

# Clustering algorithms

Konstantinos Koutroumbas

## Unit 11

- Density-based clust. Alg. (DENCLUE)
- Spectral clustering
- Clustering for high-dim. data (dim. reduction, subspace clust.)
- Combination of clusterings

# Density-based algorithms for large data sets

## DENsity-based CLUstEring (DENCLUE) Algorithm

### Definitions

The **influence function**  $f^{\mathbf{y}}(\mathbf{x})$  for a point  $\mathbf{y} \in X$  is a **positive function** that decays to zero as  $\mathbf{x}$  “moves away” from  $\mathbf{y}$  ( $d(\mathbf{x}, \mathbf{y}) \rightarrow \infty$ ). Typical examples are:

$$f^{\mathbf{y}}(\mathbf{x}) = \begin{cases} 1, & \text{if } d(\mathbf{x}, \mathbf{y}) < \sigma \\ 0, & \text{otherwise} \end{cases}, \quad f^{\mathbf{y}}(\mathbf{x}) = e^{-\frac{d(\mathbf{x}, \mathbf{y})^2}{2\sigma^2}}$$

where  $\sigma$  is a user-defined function.

The **density function based on  $X$**  is defined as (Remember the Parzen windows):

$$f^X(\mathbf{x}) = \sum_{i=1}^N f^{\mathbf{x}_i}(\mathbf{x})$$

### **The Goal:**

- (a) **Identify** all “**significant**” **local maxima**,  $\mathbf{x}_j^*$ ,  $j = 1, \dots, m$ , of  $f^X(\mathbf{x})$
- (b) **Create** a cluster  $C_j$  for each  $\mathbf{x}_j^*$  and **assign** to  $C_j$  all points  $\mathbf{x}$  of  $X$  that lie within the “**region of attraction**” of  $\mathbf{x}_j^*$ .

# Density-based algorithms for large data sets

## The DENCLUE Algorithm (cont.)

### Two clarifications

- The **region of attraction** of  $\mathbf{x}_j^*$  is defined as the **set** of points  $\mathbf{x} \in R^l$  such that if a “**hill-climbing**” (such as the steepest ascent) method is **applied** on  $f^X(\mathbf{x})$ , initialized by  $\mathbf{x}$ , it will **terminate** arbitrarily close to  $\mathbf{x}_j^*$ .
- A **local maximum** is considered as **significant** if  $f^X(\mathbf{x}_j^*) \geq \xi$  ( $\xi$  is a user-defined parameter).

### Approximation of $f^X(\mathbf{x})$

$$f^X(\mathbf{x}) = \sum_{i=1}^N f^{x_i}(\mathbf{x}) \approx \sum_{x_i \in Y(\mathbf{x})} f^{x_i}(\mathbf{x})$$

where  $Y(\mathbf{x})$  is the set of **points** in  $X$  that lie “**close**” to  $\mathbf{x}$ .

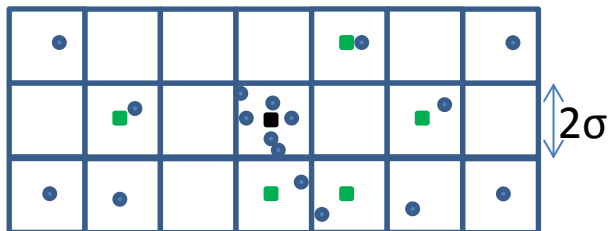
The above framework is used by the DENCLUE algorithm.

# Density-based algorithms for large data sets

## The DENCLUE Algorithm (cont.)

### DENCLUE algorithm

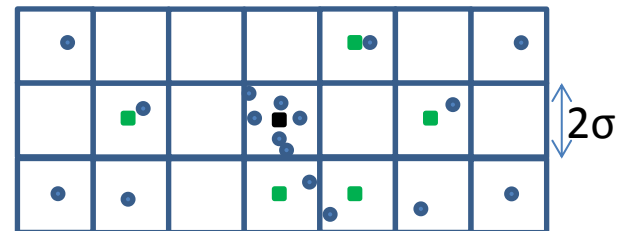
- Preclustering stage (identification of regions dense in points of  $X$ )
  - Apply an  $l$ -dimensional grid of edge-length  $2\sigma$  in the  $R^l$  space.
  - Determine the set  $D_p$  of the **hypercubes** that **contain at least one point** of  $X$ .
  - Determine the set  $D_{sp} (\subset D_p)$  that contains the “**highly populated**” cubes of  $D_p$  (that is, cubes that contain at least  $\xi_c > 1$  points of  $X$ ).
  - For each  $c \in D_{sp}$  **define** a **connection** with all **neighboring** cubes  $c_j$  in  $D_p$  for which  $d(\mathbf{m}_c, \mathbf{m}_{c_j}) \leq 4\sigma$ , where  $\mathbf{m}_c, \mathbf{m}_{c_j}$  are the means of  $c$  and  $c_j$ , respectively.
- Main stage
  - Determine the set  $D_r$  that contains:
    - the **highly populated cubes** and
    - the **cubes** that have at **least one connection** with a **highly populated** cube.



# Density-based algorithms for large data sets

## DENCLUE algorithm (cont.)

- Main stage (cont.)
  - **For** each point  $\mathbf{x}$  in a cube  $c \in D_r$ 
    - **Determine**  $Y(\mathbf{x})$  as the set of points that belong to cubes  $c_j$  in  $D_r$  such that the mean values of  $c_j$ 's lie at distance **less** than  $\lambda \cdot \sigma$  from  $\mathbf{x}$  (typically  $\lambda = 4$ ).
    - **Apply** a **hill climbing** method on  $f^X(\mathbf{x}) = \sum_{x_i \in Y(\mathbf{x})} f^{x_i}(\mathbf{x})$  **starting** from  $\mathbf{x}$  and let  $\mathbf{x}^*$  be the **local maximum** to which the method converges.
    - **If**  $\mathbf{x}^*$  is a **significant local maximum** ( $f^X(\mathbf{x}^*) \geq \xi$ ) then
      - If a cluster  $C$  **associated** with  $\mathbf{x}^*$  has already been created then
        - o  $\mathbf{x}$  is **assigned** to  $C$
      - Else
        - o **Create** a cluster  $C$  **associated** with  $\mathbf{x}^*$
        - o **Assign**  $\mathbf{x}$  to  $C$
      - End if
    - **End if**
  - **End for**



# Density-based algorithms for large data sets

## The DENCLUE Algorithm (cont.)

### Remarks:

- **Shortcuts allow** the **assignment** of **points to clusters**, **without** having to **apply** the **hill-climbing** procedure.
- **DENCLUE** is able to **detect arbitrarily shaped** clusters.
- The algorithm **deals** with **noise** very satisfactory.
- The **worst-case time complexity** of DENCLUE is  $O(N \log_2 N)$ .
- Experimental results indicate that the **average time complexity** is  $O(\log_2 N)$ .
- It **works efficiently** with **high-dimensional data**.

# Spectral clustering

**Spectral clustering** is **based** on **graph theory** concepts.

**Rationale:** It actually maps the data from their original space, where they may form arbitrarily-shaped clusters, to a new space, where (their images) form compact clusters.

## Main stages:

- **Definition** of a **similarity graph  $G$**  based on the given data set  $X$ .
- **Utilization** of the **Laplacian matrix  $L$**  associated with  $G$ .
- **Mapping** of the **data set to a space** spanned by **some** eigenvectors of  $L$ .
- **Performing** clustering on the images of the data in the transformed space.

In principle, **spectral clustering** is able to **recover arbitrarily shaped** clusters (see discussion later).

# Spectral clustering

## Similarity graph

- Data set  $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$
- Similarity graph  $G = (V, E)$

We consider **only**  
undirected graphs.

$$X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$$

$$V = \{v_1, v_2, \dots, v_N\}$$

## Definition of a similarity graph

### About $V$

- The set  $V$  consists of  $N$  vertices/nodes,  $v_1, v_2, \dots, v_N$
- Each vertex  $v_i \in V$  corresponds to a  $\mathbf{x}_i \in X, i = 1, \dots, N$ .

### About $E$

Various scenarios lead to various graphs:

(a) The  $\varepsilon$ -neighborhood graph:

- An edge  $e_{ij}$  is **added** between vertices  $v_i$  and  $v_j$ , if  $d(\mathbf{x}_i, \mathbf{x}_j) < \varepsilon$ .
- Usually it is **considered** as an **unweighted graph** (it is  $w_{ij} = 1$ , for all  $e_{ij}$ 's).

By convention,  
 $w_{ij} = 0$ , implies  
absence of  $e_{ij}$ .



# Spectral clustering

## Similarity graph

### Definition of a similarity graph

#### About $E$

(b) The  **$k$ -nearest neighbor** graph:

- An edge  $e_{ij}$  is **added** between vertices  $v_i$  and  $v_j$ , **if**  $v_i$  is **among** the  **$k$ -nearest neighbors** of  $v_j$  **OR vice versa**.
- Each  $e_{ij}$  is **weighted** by the **similarity** between  $x_i$  and  $x_j$ .

(c) The **mutual  $k$ -nearest neighbor** graph:

- An edge  $e_{ij}$  is **added** between vertices  $v_i$  and  $v_j$ , **if**  $v_i$  is **among** the  **$k$ -nearest neighbors** of  $v_j$  **AND vice versa**.
- Each  $e_{ij}$  is **weighted** by the **similarity** between  $x_i$  and  $x_j$ .

(d) The **fully connected** graph:

- All possible edges  $e_{ij}$  are added in the graph.
- Each  $e_{ij}$  is **weighted** by the **similarity** between  $x_i$  and  $x_j$ , e.g.,

$$s(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

# Spectral clustering

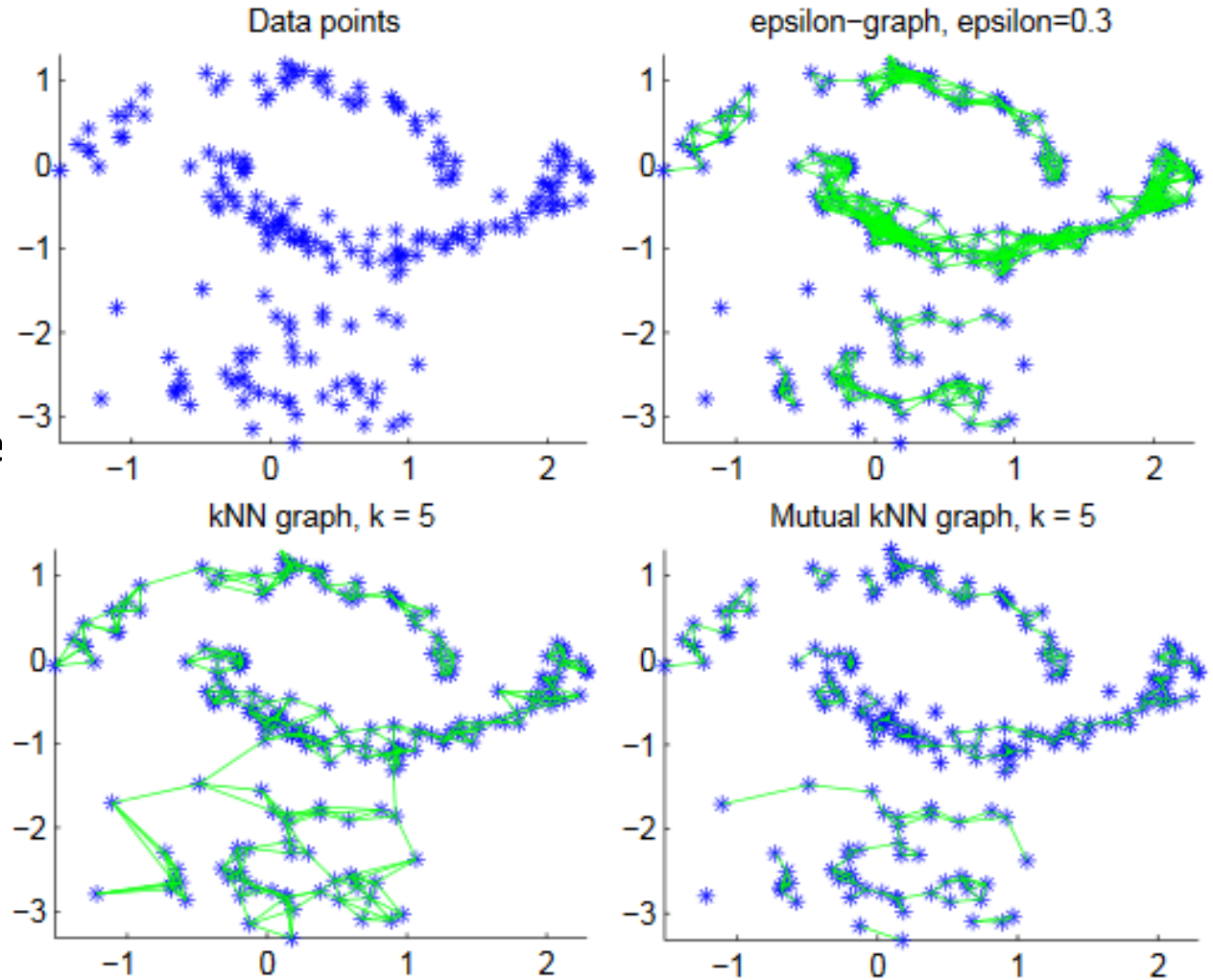
## Similarity graph

### Example:

The data set consists of

- (i) two “half moon” clusters and
- (ii) a compact cluster of different density from the previous ones.

The resulting graphs are shown in the figure.



# Spectral clustering

## Graph Laplacians

- There are **various definitions** for **graph Laplacian matrix**.
- All such matrices share some properties that allow their exploitation in the frame of clustering.

### Some definitions:

- Weighted adjacency matrix:

$$W = [w_{ij}]_{N \times N}$$

$w_{ij}$  is the **weight** of the edge connecting  $v_i$  and  $v_j$ .

- **Degree** of a vertex  $v_i$ :

$$d_i = \sum_{j=1}^N w_{ij}, \quad i = 1, \dots, N$$

- **Degree matrix**:

$$D_{N \times N} = \text{diag}(d_1, d_2, \dots, d_N) = \begin{bmatrix} d_1 & \cdots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & d_N \end{bmatrix}_{N \times N}$$

- **(Unnormalized) graph Laplacian matrix**:

$$L_{N \times N} = D - W$$

# Spectral clustering

## Graph Laplacians

Some results for the unnormalized graph Laplacian  $L$ :

1.  $\forall \mathbf{x} = [x_1, \dots, x_N]^T \in \mathbb{R}^N$  it is

$$\mathbf{x}^T L \mathbf{x} = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N w_{ij} (x_i - x_j)^2$$

2.  $L$  is **symmetric** and **positive semidefinite**.

3. The **smallest eigenvalue** of  $L$  is **0**.

4.  $L$  has  $N$  **non-negative real-valued eigenvalues**  $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_N$ .

5. Let  $G$  be an **undirected graph** with **nonnegative weights**. Then the **multiplicity  $k$**  of the **zero eigenvalue** **equals** to the **number** of the **connected components**  $A_1, \dots, A_k$ , of the graph. In addition, the **eigenspace** of the **zero eigenvalues** is **spanned** by the ( $N$ -dimensional) **indicator vectors** of those components,  $\mathbf{1}_{A_1}, \dots, \mathbf{1}_{A_k}$ .

The indicator vector  $\mathbf{1}_{A_i}$  has **all** of its **components** equal **0** **except** those corresponding to the **points** that **belong** to the  **$k$ -th connected component**., which are **equal** to **1**.

# Spectral clustering

**Graph Laplacians:** Some results for the unnormalized graph Laplacian  $L$ :

5. Let  $G$  be an **undirected graph** with **nonnegative weights** ( $w_{ij} \geq 0$ ). Then the **multiplicity**  $k$  of the **zero eigenvalue** **equals** to the **number** of the **connected components**  $A_1, \dots, A_k$ , of the graph. In addition, the **eigenspace** of the **zero eigenvalue** is **spanned** by the ( $N$ -dimensional) indicator vectors of those components,  $\mathbf{1}_{A_1}, \dots, \mathbf{1}_{A_k}$ .

- The  $k = 1$  case (**connected graph**): It is

$$d_i = \sum_{j=1}^N w_{ij}, w_{ii} = 0$$

$$0 = |L - \lambda I| = \begin{vmatrix} d_1 - \lambda & -w_{12} & \cdots & -w_{1N} \\ -w_{12} & d_2 - \lambda & \cdots & -w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -w_{1N} & -w_{2N} & \cdots & d_N - \lambda \end{vmatrix} = \begin{vmatrix} -\lambda & -w_{12} & \cdots & -w_{1N} \\ -\lambda & d_2 - \lambda & \cdots & -w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -\lambda & -w_{2N} & \cdots & d_N - \lambda \end{vmatrix} =$$

$$-\lambda \begin{vmatrix} 1 & -w_{12} & \cdots & -w_{1N} \\ 1 & d_2 - \lambda & \cdots & -w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & -w_{2N} & \cdots & d_N - \lambda \end{vmatrix} = -\lambda \begin{vmatrix} 1 & -w_{12} & \cdots & -w_{1N} \\ 0 & d_2 + w_{12} - \lambda & \cdots & -w_{2N} + w_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & -w_{2N} + w_{12} & \cdots & d_N + w_{1N} - \lambda \end{vmatrix}$$

$$= -\lambda \begin{vmatrix} d_2 + w_{12} - \lambda & \cdots & -w_{2N} + w_{1N} \\ \vdots & \ddots & \vdots \\ -w_{2N} + w_{12} & \cdots & d_N + w_{1N} - \lambda \end{vmatrix} \Leftrightarrow \lambda_1 = 0, (\lambda_2, \dots, \lambda_N > 0)$$

Thus, **multiplicity** of the **zero eigenvalue** is **1**.

The associated **eigenvector** is the **1**, since  $\mathbf{0} = 0 \cdot \mathbf{1} = L \cdot \mathbf{1}$

$$d_i = \sum_{j=1}^N w_{ij}$$

# Spectral clustering

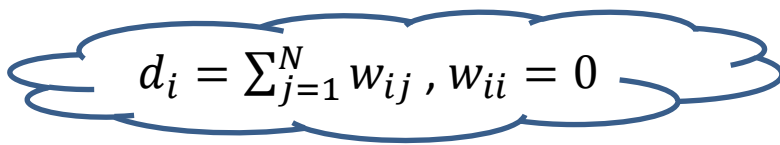
**Graph Laplacians:** Some results for the unnormalized graph Laplacian  $L$ :

5. Let  $G$  be an **undirected graph** with **nonnegative weights** ( $w_{ij} \geq 0$ ). Then the **multiplicity**  $k$  of the **zero eigenvalue** **equals** to the **number** of the **connected components**  $A_1, \dots, A_k$ , of the graph. In addition, the **eigenspace** of the **zero eigenvalue** is **spanned** by the ( $N$ -dimensional) indicator vectors of those components,  $\mathbf{1}_{A_1}, \dots, \mathbf{1}_{A_k}$ .

- The  $k = 1$  case (**connected graph**):

- The associated **eigenvector** is the  $\mathbf{1}$ , since  $\mathbf{0} = 0 \cdot \mathbf{1} = L \cdot \mathbf{1}$

$$\mathbf{0} = 0 \cdot \mathbf{1} = 0 \cdot \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} d_1 & -w_{12} & \cdots & -w_{1N} \\ -w_{12} & d_2 & \cdots & -w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -w_{1N} & -w_{2N} & \cdots & d_N \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$


$$d_i = \sum_{j=1}^N w_{ij}, w_{ii} = 0$$

# Spectral clustering

**Graph Laplacians:** Some results for the unnormalized graph Laplacian  $L$ :

5. Let  $G$  be an undirected graph with nonnegative weights ( $w_{ij} \geq 0$ ). Then the multiplicity  $k$  of the zero eigenvalue equals to the number of the connected components  $A_1, \dots, A_k$ , of the graph. In addition, the eigenspace of the zero eigenvalue is spanned by the ( $N$ -dimensional) indicator vectors of those components,  $\mathbf{1}_{A_1}, \dots, \mathbf{1}_{A_k}$ .

- The  $k > 1$  case ( $k$  connected components):

- Considering each connected component individually, the  $i$ -th component has its own associated Laplacian  $L_i$
- Then the Laplacian for the whole graph can be written as

$$L = \begin{bmatrix} L_1 & & \\ & \ddots & \\ & & L_k \end{bmatrix}$$

The spectrum of  $L$  is given by the union of the spectra of  $L_i$ 's.

- Since, the multiplicity of the zero eigenvalue is 1 for each  $L_i \Rightarrow$  the multiplicity of the zero eigenvalue is  $k$  for  $L$ .
- Denoting  $|A_1| = n_1$ ,  $\mathbf{1}_{A_1}$  has its first  $n_1$  (resp. remaining) components equal to 1 (resp. 0),  $\mathbf{1}_{A_1} = [1, 1, \dots, 1, 0, 0, \dots, 0]^T$ . Then,

$$\mathbf{0}_{n_1 \times 1} = 0 \cdot \mathbf{1}_{n_1 \times 1} = L_1 \cdot \mathbf{1}_{n_1 \times 1} \Rightarrow \mathbf{0}_{N \times 1} = 0 \cdot \mathbf{1}_{A_1, N \times 1} = L \cdot \mathbf{1}_{N \times 1}$$

# Spectral clustering

## Unnormalized spectral clustering algorithm

**Input:** (a) Similarity matrix  $S \in R^{N \times N}$ , (b) the number of clusters  $m$

- **Construct** a **similarity graph** with weighed adjacency matrix  $W$ .
- **Compute** the unnormalized Laplacian  $L$ .
- **Compute** the first  $m$  (column) **eigenvectors** of  $L$ ,  $\mathbf{u}_1, \dots, \mathbf{u}_m$ .
- **Stack**  $\mathbf{u}_1, \dots, \mathbf{u}_m$  on an  $N \times m$  matrix  $U$ .
- **Represent** each data vector  $\mathbf{x}_i$  by the  $i$ -th row  $\mathbf{y}_i$  of  $U$ .
- **Cluster** the points  $\mathbf{y}_i \in R^m$ ,  $i = 1, \dots, N$ , using e.g., the  **$k$ -means** algorithm, into clusters  $C_1', C_2', \dots, C_m'$ .

**Output:** Clusters  $C_1, C_2, \dots, C_m$ , with  $C_i = \{\mathbf{x}_j: \mathbf{y}_j \in C_i'\}$



# Spectral clustering

## Unnormalized spectral clustering algorithm

### Example:

Data set  $X = \{x_1, x_2, x_3, x_4, x_5\}$

Similarity graph:

$$G = (V, E) = (\{v_1, v_2, v_3, v_4, v_5\}, \{e_{13}, e_{24}, e_{25}, e_{45}\})$$

Nodes degree:  $d_1 = w_{13}$ ,  $d_2 = w_{24} + w_{25}$ ,  $d_3 = w_{13}$

$$d_4 = w_{24} + w_{45}, \quad d_5 = w_{25} + w_{45}$$

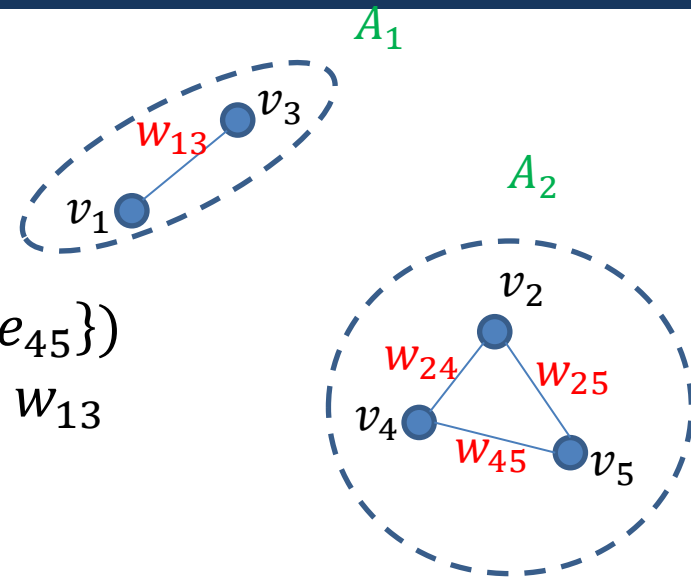
Laplacian of the whole graph:

$$L = D - W$$

$$= \begin{bmatrix} w_{13} & 0 & -w_{13} & 0 & 0 \\ 0 & w_{24} + w_{25} & 0 & -w_{24} & -w_{25} \\ -w_{13} & 0 & w_{13} & 0 & 0 \\ 0 & -w_{24} & 0 & w_{24} + w_{45} & -w_{45} \\ 0 & -w_{25} & 0 & -w_{45} & w_{25} + w_{45} \end{bmatrix}$$

$$|L - \lambda I| = \dots = \lambda^2 \begin{vmatrix} 2w_{13} - \lambda & 0 & 0 \\ 0 & 2w_{24} + w_{45} - \lambda & w_{25} - w_{45} \\ 0 & w_{24} - w_{45} & 2w_{25} + w_{45} - \lambda \end{vmatrix} = 0 \Leftrightarrow$$

$\lambda = 0$  double root



# Spectral clustering

## Unnormalized spectral clustering algorithm

### Example:

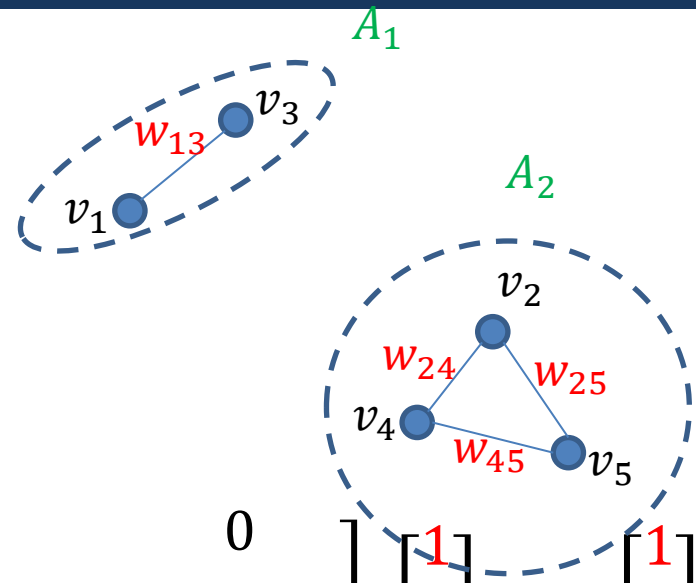
Data set  $X = \{x_1, x_2, x_3, x_4, x_5\}$

Corresponding eigenvectors  $e$  ( $L \cdot e = 0 \cdot e$ ):

$u_1 = [1, 0, 1, 0, 0]^T$  and  $u_2 = [0, 1, 0, 1, 1]^T$  since

$$\begin{bmatrix} w_{13} & 0 & -w_{13} & 0 & 0 \\ 0 & w_{24} + w_{25} & 0 & -w_{24} & -w_{25} \\ -w_{13} & 0 & w_{13} & 0 & 0 \\ 0 & -w_{24} & 0 & w_{24} + w_{45} & -w_{45} \\ 0 & -w_{25} & 0 & -w_{45} & w_{25} + w_{45} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = 0 \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} w_{13} & 0 & -w_{13} & 0 & 0 \\ 0 & w_{24} + w_{25} & 0 & -w_{24} & -w_{25} \\ -w_{13} & 0 & w_{13} & 0 & 0 \\ 0 & -w_{24} & 0 & w_{24} + w_{45} & -w_{45} \\ 0 & -w_{25} & 0 & -w_{45} & w_{25} + w_{45} \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} = 0 \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$



# Spectral clustering

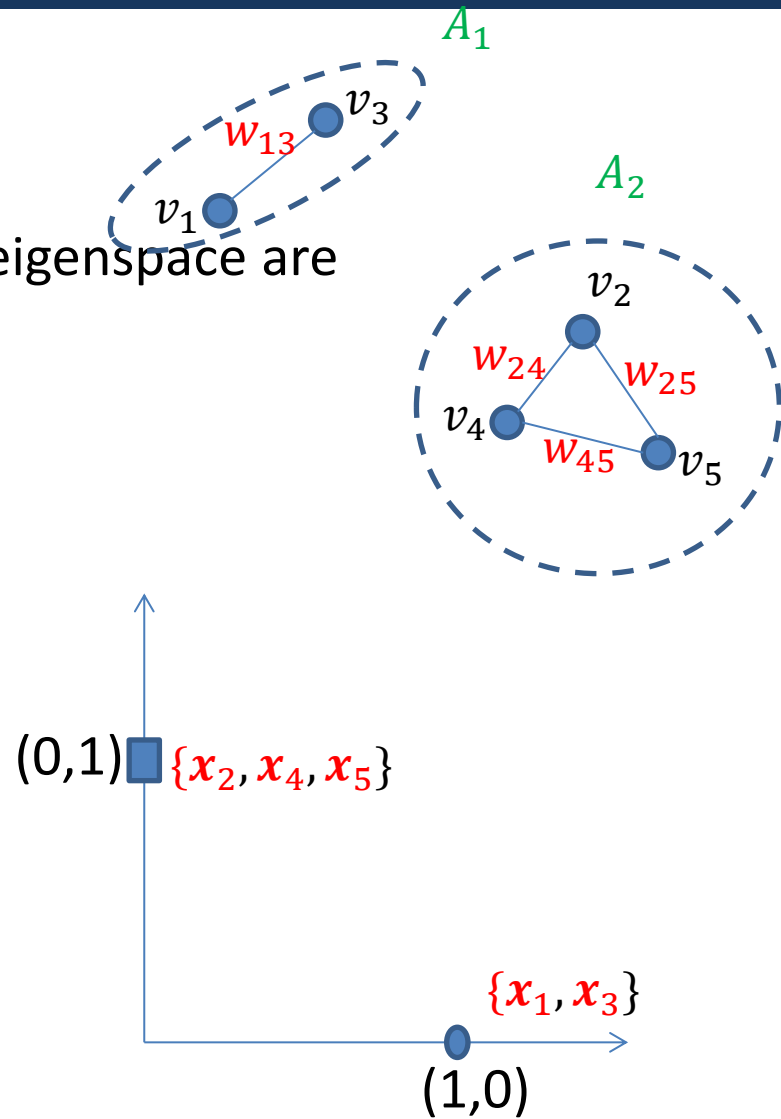
## Unnormalized spectral clustering algorithm

### Example:

The eigenvectors corresponding to the zero eigenspace are  $\mathbf{u}_1 = [1, 0, 1, 0, 0]^T$  and  $\mathbf{u}_2 = [0, 1, 0, 1, 1]^T$

The matrix  $U =$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \equiv \begin{matrix} \mathbf{y}_1 \rightarrow \mathbf{x}_1 \\ \mathbf{y}_2 \rightarrow \mathbf{x}_2 \\ \mathbf{y}_3 \rightarrow \mathbf{x}_3 \\ \mathbf{y}_4 \rightarrow \mathbf{x}_4 \\ \mathbf{y}_5 \rightarrow \mathbf{x}_5 \end{matrix}$$



# Spectral clustering

## Other Laplacian matrices

- **Symmetric** Laplacian matrix:  $L_{sym} = D^{-1/2} \cdot L \cdot D^{-1/2}$
- **Random walk** Laplacian matrix:  $L_{rw} = D^{-1} \cdot L$

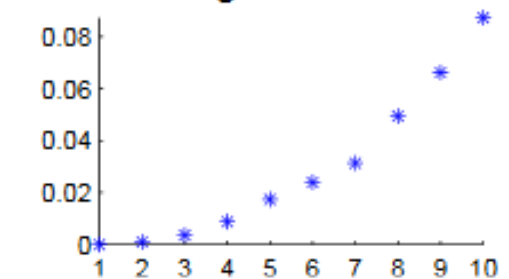
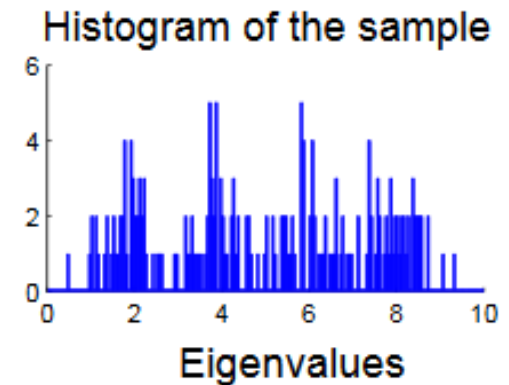
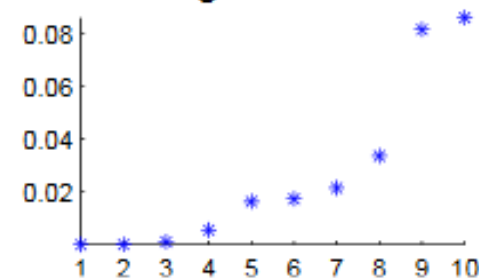
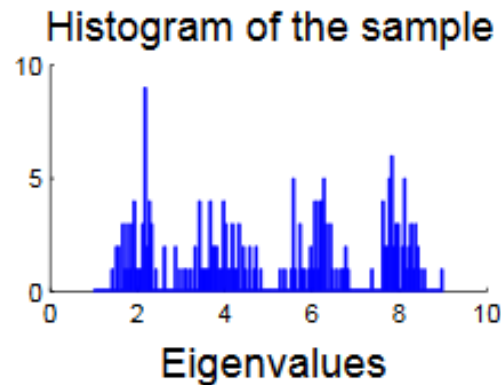
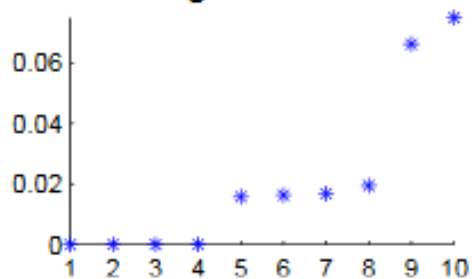
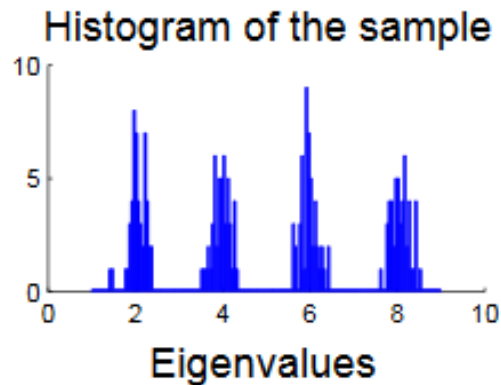
All Laplacians share similar properties concerning the zero eigenvalue. In (von Luxburg, 2007), it is suggested to use  $L_{rw}$ .

# Spectral clustering

## Choice of the number of clusters

### Example:

The ten smallest eigenvalues of  $L_{rw}$  for a 1-dim. four-clusters problem.



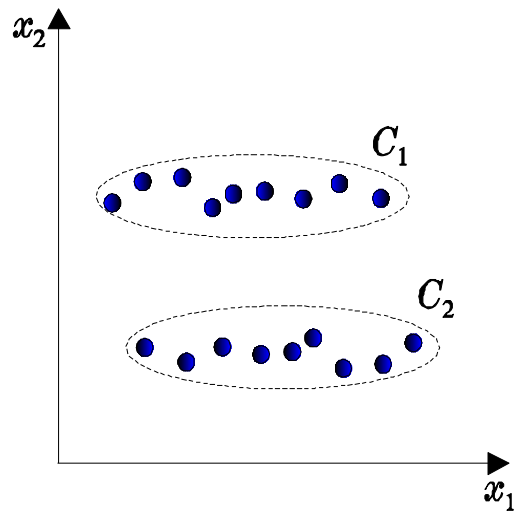
In the case where  $m$  is **not a priori known**, it can be estimated by **sorting** the **Laplacian eigenvalues** and **determining** the number of the **first  $m$  eigenvalues** that **(a)** are sufficiently close to 0 and **(b)** the  $m + 1$  differs significantly from them.

# Clustering algorithms for high dimensional data sets

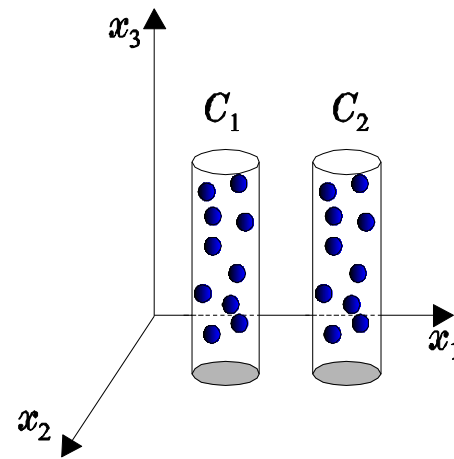
- What is a **high-dimensionality space**?  
Dimensionality  $l$  of the input space with  
$$20 \leq l \leq \text{few thousands}$$
indicate **high-dimensional data sets**.
- **Problems** of considering simultaneously **all dimensions** in high-dimensional data sets:
  - **“Curse of dimensionality”**. As a fixed number of points spread out in high-dimensional spaces, they become almost equidistant (that is, the terms **similarity and dissimilarity tend to become meaningless – alternatively, no clear structures are defined**).
  - **Several dimensions may be irrelevant to the identification of the clusters** (that is, the clusters usually are identified in **subspaces** of the original feature space).
- A way out: **Work on *subspaces of dimension lower than  $l$*** .
  - Main approaches:
    - ❑ **Dimensionality reduction** clustering approach.
    - ❑ **Subspace clustering** approach.

# Clustering algorithms for high dimensional data sets

An example:



(a)



(b)

# Clustering algorithms for high dimensional data sets

## Dimensionality Reduction Clustering Approach

### Main idea

- **Identify** an **appropriate**  $l'$ -dimensional space  $H_{l'}$  ( $l' < l$ ).
- **Project** the **data points** of  $X$  into  $H_{l'}$ .
- **Apply** a **clustering** algorithm **on the projections** of the points of  $X$  into  $H_{l'}$ .

**Identification** of  $H_{l'}$  may be carried out using either by:

- **Feature generation** methods,
- **Feature selection** methods,
- **Random projections**.



# Clustering algorithms for high dimensional data sets

## Dimensionality Reduction Clustering Approach (cont.)

### Feature generation methods

- They produce **new features** via suitable **transformations** applied on the **original ones**.
- Typical Methods in this category are:
  - Principal component analysis (PCA). Singular value decomposition (SVD).
  - Nonlinear PCA Robust PCA Independent comp. analysis (ICA).
- In general, **PCA** and **SVD** methods
  - **preserve the distances** between the points in the high-dimensional space, when these are mapped to the lower-dimensional space.
  - **produce compact representations** (with reduced number of features) of the original high-dimensional feature space.
- In some cases **feature generation** is **applied iteratively** in cooperation with a clustering algorithm (*k*-means, EM).
- They are **useful** in cases where a **significant** number of features contributes to the identification of all physical clusters.
- They are **useful** when **all clusters** are **formed** in the **same subspace** of the feature space.

# Principal Component Analysis (PCA)

## Principal component analysis (PCA):

It transforms the **original space** to a **new orthogonal space** (of the same **dimensionality**) where the **features** are **uncorrelated**. Specifically: **along** the, so called, **1<sup>st</sup> principal axis** the **maximum possible variance** of the data set is **retained**, along the **2<sup>nd</sup>** one the **maximum possible remained** variance is **retained** etc.

Projecting on the **first few principal axes space** we achieve **dimensionality reduction**.

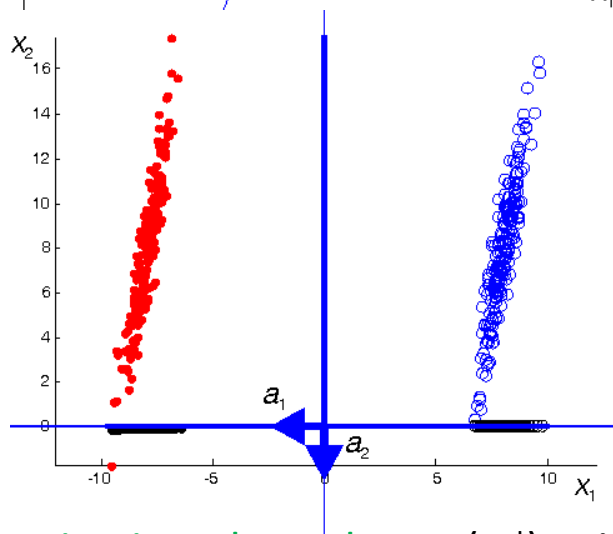
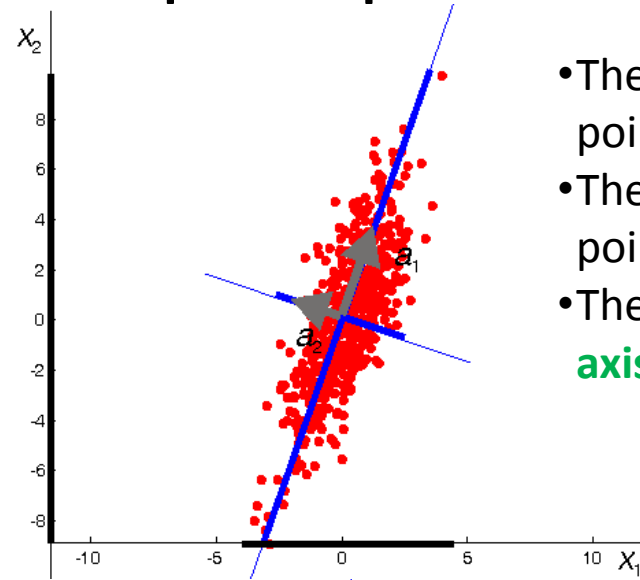


# Principal Component Analysis - PCA

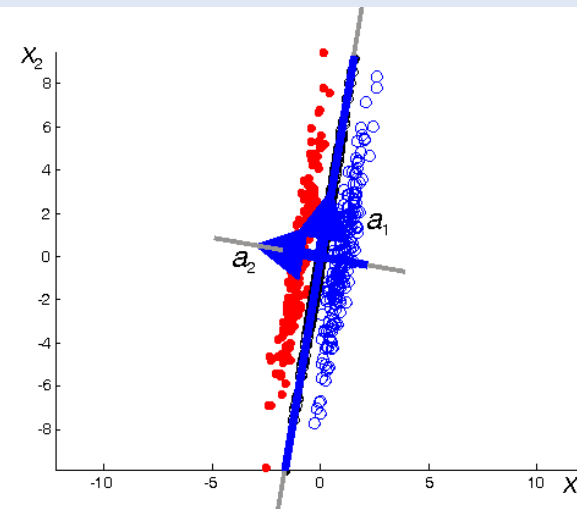
## Principal Component Analysis (PCA)

- The **black lines** show the **range of values** of the data points along the **initial axes**.
- The **blue lines** show the **range of values** of the data points along the **principal axes**.
- The **widest range** of values is along the **first principal axis**.

**CAUTION:** Retaining the **maximum possible variance** of the data set **DOES NOT** imply that we necessarily **retain the cluster separability**.



Projection along the  $a_1$  (1<sup>st</sup>) principal direction retains cluster separability.



Projection along the  $a_1$  principal direction **DOES NOT** retain cluster separability.

# Subspace clustering

- **Solution:** Principal component analysis (**PCA**)

- Let  $X_{l \times N} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_N]$  and  $Y_{l' \times N} = [\mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_N]$

- Compute  $\boldsymbol{\mu}_{l \times 1} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$

$$S = \{x \in R^l : x = \mu + B \cdot y\}$$

- Consider  $X'_{l \times N} = [\mathbf{x}_1 - \boldsymbol{\mu} \quad \mathbf{x}_2 - \boldsymbol{\mu} \quad \cdots \quad \mathbf{x}_N - \boldsymbol{\mu}]$

- Perform singular value decomposition (**SVD**) on  $X'$  taking

$$X'_{l \times N} = U'_{l \times l} \cdot \Sigma'_{l \times N} \cdot V'^T_{N \times N}$$

- Keep the **first**  $l'$  singular values (as a consequence take also (a) the first  $l'$  columns of  $U'$  and (b) the first  $l'$  columns of  $V'$  ( $\Leftrightarrow$  the first  $l'$  rows of  $V'^T$ ) and approximate  $X'$  as

$$X'^{appr}_{l \times N} = U_{l \times l'} \cdot \Sigma_{l' \times l'} \cdot V^T_{l' \times N}$$

-  $B = U_{l \times l'}$  is the **subspace basis** and

-  $Y_{l' \times N} = \Sigma_{l' \times l'} \cdot V^T_{l' \times N}$  contains (in columns) the **representations/projections** of the (shifted by  $\boldsymbol{\mu}$ ) original data in the **lower  $l'$ -dim. space**.

**Theorem:**  $X'^{appr}$ , as computed before, is the **best approximation** of  $X'$  wrt the **Frobenius norm**, subject to the **constraint** that the **rank** of  $X'^{appr}$  is  $l'$ .

$$\|X - X'\| = \sqrt{\sum_{i=1}^l \sum_{j=1}^N (x_{ij} - x'_{ij})^2}$$

# Subspace clustering

## More on SVD

Let  $X'_{l \times N} = [\mathbf{x}_1 - \boldsymbol{\mu} \quad \mathbf{x}_2 - \boldsymbol{\mu} \quad \cdots \quad \mathbf{x}_N - \boldsymbol{\mu}]$ , with  $\boldsymbol{\mu}_{l \times 1} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$

In the expression  $X'_{l \times N} = U'_{l \times l} \cdot \Sigma'_{l \times N} \cdot V'^T_{N \times N}$

$\Sigma'_{l \times N}$  (diagonal matrix) contains the **singular values** of  $X'_{l \times N}$  in decreasing order in its main diagonal ( $l < N$ )

$U'_{l \times l}$  contains in its columns the **eigenvectors** of  $X'X'^T_{l \times l}$

$V'_{N \times N}$  contains in its columns the **eigenvectors** of  $X'^T X'_{N \times N}$

Let

—  $U' = [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_l]$  ( $\mathbf{u}_i$ 's are  $l$ -dimensional **column** vectors)

—  $V' = [\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_N] \Rightarrow V'^T = \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_N^T \end{bmatrix}$  ( $\mathbf{v}_i$ 's are  $N$ -dimensional **column**

vectors and  $\mathbf{v}_i^T$ 's are  $N$ -dimensional **row** vectors)

—  $\Sigma'_{l \times N} = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \sigma_l & \vdots & 0 \end{bmatrix}$

# Subspace clustering

## More on SVD

Then

$$\begin{aligned} X'_{l \times N} &= U'_{l \times l} \cdot \Sigma'_{l \times N} \cdot V'^T_{N \times N} \\ &= [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_l] \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \sigma_l & \vdots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_N^T \end{bmatrix} \\ &= [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_l] \begin{bmatrix} \sigma_1 \mathbf{v}_1^T \\ \sigma_2 \mathbf{v}_2^T \\ \vdots \\ \sigma_l \mathbf{v}_l^T \end{bmatrix} = \end{aligned}$$

$$\sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_l \mathbf{u}_l \mathbf{v}_l^T = \sum_{i=1}^l \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

Thus,  $X'$  is expressed as a **sum of rank one** matrices  $\mathbf{u}_i \mathbf{v}_i^T$  each one **weighted** by its corresponding  $\sigma_i$ .

By **neglecting** the **terms** with “**small**”  $\sigma_i$ 's, we actually perform **dimensionality reduction**, or, in other words, we **determine** the **subspace** where the **data** “**actually live**”.

# Clustering algorithms for high dimensional data sets

## Dimensionality Reduction Clustering Approach (cont.)

### Feature selection methods

- They identify the **original features** that are the main contributors to the formation of the clusters.
- The **criteria** used to **evaluate** the “goodness” of a specific **subset of features** follow either the
  - **Wrapper model** (The clustering algorithm is first chosen and a set of features  $F_i$  is evaluated through the results obtained from the application of the algorithm to  $X$ , where for each point **only the features** in  $F_i$  are taken into account).
  - **Filter model** (The evaluation of a subset of features is carried out using **intrinsic** properties of the data, prior to the application of the clustering algorithm).
- They are **useful** when **all clusters** are **formed** in the **same subspace** of the feature space.

# Clustering algorithms for high dimensional data sets

## Dimensionality Reduction Clustering Approach (cont.)

### Clustering using Random Projections:

Here  $H_{l'}$  is identified in a **random manner**.

**Note:** The **projection** of an  $l$ -dimensional space to an  $l'$ -dimensional space ( $l' < l$ ) is **uniquely defined** via an  $l' \times l$  projection matrix  $A$ .

### Issues to be addressed:

(a) **Proper estimate** of  $l'$ . Estimates of  $l'$  **guarantee** (in probability) that the **distances** between the points of  $X$ , in the **original** data space will be **preserved** (with some distortion) **after** the **projection** to a **randomly** chosen  $l'$ -dim. space, whose **projection matrix** is **constructed** via certain **probabilistic rules**

**Note:** Preservation of **distances** does not necessarily **preserves clusters**.

(b) **Definition** of the projection matrix  $A$ . Possible **rules** for constructing  $A$  are:

1. **Set** each **entry** of  $A$  equal to a value stemming from an **i.i.d. zero mean, unit variance Gaussian** distribution and then **normalize** each **row** to the **unit length**.
2. **Set** each **entry** of  $A$  equal to  $-1$  or  $+1$ , with **probability 0.5**.
3. **Set** each **entry** of  $A$  equal to  $+\sqrt{3}$ ,  $-\sqrt{3}$  or  $0$ , with **probs**  $\frac{1}{6}$ ,  $\frac{1}{6}$  and  $\frac{2}{3}$ , resp.



# Clustering algorithms for high dimensional data sets

## Dimensionality Reduction Clustering Approach (cont.)

Having defined  $A$ :

- **Project** the **points** of  $X$  into  $H_{l'}$
- **Perform** a **clustering algorithm** on the **projections** of the points of  $X$  into  $H_{l'}$ .

**Problem:** **Different random projections may lead to totally different results.**

**Solution:**

- **Perform** **several** random projections  $H_{l'}$ .
- **Apply** a **clustering algorithm** on the **projections** of  $X$  to each  $H_{l'}$ .
- **Combine** the clustering **results** and **produce** the **final clustering**.

A method in the above spirit is described next ( $O(N^2)$ ).

# Clustering algorithms for high dimensional data sets

## Clustering using Random Projections

- **Select**  $l'$ .
- **Generate**  $A_1, \dots, A_r$  different **projection matrices** using the (b.1) rule given above.
- **For**  $s = 1$  to  $r$ 
  - **Run** GPrAS with **normal pdfs** for the  $s$ -th random projection of  $X$ .
  - **Compute** the probability that  $\mathbf{x}_i$  **belongs** to the  $j$ -th **cluster** in the  $s$ -th **projection**,  $P(C_j^s | \mathbf{x}_i)$ ,  $i = 1, \dots, N, j = 1, \dots, m_s$ .
  - **Create** the similarity matrix  $P^s = [P_{ij}^s]$ , where  $P_{ij}^s$  is the **probability** that  $\mathbf{x}_i$  and  $\mathbf{x}_j$  **belong** to the **same cluster**,

$$P_{ij}^s = \sum_{q=1}^{m_s} P(C_q^s | \mathbf{x}_i) P(C_q^s | \mathbf{x}_j)$$

$m_s$ : number of clusters in the  $s$ -th projection.

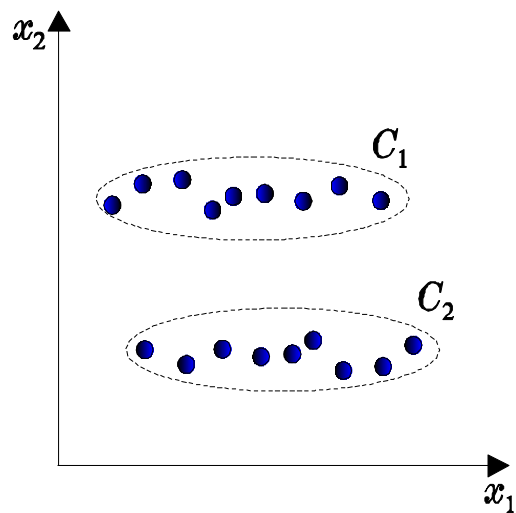
- **End for**
- **Compute** the **average** proximity matrix  $P = [P_{ij}]$ , so that  $P_{ij}$  is the **average** of  $P_{ij}^s$ 's,  $s = 1, \dots, r$ .
- **Apply** GAS (actually its complete link version) on  $P$ .
- **Plot** the **similarity** between the **closest pair of clusters** at each iteration **versus** the **number of iterations**.
- **Select** the **clustering** that **corresponds** to the **most abrupt change** in the plot.

# Clustering algorithms for high dimensional data sets

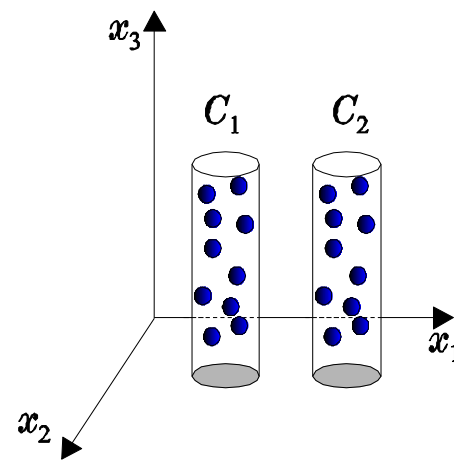
## Subspace Clustering Approach

- This approach deals with the problem where **clusters** are **formed** in **different subspaces** of the feature space.
- The subspace clustering algorithms (**SCA**) **reveal clusters** as well as the **subspaces** where they reside.

### An example:



(a)

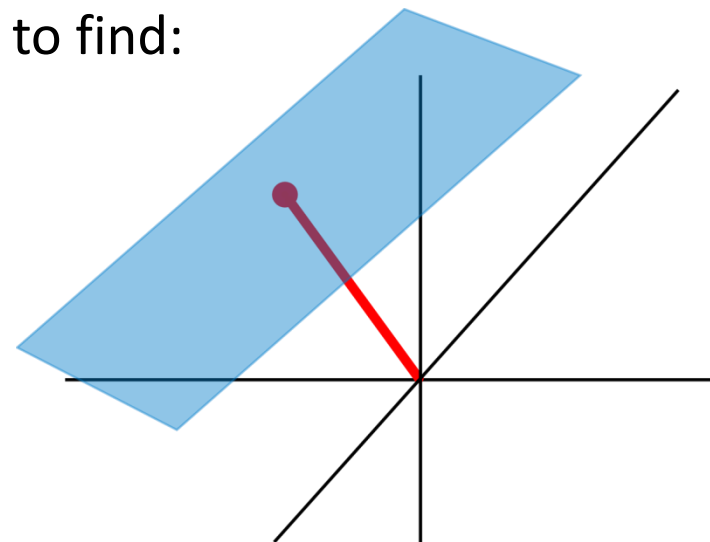


(b)

# Subspace clustering

## Preliminaries:

- The data set  $X = \{\mathbf{x}_i \in R^l, i = 1, \dots, N\}$
- (**Affine linear**) **Subspace**  $S$  of  $R^l$ : It is defined via
  - a vector  $\boldsymbol{\mu}$  in  $S$  and
  - an  $l \times l'$  (basis) matrix  $B$  ( $l' < l$ )as  $S = \{\mathbf{x} \in R^l: \mathbf{x} = \boldsymbol{\mu} + B \cdot \mathbf{y}\}$ , where  $\mathbf{y} \in R^{l'}$
- **Assuming** that **all** the **data points** of  $X$  **lie** in an  $l'$ -dimensional (**affine**) **subspace**  $S$ , in order to **determine** it, we need to find:
  - A vector  $\boldsymbol{\mu} \in S$
  - The dimensionality  $l'$  of  $S$
  - The  $l \times l'$  matrix  $B$ .



# Subspace clustering

**Basic assumption:** In subspace clustering, the **clusters** formed by the data points “live” in **subspaces** of the **original**  $l$ -dimensional data space.

$$S_j = \{x \in R^l : x = \mu_j + B_j \cdot y\}$$

- **Aim of subspace clustering: Determine**

- the number of subspaces  $m$
- The **dimensionalities**  $l_1, l_2, \dots, l_m$ , of the subspaces  $S_1, S_2, \dots, S_m$
- The basis matrices  $B_1, B_2, \dots, B_m$
- The points  $\mu_1, \mu_2, \dots, \mu_m$ , of the subspaces  $S_1, S_2, \dots, S_m$ .
- The clusters  $C_1, C_2, \dots, C_m$ .

Usually, it is the case that each subspace contains a single cluster

# Subspace clustering

## Ways to tackle the problem

- Algebraic methods
- Spectral clustering methods
- Iterative cost function optimization methods (**hard**, **probabilistic** framework)

## Iterative cost function optimization methods (**hard** framework)

The ***k***-subspace algorithm

**Assumption:** The number of clusters ***m*** and the subspaces dimensionalities ***l*<sub>1</sub>, *l*<sub>2</sub>, ..., *l*<sub>*m*</sub>**, are **known**.

Let:

- $\mathbf{U}_{N \times m} = [u_{ij}]$ , where  $u_{ij} = \begin{cases} 1, & x_i \in C_j \\ 0, & \text{otherwise} \end{cases}$
- $\mathbf{B} = \{B_1, B_2, \dots, B_m\}$
- $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_m\}$
- $\mathbf{Y} = \{Y_1, \dots, Y_m\}$ , with  $Y_j = \{\mathbf{y}_i^j, i = 1, \dots, N\}$  be the set of **projections** of the data points to the ***j***-th subspace.

# Subspace clustering

Iterative CFO methods (hard framework) - The  $k$ -subspace algorithm

$\mathbf{y}_i^j$ : Projection of  $\mathbf{x}_i$  to the  $j$ -th subspace

Consider the cost function

$$J(B, \mu, Y, U) = \sum_{i=1}^N \sum_{j=1}^m u_{ij} \|\mathbf{x}_i - \boldsymbol{\mu}_j - B_j \mathbf{y}_i^j\|^2$$

This is **minimized** in a **two-stage iterative** fashion (recall  $k$ -means)

For **fixed**  $\boldsymbol{\mu}'_j$ 's,  $B'_j$ 's,  $\mathbf{y}_i^{j'}$ 's:

Define  $u_{ij} = \begin{cases} 1, & \text{if } \|\mathbf{x}_i - \boldsymbol{\mu}_j - B_j \mathbf{y}_i^j\|^2 = \min_{q=1, \dots, m} \|\mathbf{x}_i - \boldsymbol{\mu}_q - B_q \mathbf{y}_i^q\|^2 \\ 0, & \text{otherwise} \end{cases}$

For **fixed**  $u_{ij}$ 's: Solve the following  $m$  independent problems

$$\min_{\{\boldsymbol{\mu}_j, (B_j, \mathbf{y}_i^j)\}} \sum_{\mathbf{x}_i: u_{ij}=1} \|\mathbf{x}_i - \boldsymbol{\mu}_j - B_j \mathbf{y}_i^j\|^2 \equiv \min_{\{\boldsymbol{\mu}_j, (B_j, \mathbf{y}_i^j)\}} \sum_{i=1}^N u_{ij} \|\mathbf{x}_i - \boldsymbol{\mu}_j - B_j \mathbf{y}_i^j\|^2$$

For **each** such **problem**

(a) Fix  $\boldsymbol{\mu}'_j$ 's and **apply PCA**, to **estimate**  $B'_j$ 's,  $\mathbf{y}_i^{j'}$ 's .

(b) Fix  $B'_j$ 's,  $\mathbf{y}_i^{j'}$ 's and **apply the  $k$ -means rationale**, to **estimate**  $\boldsymbol{\mu}'_j$ 's.

# Subspace clustering

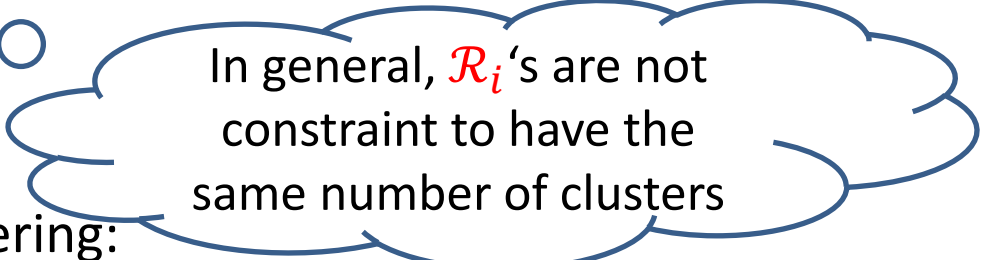
## Remark:

There are also subspace clustering methods (e.g., **CLIQUE**, **ENCLUS**) that “quantize” the region where the data belongs through the use of a grid. Then, clusters (at different subspaces) are defined through boxes that contain a significant number of data points.



# Combinations of clusterings

- The data set  $X = \{\mathbf{x}_i \in R^l, i = 1, \dots, N\}$
- **Ensemble** of clusterings of  $X$ :  $\mathcal{E} = \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_n\}$   
where  $\mathcal{R}_i = \{C_i^1, C_i^2, \dots, C_i^{m_i}\}$   
 $C_i^j$ : the  $j$ -th cluster of the  $i$ -th clustering  
 $m_i$ : the **number of clusters** in the  $i$ -th clustering.



In general,  $\mathcal{R}_i$ 's are not constraint to have the same number of clusters

- Alternative representation of a clustering:  
 $\mathcal{R}_i \leftrightarrow \mathbf{y}_i = [y_i(1), y_i(2), \dots, y_i(k), \dots, y_i(N)]$   
where  $y_i(k)$  the **cluster label** of the  $k$ -th data point.

**Example:** Let  $\mathcal{R}_i = \{C_i^1, C_i^2, C_i^3\} = \{\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_6, \mathbf{x}_{10}\}, \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_7\}, \{\mathbf{x}_5, \mathbf{x}_8, \mathbf{x}_9\}\}$   
Then  $\mathbf{y}_i = [1, 1, 2, 2, 3, 1, 2, 3, 3, 1]$ .

The two main issues in this framework are:

**(A)** The **generation** of the **ensemble of clusterings**

**(B)** The **combination** of the **clusterings**.

# Combinations of clusterings

## A. Generation of ensemble of clusterings

It involves two steps:

- (a) The **choice** of the **subspace** to **project** the data points of  $X$ .
- (b) The **application** of a **clustering algorithm** on the resulting **projections**.

## General directions:

- **All data, all features:**

- All  $l$  features and all  $N$  data points are **used**.
- Either different algorithms are **applied**
- or the same algorithm with different parameter values (e.g., in  $k$ -means, different number of cluster, or different initial conditions).

- **All data, some features:**

- All  $N$  data points are **used**.
- $n$  data sets  $X_i$  are **formed** from  $X$
- Either by **selecting** a number of features (feature distributed clustering)
- or by **projecting** onto a randomly chosen lower dimensional space.
- The same or different algorithms can be **applied** on the  $X_i$ 's.

# Combinations of clusterings

## A. Generation of ensemble of clusterings

### General directions:

- Some data, all features:

- All  $l$  features are used.
- $n$  data sets  $X_i$  are formed from  $X$  using techniques like bootstrapping and sampling.
- (Usually) the same algorithm is applied on the  $X_i$ 's.
- The points that have not been selected to participate in  $X_i$  are assigned to their nearest cluster in  $\mathcal{R}_i$ .

## B. Combination of clusterings

**Problem:** Given  $\mathcal{E} = \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_n\}$ , determine the consensus clustering  $\mathcal{F} = \{F_1, F_2, \dots, F_m\}$ .

A useful tool in this direction is the co-association matrix  $C$ .

It is an  $N \times N$  matrix  $C = [c_{ij}]$  with  $c_{ij} = \frac{n_{ij}}{n}$

where  $n_{ij}$  is the number of times where the  $i$ -th and the  $j$ -th points of  $X$  are assigned to the same cluster, among the  $n$  clusterings of  $\mathcal{E}$ .

# Combinations of clusterings

## B. Combination of clusterings

Three main directions are used:

- **Co-association matrix** based methods
- **Graph-based** methods
- **Function optimization** methods.

## Co-association matrix based methods

- **Compute** the **co-association matrix**.
- **Use it** as a **similarity matrix** and **run** a **hierarchical algorithm** (single-link, complete-link etc)
- From the produced dendrogram **determine** the **final clustering** as the one having the **larger lifetime**.

**Note:** A **large number** of **clusterings** is **required**, in order to estimate more accurately the elements of  $C$ .

# Combinations of clusterings

## B. Combination of clusterings

### Graph-based methods

- *Instance-based graph formulation (IBGF)*
- *Cluster-based graph formulation (CBGF)*
- *Hybric bipartite graph formulation (HBGF)*

# Combinations of clusterings

## B. Combination of clusterings

### Graph-based methods

- *Instance-based graph formulation (IBGF)*
  - *Cluster-based graph formulation (CBGF)*
  - *Hybrid bipartite graph formulation (HBGF)*
- 
- **Construct** a **fully connected** graph  $G = (V, E)$  where
  - Each **vertex** of  $V$  **corresponds** to a **data point** and
  - Each edge  $e_{ij}$  of  $E$  is **weighted** by  $c_{ij}$  (the  $(i, j)$  element of  $C$ ).
  - **Partition** the **graph** into  **$m$  disjoint subsets** of vertices  $V_1, V_2, \dots, V_m$  such that
    - The **sum** of **weights** of the edges that **connect vertices** between any pair of two **different subsets** is **minimized** and
    - All  $V_j$ 's have approximately the **same size**.

**Note:** The **normalized-cut** and the **Ratio-cut** criteria can be used for partitioning the graph.

# Combinations of clusterings

## B. Combination of clusterings

### Graph-based methods

- Instance-based graph formulation (IBGF)

**Example:** Consider a data set  $X = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$  and assume that the co-

association matrix is  $C = [c_{ij}] = \begin{bmatrix} 1 & 0.9 & 0.07 & 0.05 \\ 0.9 & 1 & 0.03 & 0.02 \\ 0.07 & 0.03 & 1 & 0.9 \\ 0.05 & 0.02 & 0.9 & 1 \end{bmatrix}$   $C$  indicates that the physical clusters are  $C_1 = \{\mathbf{x}_1, \mathbf{x}_2\}$ ,  $C_2 = \{\mathbf{x}_3, \mathbf{x}_4\}$ .

Consider the **fully connected graph** with **four** vertices  $\mathbf{v}_1(\mathbf{x}_1)$ ,  $\mathbf{v}_2(\mathbf{x}_2)$ ,  $\mathbf{v}_3(\mathbf{x}_3)$ ,  $\mathbf{v}_4(\mathbf{x}_4)$ , with the weight of each edge  $w_{ij}$  being equal to  $c_{ij}$ .

For the possible (equally-sized clusters) two-clusters graph partitions it is:

Partition	Edges connecting diff. clusters (weights)	Total weight of connecting edges
$\{\{v_1, v_2\}, \{v_3, v_4\}\}$	$e_{13}(0.07), e_{14}(0.05), e_{23}(0.03), e_{24}(0.02)$	0.17(*)
$\{\{v_1, v_3\}, \{v_2, v_4\}\}$	$e_{12}(0.9), e_{14}(0.05), e_{32}(0.02), e_{34}(0.9)$	1.87
$\{\{v_1, v_4\}, \{v_2, v_3\}\}$	$e_{12}(0.9), e_{13}(0.07), e_{42}(0.02), e_{43}(0.9)$	1.87

The partition with the **smallest total weight** of connecting edges corresponds to the **physical clustering** of the data set.

# Combinations of clusterings

## B. Combination of clusterings

### Function optimization methods

- *Utility function optimization*
- *Normalized mutual information*
- *Mixture model formulation*

Here, the **final clustering** (also called **median clustering**)  $\mathcal{F} = \{F_1, F_2, \dots, F_m\}$ , results from the optimization of an appropriate cost function.



# Combinations of clusterings

## B. Combination of clusterings

### Function optimization methods

- *Utility function optimization (probabilistic arguments)*
- *Normalized mutual information function optimization (information theory ingredients)*
- *Mixture model formulation*

A function  $U(\mathcal{F}', \mathcal{R}_i)$  is adopted, **measuring** the **quality** of a candidate median  $\mathcal{F}'$  against some other clustering  $\mathcal{R}_i$ .

The **overall utility** of  $\mathcal{F}'$  on  $\mathcal{E} = \{\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_n\}$  is defined as

$$U(\mathcal{F}', \mathcal{E}) = \sum_{i=1}^n U(\mathcal{F}', \mathcal{R}_i)$$

The **final (median) clustering**  $\mathcal{F}$  results as

$$\mathcal{F} = \operatorname{argmax}_{\mathcal{F}'} U(\mathcal{F}', \mathcal{E})$$

# Combinations of clusterings

## B. Combination of clusterings

### Function optimization methods

#### Mixture model formulation

- Represent the **data points** as follows

			$y_1$	$\dots$	$y_n$			
$x_1$	$\rightarrow$	[	$y_1(1)$	$\dots$	$y_n(1)$	]	$\equiv$	$x_1'$
$x_2$	$\rightarrow$	[	$y_1(2)$	$\dots$	$y_n(2)$	]	$\equiv$	$x_2'$
$\vdots$	$\rightarrow$			$\vdots$				$\vdots$
$x_N$	$\rightarrow$	[	$y_1(N)$	$\dots$	$y_n(N)$	]	$\equiv$	$x_N'$

**Note:** The representations  $x_i'$  are **discrete-valued**.

- **Define** the probability function  $P(x'; \theta)$  as the (weighted) **summation** of  $m$  ( $n$ -dimensional) **probability functions**, each one corresponding to a cluster.
- **Assuming independence** among the **components** of  $x'$ , each  $n$ -dimensional **probability function** is **written** as the **product** of  $n$  **one-dimensional** prob. functions, each one **modeled** by a **multinomial distribution**.
- The **estimation** of the respective **parameters** is carried out via the utilization of the **EM** algorithm.

# Multinomial distribution

## • Multinomial distribution $Mult(\mathbf{x}|n, \mathbf{P})$

Discrete RV distribution

$$\mathbf{x} = [x_1, x_2, \dots, x_K]^T, \mathbf{P} = [p_1, \dots, p_K]^T:$$

$$\sum_{i=1}^K p_i = 1$$

- $0 < p_i < 1, i = 1, \dots, K,$
- Sample space:  $X = \{0, 1, \dots, K\}$
- **Outcome** of the experiment: **non-binary**. No. of **repetitions**:  $n$
- $x_i$ : number of times the  $i$ -th outcome occurs in the  $n$  repetitions
- It is

$$\triangleright P(\mathbf{x}) = \binom{n}{x_1, x_2, \dots, x_K} \prod_{i=1}^K P_i^{x_i}$$

$$\text{s.t. } x_1 + x_2 + \dots + x_K = n$$

$$\triangleright E[\mathbf{x}] = n\mathbf{P}$$

$$\triangleright \sigma_i^2 = nP_i(1 - P_i), i = 1, \dots, K.$$

$$\triangleright \text{cov}(x_i, x_j) = -nP_i P_j, i \neq j.$$



$$\binom{n}{x_1, x_2, \dots, x_K} = \frac{n!}{x_1! x_2! \dots x_K!}$$