Clusters are assumed to be described by a parametric model, whose parameters are unknown.

All parameters are included in a vector denoted by $\theta$.

Examples:

- **Compact clusters.** Each cluster $C_i$ is represented by a point $m_i$ in the $l$-dimensional space. Thus $\theta = [m_1^T, m_2^T, \ldots, m_m^T]^T$.

- **Ring-shaped clusters.** Each cluster $C_i$ is modeled by a hypersphere $C(c_i, r_i)$, where $c_i$ and $r_i$ are its center and its radius, respectively. Thus $\theta = [c_1^T, r_1, c_2^T, r_2, \ldots, c_m^T, r_m]^T$.

![Figure 14.1:](image)
A cost $J(\theta)$ is defined as a function of $\theta$, which describes the data $X$. Optimization of $J(\theta)$ with respect to $\theta$ results in $\theta$ that characterizes optimally the clusters underlying $X$.

The number of clusters $m$ is a priori known in most cases.

Cost optimization clustering algorithms considered in the sequel
- Mixture decomposition schemes.
- Fuzzy clustering algorithms.
- Possibilistic clustering algorithms.
Mixture Decomposition (MD) schemes

- Each vector belongs to a cluster with a certain probability
- MD schemes rely on the Bayesian framework:
  A vector $x_i$ is appointed to cluster $C_j$ if
  $$ P(C_j | x_i) > P(C_k | x_i), \quad k=1,\ldots,m, \quad k\neq j. $$

However:

- No cluster labeling information is available for the data vectors
- The a priori cluster probabilities $P(C_j) = P_j$ are also unknown
Maximum Likelihood Estimation

- Let $C_{\alpha(i)}$ denote the true cluster of $x_i$
- Log-likelihood function

\[
L(\theta) = \sum_{i=1}^{N} \ln p(x_i, C_{\alpha(i)}; \theta) = \sum_{i=1}^{N} \ln(p(x_i|C_{\alpha(i)}; \theta)P_{\alpha(i)})
\]

However, we do not know $\alpha(i)$ and approximate $L(\theta)$ by (E-step)

\[
Q(\theta; \theta(t)) = \sum_{i=1}^{N} \sum_{\alpha(i)=1}^{m} p(C_{\alpha(i)}|x_i; \theta(t)) \ln(p(x_i|C_{\alpha(i)}; \theta)P_{\alpha(i)})
\]

A solution: Adoption of the EM algorithm

- E-step

\[
Q(\Theta; \Theta(t)) = \sum_{i=1}^{N} \sum_{j=1}^{m} P(C_j | x_i; \Theta(t)) \ln(p(x_i | C_j; \theta)P_j)
\]

where

$\theta = [\theta_1^T, ..., \theta_m^T]^T$ ($\theta_j$ the parameter vector corresponding to $C_j$)

$P = [P_1, ..., P_m]^T$ ($P_j$ the a priori probability for $C_j$)

$\Theta = [\theta^T, P^T]^T$
• M-step

\[ \Theta(t+1) = \arg\max_\Theta Q(\Theta; \Theta(t)) \]

More specifically, the M-step results in:

For \( \theta_j \)'s:

\[ \sum_{i=1}^{N} \sum_{j=1}^{m} P(C_j | x_i; \Theta(t)) \frac{\partial}{\partial \theta_j} \ln p(x_i | C_j; \theta_j) = 0 \]

(\*) Provided that all pairs of \((\theta_k, \theta_j)\) are functionally independent.

For \( P_j \)'s:

\[ P_j = \frac{1}{N} \sum_{i=1}^{N} P(C_j | x_i; \Theta(t)) \]

(\*) Taking into account the constraints \( P_k \geq 0, k=1,\ldots,m \) and \( P_1 + P_2 + \ldots P_m = 1 \).

Thus, the EM algorithm for this case may be stated as follows:
Generalized Mixture Decomposition Algorithmic Scheme (GMDAS)

Choose initial estimates, $\theta = \theta(0)$ and $P = P(0)$.

$t = 0$

Repeat

- Compute

$$P(C_j \mid x_i; \Theta(t)) = \frac{p(x_i \mid C_j; \theta_j(t))P_j(t)}{\sum_{k=1}^{m} p(x_i \mid C_k; \theta_k(t))P_k(t)} \quad , \quad i = 1, \ldots, N, j = 1, \ldots, m$$  \hspace{1cm} (1)

- Set $\theta_j(t+1)$ equal to the solution of the equation

$$\sum_{i=1}^{N} \sum_{j=1}^{m} P(C_j \mid x_i; \Theta(t)) \frac{\partial}{\partial \theta_j} \ln p(x_i \mid C_j; \theta_j) = 0$$  \hspace{1cm} (2)

with respect to $\theta_j$, for $j = 1, \ldots, m$.

- Set

$$P_j(t+1) = \frac{1}{N} \sum_{i=1}^{N} P(C_j \mid x_i; \Theta(t)) \quad , \quad j = 1, \ldots, m$$  \hspace{1cm} (3)

- $t = t + 1$

Until convergence, with respect to $\Theta$, is achieved.
Remarks:

- A termination condition for GMDAS is
  \[ \| \Theta(t+1) - \Theta(t) \| < \varepsilon \]
  where \( \| . \| \) is an appropriate vector norm and \( \varepsilon \) a small user-defined constant.
- The above scheme is guaranteed to converge to a global or a local maximum of the log-likelihood function.
- Once the algorithm has converged, \( x_i \)'s are assigned to clusters according to the Bayes rule.
Compact and Hyperellipsoidal Clusters

In this case:
- Each cluster $C_j$ is modeled by a normal distribution $N(\mu_j, \Sigma_j)$.
- $\theta_j$ consists of the parameters of $\mu_j$ and the (independent) parameters of $\Sigma_j$.

It is

$$\ln p(x | C_j; \theta_j) = \ln \frac{|\Sigma_j|^{-1/2}}{(2\pi)^{d/2}} - \frac{1}{2} (x - \mu_j)^T \Sigma_j^{-1} (x - \mu_j), \quad j=1,\ldots,m$$

For this case:
- Eq. (1) in GMDAS is replaced by

$$P(C_j | x; \Theta(t)) = \frac{|\Sigma_j(t)|^{-1/2} \exp(-\frac{1}{2} (x - \mu_j(t))^T \Sigma_j^{-1}(t) (x - \mu_j(t))) P_j(t)}{\sum_{k=1}^{m} |\Sigma_k(t)|^{-1/2} \exp(-\frac{1}{2} (x - \mu_k(t))^T \Sigma_k^{-1}(t) (x - \mu_k(t))) P_k(t)}$$

- Eq. (2) in GMDAS is replaced by the equations

$$\mu_j(t+1) = \frac{\sum_{k=1}^{N} P(C_j | x_k; \Theta(t)) x_k}{\sum_{k=1}^{N} P(C_j | x_k; \Theta(t))} \quad \Sigma_j(t+1) = \frac{\sum_{k=1}^{N} P(C_j | x_k; \Theta(t))(x_k - \mu_j(t))(x_k - \mu_j(t))^T}{\sum_{k=1}^{N} P(C_j | x_k; \Theta(t))}$$
Remark:
• The above scheme is computationally very demanding since it requires the inversion of the $m$ covariance matrices at each iteration step. Two ways to deal with this problem are:
  • The use of a single covariance matrix for all clusters.
  • The use of different diagonal covariance matrices.

Example 1: (a) Consider three two-dimensional normal distributions with mean values:

$$\mu_1 = [1, 1]^T, \mu_2 = [3.5, 3.5]^T, \mu_3 = [6, 1]^T$$

and covariance matrices

$$\Sigma_1 = \begin{bmatrix} 1 & -0.3 \\ -0.3 & 1 \end{bmatrix}, \quad \Sigma_2 = \begin{bmatrix} 1 & 0.3 \\ 0.3 & 1 \end{bmatrix}, \quad \Sigma_3 = \begin{bmatrix} 1 & 0.7 \\ 0.7 & 1 \end{bmatrix},$$

respectively.

A group of 100 vectors stem from each distribution. These form the data set $X$. 
The data set

(b) Results of GMDAS

Confusion matrix:

<table>
<thead>
<tr>
<th></th>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st distribution</td>
<td>99</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2nd distribution</td>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>3rd distribution</td>
<td>3</td>
<td>4</td>
<td>93</td>
</tr>
</tbody>
</table>

The algorithm reveals accurately the underlying structure.
(b) The same as (a) but now $\mu_1 = [1, 1]^T$, $\mu_2 = [2, 2]^T$, $\mu_3 = [3, 1]^T$ (The clusters are closer).

The data set

Results of GMDAS

Confusion matrix:

<table>
<thead>
<tr>
<th></th>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>85</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>2nd</td>
<td>35</td>
<td>56</td>
<td>9</td>
</tr>
<tr>
<td>3rd</td>
<td>26</td>
<td>0</td>
<td>74</td>
</tr>
</tbody>
</table>

The algorithm reveals the underlying structure less accurately.
Fuzzy clustering algorithms

- Each vector belongs simultaneously to more than one clusters.

- A **fuzzy** $m$-clustering of $X$, is defined by a set of functions

  $$ u_j: X \rightarrow A = [0, 1], \ j=1,...,m. $$

  If $A=\{0,1\}$, a **hard** $m$-clustering of $X$ is produced.

- $u_j(x_i)$ denotes the **degree of membership** of $x_i$ in cluster $C_j$. It is

  $$ u_1(x_i) + u_2(x_i) + ... + u_m(x_i) = 1 $$

- The number of clusters $m$ is assumed to be known **a priori**.
Fuzzy clustering algorithms (cont)

Cost function definition

Let

- $\theta_j$ be the representative vector of $C_j$.
- $\Omega = [\theta_1^T, \ldots, \theta_m^T]^T$.
- $U = [u_{ij}] = [u_j(x_i)]$
- $d(x_i, \theta_j)$ be the dissimilarity between $x_i$ and $\theta_j$
- $q (>1)$ a parameter called fuzzifier.
Fuzzy clustering algorithms (cont)

Most fuzzy clustering schemes result from the minimization of:

$$J_q(\theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij}^q d(x_i, \theta_j)$$

Subject to the constraints:

$$\sum_{j=1}^{m} u_{ij} = 1, \quad i = 1, \ldots, N$$

where

$$u_{ij} \in [0, 1], \quad i = 1, \ldots, N, \quad j = 1, \ldots, m$$

and

$$0 < \sum_{i=1}^{N} u_{ij} < N, \quad j = 1, 2, \ldots, m$$
Remarks:

- The degree of membership of \( x_i \) in \( C_j \) cluster is related to the grade of membership of \( x_i \) in rest \( m-1 \) clusters.
- If \( q=1 \), no fuzzy clustering is better than the best hard clustering in terms of \( J_q(\Theta,U) \).
- If \( q>1 \), there are fuzzy clusterings with lower values of \( J_q(\Theta,U) \) than the best hard clustering.
Example 14.4. Let $X = \{x_1, x_2, x_3, x_4\}$, where $x_1 = [0, 0]^T$, $x_2 = [2, 0]^T$, $x_3 = [0, 3]^T$, $x_4 = [2, 3]^T$. Let $\theta_1 = [1, 0]^T$, $\theta_2 = [1, 3]^T$ be the cluster representatives. Suppose also that the Euclidean distance between a vector and a representative is in use. The hard two-cluster clustering that minimizes $J_q(\theta, U)$, for the above choice of $\theta_1, \theta_2$, can be represented by

$$U_{hard} = \begin{bmatrix}
1 & 0 \\
1 & 0 \\
0 & 1 \\
0 & 1
\end{bmatrix}$$

The value of $J_q(\theta, U)$ in this case Eq. (14.19) is $J_q^{hard}(\theta, U) = 4$. Obviously, hard clusterings do not depend on $q$.

Assume now that $q = 1$ and $u_{ij}$'s are between 0 and 1. Then the value of the cost function becomes

$$J_1^{fuzzy}(\theta, U) = \sum_{i=1}^{2}(u_{i1} + u_{i2}\sqrt{10}) + \sum_{i=3}^{4}(u_{i1}\sqrt{10} + u_{i2})$$

Since for each $x_i$ both $u_{i1}$ and $u_{i2}$ are positive and $u_{i1} + u_{i2} = 1$, it easily follows that $J_1^{fuzzy}(\theta, U) > 4$. Thus, the hard clustering always results in better values of $J_q^{fuzzy}(\theta, U)$, compared with their fuzzy counterparts, when $q = 1$.

Assume now that $q = 2$. The reader should easily verify that when $u_{i2} \in [0, 0.48]$ for $i = 1, 2$ and $u_{i1} \in [0, 0.48]$ for $i = 3, 4$, and, of course, $u_{i1} = 1 - u_{i2}$, for each $x_i$, then the value of $J_2^{fuzzy}(\theta, U)$ is less than 4 (see Problem 14.7). Thus, in this case fuzzy clusterings are favored over hard ones.
Fuzzy clustering algorithms (cont)

Minimizing \( J_q(\theta, U) \):

Minimization \( J_q(\theta, U) \) with respect to \( U \), subject to the constraints leads to the following Lagrangian function,

\[
J_{Lan}(\theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij}^q d(x_i, \theta_j) - \sum_{i=1}^{N} \lambda_i \left( \sum_{j=1}^{m} u_{ij} - 1 \right)
\]

Minimizing \( J_{Lan}(\theta, U) \) with respect to \( u_{rs} \), we obtain

\[
u_{rs} = \frac{1}{\sum_{j=1}^{m} \left( \frac{d(x_r, \theta_s)}{d(x_r, \theta_j)} \right)^{q-1}}, \quad r = 1, \ldots, N, \quad s = 1, \ldots, m.
\]

Setting the gradient of \( J(\theta, U) \), with respect to \( \theta \), equal to zero we obtain,

\[
\frac{\partial J(\theta, U)}{\partial \theta_j} = \sum_{i=1}^{N} u_{ij}^q \frac{\partial d(x_i, \theta_j)}{\partial \theta_j} = 0, \quad j = 1, \ldots, m.
\]

The last two equations are coupled. Thus, no closed form solutions are expected. Therefore, minimization is carried out iteratively.
Generalized Fuzzy Algorithmic Scheme (GFAS)

- Choose $\theta_j(0)$ as initial estimate for $\theta_j$, $j=1,\ldots,m$.
- $t=0$
- Repeat
  - For $i=1$ to $N$
    - For $j=1$ to $m$
      - $u_{ij}(t) = \frac{1}{\sum_{j=1}^{m} \left( \frac{d(x_i, \theta_j(t))}{d(x_i, \theta_k(t))} \right)^{\frac{1}{q-1}}}^{(A)}$
    - End {For-j}
  - End {For-i}
  - $t=t+1$
  - For $j=1$ to $m$
    - Parameter updating: Solve
      $$\sum_{i=1}^{N} u_{ij}^q(t-1) \frac{\partial d(x_i, \theta_j)}{\partial \theta_j} = 0.$$ (B)
      with respect to $\theta_j$ and set $\theta_j(t)$ equal to this solution.
    - End {For-j}
- Until a termination criterion is met
Remarks:

• A candidate termination condition is

\[ \| \theta(t) - \theta(t-1) \| < \epsilon, \]

where \( \| . \| \) is any vector norm and \( \epsilon \) a user-defined constant.

• GFAS may also be initialized from \( U(0) \) instead of \( \theta_j(0), j=1,\ldots,m \) and start iterations with computing \( \theta_j \) first.

• If a point \( x_i \) coincides with one or more representatives, then it is shared arbitrarily among the clusters whose representatives coincide with \( x_i \), subject to the constraint that the summation of the degree of membership over all clusters sums to 1.
Fuzzy Clustering – Point Representatives

- Point representatives are used in the case of compact clusters
- Each $\theta_j$ consists of $l$ parameters
- Every dissimilarity measure $d(x_i, \theta_j)$ between two points can be used
- Common choices for $d(x_i, \theta_j)$ are
  - $d(x_i, \theta_j) = (x_i - \theta_j)^T A (x_i - \theta_j)$, where $A$ is symmetric and positive definite matrix.

In this case:
\[
\frac{\partial d(x_i, \theta_j)}{\partial \theta_j} = 2A(\theta_j - x_i).
\]

Thus the updating equation (B) in GFAS becomes
\[
\theta_j(t) = \sum_{i=1}^N u_{ij}^q(t-1)x_i / \sum_{i=1}^N u_{ij}^q(t-1)
\]

- GFAS with the above distance is also known as Fuzzy c-Means (FCM) or Fuzzy k-Means algorithm.
- FCM converges to a stationary point of the cost function or it has at least one subsequence that converges to a stationary point. This point may be a local (or global) minimum or a saddle point.
Fuzzy clustering – Point representatives (cont.)

- The *Minkowski* distance

\[
d(x_i, \theta_j) = \left( \sum_{k=1}^{l} |x_{ik} - \theta_{jk}|^p \right)^{1/p}
\]

where \( p \) is a positive integer and \( x_{ik}, \theta_{jk} \) are the \( k \)-th coordinates of \( x_i \) and \( \theta_j \).

For even and finite \( p \), the differentiability of \( d(x_i, \theta_j) \) is guaranteed. In this case the updating equation (B) of GFAS gives

\[
\sum_{i=1}^{N} u_{ij}^q (t-1) \frac{(\theta_{jr} - x_{ir})^{p-1}}{\left(\sum_{k=1}^{l'} |x_{ik} - \theta_{jk}|^p \right)^{1-p}} = 0, \quad r = 1,...,l
\]

a system of \( l \) nonlinear equations with \( l \) unknowns.

GFAS algorithms with the Minkowski distance are also known as \( pFCM \) algorithms.
Fuzzy Clustering – Point representatives (cont.)

Example 2(a):
- Consider the setup of example 1(a).
- Consider GFAS with distances
  (i) \( d(x_i, \theta_j) = (x_i - \theta_j)^T A (x_i - \theta_j) \), with \( A \) being the identity matrix.
  (ii) \( d(x_i, \theta_j) = (x_i - \theta_j)^T A (x_i - \theta_j) \), with \( A = \begin{bmatrix} 2 & 1.5 \\ 1.5 & 2 \end{bmatrix} \).
  (iii) The Minkowski distance with \( p=4 \).

Example 2(b):
- Consider the setup of example 1(b).
- Consider GFAS with the distances considered in example 2(a).
The corresponding confusion matrices for example 2(a) and 2(b) are (Here a vector is assigned to the cluster for which $u_{ij}$ has the maximum value.)

For the example 2(a)

\[
A_i = \begin{bmatrix}
98 & 2 & 0 \\
14 & 84 & 2 \\
11 & 0 & 89
\end{bmatrix}
\]
\[
A_{ii} = \begin{bmatrix}
63 & 11 & 26 \\
5 & 95 & 0 \\
39 & 23 & 38
\end{bmatrix}
\]
\[
A_{iii} = \begin{bmatrix}
96 & 0 & 4 \\
11 & 89 & 0 \\
13 & 2 & 85
\end{bmatrix}
\]

For the example 2(b)

\[
A_i' = \begin{bmatrix}
51 & 46 & 3 \\
14 & 47 & 39 \\
43 & 0 & 57
\end{bmatrix}
\]
\[
A_{ii}' = \begin{bmatrix}
79 & 21 & 0 \\
19 & 58 & 23 \\
28 & 41 & 31
\end{bmatrix}
\]
\[
A_{iii}' = \begin{bmatrix}
51 & 3 & 46 \\
37 & 62 & 1 \\
11 & 36 & 53
\end{bmatrix}
\]

Remarks:

- In $A_i$ and $A_{iii}$ (example 2(a)) almost all vectors from the same distribution are assigned to the same cluster.
- The closer the clusters are, the worse the performance of all the algorithms.
- The choice of matrix $A$ in $d(x_i, \theta_j) = (x_i - \theta_j)^{T}A(x_i - \theta_j)$ plays an important role to the performance of the algorithm.