Data Mining and Analysis: Fundamental Concepts and Algorithms

dataminingbook.info

Mohammed J. Zaki[1] Wagner Meira Jr.[2]

[1]Department of Computer Science
Rensselaer Polytechnic Institute, Troy, NY, USA

[2]Department of Computer Science
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Chapter 13: Representative-based Clustering

Representative-based Clustering

Given a dataset with n points in a d-dimensional space, $\mathbf{D} = \{\mathbf{x}_i\}_{i=1}^n$, and given the number of desired clusters k, the goal of representative-based clustering is to partition the dataset into k groups or clusters, which is called a *clustering* and is denoted as $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$.

For each cluster C_i there exists a representative point that summarizes the cluster, a common choice being the mean (also called the *centroid*) $\boldsymbol{\mu}_i$ of all points in the cluster, that is,

$$\boldsymbol{\mu}_i = \frac{1}{n_i} \sum_{x_j \in C_i} \mathbf{x}_j$$

where $n_i = |C_i|$ is the number of points in cluster C_i.

A brute-force or exhaustive algorithm for finding a good clustering is simply to generate all possible partitions of n points into k clusters, evaluate some optimization score for each of them, and retain the clustering that yields the best score. However, this is clearly infeasilbe, since there are $O(k^n/k!)$ clusterings of n points into k groups.

K-means Algorithm: Objective

The *sum of squared errors* scoring function is defined as

$$SSE(\mathcal{C}) = \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in C_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

The goal is to find the clustering that minimizes the SSE score:

$$\mathcal{C}^* = \arg\min_{\mathcal{C}} \{SSE(\mathcal{C})\}$$

K-means employs a greedy iterative approach to find a clustering that minimizes the SSE objective. As such it can converge to a local optima instead of a globally optimal clustering.

K-means Algorithm: Objective

K-means initializes the cluster means by randomly generating k points in the data space. Each iteration of K-means consists of two steps: (1) cluster assignment, and (2) centroid update.

Given the k cluster means, in the cluster assignment step, each point $\mathbf{x}_j \in \mathbf{D}$ is assigned to the closest mean, which induces a clustering, with each cluster C_i comprising points that are closer to $\boldsymbol{\mu}_i$ than any other cluster mean. That is, each point $\mathbf{x}_j$ is assigned to cluster C_{j^*},
where

$$j^* = \arg\min_{i=1}^{k} \left\{ \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2 \right\}$$

Given a set of clusters C_i, $i = 1, \ldots, k$, in the centroid update step, new mean values are computed for each cluster from the points in C_i.

The cluster assignment and centroid update steps are carried out iteratively until we reach a fixed point or local minima.

K-Means Algorithm

K-MEANS $(\mathbf{D}, k, \epsilon)$:

$t = 0$

Randomly initialize k centroids: $\boldsymbol{\mu}_1^t, \boldsymbol{\mu}_2^t, \ldots, \boldsymbol{\mu}_k^t \in \mathbb{R}^d$

repeat

  $t \leftarrow t + 1$

  $C_j \leftarrow \emptyset$ for all $j = 1, \cdots, k$

  `// Cluster Assignment Step`

  foreach $\mathbf{x}_j \in \mathbf{D}$ **do**

    $j^* \leftarrow \arg\min_i \left\{ \left\| \mathbf{x}_j - \boldsymbol{\mu}_i^t \right\|^2 \right\}$ `// Assign` $\mathbf{x}_j$ `to closest centroid`

    $C_{j^*} \leftarrow C_{j^*} \cup \{\mathbf{x}_j\}$

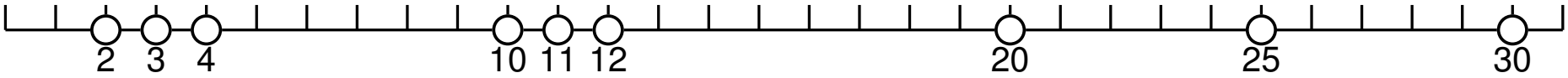
  `// Centroid Update Step`

  foreach $i = 1$ *to* k **do**

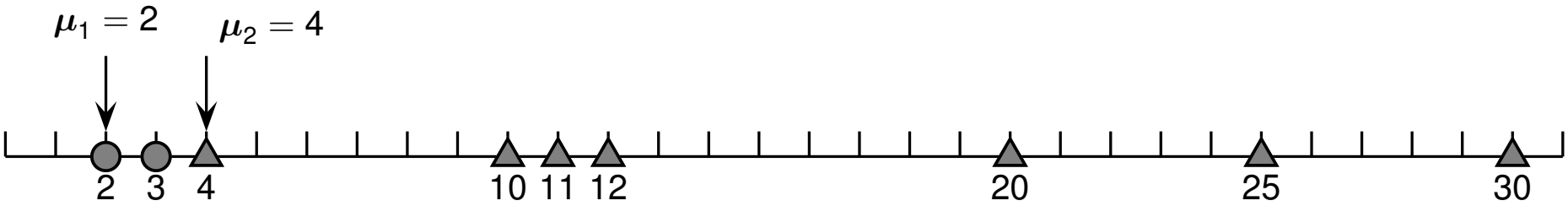
    $\boldsymbol{\mu}_i^t \leftarrow \frac{1}{|C_i|} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j$

until $\sum_{i=1}^k \left\| \boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1} \right\|^2 \leq \epsilon$

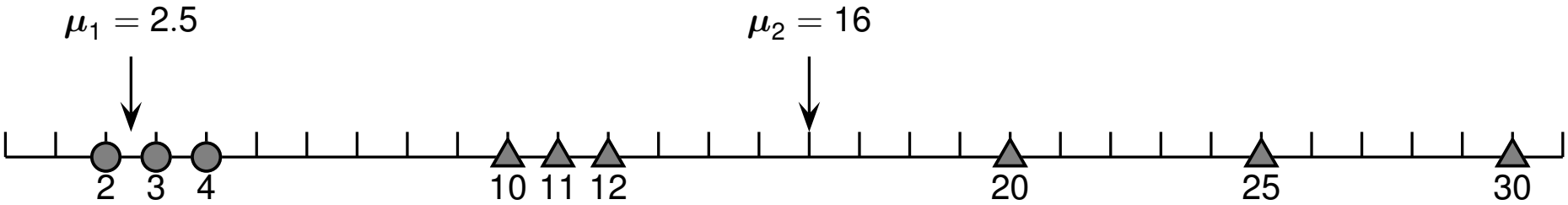
K-means in One Dimension



(a) Initial dataset

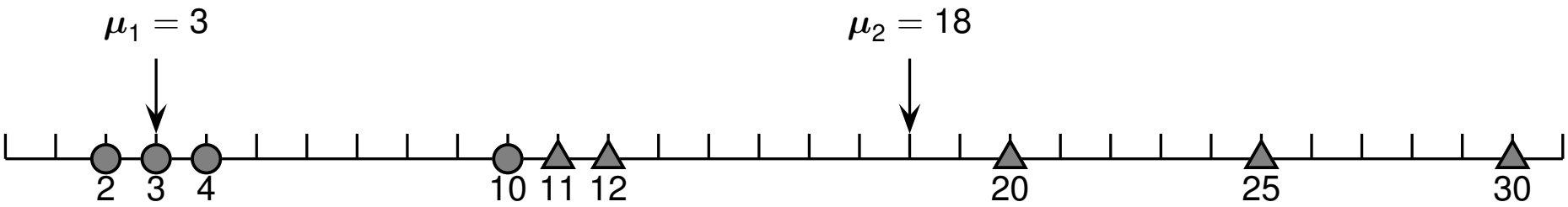


(b) Iteration: $t = 1$

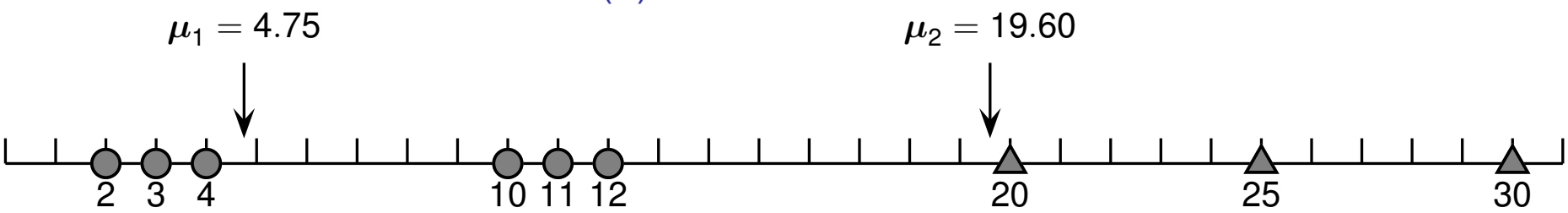


(c) Iteration: $t = 2$

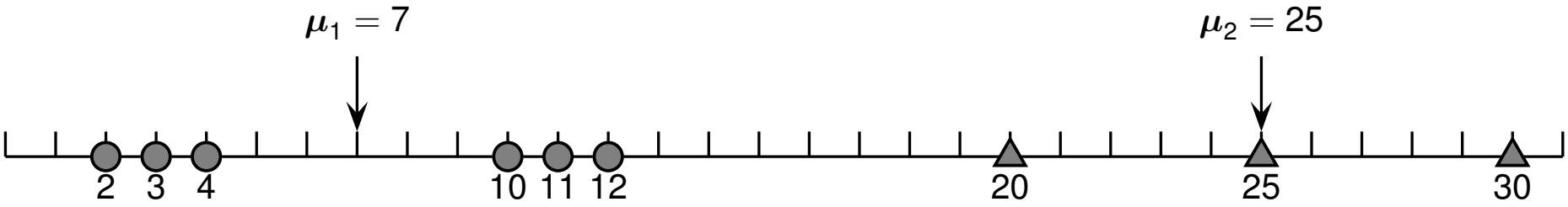
K-means in One Dimension (contd.)



(d) Iteration: $t = 3$

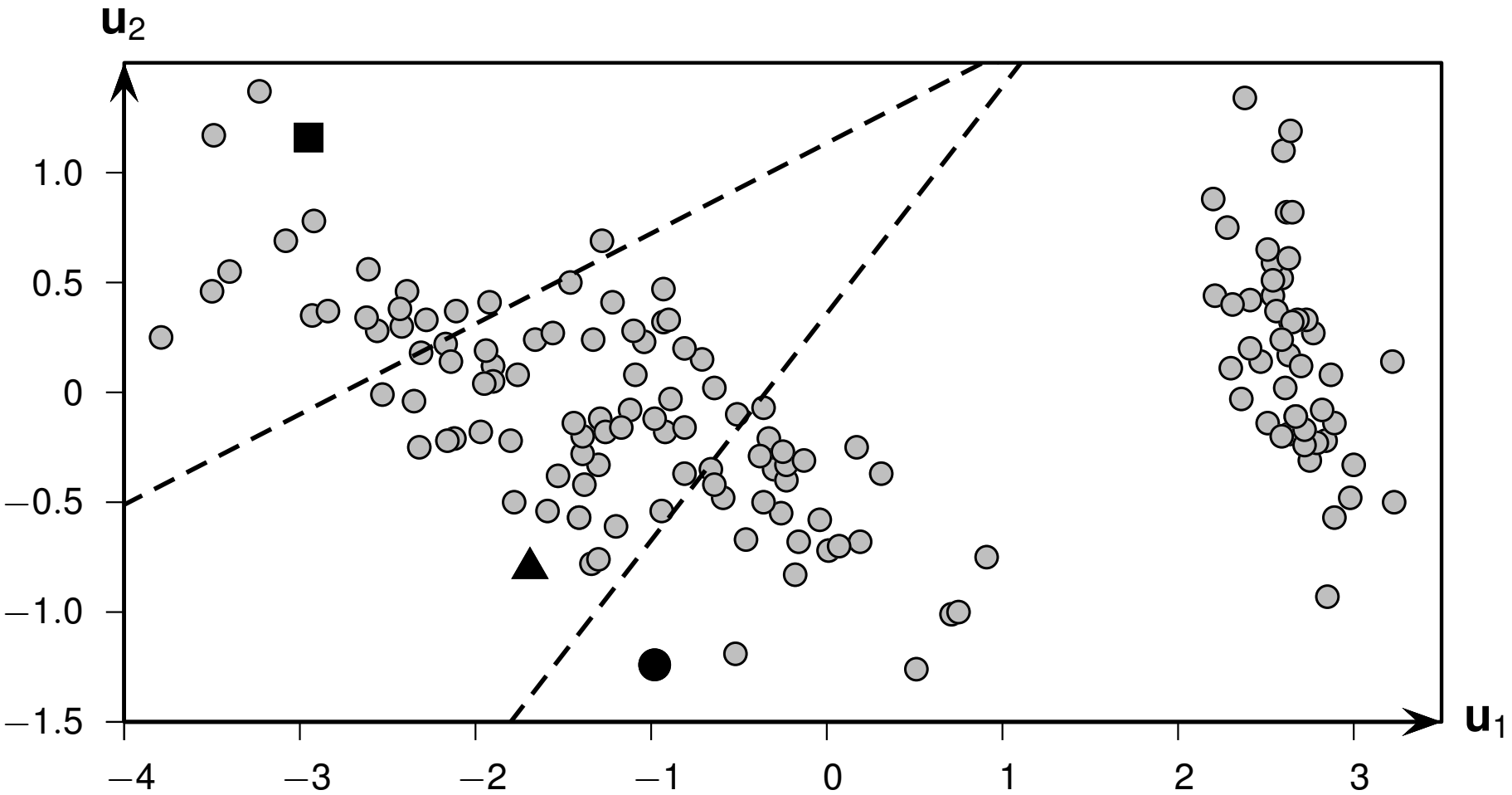


(e) Iteration: $t = 4$



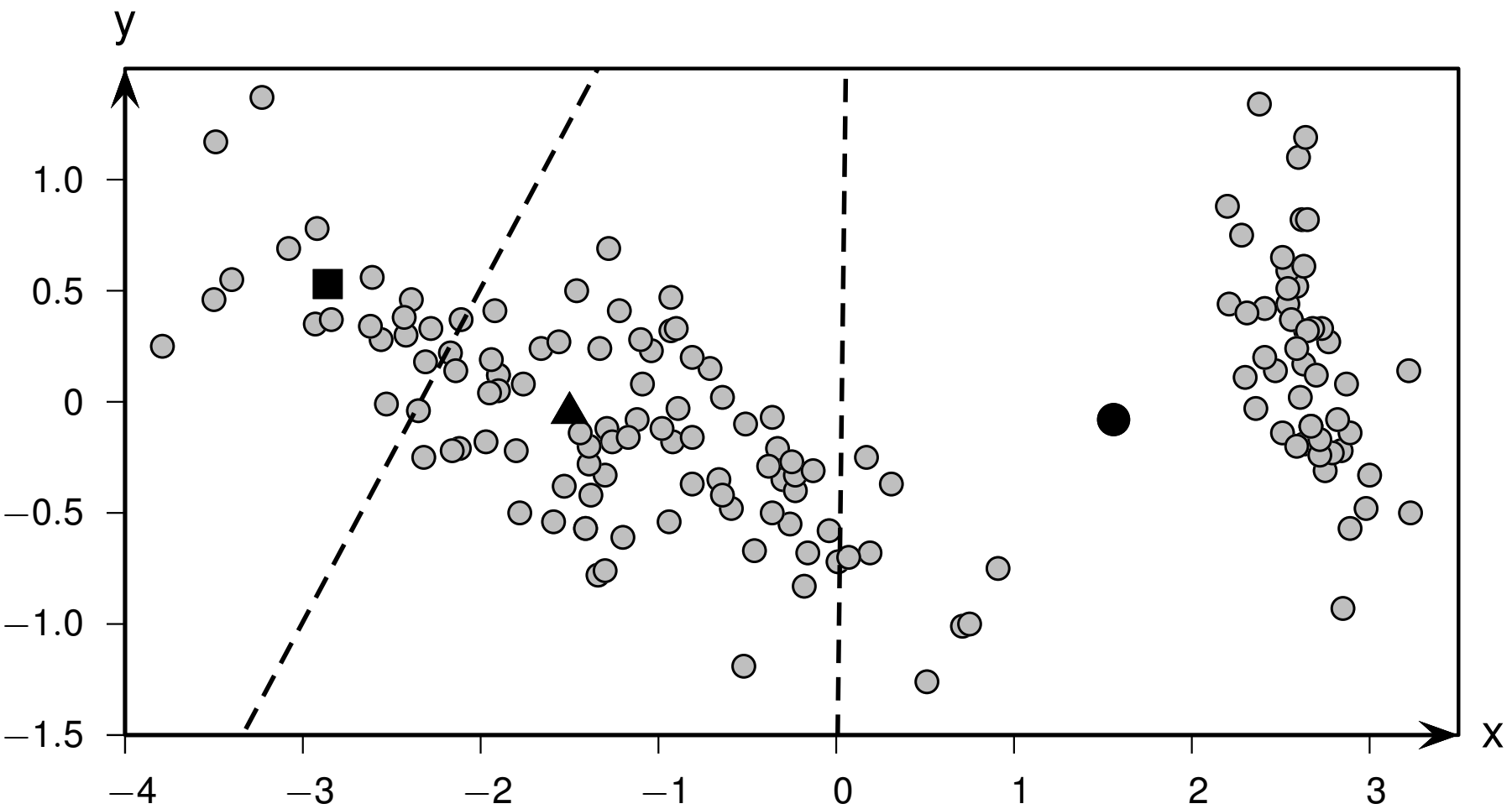
(f) Iteration: $t = 5$ (converged)

K-means in 2D: Iris Principal Components



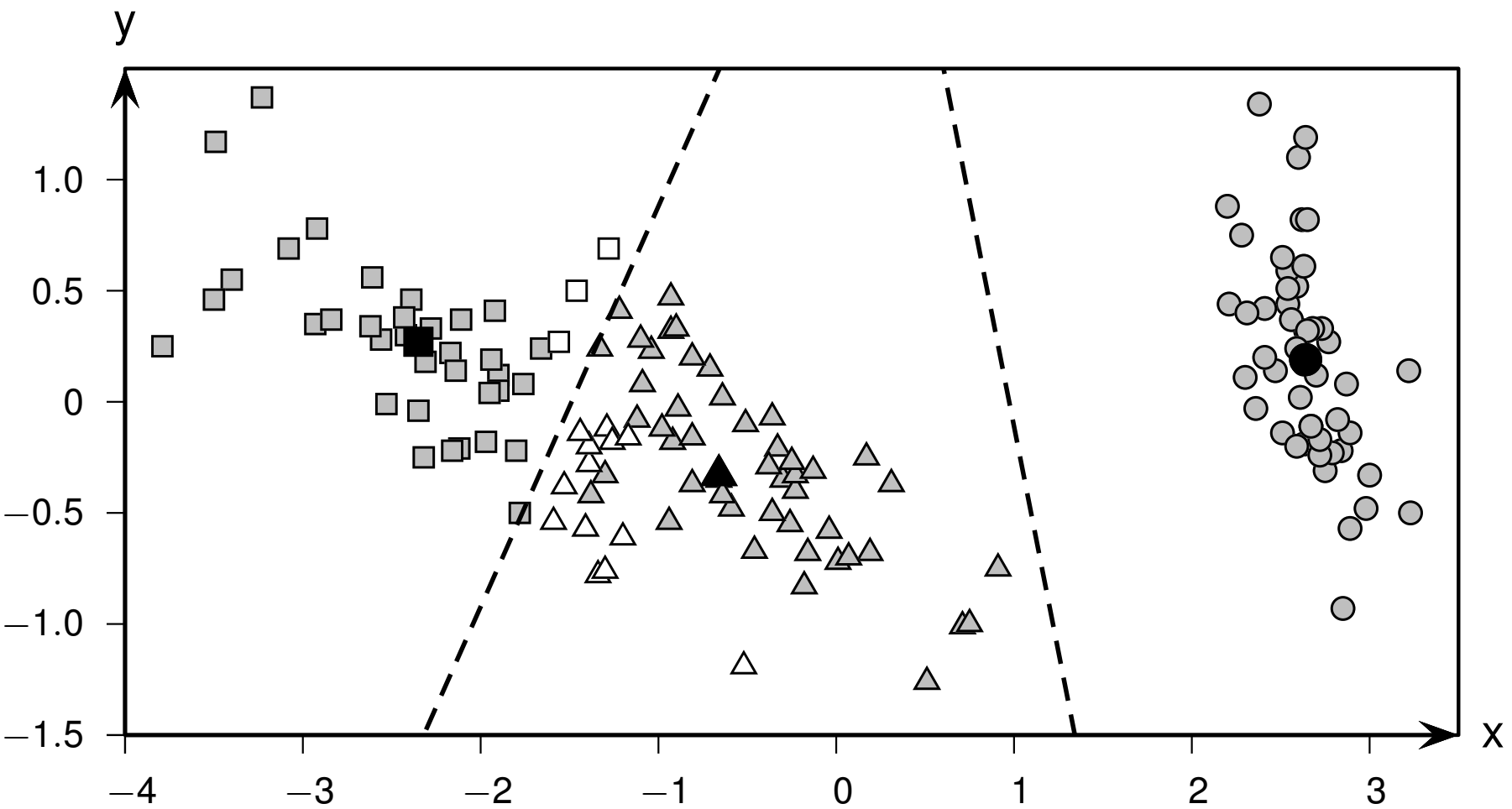
(a) Random initialization: $t = 0$

K-means in 2D: Iris Principal Components



(b) Iteration: $t = 1$

K-means in 2D: Iris Principal Components



(c) Iteration: $t = 8$ (converged)

Kernel K-means

In K-means, the separating boundary between clusters is linear. Kernel K-means allows one to extract nonlinear boundaries between clusters via the use of the kernel trick, i.e., we show that all operations involve only the kernel value between a pair of points.

Let $\mathbf{x}_i \in \mathbf{D}$ be mapped to $\phi(\mathbf{x}_i)$ in feature space. Let $\mathbf{K} = \{K(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1,\ldots,n}$ denote the $n \times n$ symmetric kernel matrix, where $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$.

The cluster means in feature space are $\{\boldsymbol{\mu}_1^\phi, \ldots, \boldsymbol{\mu}_k^\phi\}$, where $\boldsymbol{\mu}_i^\phi = \frac{1}{n_i} \sum_{\mathbf{x}_j \in C_i} \phi(\mathbf{x}_j)$.

The sum of squared errors in feature space is

$$\min_{\mathcal{C}} \; SSE(\mathcal{C}) = \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in C_i} \left\| \phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\phi \right\|^2 = \sum_{j=1}^{n} K(\mathbf{x}_j, \mathbf{x}_j) - \sum_{i=1}^{k} \frac{1}{n_i} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b)$$

Thus, the kernel K-means SSE objective function can be expressed purely in terms of the kernel function.

Kernel K-means: Cluster Reassignment

Consider the distance of a point $\phi(\mathbf{x}_j)$ to the mean $\boldsymbol{\mu}_i^\phi$ in feature space

$$\left\|\phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\phi\right\|^2 = \|\phi(\mathbf{x}_j)\|^2 - 2\phi(\mathbf{x}_j)^T\boldsymbol{\mu}_i^\phi + \left\|\boldsymbol{\mu}_i^\phi\right\|^2$$

$$= K(\mathbf{x}_j, \mathbf{x}_j) - \frac{2}{n_i}\sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j) + \frac{1}{n_i^2}\sum_{\mathbf{x}_a \in C_i}\sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b)$$

Kernel K-means assign a point to the closest cluster mean as follows:

$$C^*(\mathbf{x}_j) = \arg\min_i \left\{\left\|\phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\phi\right\|^2\right\}$$

$$= \arg\min_i \left\{\frac{1}{n_i^2}\sum_{\mathbf{x}_a \in C_i}\sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b) - \frac{2}{n_i}\sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j)\right\}$$

Kernel-Kmeans Algorithm

KERNEL-KMEANS($\mathbf{K}, k, \epsilon$):

$t \leftarrow 0$

$\mathcal{C}^t \leftarrow \{C_1^t, \ldots, C_k^t\}$ // Randomly partition points into k clusters

repeat

  $t \leftarrow t + 1$

  foreach $C_i \in \mathcal{C}^{t-1}$ **do** // Compute squared norm of cluster means

    $\text{sqnorm}_i \leftarrow \frac{1}{n_i^2} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b)$

  foreach $\mathbf{x}_j \in \mathbf{D}$ **do** // Average kernel value for $\mathbf{x}_j$ and C_i

    foreach $C_i \in \mathcal{C}^{t-1}$ **do**

      $\text{avg}_{ji} \leftarrow \frac{1}{n_i} \sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j)$

  // Find closest cluster for each point

  foreach $\mathbf{x}_j \in \mathbf{D}$ **do**

    foreach $C_i \in \mathcal{C}^{t-1}$ **do**

      $d(\mathbf{x}_j, C_i) \leftarrow \text{sqnorm}_i - 2 \cdot \text{avg}_{ji}$

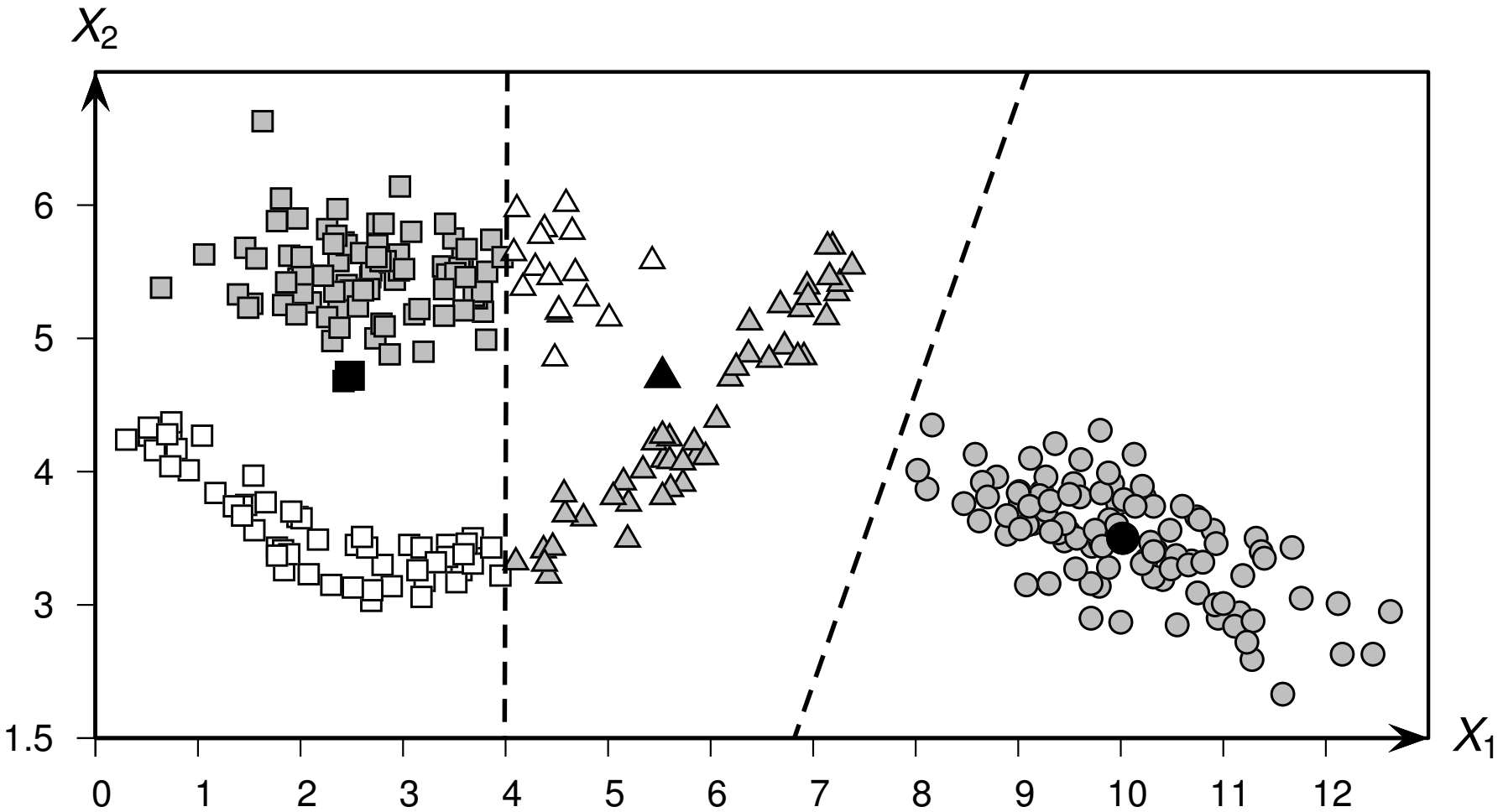
    $j^* \leftarrow \arg\min_i \{d(\mathbf{x}_j, C_i)\}$

    $C_{j^*}^t \leftarrow C_{j^*}^t \cup \{\mathbf{x}_j\}$ // Cluster reassignment

  $\mathcal{C}^t \leftarrow \{C_1^t, \ldots, C_k^t\}$

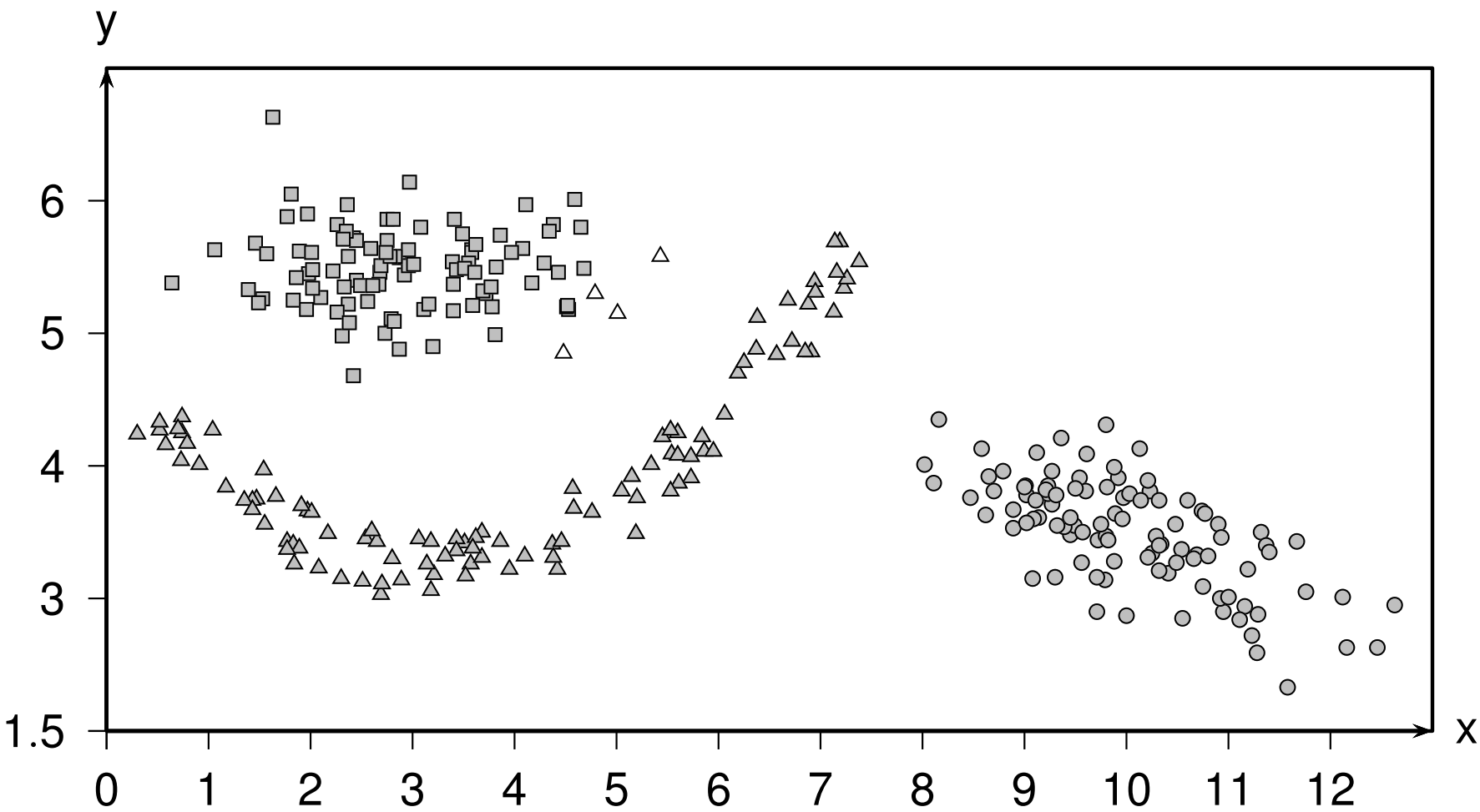
until $\frac{1}{n} \sum_{i=1}^{k} |C_i^t \ominus C_i^{t-1}| \leq \epsilon$

K-Means or Kernel K-means with Linear Kernel



(a) Linear kernel: $t = 5$ iterations

Kernel K-means: Gaussian Kernel



(b) Gaussian kernel: $t = 4$ Iterations

Expectation-Maximization Clustering: Gaussian Mixture Model

Let X_a denote the random variable corresponding to the ath attribute. Let $\mathbf{X} = (X_1, X_2, \ldots, X_d)$ denote the vector random variable across the d-attributes, with $\mathbf{x}_j$ being a data sample from $\mathbf{X}$.

We assume that each cluster C_i is characterized by a multivariate normal distribution

$$f_i(\mathbf{x}) = f(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp\left\{-\frac{(\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)}{2}\right\}$$

where the cluster mean $\boldsymbol{\mu}_i \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma}_i \in \mathbb{R}^{d\times d}$ are both unknown parameters.

The probability density function of $\mathbf{X}$ is given as a *Gaussian mixture model* over all the k clusters

$$f(\mathbf{x}) = \sum_{i=1}^{k} f_i(\mathbf{x})P(C_i) = \sum_{i=1}^{k} f(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)P(C_i)$$

where the prior probabilities $P(C_i)$ are called the *mixture parameters*, which must satisfy the condition

$$\sum_{i=1}^{k} P(C_i) = 1$$

Expectation-Maximization Clustering: Maximum Likelihood Estimation

We write the set of all the model parameters compactly as

$$\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, P(C_i) \ldots, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, P(C_k)\}$$

Given the dataset $\mathbf{D}$, we define the *likelihood* of $\boldsymbol{\theta}$ as the conditional probability of the data $\mathbf{D}$ given the model parameters $\boldsymbol{\theta}$

$$P(\mathbf{D}|\boldsymbol{\theta}) = \prod_{j=1}^{n} f(\mathbf{x}_j)$$

The goal of maximum likelihood estimation (MLE) is to choose the parameters $\boldsymbol{\theta}$ that maximize the likelihood. We do this by maximizing the log of the likelihood function

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \{\ln P(\mathbf{D}|\boldsymbol{\theta})\}$$

where the *log-likelihood* function is given as

$$\ln P(\mathbf{D}|\boldsymbol{\theta}) = \sum_{j=1}^{n} \ln f(\mathbf{x}_j) = \sum_{j=1}^{n} \ln \left(\sum_{i=1}^{k} f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) P(C_i) \right)$$

Expectation-Maximization Clustering

Directly maximizing the log-likelihood over θ is hard. Instead, we can use the expectation-maximization (EM) approach for finding the maximum likelihood estimates for the parameters θ.

EM is a two-step iterative approach that starts from an initial guess for the parameters θ. Given the current estimates for θ, in the *expectation step* EM computes the cluster posterior probabilities $P(C_i|\mathbf{x}_j)$ via the Bayes theorem:

$$P(C_i|\mathbf{x}_j) = \frac{P(C_i \text{ and } \mathbf{x}_j)}{P(\mathbf{x}_j)} = \frac{P(\mathbf{x}_j|C_i)P(C_i)}{\sum_{a=1}^{k} P(\mathbf{x}_j|C_a)P(C_a)} = \frac{f_i(\mathbf{x}_j) \cdot P(C_i)}{\sum_{a=1}^{k} f_a(\mathbf{x}_j) \cdot P(C_a)}$$

In the *maximization step*, using the weights $P(C_i|\mathbf{x}_j)$ EM re-estimates θ, that is, it re-estimates the parameters $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$, and $P(C_i)$ for each cluster C_i. The re-estimated mean is given as the weighted average of all the points, the re-estimated covariance matrix is given as the weighted covariance over all pairs of dimensions, and the re-estimated prior probability for each cluster is given as the fraction of weights that contribute to that cluster.

EM in One Dimension: Expectation Step

Let **D** comprise of a single attribute X, with each point $x_j \in \mathbb{R}$ a random sample from X. For the mixture model, we use univariate normals for each cluster:

$$f_i(x) = f(x|\mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left\{-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right\}$$

with the cluster parameters μ_i, σ_i^2, and $P(C_i)$.

Initialization: For each cluster C_i, with $i = 1, 2, \ldots, k$, we can randomly initialize the cluster parameters μ_i, σ_i^2, and $P(C_i)$.

Expectation Step: Given the mean μ_i, variance σ_i^2, and prior probability $P(C_i)$ for each cluster, the cluster posterior probability is computed as

$$w_{ij} = P(C_i|x_j) = \frac{f(x_j|\mu_i, \sigma_i^2) \cdot P(C_i)}{\sum_{a=1}^{k} f(x_j|\mu_a, \sigma_a^2) \cdot P(C_a)}$$

EM in One Dimension: Maximization Step

Given w_{ij} values, the re-estimated cluster mean is

$$\mu_i = \frac{\sum_{j=1}^{n} w_{ij} \cdot x_j}{\sum_{j=1}^{n} w_{ij}}$$

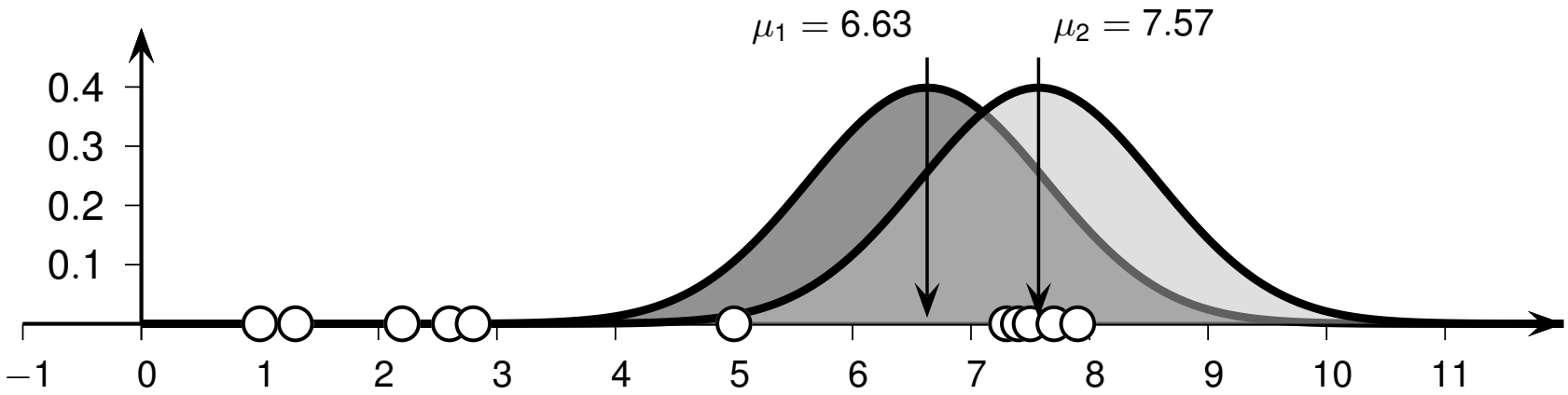
The re-estimated value of the cluster variance is computed as the weighted variance across all the points:

$$\sigma_i^2 = \frac{\sum_{j=1}^{n} w_{ij}(x_j - \mu_i)^2}{\sum_{j=1}^{n} w_{ij}}$$

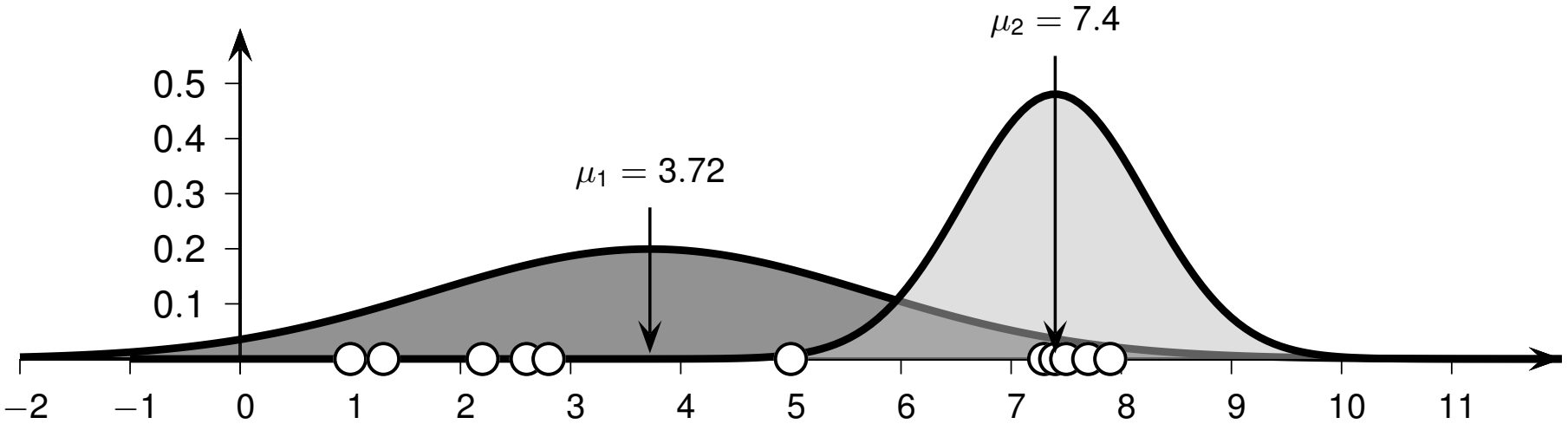
The prior probability of cluster C_i is re-estimated as

$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{n}$$

EM in One Dimension

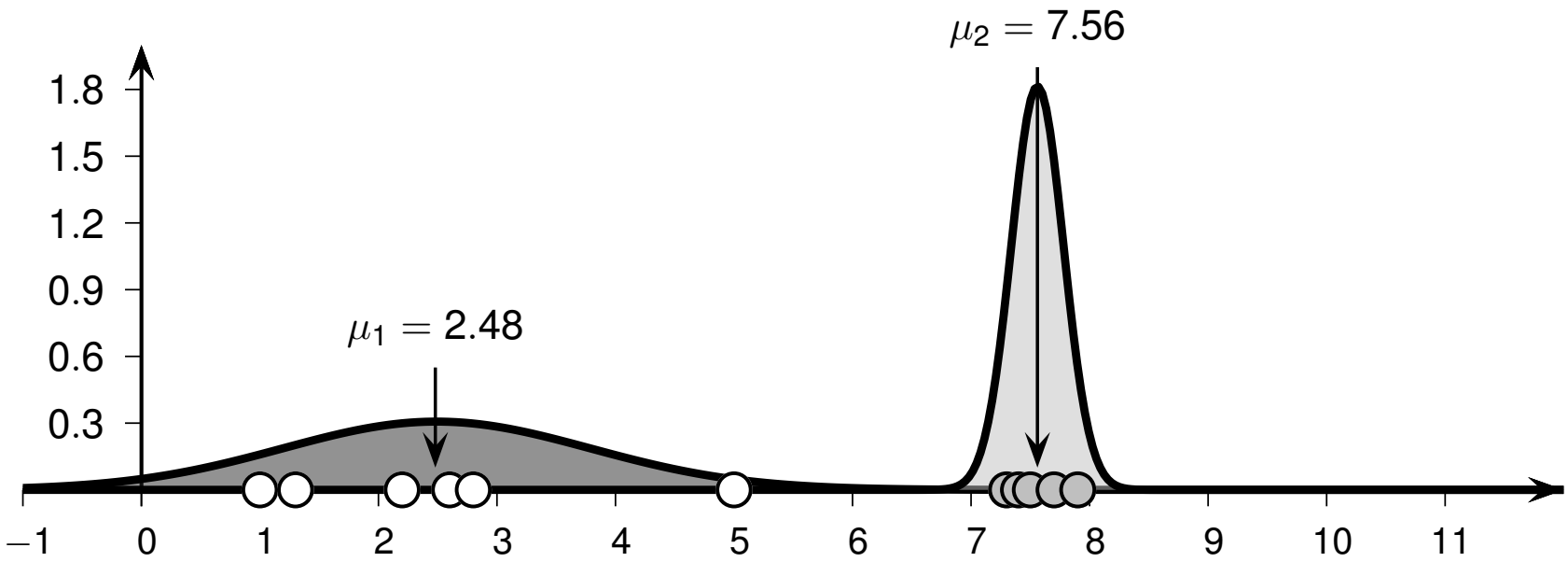


(a) Initialization: $t = 0$



(b) Iteration: $t = 1$

EM in One Dimension: Final Clusters



(c) Iteration: $t = 5$ (converged)

EM in d Dimensions

Each cluster we have to reestiamte the $d \times d$ covariance matrix:

$$\mathbf{\Sigma}_i = \begin{pmatrix} (\sigma_1^i)^2 & \sigma_{12}^i & \cdots & \sigma_{1d}^i \\ \sigma_{21}^i & (\sigma_2^i)^2 & \cdots & \sigma_{2d}^i \\ \vdots & \vdots & \ddots & \\ \sigma_{d1}^i & \sigma_{d2}^i & \cdots & (\sigma_d^i)^2 \end{pmatrix}$$

It requires $O(d^2)$ parameters, which may be too many for reliable estimation. A simplification is to assume that all dimensions are independent, which leads to a diagonal covariance matrix:

$$\mathbf{\Sigma}_i = \begin{pmatrix} (\sigma_1^i)^2 & 0 & \cdots & 0 \\ 0 & (\sigma_2^i)^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & (\sigma_d^i)^2 \end{pmatrix}$$

EM in d Dimensions

Expectation Step: Given $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$, and $P(C_i)$, the posterior probability is given as

$$w_{ij} = P(C_i|\mathbf{x}_j) = \frac{f_i(\mathbf{x}_j) \cdot P(C_i)}{\sum_{a=1}^{k} f_a(\mathbf{x}_j) \cdot P(C_a)}$$

Maximization Step: Given the weights w_{ij}, in the maximization step, we re-estimate $\boldsymbol{\Sigma}_i$, $\boldsymbol{\mu}_i$ and $P(C_i)$.
The mean $\boldsymbol{\mu}_i$ for cluster C_i can be estimated as

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^{n} w_{ij} \cdot \mathbf{x}_j}{\sum_{j=1}^{n} w_{ij}}$$

The covariance matrix $\boldsymbol{\Sigma}_i$ is re-estimated via the outer-product form

$$\boldsymbol{\Sigma}_i = \frac{\sum_{j=1}^{n} w_{ij}(\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^{n} w_{ij}}$$

The prior probability $P(C_i)$ for each cluster is

$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{n}$$

Expectation-Maximization Clustering Algorithm

EXPECTATION-MAXIMIZATION $(\mathbf{D}, k, \epsilon)$:

1 $t \leftarrow 0$

2 Randomly initialize $\boldsymbol{\mu}_1^t, \ldots, \boldsymbol{\mu}_k^t$

3 $\boldsymbol{\Sigma}_i^t \leftarrow \mathbf{I},\ \forall i = 1, \ldots, k$

4 **repeat**

5 $\quad t \leftarrow t + 1$

6 $\quad$ **for** $i = 1, \ldots, k$ *and* $j = 1, \ldots, n$ **do**

7 $\quad\quad w_{ij} \leftarrow \frac{f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \cdot P(C_i)}{\sum_{a=1}^{k} f(\mathbf{x}_j|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a) \cdot P(C_a)}$ // posterior probability $P^t(C_i|\mathbf{x}_j)$

8 $\quad$ **for** $i = 1, \ldots, k$ **do**

9 $\quad\quad \boldsymbol{\mu}_i^t \leftarrow \frac{\sum_{j=1}^{n} w_{ij} \cdot \mathbf{x}_j}{\sum_{j=1}^{n} w_{ij}}$ // re-estimate mean

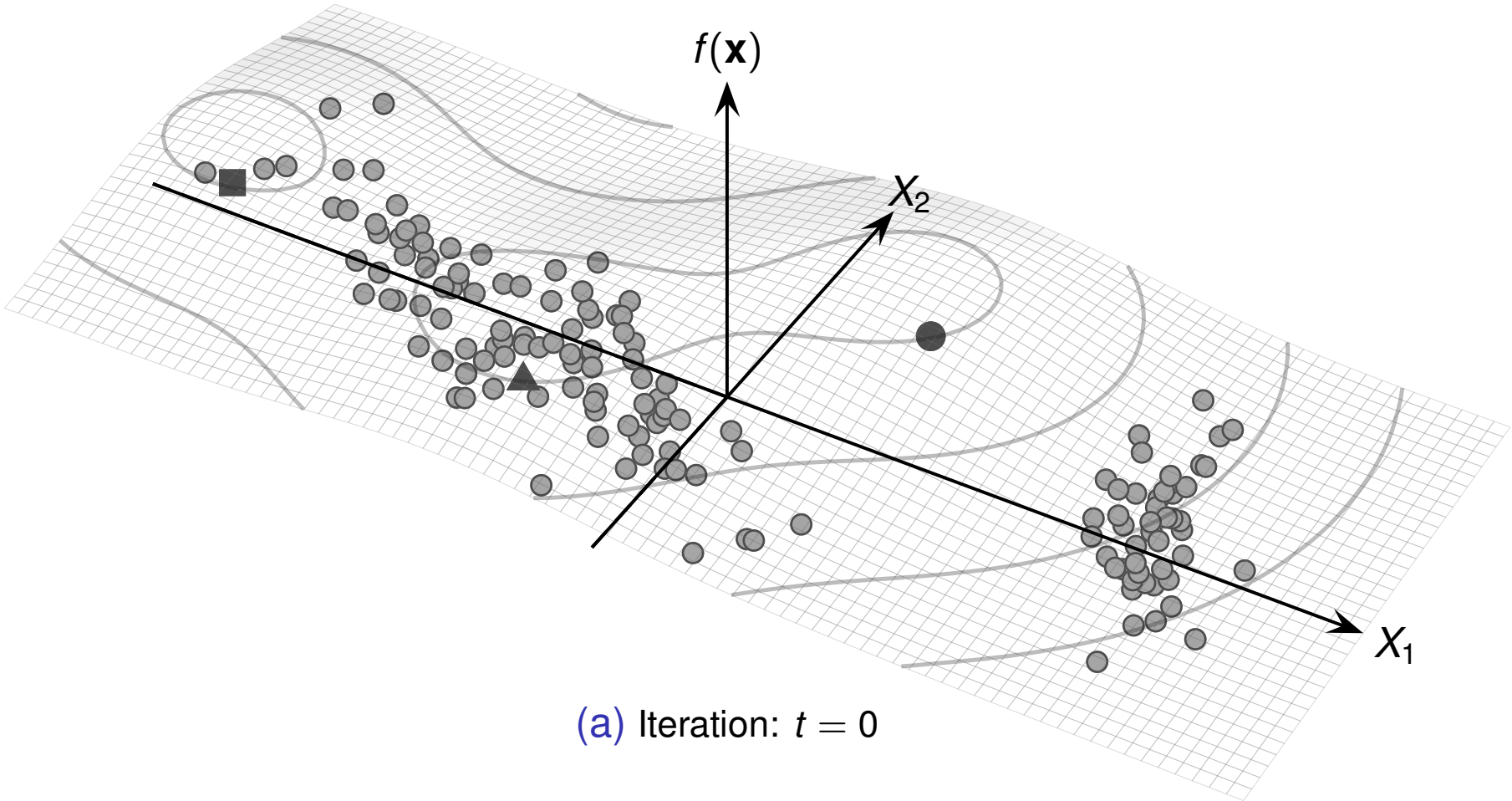
10 $\quad\quad \boldsymbol{\Sigma}_i^t \leftarrow \frac{\sum_{j=1}^{n} w_{ij} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^{n} w_{ij}}$ // re-estimate covariance matrix

11 $\quad\quad P^t(C_i) \leftarrow \frac{\sum_{j=1}^{n} w_{ij}}{n}$ // re-estimate priors

12 **until** $\sum_{i=1}^{k} \left\| \boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1} \right\|^2 \leq \epsilon$

EM Clustering in 2D

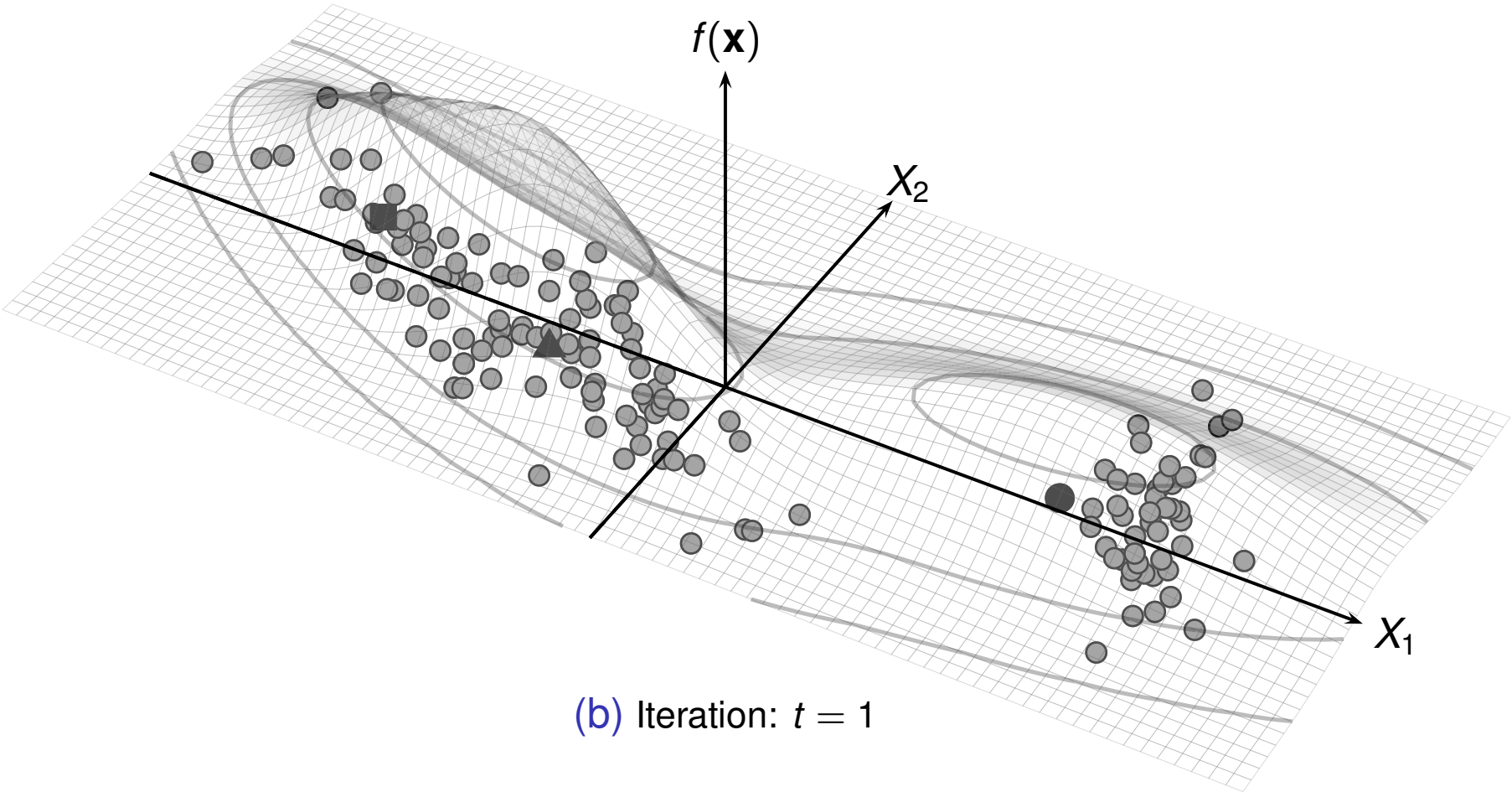
Mixture of $k = 3$ Gaussians



(a) Iteration: $t = 0$

EM Clustering in 2D

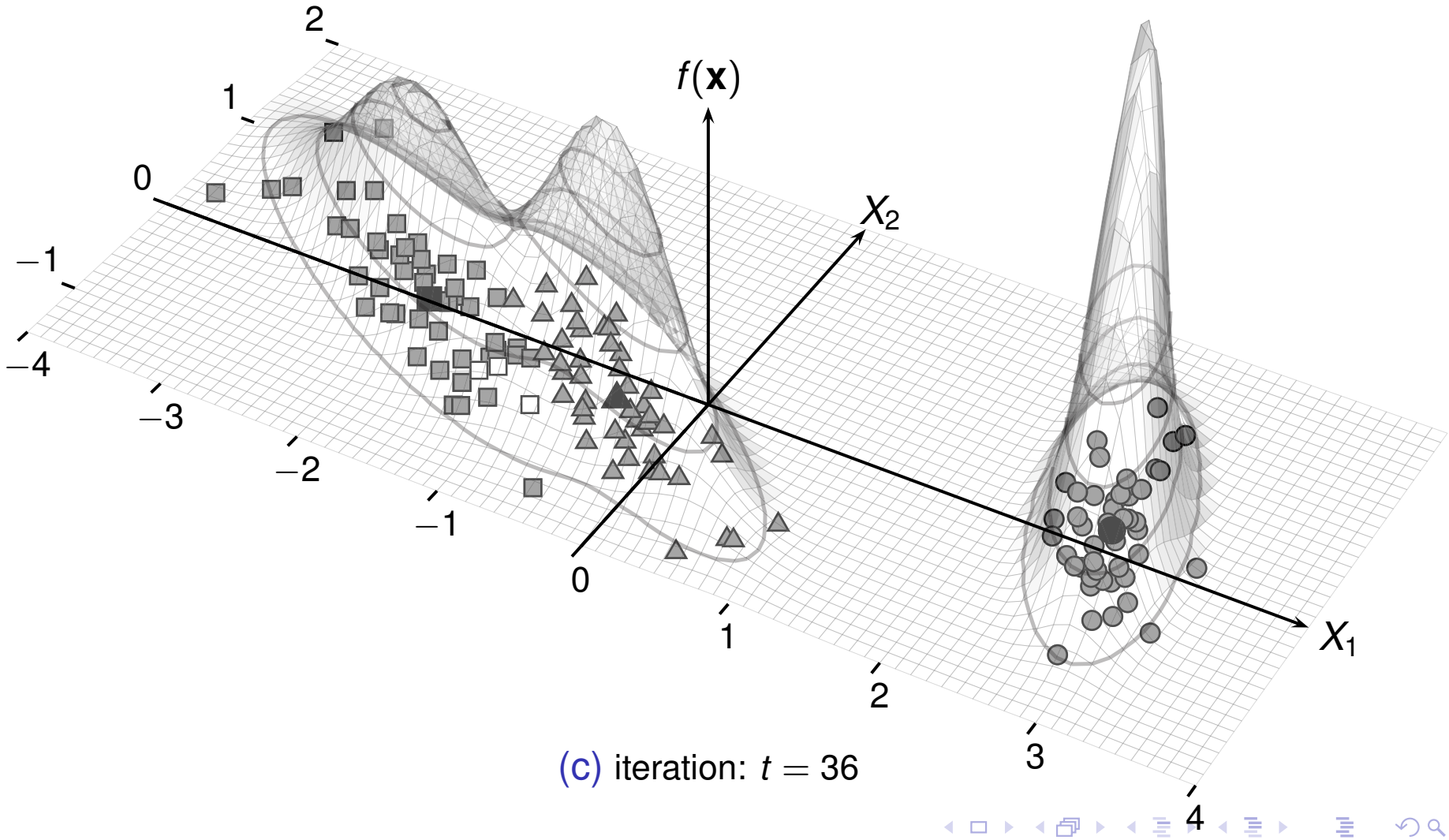
Mixture of $k = 3$ Gaussians



(b) Iteration: $t = 1$

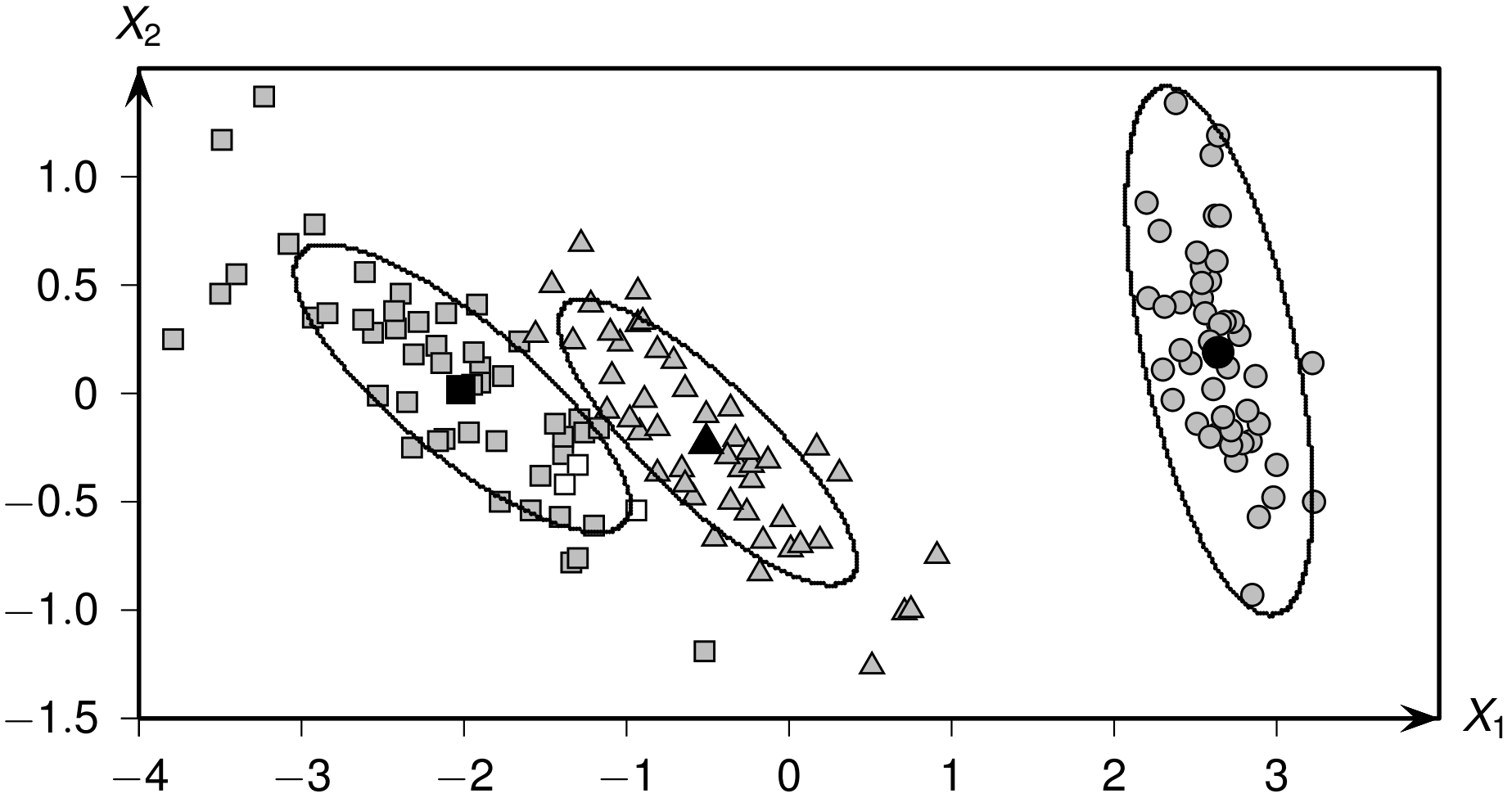
EM Clustering in 2D

Mixture of $k = 3$ Gaussians



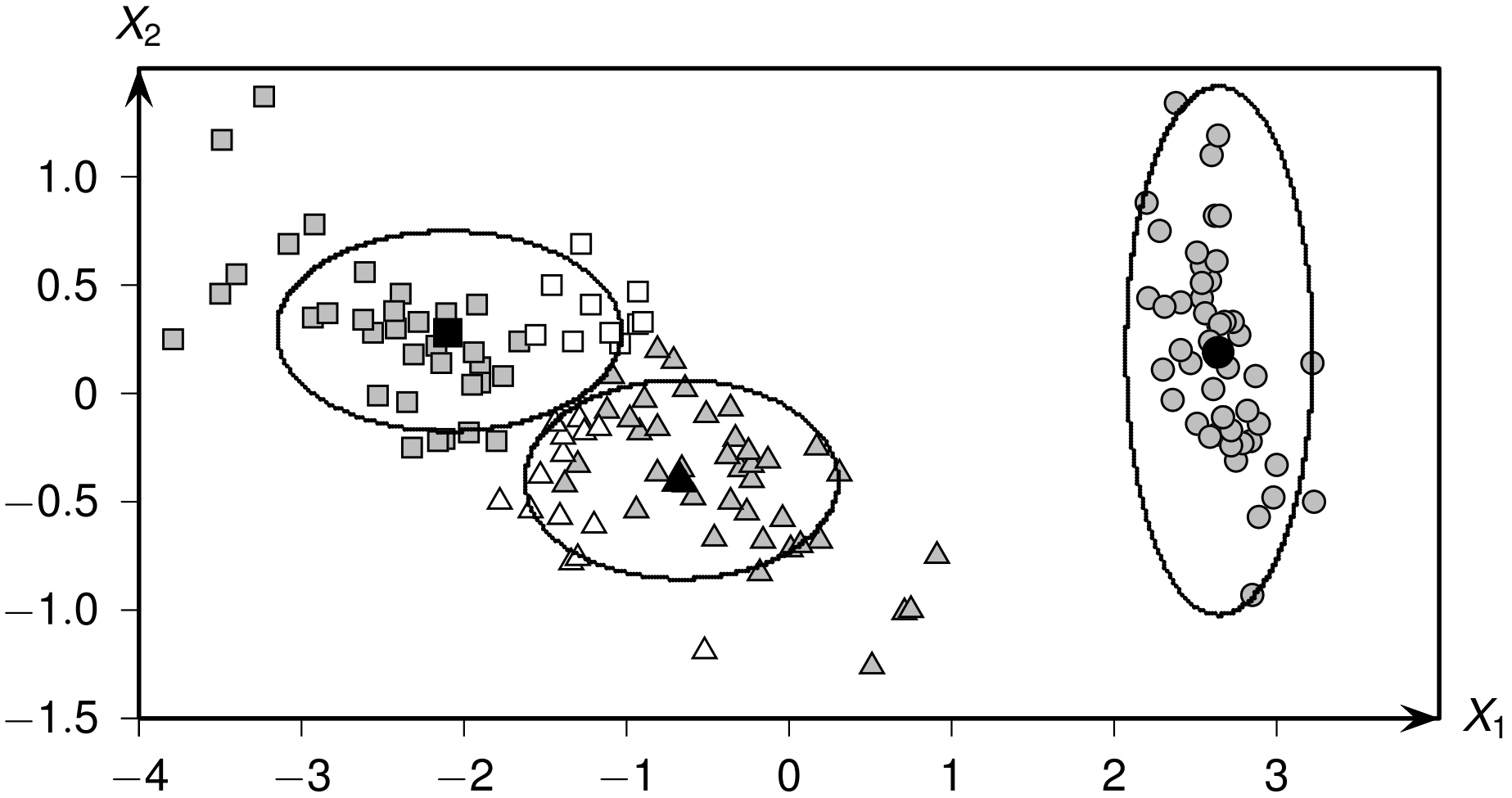
(c) iteration: $t = 36$

Iris Principal Components Data: Full Covariance Matrix



(a) Full covariance matrix ($t = 36$)

Iris Principal Components Data: Diagonal Covariance Matrix



(b) Diagonal covariance matrix ($t = 29$)