DATA MINING LECTURE 8

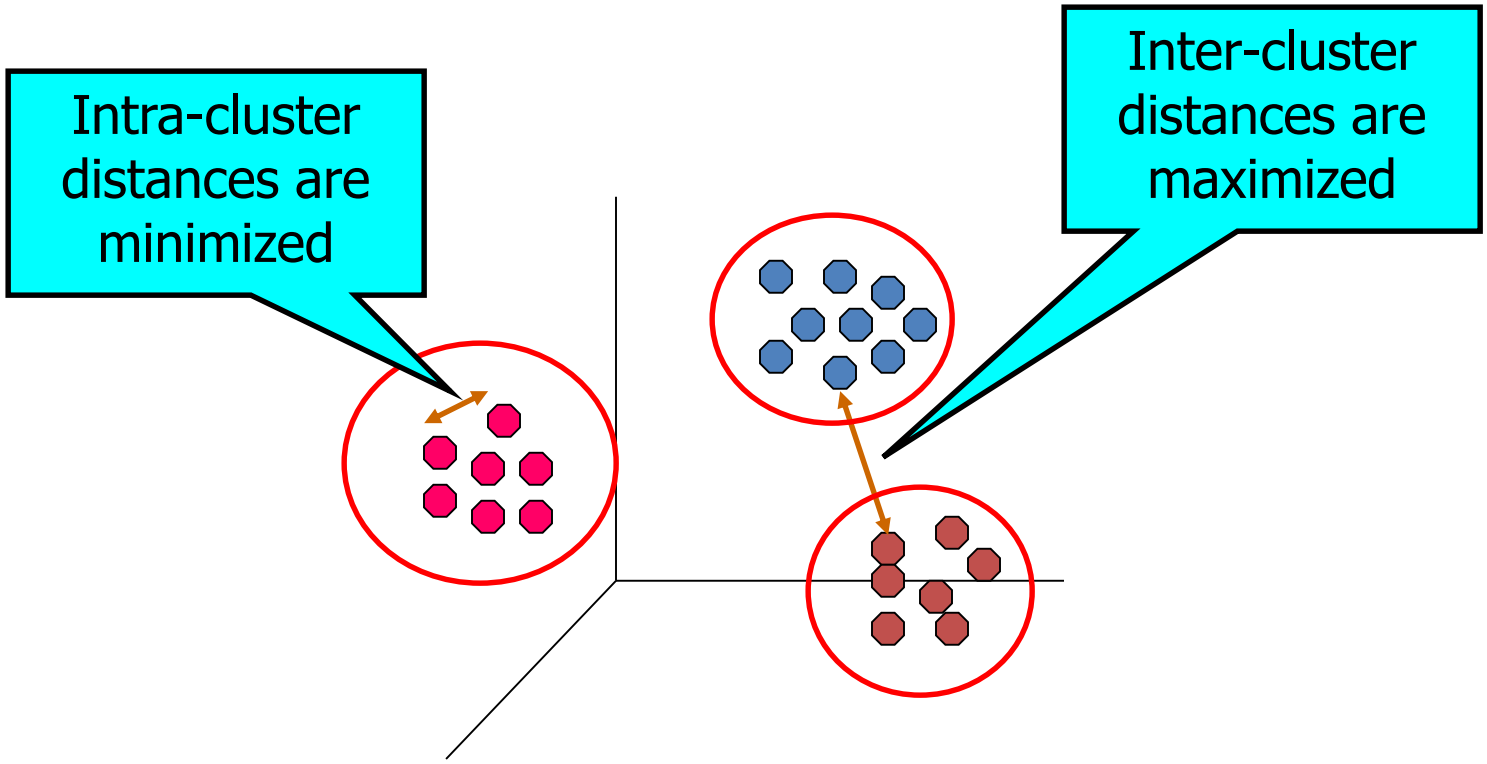
The EM Algorithm

Sequence segmentation

CLUSTERING

What is a Clustering?

- In general a grouping of objects such that the objects in a group (cluster) are similar (or related) to one another and different from (or unrelated to) the objects in other groups



Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- DBSCAN

MIXTURE MODELS AND THE EM ALGORITHM

Model-based clustering

- In order to understand our data, we will assume that there is a generative process (a model) that creates/describes the data, and we will try to find the model that best fits the data.
  - Models of different complexity can be defined, but we will assume that our model is a distribution from which data points are sampled
  - Example: the data is the height of all people in Greece
- In most cases, a single distribution is not good enough to describe all data points: different parts of the data follow a different distribution
  - Example: the data is the height of all people in Greece and China
  - We need a mixture model
  - Different distributions correspond to different clusters in the data.

Gaussian Distribution

- Example: the data is the height of all people in Greece
  - Experience has shown that this data follows a Gaussian (Normal) distribution
  - Reminder: Normal distribution:

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

  - μ = mean, σ = standard deviation

Gaussian Model

- What is a model?
  - A Gaussian distribution is fully defined by the mean μ and the standard deviation σ
  - We define our model as the pair of parameters $\theta = (\mu, \sigma)$
- This is a general principle: a model is defined as a vector of parameters θ

Fitting the model

- We want to find the normal distribution that best fits our data
  - Find the best values for μ and σ
  - But what does best fit mean?

Maximum Likelihood Estimation (MLE)

- Find the most likely parameters given the data. Given the data observations X, find θ that maximizes $P(\theta|X)$
  - Problem: We do not know how to compute $P(\theta|X)$

- Using Bayes Rule:

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

- If we have no prior information about θ, or X, we can assume uniform. Maximizing $P(\theta|X)$ is the same as maximizing $P(X|\theta)$

Maximum Likelihood Estimation (MLE)

- We have a vector $X = (x_1, \dots, x_n)$ of values and we want to fit a Gaussian $N(\mu, \sigma)$ model to the data
  - Our parameter set is $\theta = (\mu, \sigma)$
- Probability of observing point x_i given the parameters θ

$$P(x_i|\theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- Probability of observing all points (assume independence)

$$P(X|\theta) = \prod_{i=1}^{n} P(x_i|\theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- We want to find the parameters $\theta = (\mu, \sigma)$ that maximize the probability $P(X|\theta)$

Maximum Likelihood Estimation (MLE)

- The probability $P(X|\theta)$ as a function of θ is called the Likelihood function

$$L(\theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- It is usually easier to work with the Log-Likelihood function

$$LL(\theta) = -\sum_{i=1}^{n} \frac{(x_i-\mu)^2}{2\sigma^2} - \frac{1}{2} n \log 2\pi - n \log \sigma$$

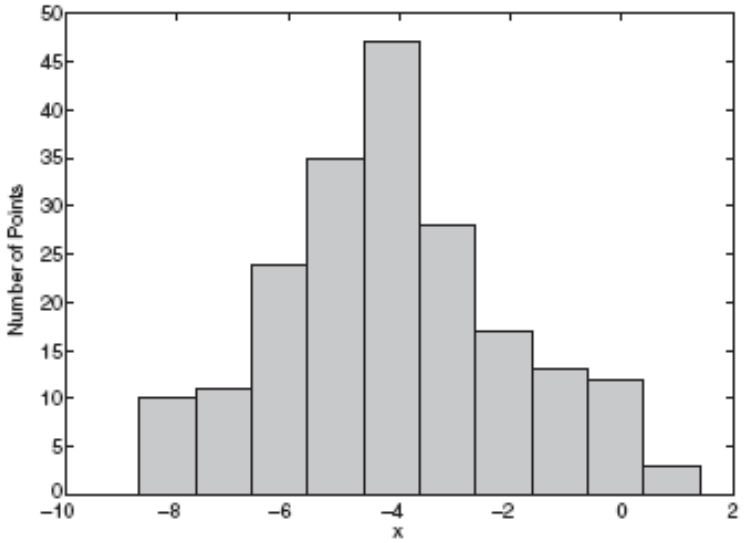
- Maximum Likelihood Estimation
  - Find parameters μ, σ that maximize $LL(\theta)$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i = \mu_X$$

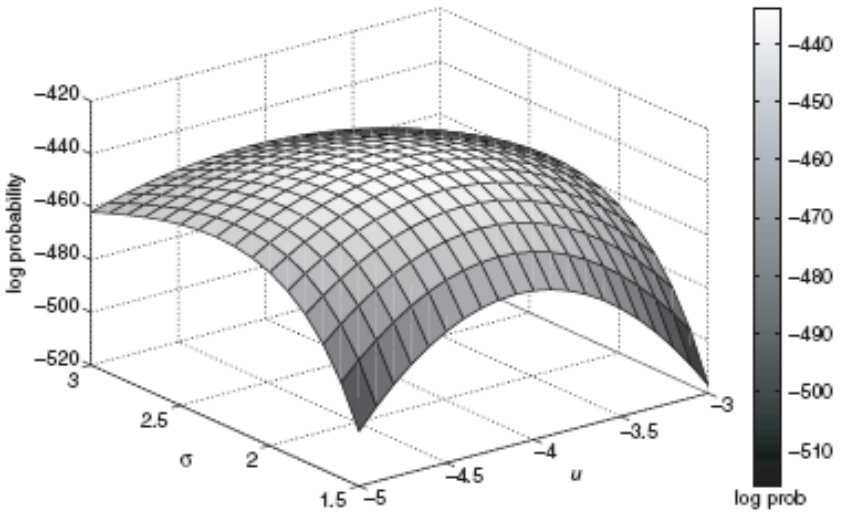
Sample Mean

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i-\mu)^2 = \sigma_X^2$$

Sample Variance



(a) Histogram of 200 points from a Gaussian distribution.

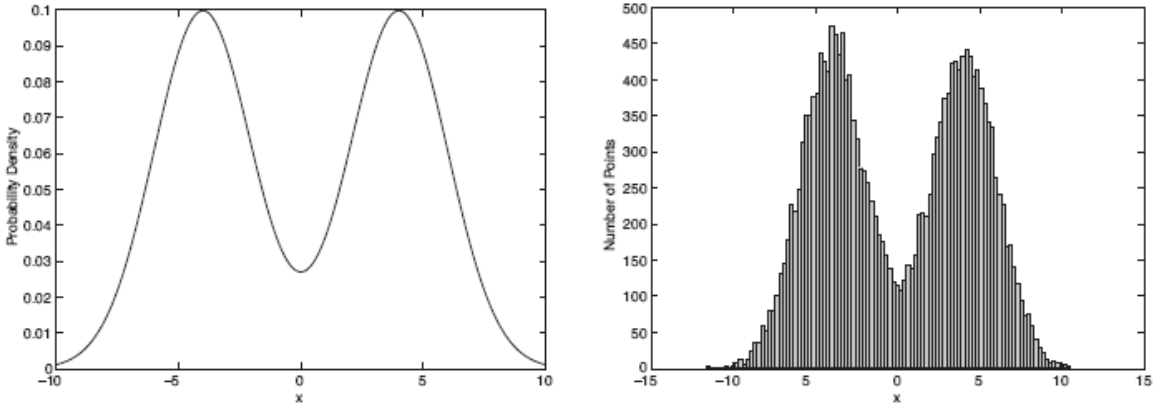


(b) Log likelihood plot of the 200 points for different values of the mean and standard deviation.

Figure 9.3. 200 points from a Gaussian distribution and their log probability for different parameter values.

Mixture of Gaussians

- Suppose that you have the heights of people from Greece and China and the distribution looks like the figure below (dramatization)



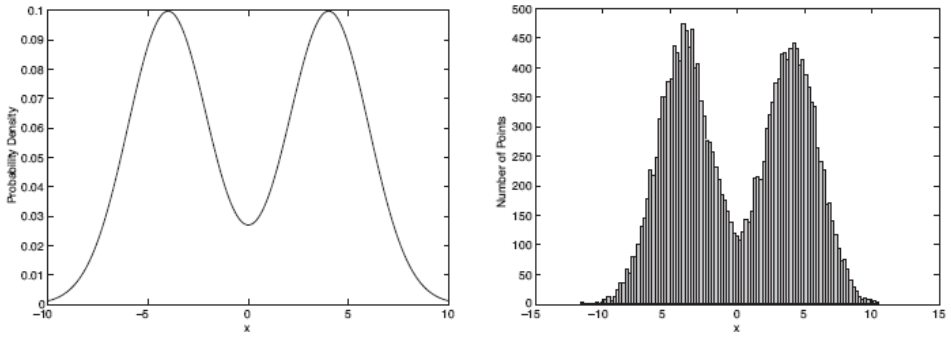
(a) Probability density function for the mixture model.

(b) 20,000 points generated from the mixture model.

Figure 9.2. Mixture model consisting of two normal distributions with means of -4 and 4, respectively. Both distributions have a standard deviation of 2.

Mixture of Gaussians

- In this case the data is the result of the mixture of two Gaussians
  - One for Greek people, and one for Chinese people
  - Identifying for each value which Gaussian is most likely to have generated it will give us a clustering.



(a) Probability density function for the mixture model.

(b) 20,000 points generated from the mixture model.

Figure 9.2. Mixture model consisting of two normal distributions with means of -4 and 4, respectively. Both distributions have a standard deviation of 2.

Mixture model

- A value x_i is generated according to the following process:
  - First select the nationality
    - With probability π_G select Greece, with probability π_C select China $(\pi_G + \pi_C = 1)$

We can also thing of this as a Hidden Variable Z that takes two values: Greece and China

  - Given the nationality, generate the point from the corresponding Gaussian
    - $P(x_i|\theta_G) \sim N(\mu_G, \sigma_G)$ if Greece
    - $P(x_i|\theta_C) \sim N(\mu_C, \sigma_C)$ if China

θ_G: parameters of the Greek distribution
θ_C: parameters of the China distribution

Mixture Model

- Our model has the following parameters

$$\Theta = (\pi_G, \pi_C, \mu_G, \sigma_G, \mu_C, \sigma_C)$$

Mixture probabilities

θ_G: parameters of the Greek distribution

θ_C: parameters of the China distribution

Mixture Model

- Our model has the following parameters

$$\Theta = (\pi_G, \pi_C, \mu_G, \sigma_G, \mu_C, \sigma_C)$$

Mixture probabilities | Distribution Parameters

- For value x_i, we have:

$$P(x_i|\Theta) = \pi_G P(x_i|\theta_G) + \pi_C P(x_i|\theta_C)$$

- For all values $X = (x_1, \ldots, x_n)$

$$P(X|\Theta) = \prod_{i=1}^{n} P(x_i|\Theta)$$

- We want to estimate the parameters that maximize the Likelihood of the data

Mixture Models

- Once we have the parameters $\Theta = (\pi_G, \pi_C, \mu_G, \mu_C, \sigma_G, \sigma_C)$ we can estimate the membership probabilities $P(G|x_i)$ and $P(C|x_i)$ for each point x_i:
  - This is the probability that point x_i belongs to the Greek or the Chinese population (cluster)

Given from the Gaussian distribution $N(\mu_G, \sigma_G)$ for Greek

$$P(G|x_i) = \frac{P(x_i|G)P(G)}{P(x_i|G)P(G) + P(x_i|C)P(C)}$$

$$= \frac{P(x_i|\theta_G)\pi_G}{P(x_i|\theta_G)\pi_G + P(x_i|\theta_C)\pi_C}$$

EM (Expectation Maximization) Algorithm

- Initialize the values of the parameters in Θ to some random values
- Repeat until convergence
  - E-Step: Given the parameters Θ estimate the membership probabilities $P(G|x_i)$ and $P(C|x_i)$
  - M-Step: Compute the parameter values that (in expectation) maximize the data likelihood

$$\pi_C = \frac{1}{n}\sum_{i=1}^{n} P(C|x_i) \qquad \pi_G = \frac{1}{n}\sum_{i=1}^{n} P(G|x_i)$$

Fraction of population in G,C

$$\mu_C = \frac{1}{n * \pi_C}\sum_{i=1}^{n} P(C|x_i)x_i \qquad \mu_G = \frac{1}{n * \pi_G}\sum_{i=1}^{n} P(G|x_i)x_i$$

MLE Estimates if π's were fixed

$$\sigma_C^2 = \frac{1}{n * \pi_C}\sum_{i=1}^{n} P(C|x_i)(x_i - \mu_C)^2 \qquad \sigma_G^2 = \frac{1}{n * \pi_G}\sum_{i=1}^{n} P(G|x_i)(x_i - \mu_G)^2$$

Relationship to K-means

- E-Step: Assignment of points to clusters
  - K-means: hard assignment, EM: soft assignment
- M-Step: Computation of centroids
  - K-means assumes common fixed variance (spherical clusters)
  - EM: can change the variance for different clusters or different dimensions (ellipsoid clusters)
- If the variance is fixed then both minimize the same error function

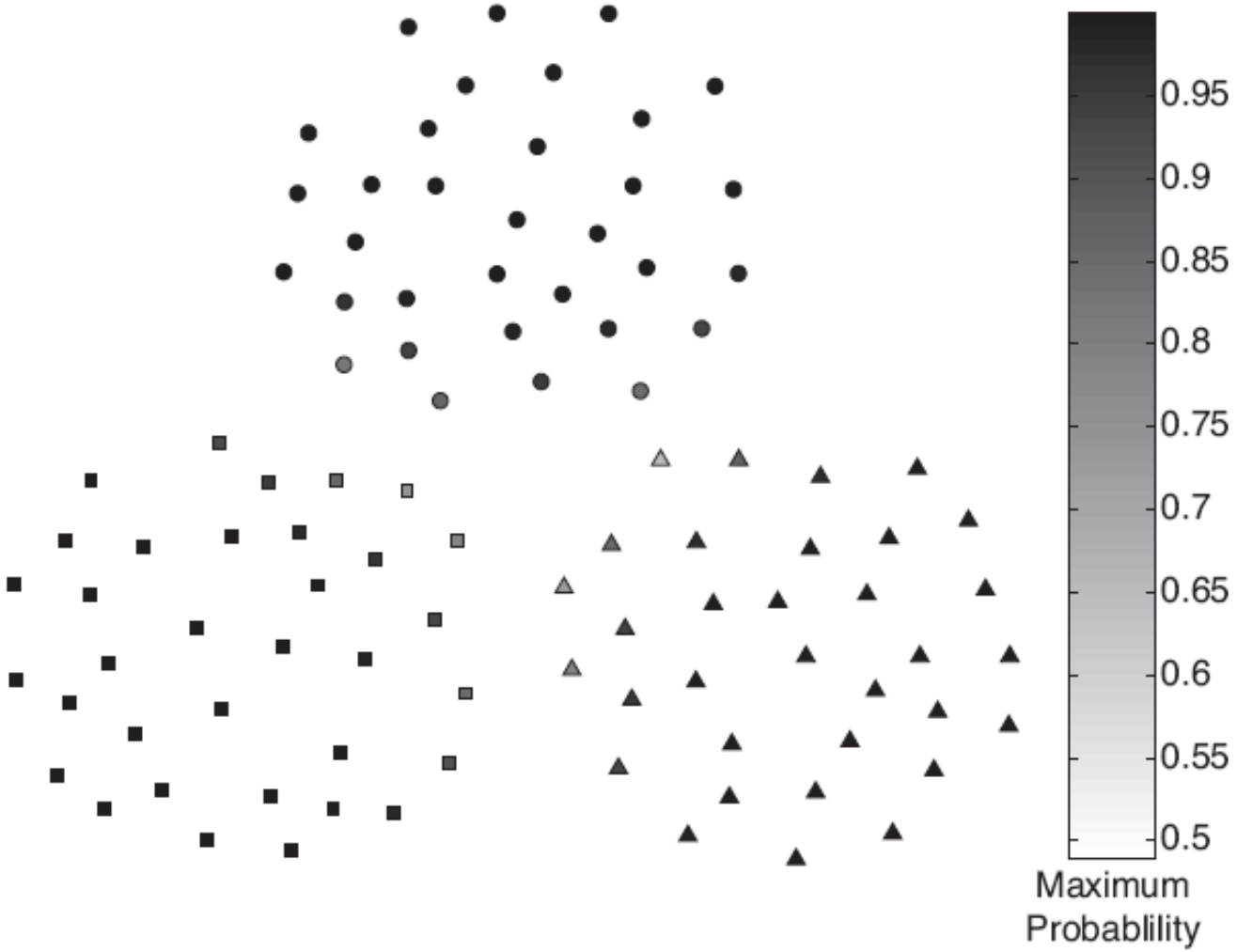


Figure 9.4. EM clustering of a two-dimensional point set with three clusters.

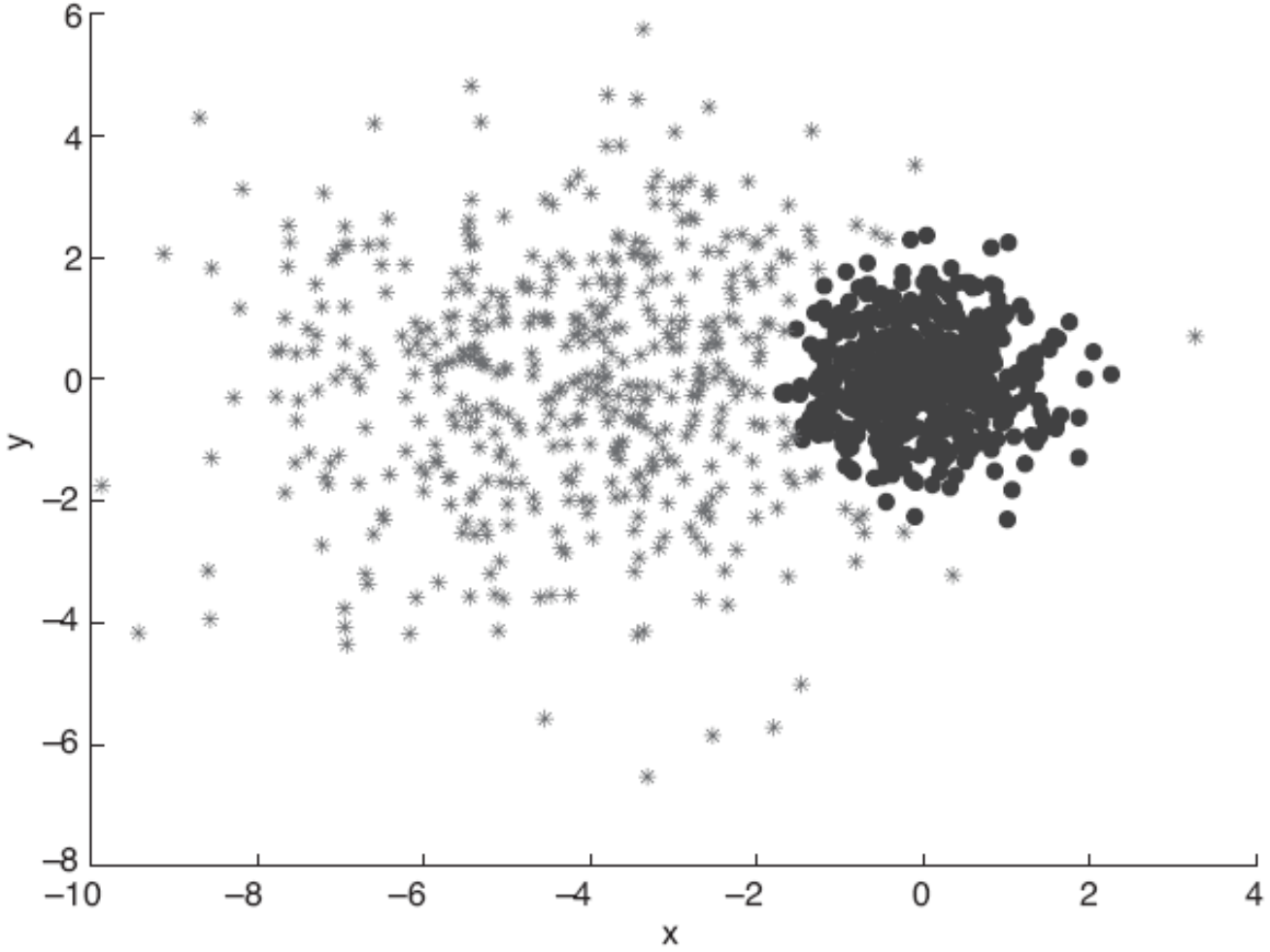
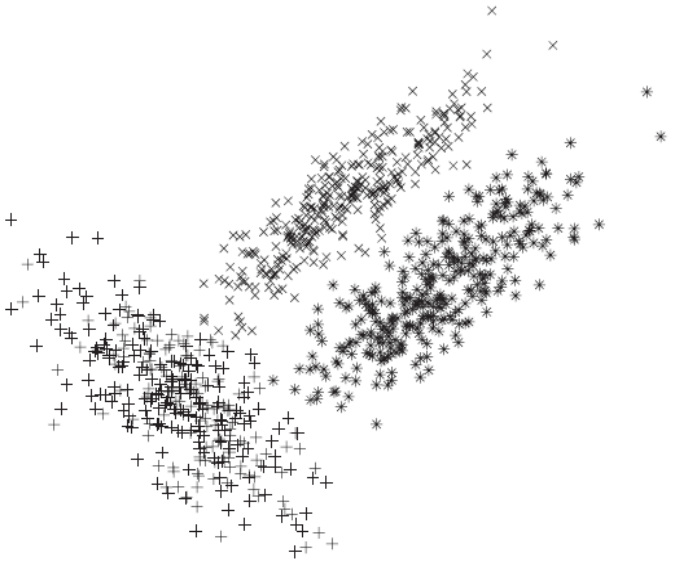
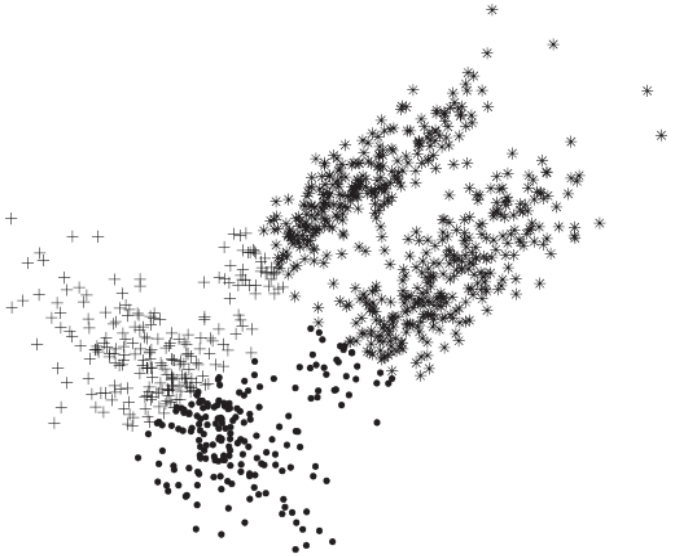


Figure 9.5. EM clustering of a two-dimensional point set with two clusters of differing density.



(a) Clusters produced by mixture model clustering.



(b) Clusters produced by K-means clustering.

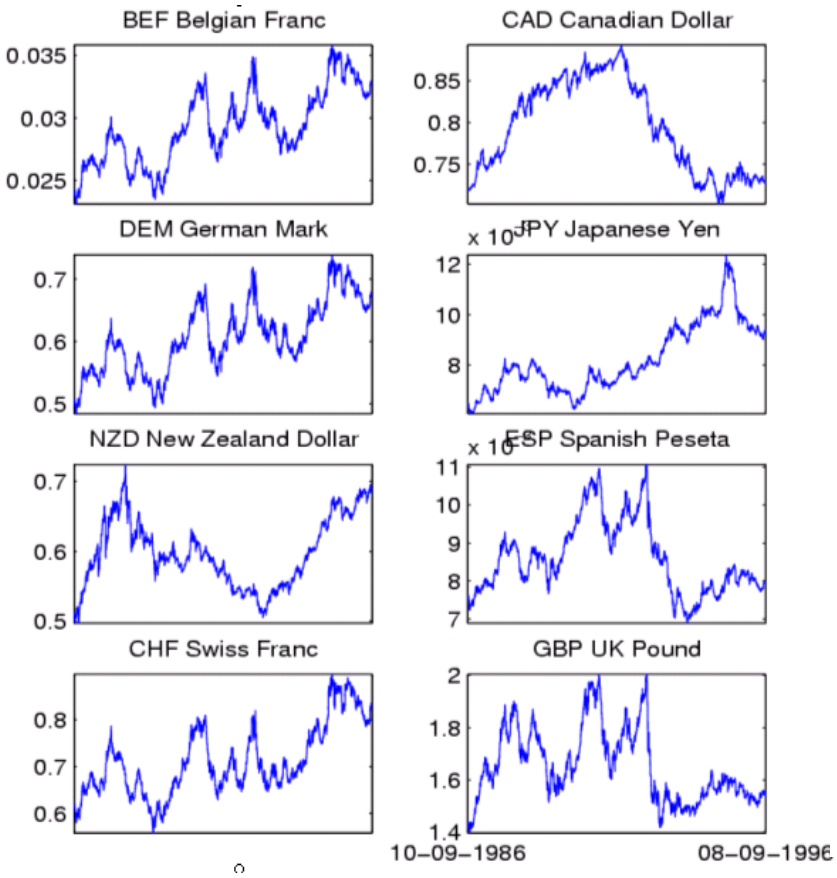
Figure 9.6. Mixture model and K-means clustering of a set of two-dimensional points.

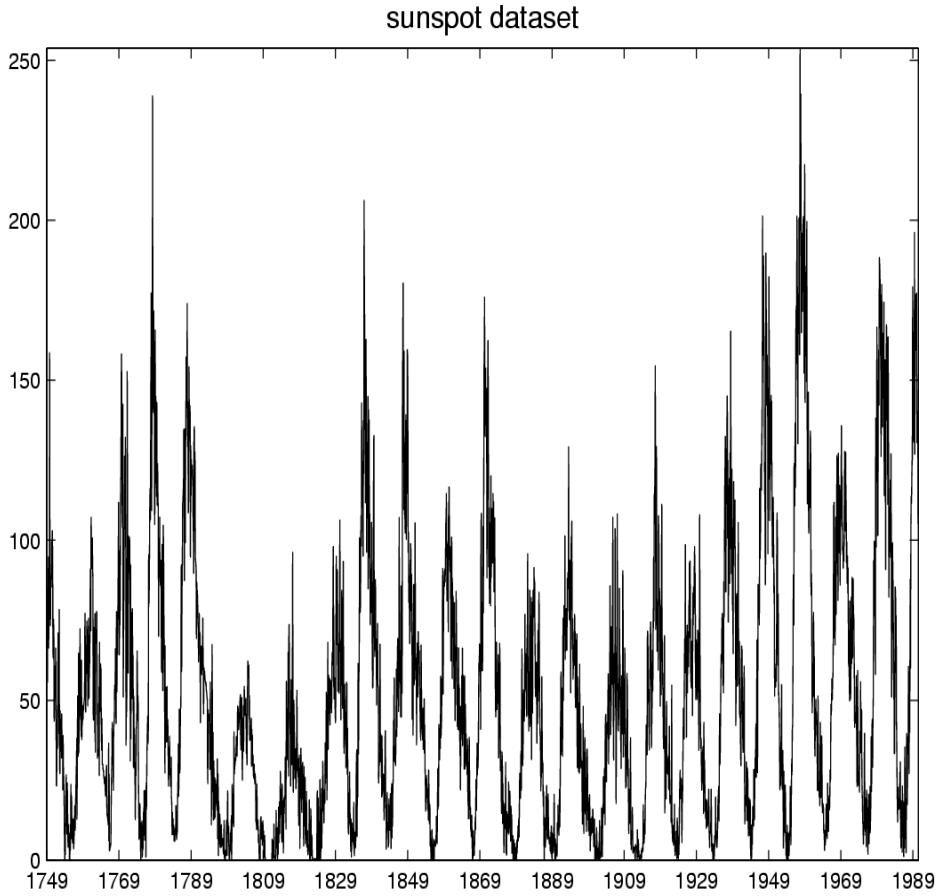
SEQUENCE SEGMENTATION

Sequential data

- Sequential data (or time series) refers to data that appear in a specific order.
  - The order defines a time axis, that differentiates this data from other cases we have seen so far
- Examples
  - The price of a stock (or of many stocks) over time
  - Environmental data (pressure, temperature, precipitation etc) over time
  - The sequence of queries in a search engine, or the frequency of a single query over time
  - The words in a document as they appear in order
  - A DNA sequence of nucleotides
  - Event occurrences in a log over time
  - Etc…
- Time series: usually we assume that we have a vector of numeric values that change over time.

Time-series data





- Financial time series, process monitoring…

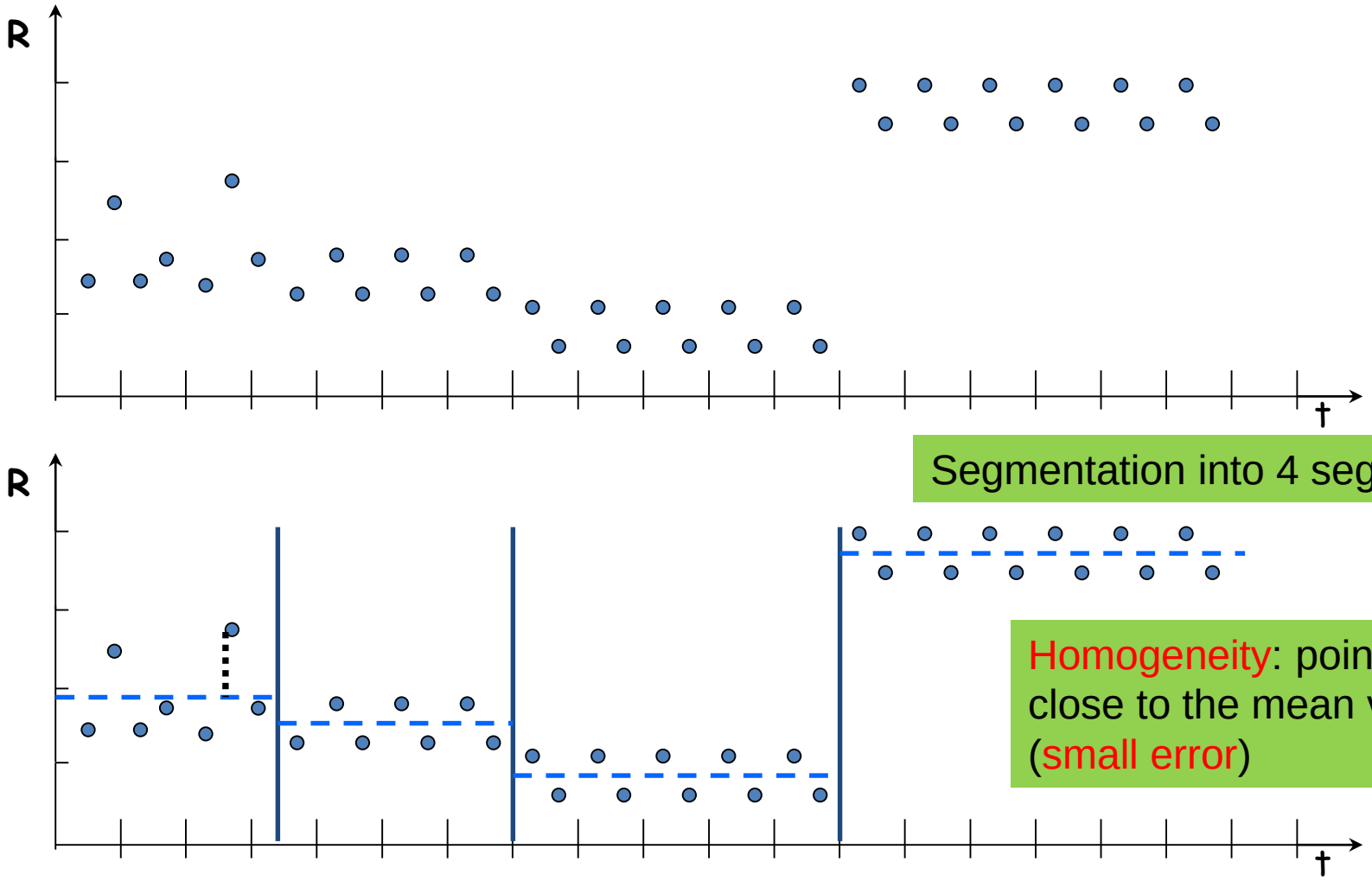
Time series analysis

- The addition of the time axis defines new sets of problems
  - Discovering periodic patterns in time series
  - Defining similarity between time series
  - Finding bursts, or outliers
- Also, some existing problems need to be revisited taking sequential order into account
  - Association rules and Frequent Itemsets in sequential data
  - Summarization and Clustering: Sequence Segmentation

Sequence Segmentation

- Goal: discover structure in the sequence and provide a concise summary
- Given a sequence T, segment it into K contiguous segments that are as homogeneous as possible
- Similar to clustering but now we require the points in the cluster to be contiguous
- Commonly used for summarization of histograms in databases

Example



Segmentation into 4 segments

Homogeneity: points are close to the mean value (small error)

Basic definitions

- Sequence $T = \{t_1, t_2, \ldots, t_N\}$: an ordered set of N d-dimensional real points $t_i \in R^d$
- A K-segmentation S: a partition of T into K contiguous segments $\{s_1, s_2, \ldots, s_K\}$.
  - Each segment $s \in S$ is represented by a single vector $\mu \in \mathbb{R}^d$(the representative of the segment -- same as the centroid of a cluster)
- Error E(S): The error of replacing individual points with representatives
  - Different error functions, define different representatives.
- Sum of Squares Error (SSE):

$$E(S) = \sum_{s \in S} \sum_{t \in s} (t - \mu_s)^2$$

- Representative of segment s with SSE: mean $\mu_s = \frac{1}{|s|} \sum_{t \in s} t$

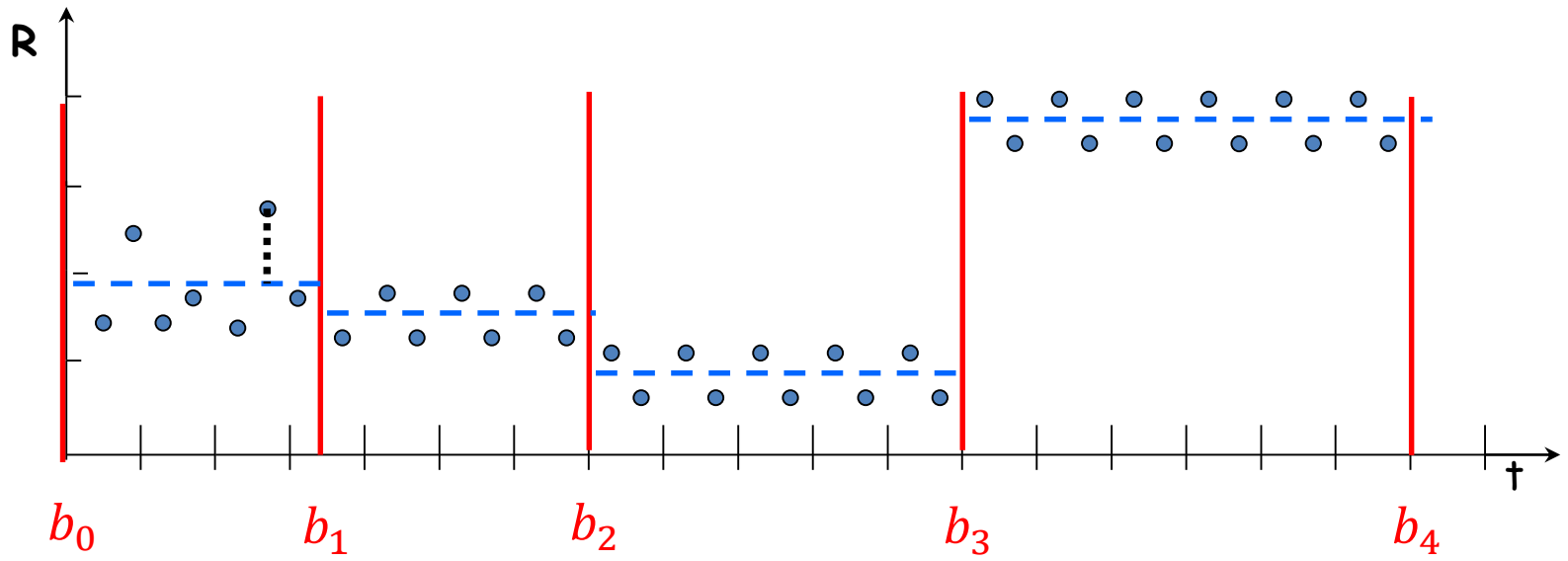
The K-segmentation problem

Given a sequence T of length N and a value K, find a K-segmentation $S = \{s_1, s_2, \ldots, s_K\}$ of T such that the SSE error E is minimized.

- Similar to K-means clustering, but now we need the points in the clusters to respect the order of the sequence.
  - This actually makes the problem easier.

Basic Definitions

- Observation: a K-segmentation S is defined by $K+1$ boundary points $b_0, b_1, \ldots, b_{K-1}, b_K$.



- $b_0 = 0, b_k = N + 1$ always.
  - We only need to specify $b_1, \ldots, b_{K-1}$

Optimal solution for the k-segmentation problem

- Bellman'61: The K-segmentation problem can be solved optimally using a standard dynamic programming algorithm

- Dynamic Programming:
  - Construct the solution of the problem by using solutions to problems of smaller size
    - Define the dynamic programming recursion
  - Build the solution bottom up from smaller to larger instances
    - Define the dynamic programming table that stores the solutions to the sub-problems

Rule of thumb

- Most optimization problems where order is involved can be solved optimally in polynomial time using dynamic programming.
  - The polynomial exponent may be large though

Dynamic Programming Recursion

- Terminology:
  - $T[1,n]$: subsequence $\{t_1, t_2, \ldots, t_n\}$ for $n \leq N$
  - $E(S[1,n],k)$: error of optimal segmentation of subsequence $T[1,n]$ with k segments for $k \leq K$

- Dynamic Programming Recursion:

$$E(S[1,n],k) = \min_{k \leq j \leq n-1} \left\{ E(S[1,j],k-1) + \sum_{j+1 \leq t \leq n} \left(t - \mu_{[j+1,n]}\right)^2 \right\}$$

Minimum over all possible placements of the last boundary point b_{k-1}

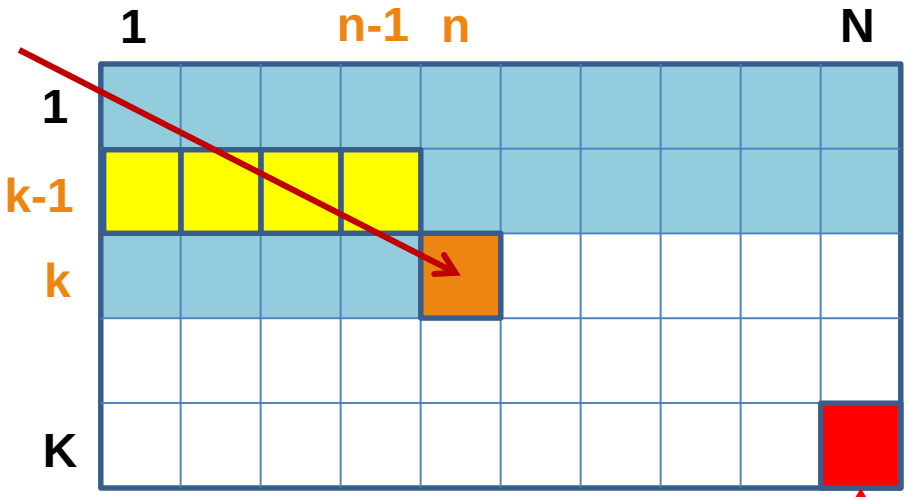
Error of optimal segmentation $S[1,j]$ with k-1 segments

Error of k-th (last) segment when the last segment is $[j+1,n]$

Dynamic programming table

- Two−dimensional table $A[1 \ldots K, 1 \ldots N]$

$$A[k,n] = E(S[1,n],k)$$

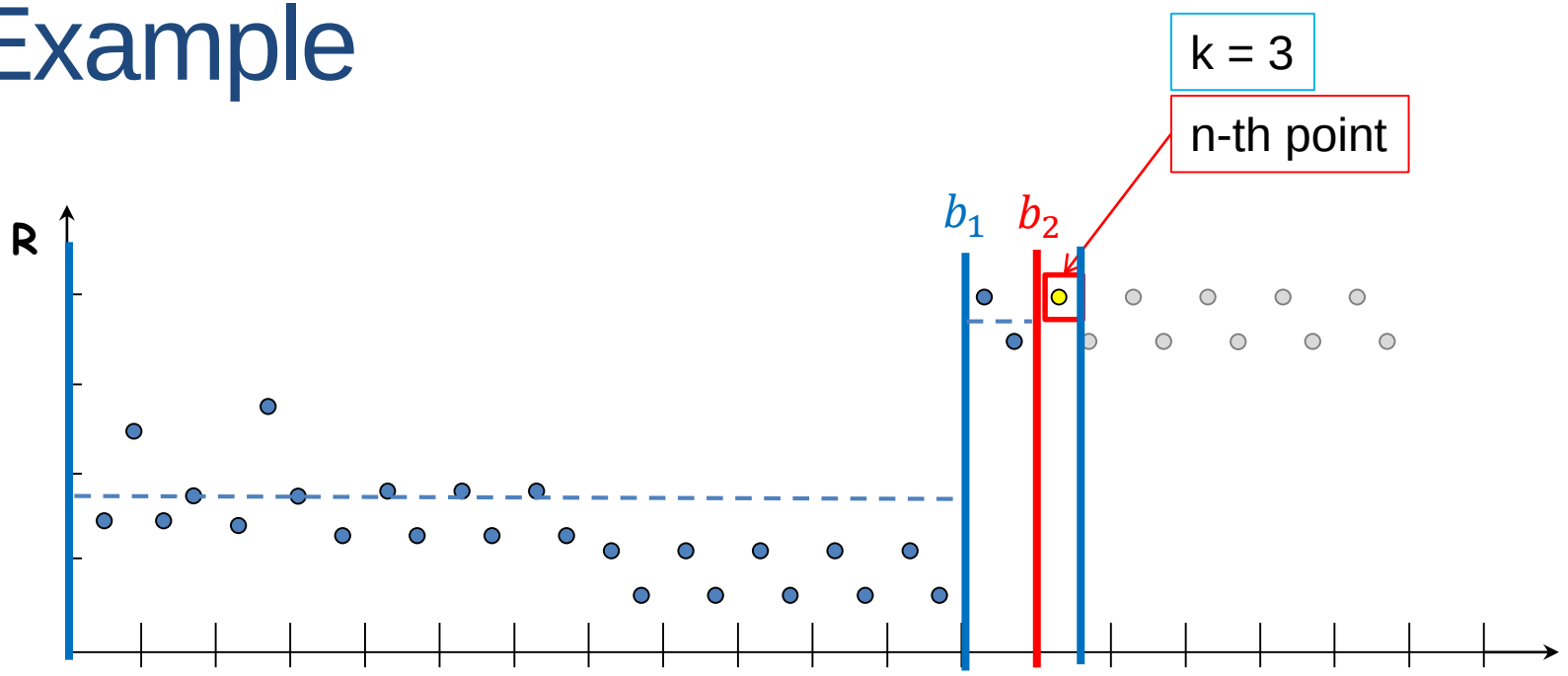


$$E(S[1,n],k) = \min_{k \le j \le n-1} \left\{ E(S[1,j],k-1) + \sum_{j+1 \le t \le n} \left(t - \mu_{[j+1,n]}\right)^2 \right\}$$

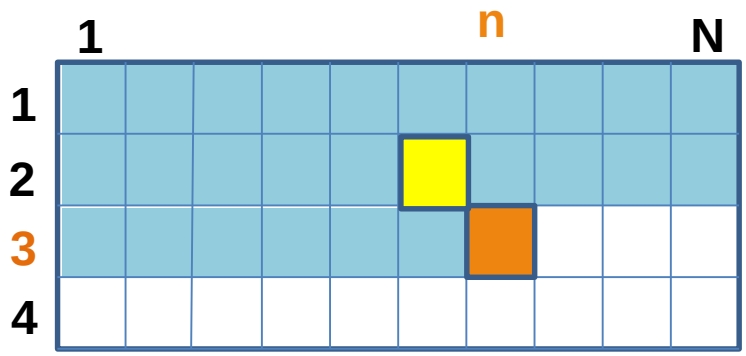
- Fill the table top to bottom, left to right.

Error of optimal K-segmentation

Example



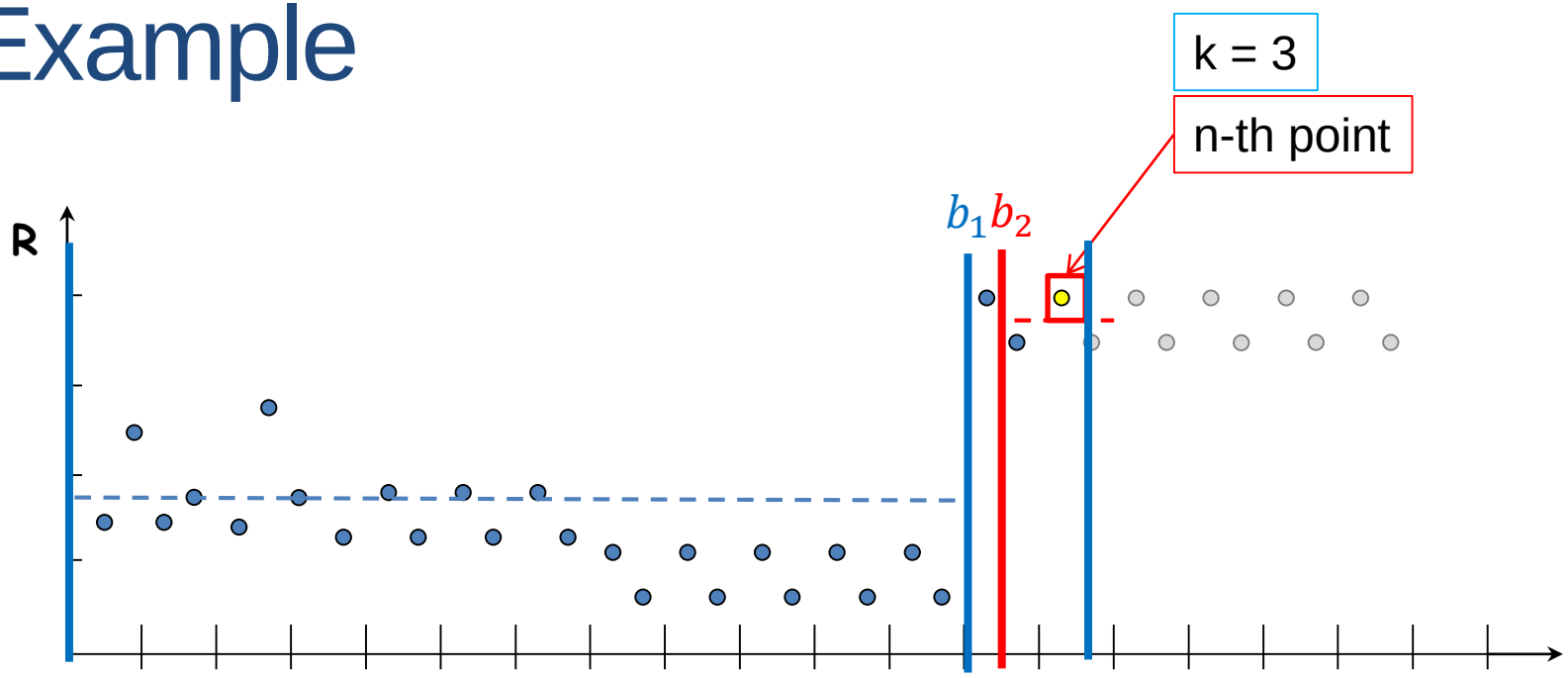
$$E(S[1,n],k) = \min_{k \le j \le n-1} \{E(S[1,j],k-1)$$



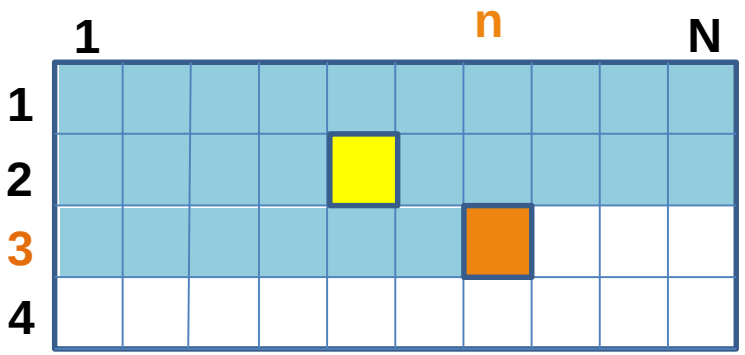
Three segnments means two boundaries b_1, b_2

Where should we place boundary b_2 ?

Example

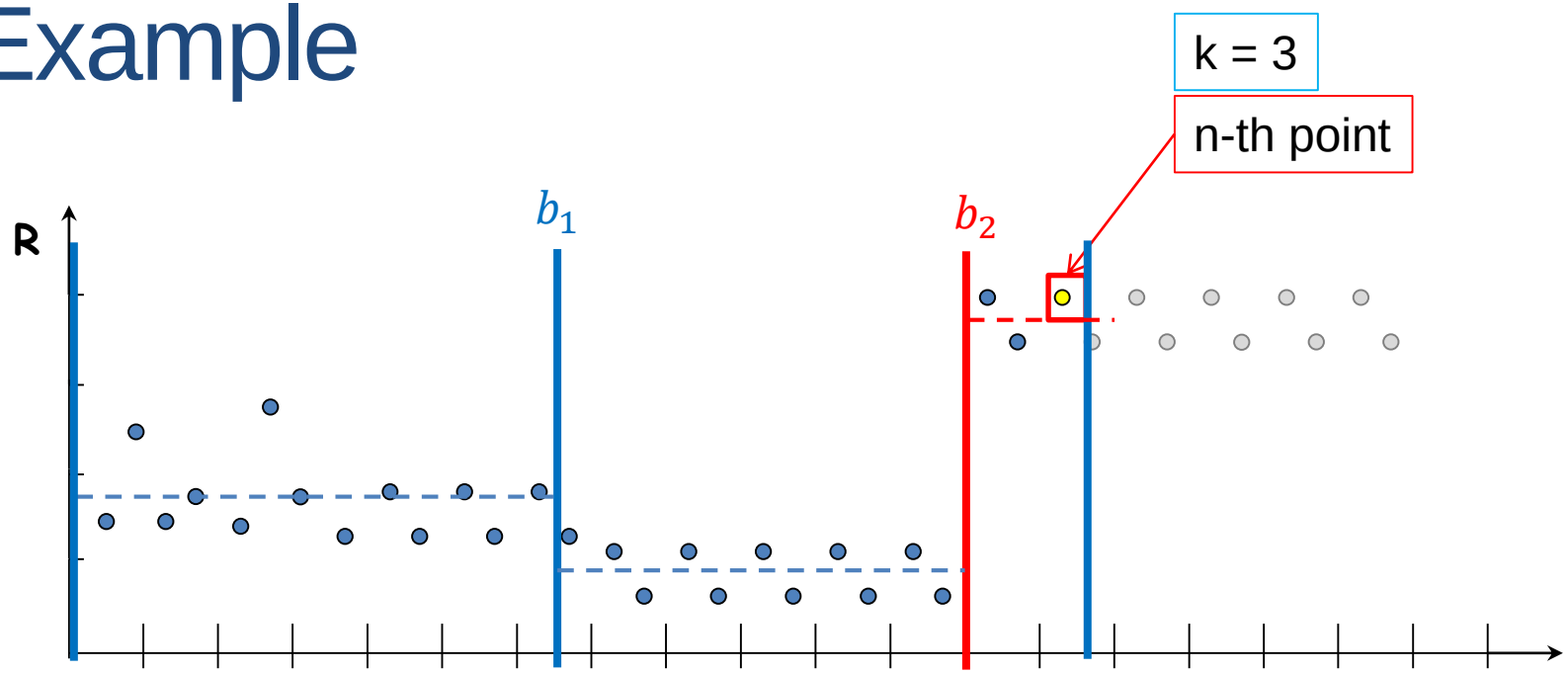


$$E(S[1,n],k) = \min_{k \le j \le n-1} \{ E(S[1,j],k-1) \}$$

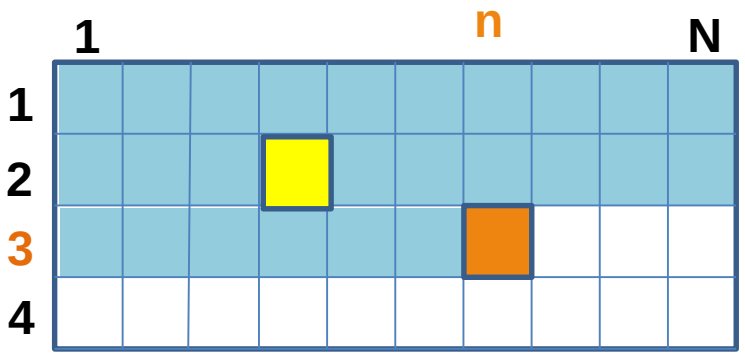


Where should we place boundary b_2 ?

Example

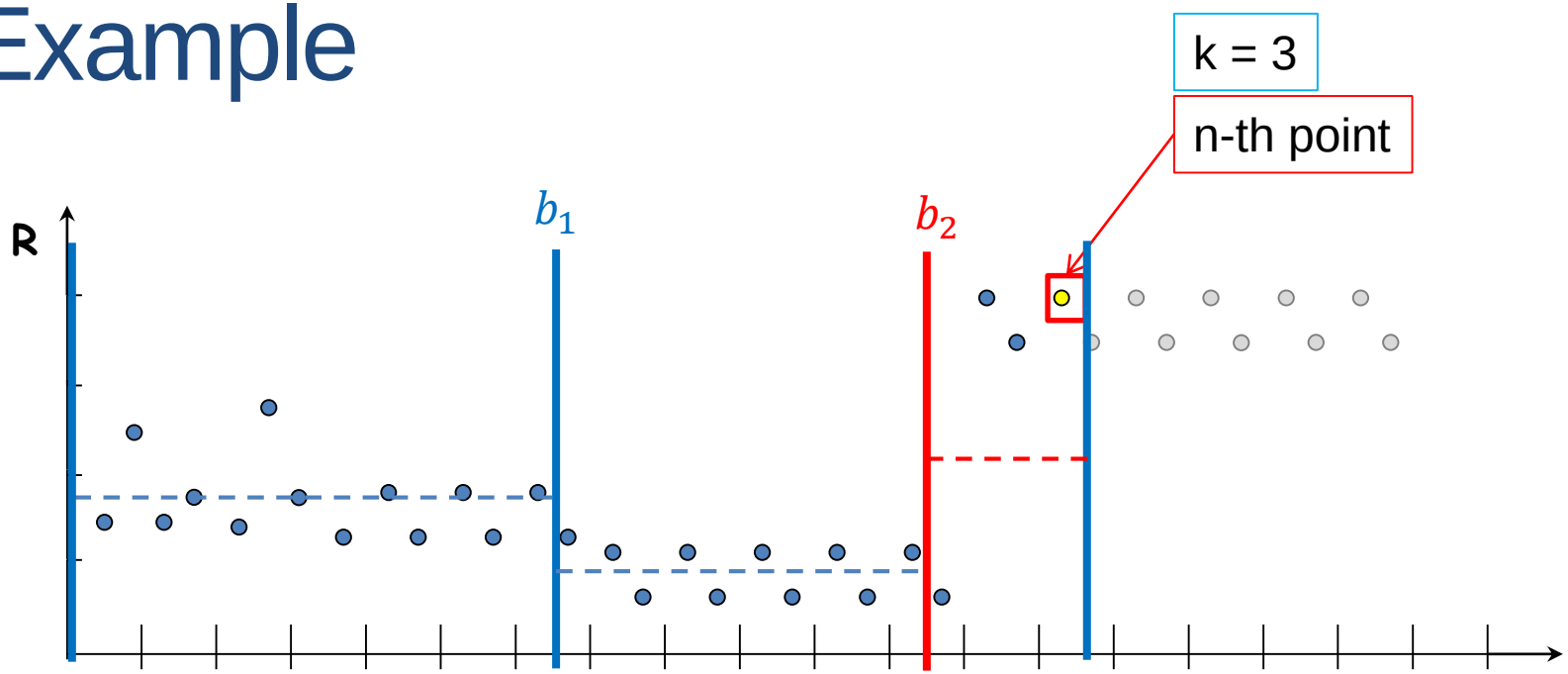


$$E(S[1,n],k) = \min_{k \le j \le n-1} \{ E(S[1,j],k-1) \}$$

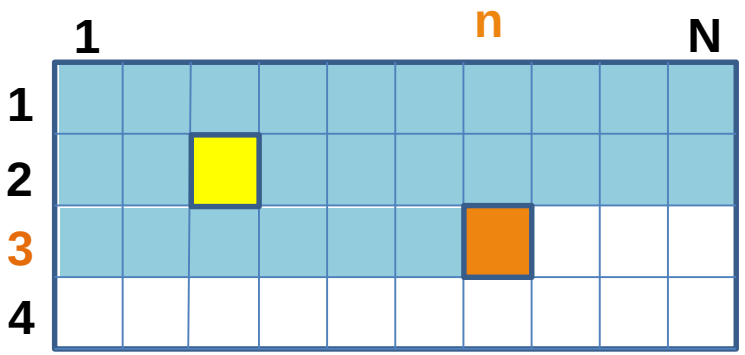


Where should we place boundary b_2 ?

Example

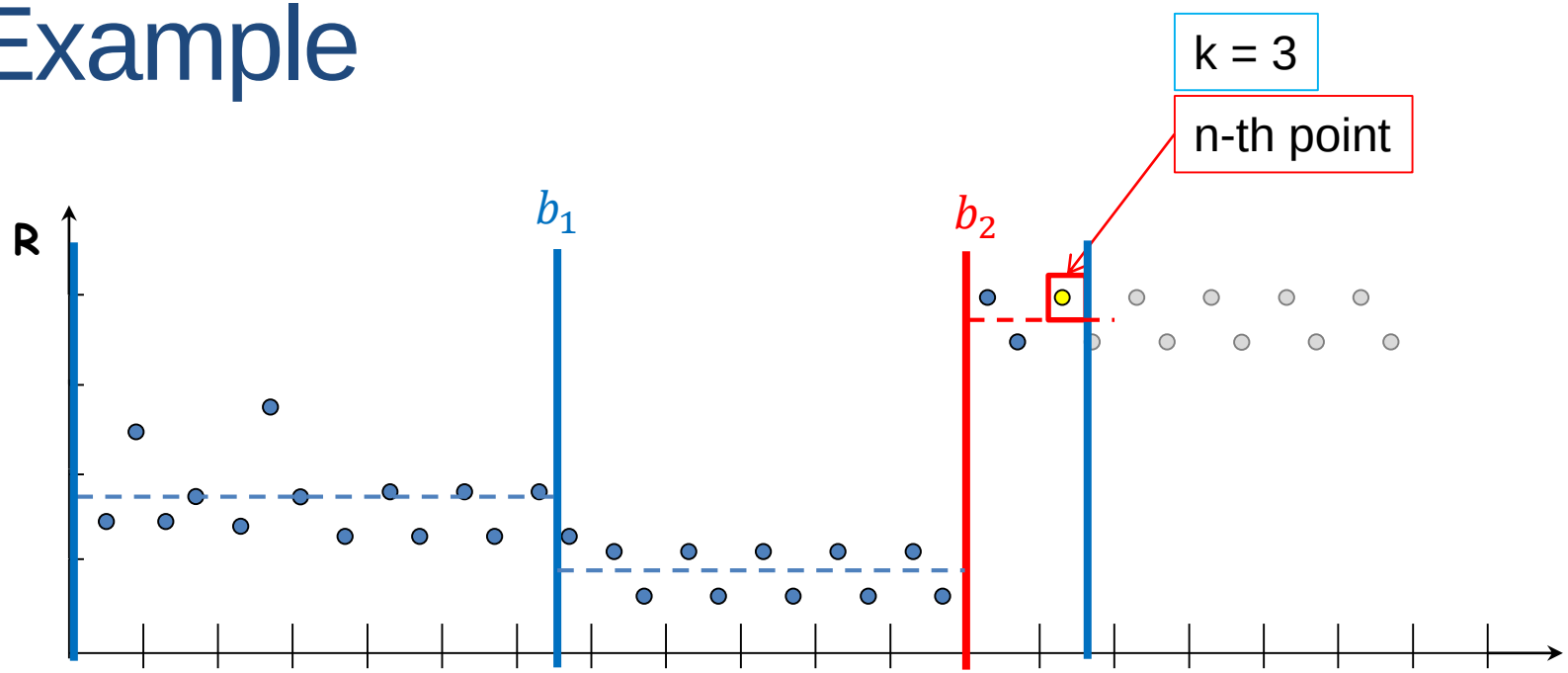


$$E(S[1,n],k)$$
$$= \min_{k \le j \le n-1} \{ E(S[1,j],k-1)$$



Where should we place boundary b_2 ?

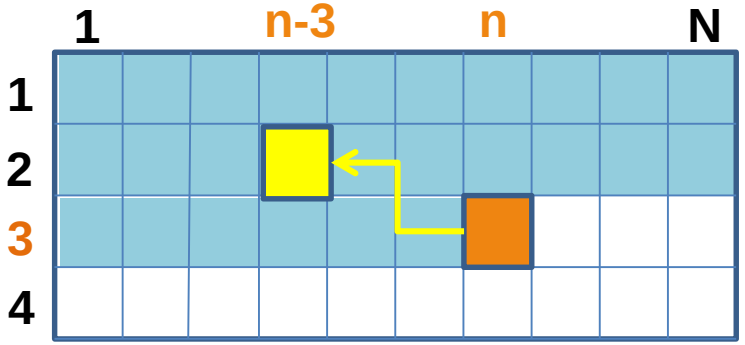
Example



Optimal segmentation S[1:n]

The cell $A[3, n]$ stores the error of the optimal solution 3-segmentation of $T[1, n]$

In the cell (or in a different table) we also store the position $n - 3$ of the boundary so we can trace back the segmentation



Dynamic-programming algorithm

- Input: Sequence **T**, length **N**, **K** segments, error function **E()**

  - For **i=1** to **N** //Initialize first row
    - **A[1,i]=E(T[1…i])** //Error when everything is in one cluster
  - For **k=1** to **K** // Initialize diagonal
    - **A[k,k] = 0** // Error when each point in its own cluster
  - For **k=2** to **K**
    - For **i=k+1** to **N**
      - **$A[k,i] = \min_{j<i}\{A[k-1,j]+E(T[j+1…i])\}$**

- To recover the actual segmentation (not just the optimal cost) store also the minimizing values **j**

Algorithm Complexity

- What is the complexity?
- NK cells to fill
- Computation per cell $E(S[1,n],k) = \min_{k \le j < n} \{E(S[1,j],k-1) + \sum_{j+1 \le t \le n}(t -$

Heuristics

- Top-down greedy (TD): O(NK)
  - Introduce boundaries one at the time so that you get the largest decrease in error, until K segments are created.
- Bottom-up greedy (BU): O(NlogN)
  - Merge adjacent points each time selecting the two points that cause the smallest increase in the error until K segments
- Local Search Heuristics: O(NKI)
  - Assign the breakpoints randomly and then move them so that you reduce the error

Local Search Heuristics

- Local Search refers to a class of heuristic optimization algorithms where we start with some solution and we try to reach an optimum by iteratively improving the solution with small (local) changes
  - Each solution has a set of neighboring solutions:
    - The set of solutions that can be created with the allowed local changes.
  - Usually we move to the best of the neighboring solutions, or one that improves our optimization function
- Local Search algorithms are surprisingly effective
  - For some problems they yield optimal solutions or solutions with good approximation bounds
- They have been studied extensively
  - Simulated Annealing
  - Taboo Search

Other time series analysis

- Using signal processing techniques is common for defining similarity between series
  - Fast Fourier Transform
  - Wavelets
- Rich literature in the field