

수치해석 HW #4-2

Numerical analysis

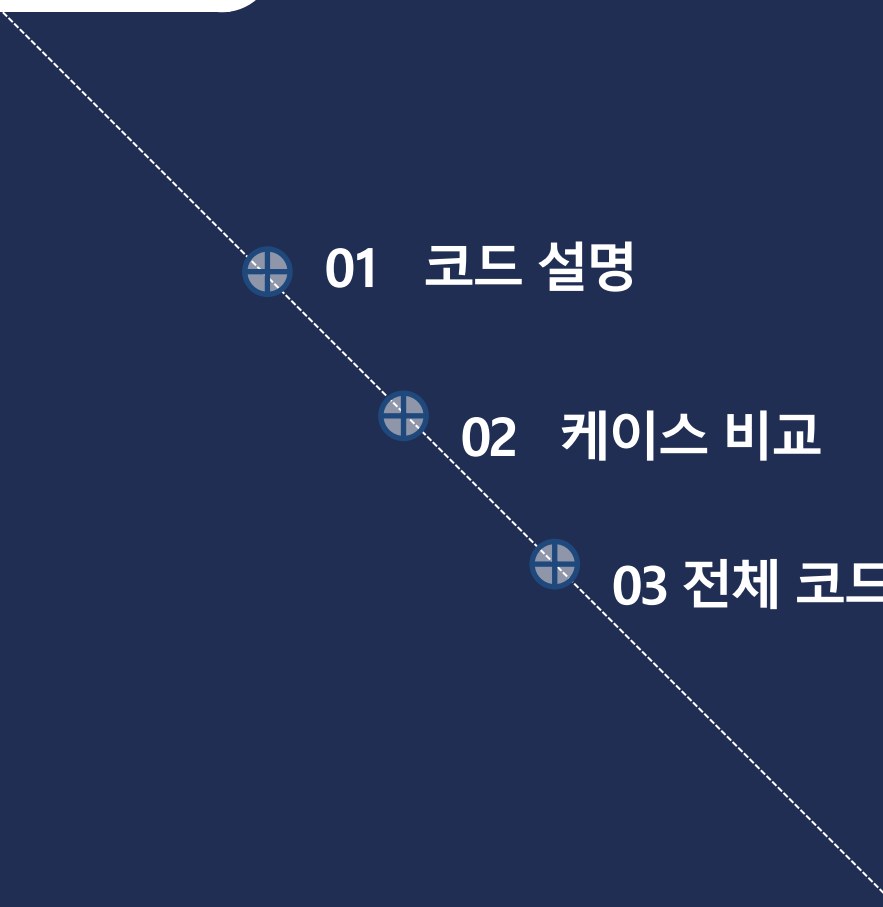
JaeHoon KANG

강 재 훈

HW #4-2

Curve fitting using Least squares

CONTENTS

- 
- 01 코드 설명
 - 02 케이스 비교
 - 03 전체 코드

```
import numpy as np
import random
import itertools
```

사용된 library

Numpy = 행렬의 transpose와 inverse를 구하기 위해 사용된 library

Random = 6개의 point를 랜덤으로 뽑기 위해 random한 index를 만들기위해 사용될 randint를 포함하는 library

Itertools = 8개의 point들로 만들 수 있는 모든 경우의 수를 만들어줄 combination 함수를 포함하고 있는 library

01

HW #4-2

코드 설명

```
def ROW(x) :  
    l = [x**2, x, 1]  
    return l  
  
def pseudoInverse(A,b):  
    AtA = np.dot( np.transpose(A), A )  
    AtA_1At = np.dot( np.linalg.inv(AtA), np.transpose(A) )  
    x = np.dot(AtA_1At,b)  
    return x
```

사용된 함수 설명

Row = parameter로 받은 x값을 이용해

행렬 A의 row 한줄을 만들어줄 함수

pseudoinverse = $(A^T A)^{-1} A^T b$ 를 계산하여

x값을 유추해줄 함수

사용될 변수 설정

```
points = [ [-2.9, 35.4], [-2.1, 19.7], [-0.9, 5.7], [1.1, 2.1], [0.1, 1.2], [1.9, 8.7], [3.1, 25.7], [4.0, 41.5] ]  
C = list(itertools.combinations(points,6))  
idx1, idx2 = 0, 0  
l = len(C)-1  
while idx1 == idx2 :  
    idx1, idx2 = random.randint(0,l), random.randint(0,l)  
case1, case2 = C[idx1], C[idx2]  
A1, A2, b1, b2 = [], [], [], []  
  
for i in range(6):  
    A1.append(ROW(case1[i][0]))  
    b1.append(case1[i][1])  
    A2.append(ROW(case2[i][0]))  
    b2.append(case2[i][1])
```

points = 명세로 주어진 점 8개를 담고 있는 배열

C = points들을 6개씩 묶은 조합 생성

idx1, idx2 = case1과 case2에서 각각 다른 조합(경우의 수)를 선택하기 위해 존재하는 변수 (0부터 C의 length-1 만큼의 수 중 random한 서로 다른 값이 저장됨)

이후 선택된 점 6개로 행렬 A와 b를 만들어줌

최소제곱법을 이용한 Curve fitting

```
A1, A2, b1, b2 = np.array(A1), np.array(A2), np.array(b1), np.array(b2)
result1, result2 = pseudoInverse(A1,b1), pseudoInverse(A2,b2)
print("case1 points :",case1, "& result1 :",result1)
print("case2 points :",case2, "& result2 :",result2)
```

행렬 A와 b를 각각 numpy array로 변환시켜 주고 result1, result2에 각각의 case의 결과값 (도출된 a,b,c의 값)을 저장하고 출력해줌

02

HW #4-2

케이스 비교

```
case1 points : ([-2.9, 35.4], [-2.1, 19.7], [-0.9, 5.7], [1.1, 2.1], [3.1, 25.7], [4.0, 41.5]) & result1 : [ 3.18054193 -2.39915013  1.16085405]
case2 points : ([-0.9, 5.7], [1.1, 2.1], [0.1, 1.2], [1.9, 8.7], [3.1, 25.7], [4.0, 41.5]) & result2 : [ 3.02711584 -1.89923592  1.32758734]
```

Case1에선 [-2.9, 35.4], [-2.1, 19.7] [-0.9, 5.7],
[1.1, 2.1], [3.1, 25.7], [4.0, 41.5]의 6개의 점이 선택되었고

그 결과 $a = 3.18054193$

$b = -2.39915013$

$c = 1.16085405$ 가 나왔습니다.

따라서 소수 첫째자리에서 반올림 하면 curve는

$Y = 3x^2 - 2x + 1$ 이 됩니다.

02

HW #4-2

케이스 비교

```
case1 points : ([-2.9, 35.4], [-2.1, 19.7], [-0.9, 5.7], [1.1, 2.1], [3.1, 25.7], [4.0, 41.5]) & result1 : [ 3.18054193 -2.39915013  1.16085405]
case2 points : ([-0.9, 5.7], [1.1, 2.1], [0.1, 1.2], [1.9, 8.7], [3.1, 25.7], [4.0, 41.5]) & result2 : [ 3.02711584 -1.89923592  1.32758734]
```

Case2에선 [-0.9, 5.7], [1.1, 2.1], [0.1, 1.2]
[1.9, 8.7], [3.1, 25.7], [4.0, 41.5]의 6개의 점이 선택되었고

그 결과 $a = 3.02711584$

$b = -1.89923592$

$c = 1.32758734$ 가 나왔습니다.

따라서 소수 첫째자리에서 반올림 하면 curve는

$Y = 3x^2 - 2x + 1$ 이 됩니다.

02

HW #4-2

케이스 비교

```
case1 points : [-2.9, 35.4], [-2.1, 19.7], [-0.9, 5.7], [1.1, 2.1], [3.1, 25.7], [4.0, 41.5]) & result1 : [ 3.18054193 -2.39915013  1.16085405]
case2 points : [-0.9, 5.7], [1.1, 2.1], [0.1, 1.2], [1.9, 8.7], [3.1, 25.7], [4.0, 41.5]) & result2 : [ 3.02711584 -1.89923592  1.32758734]
```

소수점 첫째자리에서 반올림을 했기 때문에 두 curve가 똑같아 보이지만 사실은 그렇지 않습니다. 왜냐하면 명세에 주어진 8개의 점은 애초에 $y = ax^2 + bx + 1$ 라는 한개의 곡선 위에 동시에 모두 존재 할 수 없기 때문입니다. 주어진 8개의 점 중 어느 것을 선택하더라도 일그러진 2차원 함수 밖에 구할 수 없습니다. 하지만 그 일그러진 정도가 크지 않기 때문에 8개의 점 중 어느 6개를 선택하더라도 비슷한 값이 나오게 됩니다.

03

HW #4-2

전체 코드

```

import numpy as np
import random
import itertools
def ROW(x) :
    l = [x**2, x, 1]
    return l
def pseudoInverse(A,b):
    AtA = np.dot( np.transpose(A), A )
    AtA_1At = np.dot( np.linalg.inv(AtA), np.transpose(A) )
    x = np.dot(AtA_1At,b)
    return x

points = [ [-2.9, 35.4], [-2.1, 19.7], [-0.9, 5.7], [1.1, 2.1], [0.1, 1.2], [1.9, 8.7], [3.1, 25.7], [4.0, 41.5] ]
C = list(itertools.combinations(points,6))
idx1, idx2 = 0, 0
l = len(C)-1
while idx1 == idx2 :
    idx1, idx2 = random.randint(0,l), random.randint(0,l)
case1, case2 = C[idx1], C[idx2]
A1, A2, b1, b2 = [], [], [], []

for i in range(6):
    A1.append(ROW(case1[i][0]))
    b1.append(case1[i][1])
    A2.append(ROW(case2[i][0]))
    b2.append(case2[i][1])

A1, A2, b1, b2 = np.array(A1), np.array(A2), np.array(b1), np.array(b2)
result1, result2 = pseudoInverse(A1,b1), pseudoInverse(A2,b2)
print("case1 points :",case1, "& result1 :",result1)
print("case2 points :",case2, "& result2 :",result2)

```

감사합니다

THANK YOU

JaeHoon KANG

강 재 훈