

Segmented Least Squares

Input: n 2-D points $(x_1, y_1) \dots (x_n, y_n)$, $C > 0$.

$$x_1 < x_2 < \dots < x_n$$

Output: Given a line L defined by $y = ax + b$
the error of a set of points P w.r.t L is

$$\underline{\text{err}}(L, P) = \sum_{i \in P} (y_i - ax_i - b)^2$$

The goal is to partition $\{1, 2, \dots, n\}$ into segments,
where each segment is a consecutive set of points

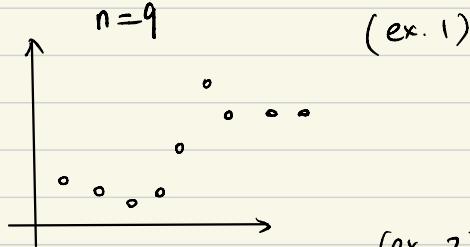
$$(P_i, P_{i+1}, \dots, P_{j-1}, P_j)$$

and minimize $\underline{\text{penalty}} = \underline{\text{①}} + \underline{\text{②}}$

$$\textcircled{1} (\# \text{ of segments}) \times C \quad \textcircled{2} = \sum_{\text{all seg. } L} \text{min. error}(L, (P_i \dots P_j))$$

$\textcircled{2}$ for each segment, the error value of the
optimal line for that segment.

Example :



(ex. 1)

$$C = 0$$

A: have n segments

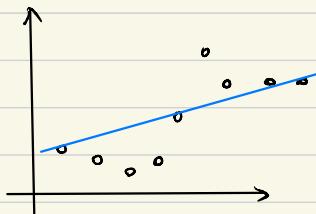
error is 0 for a single point.
penalty = 0.

$$(ex. 2) \quad C = 10^9 \quad (\gg x_i \text{ or } y_i)$$

A: use one segment.

$$a = \frac{n \sum_i x_i y_i - (\sum_i x_i)(\sum_i y_i)}{n \sum_i x_i^2 - (\sum_i x_i)^2}$$

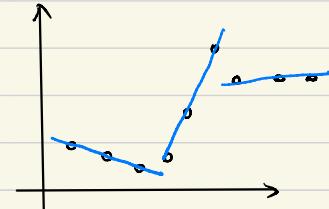
$$b = \frac{\sum_i y_i - a \sum_i x_i}{n}$$



(ex. 3) moderate value of $C = 0.01$

$$\text{penalty} = 0.03$$

for having 3 segments



A dynamic-programming Solution

The last point p_n must belong to some segment
 $(p_i, p_{i+1}, \dots, p_n)$ (*)

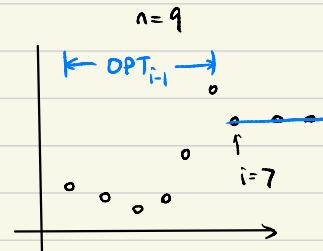
e_{ij} = minimum error for the segment (p_i, \dots, p_j)
 $= \min_L \text{error}(L, (p_i, p_{i+1}, \dots, p_j))$

OPT_k = the minimum penalty for the first k points.

Final answer = OPT_n .

If (*) is true, $\text{OPT}_n = e_{i,n} + \text{OPT}_{i-1}$.

Remark. 1. e_{ij} can be computed efficiently
(closed-form formula)
2. we do know $i \Rightarrow$ we enumerate/guess i .



For the subproblem on (p_1, p_2, \dots, p_k)

$$\text{OPT}_k = \min_{1 \leq i \leq k} (e_{i,k} + C + \text{OPT}_{i-1}) \quad \text{OPT}_0 = 0. \quad (\text{**})$$

We can turn this into an algorithm.

Runtime

$O(n^3) \leftarrow \begin{cases} \text{For } i = 1 \text{ to } n \\ \quad \text{For } j = i \text{ to } n \\ \quad \quad \text{compute } e_{i,j} \\ \quad \text{End for} \\ \text{End for} \\ \text{OPT}[0] = 0 \end{cases}$

$+ O(n^2) \leftarrow \begin{cases} \text{For } k = 1 \text{ to } n. \\ \quad \text{Compute } \text{OPT}[k] \text{ using (**).} \\ \quad \quad \text{(which requires a for loop)} \\ \quad \text{End for} \\ \text{Return } \text{OPT}[n] \end{cases}$

$O(n^3)$