

알고리즘 실습 보고서

-Segmented Least Squares-

전공 :컴퓨터공학과

분반 : 05반

학번 :201701988

이름 :김수빈

1. 실행 환경

본 실습은 Windows10 64bit, jdk 1.8.0_221, Eclipse EE가 설치된 환경에서 실행되었다

2. 과제 설명

- 출제된 과제에 대한 설명

이번 과제 Segmented Least Squares는 data07.txt 파일을 읽어 $p_1(x_1, y_1)$, $p_2(x_2, y_2)$, ..., $p_n(x_n, y_n)$ 으로 구성된 n 개의 점들을 이을 Square Error를 최소화시키는 직선들을 찾는 문제다.

3. 문제 해결 방법

- 문제를 해결하기 위해 자신이 사용한 방법, 아이디어에 대한 설명

문제 해결

먼저, 점의 x 좌표, y 좌표를 표현하기 위해, double형 값 x 와 y 를 갖는 Point 객체와, 또 OPT(j)에서의 segment로 나누어진 지점, OPT(j)에서의 Least Square값, Segment의 수를 표현하기 위해 OPT 객체를 생성했다.

주어진 문제를 해결하기 위해 필요한 메소드는 주어진 점들을 여러 segment로 분할하여 최소 Square Error를 반환하는 Segmented_Least_Squares 메소드가 있다. 또 p_1 에서 p_2 까지의 점들을 지나는 최소 Square Error를 갖는 하나의 직선($y = a \times x + b$)의 a 와 b 값 그리고 그때의 최소 Square Error를 double형 배열로 묶어 반환하는 메소드인 SSE가 있다.

data07.txt 파일을 읽어 Point 배열을 구성하고, segment 추가할 때의 패널티 값인 c 를 얻는다. Point 배열과 c 를 매개변수로 하여 Segmented_Least_Squares 메소드를 호출한다.

Segmented_Least_Squares 메소드에서는 첫 번째 점부터 각 점들까지의 OPT 값을 배열로 저장하는 M 과, 포인트 i 부터 j 까지 하나의 직선을 사용했을 때의 최소 Square Error를 저장하는 2차원 배열 E 를 사용하여 문제를 해결한다. 각 포인트 j ($1 \leq j \leq \text{point 개수}$)에 다음을 반복한다. 0부터 j 보다 작은 i 에 대해 포인트 i 부터 $j-1$ 까지 하나의 직선으로 이을 때의 최소 Square Error를 $E[j][i]$ 에 저장한다. 이때 SSE를 호출하여 최소 Square Error를 구하므로, 삼중 loop를 사용한다.

$1 \leq j \leq \text{point 개수}$ 인 j 에 대해 다음을 반복한다. point 0부터 j 까지 이을 선들이 최소 Square Error를 가지는 Segment 분할 지점 i 를 찾는다. 각 Square Error는 $E[j][i] + c + M[i]$ 의 최소 Square Error로 구한다. 그 최솟값을 $M[j]$ 의 최소 Square Error값으로 하고, 최솟값을 갖는 i 를 $M[j]$ 의 Segment 분할 지점으로 하고, $M[j]$ 의 segment 개수를 분할지점 OPT 값의 segment개수에서 1 증가시킨 값으로 한다.

결과 화면 출력

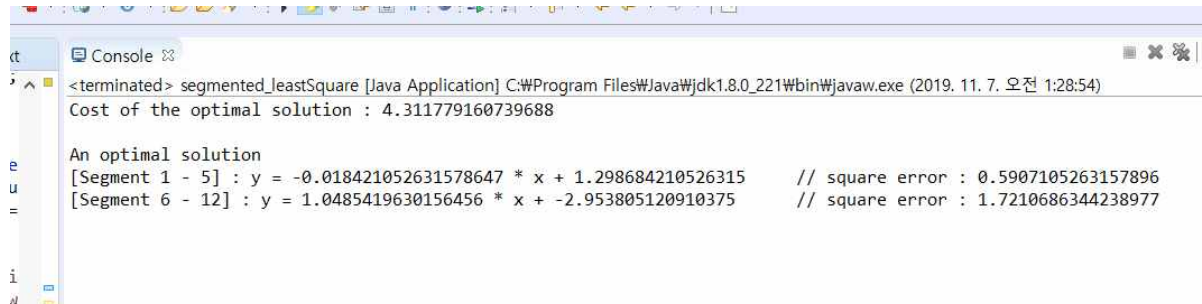
$M[M.length-1]$ 에서의 각 segment를 출력하기 위해 segment 개수 by 3만큼의 2차원 배열을 생성한다. pre_brokenPoint는 segment 시작 직전 지점이며, post_brokenPoint는 segment 끝 지점 인덱스를 의미한다. 각각의 초기 값은 0과 point 개수이다.

$M[M.length-1]$ 에서의 segment 개수만큼 아래를 반복한다.

반복문을 통해 post_brokenPoint가 pre_brokenPoint 보다 클 때까지 post_brokenPoint를 $M[\text{post_brokenPoint}]$ 의 segment 분할지점으로 한다. 즉, pre_brokenPoint에 가장 가까운 분할 지점을 찾는다. SSE를 호출하여 segment의 최소 Square Error와 직선의 a 와 b 값을 출력하

고, pre_brokenPoint를 post_brokenPoint로 옮기고 post_brokenPoint를 다시 point 개수로 한다.

결과 화면



```
<terminated> segmented_leastSquare [Java Application] C:\Program Files\Java\jdk1.8.0_221\bin\javaw.exe (2019. 11. 7. 오전 1:28:54)
Cost of the optimal solution : 4.311779160739688

An optimal solution
[Segment 1 - 5] : y = -0.018421052631578647 * x + 1.298684210526315 // square error : 0.5907105263157896
[Segment 6 - 12] : y = 1.0485419630156456 * x + -2.953805120910375 // square error : 1.7210686344238977
```

5. 느낀점 및 고찰

psedo code와 인덱스 시작점이 달라 많이 헷갈렸습니다. 또 data07.txt에서는 두 개의 segment밖에 없어서, 제 코드가 다른 데이터 상에서도 제대로 동작하는지 확인해보기 위한 테스트 데이터들을 직접 만들어봐야 하는 부분이 힘들었습니다.