科学計算研究室 Python ゼミ　フォローアップ

～6. 最小二乗法～

2021-03-24 福田　浩

1　Python スクリプト

- 基本的には C++のソースファイルを 1 行毎 Python に変換している．
- 1, 2 行目：n は，データ点数
- 1, 3 行目：m は，近似多項式の次数
- 4 行目：jordan.py に引数として近似多項式の次数を渡すために，標準出力に出力
- 5, 6 行目：データ点数を格納する配列 (リスト) を作成
- 7～10 行目：データ x，y をスペース区切りで取得
- 12 行目：近似多項式の係数を格納する配列 (リスト)a[0 n] を 0 で初期化
- 14～20 行目：最小二乗法
- 21 行目：jordan.py に引数として近似多項式の係数を渡すために，標準出力に出力

```
 1  n, m = input().split(" ")
 2  n = int(n)
 3  m = int(m)
 4  print(m)
 5  x = []
 6  y = []
 7  for i in range(n):
 8      xx, yy = map(float, input().split(" "))
 9      x.append(xx)
10      y.append(yy)
11
12  a = [[0 for i in range(n+1)] for j in range(n)]
13
14  for i in range(m):
15      for j in range(m):
16          for k in range(n):
17              a[i][j]+= x[k]**(i+j)
18          print(a[i][j])
19      for k in range(n):
20          a[i][n] += y[k]*x[k]**i
21      print(a[i][n])
```

2　課題の解答

2.1　$(-2,1),(0,-1),(2,5)$ の 1 次近似

```
y= + 1.6666666666666667 x^0 + 1.0 x^1
```

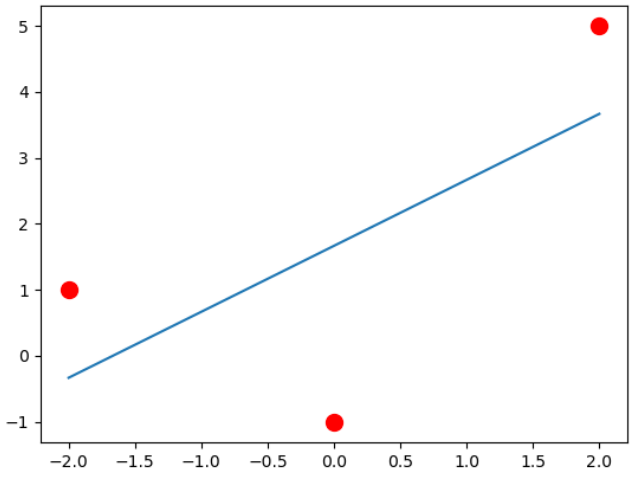

Figure 1: $(-2,1),(0,-1),(2,5)$ とその 1 次近似

2.2　$(1,2),(2,5),(3,7),(5,10),(6,17)$ の 3 次近似

```
y= + -6.1764705882361035 x^0
+ 11.00840336134552 x^1
+ -3.3109243697482205 x^2
+ 0.35294117647061857 x^3
```

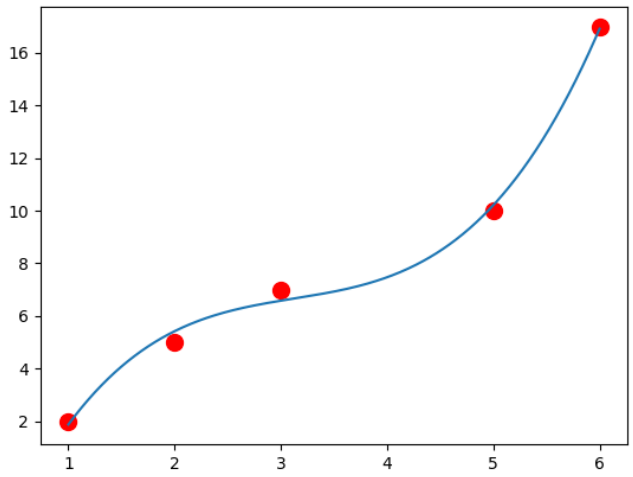

Figure 2: $(1,2),(2,5),(3,7),(5,10),(6,17)$ とその 3 次近似

3 可視化処理

- 1 行目：Jordan 法のプログラムライブラリ jordan_lib.py のインポート
- 2, 3 行目：グラフツールのインポート
- 4 行目：グラフ点数の定義
- 5~23 行目：前記プログラムとおんなじ
- 25 行目：Jordan 法の関数 solve_jordan() のコール
- 27~29 行目：近似多項式の数式表示
- 31~42 行目：近似多項式のグラフ表示

```
 1  import jordan_lib
 2  import numpy as np
 3  import matplotlib.pyplot as plt
 4  N = 400
 5  n, m = input().split(" ")
 6  n = int(n)
 7  m = int(m)
 8
 9  x = []
10  y = []
11  for i in range(n):
12      xx, yy = map(float, input().split(" "))
13      x.append(xx)
14      y.append(yy)
15
16  a = [[0 for i in range(m+1)] for j in range(m)]
17
18  for i in range(m):
19      for j in range(m):
20          for k in range(n):
21              a[i][j]+= x[k]**(i+j)
22      for k in range(n):
23          a[i][m] += y[k]*x[k]**i
24
25  ans = jordan_lib.solve_jordan(m,a)
26
27  print("y=",end=" ")
28  for i in range(m):
29      print("+",ans[i],"x^",i)
30
31  xp = np.linspace(x[0],x[n-1],N+1)
32
33  z = []
34  for j in range(N+1):
35      zz = 0
36      for i in range(m):
```

```
37          zz += ans[i]*xp[j]**i
38      z.append(zz)
39
40  plt.scatter(x,y, s=100, c="red")
41  plt.plot(xp,z)
42  plt.show()
```

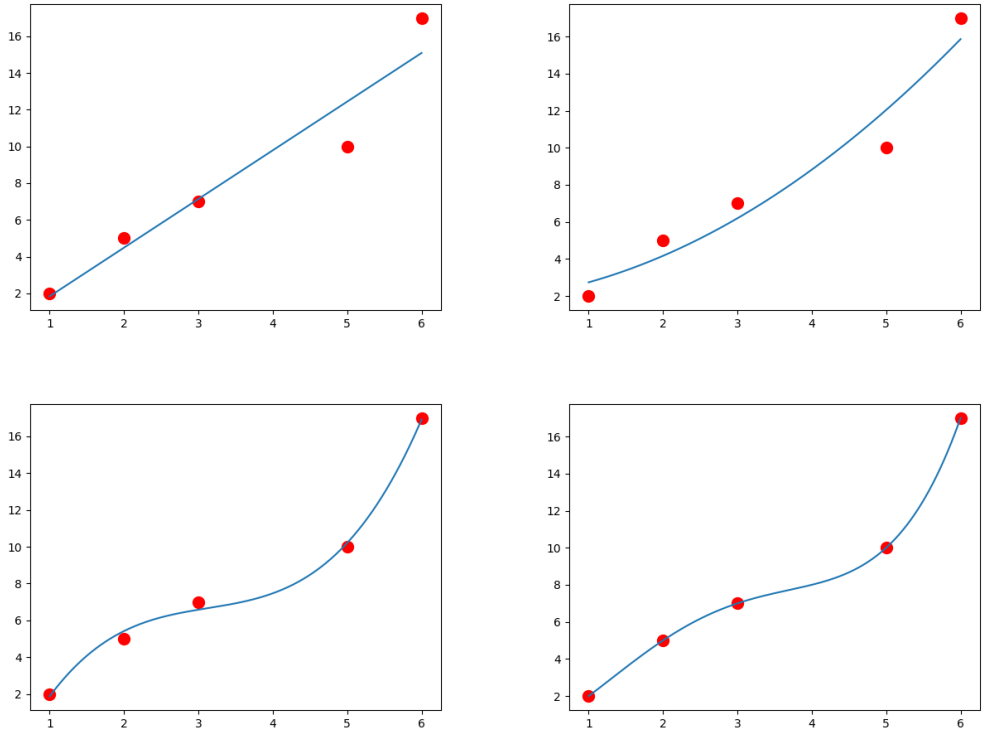

Figure 3: $(1,2),(2,5),(3,7),(5,10),(6,17)$ とその 1~4 次近似

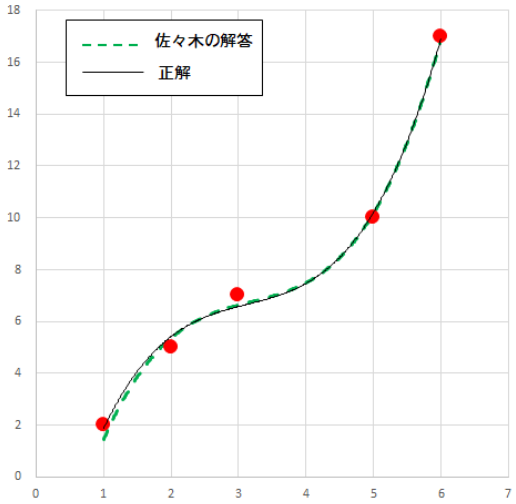

Figure 4: 答え合わせ

ちなみに $(1,2),(2,5),(3,7),(5,10),(6,17)$ の 4 次近似は，Lagrange 補間した場合と完全におんなじ．

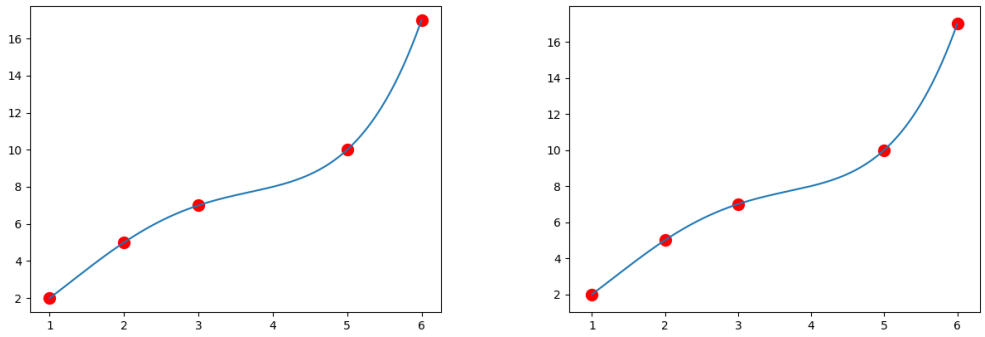

Figure 5: $(1,2),(2,5),(3,7),(5,10),(6,17)$ の 4 次近似 (左図) と Lagrange 補間 (右図)