聚 类 分 析

刘 新 旺

https://xinwangliu.github.io/

国 防 科 技 大 学　　计 算 机 学 院

计 算 科 学 系 人 工 智 能 与 大 数 据 教 研 室

2020 年 12 月 8 日

Content

Unsupervised Learning

- supervised learning used labeled data pairs $(\mathbf{x}, y)$ to learn a function $f : \mathcal{X} \to \mathcal{Y}$.

Unsupervised Learning

- supervised learning used labeled data pairs $(\mathbf{x}, y)$ to learn a function $f : \mathcal{X} \to \mathcal{Y}$.
- But, what if we don't have labels?

Unsupervised Learning

- supervised learning used labeled data pairs $(\mathbf{x}, y)$ to learn a function $f : \mathcal{X} \to \mathcal{Y}$.
- But, what if we don't have labels?
- No labels = unsupervised learning

Unsupervised Learning

- supervised learning used labeled data pairs $(\mathbf{x}, y)$ to learn a function $f : \mathcal{X} \to \mathcal{Y}$.
- But, what if we don't have labels?
- No labels = unsupervised learning
- Useful for:
  - Automatically organizing data
  - Understanding hidden structure in some data
  - Representing high-dimensional data in a low-dimensional space

What is Data Clustering?

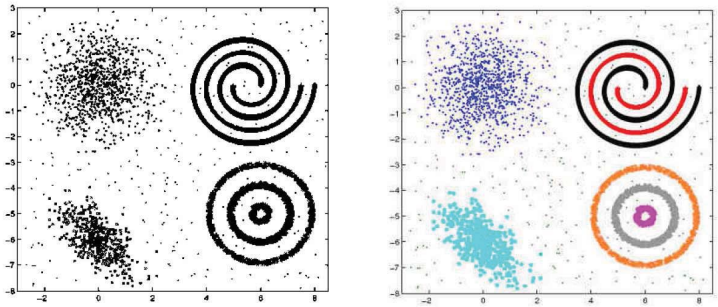
- Data Clustering is an unsupervised learning problem

What is Data Clustering?

- Data Clustering is an unsupervised learning problem
- Given: n unlabeled examples $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$; the number of partitions K

What is Data Clustering?

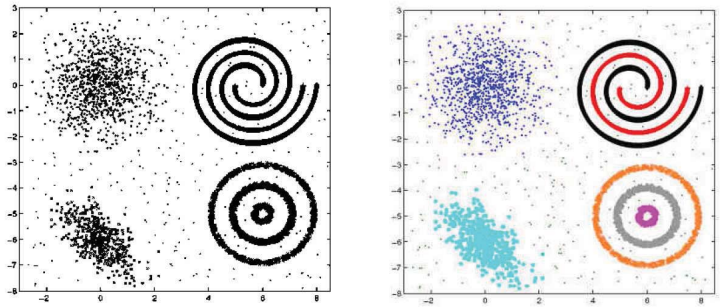
- Data Clustering is an unsupervised learning problem
- Given: n unlabeled examples $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$; the number of partitions K
- Goal: Group the examples into K partitions



(a) Input data (b) Desired clustering

What is Data Clustering?

- Data Clustering is an unsupervised learning problem
- Given: n unlabeled examples $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$; the number of partitions K
- Goal: Group the examples into K partitions

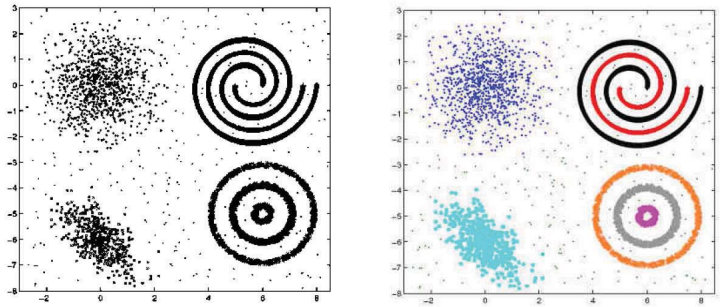


(a) Input data (b) Desired clustering

- The only information clustering uses is the similarity between examples

What is Data Clustering?

- Data Clustering is an unsupervised learning problem
- Given: n unlabeled examples $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$; the number of partitions K
- Goal: Group the examples into K partitions

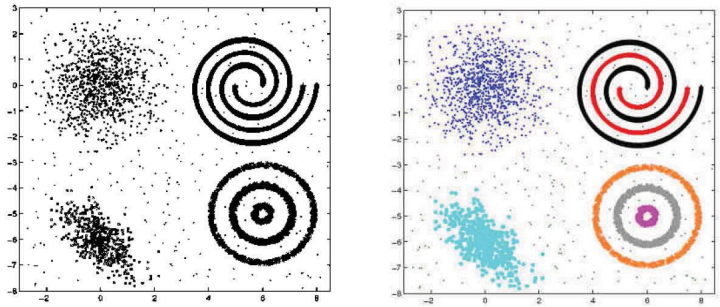


(a) Input data (b) Desired clustering

- The only information clustering uses is the similarity between examples
- Clustering groups examples based of their mutual similarities

What is Data Clustering?

- Data Clustering is an unsupervised learning problem
- Given: n unlabeled examples $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$; the number of partitions K
- Goal: Group the examples into K partitions



(a) Input data (b) Desired clustering

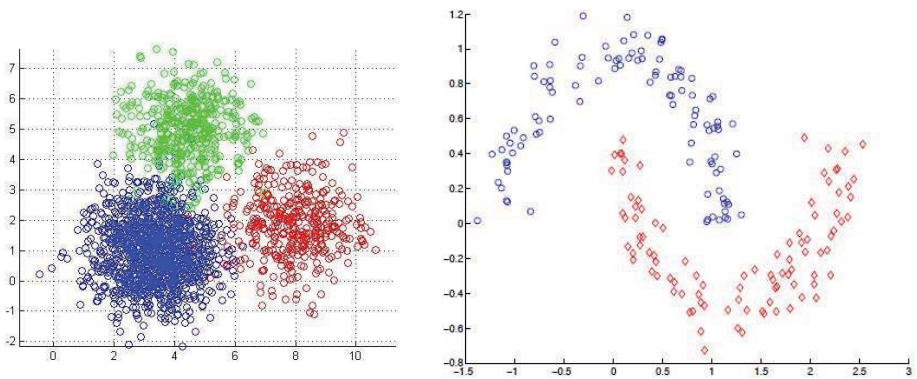
- The only information clustering uses is the similarity between examples
- Clustering groups examples based of their mutual similarities
- A good clustering is one that achieves: (1) High within-cluster similarity (2) Low inter-cluster similarity

Notions of Similarity

- Choice of the **similarity measure** is very important for clustering

Notions of Similarity

- Choice of the **similarity measure** is very important for clustering
- Similarity is inversely related to distance
- Different ways exist to measure distances. Some examples:
  - Euclidean distance: $d(\mathbf{x}, \mathbf{z}) = \sqrt{\sum_{d=1}^{D} (x_d - z_d)^2}$
  - Manhattan distance: $d(\mathbf{x}, \mathbf{z}) = \sum_{d=1}^{D} |x_d - z_d|$
  - Kernelized (non-linear) distance: $d(\mathbf{x}, \mathbf{z}) = \|\phi(\mathbf{x}) - \phi(\mathbf{z})\|$



- For the left figure, Euclidean distance may be reasonable
- For the right figure, kernelized distance seems more reasonable

Similarity is Subjective

- Similarity is often hard to define

Similarity is Subjective

- Similarity is often hard to define

- Different similarity criteria can lead to different clusterings

Data Clustering: Some Real-World Examples

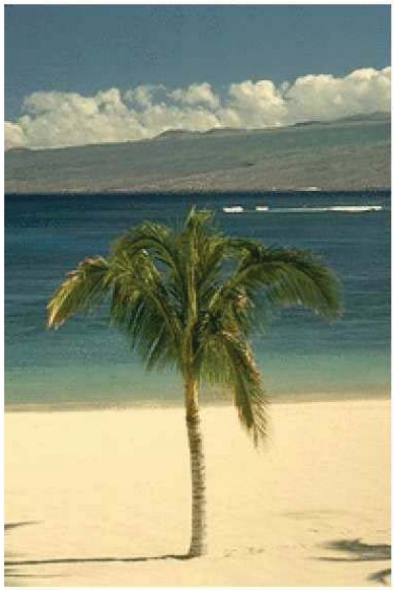
- Clustering images based on their perceptual similarities

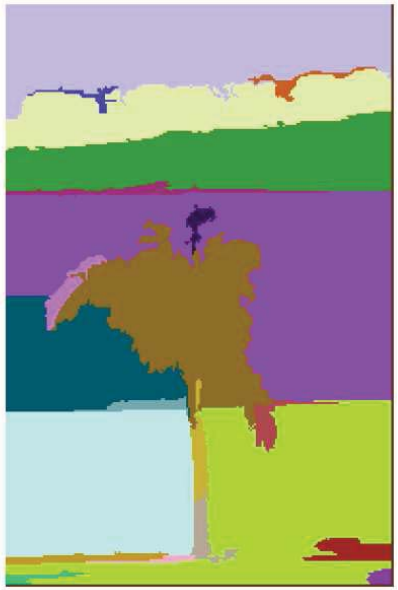
Data Clustering: Some Real-World Examples

- Clustering images based on their perceptual similarities
- Image segmentation (clustering pixels)

Data Clustering: Some Real-World Examples

- Clustering images based on their perceptual similarities
- Image segmentation (clustering pixels)

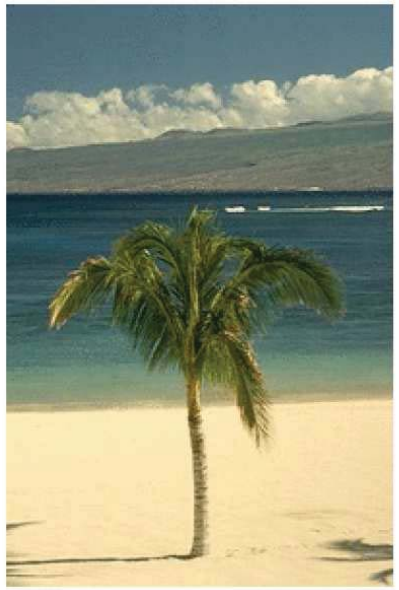


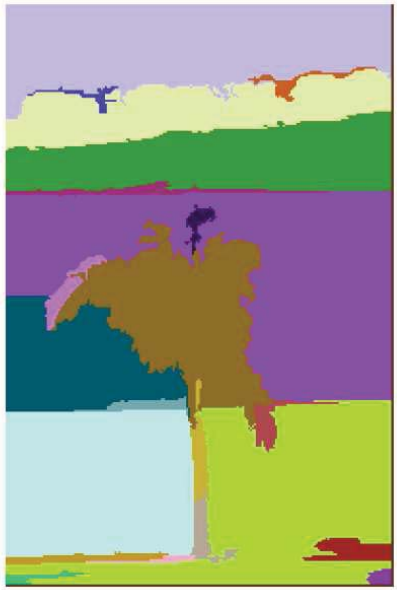


- Clustering web-pages based on their content

Data Clustering: Some Real-World Examples

- Clustering images based on their perceptual similarities
- Image segmentation (clustering pixels)

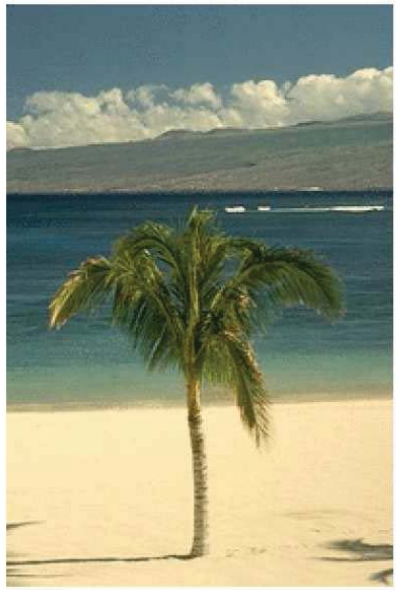


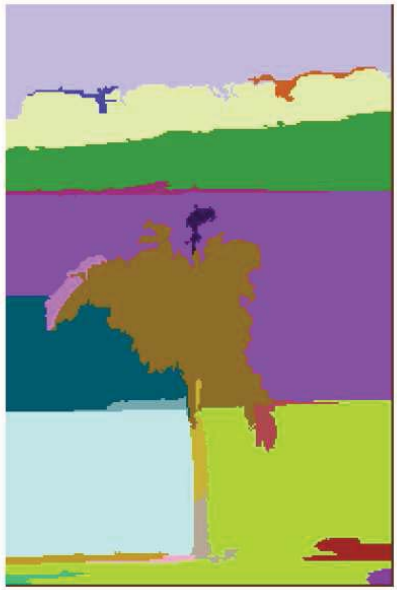


- Clustering web-pages based on their content
- Clustering people in social networks based on user properties/preferences

Data Clustering: Some Real-World Examples

- Clustering images based on their perceptual similarities
- Image segmentation (clustering pixels)





- Clustering web-pages based on their content
- Clustering people in social networks based on user properties/preferences
- . . .

Content

Main Idea of Hierarchical clustering

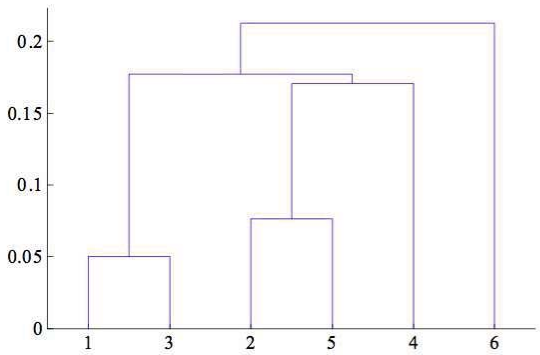
Hierarchical clustering

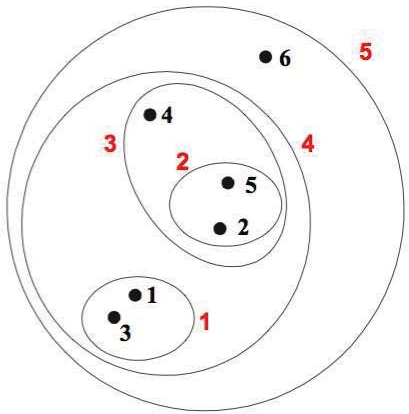
- agglomerative clustering, divisive clustering

Main Idea of Hierarchical clustering

Hierarchical clustering

- agglomerative clustering, divisive clustering
- Partitions can be visualized using a tree structure (a dendrogram)



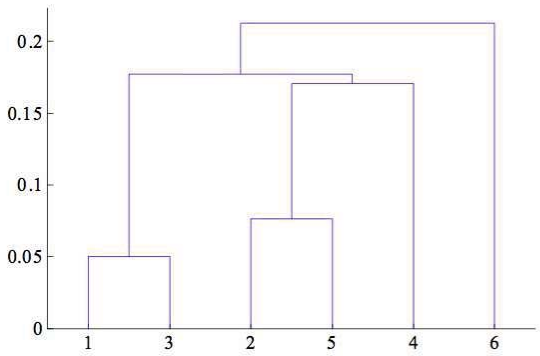


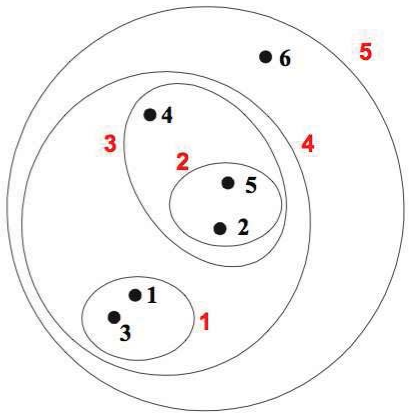
Main Idea of Hierarchical clustering

Hierarchical clustering

- agglomerative clustering, divisive clustering
- Partitions can be visualized using a tree structure (a dendrogram)
- Does not need the number of clusters as input

不需设定K

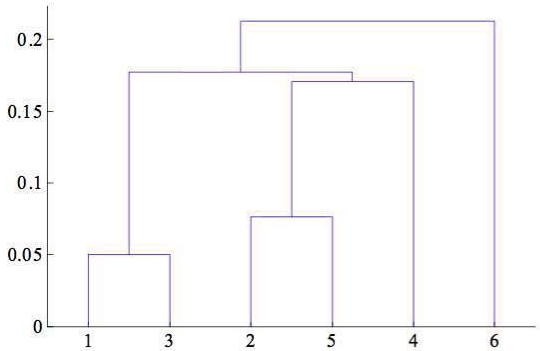


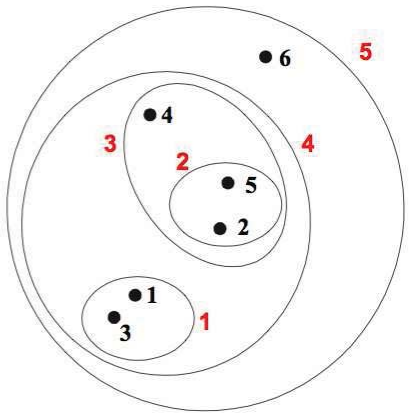


Main Idea of Hierarchical clustering

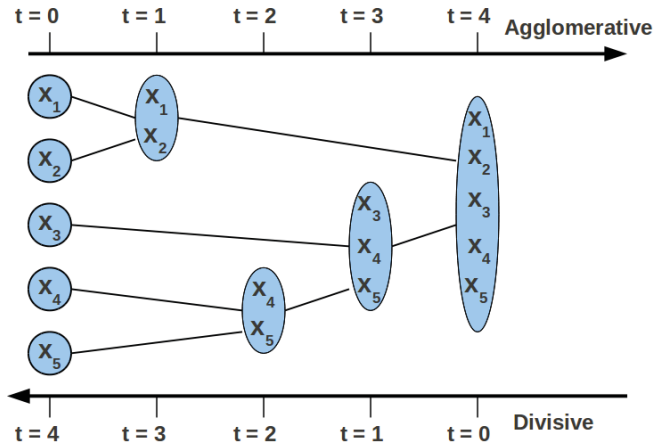
Hierarchical clustering

- agglomerative clustering, divisive clustering
- Partitions can be visualized using a tree structure (a dendrogram)
- Does not need the number of clusters as input
- Possible to view partitions at different levels of granularities (i.e., can refine/coarsen clusters) using different K





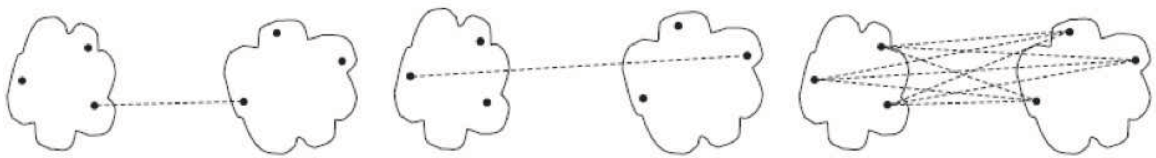
Hierarchical Clustering



- **Agglomerative (bottom-up) Clustering**
  1. Start with each example in its own singleton cluster
  2. At each step, greedily merge 2 most similar clusters
  3. Stop when there is a single cluster of all examples, else go to 2
- **Divisive (top-down) Clustering**
  1. Start with all examples in the same cluster
  2. At each step, remove "outsiders" from the least cohesive cluster
  3. Stop when each example is in its own singleton cluster, else go to 2

Hierarchical Clustering: (Dis)similarity between clusters

- We know how to compute the dissimilarity $d(\mathbf{x}_i, \mathbf{x}_j)$ between two examples.
- How to compute the dissimilarity between two clusters R and S?
- **Min-link or single-link**: $d(R,S) = \min_{\mathbf{x}_R \in R, \mathbf{x}_S \in S} \; d(\mathbf{x}_R, \mathbf{x}_S)$
- **Max-link or complete-link**: $d(R,S) = \max_{\mathbf{x}_R \in R, \mathbf{x}_S \in S} \; d(\mathbf{x}_R, \mathbf{x}_S)$
- **Average-link**: $d(R,S) = \frac{1}{|R| \cdot |S|} \sum_{\mathbf{x}_R \in R, \mathbf{x}_S \in S} \; d(\mathbf{x}_R, \mathbf{x}_S)$



(a) MIN (single link.) (b) MAX (complete link.) (c) Group average.

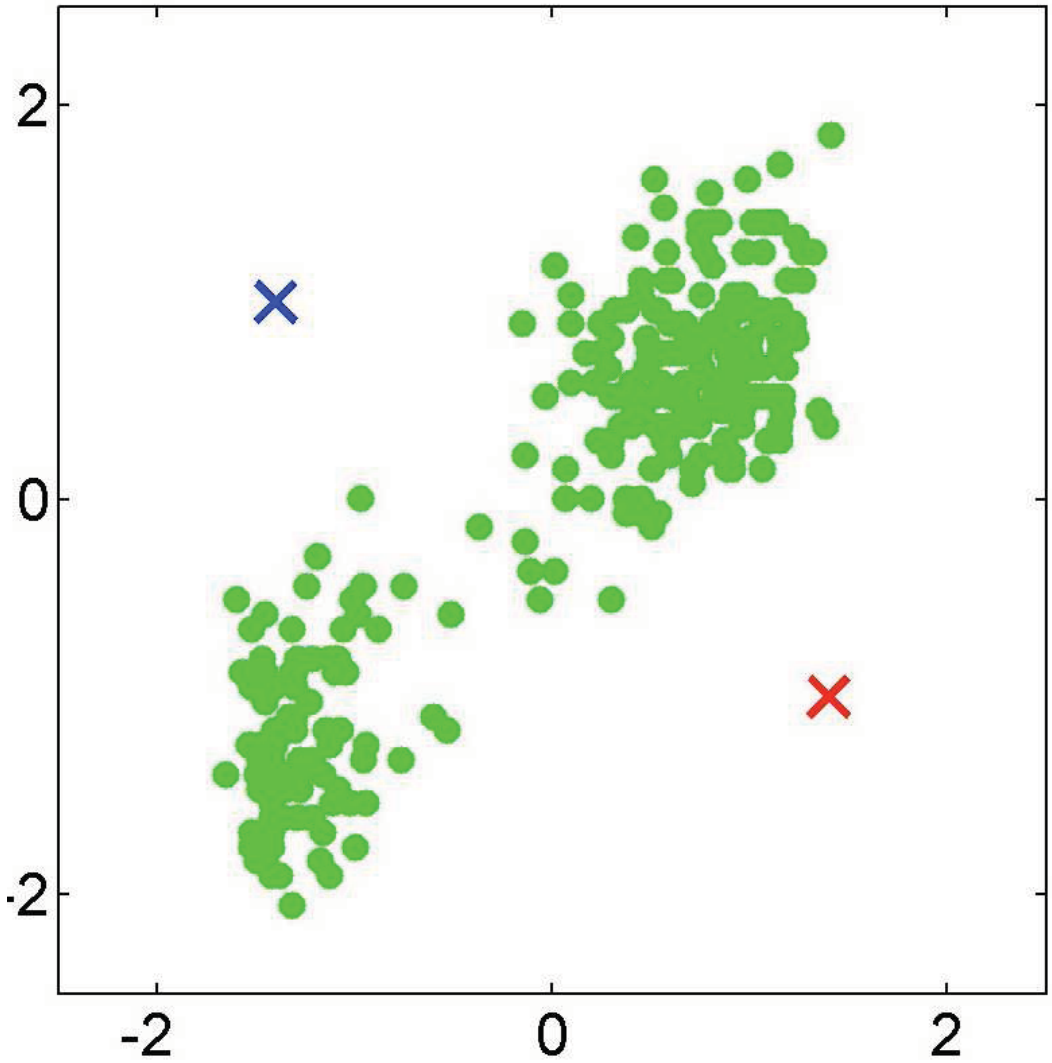
Hierarchical Clustering—Comments

- Hierarchical Clustering can give different partitions depending on the level-of-resolution we are looking at
- Hierarchical clustering doesn't need the number of clusters to be specified
- Hierarchical clustering can be slow (has to make several merge/split decisions)

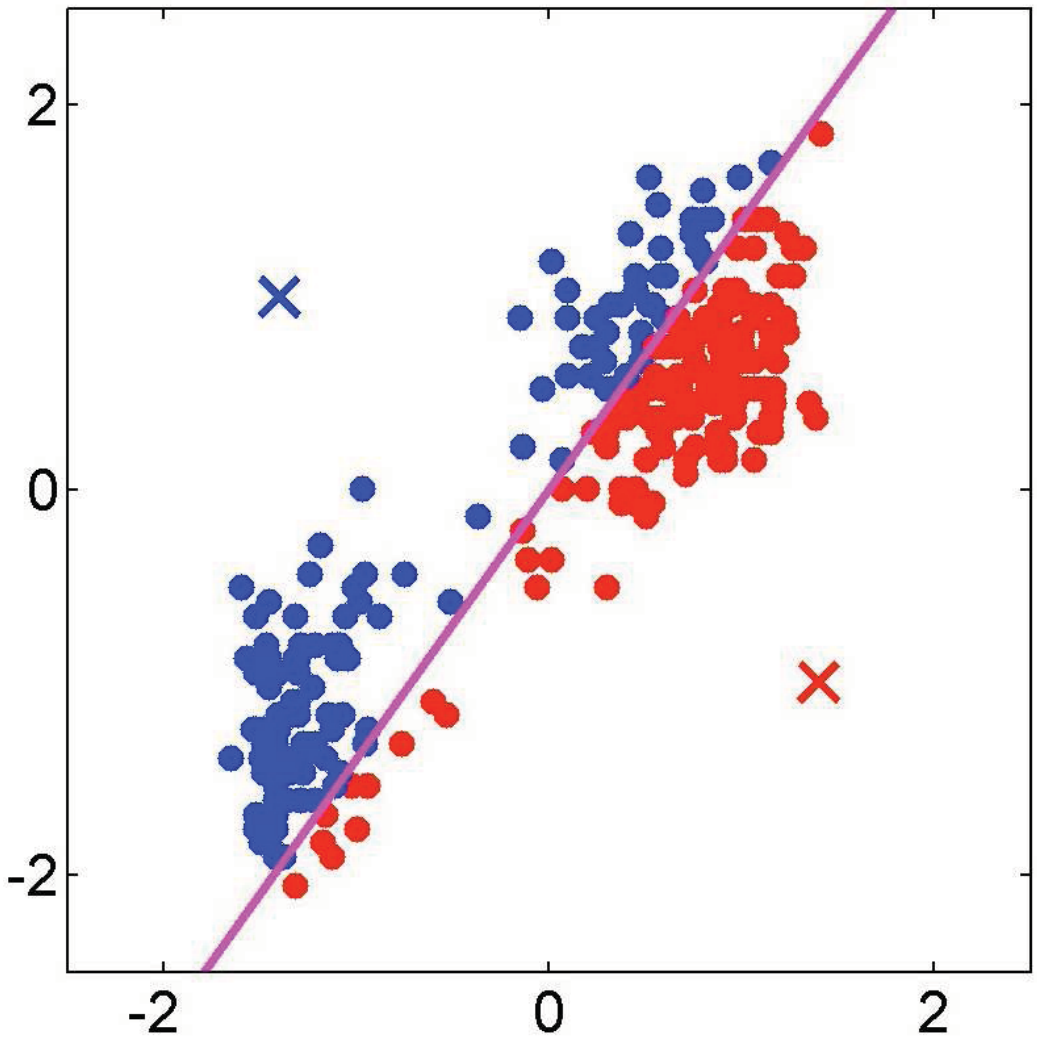
K-means Clustering

- **Input**: $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ $(\mathbf{x}_i \in \mathbb{R}^D)$; the number of partitions K
- **Initialize**: K cluster centers $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K$. Several initialization options:
  - Randomly initialized anywhere in $\mathbb{R}^D$
  - Choose any K examples as the cluster centers
- **Iterate**:
  - Assign each example $\mathbf{x}_i$ to its closest cluster center:
    $k = \arg\min_{k \in \{1, \cdots, K\}} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|$
  - Recompute the new cluster centers $\boldsymbol{\mu}_k$ (mean/centroid of the set $\mathcal{C}_k$):
    $\boldsymbol{\mu}_k = \frac{1}{|\mathcal{C}_k|} \sum_{j \in \mathcal{C}_k} \mathbf{x}_j$, where $\mathcal{C}_k$ is the set of examples of the k-th cluster.
  - Repeat while not converged.
- A possible convergence criteria: cluster centers do not change anymore. 一定收敛

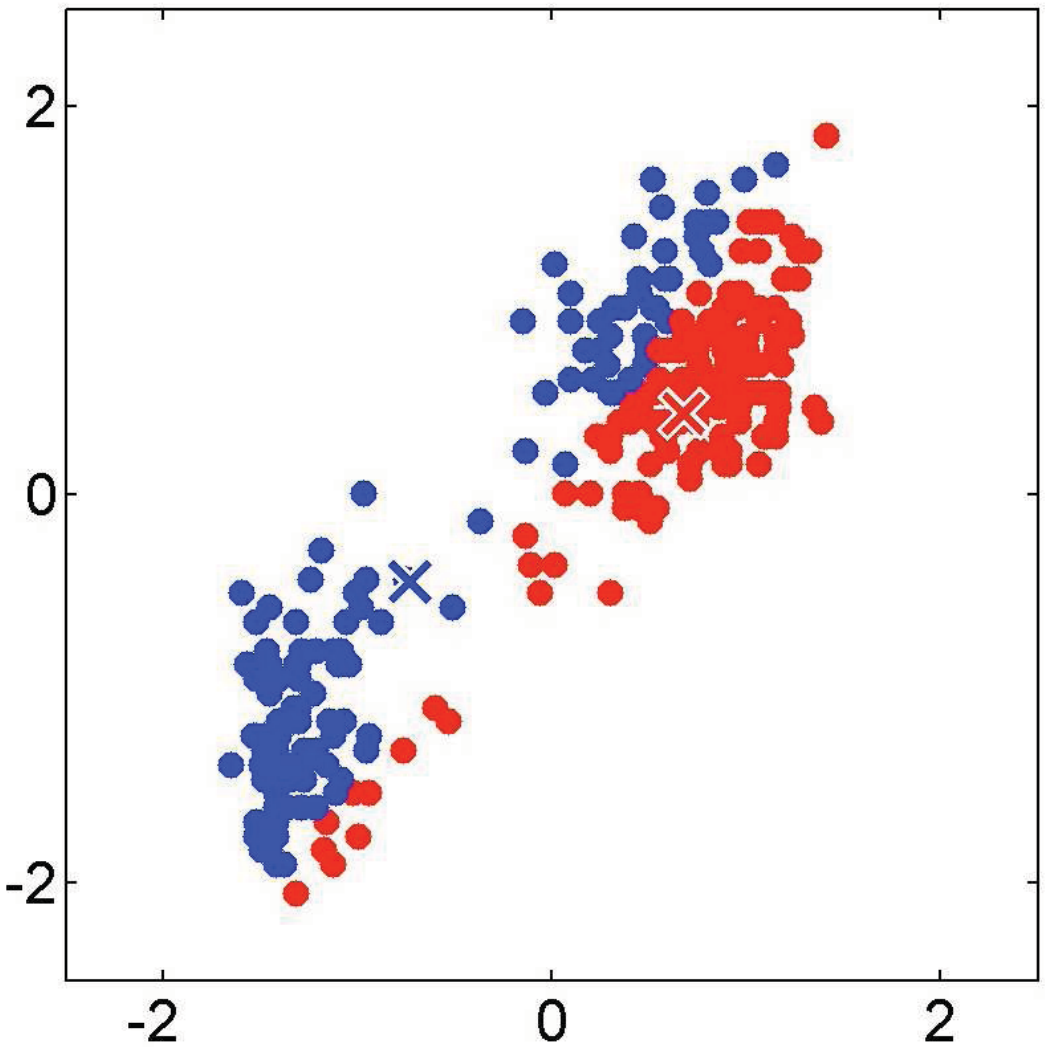
K-means: Initialization (assume $K = 2$)



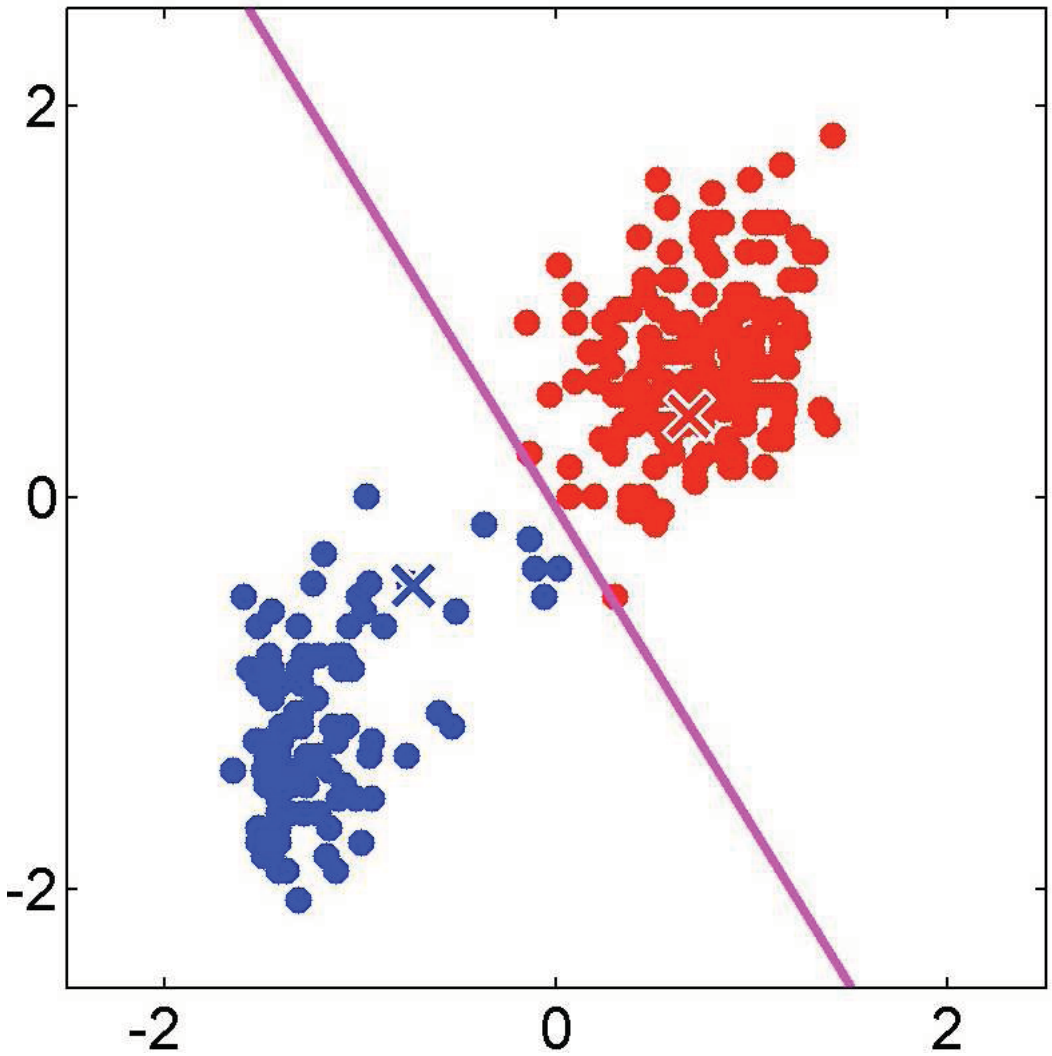
K-means iteration 1: Assigning points



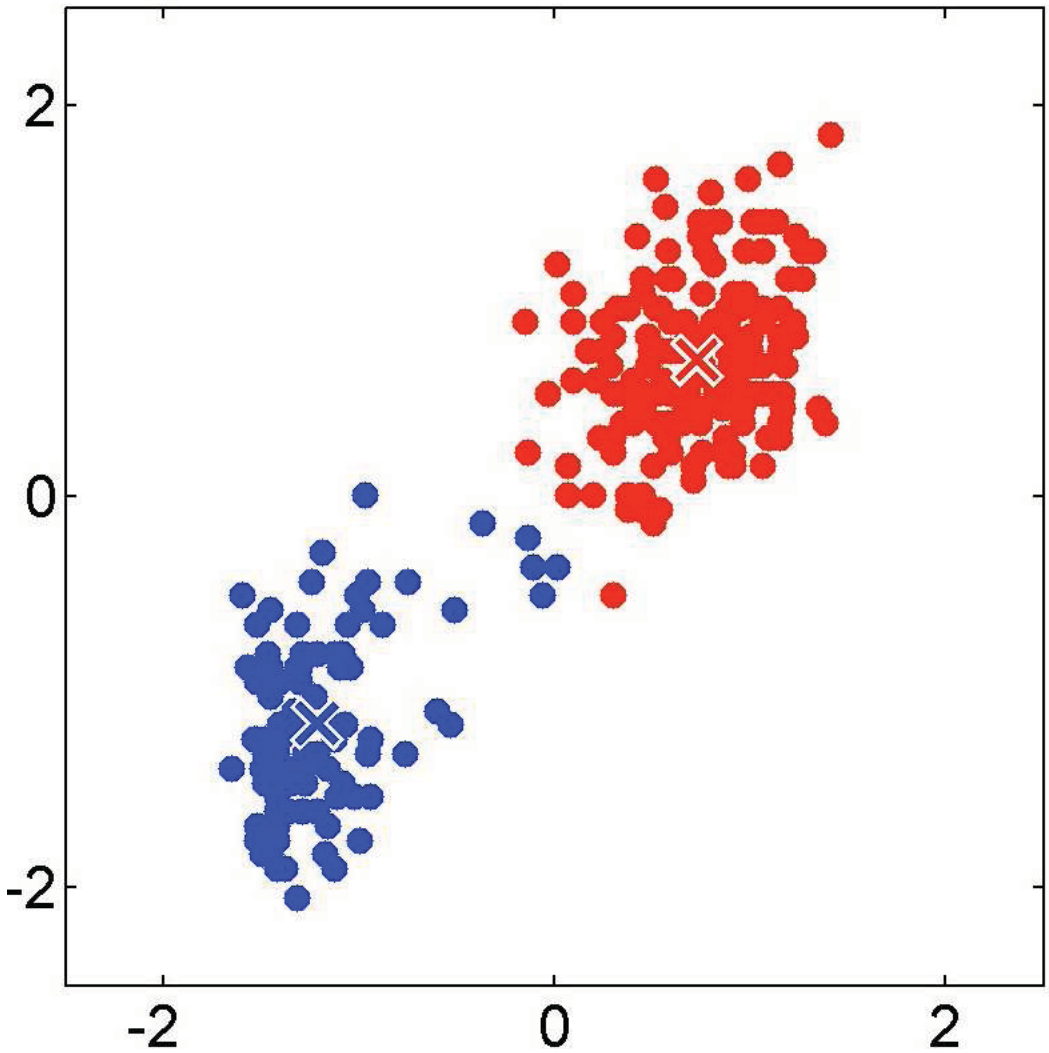
K-means iteration 1: Recomputing the cluster centers



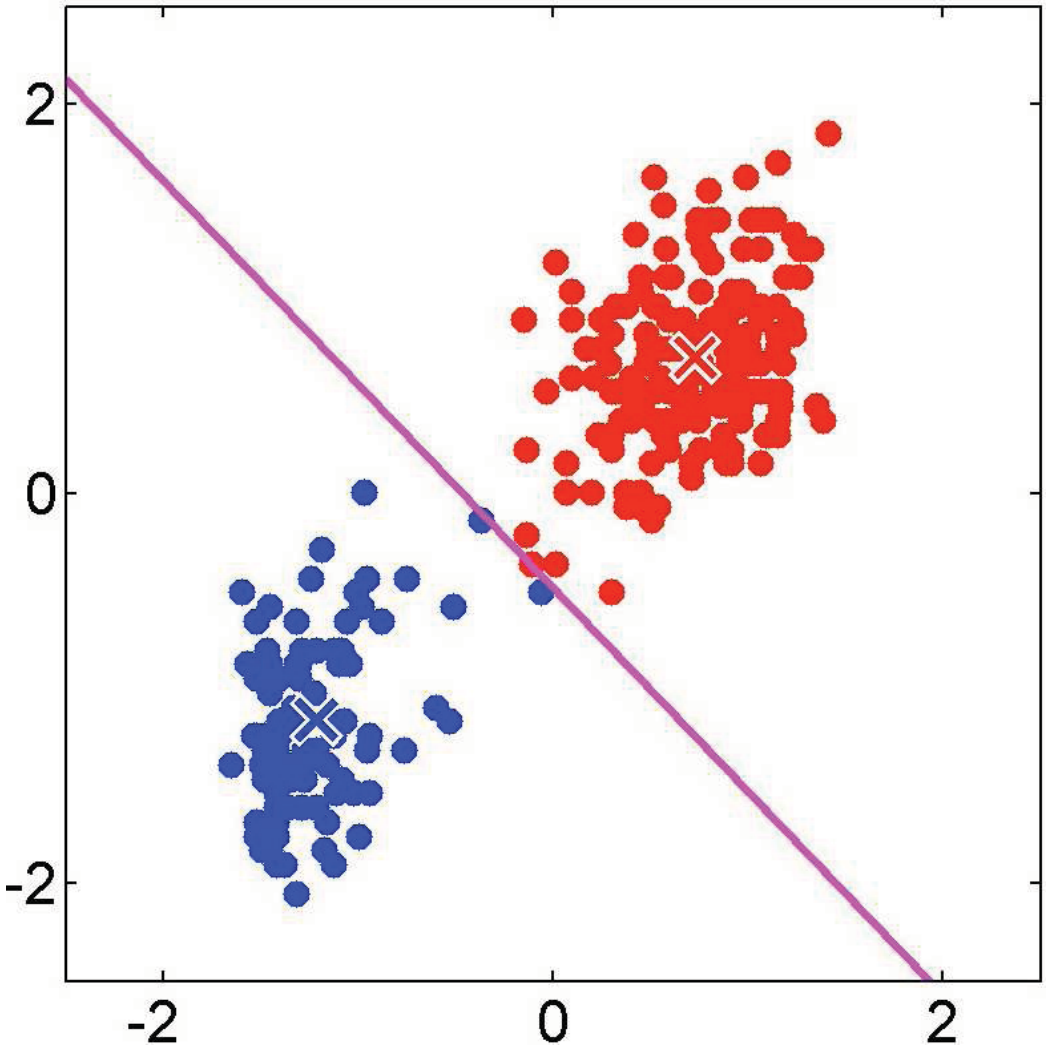
K-means iteration 2: Assigning points



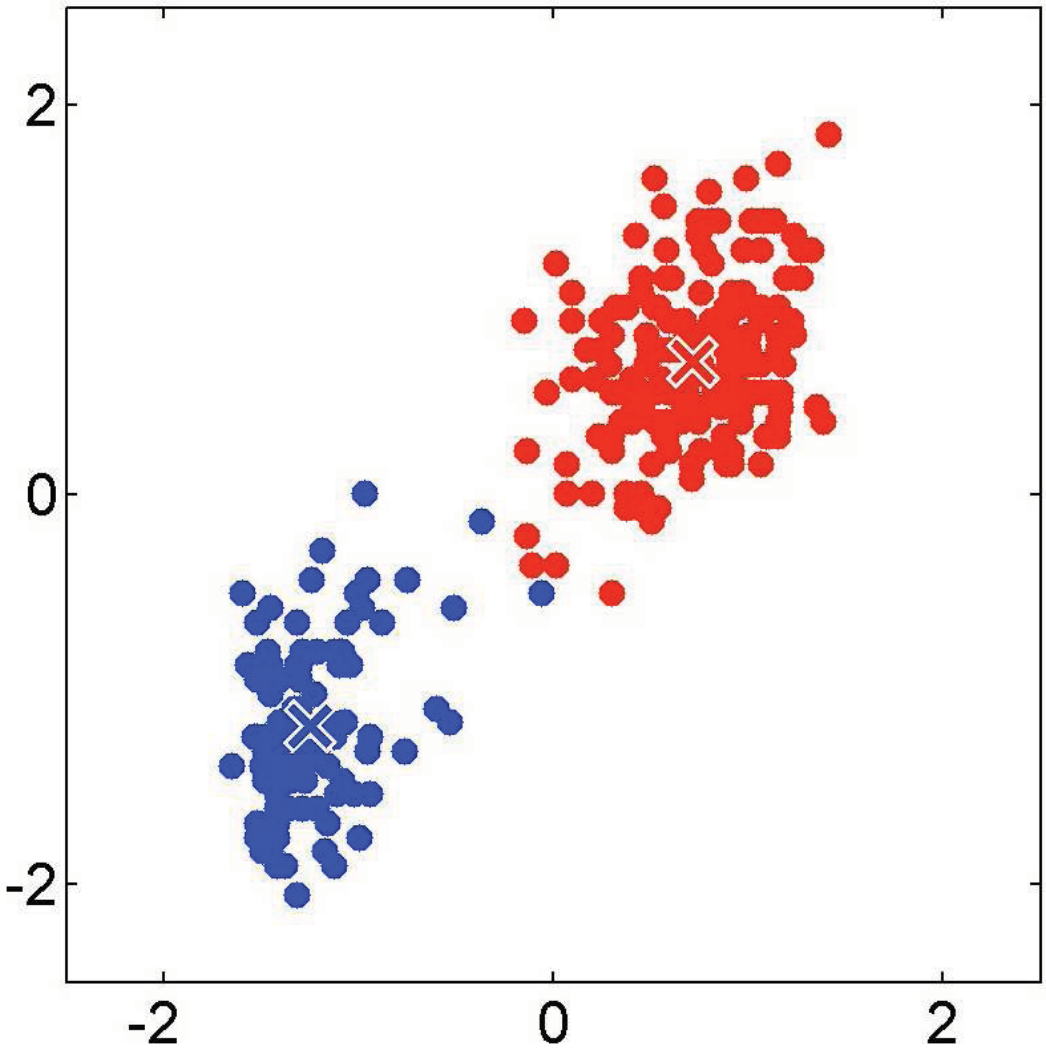
K-means iteration 2: Recomputing the cluster centers



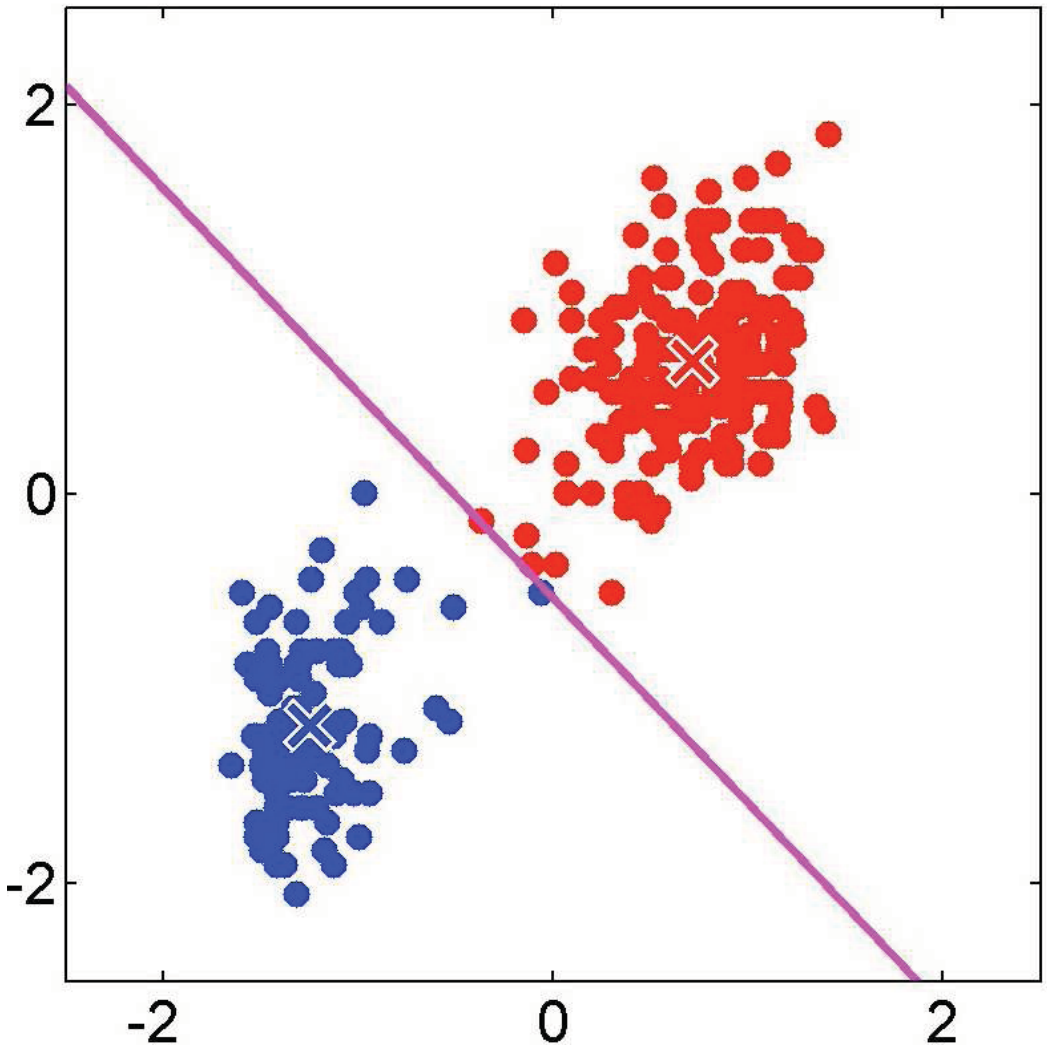
K-means iteration 3: Assigning points



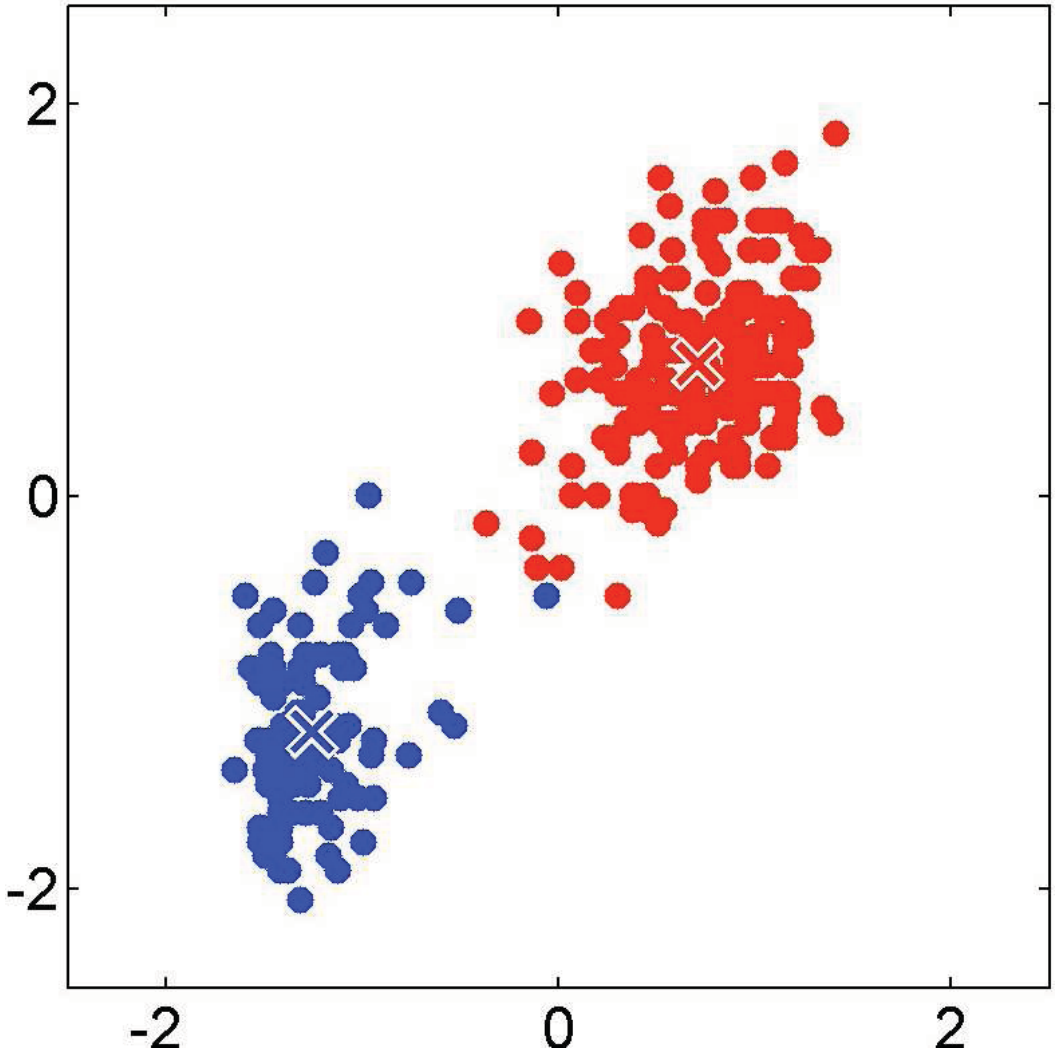
K-means iteration 3: Recomputing the cluster centers



K-means iteration 4: Assigning points



K-means iteration 4: Recomputing the cluster centers



K-means: The Objective Function

The K-means objective function

- Let $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K$ be the K cluster centroids (means)
- Let $r_{ik} \in \{0, 1\}$ be indicator denoting whether point $\mathbf{x}_i$ belongs to cluster k
- K-means objective minimizes the total distortion (sum of distances of points from their cluster centers)

$$J(\boldsymbol{\mu}, \mathbf{r}) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \tag{1}$$

- Note: Exact optimization of the K-means objective is NP-hard
- The K-means algorithm is a heuristic that converges to a local optimum

K-means: Initialization issues

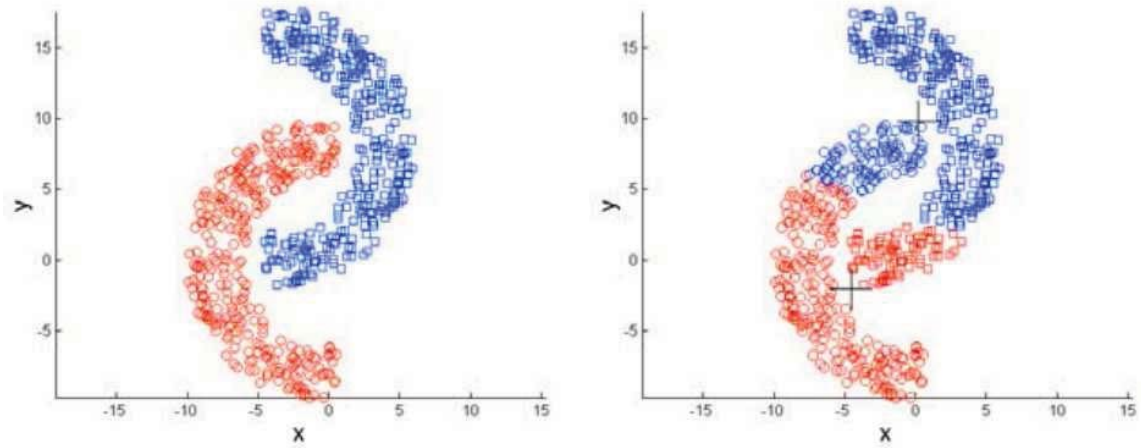
- K-means is extremely sensitive to cluster center initialization
- Bad initialization can lead to
  - Poor convergence speed
  - Bad overall clustering
- Safeguarding measures:
  - Choose first center as one of the examples, second which is the farthest from the first, third which is the farthest from both, and so on.
  - Try multiple initializations and choose the best result
  - Other smarter initialization schemes (e.g., look at the K-means++ algorithm by Arthur and Vassilvitskii)

K-means Limitations

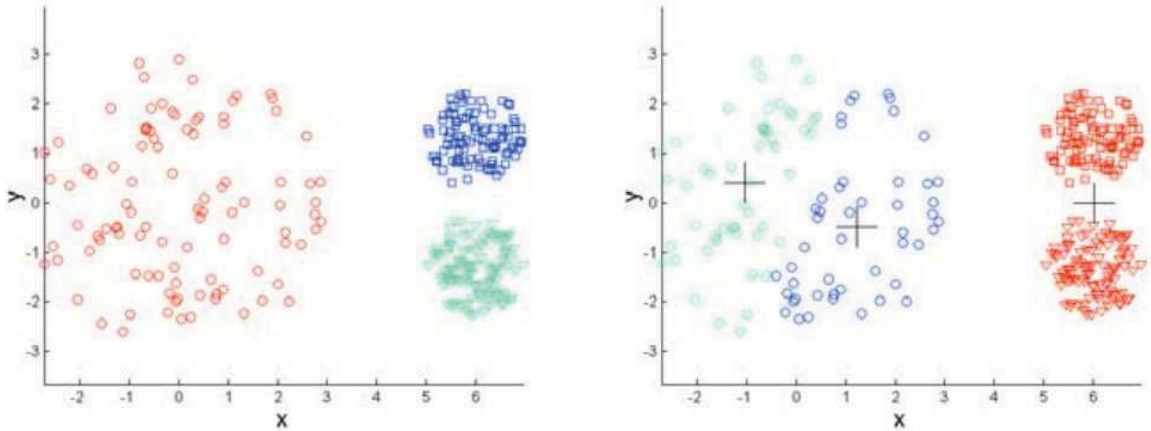
- Makes hard assignments of points to clusters
  - A point either completely belongs to a cluster or not belongs at all
  - No notion of a soft assignment (i.e., probability of being assigned to each cluster: say $K = 3$ and for some point $\mathbf{x}_n$, $p_1 = 0.7, p_2 = 0.2, p_3 = 0.1$)
  - Gaussian mixture models and Fuzzy K-means allow soft assignments
- Sensitive to outlier examples (such examples can affect the mean by a lot)
  - K-medians algorithm is a more robust alternative for data with outliers
  - Reason: Median is more robust than mean in presence of outliers
- Works well only for round shaped, and of roughly equal sizes/density clusters
- Does badly if the clusters have non-convex shapes
  - Spectral clustering or kernelized K-means can be an alternative

K-means Limitations Illustrated

Non-convex/non-round-shaped clusters: Standard K-means fails!



Clusters with different densities



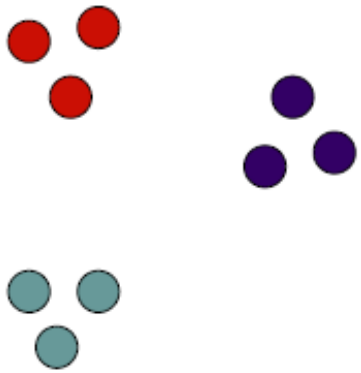
Graph Clustering

谱聚类

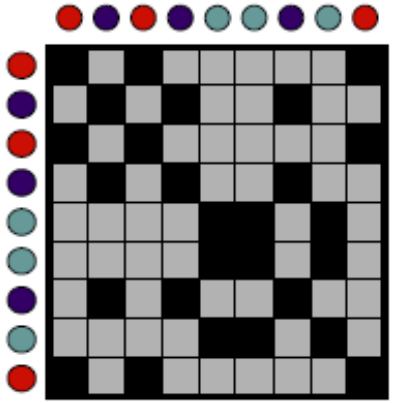
Goal: Given data points $X_1, \ldots, X_n$ and similarities $w(X_i,X_j)$, partition the data into groups so that points in a group are similar and points in different groups are dissimilar.

Similarity Graph: G(V,E,W)

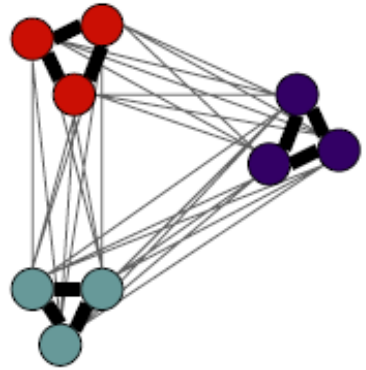
V – Vertices (Data points)
E – Edge if similarity > 0
W - Edge weights (similarities)



Data



Similarities



Similarity graph

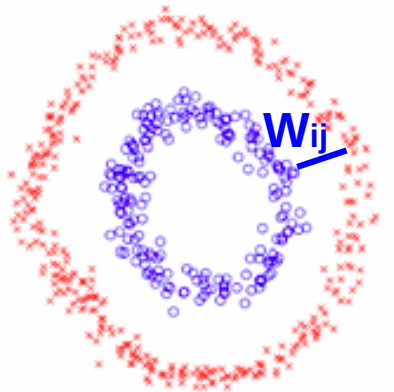
Partition the graph so that edges within a group have large weights and edges across groups have small weights.

Similarity graph construction

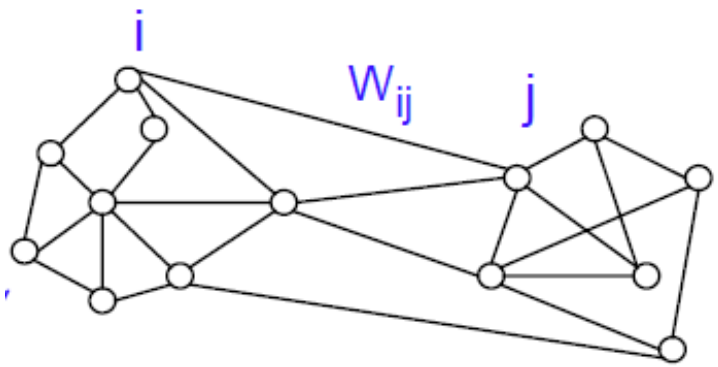
Similarity Graphs: Model local neighborhood relations between data points

E.g. Gaussian kernel similarity function

$$W_{ij} = e^{\frac{-\|x_i - x_j\|^2}{2\sigma^2}}$$ → Controls size of neighborhood



Data clustering

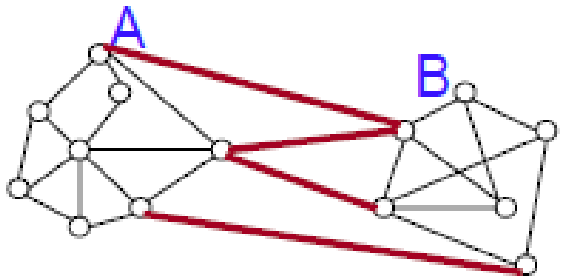


$G = \{V, E\}$

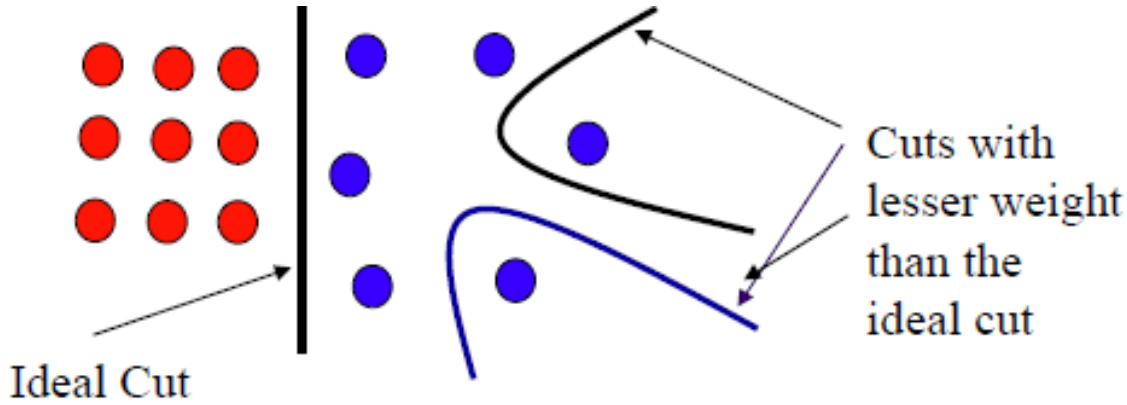
Partitioning a graph into two clusters

Min-cut: Partition graph into two sets A and B such that weight of edges connecting vertices in A to vertices in B is minimum.

$$\text{cut}(A, B) := \sum_{i \in A, j \in B} w_{ij}$$



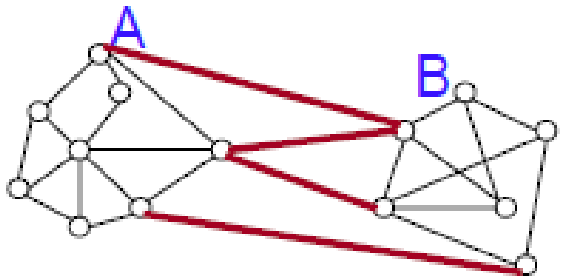
- Easy to solve O(VE) algorithm
- Not satisfactory partition – often isolates vertices



Partitioning a graph into two clusters

Partition graph into two sets A and B such that weight of edges connecting vertices in A to vertices in B is minimum & size of A and B are very similar.

$$\text{cut}(A, B) := \sum_{i \in A, j \in B} w_{ij}$$



Normalized cut:

$$\text{Ncut}(A, B) := \text{cut}(A, B)\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)$$

$$\text{vol}(A) = \sum_{i \in A} d_i$$

But NP-hard to solve!!

Spectral clustering is a relaxation of these.

Normalized Cut and Graph Laplacian

$$\text{Ncut}(A, B) := \text{cut}(A, B)\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)$$

$D = \text{diag}\{d_1, d_2, \cdots, d_n\}$

$L = D - W.$

Let $\mathbf{f} = [f_1\ f_2\ \ldots\ f_n]^T$ with $f_i = \begin{cases} \frac{1}{\text{vol}(A)} & \text{if } i \in A \\ -\frac{1}{\text{vol}(B)} & \text{if } i \in B \end{cases}$

$$\mathbf{f}^T\mathbf{L}\mathbf{f} = \sum_{ij} w_{ij}(\mathbf{f}_i - \mathbf{f}_j)^2 = \sum_{i\in A, j\in B} w_{ij}\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)^2$$

$$\mathbf{f}^T\mathbf{D}\mathbf{f} = \sum_j d_i \mathbf{f}_i^2 = \sum_{i\in A} \frac{d_i}{\text{vol}(A)^2} + \sum_{j\in B} \frac{d_i}{\text{vol}(B)^2} = \frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}$$

$$\text{Ncut}(A, B) = \frac{\mathbf{f}^T\mathbf{L}\mathbf{f}}{\mathbf{f}^T\mathbf{D}\mathbf{f}}$$

Normalized Cut and Graph Laplacian

$$\min \text{Ncut}(A,B) = \min \frac{\mathbf{f}^T \mathbf{L} \mathbf{f}}{\mathbf{f}^T \mathbf{D} \mathbf{f}}$$

$f^T D 1 = \sum_{j=1}^{n} f_j d_j = \sum_{j \in A} \frac{1}{vol(A)} d_j + \sum_{j \in B} \frac{-1}{vol(B)} d_j = \frac{vol(A)}{vol(A)} - \frac{vol(B)}{vol(B)} = 0$

where $f = [f_1\ f_2 \ldots f_n]^T$ with $f_i = \begin{cases} \frac{1}{\text{vol}(A)} & \text{if } i \in A \\ -\frac{1}{\text{vol}(B)} & \text{if } i \in B \end{cases}$

Relaxation: $\min \frac{\mathbf{f}^T \mathbf{L} \mathbf{f}}{\mathbf{f}^T \mathbf{D} \mathbf{f}}$ s.t. $f^T D 1 = 0$ 使f可以连续取值

Solution: f – second eigenvector of generalized eval problem

$$\boxed{\mathbf{L}\mathbf{f} = \lambda \mathbf{D}\mathbf{f}}$$

就是使λ最小的f. ($\lambda \neq 0$)

Obtain cluster assignments by thresholding f at 0

Approximation of Normalized cut

$$\text{Ncut}(A, B) := \text{cut}(A, B)\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)$$

Let *f* be the eigenvector corresponding to the second smallest eval of the generalized eval problem.

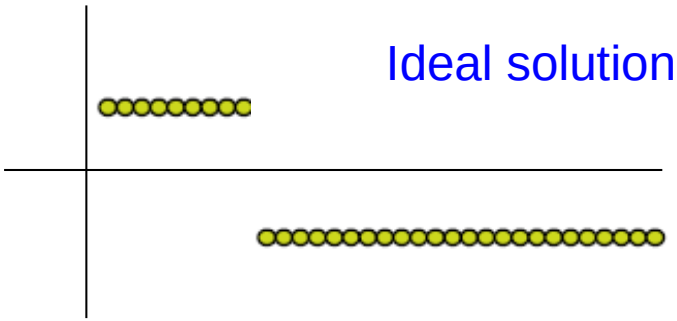
$$\mathbf{Lf} = \lambda \mathbf{Df}$$

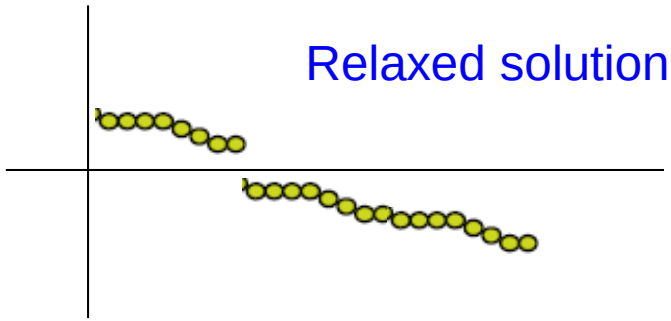
Equivalent to eigenvector corresponding to the second smallest eval of the normalized Laplacian $L' = D^{-1}L = I - D^{-1}W$

Recover binary partition as follows:

$i \in A$ *if* $f_i \geq 0$

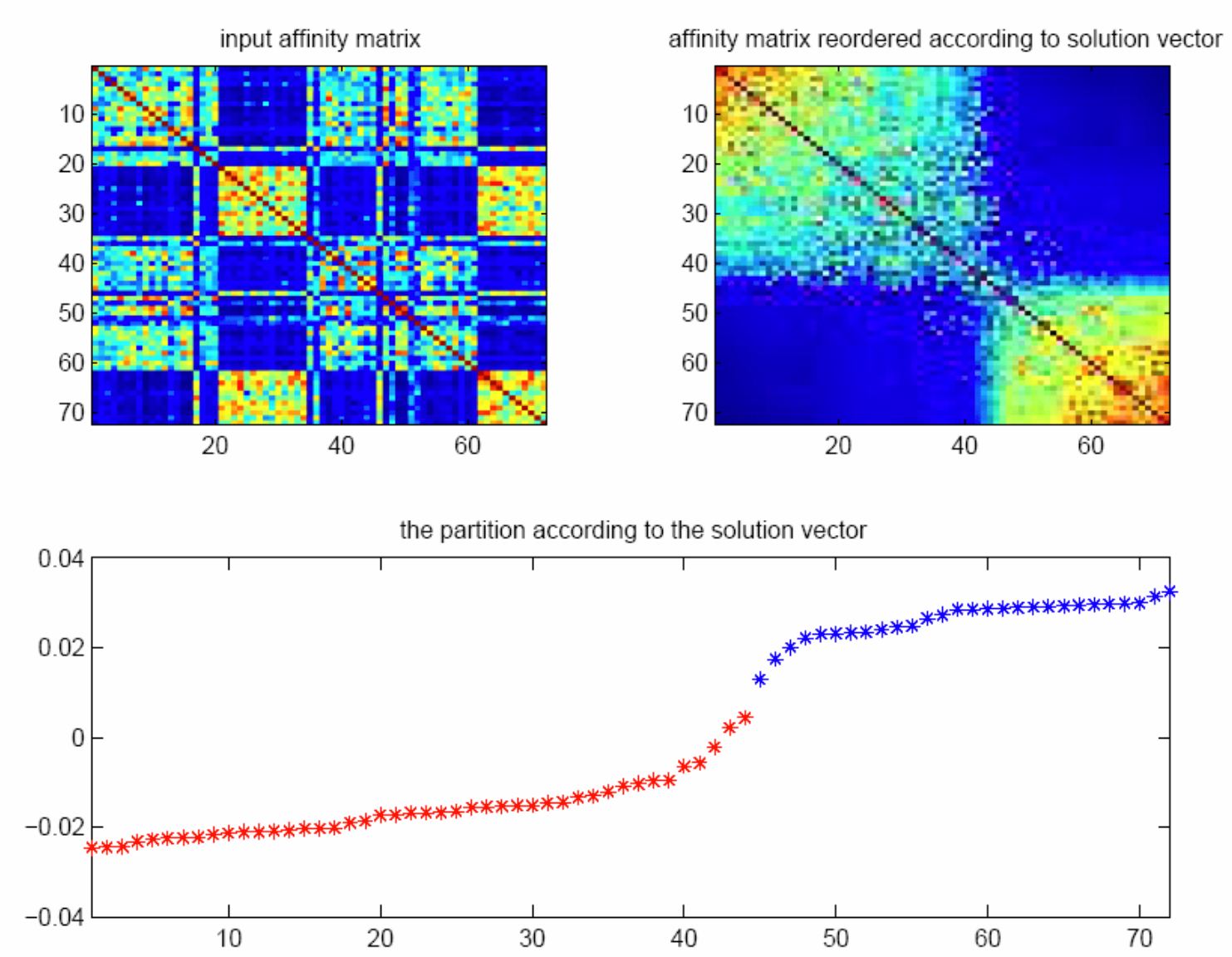
$i \in B$ *if* $f_i < 0$





Example

Xing et al 2001



How to partition a graph into k clusters?

Spectral Clustering Algorithm

Input: Similarity matrix W, number k of clusters to construct

- Build similarity graph
- Compute the first k eigenvectors $v_1, \ldots, v_k$ of the matrix

$$\begin{cases} L & \text{for unnormalized spectral clustering} \\ L' & \text{for normalized spectral clustering} \end{cases}$$

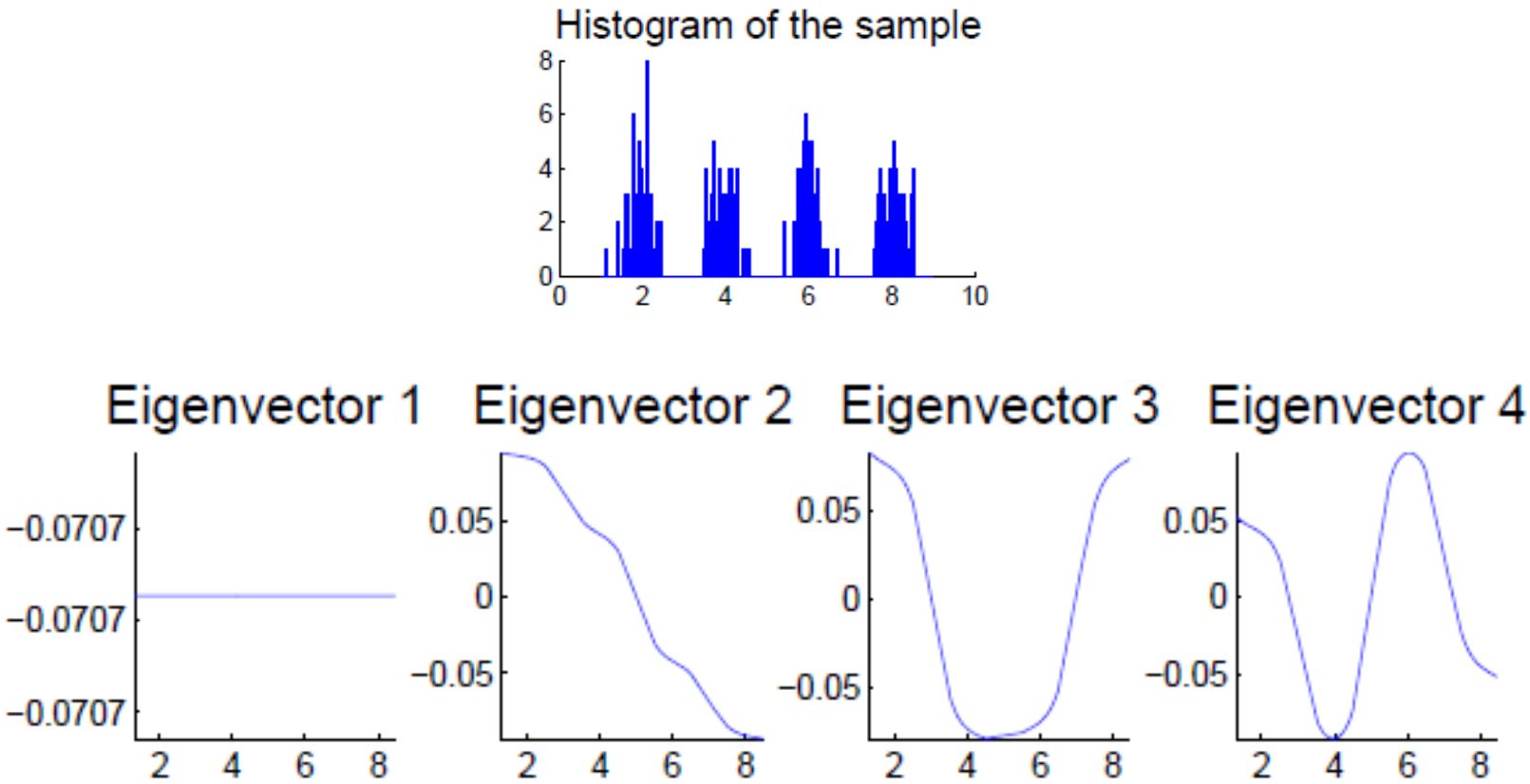
- Build the matrix $V \in \mathbb{R}^{n \times k}$ with the eigenvectors as columns
- Interpret the rows of V as new data points $Z_i \in \mathbb{R}^k$

	v_1	v_2	v_3
Z_1	v_{11}	v_{12}	v_{13}
$\vdots$	$\vdots$	$\vdots$	$\vdots$
Z_n	v_{n1}	v_{n2}	v_{n3}

Dimensionality Reduction
n x n → n x k

- Cluster the points Z_i with the k-means algorithm in $\mathbb{R}^k$.

Eigenvectors of Graph Laplacian

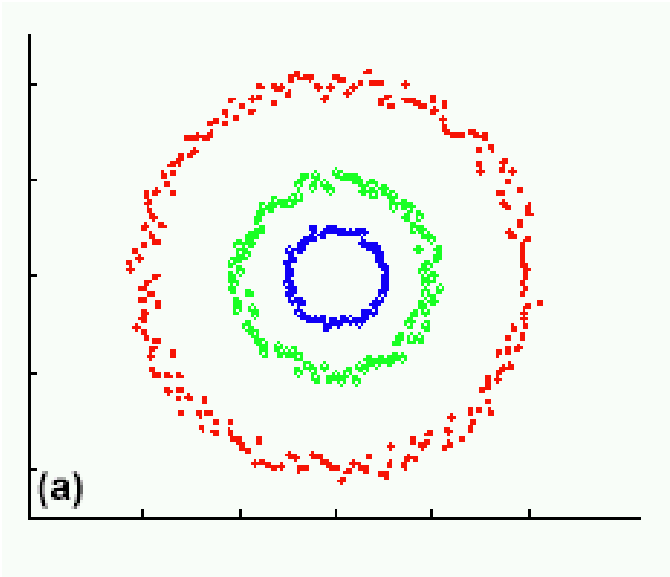


- 1st Eigenvector is the all ones vector ***1*** (if graph is connected)
- 2nd Eigenvector thresholded at 0 separates first two clusters from last two
- k-means clustering of the 4 eigenvectors identifies all clusters

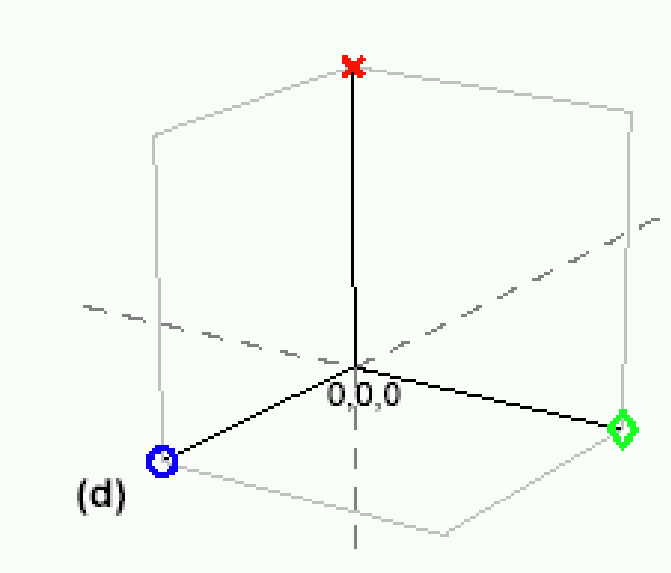
Why does it work?

Data are projected into a lower-dimensional space (the spectral/eigenvector domain) where they are easily separable, say using k-means.

Original data



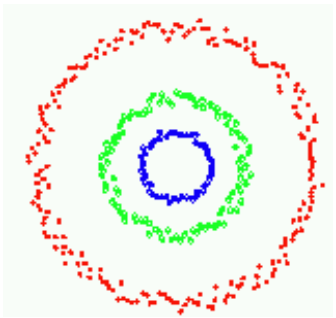
Projected data



Graph has 3 connected components – first three eigenvectors are constant (all ones) on each component.

Understanding Spectral Clustering

- If graph is connected, first Laplacian evec is constant (all 1s)
- If graph is disconnected (k connected components), Laplacian is block diagonal and first k Laplacian evecs are:



OR

$$L = \begin{bmatrix} L_1 & & 0 \\ & L_2 & \\ 0 & & L_3 \end{bmatrix}$$

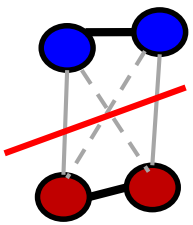
$$\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

First three eigenvectors

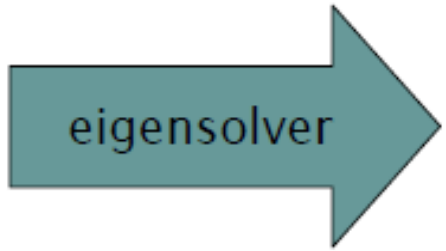
Understanding Spectral Clustering

- Is all hope lost if clusters don't correspond to connected components of graph? No!
- If clusters are connected loosely (small off-block diagonal enteries), then 1st Laplacian even is all 1s, but second evec gets first cut (min normalized cut)

$$\text{Ncut}(A, B) := \text{cut}(A, B)\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)$$



1	1	.2	0
1	1	0	.1
.2	0	1	1
0	.1	1	1



.50
.50
.50
.50

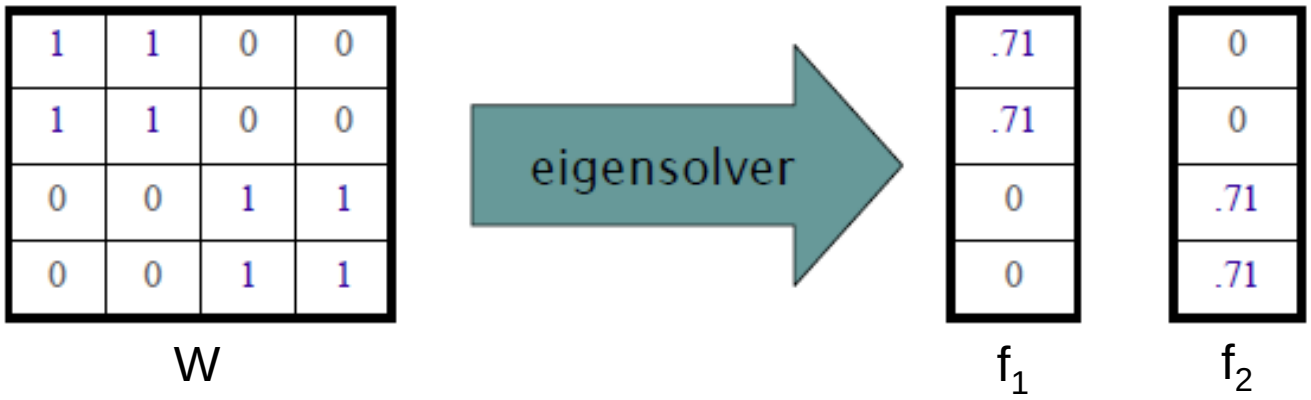
.47
.52
-.47
-.52

1st evec is constant since graph is connected

Sign of 2nd evec indicates blocks

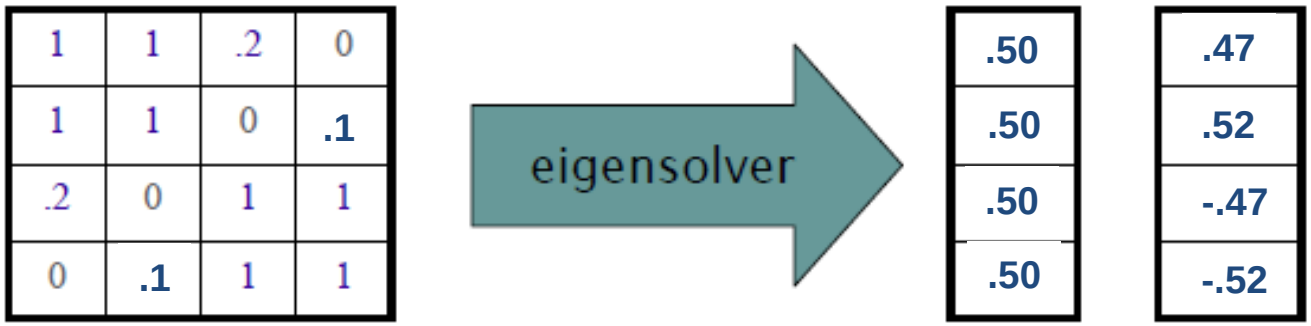
Why does it work?

Block weight matrix (disconnected graph) results in block eigenvectors:



Normalized to have unit norm

Slight perturbation does not change span of eigenvectors significantly:



1st evec is constant since graph is connected

Sign of 2nd evec indicates blocks

Why does it work?

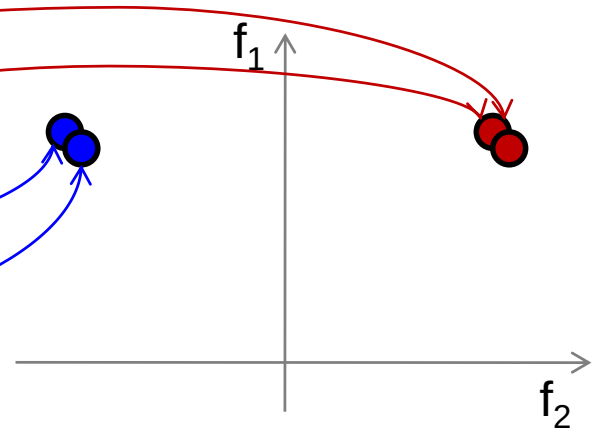
Can put data points into blocks using eigenvectors:

1	1	.2	0
1	1	0	.1
.2	0	1	1
0	.1	1	1

W

.50	.47
.50	.52
.50	-.47
.50	-.52

f_1 f_2



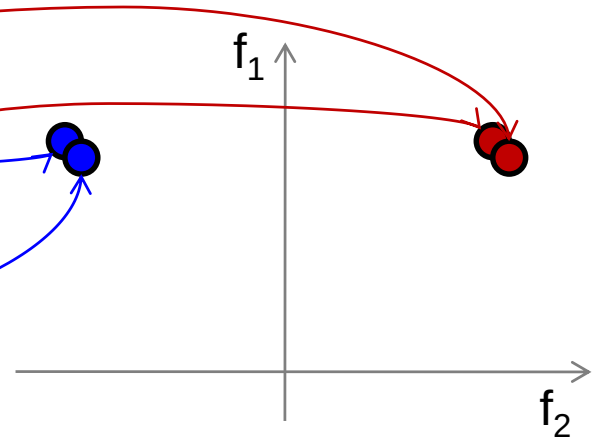
Embedding is same regardless of data ordering:

1	.2	1	0
.2	0	1	1
1	1	0	.1
0	1	.1	1

W

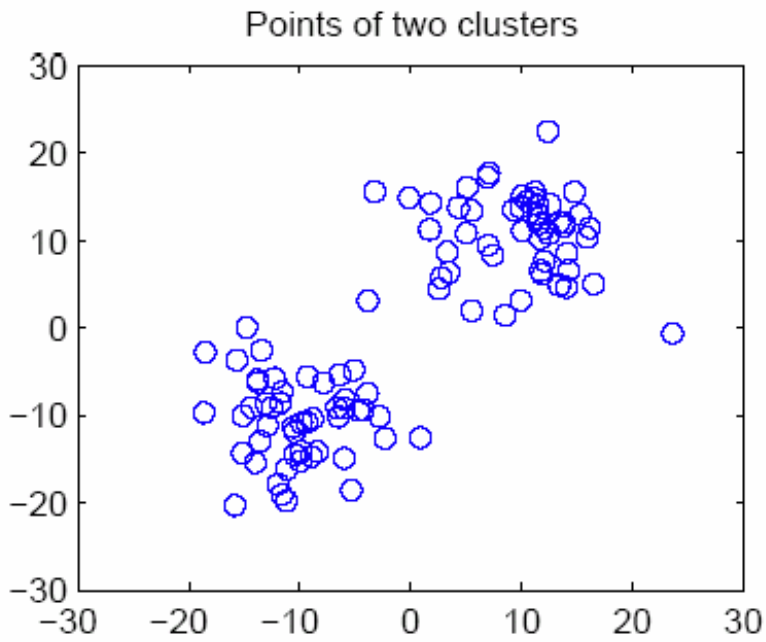
.50	.47
.50	-.47
.50	.52
.50	-.52

f_1 f_2

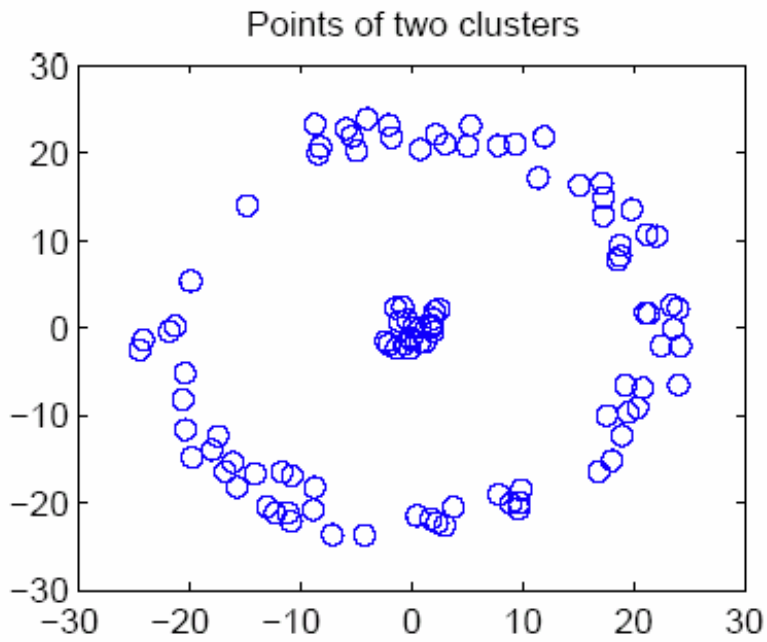


k-means vs Spectral clustering

Applying k-means to laplacian eigenvectors allows us to find cluster with non-convex boundaries.



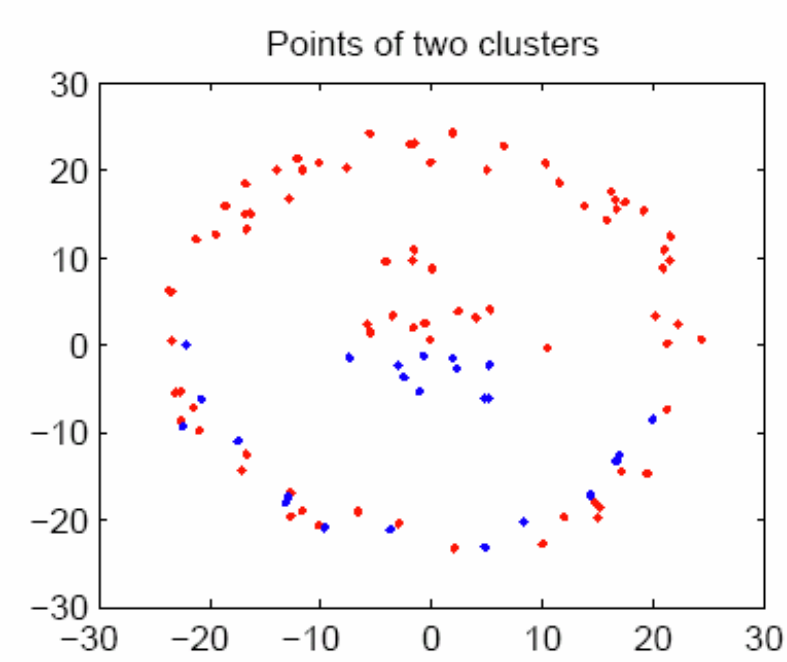
Both perform same



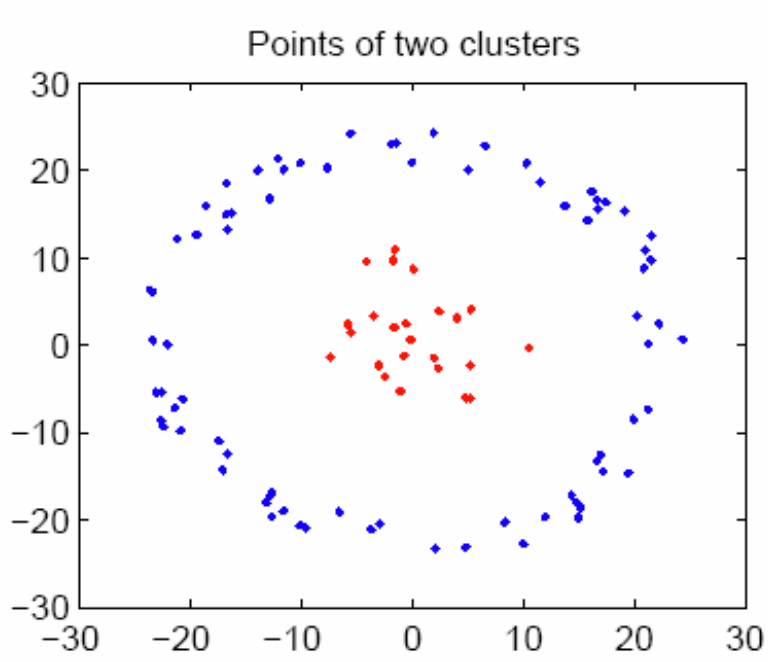
Spectral clustering is superior

k-means vs Spectral clustering

Applying k-means to laplacian eigenvectors allows us to find cluster with non-convex boundaries.



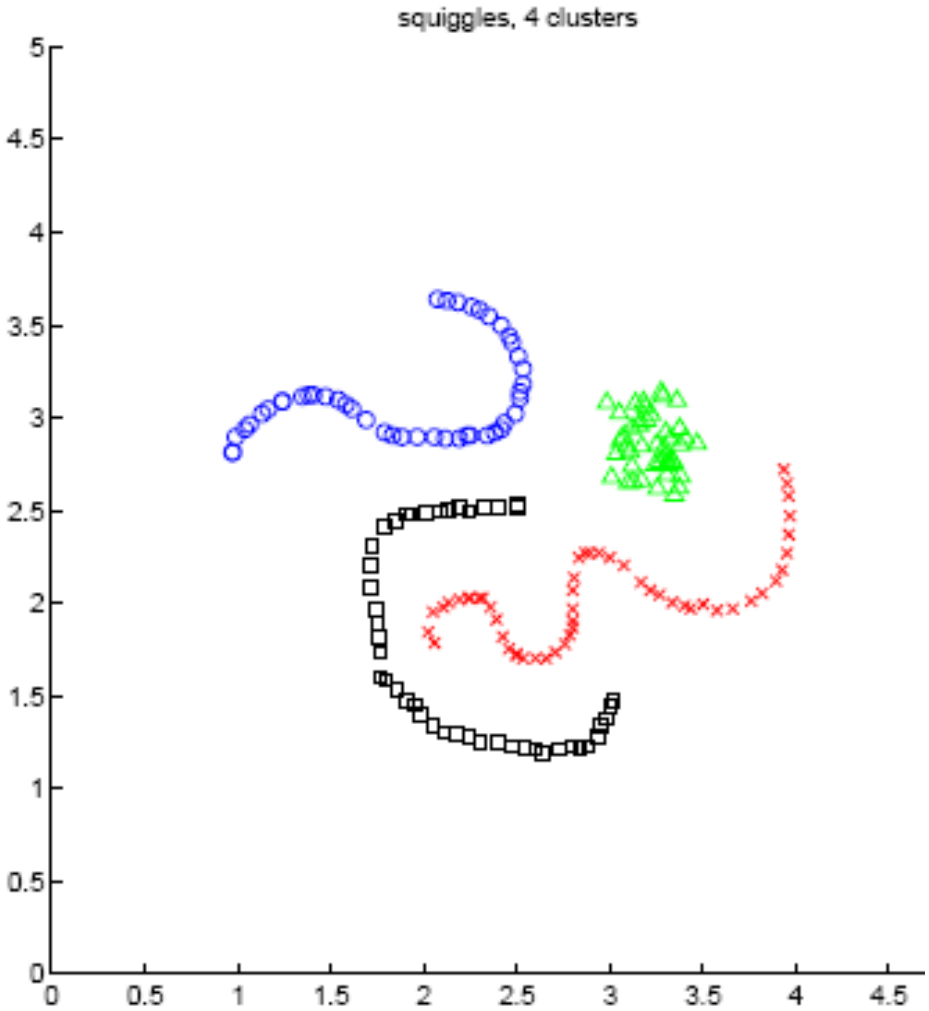
k-means output

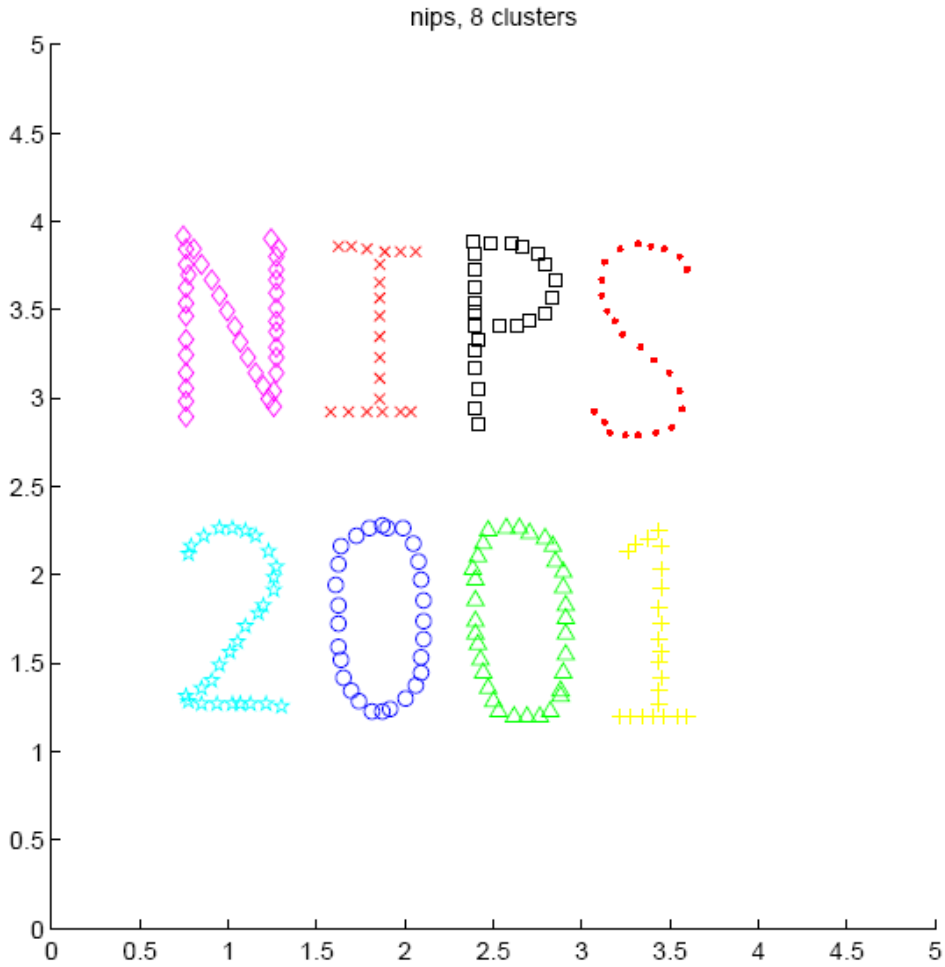


Spectral clustering output

Examples

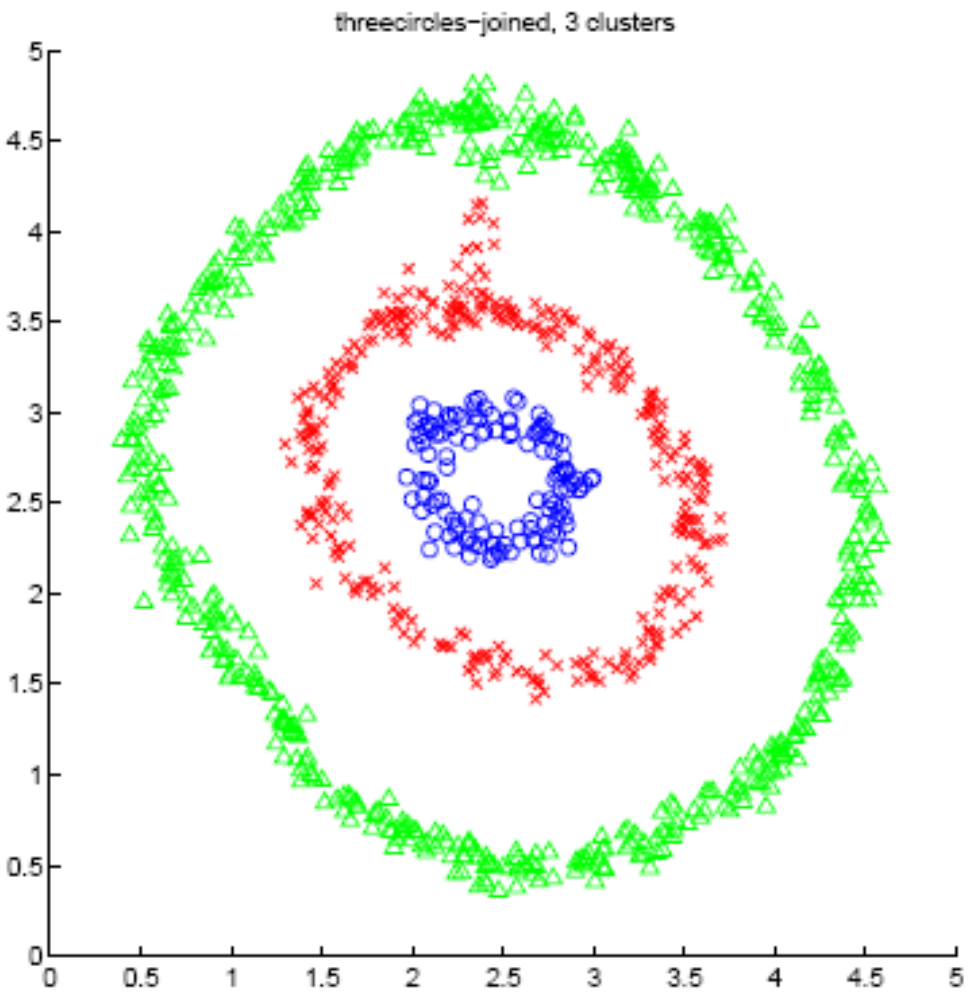
Ng et al 2001

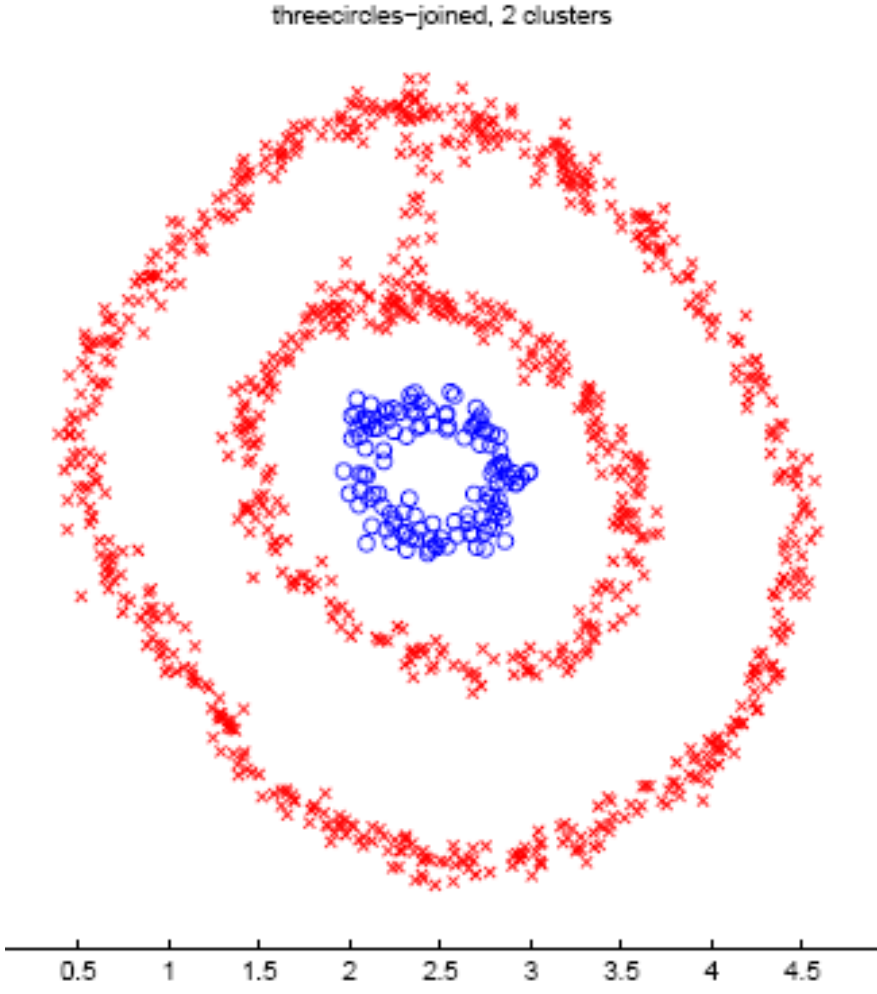




Examples (Choice of k)

Ng et al 2001



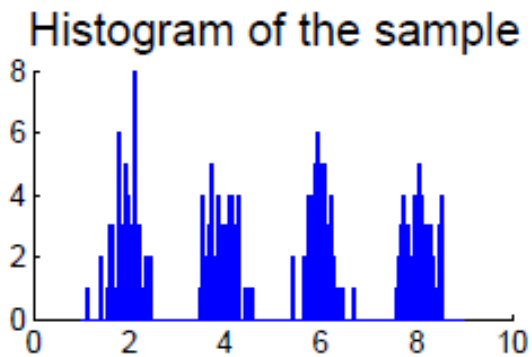


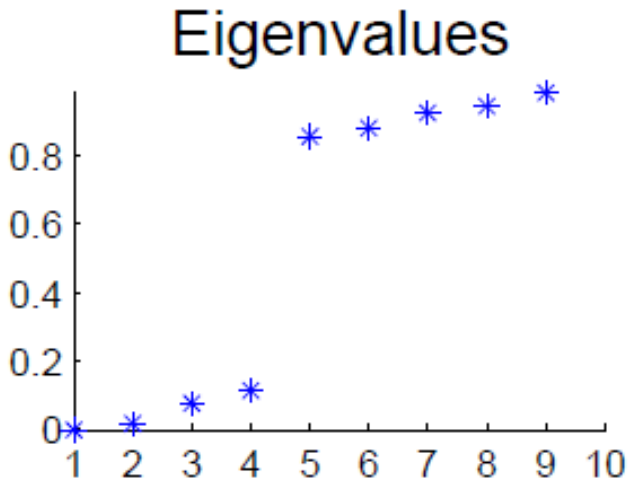
Some Issues

- Choice of number of clusters k

  Most stable clustering is usually given by the value of k that maximizes the eigengap (difference between consecutive eigenvalues)

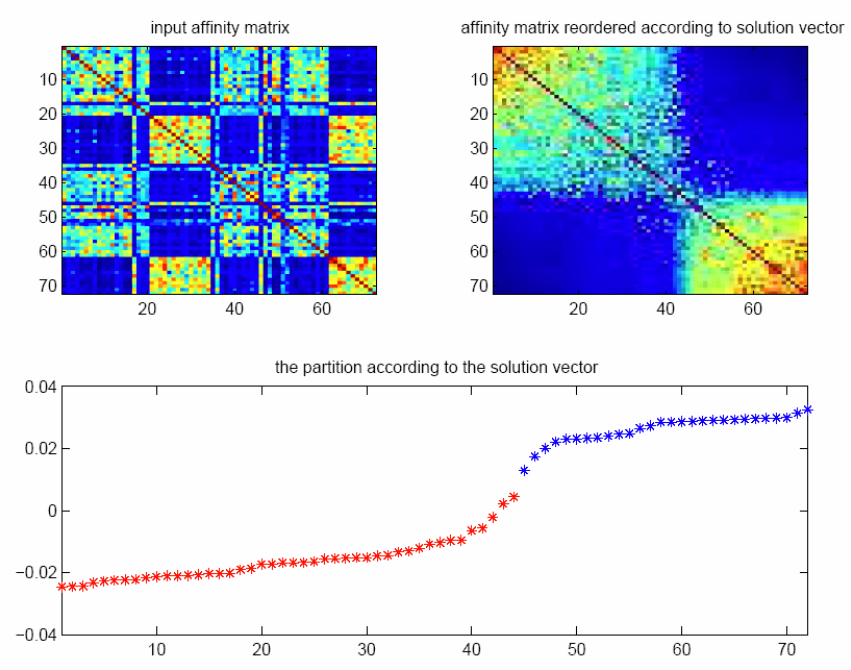
$$\Delta_k = \left| \lambda_k - \lambda_{k-1} \right|$$



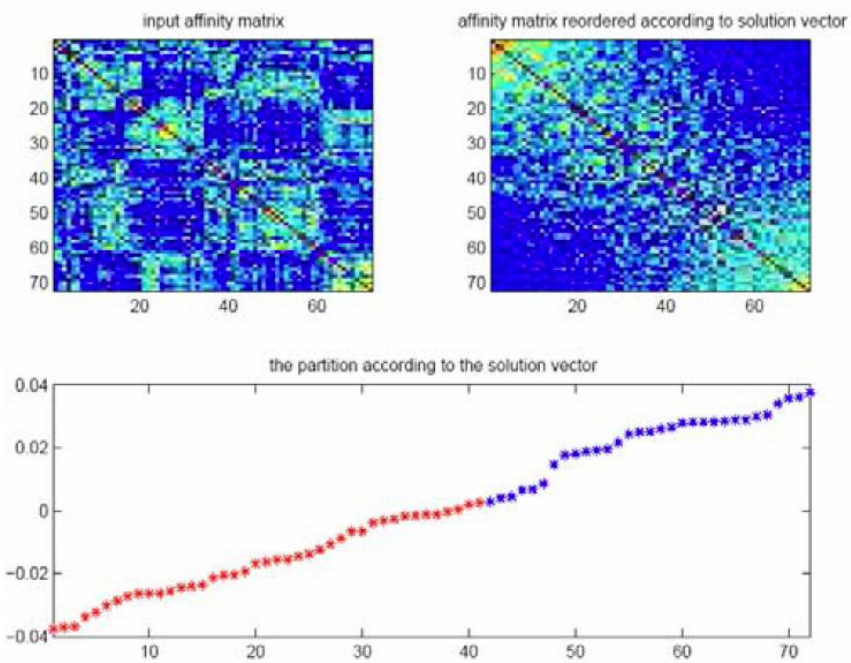


Some Issues

- Choice of number of clusters k
- Choice of similarity
  - choice of kernel
  - for Gaussian kernels, choice of σ



Good similarity measure



Poor similarity measure

Content

The K-means objective function

- Let $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K$ be the K cluster centroids (means)
- Let $r_{ik} \in \{0, 1\}$ be indicator denoting whether point $\mathbf{x}_i$ belongs to cluster k
- K-means objective minimizes the total distortion (sum of distances of points from their cluster centers)

$$J(\boldsymbol{\mu}, \mathbf{r}) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \tag{2}$$

- Note: Exact optimization of the K-means objective is NP-hard
- The K-means algorithm is a heuristic that converges to a local optimum

The kernel K-means objective function

- Let $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K$ be the K cluster centroids (means)
- Let $r_{ik} \in \{0, 1\}$ be indicator denoting whether point $\mathbf{x}_i$ belongs to cluster k
- Kernel K-means objective minimizes the total distortion (sum of distances of points from their cluster centers)

$$J(\boldsymbol{\mu}, \mathbf{r}) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|\phi(\mathbf{x}_i) - \boldsymbol{\mu}_k\|^2 \tag{3}$$

- Note: $\phi(\cdot) : \mathcal{X} \mapsto \mathcal{F}$为将原始样本映射到高维空间的特征映射，且$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$

$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^n r_{ik}\phi(\mathbf{x}_i)}{\sum_{i=1}^n r_{ik}}$，核K-means的目标函数变为

$$
\begin{aligned}
J(\boldsymbol{\mu}, \mathbf{r}) &= \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}\|\phi(\mathbf{x}_i) - \boldsymbol{\mu}_k\|^2 \\
&= \sum_{k=1}^{K}\sum_{\phi(\mathbf{x}_i)\in\mathcal{C}_k} \|\phi(\mathbf{x}_i) - \boldsymbol{\mu}_k\|^2 \\
&= \sum_{k=1}^{K}\sum_{\phi(\mathbf{x}_i)\in\mathcal{C}_k} \left(\kappa(\mathbf{x}_i, \mathbf{x}_i) - \frac{2}{n_k}\sum_{\phi(\mathbf{x}_j)\in\mathcal{C}_k} \kappa(\mathbf{x}_i, \mathbf{x}_j)\right. \\
&\quad \left. + \frac{1}{n_k^2}\sum_{\substack{\phi(\mathbf{x}_j)\in\mathcal{C}_k \\ \phi(\mathbf{x}_l)\in\mathcal{C}_k}} \kappa(\mathbf{x}_j, \mathbf{x}_l)\right) \\
&= \mathrm{Tr}(\mathbf{K}) - \mathrm{Tr}(\mathbf{L}^{\frac{1}{2}}\mathbf{R}^\top\mathbf{K}\mathbf{R}\mathbf{L}^{\frac{1}{2}})
\end{aligned}
\tag{4}
$$

其中$\mathbf{K}$是核矩阵

且$K_{ij} = \phi(\mathbf{x}_i)^\top\phi(\mathbf{x}_j)$, $\mathbf{L} = \mathrm{diag}([n_1^{-1}, n_2^{-1}, \cdots, n_K^{-1}])$，$R_{ik} = r_{ik}$

Kernel K-means (Cont.)

- 注意$\mathbf{R}\mathbf{1}_K = \mathbf{1}_n$(每个样本只属于某一个聚类)
- 定义$\mathbf{H} = \mathbf{R}\mathbf{L}^{\frac{1}{2}}$并允许$\mathbf{H}$取实数值，我们得到如下的优化问题

$$\min_{\mathbf{H}\in\mathbb{R}^{n\times K}} \ \mathrm{Tr}\left(\mathbf{K}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right) \ \ s.t. \ \ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_K, \tag{5}$$

- 当指定核矩阵$\mathbf{K}$时，上述优化问题等价于

$$\max_{\mathbf{H}\in\mathbb{R}^{n\times K}} \ \mathrm{Tr}\left(\mathbf{H}^\top\mathbf{K}\mathbf{H}\right) \ \ s.t. \ \ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_K, \tag{6}$$

- 矩阵$\mathbf{K}$与$\mathbf{H}\mathbf{H}^\top$之间的(全局)对齐？局部对齐会有更好的效果？
- $\mathbf{K}$对聚类起着至关重要的作用，如何选择最优的核矩阵？

谱聚类优化目标：

$$\min_{\mathbf{H}\in\mathbb{R}^{n\times K}} \ \mathrm{Tr}\left(\mathbf{H}^\top\mathbf{L}\mathbf{H}\right) \ \ s.t. \ \ \mathbf{H}^\top\mathbf{H}=\mathbf{I}_K, \tag{7}$$

其中$\mathbf{L}=\mathbf{D}-\mathbf{W}$, $\mathbf{D}=\mathrm{diag}(\mathbf{W1})$，且$W_{ij}=\exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}\right)$ if $\mathbf{x}_j\in\mathcal{N}(\mathbf{x}_i)$.

注意：谱聚类的权重矩阵$\mathbf{W}$必须要求为非负，否则并不能保证对应的Laplacian矩阵一定是半正定的。

核K均值优化目标：

$$\max_{\mathbf{H}\in\mathbb{R}^{n\times K}} \ \mathrm{Tr}\left(\mathbf{H}^\top\mathbf{K}\mathbf{H}\right) \ \ s.t. \ \ \mathbf{H}^\top\mathbf{H}=\mathbf{I}_K, \tag{8}$$

其中$K_{ij}=\exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}\right)$.

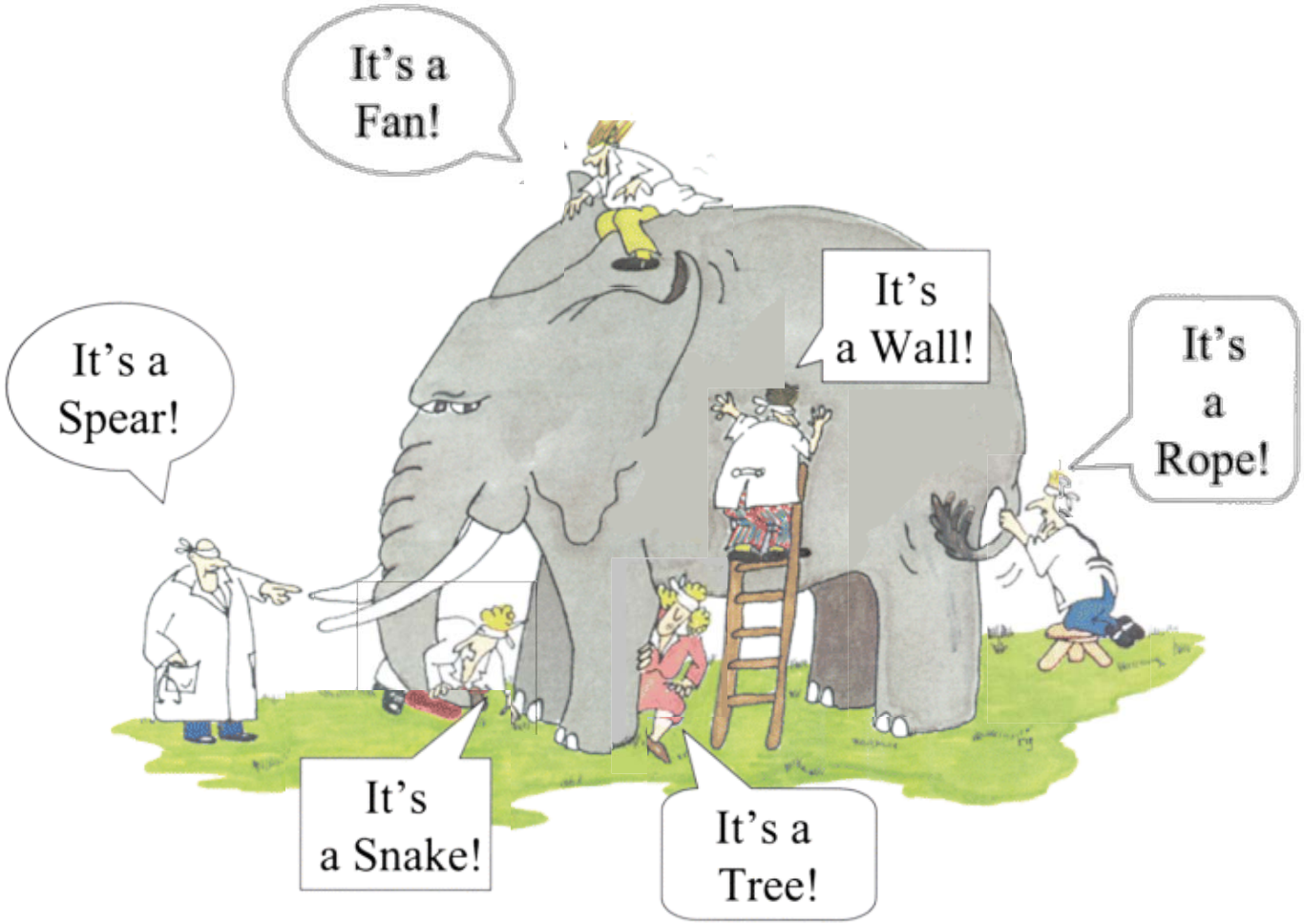
区别在哪里？样本间的邻居关系

与谱聚类之间的联系

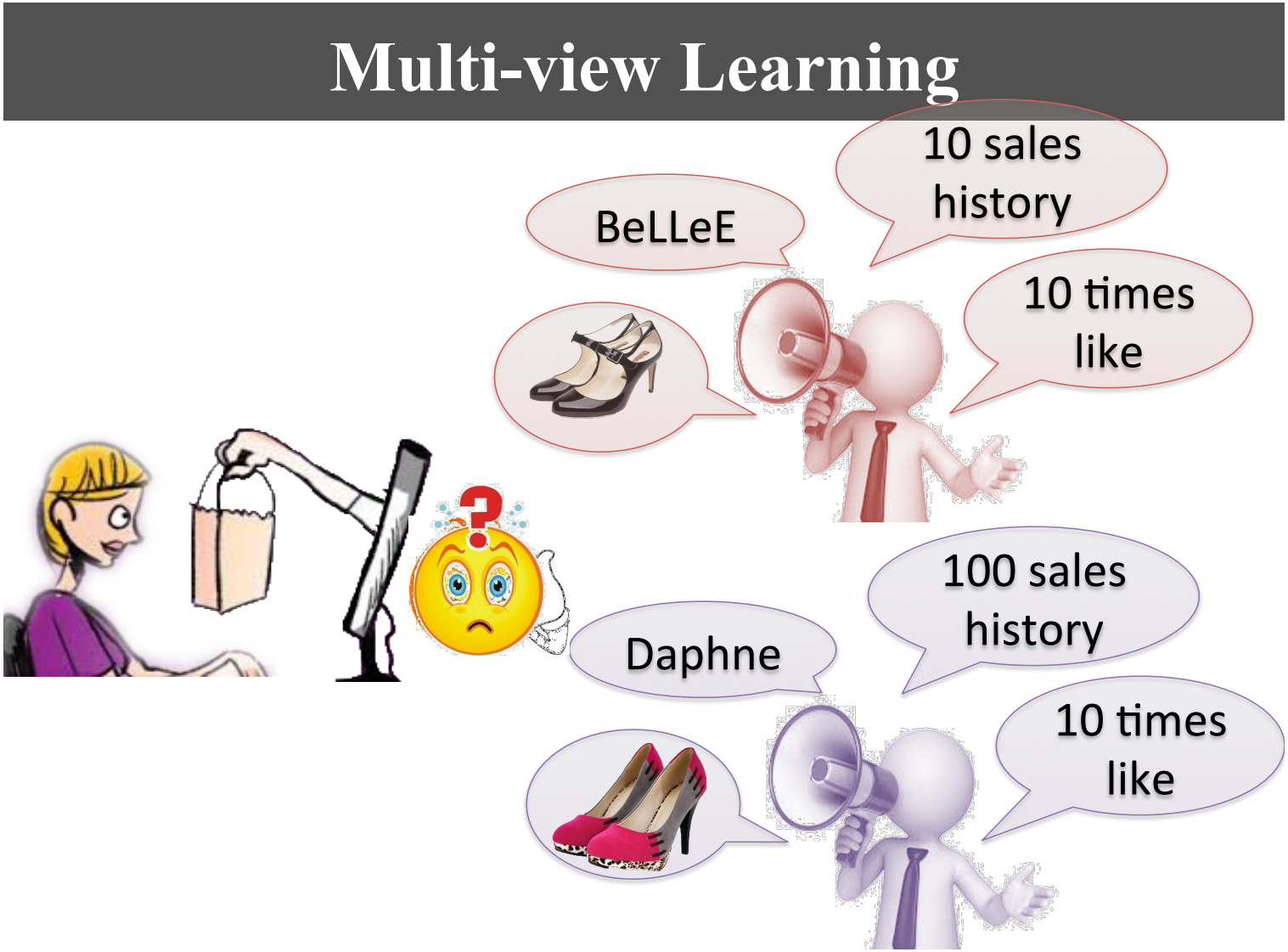
- 共同点：都是基于相似度矩阵的聚类算法
- 差异性：1)谱聚类利用了样本间的邻居信息，核均值聚类算法没有；2)谱聚类要求样本间权重非负，核均值聚类算法没有
- 如何利用样本间的邻居信息以改进核均值聚类算法的性能？
- 开放性问题：如何学习"最优"相似度矩阵？如何有效处理低质量数据(有缺失、有噪声等)

Content

多视图学习—案例一



Multi-view learning is necessary!


Multi-view Learning
BeLLeE
10 sales history
10 times like
Daphne
100 sales history
10 times like

The kernel alignment maximization for clustering can be fulfilled as,

$$\max_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\mu}\in\mathbb{R}_{+}^{m}} \frac{\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{H}\mathbf{H}^{\top}\rangle_{\mathrm{F}}}{\sqrt{\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{K}_{\boldsymbol{\mu}}\rangle_{\mathrm{F}}}} \quad s.t.\ \ \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}_k,\ \boldsymbol{\mu}^{\top}\mathbf{1}_m = 1, \tag{9}$$

where $\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{H}\mathbf{H}^{\top}\rangle_{\mathrm{F}} = \mathrm{Tr}(\mathbf{K}_{\boldsymbol{\mu}}\mathbf{H}\mathbf{H}^{\top})$, $\langle \mathbf{K}_{\boldsymbol{\mu}}, \mathbf{K}_{\boldsymbol{\mu}}\rangle_{\mathrm{F}} = \hat{\boldsymbol{\mu}}^{\top}\mathbf{M}\hat{\boldsymbol{\mu}}$ with $\hat{\boldsymbol{\mu}} = [\mu_1^2, \cdots, \mu_m^2]^{\top}$ and $\mathbf{M}$ is a positive semi-definite matrix with $M_{pq} = \mathrm{Tr}(\mathbf{K}_p^{\top}\mathbf{K}_q)$.

Eq.(**??**) is difficult to optimize since it is a fourth-order fractional optimization problem. Instead of maximizing the kernel alignment by solving a fractional optimization in Eq.(**??**), we turn to minimize the following,

$$\begin{aligned} \min_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\mu}\in\mathbb{R}_{+}^{m}} \quad & \mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}(\mathbf{I}_n-\mathbf{H}\mathbf{H}^{\top})\right)+\frac{\lambda}{2}\boldsymbol{\mu}^{\top}\mathbf{M}\boldsymbol{\mu} \\ s.t. \quad & \mathbf{H}^{\top}\mathbf{H}=\mathbf{I}_k,\ \boldsymbol{\mu}^{\top}\mathbf{1}_m=1, \end{aligned} \tag{10}$$

where λ is a trade-off parameter.

矩阵导出正则化的多视图聚类算法(AAAI2016)

（3）矩阵范数正则化的多核聚类算法研究

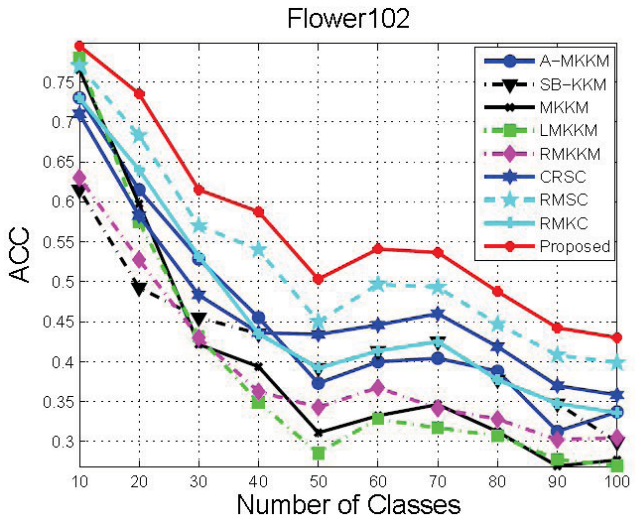
- **动机：现有多核聚类算法一方面强化相关性高的数据源被同时选中，另一方面弱化相关性低的数据源被同时选中。这会导致数据源不能被充分利用，导致不能令人满意的聚类性能。**
- **基本思想：提出了一种矩阵范数正则化的思想来克服上述问题。**

$$\min_{\mathbf{H}\in\mathbb{R}^{n\times k},\mu\in\mathbb{R}_+^m} \quad \mathrm{Tr}\left(\mathbf{K}_\mu(\mathbf{I}_n-\mathbf{H}\mathbf{H}^\top)\right)+\frac{\lambda}{2}\mu^\top\mathbf{M}\mu$$

$$s.t.\ \ \mathbf{H}^\top\mathbf{H}=\mathbf{I}_k,\ \ \mu^\top 1_m=1$$

Table 2: ACC comparison of different clustering algorithms on five benchmark data sets.

Datasets	A-MKKM	SB-KKM	MKKM	LMKKM	RMKKM	CRSC	RMSC	RMKC	Proposed
Flower17	51.03	42.06	45.37	42.94	48.38	52.72	53.90	52.35	**60.00**
Digital	88.75	75.40	47.00	47.00	40.45	84.80	90.40	88.90	**90.95**
ProteinFold	28.10	33.86	27.23	23.49	26.95	34.87	33.00	28.82	**37.32**
Flower102	45.44	42.53	40.27	39.22	39.37	46.99	52.56	46.28	**56.72**
Caltech102	40.79	40.26	40.36	38.12	36.39	41.36	39.86	41.77	**45.39**



- Xinwang Liu, Yong Dou, Jianping Yin, Lei Wang, En Zhu, : Absent Multiple Kernel Learning. Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, 2016, (CCF Rank A)

局部核对齐多视图聚类算法(IJCAI2016)

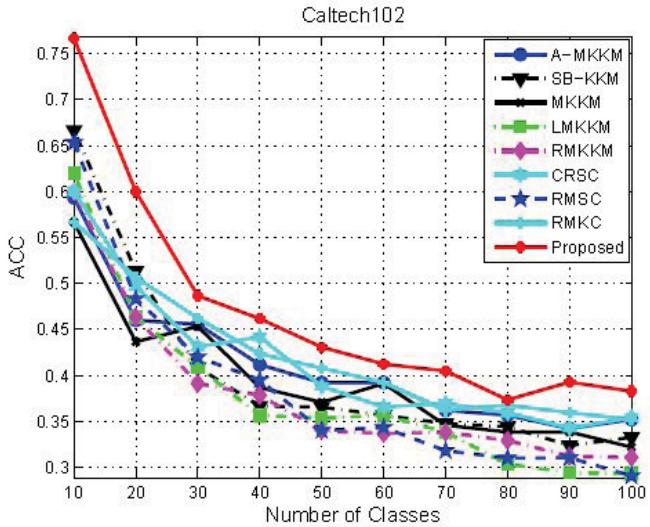
（4）基于局部核对齐最大化的多核聚类算法

- **动机：现有多核聚类算法通过全局核对齐来优化核组合系数。这种做法忽略了样本间潜在的差异性，导致不能令人满意的聚类性能。**
- **基本思想：提出了局部核对齐最大化的多核聚类算法来克服上述问题。**

$$\min_{\mathbf{H}\in\mathbb{R}^{n\times k},\boldsymbol{\mu}\in\mathbb{R}_{+}^{m}} \sum_{i=1}^{n}\left(\mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\mu}}\left(\mathbf{A}^{(i)}-\mathbf{A}^{(i)}\mathbf{H}\mathbf{H}^{\top}\mathbf{A}^{(i)}\right)\right)+\frac{\lambda}{2}\boldsymbol{\mu}^{\top}\mathbf{M}^{(i)}\boldsymbol{\mu}\right)$$
$$s.t.\ \ \mathbf{H}^{\top}\mathbf{H}=\mathbf{I}_k,\ \boldsymbol{\mu}^{\top}1_m=1$$

Datasets	A-MKKM	SB-KKM	MKKM	LMKKM	RMKKM	CRSC	RMSC	RMKC	Proposed
YALE	55.76	52.12	51.52	53.94	56.97	56.36	60.00	55.15	**62.42**
ProteinFold	28.10	33.86	27.23	23.49	30.98	34.87	33.00	28.82	**37.18**
Digital	88.75	75.40	47.00	47.00	40.45	84.80	90.40	88.90	**96.25**
Flower17	51.03	42.06	45.37	42.94	48.38	52.72	53.90	52.35	**63.75**
Flower102	27.29	33.13	21.96	22.57	28.17	37.26	32.97	33.54	**40.84**
Caltech102	35.56	33.14	34.77	27.97	29.67	33.33	31.50	35.56	**39.48**

Table 2: ACC comparison of different clustering algorithms on five benchmark data sets.



- **Xinwang Liu, Yong Dou, Jianping Yin, Lei Wang: Multiple Kernel Clustering with Local Kernel Alignment Maximization. IJCAI 2016 (CCF Rank A)**

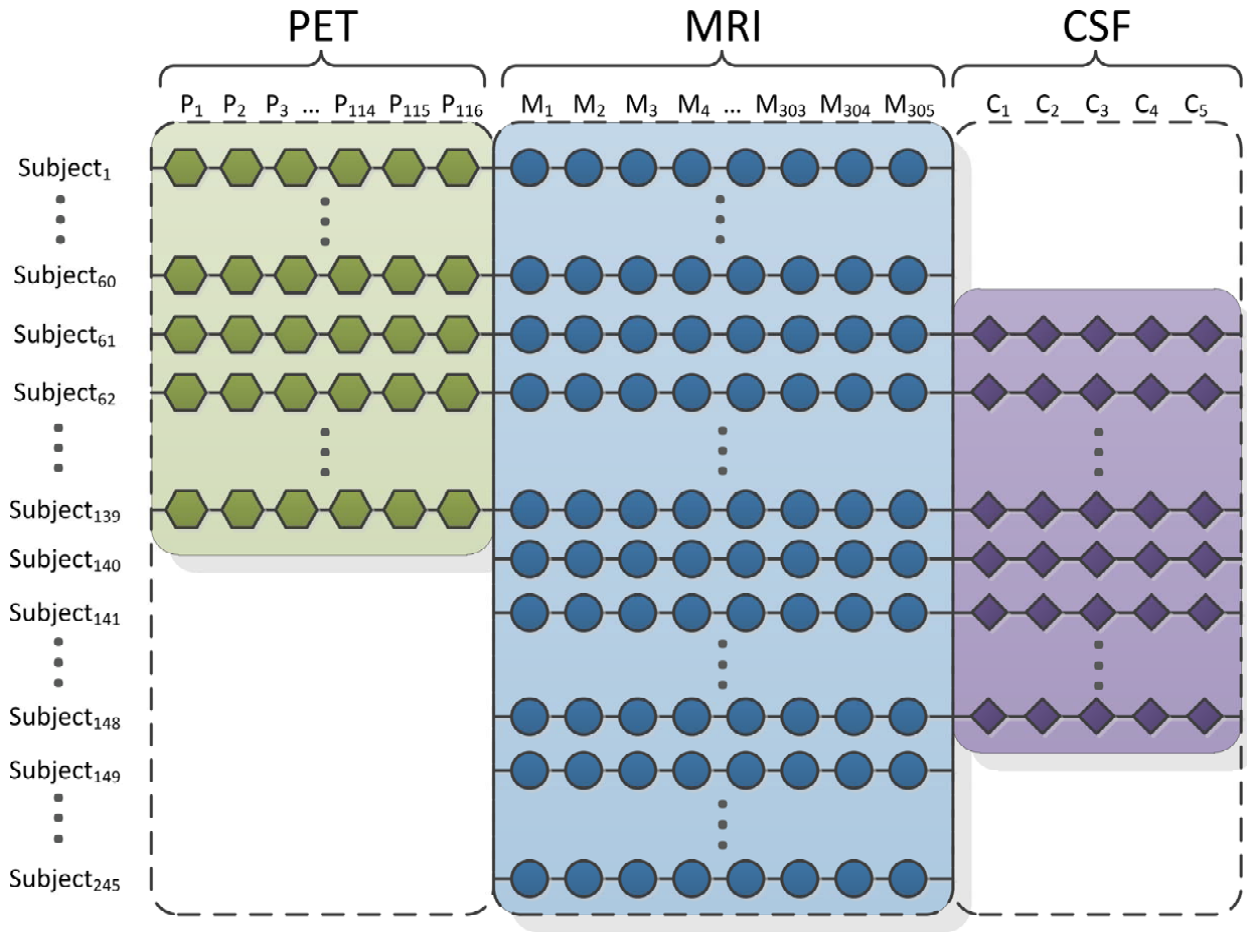


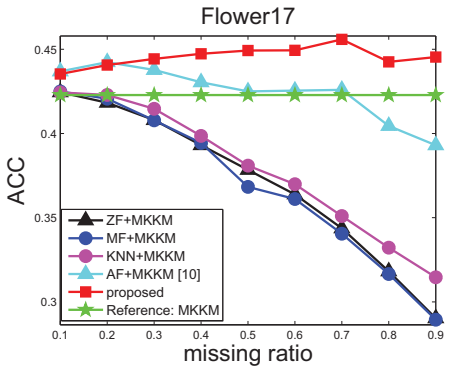
Figure: Figure **??** gives an illustration in the predication of Alzheimer's disease (AD), where a subset of measures of many objects are absent. Each subject of AD is represented by cerebrospinal fluid (CSF), magnetic resonance imaging (MRI) and positron emission tomography (PET). However, many subjects lack a subset of measures, as marked by the white areas.

优化目标(AAAI2017)

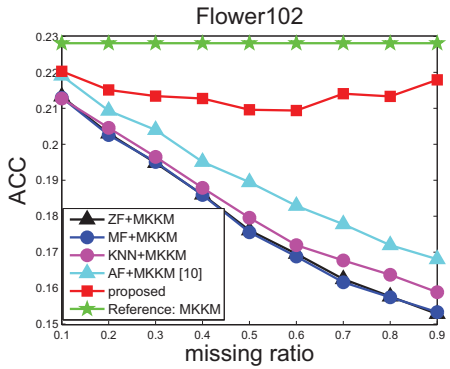
These two arguments motivate us to seek a more natural and reasonable manner to deal with the absence in multiple kernel clustering. That is to perform imputation and clustering in a joint way: 1) impute the absent kernels under the guidance of clustering; and 2) update the clustering with the imputed kernels. By this way, *the above two learning processes can be seamlessly coupled and they are allowed to negotiate with each other to achieve better clustering*.

$$\begin{aligned} \min_{\mathbf{H},\, \boldsymbol{\beta},\, \{\mathbf{K}_p\}_{p=1}^m} \quad & \mathrm{Tr}\left(\mathbf{K}_{\boldsymbol{\beta}}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\right) \\ s.t. \quad & \mathbf{H} \in \mathbb{R}^{n\times k},\ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k,\ \boldsymbol{\beta}^\top \mathbf{1}_m = 1,\ \beta_p \geq 0, \\ & \mathbf{K}_p(\mathbf{s}_p, \mathbf{s}_p) = \mathbf{K}_p^{(cc)},\ \mathbf{K}_p \succeq 0,\ \forall p, \end{aligned} \tag{11}$$

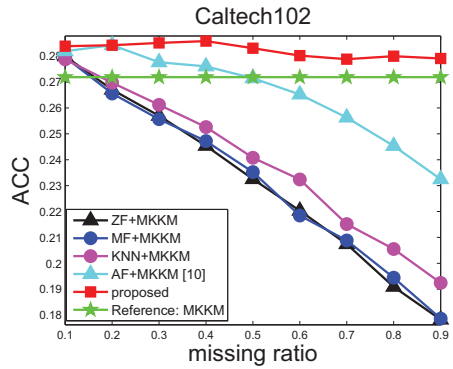
实验结果



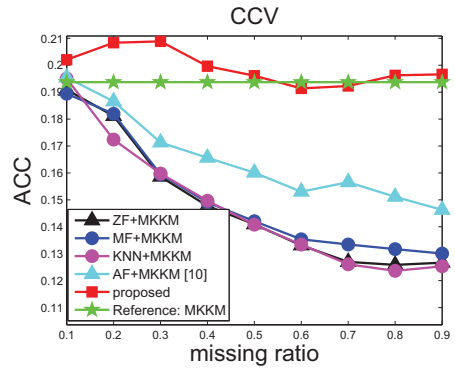
(a)



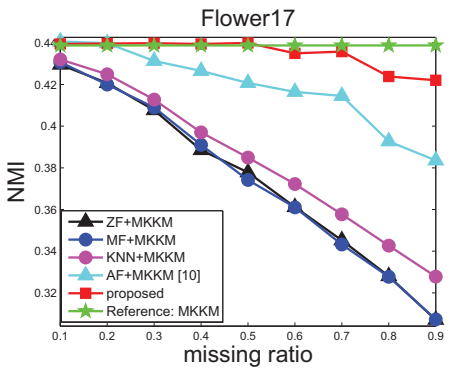
(b)



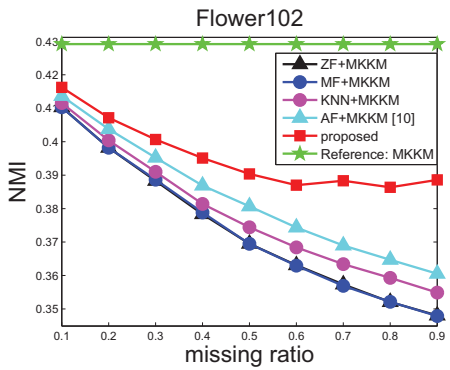
(c)



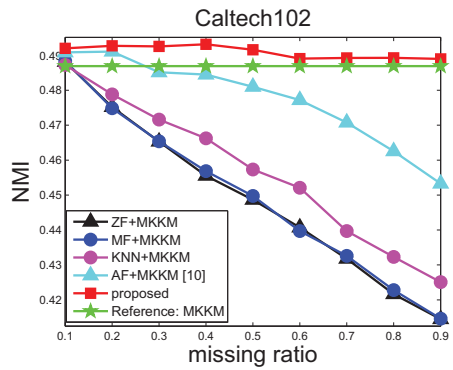
(d)



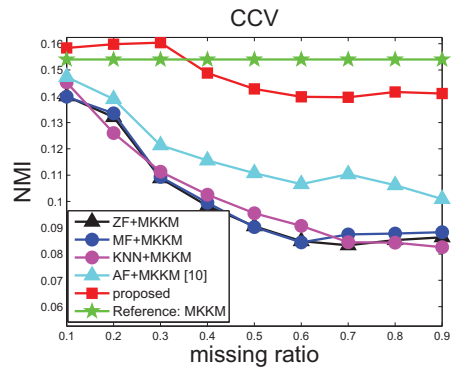
(e)



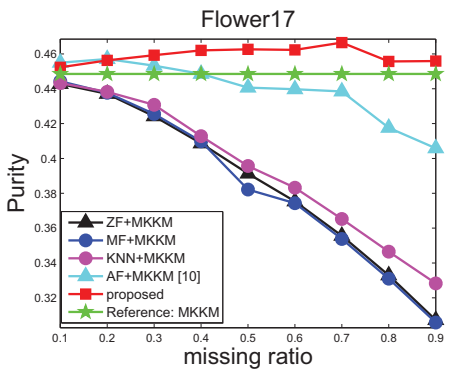
(f)



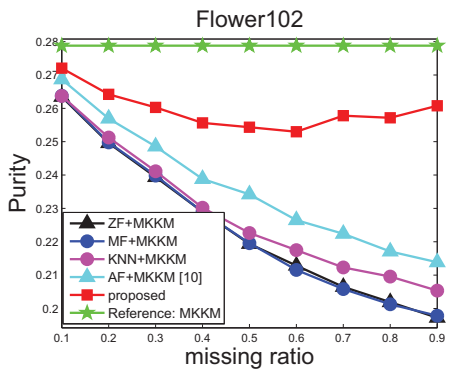
(g)



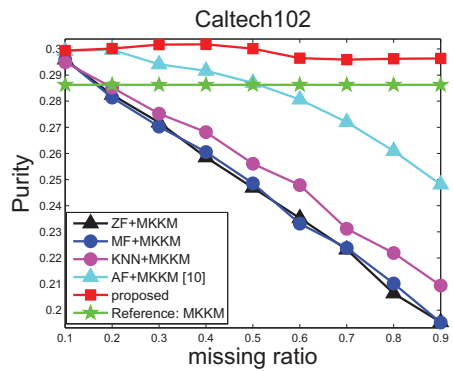
(h)



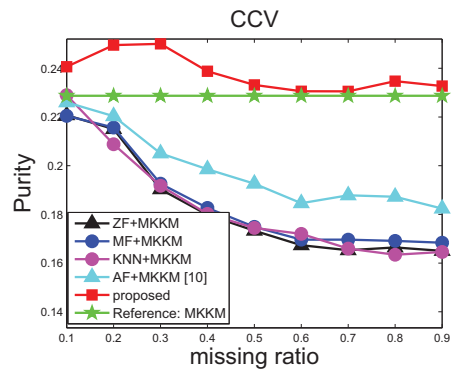
(i)



(j)



(k)



(l)

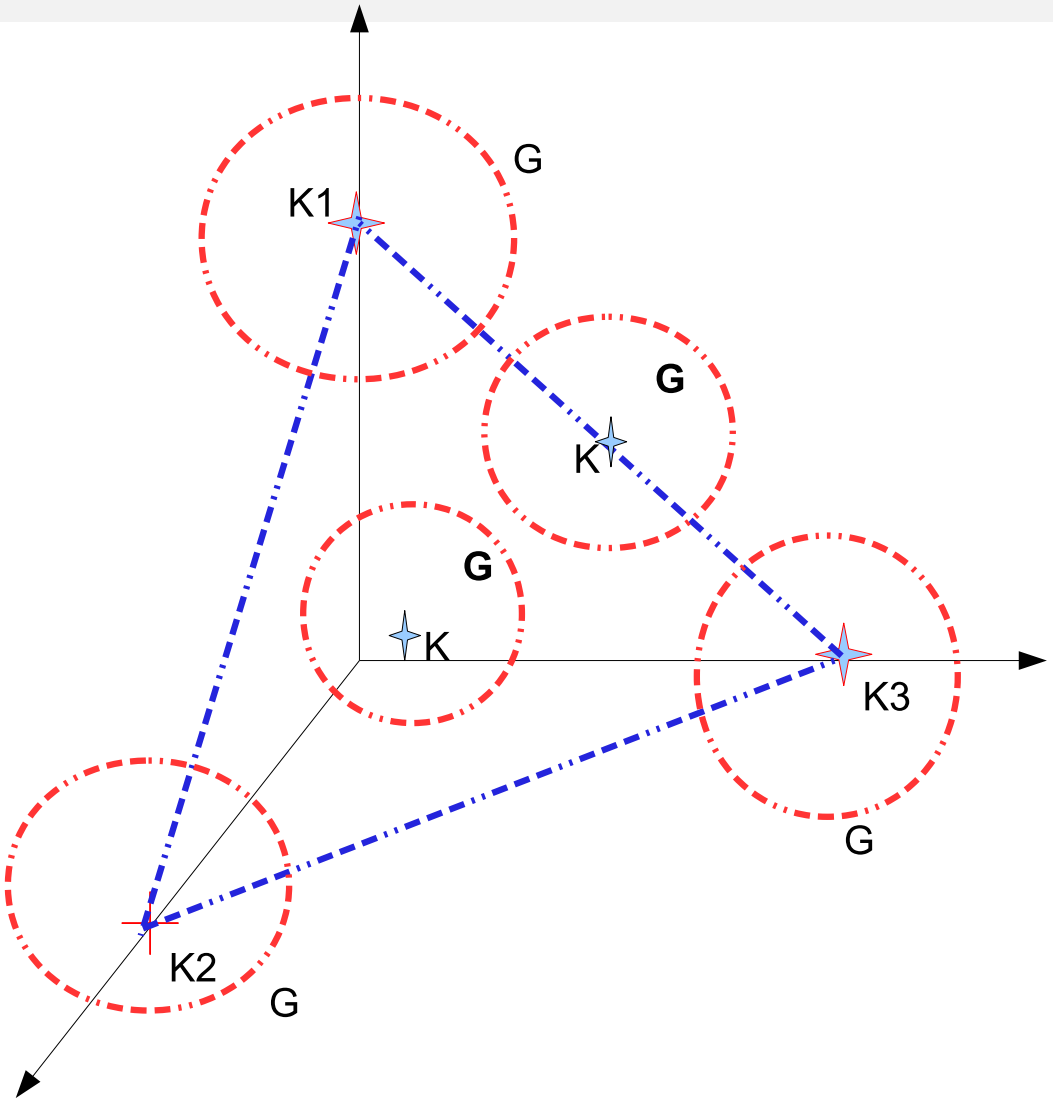


Figure: An illustration of the solution $\mathbf{K}$ in MKKM and $\mathbf{G}$ in our algorithm.

优化目标(AAAI2017)

$$
\begin{aligned}
&\min_{\mathbf{H},\boldsymbol{\gamma},\mathbf{G}} \operatorname{Tr}\Big(\mathbf{G}(\mathbf{I}_n - \mathbf{H}\mathbf{H}^\top)\Big) + \frac{\rho}{2}\|\mathbf{G} - \mathbf{K}_{\boldsymbol{\gamma}}\|_{\mathrm{F}}^2 + \frac{\lambda}{2}\boldsymbol{\gamma}^\top \mathbf{M}\boldsymbol{\gamma} \\
&\quad s.t.\ \ \mathbf{H} \in \mathbb{R}^{n\times k},\ \mathbf{H}^\top\mathbf{H} = \mathbf{I}_k, \\
&\qquad\ \ \boldsymbol{\gamma}^\top \mathbf{1}_m = 1,\ \gamma p \geq 0,\ \forall p, \\
&\qquad\ \ \mathbf{G} \succeq 0,
\end{aligned}
\tag{12}
$$

where $\mathbf{G}$ is an optimal kernel which is required to be PSD, $\mathbf{K}_{\boldsymbol{\gamma}} = \sum_{p=1}^{m} \gamma_p \mathbf{K}_p$, the distance of $\mathbf{G}$ and $\mathbf{K}_{\boldsymbol{\gamma}}$ is measured by $\|\mathbf{G} - \mathbf{K}_{\boldsymbol{\gamma}}\|_{\mathrm{F}}^2$, $\mathbf{M}$ is a matrix with element M_{pq} measuring the correlation between $\mathbf{K}_p$ and $\mathbf{K}_q$, and ρ, λ are regularization parameters.

实验结果

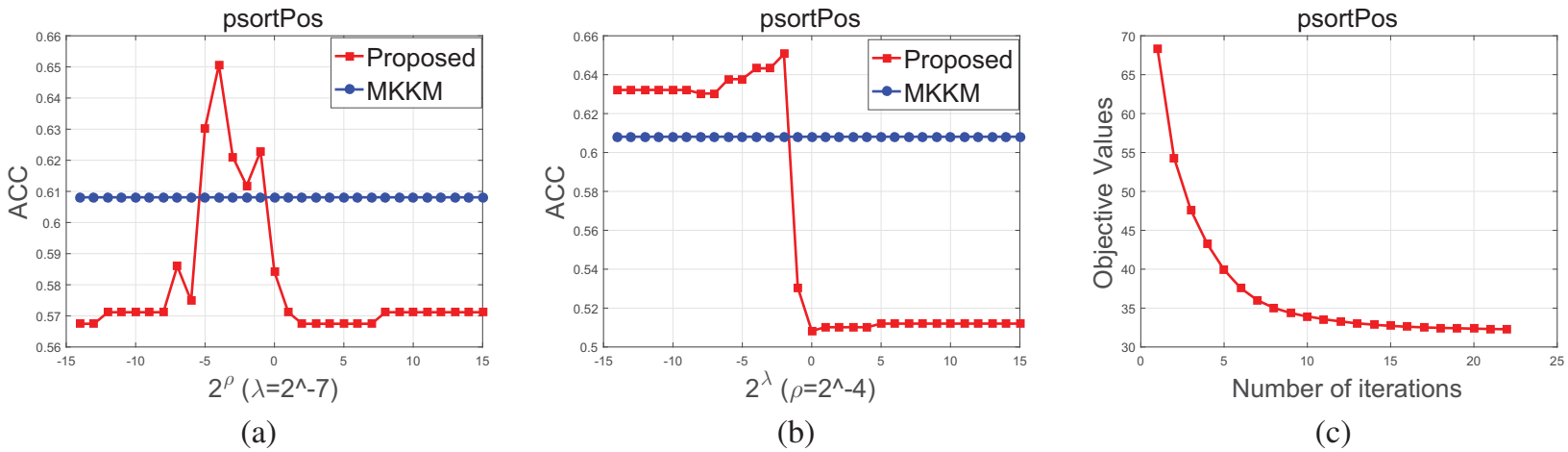


Figure 1: (a) The effect of ρ on clustering accuracy. (b) The effect of λ on clustering accuracy. (c) The objective value of our algorithm at each iteration.

讨论

- 自动修正多视图聚类算法
- 大规模聚类
- 增量聚类