Clustering algorithms

Konstantinos Koutroumbas

<u>Unit 11</u>

– Clust. algs. for high dim. data sets (dim. reduction (PCA), subspace clust.)
– Combinations of clusterings

koutroum@noa.gr

Clustering algorithms for high dimensional data sets

- What is a high-dimensionality space?
  Dimensionality l of the input space with
  $$20 \leq l \leq few\ thousands$$
  indicate high-dimensional data sets.

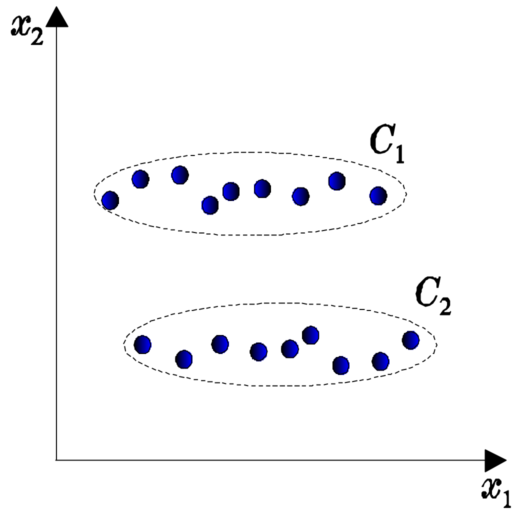
- Problems of *considering simultaneously* all dimensions in high-dimensional data sets:
  - "Curse of dimensionality". As a fixed number of points spread out in high-dimensional spaces, they become almost equidistant (that is, the terms similarity and dissimilarity tend to become meaningless – alternatively, 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).

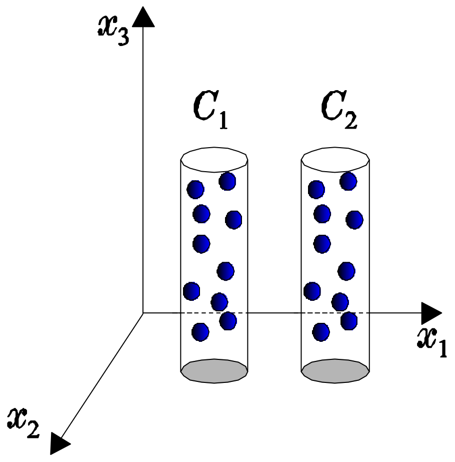
- A way out: ***Work*** *on subspaces of dimension lower than l.*
  - Main approaches:
    - ❑ Dimensionality reduction clustering approach.
    - ❑ Subspace clustering approach.

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 $H_{l'}$.

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.

- **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)

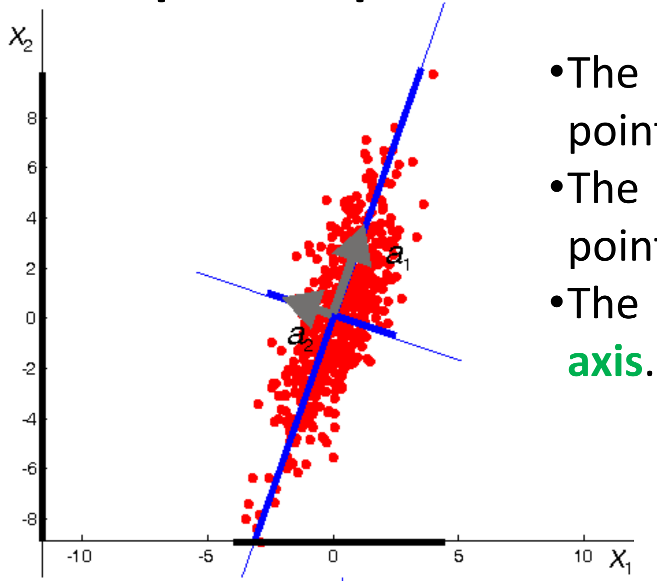
Principal component analysis (**PCA**):

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

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

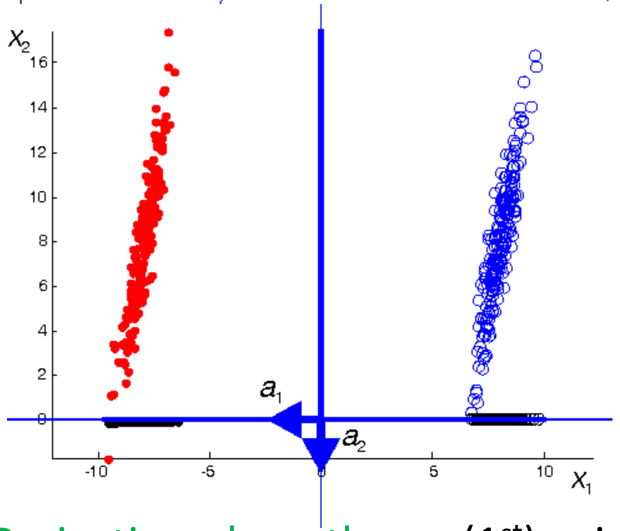
Principal Component Analysis - PCA

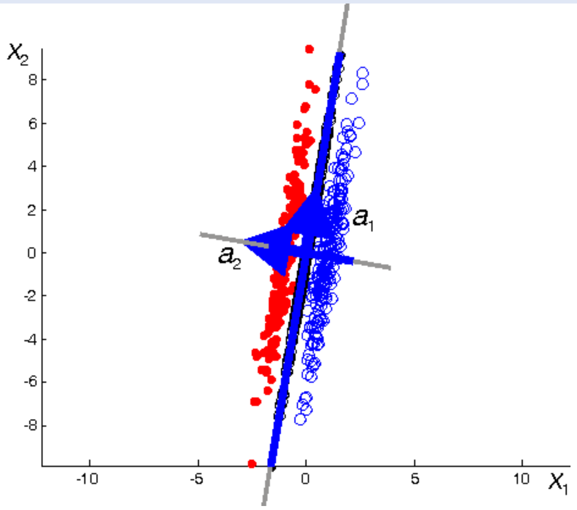
Principal Component Analysis (PCA)



- 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 $\boldsymbol{a}_1$ (1st) principal direction retains cluster separability.

Projection along the $\boldsymbol{a}_1$ principal direction **DOES NOT** retain cluster separability.

Subspace clustering

- **Solution:** Principal component analysis (PCA)
- **Let** $X_{l\times N} = [\boldsymbol{x}_1 \quad \boldsymbol{x}_2 \quad \cdots \quad \boldsymbol{x}_N]$ and $Y_{l'\times N} = [\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad \cdots \quad \boldsymbol{y}_N]$

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

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

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 $X'X'^{T}_{lxl}$

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

Let

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

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

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

Subspace clustering

More on SVD

Then

$$X'_{l\times N} = U'_{l\times l} \cdot \Sigma'_{l\times N} \cdot {V'^T}_{N\times N}$$

$$= \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \cdots & \boldsymbol{u}_l \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \sigma_l & \vdots & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1{}^T \\ \boldsymbol{v}_2{}^T \\ \vdots \\ \boldsymbol{v}_N{}^T \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \cdots & \boldsymbol{u}_l \end{bmatrix} \begin{bmatrix} \sigma_1 \boldsymbol{v}_1{}^T \\ \sigma_2 \boldsymbol{v}_2{}^T \\ \vdots \\ \sigma_l \boldsymbol{v}_l{}^T \end{bmatrix} =$$

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

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

By **neglecting** the **terms** with **"small" $\boldsymbol{\sigma_i}$**'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.

(*) R. Kohani, G. John, Wrappers for feature subset selection, Artificial Intelligence, Vol. 97 (1-2), 1997

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

<u>Clustering using Random Projections</u>

- **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 $\boldsymbol{x}_i$ **belongs** to the j-th cluster in the s-th projection, $P(C_j{}^s|\boldsymbol{x}_i), i = 1, \dots, N, j = 1, \dots, m_s$.
  - ➢ **Create** the $N \times N$ similarity matrix $P^s = [P_{ij}^s]$, where P_{ij}^s is the probability that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ **belong** to the same cluster,

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

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

- **End for**
- **Compute** the $N \times N$ 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.

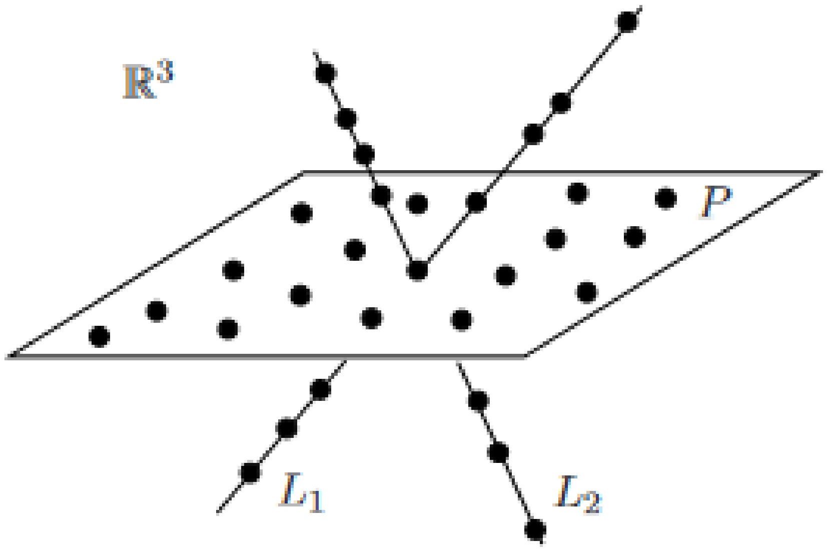
Clustering algorithms for high dimensional data sets

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:



Subspace clustering

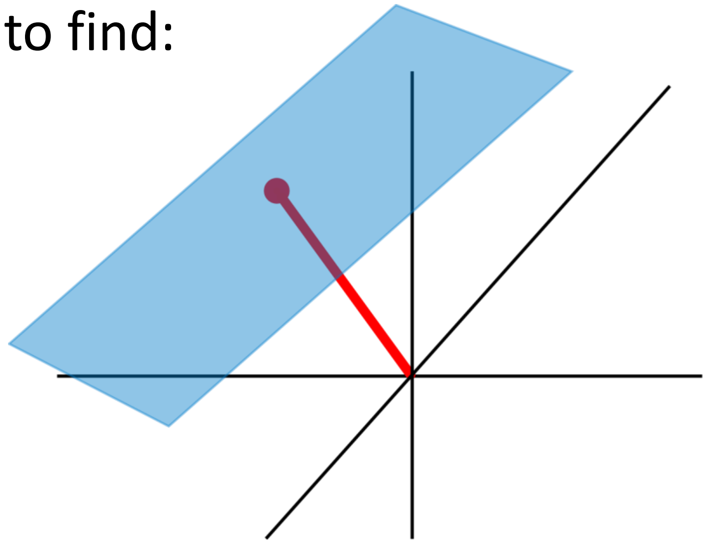
Preliminaries:

- The data set $X = \{\boldsymbol{x}_i \in R^l, i = 1, \dots, N\}$

- (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 $S = \{\boldsymbol{x} \in R^l : \boldsymbol{x} = \boldsymbol{\mu} + B \cdot \boldsymbol{y}\}$, where $\boldsymbol{y} \in R^{l'}$

- **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.



Vidal R., "Subspace Clustering", IEEE Transactions on Signal Processing, 28(2), 2011.

Subspace clustering

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

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

- **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 $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_m$, 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 $l_1, l_2, \ldots, l_m$, are **known**.

Let:

- $U_{N\times m} = [u_{ij}]$, where $u_{ij} = \begin{cases} 1, & x_i \in C_j \\ 0, & otherwise \end{cases}$,
- $B = \{B_1, B_2, \ldots, B_m\}$
- $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_m\}$
- $Y = \{Y_1, \ldots, Y_m\}$, with $Y_j = \left\{\boldsymbol{y}_i^j, i = 1, \ldots, N\right\}$ 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

$x'^{j}_{i} = \mu_j + B_j y^{j}_{i}$: **Projection** of x_i to the j-th subspace

Consider the cost function

$$J(B, \mu, Y, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} \left\| x_i - x'^{j}_{i} \right\|^2 = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij} \left\| x_i - \mu_j - B_j y^{j}_{i} \right\|^2$$

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

For <u>**fixed**</u> $\mu'_j s,\ B'_j s, y^{j\prime}_{i} s$:

Define $u_{ij} = \begin{cases} 1, & if \left\| x_i - \mu_j - B_j y^{j}_{i} \right\|^2 = min_{q=1,\ldots,m} \left\| x_i - \mu_q - B_q y^{q}_{i} \right\|^2 \\ 0, & otherwise \end{cases}$

For <u>**fixed** u_{ij}**'s:**</u> Solve the following <u>m</u> independent problems

$$min_{\{\mu_j, (B_j, y^{j}_{i})\}} \sum_{x_i : u_{ij}=1} \left\| x_i - \mu_j - B_j y^{j}_{i} \right\|^2 \equiv min_{\{\mu_j, (B_j, y^{j}_{i})\}} \sum_{i=1}^{N} u_{ij} \left\| x_i - \mu_j - B_j y^{j}_{i} \right\|^2$$

For **each** such problem

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

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

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 $X = \{\boldsymbol{x}_i \in R^l, i = 1, \dots, N\}$

- 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:

$$\mathcal{R}_i \leftrightarrow \boldsymbol{y}_i = [y_i(1), y_i(2), \dots, y_i(k), \dots, y_i(N)]$$

  where $y_i(k)$ the cluster label of the k-th data point.

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

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 largest 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)
- *Hybric 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 $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4\}$ 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 $C_1 = \{\boldsymbol{x}_1, \boldsymbol{x}_2\}$, $C_2 = \{\boldsymbol{x}_3, \boldsymbol{x}_4\}$.

Consider the **fully connected graph** with four vertices $\boldsymbol{v}_1(\boldsymbol{x}_1), \boldsymbol{v}_2(\boldsymbol{x}_2)$, $\boldsymbol{v}_3(\boldsymbol{x}_3), \boldsymbol{v}_4(\boldsymbol{x}_4)$, 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

$$\mathcal{F} = argmax_{\mathcal{F}'} U(\mathcal{F}', \mathcal{E})$$

Combinations of clusterings

B. Combination of clusterings

Function optimization methods

Mixture model formulation

- **Represent** the data points as follows

			$\boldsymbol{y}_1$	$\cdots$	$\boldsymbol{y}_n$			
$\boldsymbol{x}_1$	$\rightarrow$	[	$y_1(1)$	$\cdots$	$y_n(1)$	]	$\equiv$	$\boldsymbol{x}_1{}'$
$\boldsymbol{x}_2$	$\rightarrow$	[	$y_1(2)$	$\cdots$	$y_n(2)$	]	$\equiv$	$\boldsymbol{x}_2{}'$
$\vdots$	$\rightarrow$			$\vdots$				$\vdots$
$\boldsymbol{x}_N$	$\rightarrow$	[	$y_1(N)$	$\cdots$	$y_n(N)$	]	$\equiv$	$\boldsymbol{x}_N{}'$

Note: The representations $\boldsymbol{x}_i'$ are discrete-valued.

- **Define** the probability function $P(\boldsymbol{x}'; \boldsymbol{\Theta})$ as the (weighted) summation of m (n-dimensional) probability functions, *each one corresponding to a cluster*.
- **Assuming** independence among the components of $\boldsymbol{x}'$, 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** $Mult(\boldsymbol{x}|n, \boldsymbol{P})$

  Discrete RV distribution

  $\mathbf{x} = [\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_K]^T$, $\boldsymbol{P} = [p_1, \dots, p_K]^T$:

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

  - $0 < p_i < 1, i = 1, \dots, K,$
  - Sample space: $X = \{0,1, \dots, K\}$
  - Outcome of the experiment: **non-binary**. No. of repetitions: $\boldsymbol{n}$
  - x_i: number of times the i-th outcome occurs in the n repetitions
  - It is
    - $P(\boldsymbol{x}) = \binom{n}{x_1, x_2, \dots, x_K} \prod_{i=1}^{K} {p_i}^{x_i}$

    s.t. $x_1 + x_2 + \dots + x_K = n$
    - $E[\boldsymbol{x}] = n\boldsymbol{P}$
    - $\sigma_i^2 = nP_i(1 - P_i), i = 1, \dots, K.$
    - $cov(x_i, x_j) = -nP_i P_j, i \neq j.$

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