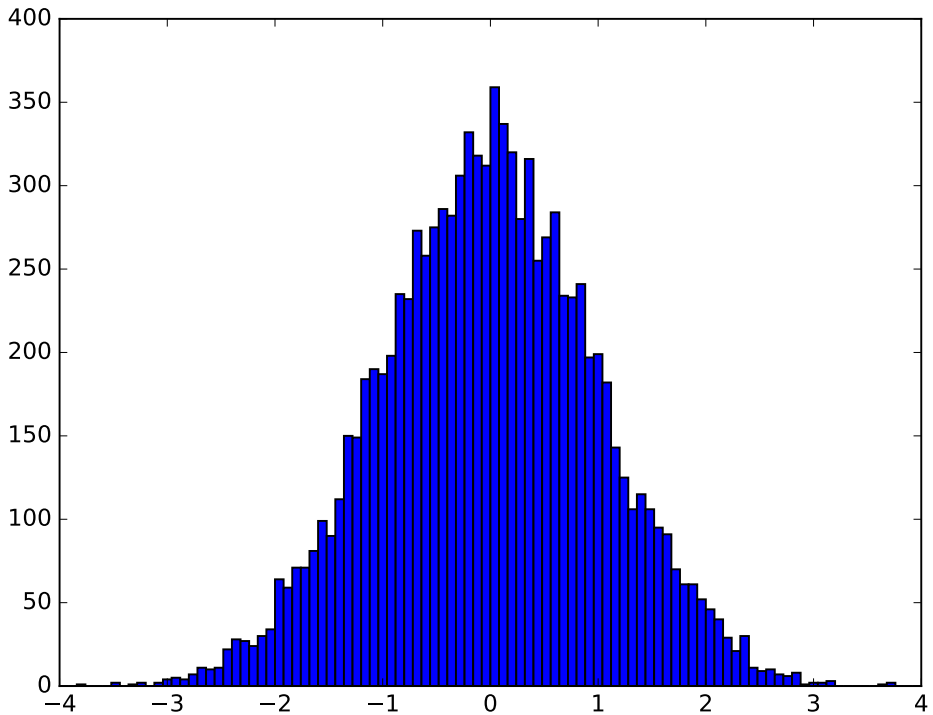
散布図

```
import random
import matplotlib.pyplot as plt

xs = [random.random() for i in range(100)]
ys = [random.random() for i in range(100)]
plt.scatter(xs,ys) # 散布図
plt.show() # 表示
```



ヒストグラム

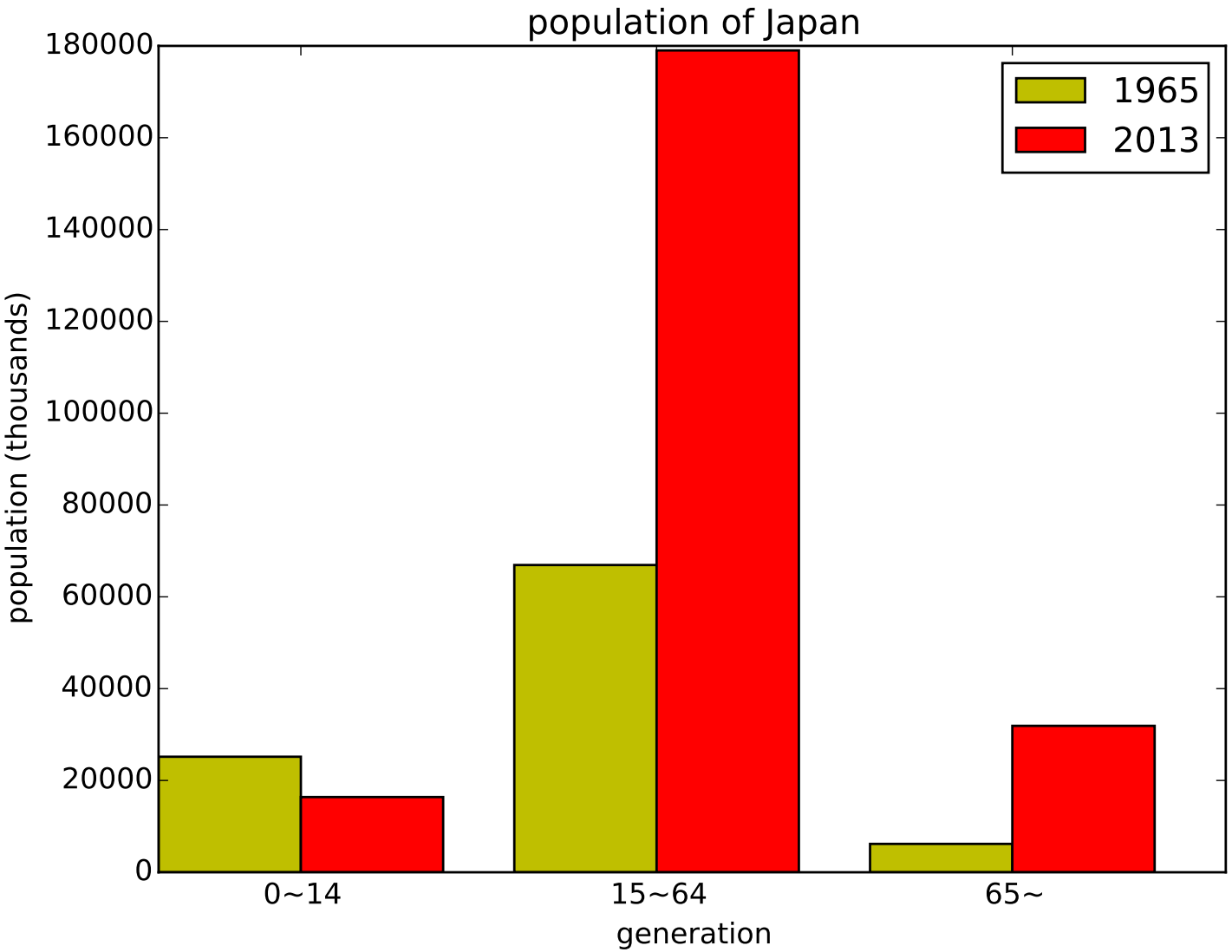
```
import random
import matplotlib.pyplot as plt

l = [random.gauss(0,1) for i in range(10000)]
plt.hist(l,bins=100,range=(-4,4)) # (-4,4) の区間を 100 個に分割
plt.show() # 表示
```



棒グラフ

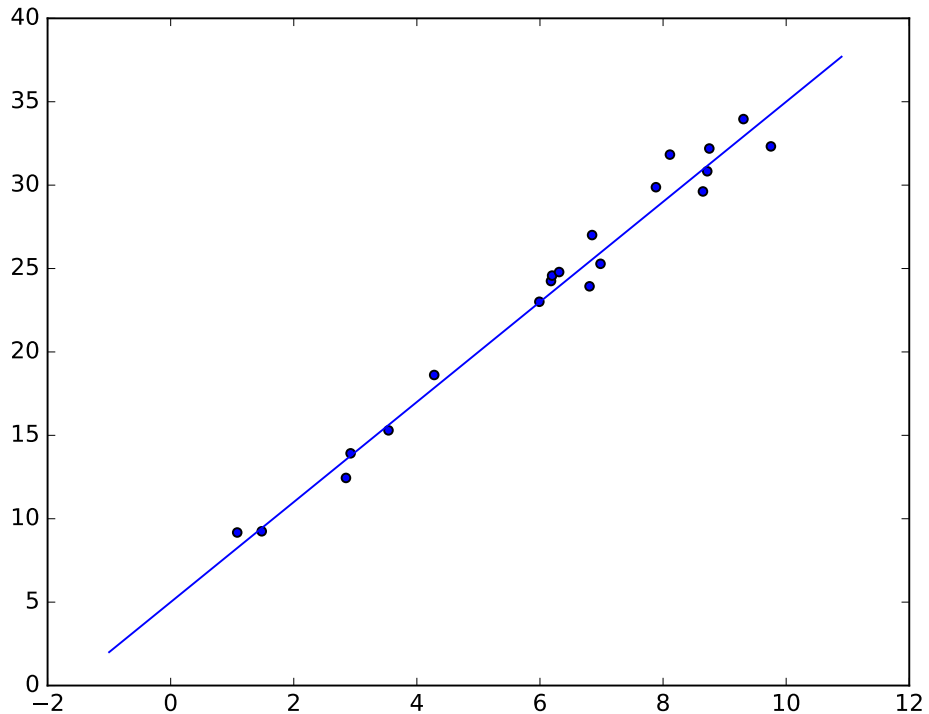
```
import matplotlib.pyplot as plt

xs1 = [1,2,3]
ys1 = [25166,66928,6181]

xs2 = [1.4, 2.4, 3.4]
ys2 = [16390,179010,31898]

plt.bar(xs1, ys1, color='y', width=0.4, label='1965')
plt.bar(xs2, ys2, color='r', width=0.4, label='2013')

plt.legend()
plt.title('population of Japan (thousands)')
plt.xticks([1.4, 2.4, 3.4], ['0~14','15~64','65~'])
plt.xlabel('generation') # x 軸ラベル
plt.ylabel('population (thousands)') # y 軸ラベル

plt.show()
```

棒グラフ


population of Japan
1965
2013
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
population (thousands)
0~14
15~64
65~
generation

最小二乗法

- データの組 (x_i, y_i) が n 組与えられた時に，
  そのデータの関係を表す最もらしい関数 $f(x)$ を求める方法
- $f(x) = \sum_j a_j g_j(x)$ と仮定 ($g_j(x)$ は既知の関数)
- $\sum_i (y_i - f(x_i))^2$ を最小にする a_j によって定める



最小二乗法

```
import random
import scipy.optimize

a,b = 3,5 # y=3x+5+noise のデータ生成
xs, ys = [], []
for i in range(20):
    r = random.uniform(0,10.0)
    xs.append(r)
    ys.append(a*r+b+random.gauss(0,1))

# y=ax+b でフィッティング
def func(x,a,b):
    return a*x+b

result,covariance=scipy.optimize.curve_fit(func,xs,ys)
print('a =', result[0])
print('b =', result[1])
```

アウトライン

演習問題提出方法

解答プログラムをまとめたテキストファイルを作成して，OCW-i で提出

- ファイル名は practice2.txt
- 次回授業の開始時間が締め切り
  (登録が間に合わないなどの場合は別途相談)
- ファイルの最初に学籍番号と名前を書く
- どの演習問題のプログラムかわかるように記述
- 出力結果もつける（描画する問題の場合はどのような結果が得られたか一言で説明）
- 途中までしかできなくても，どこまでできてどこができなかったかを書けば部分点を付けます
- 問 3,4 の出力結果は PDF で提出してください．

演習問題

問 1

リスト [1,2,3] を Shuffle1 によってシャッフルしたとき，[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1] はそれぞれどれだけの確率で出現するか？

- プログラムを用いずに答えてもよい．
- 余力がある人は Shuffle2 についても計算せよ．

問 2

大文字または小文字のアルファベット 8 文字からなるランダムなパスワードを生成するプログラムを作成せよ．

演習問題

問 3

http://yambi.jp/lecture/advanced_programming2018/data.csvには，各行が 2 つのデータの組からなる csv ファイルが置いてある．このファイルを読み込み，そのデータたちの関係を表すもっともらしい直線を最小二乗法を用いて求めよ．また，もっともらしい直線であることを，プロットすることにより確かめよ．

問 4

確率 1/100 で欲しいキャラが出るガチャをまわしたとき，何回で初めてそのキャラがでるか．10000 回試してその頻度分布をヒストグラムで表せ．