

# Data Clustering

**P. Papageorgiou, Prof. Ioannis Pitas**  
**Aristotle University of Thessaloniki**  
**[pitass@csd.auth.gr](mailto:pitass@csd.auth.gr)**  
**[www.aiia.csd.auth.gr](http://www.aiia.csd.auth.gr)**  
**Version 3.5.1**

# Outline

- Clustering Definitions
- Distance/similarity measures
- Clustering categories
  - Exhaustive Clustering
  - Sequential Clustering
  - Clustering by optimization
  - Vector quantization
  - Graph-based clustering.

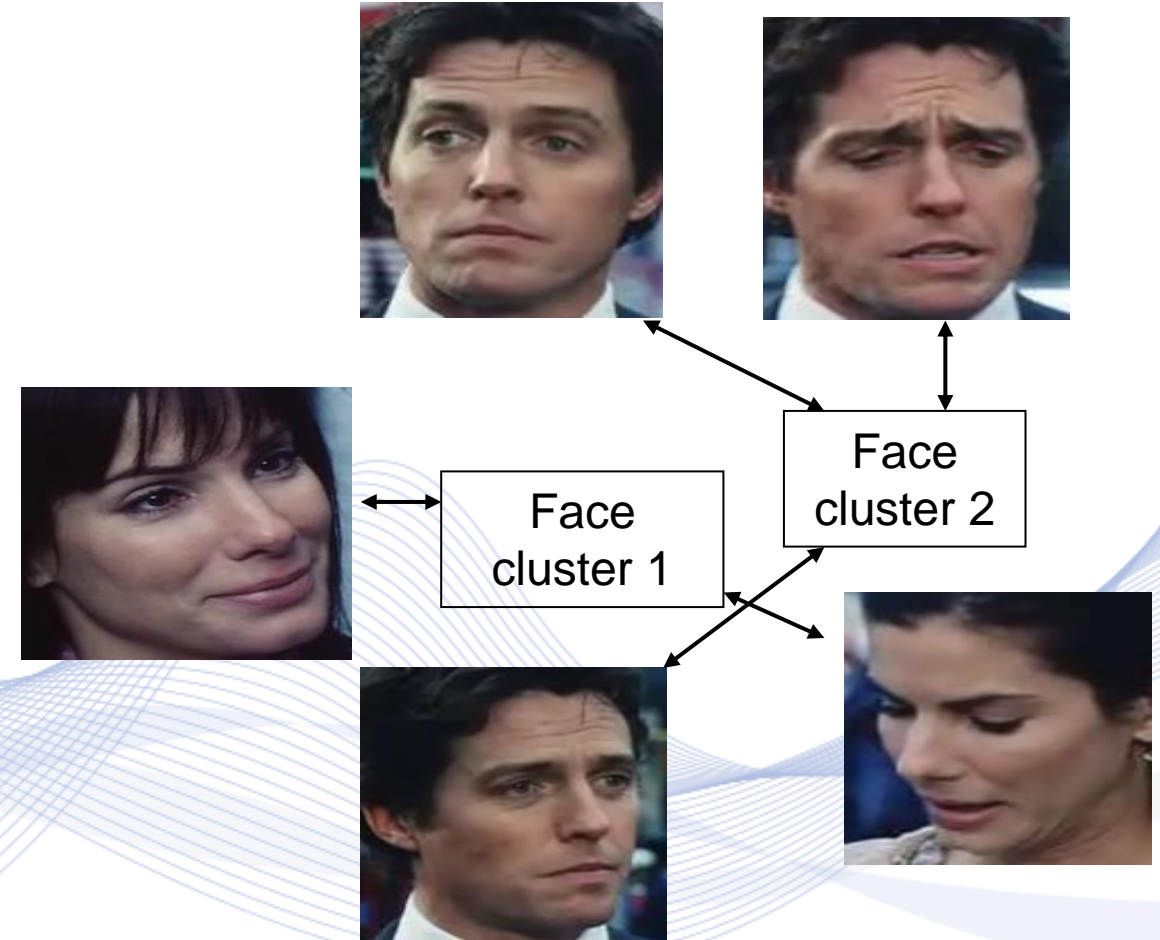
# Introduction

- ***Data clustering***: special case of ***unsupervised learning***.
  - class labelling of the training patterns is not available.
- Goal: to reveal the ***geometrical data organization*** into ***sensible clusters***, in order to:
  - discover data (dis)similarities.
  - discover geometrical cluster structure.
- Applications:
  - life sciences, earth sciences and engineering.

# Face clustering

## ***Problem statement:***

- To cluster facial images
  - ***Input:*** many facial ROIs
  - ***Output:*** facial image clusters.
- 
- Unsupervised learning
  - Applications:
    - Biometrics
    - Surveillance applications
    - Video analytics.

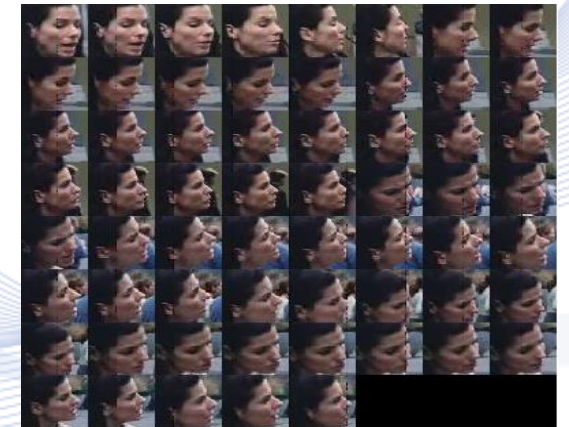




# Face Clustering

## *Problem statement.*

- To cluster a set of facial ROIs
- **Input:** a set of face image ROIs
- **Output:** several face clusters, each containing faces of only one person.
- Applications
  - Cluster actor images, even if they belong to different shots.
  - Cluster various views of the same actor.
  - Generate the cast of a movie.
  - Semi automatic face recognition.



# Introduction

- Clustering criterion in image data:
  - ***Color similarity***: e.g., all facial image regions are pink.
  - ***Texture similarity***: tree foliage regions have fine unstructured visual texture.
  - ***Edge similarity***: building images have vertical/horizontal edges.
  - ***Intensity similarity***: black people have dark facial images.

# Introduction

- **Outliers:** black albinos have brighter facial images.
- Clusters may consist of **sub-clusters**:
  - Caucasian and black facial images belong to 'facial image' cluster.



# Introduction

- Clustering criteria greatly influence clustering results.
- Clustering is a key human cognitive ability:
  - Clusters are characterized by the common data attributes.
  - Cluster labeling leads to logical ***concepts***.



**Data clustering input:** data samples described by feature vectors, without neither labels nor any information about the specific desired output:

$$\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N.$$

Typically:  $\mathbf{x} \in \mathbb{R}^n$ .

**Data clustering output:**

- Sample data set  $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$  partition to clusters  $\mathcal{C}_i, i = 1, \dots, m$ .
- Cluster samples are similar and dissimilar to the samples of other clusters based on similarity/distance metric  $\|\cdot\|$ .
- Number of clusters  $m$  may be unknown.

# Feature types

- ***Real-valued feature vectors:***  $\mathbf{x} \in \mathbb{R}^n$ .
- ***Finite discrete set feature vectors:***  $\mathbf{x} \in \mathcal{F}^n$ .
- Discrete set cardinality  $k$ :  $\mathcal{F} = \{0, 1, \dots, k - 1\}$ .
- Special case:
  - binary set  $k = 2$ ,  $\mathcal{F} = \{0, 1\}$ .

# Feature types

- **Labeled (nominal) features:**  $\mathcal{F} = \{f_0, f_1, \dots, f_{k-1}\}$ .
- Feature values  $f_i, i = 0, \dots, k - 1$  may have symbolic meaning (symbolic labels):
  - Facial image labels  $\mathcal{F} = \{\text{'John'}, \text{'Alice'}, \dots, \text{'Vladimir'}\}$ .
- Nominal feature vectors:  $\mathbf{x} \in \mathcal{F}_1 \times \mathcal{F}_2 \times \dots \times \mathcal{F}_n$ .
- Feature vector  $\mathbf{x} \in \mathcal{F}_1 \times \mathcal{F}_2$  for describing apples:
  - $\mathcal{F}_1 = \{\text{'small'}, \text{'medium'}, \text{'big'}\}$ ,
  - $\mathcal{F}_2 = \{\text{'red'}, \text{'yellow'}, \text{'green'}\}$ .

# Feature types

- Feature categorization:
  - **Nominal features:**  $\mathcal{F}$  is a set.
    - No feature value ordering is possible.
  - **Ordinal features:** e.g.,  $\mathcal{F} = \mathbb{R}$ .
    - Feature values can be meaningfully ordered.
  - **Angular features:**  $\mathcal{F} = [0, 2\pi]$ .
    - Feature values are angles (or on a unit circle).
  - Interval-scaled: feature value difference is meaningful.
  - Ratio-scaled: feature value ratio is meaningful.



# Introduction

Clustering subtasks:

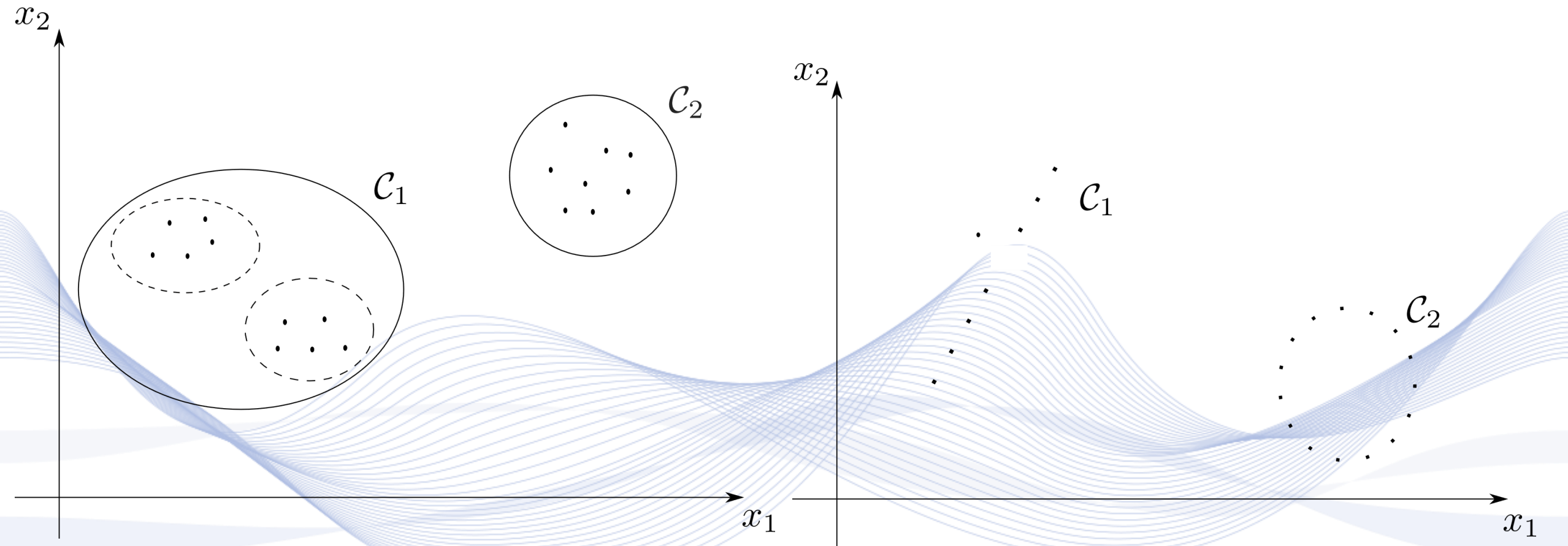
- ***Feature selection***: Create a feature vector  $\mathbf{x}$  with minimum information redundancy.
- ***Data similarity measurement***: Quantify feature vector '(dis)similarity'.
- ***Clustering criterion***: It quantifies clustering 'sensibility'.
- ***Cost function optimization***:
  - Maximize intra-class similarity and maximize inter-class dissimilarity.

# Introduction

Clustering subtasks:

- ***Choosing a clustering algorithm:*** to best unravel data structure.
- ***Clustering validation:*** verify the correctness of clustering results using appropriate tests.
- ***Clustering explainability:*** Interpretation clustering results.
- ***Concept creation:*** Cluster labeling to create logical 'concepts'.

# Introduction



Clustering granularity.

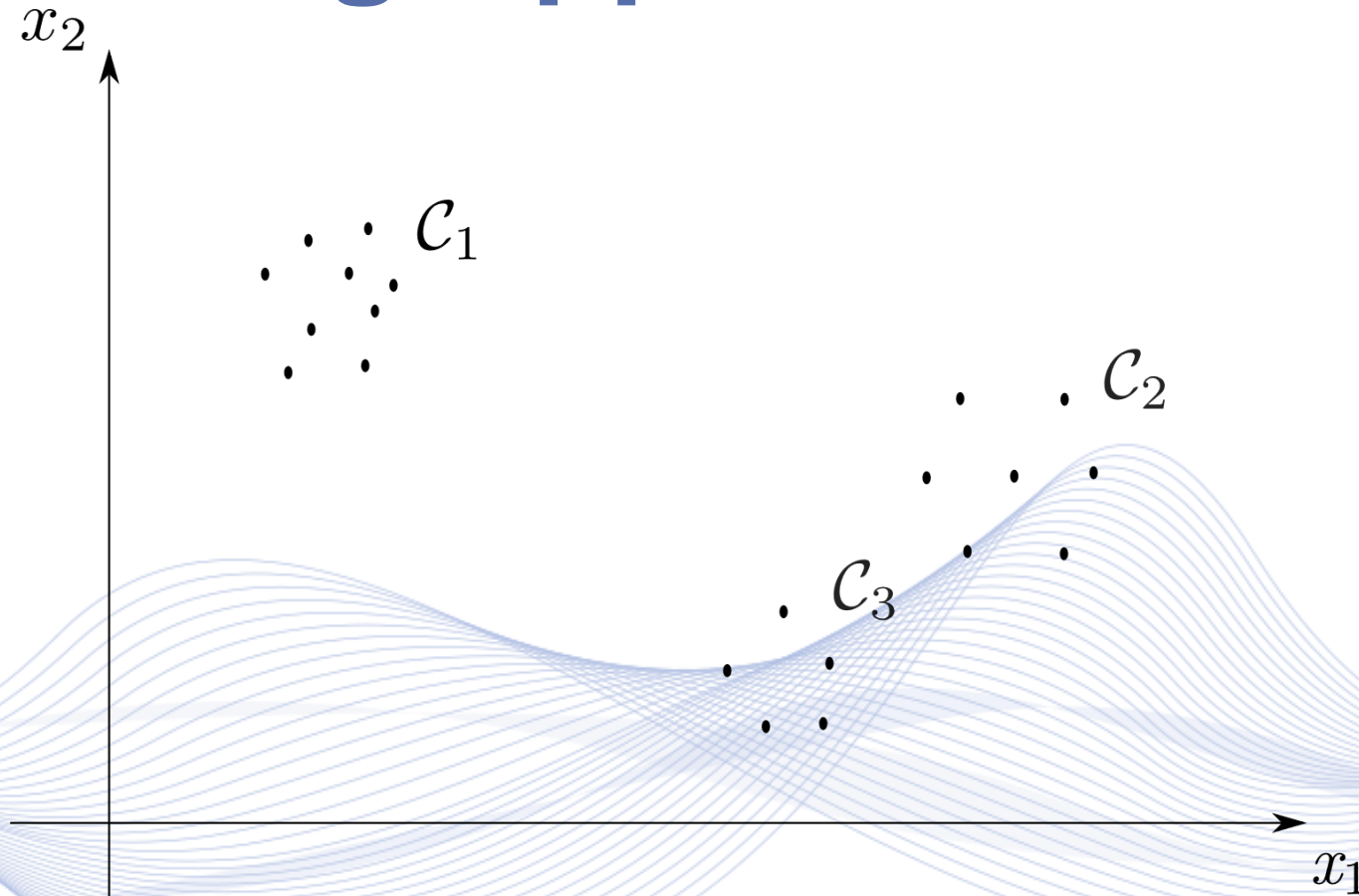
Cluster manifolds.

# Clustering Applications

- ***Better data description:***
  - Data set  $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$ ,  $\mathbf{x} \in \mathbb{R}^n$  cardinality  $N$  or data dimensionality  $n$  may be too large;
  - Clustering groups the data into  $m \ll N$  clusters, providing much better data description.
- ***Data cluster visualization:*** clusters are well visualized if they are mapped on  $\mathbb{R}^2$ .
- ***Hypothesis generation:*** Help forming and validating hypotheses on data structure.



# Clustering Applications



Data cluster visualization.

# Crisp/fuzzy Clustering

- Let  $\mathcal{D}$  be a feature data set:  $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ .
- **Crisp clustering** is the partition of  $\mathcal{D}$  into  $m$  disjoint sets  $\mathcal{C}_1, \dots, \mathcal{C}_m$ , satisfying the following conditions:
  - $\mathcal{C}_i \neq \emptyset, \quad i = 1, \dots, m,$
  - $\cup_{i=1}^m \mathcal{C}_i = \mathcal{D},$
  - $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset, \quad i \neq j, \quad i, j = 1, \dots, m.$
- Feature vectors in a cluster  $\mathcal{C}_i$  are ‘similar’, while they are ‘dissimilar’ to the ones of other clusters  $\mathcal{C}_j, i \neq j$ .

# Crisp/fuzzy Clustering

- **Fuzzy clustering** of  $\mathcal{D}$  into  $m$  clusters:
  - For each sample  $\mathbf{x}_i$ ,  $i = 1, \dots, N$  find  $m$  **membership functions**  $u_j$ :

$$u_j: \mathcal{D} \rightarrow [0,1], \quad j = 1, \dots, m.$$

$$\sum_{j=1}^m u_j(\mathbf{x}_i) = 1, \quad i = 1, \dots, N,$$

$$0 < \sum_{j=1}^m u_j(\mathbf{x}_i) < N, \quad j = 1, \dots, m.$$

# Crisp/fuzzy Clustering

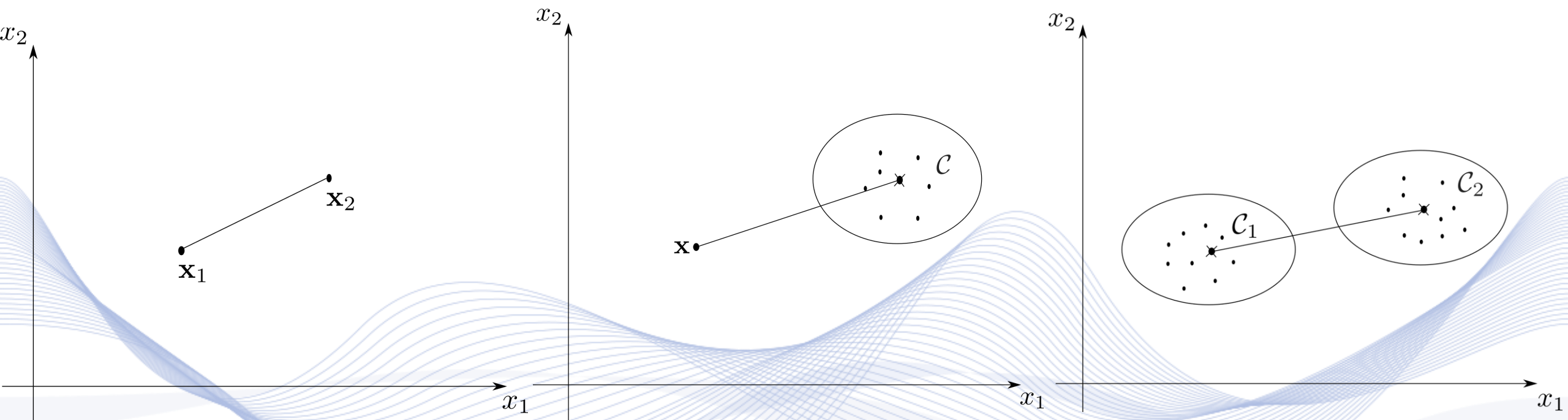
- Each vector  $\mathbf{x}$  belongs to more than one clusters simultaneously, which depends on the value of  $u_j$  in  $[0,1]$ .
- Fuzzy membership values  $u_j \rightarrow 1$ : high cluster membership possibility.
- Fuzzy membership values  $u_j \rightarrow 0$ : low cluster membership possibility.
- Feature vector similarity: the membership function vector difference  $|\mathbf{u}_k - \mathbf{u}_n|$  for two feature vectors  $\mathbf{x}_k, \mathbf{x}_n$  is small.



# Distance/Similarity Measures

- Proximity or distance measures
  - Similarity or dissimilarity measures.
- 
- Distance between two feature points (feature vectors).
  - Distance between a feature point and a feature point set.
  - Distance between two feature point sets.

# Distance Measures



Distance between two points.

Distance between point and set (set center).

Distance between sets (set center).

# Distance Measures

## ***Dissimilarity measure (DM) $d$ :***

- a function  $d: \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}$ , satisfying:
  - $\exists d_0 \in \mathbb{R}: -\infty < d_0 \leq d(\mathbf{x}, \mathbf{y}) < +\infty, \forall \mathbf{x}, \mathbf{y} \in \mathcal{F},$
  - $d(\mathbf{x}, \mathbf{x}) = d_0, \forall \mathbf{x} \in \mathcal{F},$
  - $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x}), \forall \mathbf{x}, \mathbf{y} \in \mathcal{F}.$

Typically:  $d(\mathbf{x}, \mathbf{x}) = 0$ .

- Also called ***distance measure*** for Euclidean spaces:  
 $\mathcal{F} = \mathbb{R}^n$ .

# Distance Measures

If:

- $d(\mathbf{x}, \mathbf{y}) = d_0$ , if and only if  $\mathbf{x} = \mathbf{y}$
- and **triangular inequality** holds:

$$d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}), \quad \forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{F}.$$

$d$  is a ***metric*** or ***norm***.



# Distance Measures

Weighted  $L_p$  metric between two real-valued feature points (vectors)  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ :

$$d(\mathbf{x}, \mathbf{y}) = \left( \sum_{i=1}^n w_i |x_i - y_i|^p \right)^{1/p}.$$

- $w_i \geq 0$ : is weight coefficient.
- Unweighted  $L_p$  metric:  $w_i = 1, \quad i = 1, \dots, n$ .
- $L_2$  metric:

$$d(\mathbf{x}, \mathbf{y}) = \left( \sum_{i=1}^n (x_i - y_i)^2 \right)^{1/2}.$$

# Distance Measures

- ***Mahalanobis distance:***

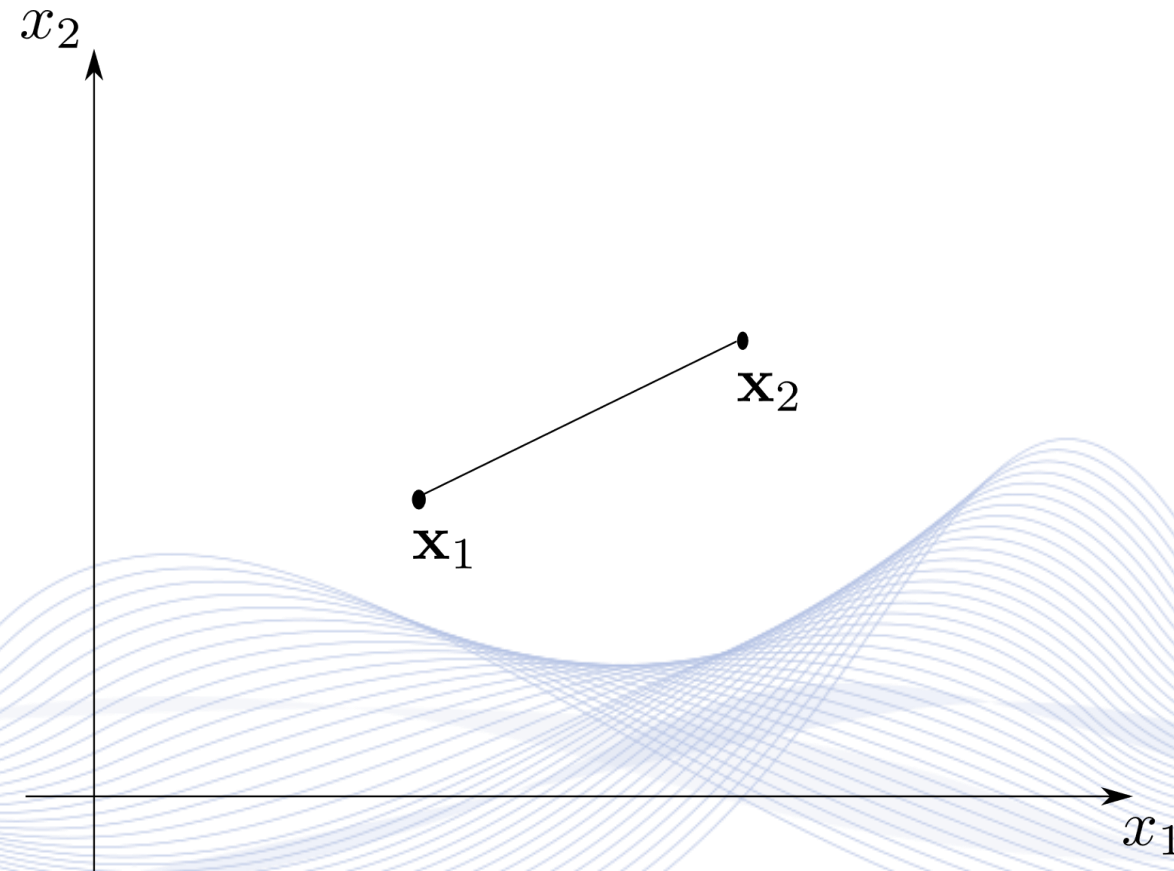
$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \mathbf{A} (\mathbf{x} - \mathbf{y})}.$$

- $\mathbf{A}$  is a  $n \times n$  symmetric, positive-definite matrix.
- ***Euclidean distance:***

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y})}.$$

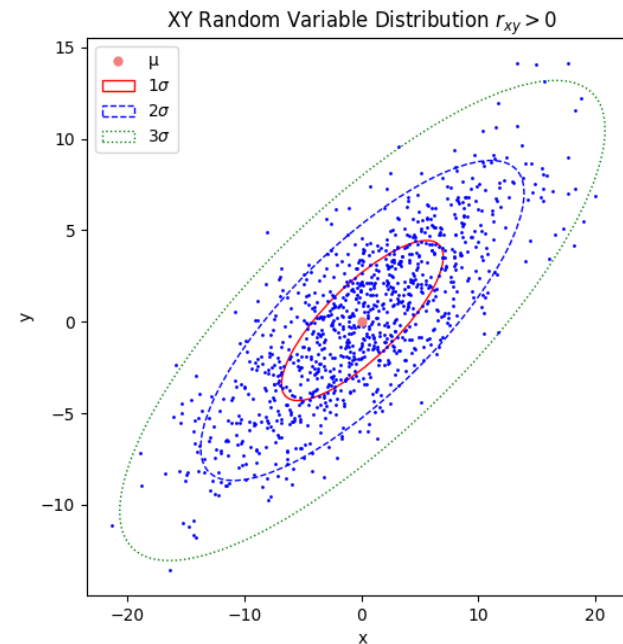
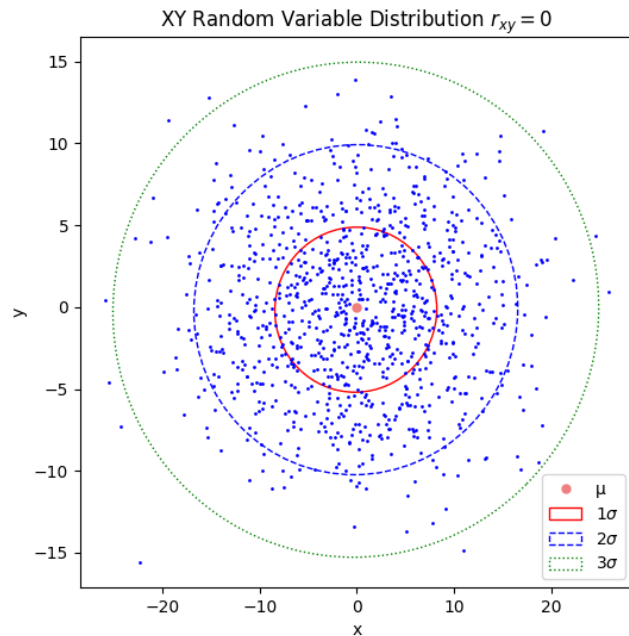
- $\mathbf{A} = \mathbf{I}$ .
- It is equal to the length of the straight line segment connecting points  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ .

# Distance Measures



Euclidean distance between two points.

# Distance Measures



- a) Euclidean equidistant points (circles);
- b) Mahalanobis equidistant points (ellipses).



# Distance Measures

- Weighted  $L_1$  **norm**:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n w_i |x_i - y_i|.$$

- **Manhattan norm**:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n |x_i - y_i|.$$

- $L_\infty$  **norm**:

$$d(\mathbf{x}, \mathbf{y}) = \max_{1 \leq i \leq n} |x_i - y_i|.$$

- $L_\infty$  norm is a special case of  $L_p$  norm for  $p \rightarrow \infty$ .

# Distance Measures

## *Discrete-Valued Vectors*

- Vectors  $\mathbf{x}$  with coordinates belonging to the finite set  $\mathcal{F} = \{0, 1, \dots, k - 1\}$ , ( $k$  is a positive integer).
- There are exactly  $k^n$  vectors  $\mathbf{x}, \mathbf{y} \in \mathcal{F}^n$ .

- $k \times k$  **contingency table**:

$$\mathbf{A}(\mathbf{x}, \mathbf{y}) = [a_{ij}] \quad i, j = 0, 1, \dots, k - 1.$$

- $a_{ij}$  is the number of vectors  $\mathbf{x}, \mathbf{y}$  entries having values  $i, j \in \mathcal{F}$  symbols, respectively.

# Distance Measures

## ***Discrete-Valued Vectors***

- ***Edit distance.***
- ***Hamming distance:***

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{k-1} \sum_{j=0, j \neq i}^{k-1} a_{ij}.$$

- Equal to the summation of all the off-diagonal elements of  $\mathbf{A}$ , indicating the entries, where  $\mathbf{x}$  and  $\mathbf{y}$  differ.

# Distance Measures

- For  $k = 2$  the Hamming distance is:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n (x_i - y_i)^2.$$

- **$L_1$  distance:**

$$d_1(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n |x_i - y_i|.$$



# Similarity Measures

- **Similarity measure (SM):**
- a function  $s: \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}$  such that:

$$\exists s_0 \in \mathbb{R}: -\infty < s(\mathbf{x}, \mathbf{y}) \leq s_0 < +\infty, \forall \mathbf{x}, \mathbf{y} \in \mathcal{F},$$

$$\begin{aligned} s(\mathbf{x}, \mathbf{x}) &= s_0, \forall \mathbf{x} \in \mathcal{F}, \\ s(\mathbf{x}, \mathbf{y}) &= s(\mathbf{y}, \mathbf{x}), \forall \mathbf{x}, \mathbf{y} \in \mathcal{F}. \end{aligned}$$

- Typically,  $s(\mathbf{x}, \mathbf{x}) = 1$ .

# Similarity Measures

If:

- $s(\mathbf{x}, \mathbf{y}) = s_0$ , if and only if  $\mathbf{x} = \mathbf{y}$
- and:

$$s(\mathbf{x}, \mathbf{y})s(\mathbf{y}, \mathbf{z}) \leq [s(\mathbf{x}, \mathbf{y}) + s(\mathbf{y}, \mathbf{z})]s(\mathbf{x}, \mathbf{z}), \quad \forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{F}$$

then  $s$  is a ***metric SM***.

# Similarity Measures

Similarity measures between two feature vectors  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ :

- ***Inner vector product***:

$$s(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^n x_i y_i .$$

- If the vectors  $\mathbf{x}, \mathbf{y}$  are normalized to length  $a$ :  $s \in [-a^2, a^2]$ .
- ***Cosine similarity*** measure:

$$s(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x}|| ||\mathbf{y}||}, \quad ||\mathbf{x}|| = \sqrt{\sum_{i=1}^n x_i^2}, \quad ||\mathbf{y}|| = \sqrt{\sum_{i=1}^n y_i^2}.$$

# Similarity Measures

- **Correlation coefficient:**

$$r(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}_c^T \mathbf{y}_c}{\|\mathbf{x}_c\| \|\mathbf{y}_c\|},$$

- $r(\mathbf{x}, \mathbf{y}) \in [-1, 1]$ .
- Central difference vectors:

$$\mathbf{x}_c = [x_1 - \bar{x}, \dots, x_n - \bar{x}]^T, \quad \mathbf{y}_c = [y_1 - \bar{y}, \dots, y_l - \bar{y}]^T,$$

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i, \quad \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i.$$



# Similarity Measures

***Fuzzy Similarity Measure*** between two vectors:

$$s(\mathbf{x}, \mathbf{y}) = \left( \sum_{i=1}^n s(x_i, y_i) \right)^{\frac{1}{q}}.$$

- Similarity between two real-valued variables  $x_i$  and  $y_i$ :

$$s(x_i, y_i) = \max(\min(1 - x_i, 1 - y_i), \min(x_i, y_i)).$$

# Similarity Measures

- Motivation:
  - Equivalence between two logic variables  $a$  and  $b$ :
$$(a \equiv b) = (\bar{a} \text{ AND } \bar{b}) \text{ OR } (a \text{ AND } b).$$
  - Fuzzy AND, OR,  $\bar{a}$  (NOT) operators:  $\max, \min, 1 - a$ .

# Similarity Measures

**Tanimoto similarity measure** between two feature vectors  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ :

$$s(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x} - \mathbf{y}||^2} = \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x}||^2 + ||\mathbf{y}||^2 - \mathbf{x}^T \mathbf{y}}.$$

- It is inversely proportional to the squared Euclidean distance divided by their inner product.
- Similarity measure:

$$s(\mathbf{x}, \mathbf{y}) = 1 - \frac{d_2(\mathbf{x}, \mathbf{y})}{||\mathbf{x}|| + ||\mathbf{y}||}.$$

- Maximum when  $\mathbf{x} = \mathbf{y}$  and minimum when  $\mathbf{x} = -\mathbf{y}$ .

# Similarity Measures

Tanimoto measure for discrete-valued vectors:

$$s(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{k-1} a_{ii}}{n_x + n_y - \sum_{i=1}^{k-1} \sum_{j=1}^{k-1} a_{ij}},$$

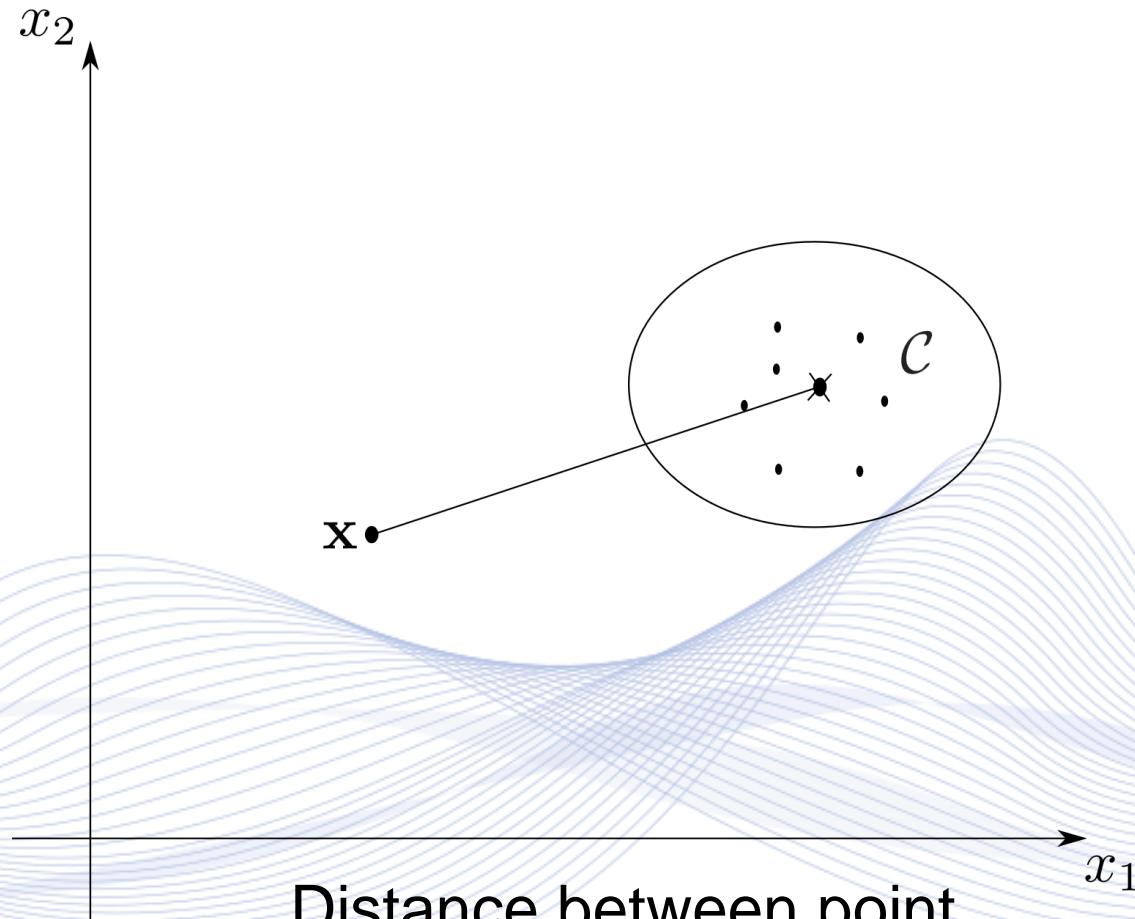
$$n_x = \sum_{i=1}^{k-1} \sum_{j=0}^{k-1} a_{ij}, \quad n_y = \sum_{i=0}^{k-1} \sum_{j=1}^{k-1} a_{ij}.$$

- It takes into account all pairs of corresponding  $\mathbf{x}$  and  $\mathbf{y}$  coordinates, except when both  $x_i = 0, y_i = 0$ .
- Motivation: set ***Intersection over Union (IoU)*** for sets  $\mathcal{X}, \mathcal{Y}$ :

$$s = \frac{|\mathcal{X} \cap \mathcal{Y}|}{|\mathcal{X} \cup \mathcal{Y}|}.$$



# Distance Measures



Distance between point  
and set (set center).

# Distance Measures

***Distance functions between a point and a set (cluster).***

- Distance  $d'(\mathbf{x}, \mathcal{C})$  between vector  $\mathbf{x}$  and cluster  $\mathcal{C}$  :
  - ***Distance to cluster center  $\mathbf{m}$*** :  $d'(\mathbf{x}, \mathcal{C}) = d(\mathbf{x}, \mathbf{m})$ .
  - Max Distance function:  $d'(\mathbf{x}, \mathcal{C}) = \max_{\mathbf{y} \in \mathcal{C}} d(\mathbf{x}, \mathbf{y})$ .
  - Min Distance function:  $d'(\mathbf{x}, \mathcal{C}) = \min_{\mathbf{y} \in \mathcal{C}} d(\mathbf{x}, \mathbf{y})$ .
  - Average Distance function:  $d'(\mathbf{x}, \mathcal{C}) = \frac{1}{|\mathcal{C}|} \sum_{\mathbf{y} \in \mathcal{C}} d(\mathbf{x}, \mathbf{y})$ 
    - $|\mathcal{C}|$  : set  $\mathcal{C}$  cardinality.

# Distance Measures

**Cluster center** is a representative vector of a data vector set:

- **Mean vector:**

$$\mathbf{m} = \frac{1}{|C|} \sum_{\mathbf{x} \in C} \mathbf{x}.$$

- Sensitive to outliers.

# Distance Measures

- **Vector median:**

$$\sum_{\mathbf{y} \in \mathcal{C}} d(\mathbf{m}_v, \mathbf{y}) \leq \sum_{\mathbf{y} \in \mathcal{C}} d(\mathbf{z}, \mathbf{y}), \mathbf{m}_v \in \mathcal{C}, \forall \mathbf{z} \in \mathcal{C}.$$

- **Median center:**

$$\text{med}(d(\mathbf{m}_m, \mathbf{y}) | \mathbf{y} \in \mathcal{C}) \leq \text{med}(d(\mathbf{z}, \mathbf{y}) | \mathbf{y} \in \mathcal{C}), \mathbf{m}_m \in \mathcal{C}, \forall \mathbf{z} \in \mathcal{C}.$$

- med: **median** operator.



# Distance Measures

## *Data manifold representations:*

- Hyperplane  $\mathbb{H}$ :

$$\sum_{j=1}^n a_j x_j + a_0 = \mathbf{a}^T \mathbf{x} + a_0 = 0, \quad \mathbf{x} = [x_1, \dots, x_n]^T.$$

- Hyperplane parameters  $a_0$ ,  $\mathbf{a} = [a_1, \dots, a_n]^T$ .
- Distance of a point  $\mathbf{x}$  from hyperplane  $\mathbb{H}$ :

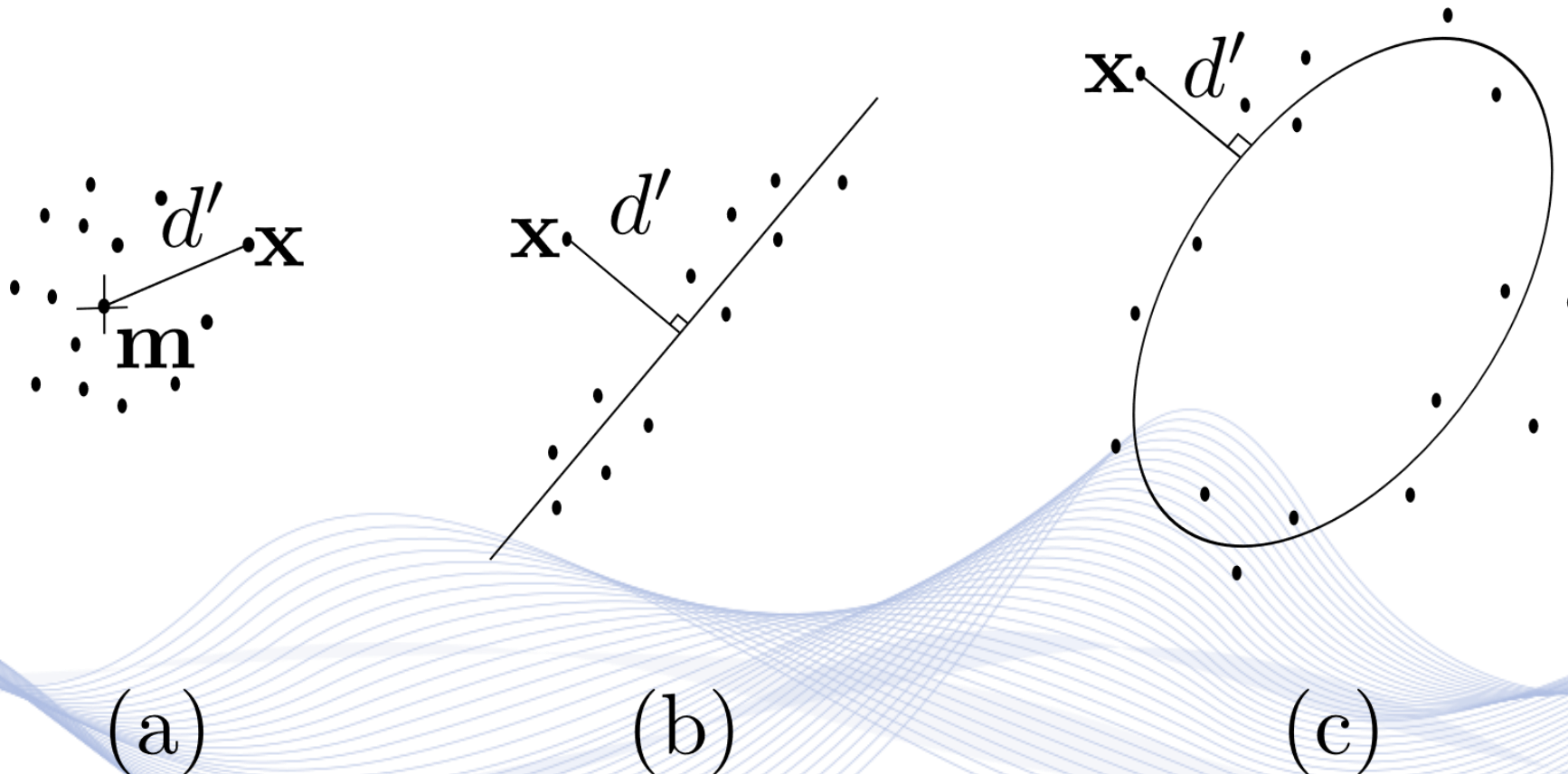
$$d'(\mathbf{x}, \mathbb{H}) = \frac{|\mathbf{a}^T \mathbf{x} + a_0|}{\|\mathbf{a}\|}.$$

# Distance Measures

- **Quadratic surface**  $\mathcal{S}$  representations:
  - **Hypersphere** equation:  $(\mathbf{x} - \mathbf{c})^T (\mathbf{x} - \mathbf{c}) = r^2$ .
  - $\mathbf{c}, r$ : hypersphere center, radius.
  - **Hyperellipsoid** equation having parameters  $\mathbf{A}, \mathbf{c}, r$ :
$$(\mathbf{x} - \mathbf{c})^T \mathbf{A} (\mathbf{x} - \mathbf{c}) = r^2.$$
- Distance of a point  $\mathbf{x}$  from  $\mathcal{S}$ :

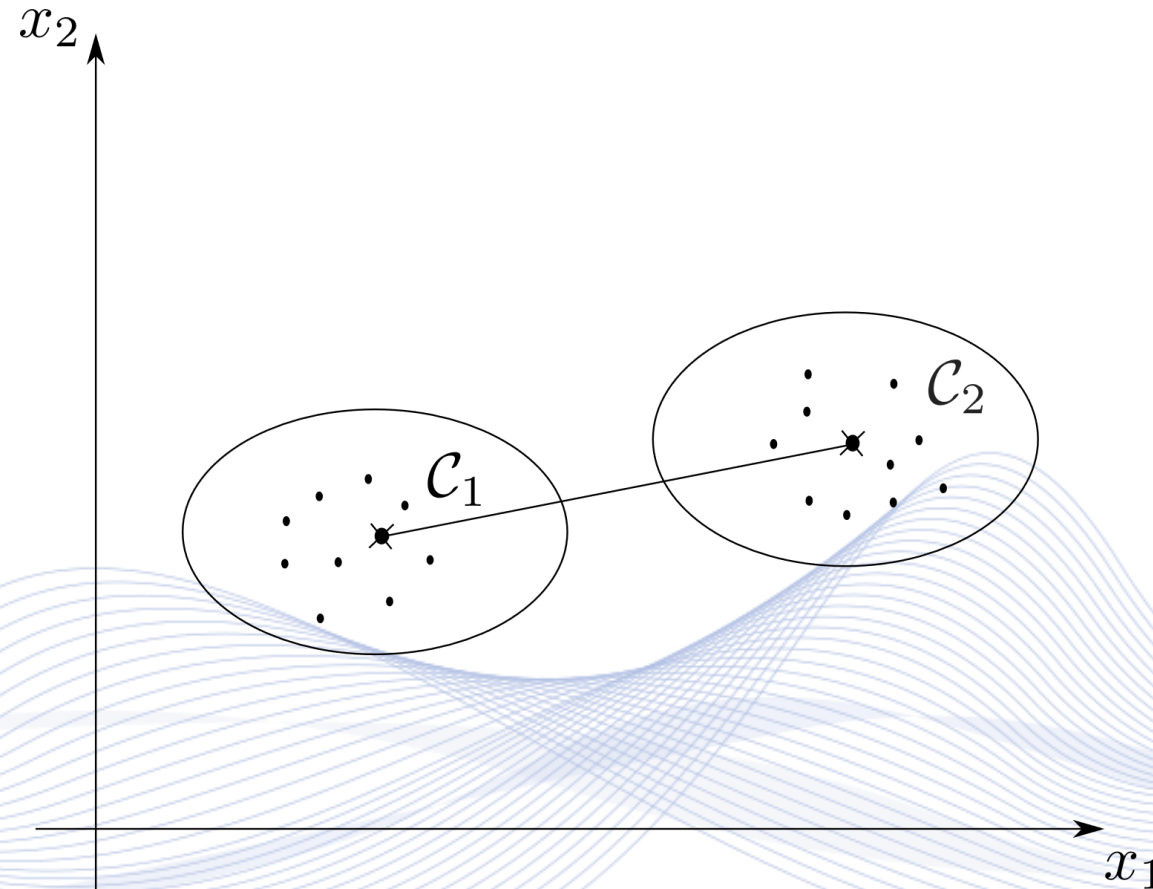
$$d'(\mathbf{x}, \mathcal{S}) = \min_{\mathbf{z} \in \mathcal{S}} d(\mathbf{x}, \mathbf{z}).$$

# Distance Measures



a) Compact cluster; b) Linear cluster; c) Ellipsoid cluster representations.

# Distance Measures



Distance between set centers.



# Distance Measures

## *Distance Functions between Two Sets*

- If  $\mathcal{C}_i, \mathcal{C}_j$  are two sets of vectors the most common proximity functions are:
  - Max distance function:  $d''(\mathcal{C}_i, \mathcal{C}_j) = \max_{\mathbf{x} \in \mathcal{C}_i, \mathbf{y} \in \mathcal{C}_j} d(\mathbf{x}, \mathbf{y})$ .
  - Min distance function:  $d''(\mathcal{C}_i, \mathcal{C}_j) = \min_{\mathbf{x} \in \mathcal{C}_i, \mathbf{y} \in \mathcal{C}_j} d(\mathbf{x}, \mathbf{y})$ .
  - Min distance function is not a metric:  
 $d''(\mathcal{C}_i, \mathcal{C}_j) = 0$ , even if  $\mathcal{C}_i \neq \mathcal{C}_j$ , when  $\mathcal{C}_i \cap \mathcal{C}_j \neq \emptyset$ .

# Distance Measures

## *Distance Functions between Two Sets*

- Average proximity function:
- $d''(\mathcal{C}_i, \mathcal{C}_j) = \frac{1}{|\mathcal{C}_i||\mathcal{C}_j|} \sum_{\mathbf{x} \in \mathcal{C}_i} \sum_{\mathbf{y} \in \mathcal{C}_j} d(\mathbf{x}, \mathbf{y}) .$
- **Cluster center distance:**

$$d''(\mathcal{C}_i, \mathcal{C}_j) = d(\mathbf{m}_{\mathcal{C}_i}, \mathbf{m}_{\mathcal{C}_j}) .$$

- $\mathbf{m}_{\mathcal{C}_i}, i = 1, 2$ : set representative vectors.

# Clustering Algorithms

- ***Exhaustive clustering.***
- ***Sequential Clustering:***
  - Produce single clustering with straightforward and fast methods.
  - Produce compact and hyperspherical/hyperellipsoidal clusters.
- ***Hierarchical Clustering:***
  - Cluster merge: produce a decreasing number of clusters at each step, by merging two clusters into one.
  - Cluster split: produce clusterings of increasing  $m$ .

# Clustering Algorithms

- ***Clustering by cost function optimization:***
  - Optimization of cost function  $J$  representing a clustering criterion.
  - Optimization by differential calculus.
- ***Vector quantization***
- ***Graph-based clustering***



# Exhaustive Clustering

***Exhaustive clustering*** of a vector data set  $\mathcal{D}$ :

- identify all possible partition,
- select the one optimizing a clustering criterion.
- $S(N, m)$ : number of all possible cluster outcomes, by clustering of  $N$  vectors into  $m$  groups.
- $S(N, m)$  properties:
  - $S(N, 1) = 1$  (one cluster of  $N$  vectors),
  - $S(N, N) = 1$  ( $N$  clusters of 1 vector each),
  - $S(N, m) = 0$  for  $m > N$ .

# Exhaustive Clustering

- ***Iterative equation:***

$$S(N, m) = mS(N - 1, m) + S(N - 1, m - 1).$$

- Solution: ***Stirling numbers of the second kind:***

$$S(N, m) = \frac{1}{m!} \sum_{i=0}^m (-1)^{m-i} \binom{m}{i} i^N.$$

- Prohibitive computational complexity!

# Sequential Clustering

## ***Iterative sequential algorithm:***

- Assigning data vectors  $\mathbf{x}$  to its closest cluster  $\mathcal{C}$ .
- $d(\mathbf{x}, \mathcal{C})$ : distance between a feature vector  $\mathbf{x}$  and cluster  $\mathcal{C}$ .
- User-defined parameters:
  - the distance threshold  $\varepsilon$ .
  - the maximum allowable number of clusters  $M$ .

# Sequential Clustering

- $d(\mathbf{x}, \mathcal{C}) = d(\mathbf{x}, \mathbf{m}_{\mathcal{C}})$ , when  $\mathcal{C}$  is represented by cluster center vector  $\mathbf{m}_{\mathcal{C}}$ .
- When  $\mathbf{x}$  is assigned to its closest cluster  $\mathcal{C}$  at iteration  $(t + 1)$ , iterative cluster center vector update is given by:

$$\mathbf{m}_{\mathcal{C}}^{(t+1)} = \frac{n_{\mathcal{C}}^{(t)} \mathbf{m}_{\mathcal{C}}^{(t)} + \mathbf{x}}{n_{\mathcal{C}}^{(t+1)}}.$$



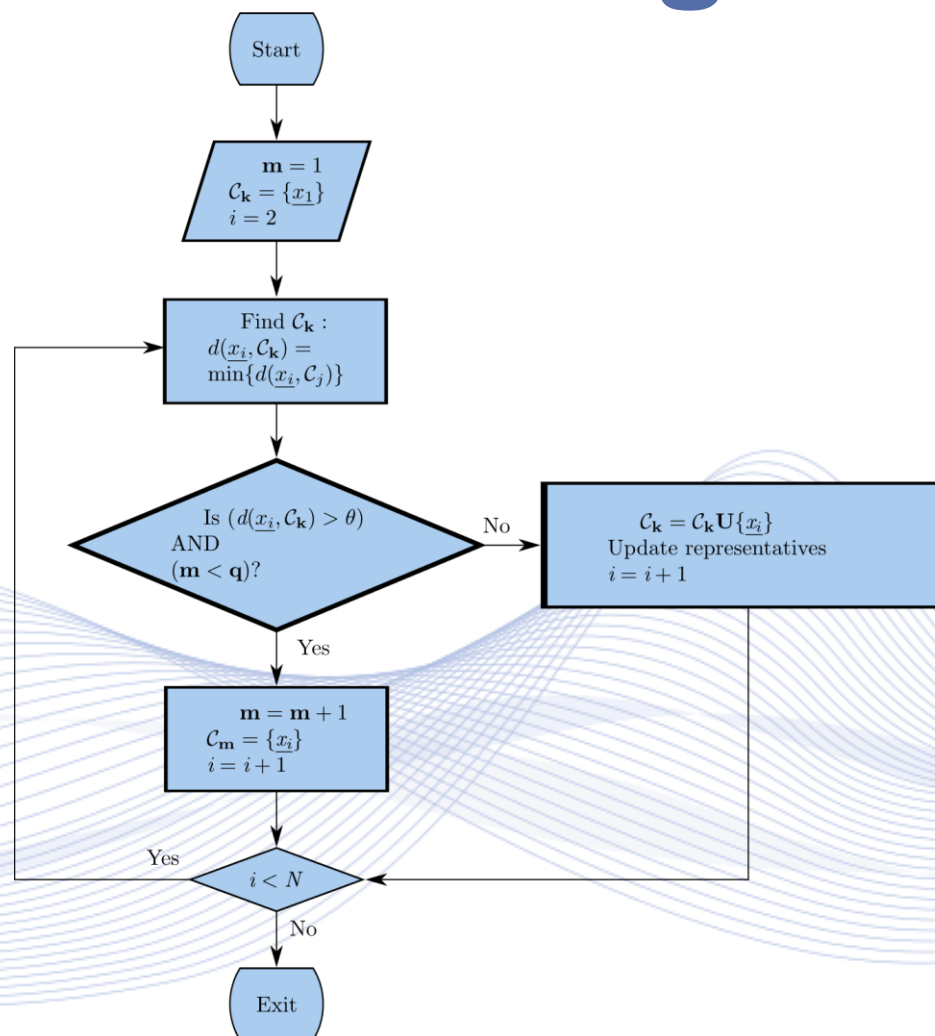
# Sequential Clustering

## *Sequential clustering algorithm*

$m = 1$

- $\mathcal{C}_m = \{\mathbf{x}_1\}$
- For  $i = 2$  to  $N$ 
  - Find  $\mathcal{C}_k: d(\mathbf{x}_i, \mathcal{C}_k) = \min_{i \leq j \leq m} d(\mathbf{x}_i, \mathcal{C}_j)$
  - If  $(d(\mathbf{x}_i, \mathcal{C}_k) > \varepsilon)$  AND  $(m < M)$  then
    - $m = m + 1$
    - $\mathcal{C}_m = \{\mathbf{x}_i\}$
  - Else
    - $\mathcal{C}_k = \mathcal{C}_k \cup \{\mathbf{x}_i\}$
    - Update class  $\mathcal{C}_k$  representation.
  - End{if}
- End{for}

# Sequential Clustering



# Sequential Clustering

## ***Properties:***

- Performance depends on data presentation to the algorithm.
- It may be used with similarity instead of distance measures by replacing min operator with max.
- Class representation by its center favors compact clusters.
- A single pass on the entire data set has  $O(Nm)$  complexity to compute  $d(\mathbf{x}_i, \mathcal{C}_k)$  for  $N$  samples and  $m < N$  clusters.

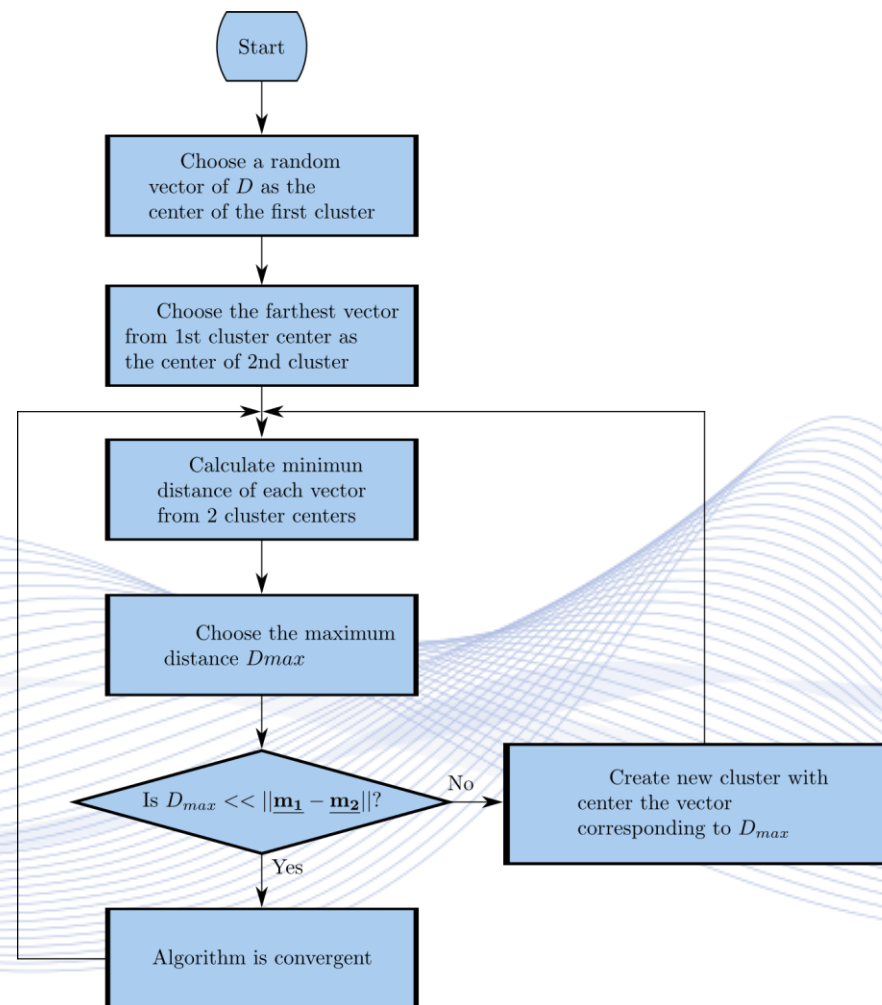
# Sequential Clustering

## Maximin Algorithm

- $m = 1$
- $\mathcal{C}_m = \{\mathbf{x}_l\}$ ,  $\mathbf{x}_l, l = 1, \dots, N$  is chosen randomly.
- For  $i = 2$  to  $N$ 
  - Find  $d'(\mathbf{x}_l, \mathcal{C}_k) = \max_{i \leq j \leq m} d'(\mathbf{x}_i, \mathcal{C}_j)$
  - If  $(d'(\mathbf{x}_l, \mathcal{C}_k) \gg d''(\mathcal{C}_i, \mathcal{C}_j), i = 1, \dots, m, j = 1, \dots, m)$ 
    - $m = m + 1$
    - $\mathcal{C}_m = \{\mathbf{x}_l\}$
  - Else
    - Find  $\mathcal{C}_k$ :  $d(\mathbf{x}_i, \mathcal{C}_k) = \min_{i \leq j \leq m} d(\mathbf{x}_i, \mathcal{C}_j)$
    - $\mathcal{C}_k = \mathcal{C}_k \cup \{\mathbf{x}_i\}$
    - Where necessary, update representatives
  - End{if}
- End{for}



# Sequential Clustering



Maximin Algorithm

# Clustering Based on Function Optimization

- Cost  $J(\mathcal{D}, \Theta)$  is a function of:
  - data set  $\mathcal{D}$  vectors and
  - an unknown cluster parameter vector/matrix/set  $\Theta$ .
- Number of clusters  $m$  is fixed.
- Goal: estimate  $\Theta$  that optimizes cost function  $J(\mathcal{D}, \Theta)$ .
- $\Theta$  is strongly depends on cluster topology.
- Compact clusters are best represented by their centers:

$$\Theta = [\mathbf{m}_1^T, \dots, \mathbf{m}_m^T]^T.$$

# K-means Algorithm

- Distances between a feature vector and a cluster:
  - Mahalanobis distance:

$$d(\mathbf{x}_i, \mathbf{m}_j) = (\mathbf{x}_i - \mathbf{m}_j)^T \mathbf{A} (\mathbf{x}_i - \mathbf{m}_j).$$

- $\mathbf{A}$ : symmetric, positive definite matrix.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{m}_j) = (\mathbf{x}_i - \mathbf{m}_j)^T (\mathbf{x}_i - \mathbf{m}_j).$$

- Minkowski distance:  $d(\mathbf{x}_i, \mathbf{m}_j) = \left( \sum_{k=1}^l |\mathbf{x}_{ik} - \mathbf{m}_{jk}|^p \right)^{\frac{1}{p}}.$
- $\mathbf{x}_{ik}, \mathbf{m}_{jk}$  are the  $k$ -th coordinates of  $\mathbf{x}_i, \mathbf{m}_j$  respectively.

# K-means Algorithm

- Cost function minimization:

$$J(\mathbf{m}_1, \dots, \mathbf{m}_m) = \sum_{i=1}^N \sum_{j=1}^m d(\mathbf{x}_i, \mathbf{m}_j) .$$

Using Euclidean distance:

$$J(\mathbf{m}_1, \dots, \mathbf{m}_m) = \sum_{i=1}^N \sum_{j=1}^m (\mathbf{x}_i - \mathbf{m}_j)^T (\mathbf{x}_i - \mathbf{m}_j) .$$



# K-means Algorithm

- Differentiation of  $J(\mathbf{m}_1, \dots, \mathbf{m}_m)$ :

$$\frac{\partial J(\mathbf{m}_1, \dots, \mathbf{m}_m)}{\partial \mathbf{m}_j} = 2 \sum_{i=1}^N (\mathbf{m}_j - \mathbf{x}_i).$$

$$\mathbf{m}_j(t) = \frac{\sum_{i=1}^N \mathbf{x}_i}{N}.$$

# K-means Algorithm

- Step 0: Initialize cluster centers  $\mathbf{m}_1(0), \dots, \mathbf{m}_m(0)$  randomly.
- Step 1: At each step ( $t$ ) assign each data sample  $\mathbf{x}_i, i = 1, \dots, N$  to the closest cluster center:

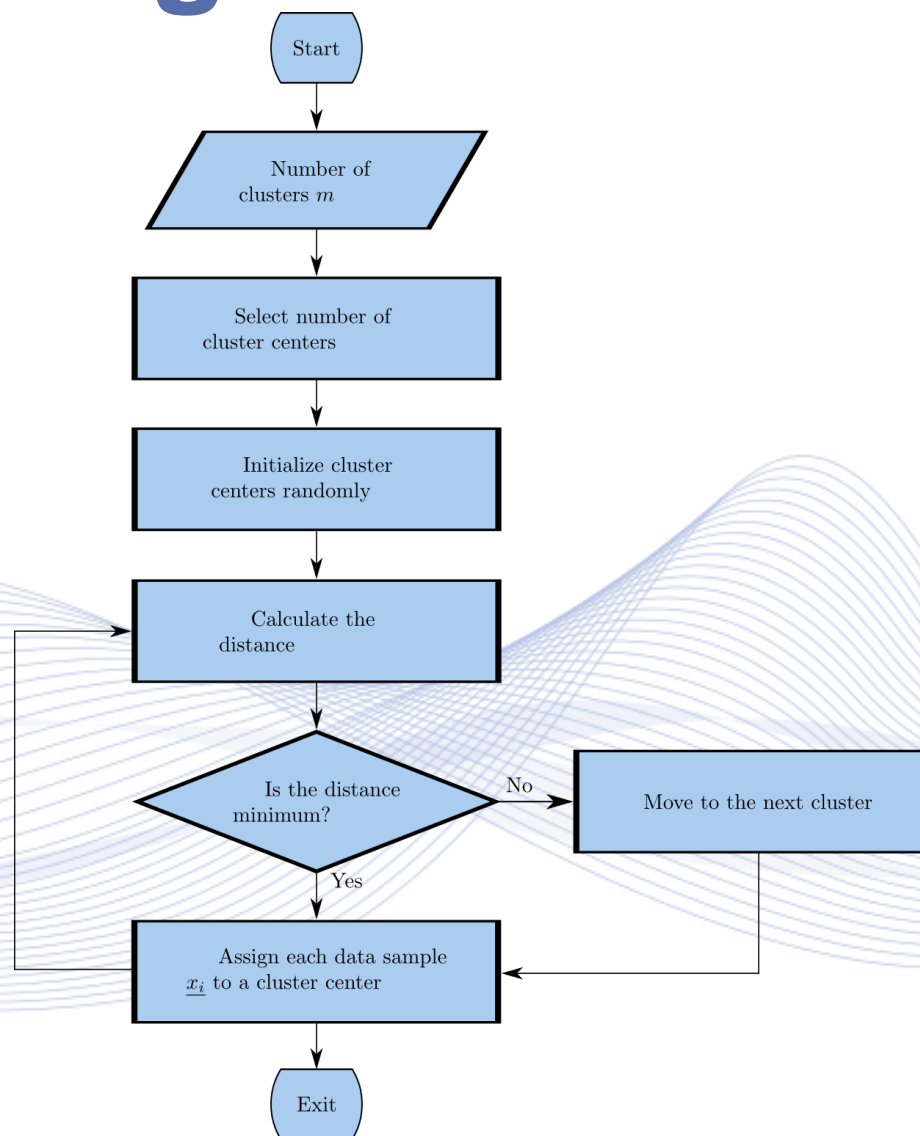
$$d(\mathbf{x}_i, \mathbf{m}_k(t)) < d(\mathbf{x}_i, \mathbf{m}_i(t)), \quad k \neq i$$

- Step 2: Update cluster  $\mathcal{C}_j, j = 1, \dots, m$  centers:

$$\mathbf{m}_j(t + 1) = \frac{\sum_{i=1}^{|\mathcal{C}_j|} \mathbf{x}_i}{|\mathcal{C}_j|}.$$

- Step 3: If  $\mathbf{m}_j(t + 1) = \mathbf{m}_j(t)$ , for every  $j = 1, \dots, m$ , stop.

# K-means Algorithm



# Isodata algorithm

- Step 1: Choose the initial cluster number  $m$  and initial cluster centers  $\mathbf{m}_1(0), \dots, \mathbf{m}_m(0)$ .
- Step 2: Classification of vectors of  $\mathcal{D}$  in  $m$  clusters, based on their minimal distance from cluster centers.
- Step 3: Update the centers, as in the algorithm of K-means.
- Step 4: If cluster cardinality is smaller than a predetermined percentage of the cardinality of  $\mathcal{D}$ , this cluster is deleted.



# Isodata algorithm

- Step 5 (**Cluster split**): Calculate the mean sample variance  $\sigma_{ij}^2$  of each cluster  $\mathcal{C}_j$  vectors along each data axis  $i$  :

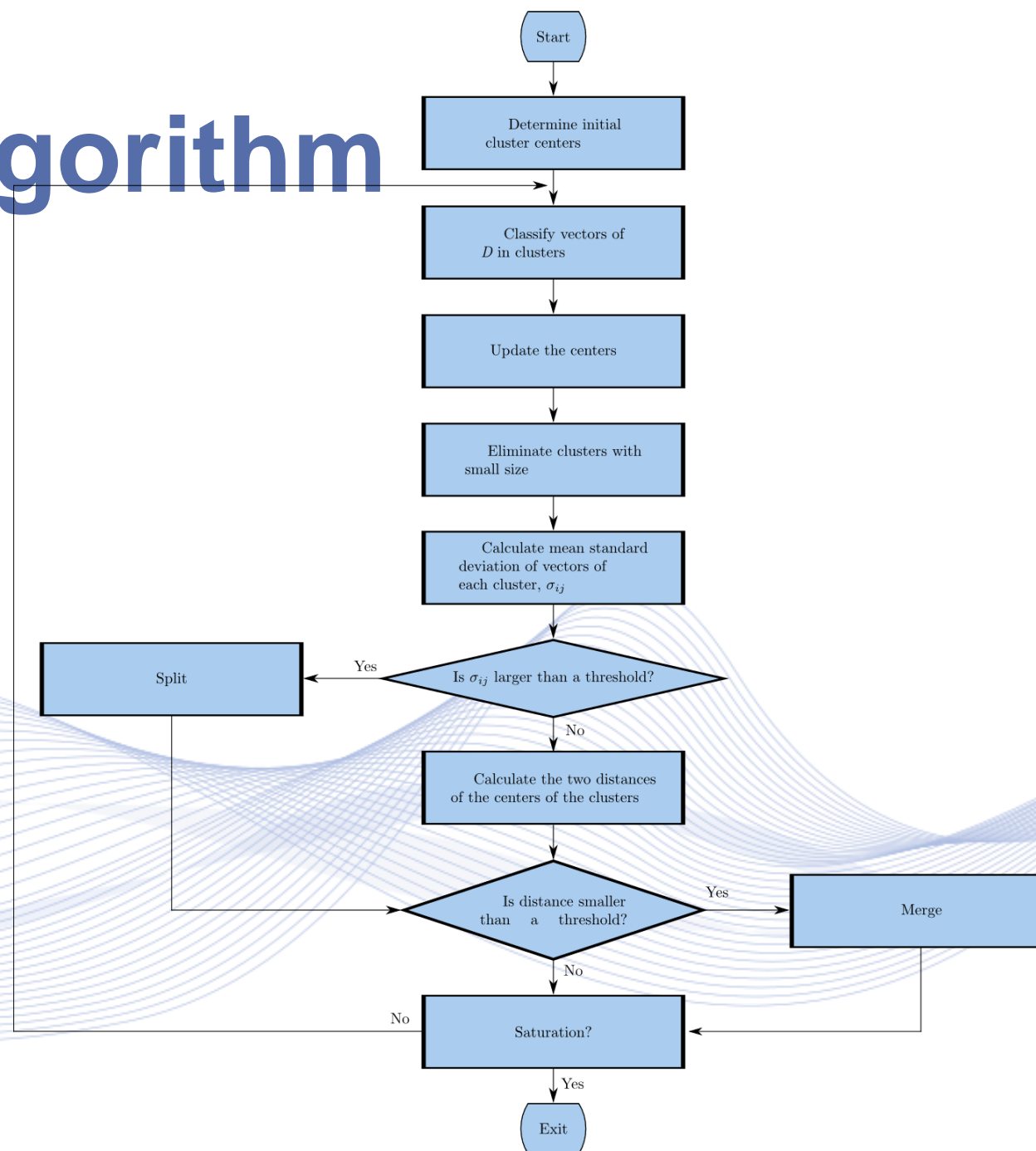
$$\sigma_{ij}^2 = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{x} \in \mathcal{C}_j} (x_{ij} - m_{ij})^2, \quad i = 1, \dots, n, j = 1, \dots, m.$$

- If some  $\sigma_{ij}$  is larger than a predetermined threshold, split  $\mathcal{C}_j$  in two and create their centers  $\mathbf{m}_j - \mathbf{c}, \mathbf{m}_j + \mathbf{c}$ .
- Update number of clusters  $m$ .

# Isodata algorithm

- Step 6 (**Cluster merge**):
  - Calculate the distances  $d''(\mathcal{C}_i, \mathcal{C}_j)$ ,  $i = 1, \dots, m, j = 1, \dots, m$  of any two clusters  $\mathcal{C}_i, \mathcal{C}_j$ .
  - If  $d''(\mathcal{C}_i, \mathcal{C}_j)$  is smaller than a threshold, merge two clusters  $\mathcal{C}_i, \mathcal{C}_j$ .
  - Update number of clusters  $m$ .

# Isodata algorithm



# Fuzzy Clustering

- In fuzzy set partitions, a vector belongs simultaneously to more than one cluster:
  - Fuzzy membership functions  $u_j, j = 1, \dots, m: \mathcal{D} \rightarrow [0,1]$ .
  - $\mathbf{m}_j$ : representative vector of  $j$ -th cluster (cluster center).
  - $\Theta = [\mathbf{m}_1^T, \dots, \mathbf{m}_m^T]^T$
  - $\mathbf{U}: N \times m$  matrix whose element  $(i,j)$  equals to  $u_j(\mathbf{x}_i)$ .
- $d(\mathbf{x}_i, \mathbf{m}_j)$ : distance between  $\mathbf{x}_i$  and  $\mathbf{m}_j$ .



# Fuzzy Clustering

- Distances between a feature vector and a cluster:

- Mahalanobis distance:

$$d(\mathbf{x}_i, \mathbf{m}_j) = (\mathbf{x}_i - \mathbf{m}_j)^T \mathbf{A} (\mathbf{x}_i - \mathbf{m}_j).$$

- $\mathbf{A}$ : symmetric, positive definite matrix.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{m}_j) = (\mathbf{x}_i - \mathbf{m}_j)^T (\mathbf{x}_i - \mathbf{m}_j).$$

- Minkowski distance:  $d(\mathbf{x}_i, \mathbf{m}_j) = \left( \sum_{k=1}^l |\mathbf{x}_{ik} - \mathbf{m}_{jk}|^p \right)^{\frac{1}{p}}.$
- $\mathbf{x}_{ik}, \mathbf{m}_{jk}$  are the  $j$ th coordinates of  $\mathbf{x}_i, \mathbf{m}_j$  respectively.

# Fuzzy Clustering Algorithms

- Cost function minimization:

$$J_q(\mathbf{\Theta}, \mathbf{U}) = \sum_{i=1}^N \sum_{j=1}^m u_{ij}^q d(\mathbf{x}_i, \mathbf{m}_j),$$

with respect to  $\mathbf{\Theta}$  and  $\mathbf{U}$ , subject to the constraints:

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

- $u_{ij} \in [0,1], i = 1, \dots, N, j = 1, \dots, m,$
- $0 < \sum_{i=1}^N u_{ij} < N, j = 1, \dots, m.$

- $q > 1$ : **fuzzifier parameter.**

# Fuzzy Clustering Algorithms

Minimization of  $J_q(\Theta, \mathbf{U})$  with respect to  $\mathbf{U}$  under constraints:

- Lagrangian function minimization:

$$\mathcal{J}(\Theta, \mathbf{U}) = \sum_{i=1}^N \sum_{j=1}^m u_{ij}^q d(\mathbf{x}_i, \mathbf{m}_j) - \sum_{i=1}^N \lambda_i (\sum_{j=1}^m u_{ij} - 1).$$

- Partial differentiation of  $\mathcal{J}(\Theta, \mathbf{U})$  with respect to  $u_{rs}$ :

$$\frac{\partial \mathcal{J}(\Theta, \mathbf{U})}{\partial u_{rs}} = q u_{rs}^{q-1} d(\mathbf{x}_r, \mathbf{m}_s) - \lambda_r = 0.$$

# Fuzzy Clustering Algorithms

Solution:

$$u_{rs} = \left( \frac{\lambda_r}{q d(\mathbf{x}_r, \mathbf{m}_s)} \right)^{\frac{1}{q-1}}, \quad s = 1, \dots, m.$$

Substitution of  $u_{rs}$  in the constraint  $\sum_{j=1}^m u_{rj} = 1$  leads to:

$$\lambda_r = \frac{q}{\left( \sum_{j=1}^m \left( \frac{1}{d(\mathbf{x}_r, \mathbf{m}_j)} \right)^{\frac{1}{q-1}} \right)^{q-1}}.$$



# Fuzzy Clustering Algorithms

- Combining the two previous equations:

$$u_{rs} = \frac{1}{\sum_{j=1}^m \left( \frac{d(\mathbf{x}_r, \mathbf{m}_s)}{d(\mathbf{x}_r, \mathbf{m}_j)} \right)^{\frac{1}{q-1}}}, \quad r = 1, \dots, N, \quad s = 1, \dots, m.$$

- Gradient of  $J(\Theta, \mathbf{U})$  with respect to  $\mathbf{m}_j$ :

$$\frac{\partial J(\Theta, \mathbf{U})}{\partial \mathbf{m}_j} = \sum_{i=1}^N u_{ij}^q \frac{\partial d(\mathbf{x}_i, \mathbf{m}_j)}{\partial \mathbf{m}_j} = \mathbf{0}, \quad j = 1, \dots, m.$$

# Fuzzy Clustering Algorithms

## *Fuzzy k-means algorithm:*

- Using Mahalanobis distance:

$$\frac{\partial d(\mathbf{x}_i, \mathbf{m}_j)}{\partial \mathbf{m}_j} = 2\mathbf{A}(\mathbf{m}_j - \mathbf{x}_i).$$

- Substituting the above equation in  $\frac{\partial J(\Theta, \mathbf{U})}{\partial \mathbf{m}_j}$ , we obtain:

$$\sum_{i=1}^N u_{ij}^q(t-1) \mathbf{A}(\mathbf{m}_j - \mathbf{x}_i) = \mathbf{0}.$$

# Fuzzy Clustering Algorithms

- Since  $\mathbf{A}$  is positive definite, it can be discarded:

$$\mathbf{m}_j(t) = \frac{\sum_{i=1}^N u_{ij}^q(t-1) \mathbf{x}_i}{\sum_{i=1}^N u_{ij}^q(t-1)}.$$

- Termination criterion:

$$||\mathbf{m}_j(t) - \mathbf{m}_j(t-1)|| < \epsilon, \quad j = 1, \dots, m.$$

# Vector Quantization

## ***Vector Quantization:***

- Compact representation of the data set  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_i \in \mathbb{R}^n$  by much fewer vectors  $\mathbf{m}_i \in \mathbb{R}^n$ ,  $i = 1, \dots, m$ ,  $m \ll N$  of the same dimensionality.
- Each  $\mathbf{m}_i \in \mathbb{R}^n$  corresponds to one cluster  $\mathcal{C}_i$ ,  $i = 1, \dots, m$ .



# Vector Quantization

- A data set  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_i \in \mathbb{R}^n$  is to be clustered (partitioned).
- Desired cluster number  $m \ll N$ .
- Distance measure  $d(\mathbf{x}, \mathbf{y})$  between two vectors  $\mathbf{x}, \mathbf{y}$ .
- Calculation of cluster centers.
- Sorting algorithm to decide vector proximity.

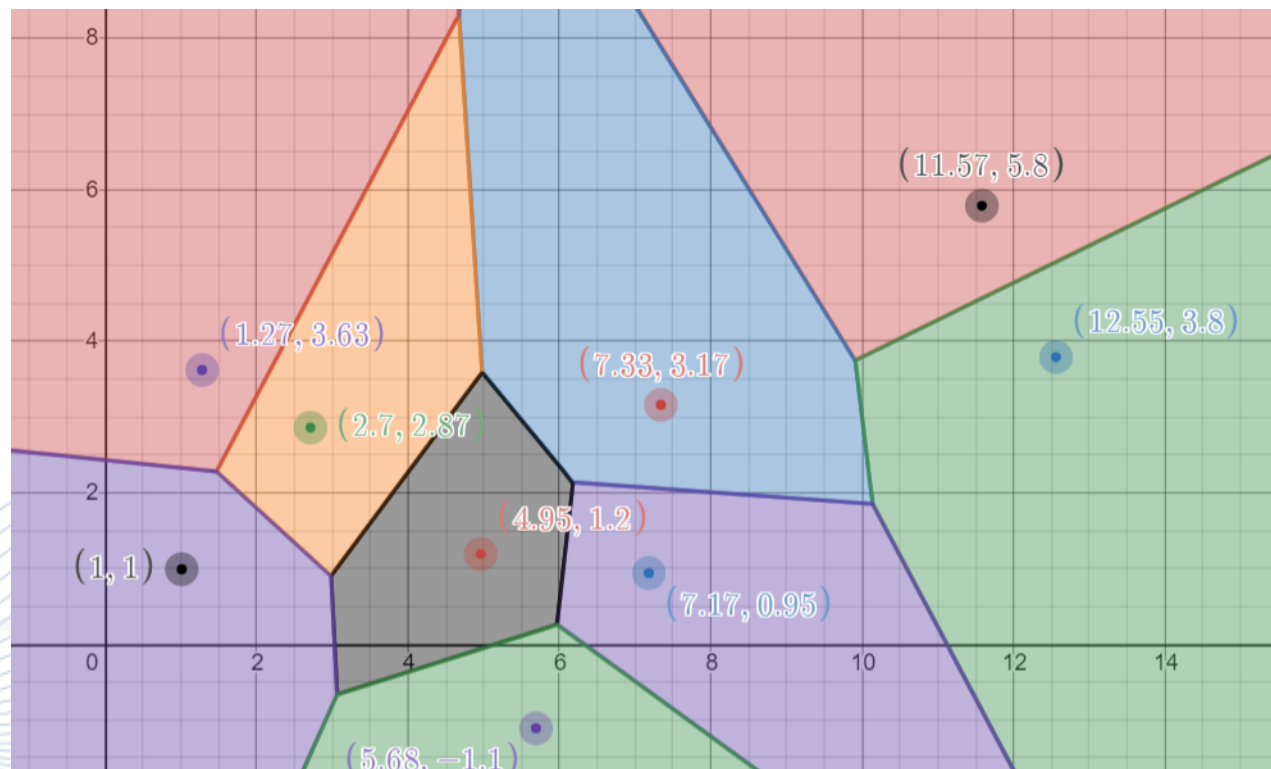
# Vector Quantization

- Data vectors are partitioned in  $m$  clusters  $\{\mathcal{C}_i, i = 1, \dots, m\}$ .
- Mapping:  $\mathbf{m} = Q(\mathbf{x})$ .
- $\mathbb{R}^n$  is partitioned in  $m$  **Voronoi regions** (one per cluster).
- Each Voronoi region (cell)  $\mathcal{R}_i$  is represented by  $\mathbf{m}_i \in \mathbb{R}^n$ ,  $i = 1, \dots, m$ :

$$|\mathbf{x} - \mathbf{m}_i| < |\mathbf{x} - \mathbf{m}_j|, \quad i \neq j.$$

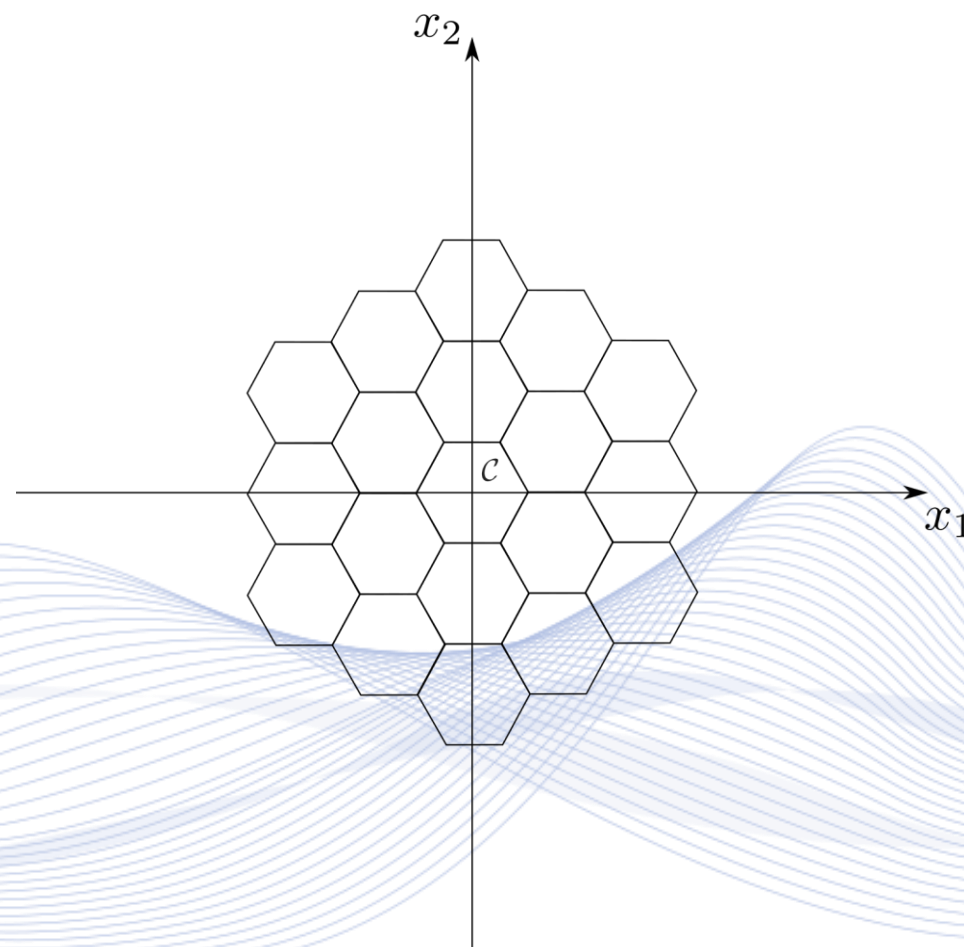
- Cluster  $\mathcal{C}_i, i = 1, \dots, m$  vectors reside in  $\mathcal{R}_i$ .
- Voronoi cells may have regular structure.

# Vector Quantization



Voronoi regions and clusters in  $\mathbb{R}^2$ .

# Vector Quantization



Hexagonal Voronoi cells in  $\mathbb{R}^2$ .



# Vector Quantization

- **Codevectors**  $\mathbf{m}_i \in \mathbb{R}^n$ ,  $i = 1, \dots, m$ : cluster centers.
- Any vector in Voronoi region  $\mathcal{C}_i$  are represented by  $\mathbf{m}_i$ .
- **Quantization error (distortion)**  $|\mathbf{x} - \mathbf{m}_i|$ .

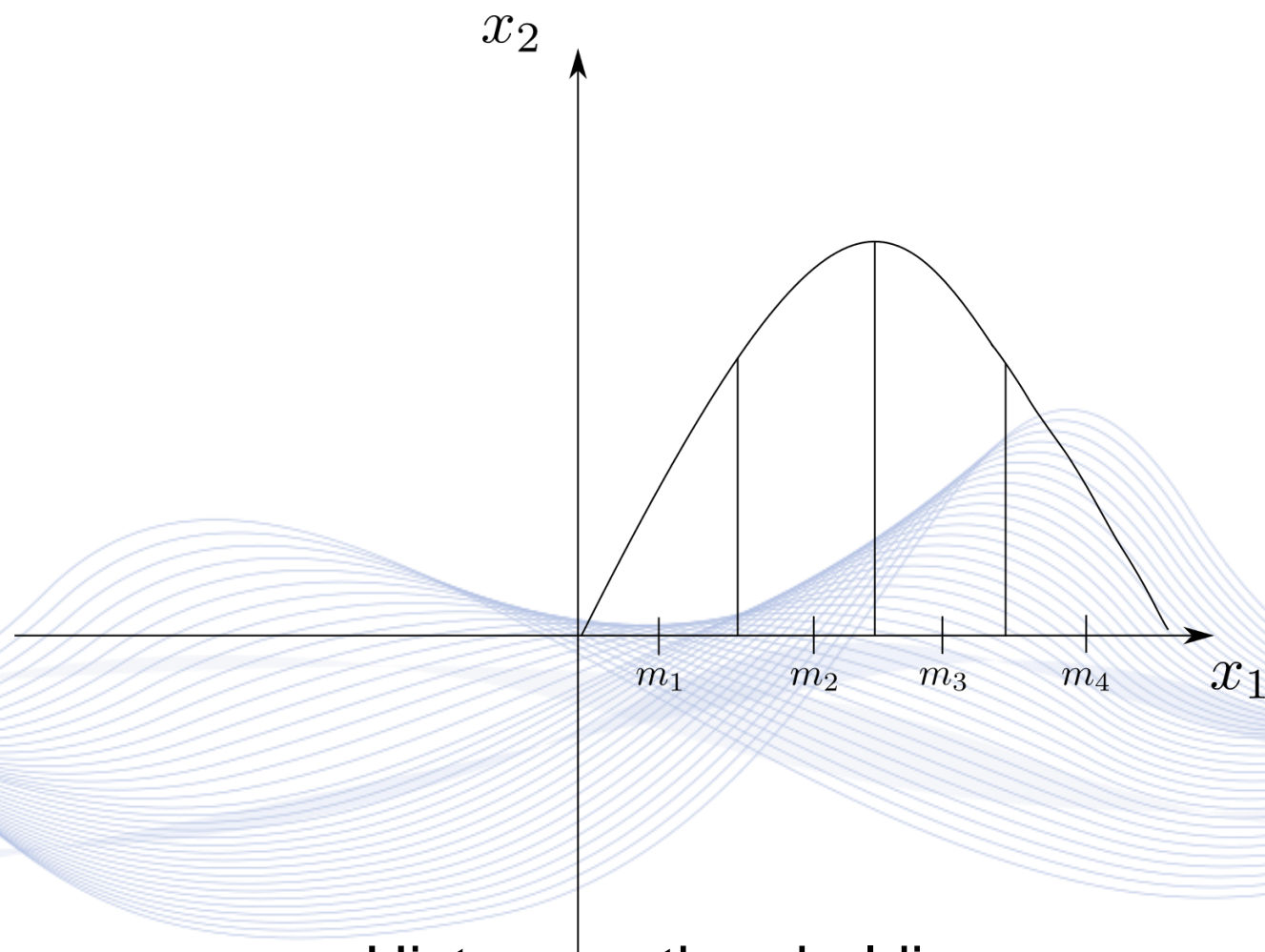
- If  $x_i \in \mathbb{R}$ , classical (scalar) quantization:

$$|x - m_i| < |x - m_j|, \quad i \neq j.$$

$$\mathcal{R}_i = [(m_{i-1} + m_i)/2, (m_i + m_{i+1})/2].$$

- It can be applied for 1D **histogram thresholding**.

# Vector Quantization



Histogram thresholding.

# Vector Quantization

- Advantages:
  - Reduced storage requirements and faster processing.
  - Comparing two vectors has little computational complexity, thus VQ algorithms are not time-consuming.
- Disadvantages:
  - Quantization error.

# Vector Quantization

## *Linde-Buzo-Gray Algorithm*

- Initialization: Choose  $m$  random vectors  $\mathbf{m}_i, i = 1, \dots, m$ .
- Recursion:
  - Each vector  $\mathbf{x}$  from set  $\mathcal{D}$  is assigned to vector  $\mathbf{m}_k$ :
 
$$k = \arg_i \min d(\mathbf{x}, \mathbf{m}_i).$$
  - Calculate total quantization error:  $J = \sum_i \sum d(\mathbf{x}, \mathbf{m}_i)$ .
  - If  $J < \varepsilon$  is smaller than a threshold, stop.
  - Else calculate new centers  $\mathbf{m}_i, i = 1, \dots, m$  and repeat previous steps until convergence.
- Very similar to k-means algorithm.



# Vector Quantization

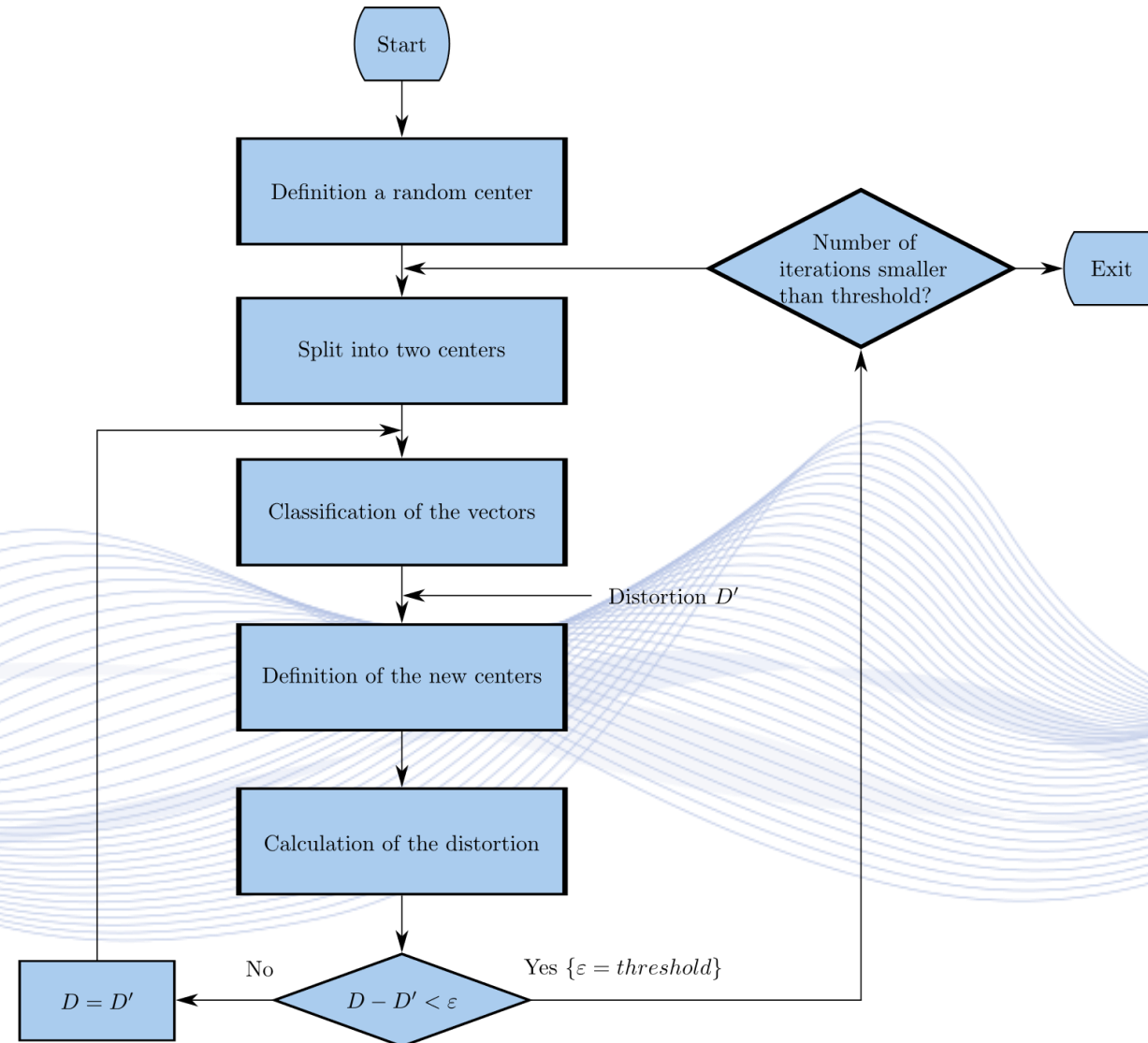
## *Binary Split Algorithm*

Initialization: Define a random cluster center. All vectors of data set  $\mathcal{D}$  belong to the same cluster.

- Iteration  $t$ : (Total  $L$  iterations producing  $2^L$  codevectors)
  - Each codevector  $\mathbf{m}_i$  is broken into two vectors  $\mathbf{m}_i(1 + \epsilon)$ ,  $\mathbf{m}_i(1 - \epsilon)$ ,  $\epsilon \in [0.01, 0.005]$
  - Each vector  $\mathbf{x}$  from training set is assigned to the closest vector  $\mathbf{m}_k$ :  

$$k = \arg_i \min d(\mathbf{x}, \mathbf{m}_i).$$
  - Calculate total quantization error:  $J = \sum_i \sum d(\mathbf{x}, \mathbf{m}_i)$ .
  - If  $J$  is smaller than a threshold, stop.
  - Calculate new center for each  $i$ . If  $t < L$  repeat previous steps, else stop.

# Vector Quantization



# Learning Vector Quantization

- ***Learning Vector Quantization (LVQ)*** was proposed by Kohonen.
- Also called ***Self-Organizing Maps (SOM)***.
- Initial values are set based on classic cluster algorithms.
- Code vectors  $\mathbf{m}_i$  are iteratively optimized.
- Goal: clustering based on the nearest-neighbor rule.
- Clusters are described by their codevectors.
- Cluster boundaries matter.

# Learning Vector Quantization

## *Mathematical model of vector quantization*

- $\mathbf{x}$ : vector to be assigned to a cluster.
- Employ Euclidean distance.
- Find the winner cluster:
  - Closest cluster center  $\mathbf{m}_k$ :

$$d(\mathbf{x}, \mathbf{m}_k) = \min_i \{d(\mathbf{x}, \mathbf{m}_i)\}, \quad \forall i \neq k.$$



# Learning Vector Quantization

- Cluster center updating:

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) + a(t)[\mathbf{x} - \mathbf{m}_k(t)]$$

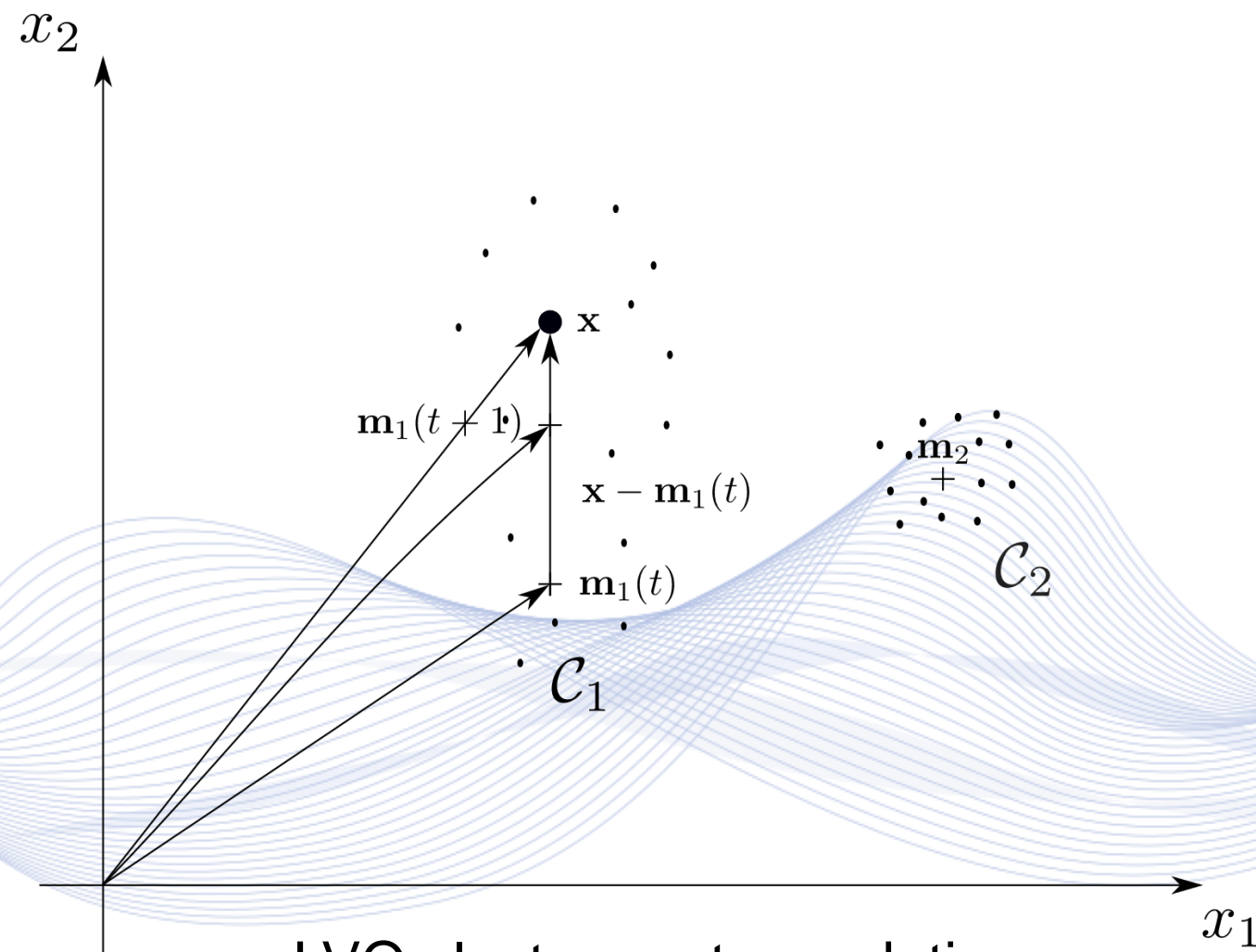
$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t), \quad \text{for } i \neq k,$$

- $0 \leq a(t) \leq 1$ .
- Distance  $d(\mathbf{x}, \mathbf{m}_k)$  is monotonically decreasing:
  - If  $\delta \mathbf{x}_i = \mathbf{m}_i(t+1) - \mathbf{m}_i(t)$ , then  $[\nabla_{\mathbf{m}_k} d(\mathbf{x}, \mathbf{m}_k)]^T \delta \mathbf{m}_i < 0$ .

# Learning Vector Quantization

- Incremental algorithm: data may come on the fly.
- For the first steps,  $a(t)$  value shall be close to 1.
- Depending on total number of steps,  $a(t)$  decreases:
  - Linear, exponential decrease.
- When  $a(t)$  falls below the threshold, the algorithm freezes.
- Updating of winning cluster neighborhood  $\mathcal{N}(\mathcal{C})$  can be performed.

# Vector Quantization



LVQ cluster center updating.

# Learning Vector Quantization Algorithms

## *Competitive cluster center updating*

- For the winner cluster:

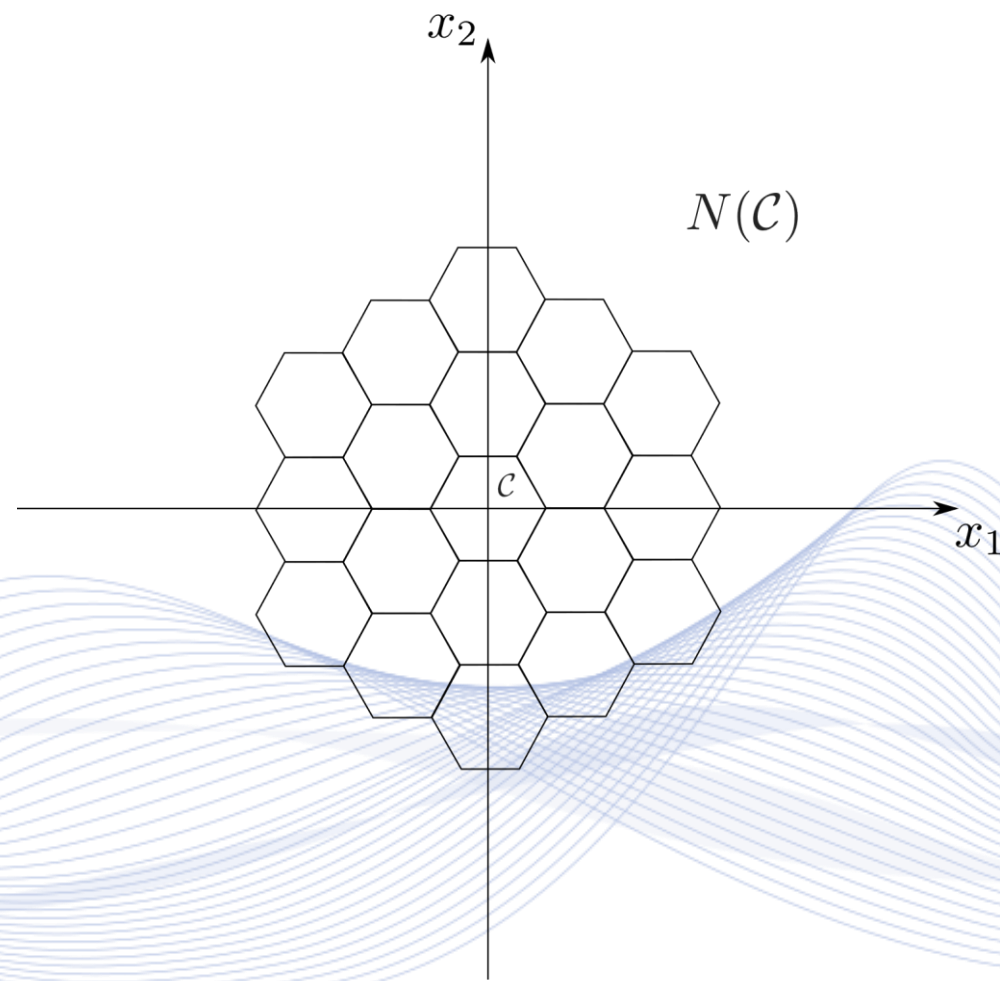
$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) + a(t)[\mathbf{x}(t) - \mathbf{m}_k(t)].$$

- For the rest of the clusters:

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) - a(t)[\mathbf{x}(t) - \mathbf{m}_k(t)].$$



# Vector Quantization



Hexagonal Voronoi cell neighborhood  $\mathcal{N}(\mathcal{C})$  in  $\mathbb{R}^2$ .

# Learning Vector Quantization Algorithms

## *Cooperative cluster center updating*

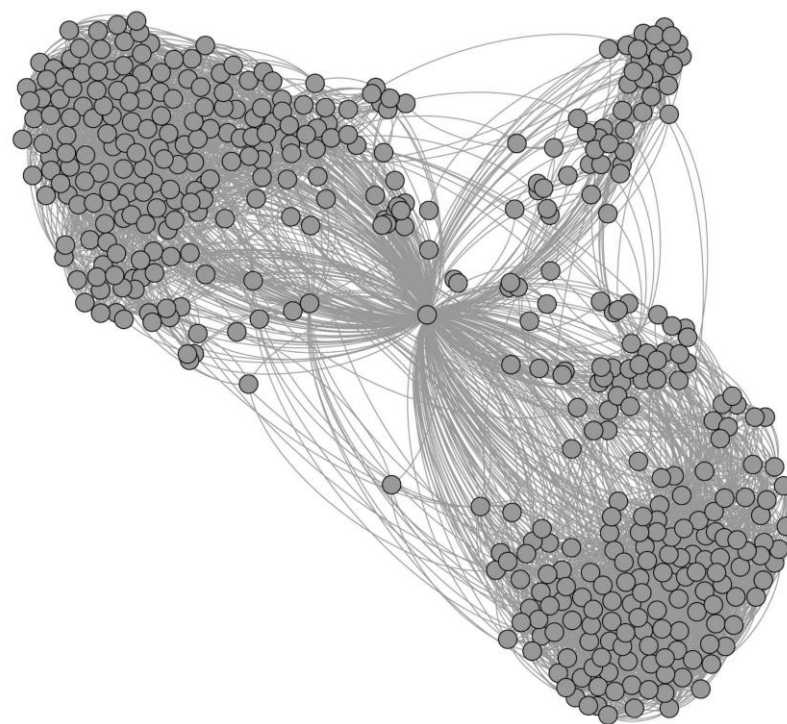
- For clusters within neighborhood  $\mathcal{N}(\mathcal{C}_k)$ :

$$\mathbf{m}_k(t + 1) = \mathbf{m}_k(t) - a(t)[\mathbf{x}(t) - \mathbf{m}_k(t)].$$

- For the rest of the clusters:

$$\mathbf{m}_k(t + 1) = \mathbf{m}_k(t), \quad i \neq k.$$

# Graph-based Clustering



Data graph visualization.

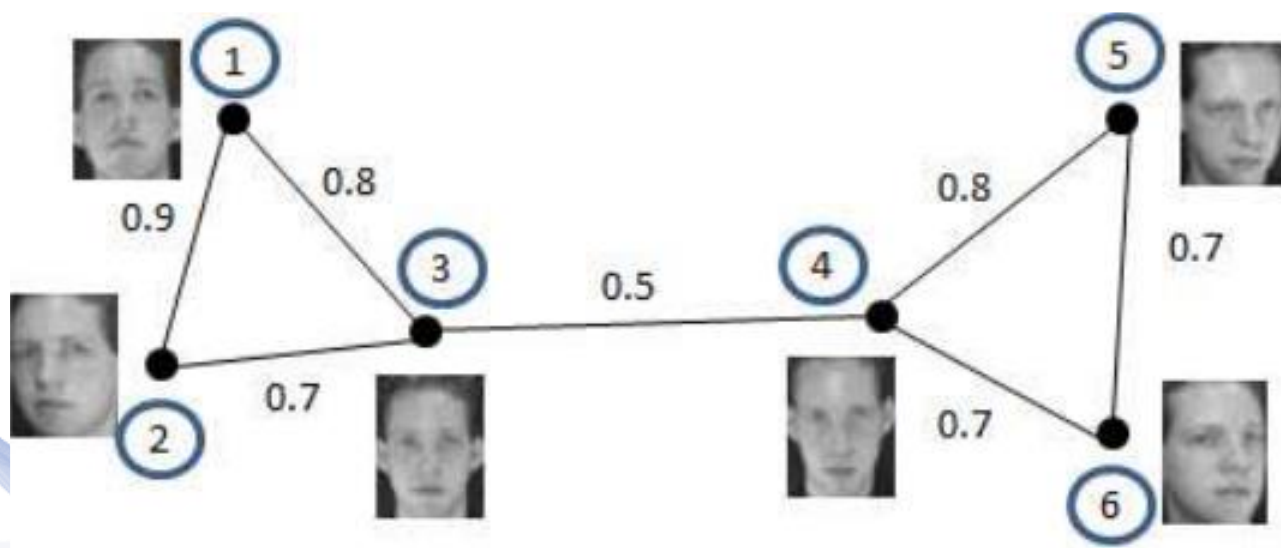
# Graph-based Clustering

## ***Similarity graph, Adjacency/Similarity matrix***

- Let  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  be the data set where  $\mathbf{x}_i \in \mathbb{R}^n$ .
- Construct a graph  $\mathcal{G}(\mathcal{V}, \mathcal{E})$ , where each graph vertex corresponds to a point  $\mathbf{x}_i, i = 1, \dots, N$ .
- Similarity graphs can be weighted connected and undirected.
- Graph  $N \times N$  ***adjacency matrix***:  $\mathbf{A} \in \{0,1\}^{N \times N}$ .
- ***Similarity (weight) matrix***:  $\mathbf{W} = [W_{ij}] \in \mathbb{R}^{N \times N}$ .



# Graph-based Clustering



1	0.9	0.8	0	0	0
0.9	1	0.7	0	0	0
0.8	0.7	1	0.5	0	0
0	0	0.5	1	0.8	0.7
0	0	0	0.8	1	0.7
0	0	0	0.7	0.7	1

a) Similarity graph; b) Similarity matrix.

# Graph-based Clustering

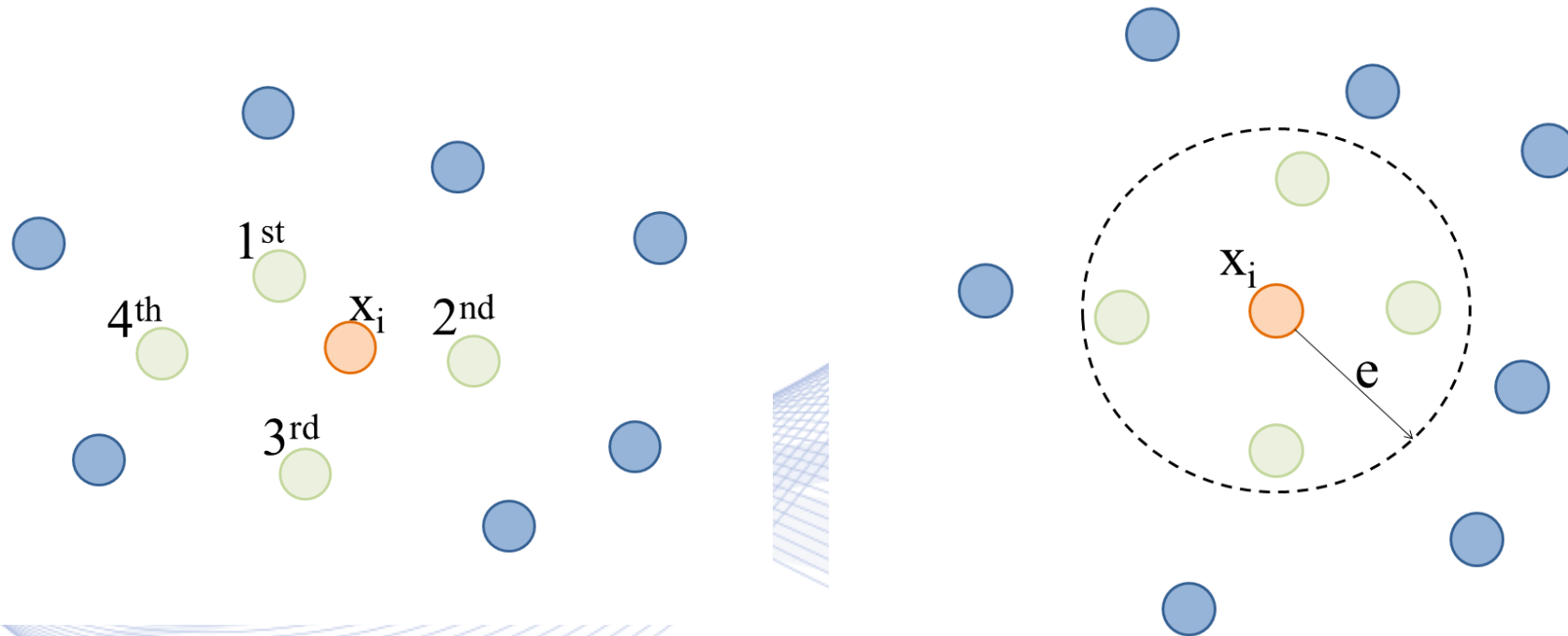
- **Vertex degree**: number of vertex connection in  $A$ .
- **Gaussian kernel** for edge weight calculation:

$$W(i, j) = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}, & \text{if } \|\mathbf{x}_i - \mathbf{x}_j\| < e, \\ 0, & \text{otherwise.} \end{cases}$$

- $e$ : is a user-defined constant.
- $\|\cdot\|$  is Euclidean norm.

# Graph-based Clustering

## *Nearest neighbor graphs*



a)  $k$ -nearest neighbors graph; b)  $e$ -neighborhood graph.

# Graph-based Clustering

## *Graph Clustering*

- Cluster graph vertices (data vectors) into tightly linked clusters.
- Vertices of the same cluster are:
  - Strongly connected to each other and
  - sparsely connected to the rest of the graph.
- ***Intra-cluster connectivity***: measured by the cluster density.
- ***Inter-cluster connectivity***: measured by ***graph cut*** cardinality.



# Graph-based Clustering

## ***Global clustering algorithms***

- ***Iterative methods:***
  - Go through all vertices and assign them to clusters.
  - Decisions based on optimization of a node connectivity metric.
- ***Online method:***
  - Process one vertex at a time and update clusters based on what has been encountered thus far.

# Graph-based Clustering

- ***Hierarchical structure:***
  - Clusters not rigidly defined.
  - Subclusters can be contained in the same cluster.

# Graph-based Clustering

## ***Adjacency matrix eigenanalysis:***

- Adjacency matrix eigenvalues and eigenvectors:

$$\mathbf{A}\mathbf{u}_i = \lambda_i \mathbf{u}_i, \quad i = 0, \dots, N - 1.$$

- $\lambda_i, i = 0, \dots, N - 1$ : roots of ***characteristic polynomial***:

$$\det(\mathbf{A} - \lambda \mathbf{I}) = 0.$$

- Adjacency matrix eigen-decomposition:  $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ .

# Graph-based Clustering

***Laplacian matrix eigenanalysis:***

$$\mathbf{L} = \mathbf{D} - \mathbf{A}.$$

- **D**:  $N \times N$  diagonal **vertex degree** matrix.

- Symmetric Laplacian matrix :

$$\mathbf{L}_S \triangleq \mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}.$$

- Random walk Laplacian matrix :

$$\mathbf{L}_R \triangleq \mathbf{D}^{-1} \mathbf{L} = \mathbf{I} - \mathbf{D}^{-1} \mathbf{A}.$$



# Graph-based Clustering

## *Laplacian matrix eigenanalysis:*

- Non-decreasing eigenvalue order:

$$\lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}.$$

- **Graph spectrum** is the eigenvalue set:  $\{\lambda_i, \quad i = 0, \dots, N - 1\}$
- It is invariant to **graph isomorphism**
  - Graph vertex permutations.
- Non-isomorphic graphs can be co-spectral.

# Graph-based Clustering

- $\lambda_0$  is always zero,  $0 = \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1} \leq 2$ .

$$\sum_{i=0}^{N-1} \lambda_i = N.$$

- $\lambda_{N-1} = 2$ , if graph  $\mathcal{G}$  is **bipartite**.

# Graph-based Clustering

- ***Algebraic connectivity*** (eigenvalue  $\lambda_1$ ):
  - If  $\lambda_1 > 0$ :
    - graph  $\mathcal{G}$  is connected.
  - else:
    - multiplicity of eigen value 0 is equal number of connected graph components.

# Graph-based Clustering

- Graph comprised of  $k$  disjoint **cliques**:
  - $k$  smallest eigenvalues of normalized Laplacian matrix are 0.
  - $i$ -th corresponding eigenvector ( $0 \leq i \leq k - 1$ ) has non-zero values for vertices of the  $i$ -th clique.
- Adding edges cause the eigenvalues to increase and change slightly corresponding eigenvectors.



# Graph-based Clustering

Graph clustering based on ***spectral bisection***:

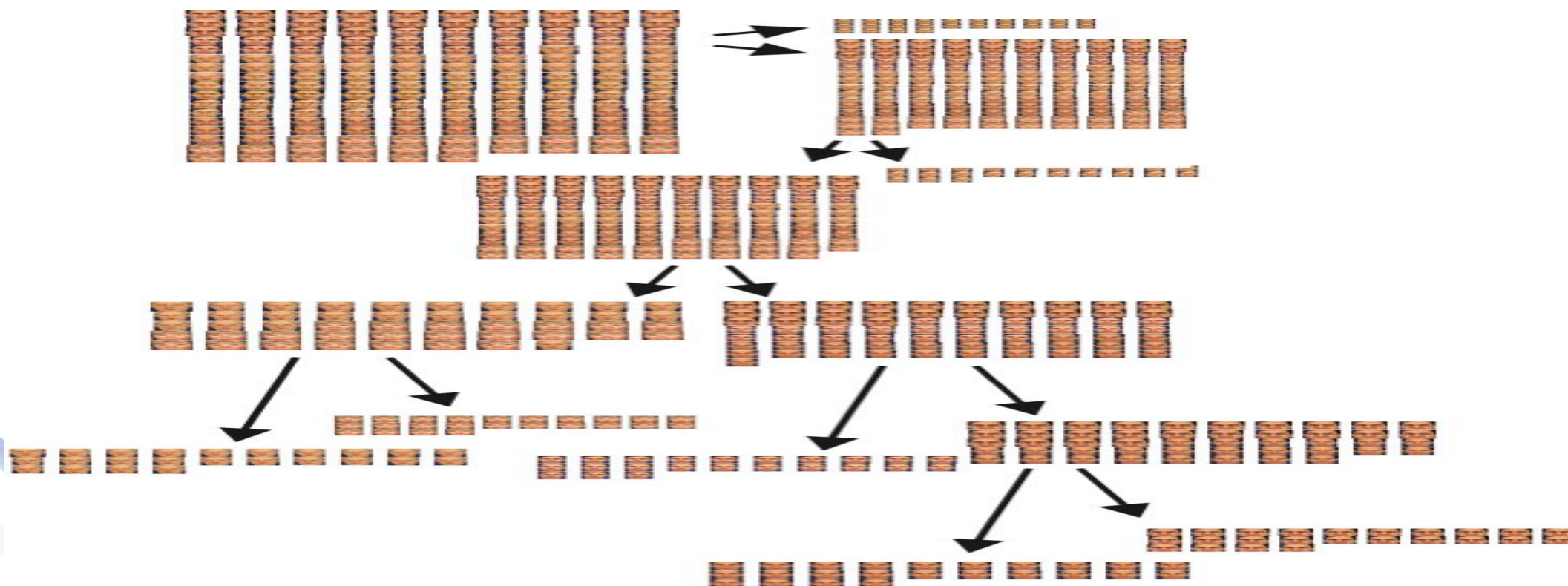
- 2-way graph partitioning.
- It uses the so-called ***Fiedler vector***:
  - eigenvector  $\mathbf{u}_1$  corresponding to eigenvalue  $\lambda_1$  of Laplacian matrix.

# Graph-based Clustering

## ***N-Cut Graph Clustering***

- When there are 2 clusters with strong internal connectivity and sparsely connected:
  - Positive Fiedler vector entries correspond to one cluster and negative to the other.
  - This provide a bisection of the graph in two subgraphs.
- Iterative bisection of the resulting subgraphs.

# Graph-based Clustering



N-Cut Graph Clustering (2-way partitioning).

# Graph-based Clustering

## ***Edge-based bisection:***

- Compute Fiedler vector.
- Split vertices into 2 groups:
  - their relevant Fiedler vector entries are below/above the Fiedler vector entries median.
- Edges between these two groups are cut.



# Graph-based Clustering

## ***Vertex-based bisection:***

- Compute Fiedler vector.
- Find the largest gap in Fiedler vector entries
- Split Fiedler vector entries accordingly.
- Split the graph at the cut provides the best cut quotient.

# Graph-based Clustering

## ***Spectral graph clustering:***

- Perform eigenanalysis on one of the normalized Laplacians.
- extract  $r$  eigenvectors corresponding to the smallest eigenvalues excluding  $\lambda_0$ .
- Store eigenvectors in a  $N \times r$  matrix  $\mathbf{U}$ .
- Its rows are the new data representation.
- Use any standard clustering algorithm to cluster them.

# Graph-based Clustering

Graph-based clustering properties:

- Little user input is needed.
- Trivial clusters easily avoided.
- Unlikely to get bad clustering results.
- They cannot be employed in extremely large graphs:
  - memory limitations.
- Eigenanalysis has  $O(N^3)$  computational complexity.

# Q & A

**Thank you very much for your attention!**

**More material in  
<http://icarus.csd.auth.gr/cvml-web-lecture-series/>**

**Contact: Prof. I. Pitas  
[pitass@csd.auth.gr](mailto:pitass@csd.auth.gr)**