Plan for today

Quiz

Go over quiz

Last challenge problem: segmented least squares

1. (3 points) Suppose you get input prices for days 1 through 10 as

$$797, 309, 850, 472, 398, 419, 383, 955, 279, 619$$

.

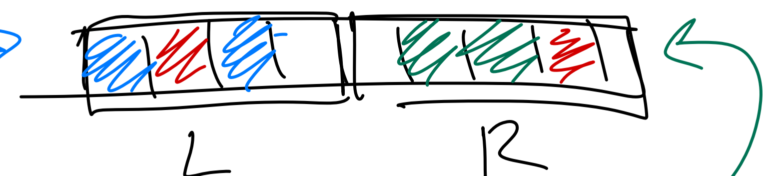
What is the maximum profit for this input?

646

2. (3 points) We would like to write a divide and conquer algorithm for this problem. Just as we have done for many other problems, we will do this by dividing the array in half, using recursion to find the optimal solution in each half, and then finding a way to merge the optimal solutions from each half into one single optimal solution.

Let the array you are trying find the max profit for be A. If you divide array in A into two halves, L and R (for left and right), the max profit is made in exactly one of these ways:

- Buying and selling on days in L
- Buying and selling on days in R
- ??? Buy in L, sell in R



What is the third way that you could make the max profit from the prices in A?

3. (3 points) What is a base case for the max profit problem? You may assume that the number of days is a power of 2 if you would like.

n=1 : return 0

[100]

---

n=2 return max(price day$_2$ - prize day$_1$, 0)

1 2

[300, 250]

Max_Profit($p_1, p_2, \ldots, p_n$):

if $n = 2$:

  return max(0, $p_2 - p_1$)

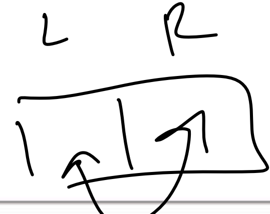
else:

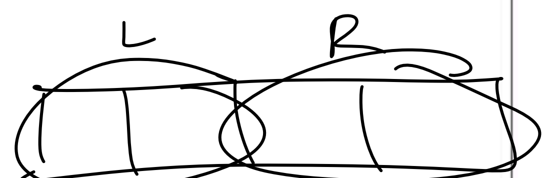
  m_L = Max_Profit(L half of p_1 to p_n)

  m_R = Max_Profit(R half of p_1 to p_n)

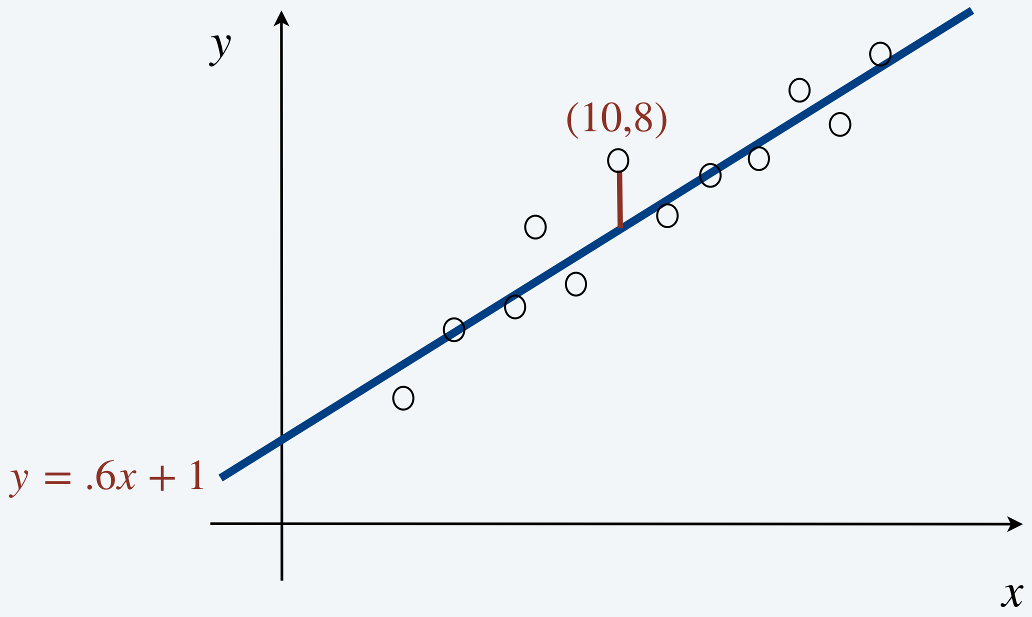
  m_X = max(R) − min(L)

  return max(m_L, m_R, m_X)



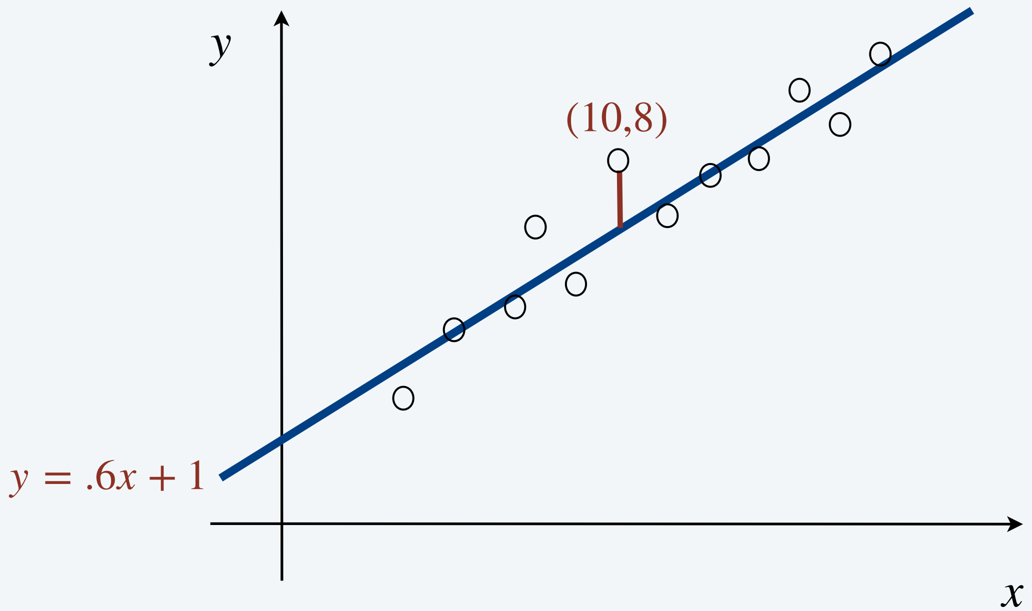


Least squares



Least squares

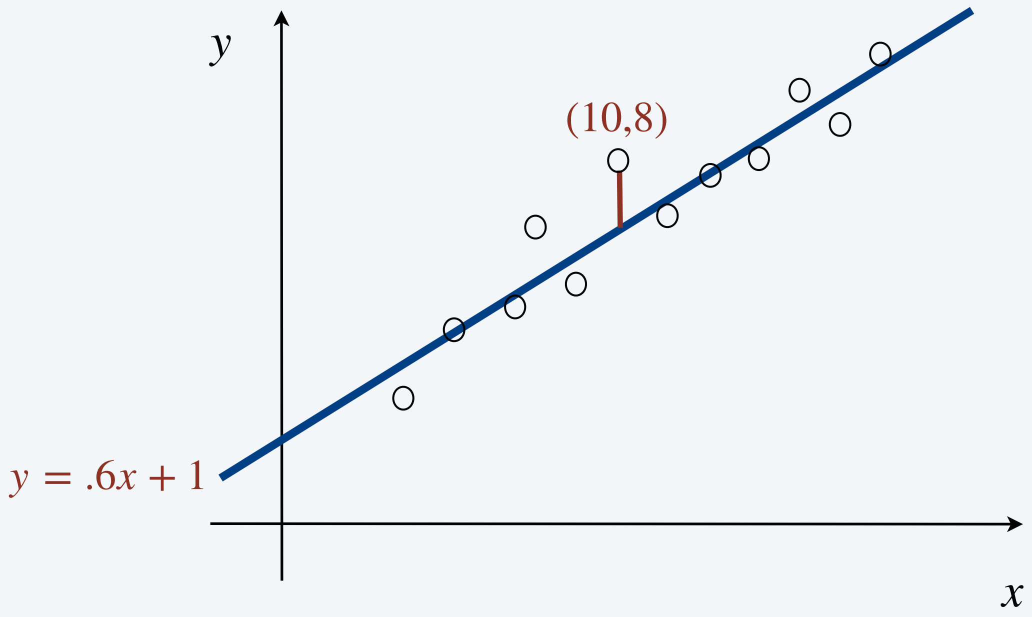
Least squares. Foundational problem in statistics.



Least squares

Least squares. Foundational problem in statistics.

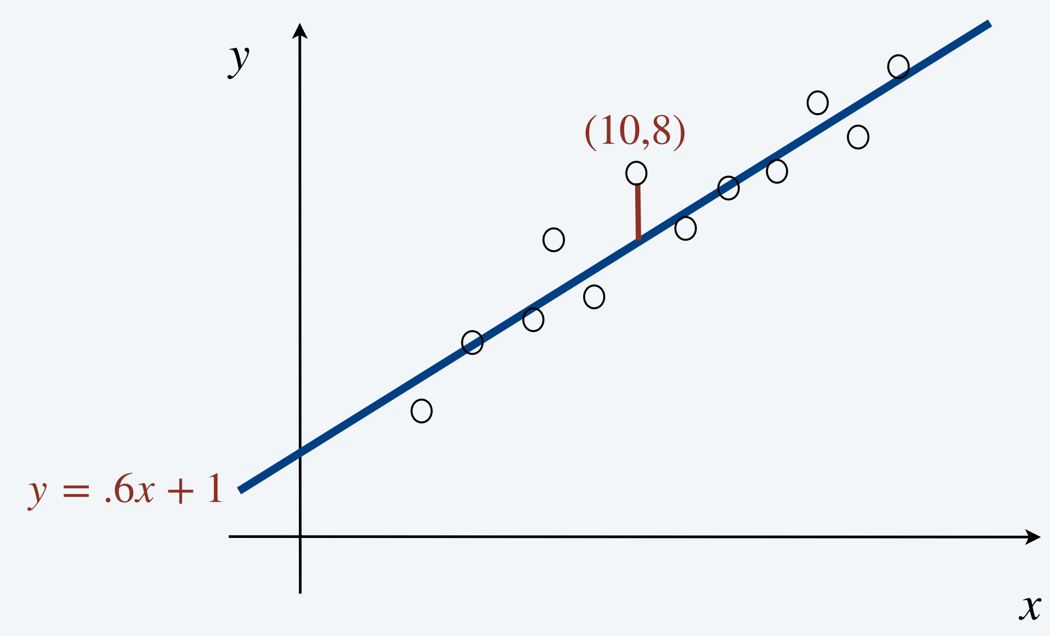
- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.



Least squares

Least squares. Foundational problem in statistics.

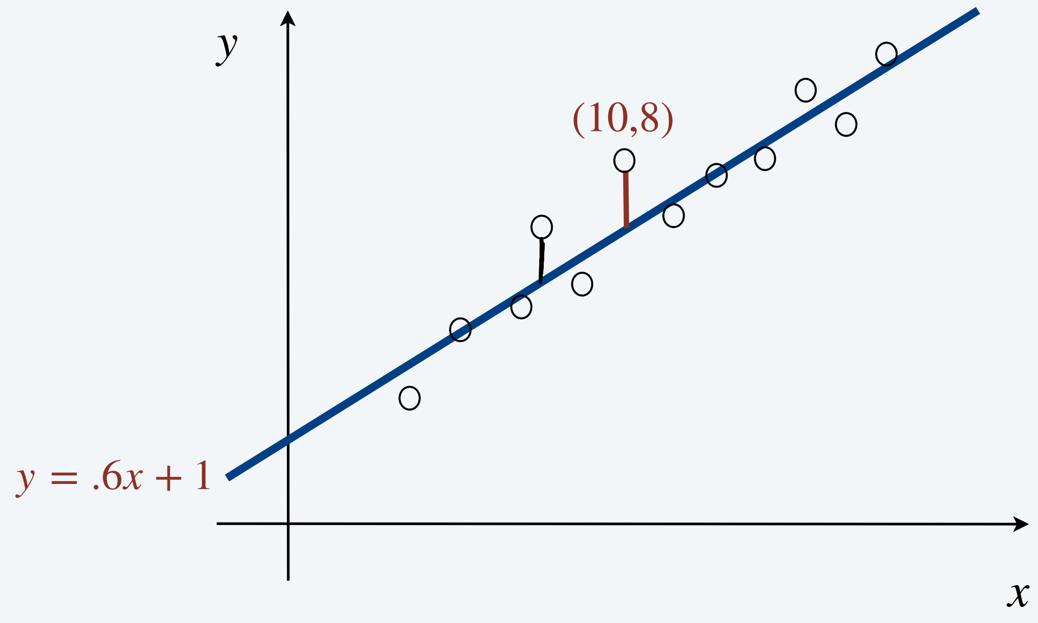
- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- Find a line $y = ax + b$ that minimizes the sum of squared error



Least squares

Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$



Least squares

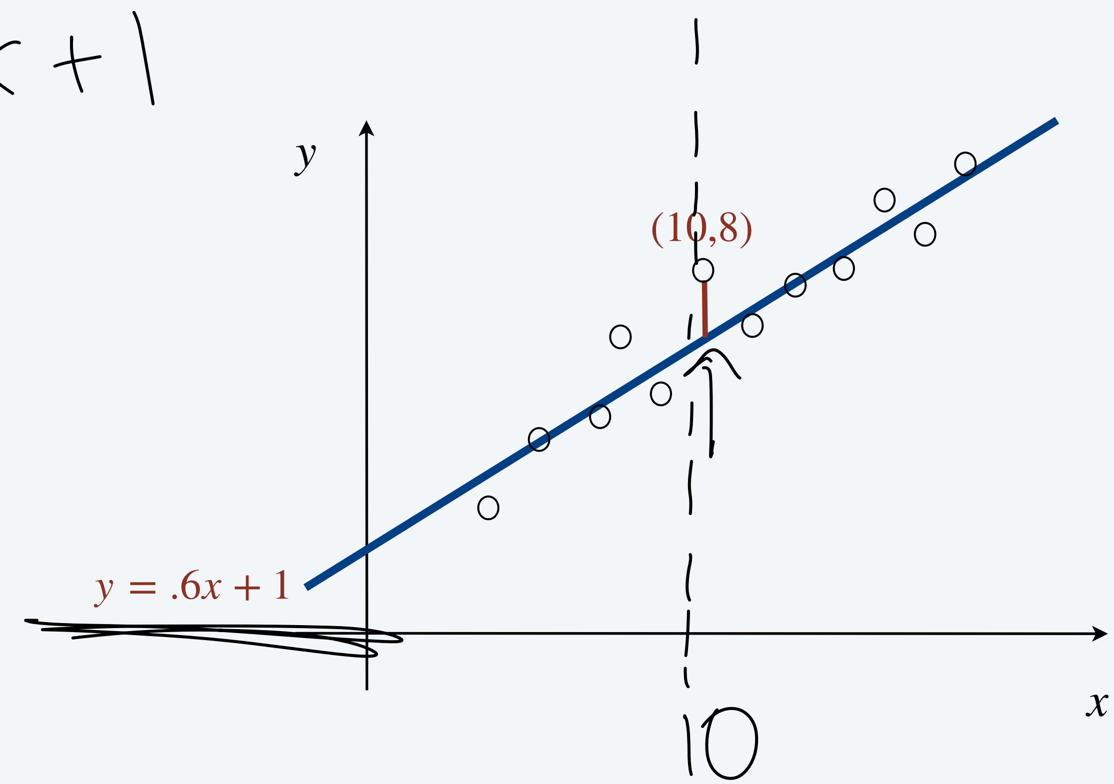
Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$
- What is squared error for this point? (On own, then check with table)

(10, 8)
x coord y coord

$y = .6x + 1$

$(8 - 7)^2 = 1^2 = 1$



Least squares

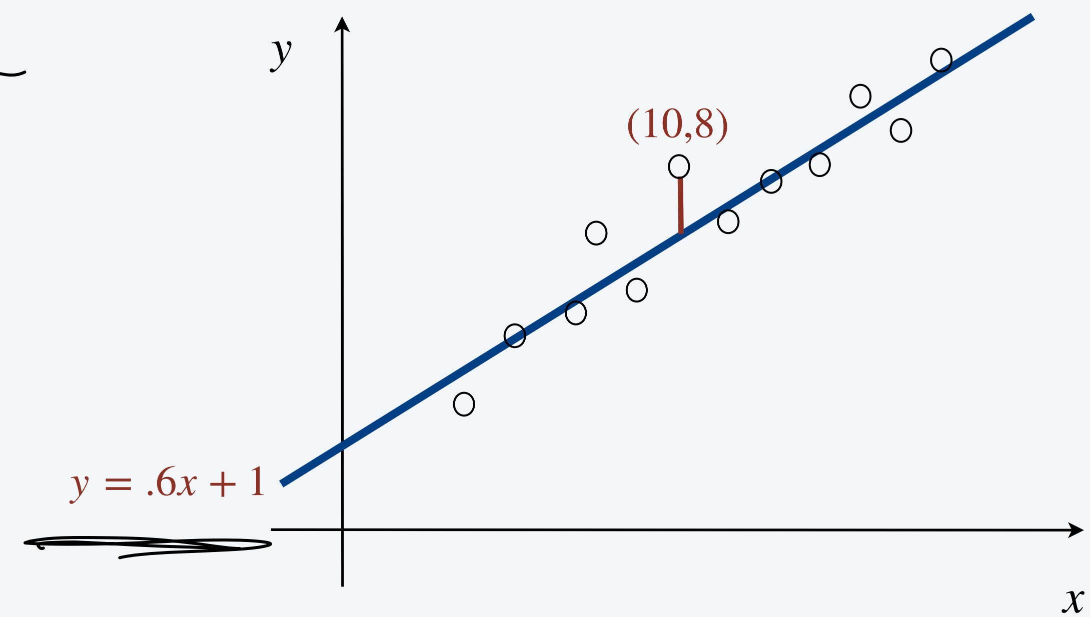
Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$
- What is squared error for this point? (On own, then check with table)

given point y val

y val of line

$$SSE = \sum_{i=1}^{n} (y_i - ax_i - b)^2$$

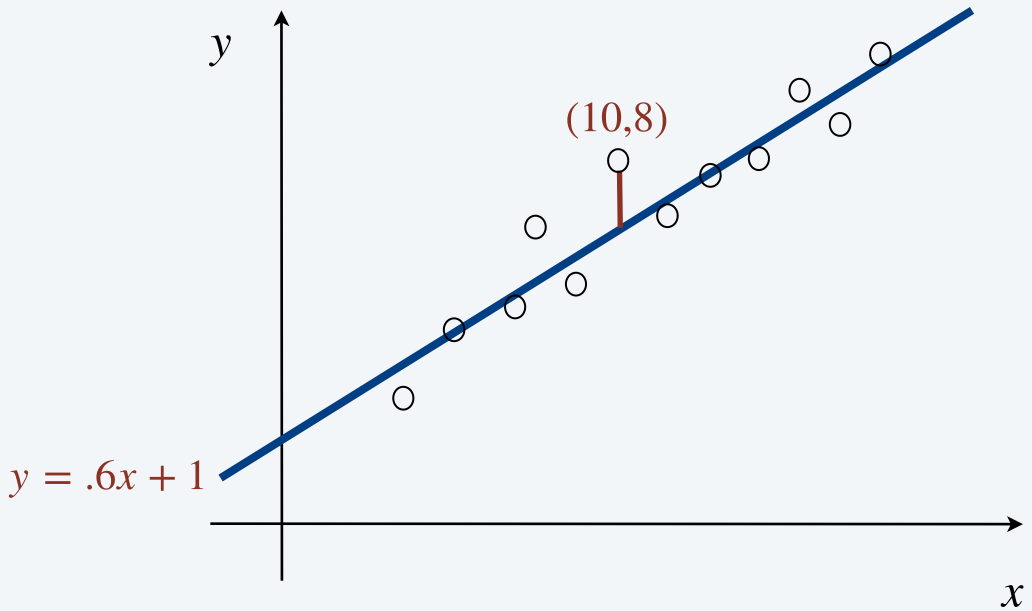


Least squares

Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2)$, …, (x_n, y_n).
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$
- What is squared error for this point? (On own, then check with table)

$$SSE = \sum_{i=1}^{n} (y_i - ax_i - b)^2$$



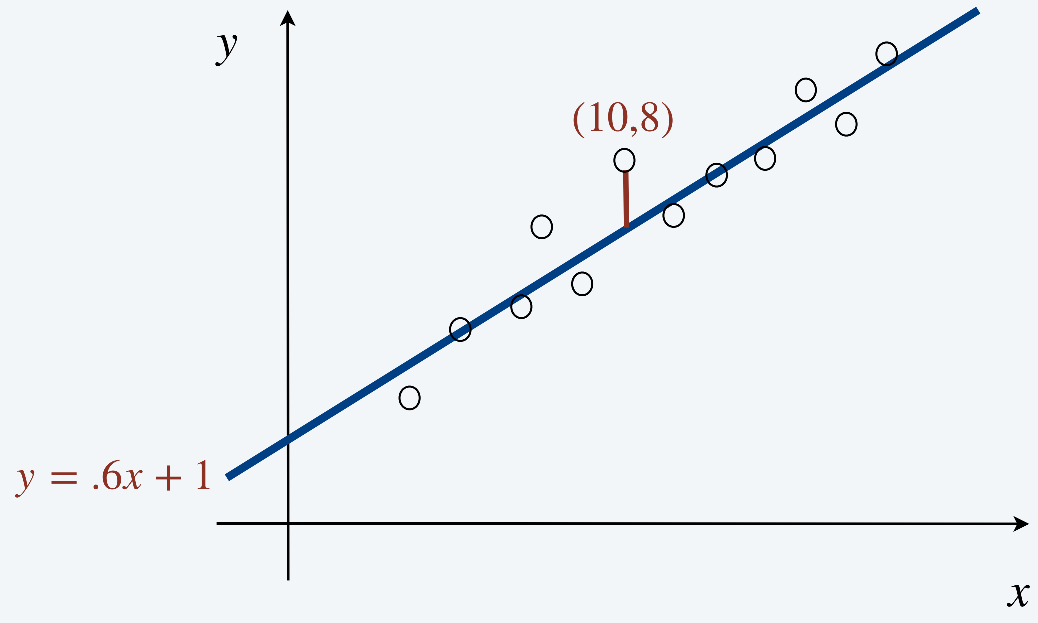
Solution. Calculus $\Rightarrow$ min error is achieved when

Least squares

Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$
- What is squared error for this point? (On own, then check with table)

$$SSE = \sum_{i=1}^{n} (y_i - ax_i - b)^2$$



Solution. Calculus $\Rightarrow$ min error is achieved when

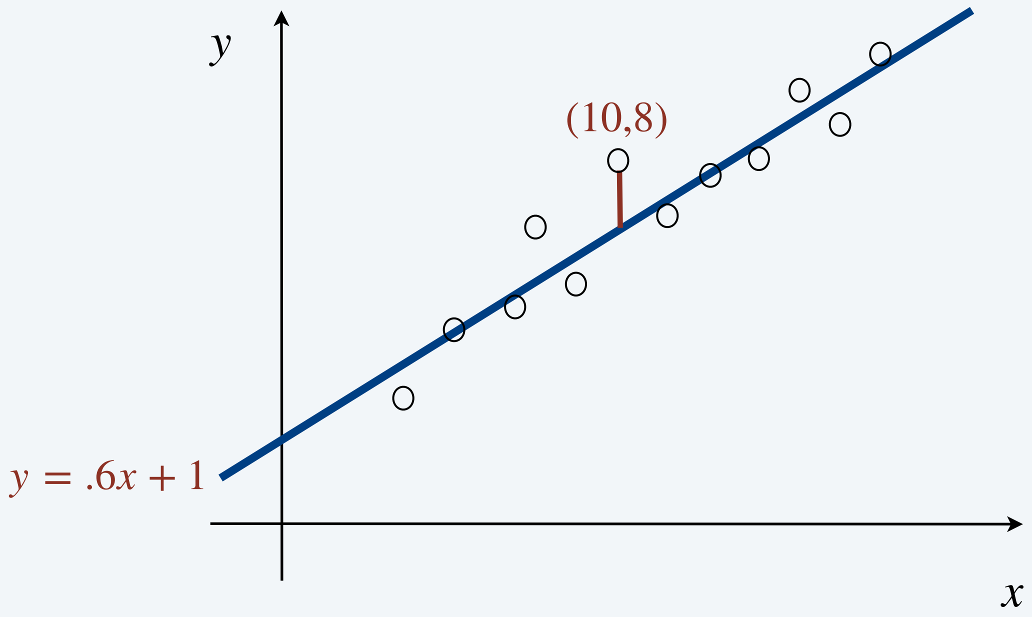
$$a = \frac{n \sum_i x_i y_i - (\sum_i x_i)(\sum_i y_i)}{n \sum_i x_i^2 - (\sum_i x_i)^2}, \quad b = \frac{\sum_i y_i - a \sum_i x_i}{n}$$

Least squares

Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$
- What is squared error for this point? (On own, then check with table)

$$SSE = \sum_{i=1}^{n} (y_i - a x_i - b)^2$$



Solution. Calculus $\Rightarrow$ min error is achieved when

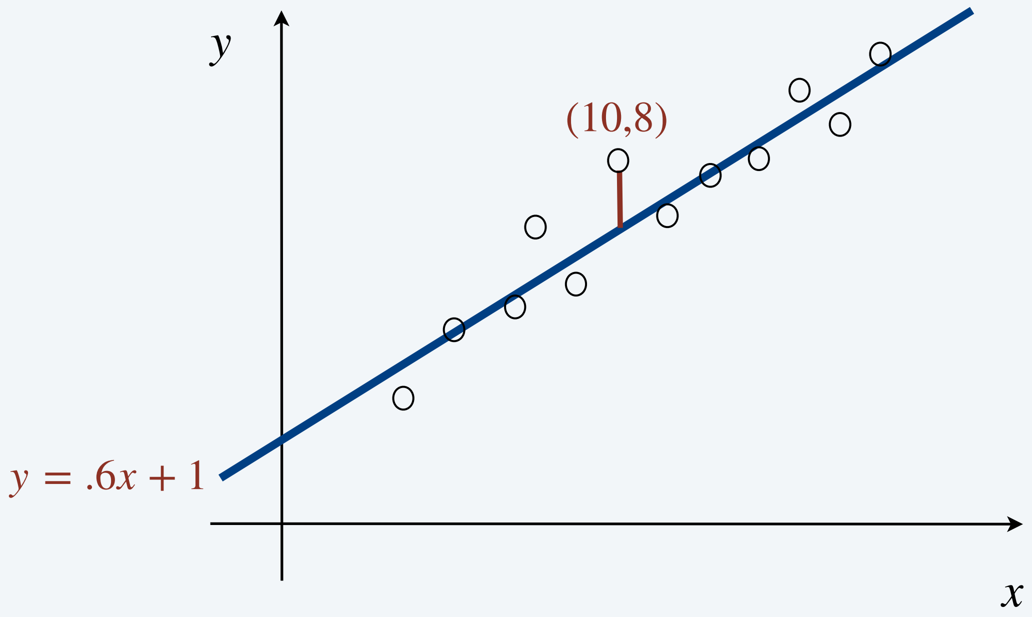
$$a = \frac{n \sum_i x_i y_i - (\sum_i x_i)(\sum_i y_i)}{n \sum_i x_i^2 - (\sum_i x_i)^2}, \quad b = \frac{\sum_i y_i - a \sum_i x_i}{n}$$

Least squares

Least squares. Foundational problem in statistics.

- Given n points in the plane: $(x_1, y_1), (x_2, y_2)$, …, (x_n, y_n).
- Find a line $y = ax + b$ that minimizes the sum of squared error
- Error just in y coordinates: $(y_1 - y_2)^2$
- What is squared error for this point? (On own, then check with table)

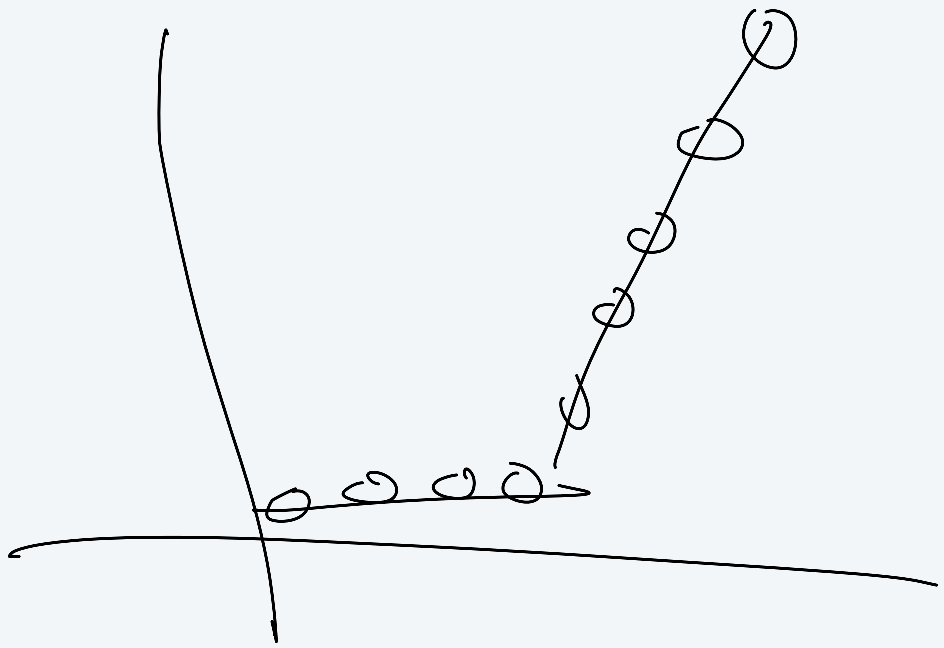
$$SSE = \sum_{i=1}^{n} (y_i - ax_i - b)^2$$

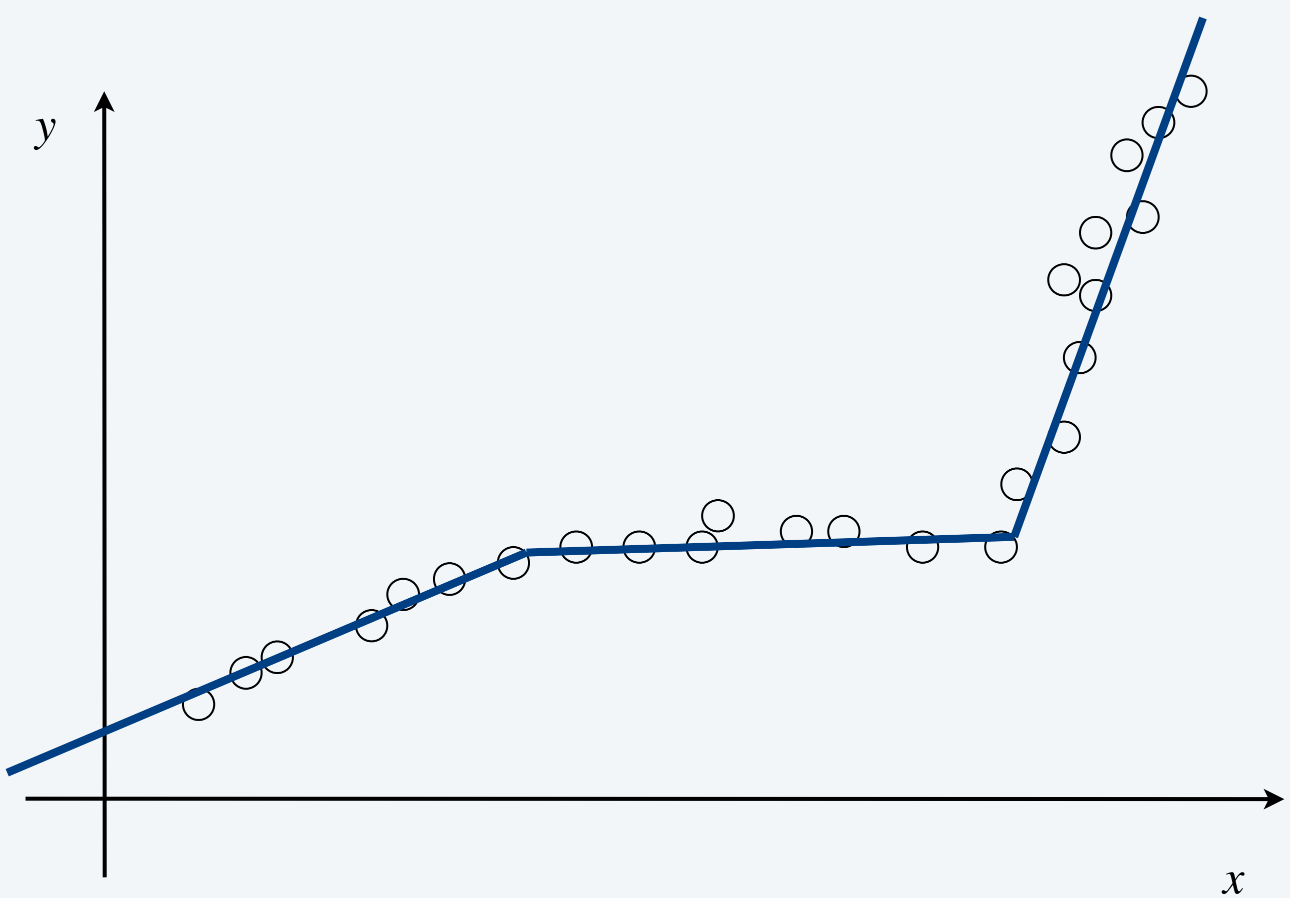


Solution. Calculus $\Rightarrow$ min error is achieved when

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

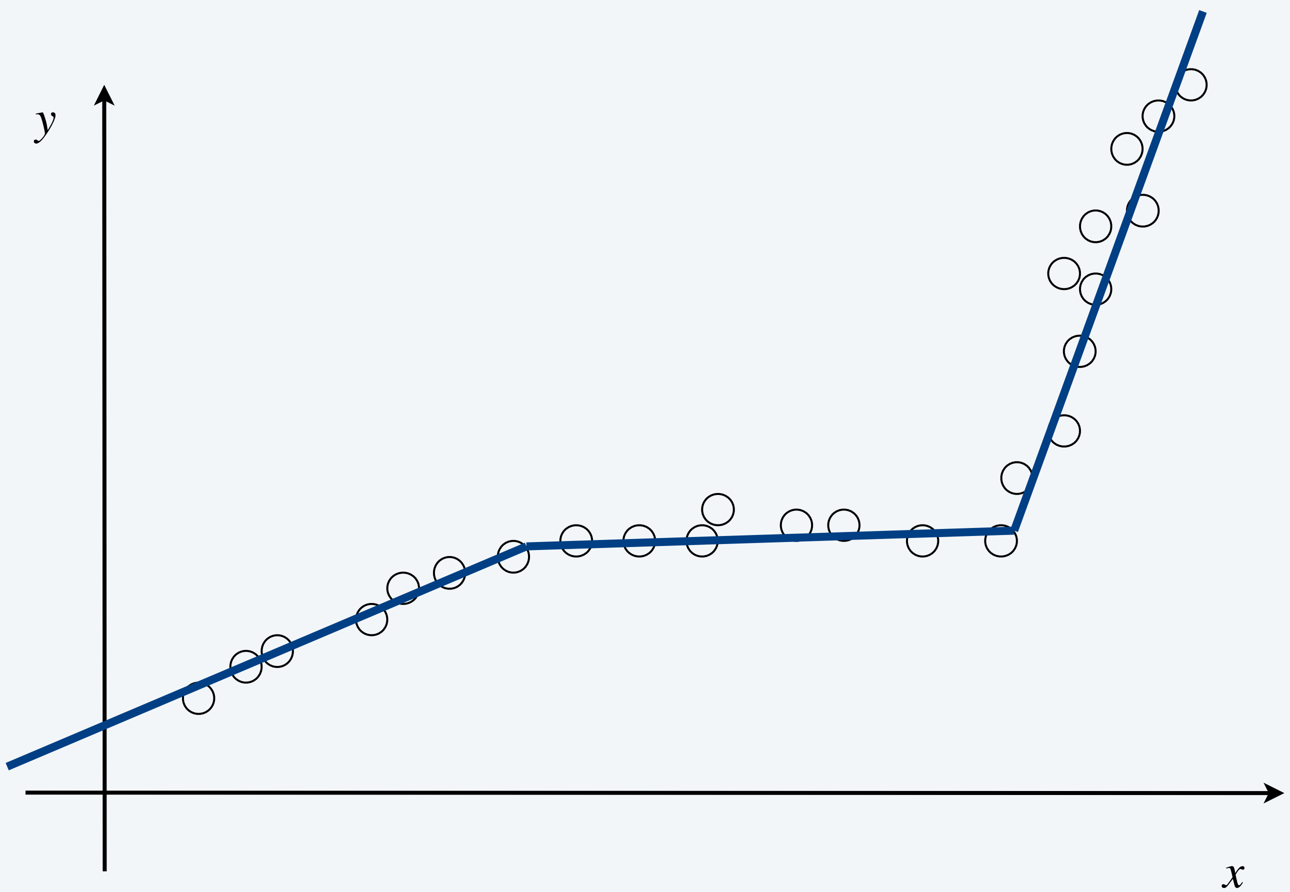
Segmented least squares





Segmented least squares

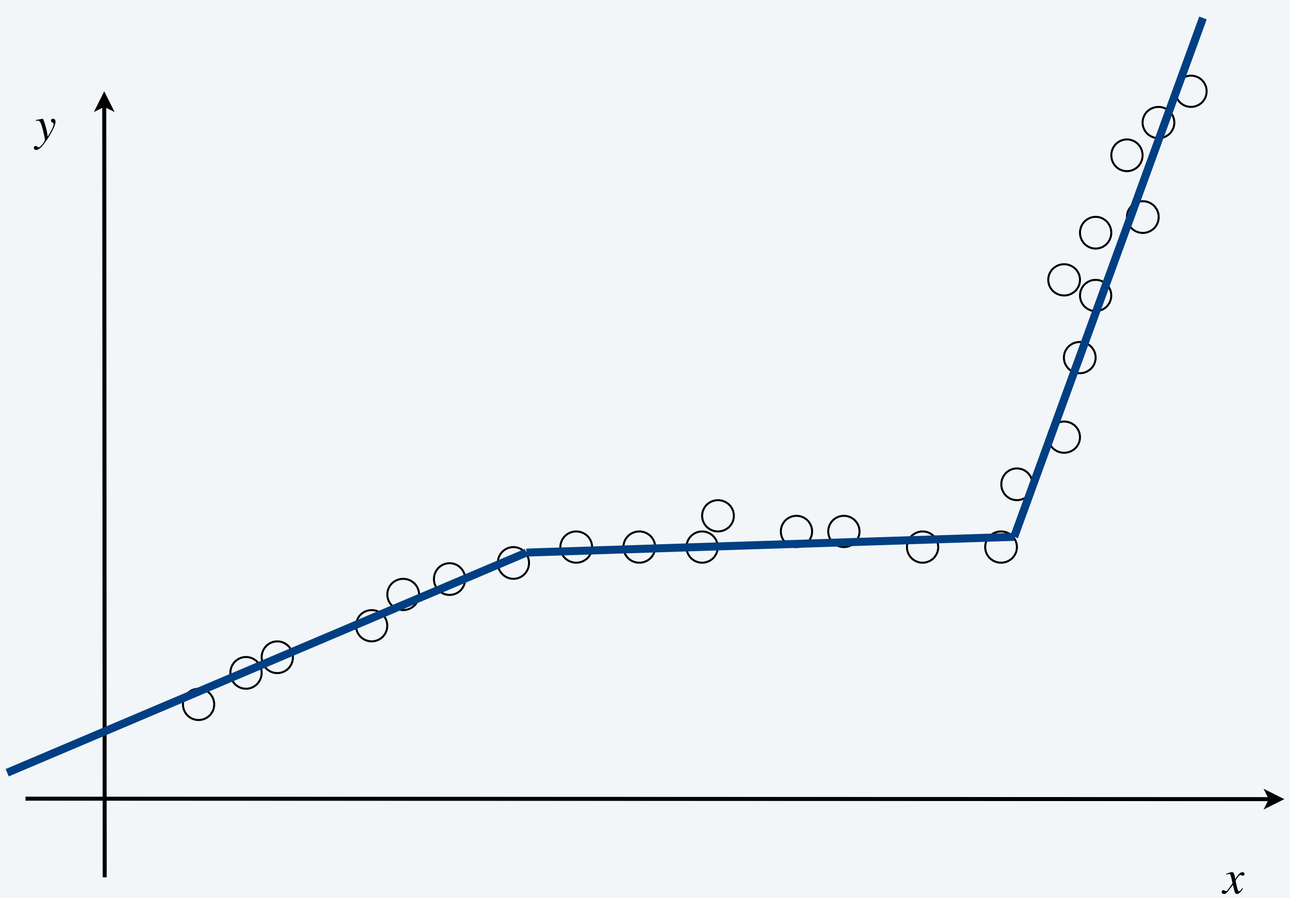
Segmented least squares.



Segmented least squares

Segmented least squares.

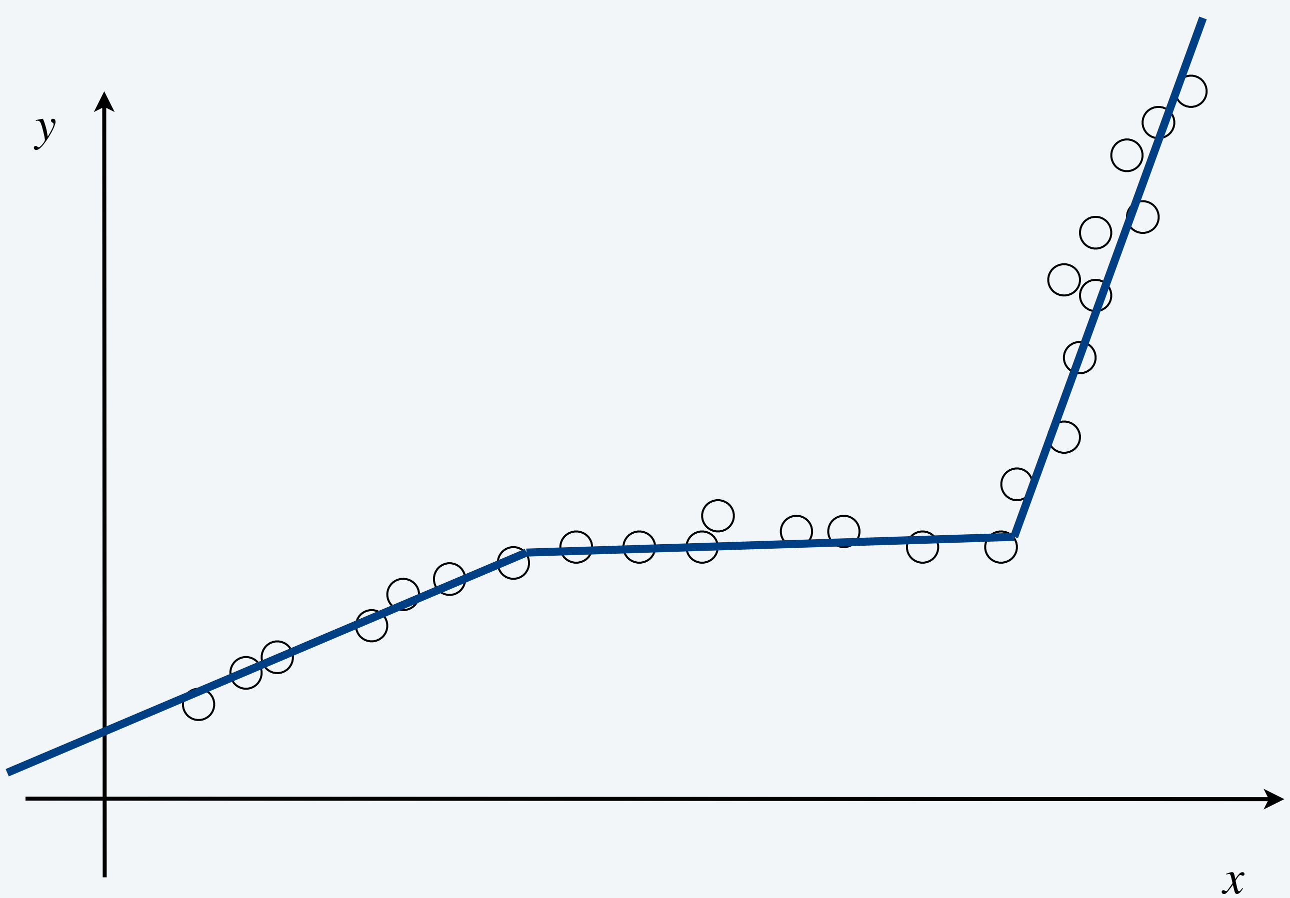
- Points lie roughly on a sequence of several line segments.



Segmented least squares

Segmented least squares.

- Points lie roughly on a sequence of several line segments.
- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ with $x_1 < x_2 < ... < x_n$, find a sequence of lines that minimizes error. What is the issue here?

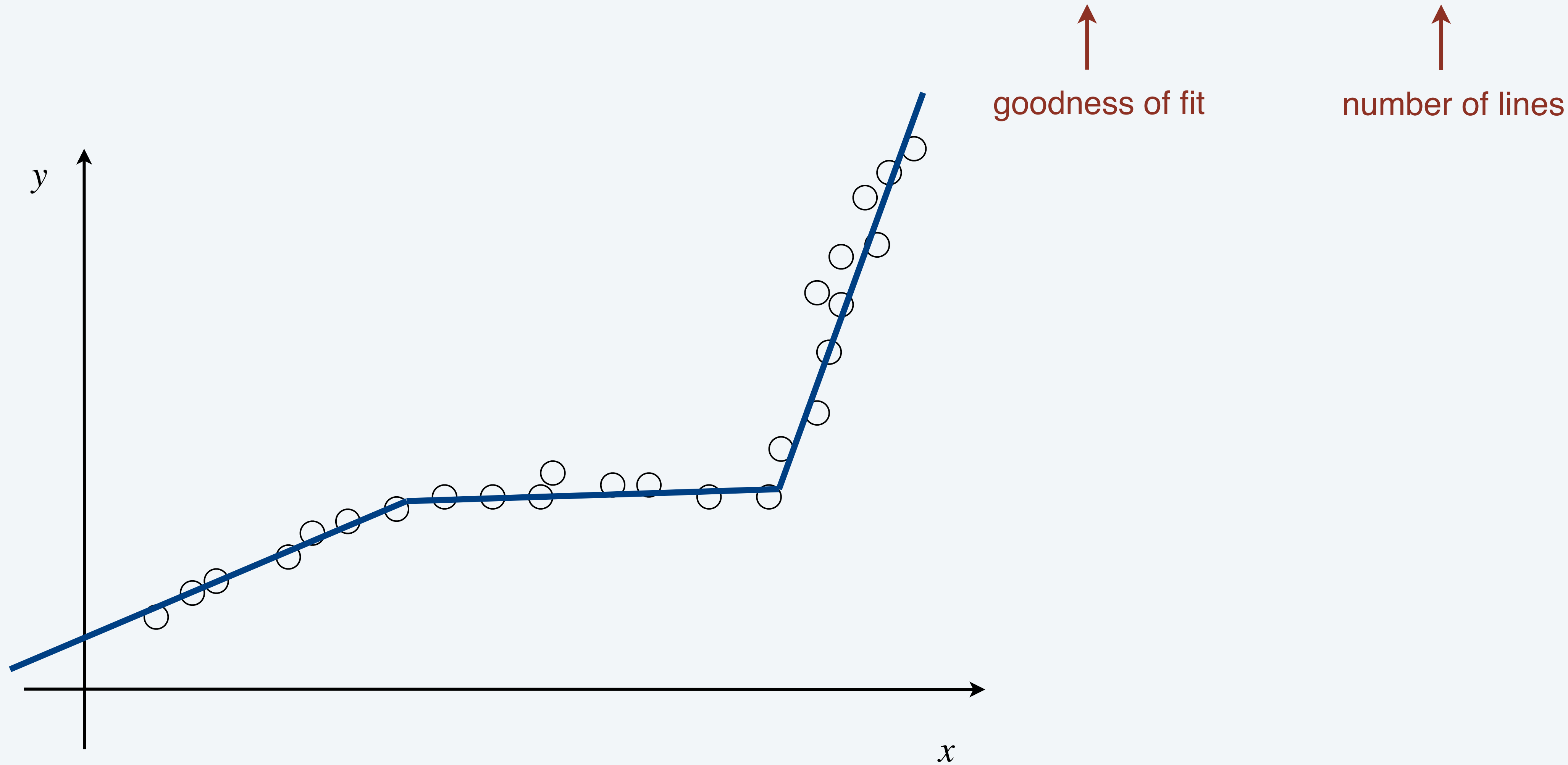


Segmented least squares

Segmented least squares.

- Points lie roughly on a sequence of several line segments.
- Given n points in the plane: $(x_1, y_1), (x_2, y_2)$, …, (x_n, y_n) with $x_1 < x_2 < ... < x_n$, find a sequence of lines that minimizes error. What is the issue here?

Q. What is a reasonable choice for $f(x)$ to balance accuracy and parsimony?



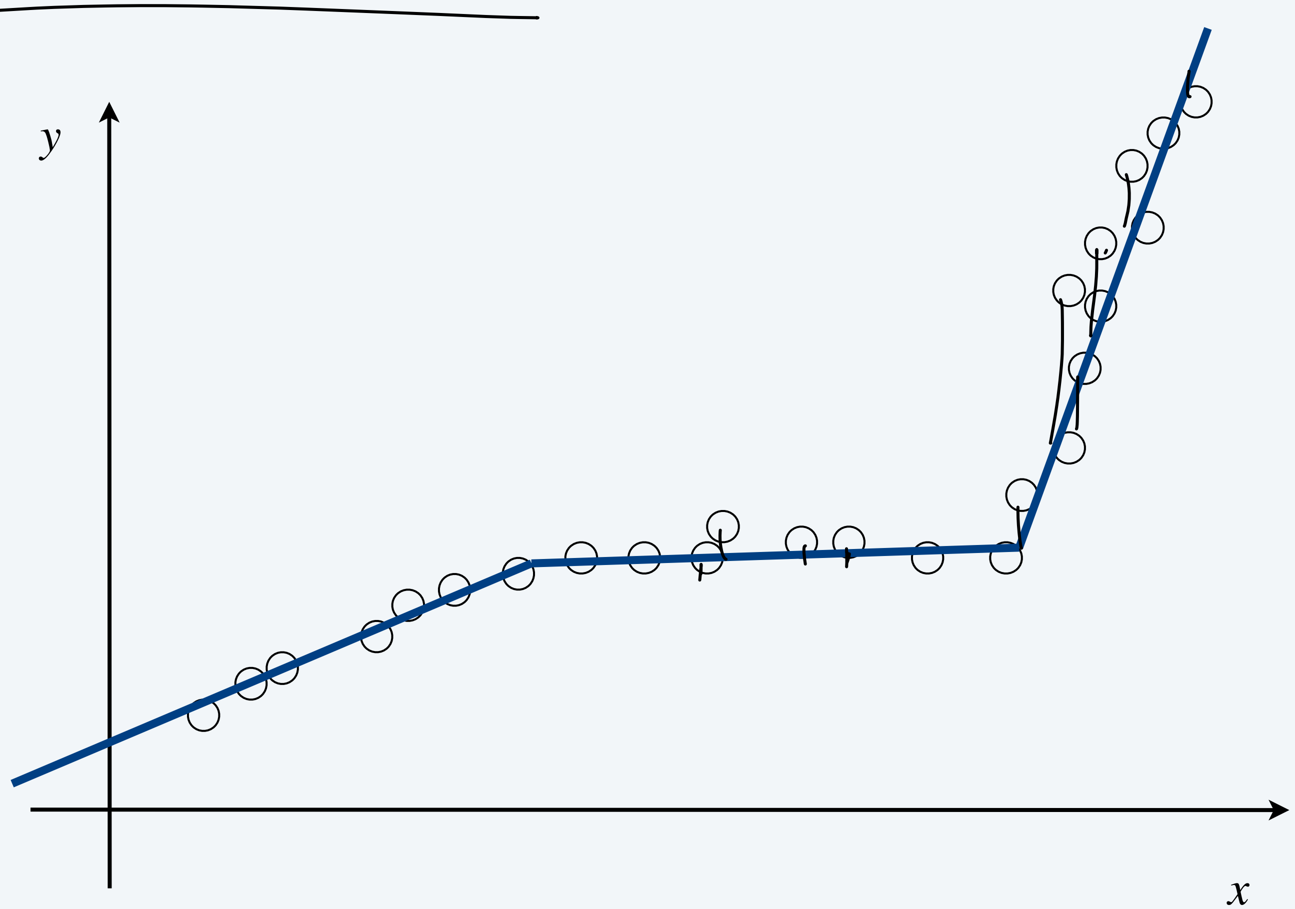
Segmented least squares

Segmented least squares.

- Points lie roughly on a sequence of several line segments.
- Given n points in the plane: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ with $x_1 < x_2 < \ldots < x_n$, find a sequence of lines that minimizes $f(x)$.

Goal. Minimize $f(x) = E + c\,L$ for some constant $c > 0$, where

- E = sum of the sums of the squared errors in each segment.
- L = number of lines.



$L = 3$

c

E

Dynamic programming: multiway choice

$(x_1, y_1), (x_2, y_2), (x_3, y_3) \dots (x_n, y_n)$

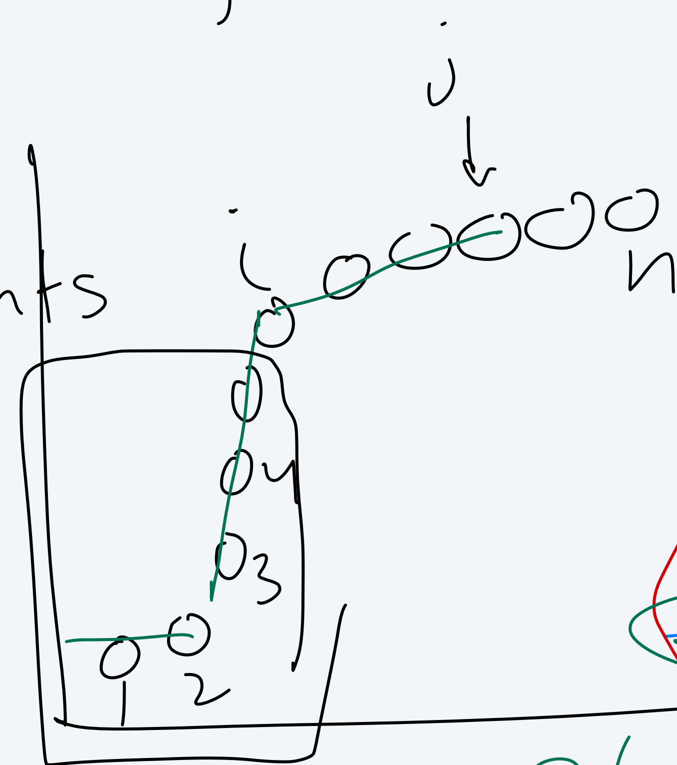
$OPT(j)$ = minimum value of $f(x)$ using all points up to point j

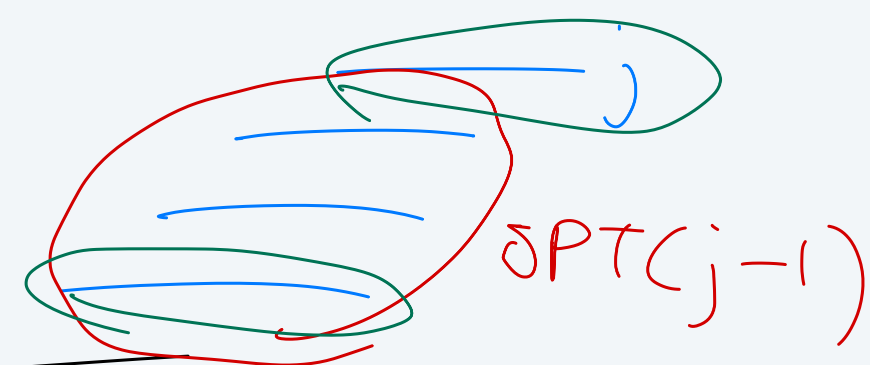
$f(x) = E + cL$

$OPT(j) = c + e_{ij} + OPT(i-1)$

- c: cost of line
- e_{ij}: cost of errors on line
- points i through j are on a line

cost of line through points i to j





$opt(p(j)) + w_j$

Dynamic programming: multiway choice

Notation.

Dynamic programming: multiway choice

Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.

Dynamic programming: multiway choice

Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.
- e_{ij} = SSE for for points $p_i, p_{i+1}, \ldots, p_j$.

Dynamic programming: multiway choice

Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.
- e_{ij} = SSE for for points $p_i, p_{i+1}, \ldots, p_j$.

To compute $OPT(j)$:

Dynamic programming: multiway choice

Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.
- e_{ij} = SSE for for points $p_i, p_{i+1}, \ldots, p_j$.

To compute $OPT(j)$:

- Last segment uses points $p_i, p_{i+1}, \ldots, p_j$ for some $i \leq j$.

Dynamic programming: multiway choice

Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.
- e_{ij} = SSE for for points $p_i, p_{i+1}, \ldots, p_j$.

To compute $OPT(j)$:

- Last segment uses points $p_i, p_{i+1}, \ldots, p_j$ for some $i \leq j$.
- Cost = $e_{ij} + c + OPT(i-1)$.

Dynamic programming: multiway choice

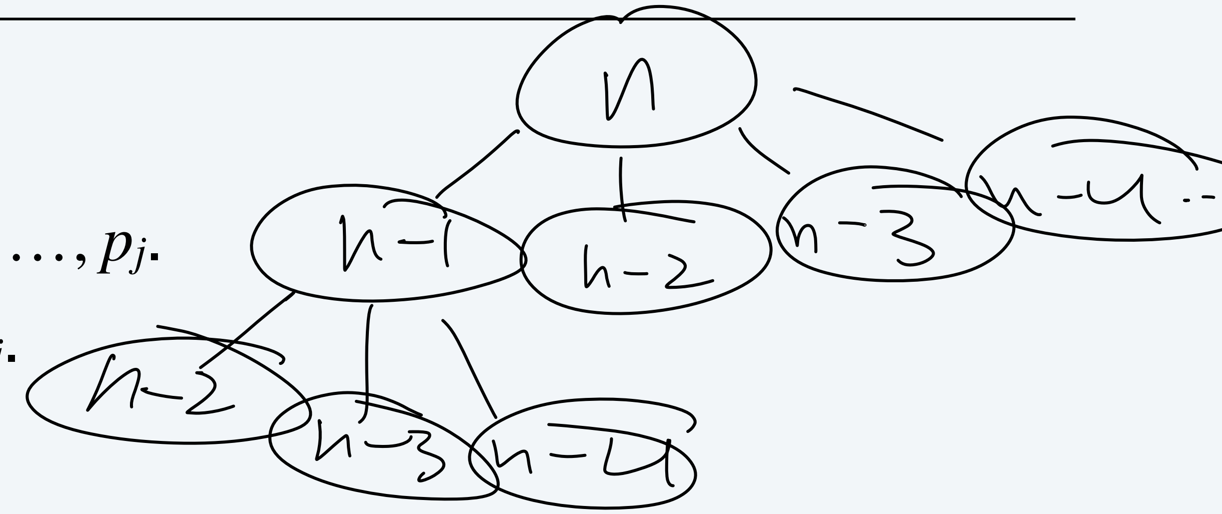
Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.
- e_{ij} = SSE for for points $p_i, p_{i+1}, \ldots, p_j$.

To compute $OPT(j)$:

- Last segment uses points $p_i, p_{i+1}, \ldots, p_j$ for some $i \leq j$.
- Cost = $e_{ij} + c + OPT(i-1)$.
- But which i?

Dynamic programming: multiway choice

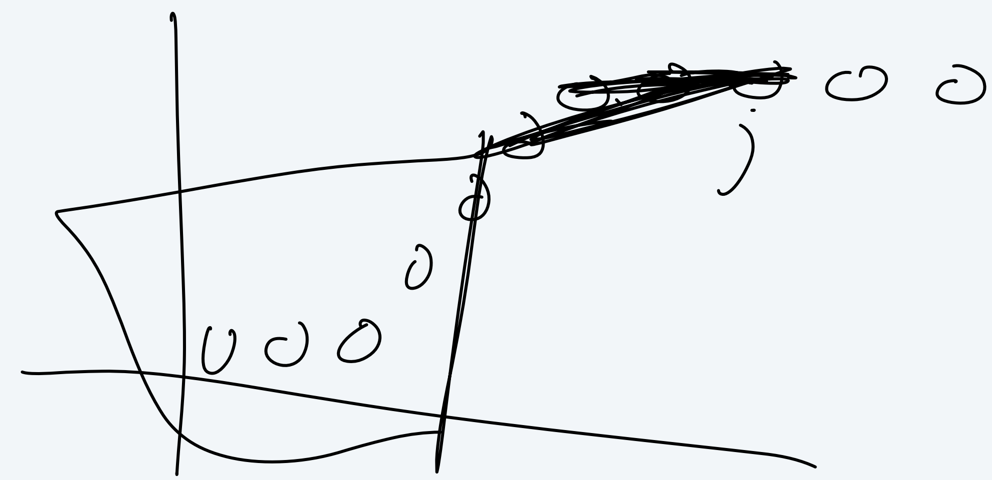


Notation.

- $OPT(j)$ = minimum cost for points $p_1, p_2, \ldots, p_j$.
- e_{ij} = SSE for for points $p_i, p_{i+1}, \ldots, p_j$.

To compute $OPT(j)$:

- Last segment uses points $p_i, p_{i+1}, \ldots, p_j$ for some $i \le j$.
- Cost = $e_{ij} + c + OPT(i-1)$.
- But which i?



Answer: consider all of them!

$$OPT(j) = \begin{cases} 0 & \text{if } j = 0 \\ \min\limits_{1 \le i \le j} \{ e_{ij} + c + OPT(i-1) \} & \text{if } j > 0 \end{cases}$$

Segmented least squares algorithm

Segmented least squares algorithm

SEGMENTED-LEAST-SQUARES($n, p_1, \ldots, p_n, c$)

Segmented least squares algorithm

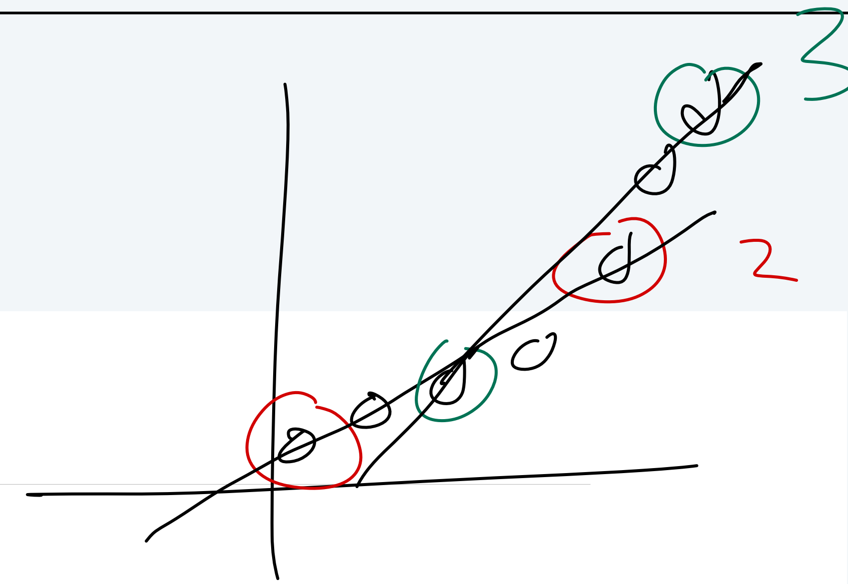
SEGMENTED-LEAST-SQUARES(n, p_1, …, p_n, c)

FOR j = 1 TO n

  FOR i = 1 TO j

    Compute the SSE e_{ij} for the points $p_i, p_{i+1}, \ldots, p_j$.

Segmented least squares algorithm



SEGMENTED-LEAST-SQUARES$(n, p_1, \ldots, p_n, c)$

pre-compute e_{ij} values

FOR j = 1 TO n

  FOR i = 1 TO j

    Compute the SSE e_{ij} for the points $p_i, p_{i+1}, \ldots, p_j$.

$\Theta(n^3)$

runtime?

what's a lower bound?

Segmented least squares algorithm

SEGMENTED-LEAST-SQUARES$(n, p_1, \ldots, p_n, c)$

FOR $j = 1$ TO n

FOR $i = 1$ TO j

Compute the SSE e_{ij} for the points $p_i, p_{i+1}, \ldots, p_j$.

runtime?

$M[0] \leftarrow 0$.

FOR $j = 1$ TO n

$M[j] \leftarrow \min_{1 \le i \le j} \{ e_{ij} + c + M[i-1] \}$.

previously computed value

Segmented least squares algorithm

SEGMENTED-LEAST-SQUARES(n, p_1, …, p_n, c)

FOR j = 1 TO n

  FOR i = 1 TO j

    Compute the SSE e_{ij} for the points $p_i, p_{i+1}, \ldots, p_j$.

runtime?

$M[0] \leftarrow 0$.

FOR $j = 1$ TO n

  $M[j] \leftarrow \min_{1 \le i \le j} \{ e_{ij} + c + M[i-1] \}$.

previously computed value

RETURN $M[n]$.

Segmented least squares algorithm

SEGMENTED-LEAST-SQUARES$(n, p_1, \ldots, p_n, c)$

FOR $j = 1$ TO n

  FOR $i = 1$ TO j

    Compute the SSE e_{ij} for the points $p_i, p_{i+1}, \ldots, p_j$.

runtime?

$M[0] \leftarrow 0$.

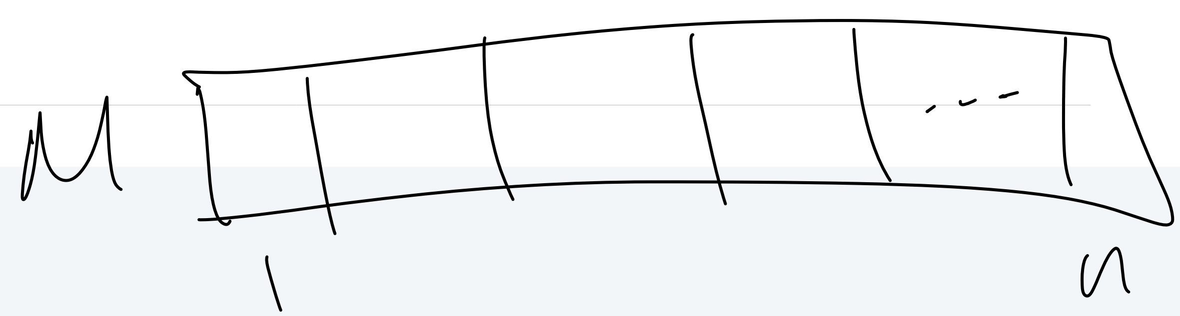
FOR $j = 1$ TO n

  $M[j] \leftarrow \min_{1 \le i \le j} \{ e_{ij} + c + M[i-1] \}$.

previously computed value

runtime?

RETURN $M[n]$.



Segmented least squares analysis

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Pf.

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Pf.

- Bottleneck = computing SSE e_{ij} for each i and j.

$$a_{ij} \;=\; \frac{n \sum_k x_k y_k - (\sum_k x_k)(\sum_k y_k)}{n \sum_k x_k^2 - (\sum_k x_k)^2}, \quad b_{ij} \;=\; \frac{\sum_k y_k - a_{ij} \sum_k x_k}{n}$$

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Pf.

- Bottleneck = computing SSE e_{ij} for each i and j.

$$a_{ij} = \frac{n \sum_k x_k y_k - (\sum_k x_k)(\sum_k y_k)}{n \sum_k x_k^2 - (\sum_k x_k)^2}, \quad b_{ij} = \frac{\sum_k y_k - a_{ij} \sum_k x_k}{n}$$

- $O(n)$ to compute e_{ij}. ▪

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Pf.

- Bottleneck = computing SSE e_{ij} for each i and j.

$$a_{ij} = \frac{n \sum_k x_k y_k - (\sum_k x_k)(\sum_k y_k)}{n \sum_k x_k^2 - (\sum_k x_k)^2}, \quad b_{ij} = \frac{\sum_k y_k - a_{ij} \sum_k x_k}{n}$$

- $O(n)$ to compute e_{ij}. ▪

Remark. Can be improved to $O(n^2)$ time.

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Pf.

- Bottleneck = computing SSE e_{ij} for each i and j.

$$a_{ij} = \frac{n \sum_k x_k y_k - (\sum_k x_k)(\sum_k y_k)}{n \sum_k x_k^2 - (\sum_k x_k)^2}, \quad b_{ij} = \frac{\sum_k y_k - a_{ij} \sum_k x_k}{n}$$

- $O(n)$ to compute e_{ij}. ▪

Remark. Can be improved to $O(n^2)$ time.

- For each i : precompute cumulative sums $\sum_{k=1}^{i} x_k, \quad \sum_{k=1}^{i} y_k, \quad \sum_{k=1}^{i} x_k^2, \quad \sum_{k=1}^{i} x_k y_k$

Segmented least squares analysis

Theorem. [Bellman 1961] DP algorithm solves the segmented least squares problem in $O(n^3)$ time and $O(n^2)$ space.

Pf.

- Bottleneck = computing SSE e_{ij} for each i and j.

$$a_{ij} = \frac{n \sum_k x_k y_k - (\sum_k x_k)(\sum_k y_k)}{n \sum_k x_k^2 - (\sum_k x_k)^2}, \quad b_{ij} = \frac{\sum_k y_k - a_{ij} \sum_k x_k}{n}$$

- $O(n)$ to compute e_{ij}. ▪

Remark. Can be improved to $O(n^2)$ time.

- For each i : precompute cumulative sums $\sum_{k=1}^{i} x_k, \ \sum_{k=1}^{i} y_k, \ \sum_{k=1}^{i} x_k^2, \ \sum_{k=1}^{i} x_k y_k$
- Using cumulative sums, can compute e_{ij} in $O(1)$ time.