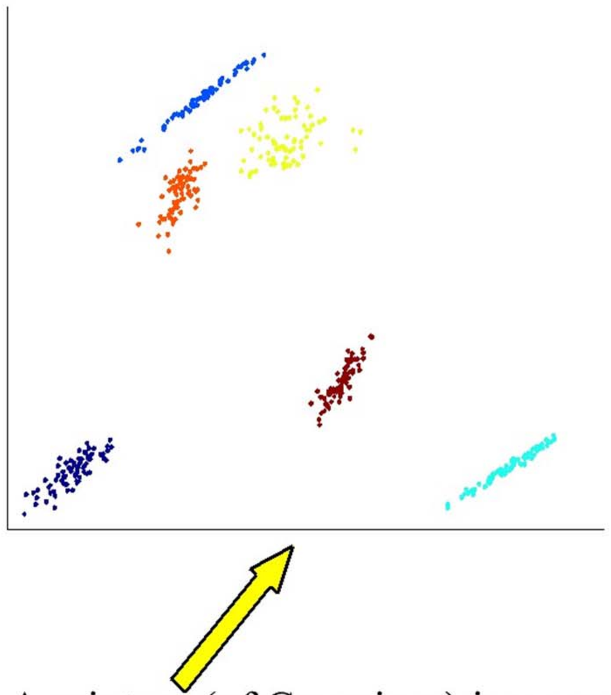
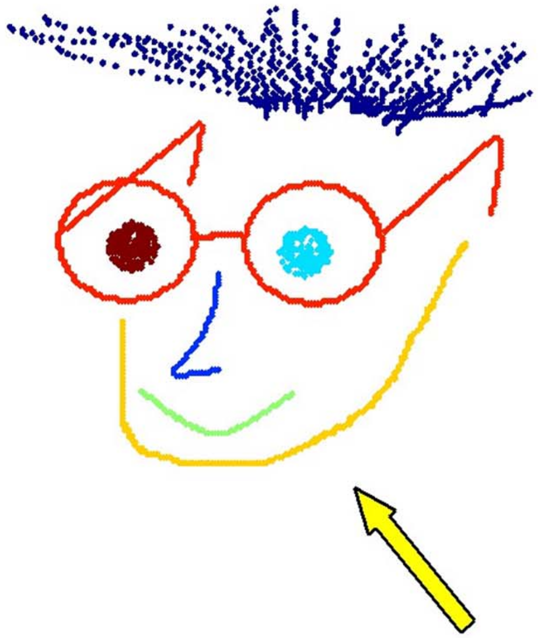
Spectral Clustering

CS534

Good clustering – we know it when we see it

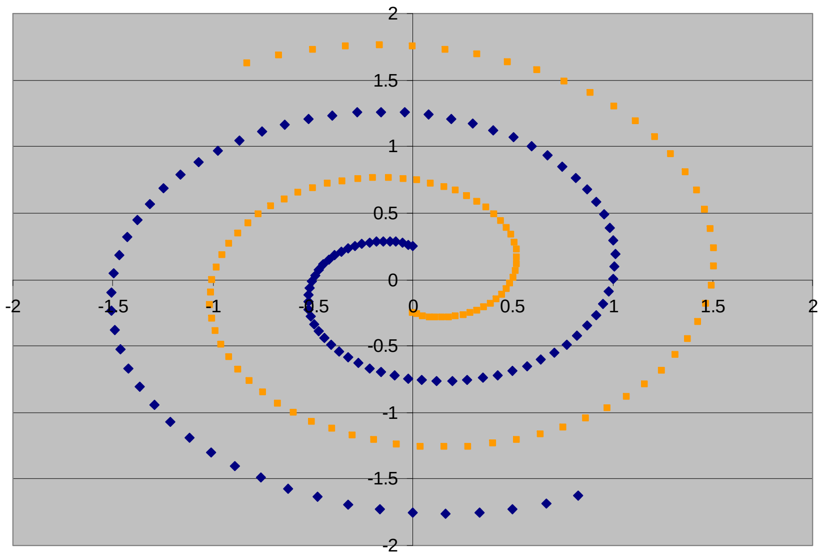


A mixture (of Gaussians) is a good model for this data set



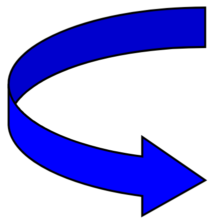
But clusters are not always about Euclidean distance and parametric models are not always appropriate

Spectral Clustering Example – 2 Spirals

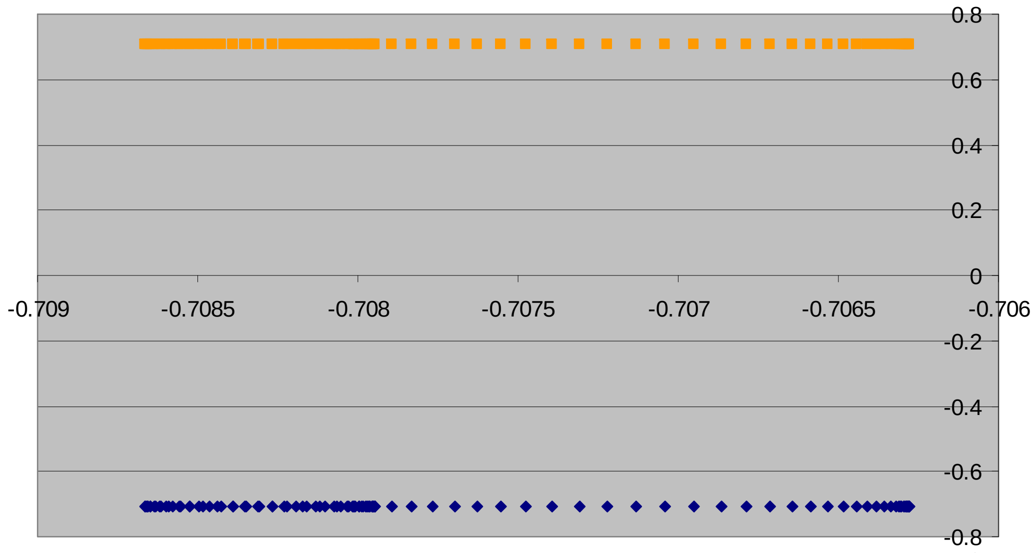


Dataset exhibits complex cluster shapes

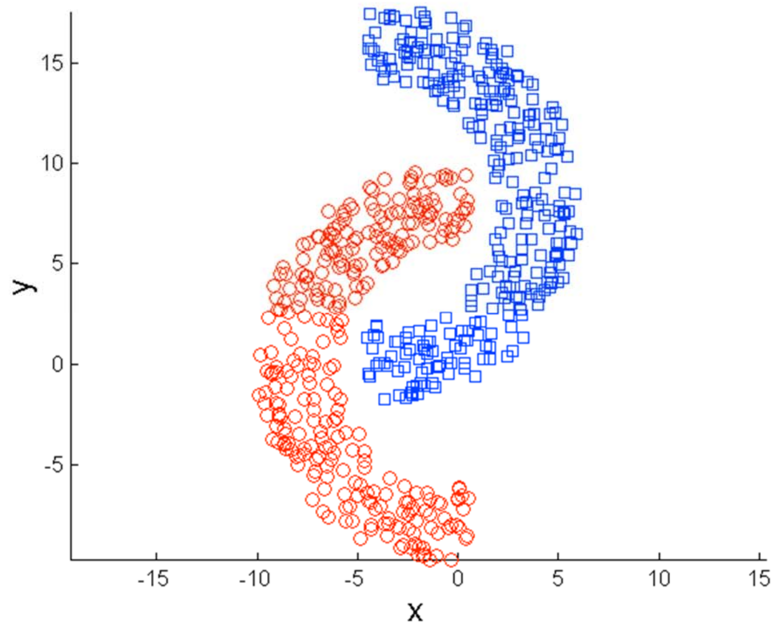
⇒ K-means performs very poorly in this space due bias toward dense spherical clusters.

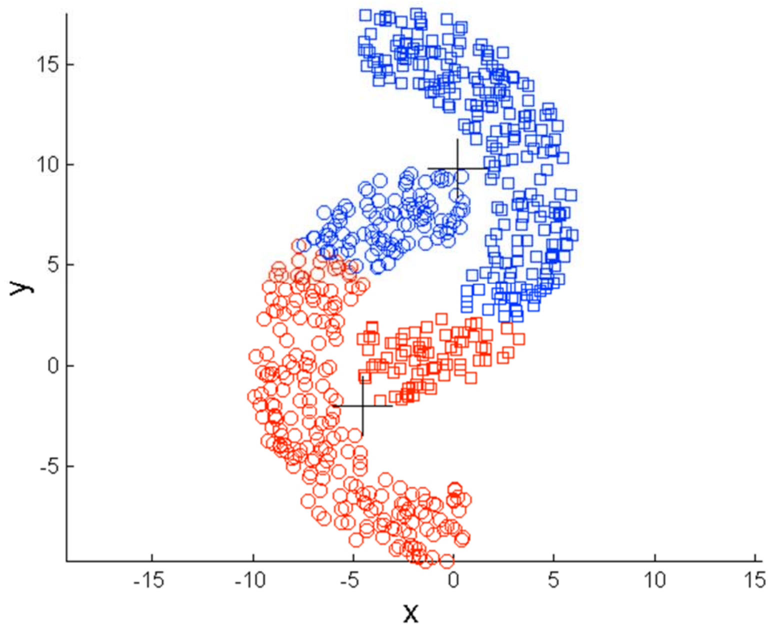


In the embedded space given by two leading eigenvectors, clusters are trivial to separate.



Spectral Clustering Example





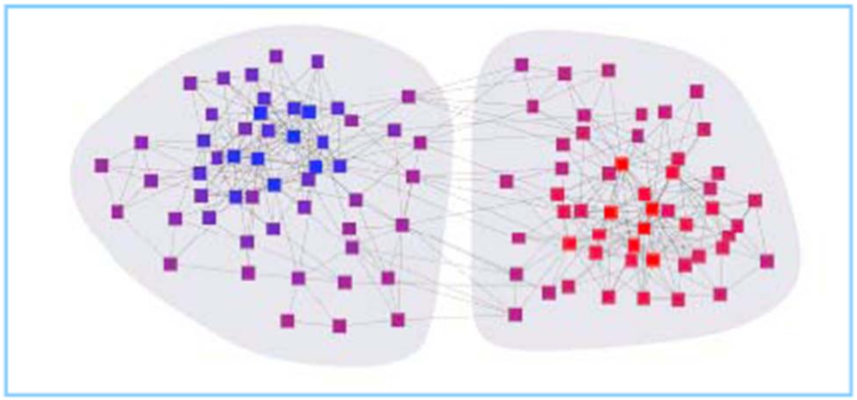
Original Points

K-means (2 Clusters)

Why k-means fail for these two examples?

Spectral Clustering

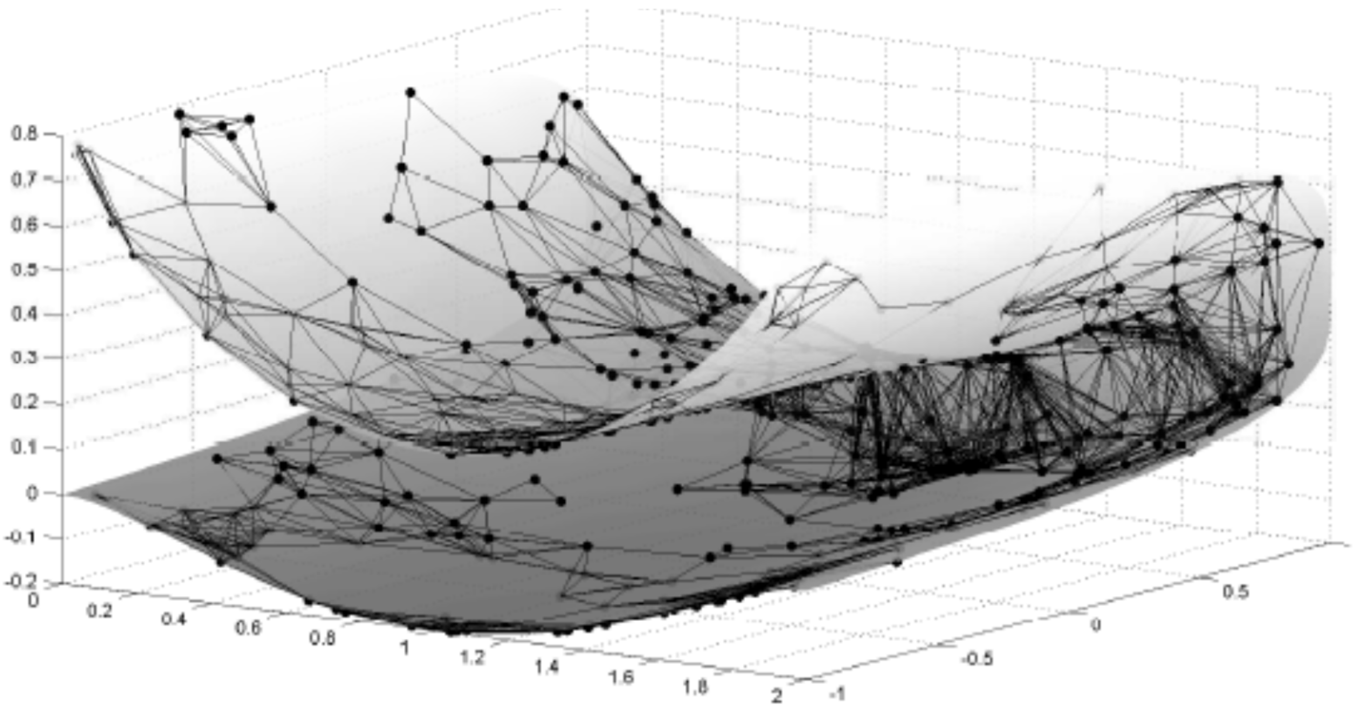
- Represent data points as the vertices V of a graph G.
- Vertices are connected by edges E
- Edges have weights *W*
  - Large weights mean that the adjacent vertices are very similar; small weights imply dissimilarity



Methods that use the spectrum of the similarity matrix *W* to cluster are known as ***spectral clustering***

Graph based representation

- Rely on local similarity information to costruct graph
- Avoid the misleading global distance

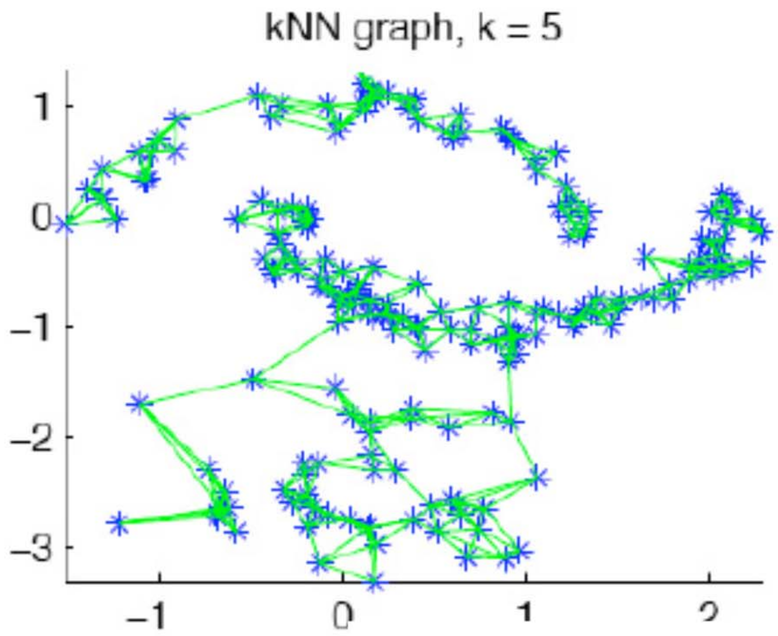


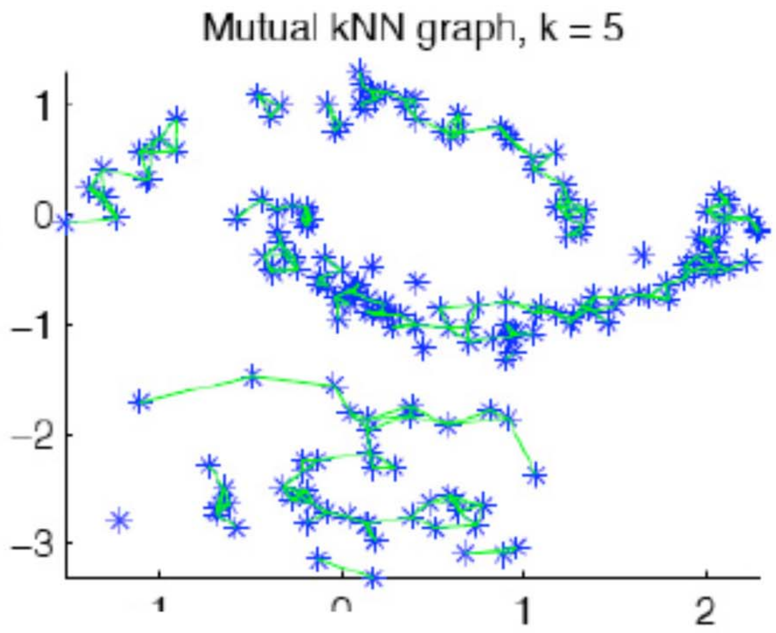
How to Create the Graph ?

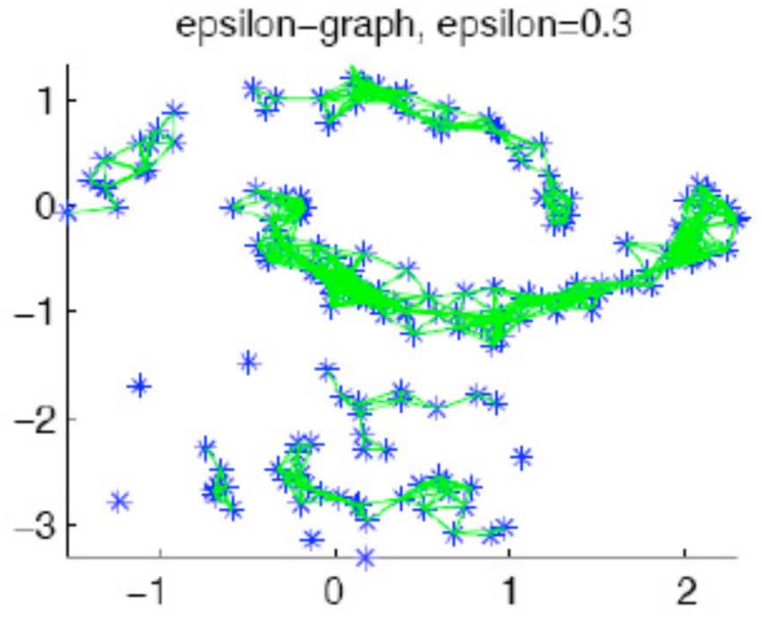
- It is common to use a Gaussian Kernel to compute similarity between objects

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

- One could create
  - A fully connected graph
  - K-nearest neighbor graph (each node is only connected to its K-nearest neighbors)
  - ϵ-neighborhood graph


kNN graph, k = 5
1
0
−1
−2
−3
−1
0
1
2

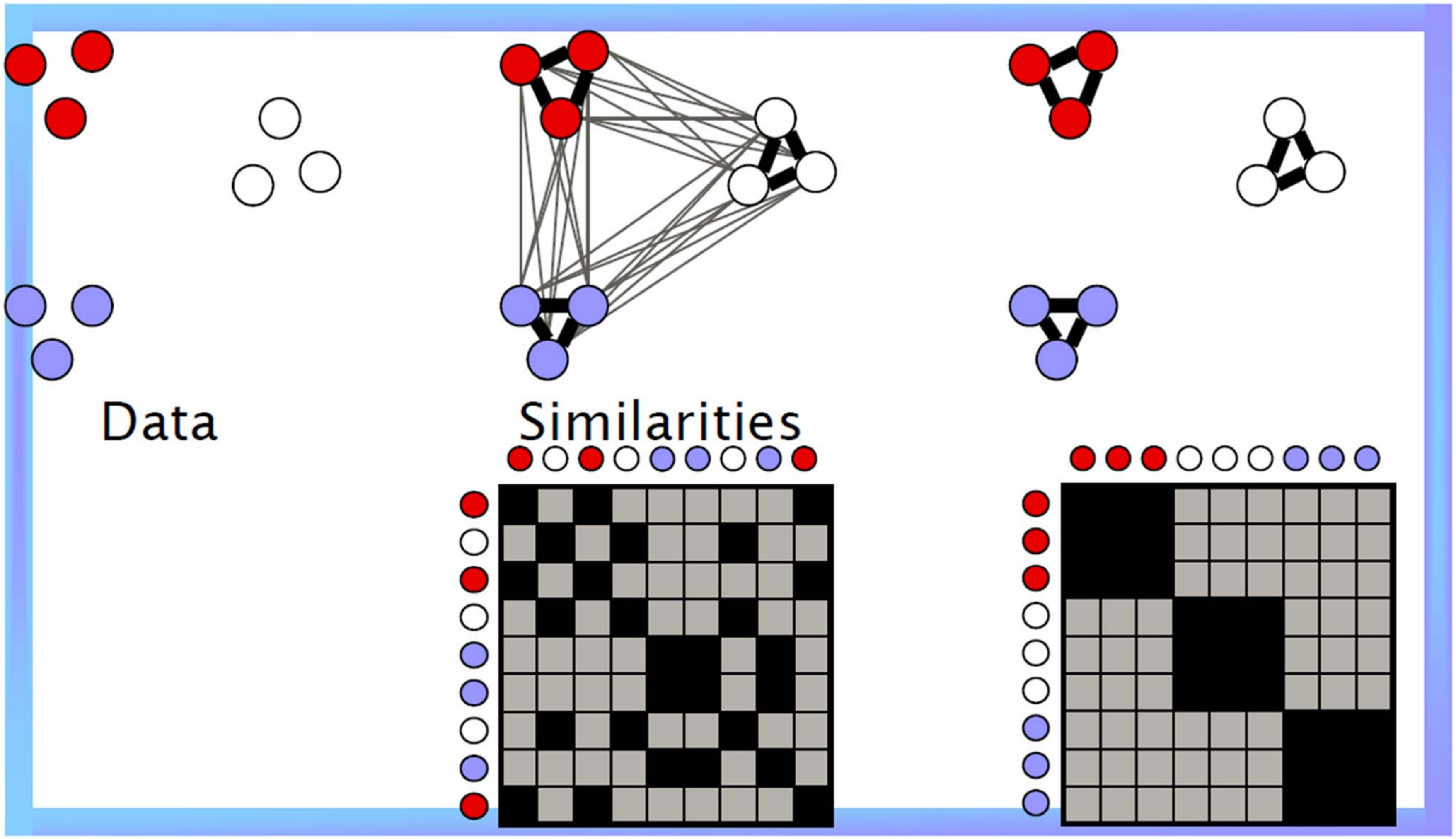

Mutual kNN graph, k = 5
1
0
−1
−2
−3
−1
0
1
2


epsilon−graph, epsilon=0.3
1
0
−1
−2
−3
−1
0
1
2

Motivations/Objectives

- There are different ways to interpret the spectral clustering
- One can view spectral clustering as finding partitions of the graph that minimizes Normalized Cut
- Alternatively, we can also view this as performing a random walk on the graph

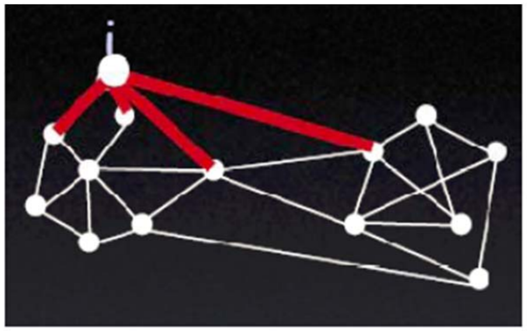
Graph partitioning

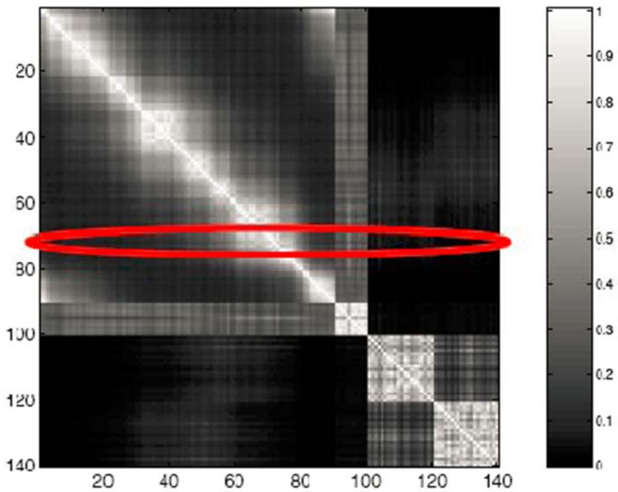


Graph Terminologies

- Degree of nodes

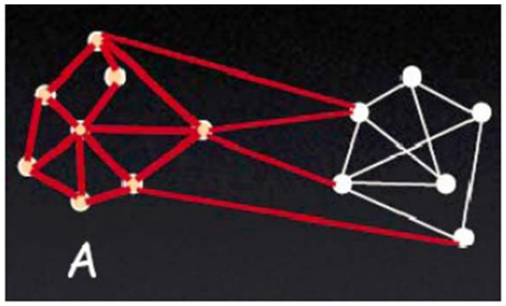
$$d_i = \sum_j w_{i,j}$$

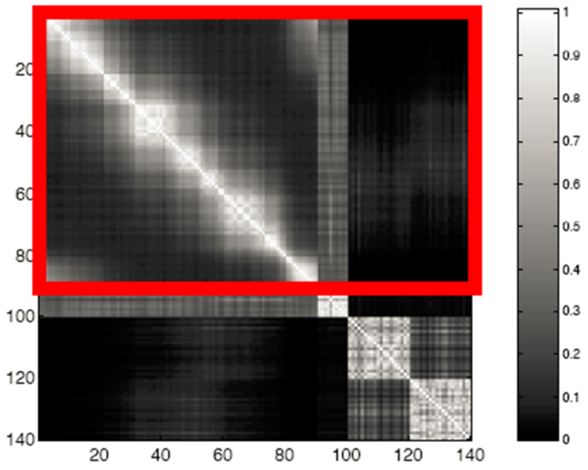




- Volume of a set

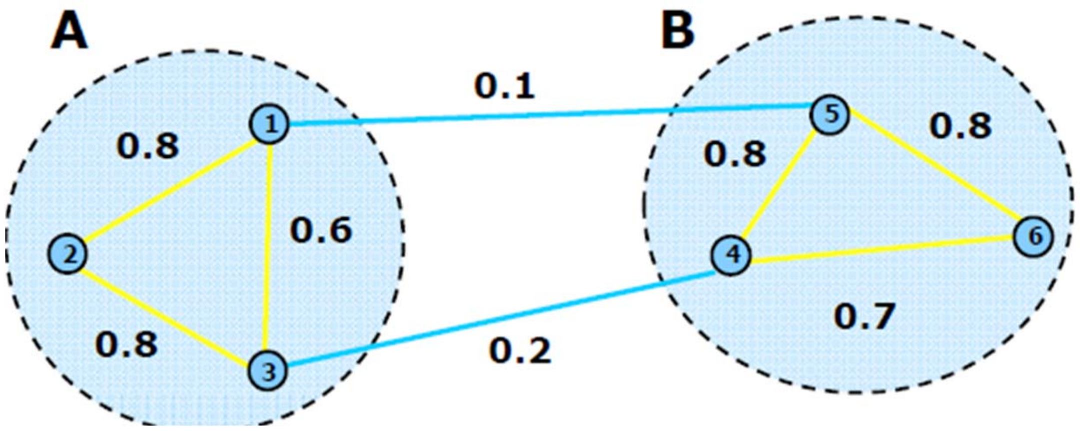
$$vol(A) = \sum_{i \in A} d_i, A \subseteq V$$





Graph Cut

- Consider a partition of the graph into two parts A and B



- ***Cut(A, B)***: sum of the weights of the set of edges that connect the two groups

$$cut(A,B) = \sum_{i \in A, j \in B} w_{ij} = \boldsymbol{0.3}$$

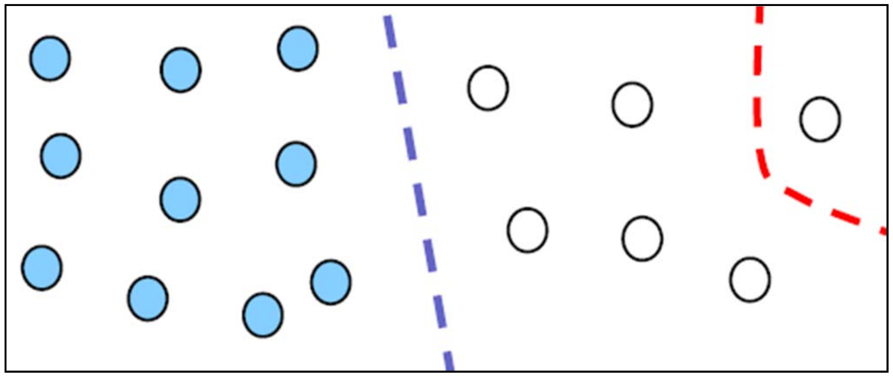
- An intuitive goal is find the partition that minimizes the cut

Min Cut Objective

- **Mincut:** Minimize weight of connections between groups

$$\min_{A \cap B = \emptyset, A \cup B = V} Cut(A, B)$$

- Problem:
  - Prefer degenerate solution (e.g. the red partition)

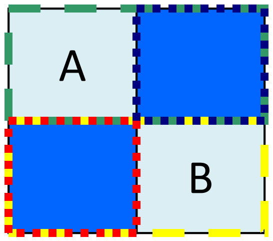


  - Need to express preference for more balanced solution

Normalized Cut

- Consider the connectivity between groups relative to the volume of each group

$$Ncut(A,B) = \frac{cut(A,B)}{Vol(A)} + \frac{cut(A,B)}{Vol(B)}$$



$$Ncut(A,B) = cut(A,B)\frac{Vol(A)+Vol(B)}{Vol(A)Vol(B)}$$

Minimized when Vol(A) and Vol(B) are equal.
Thus encourage balanced cut

Solving NCut

- How to minimize $Ncut$?

  Let W be the similarity matrix, $W(i, j) = W_{i,j}$;
  Let D be the diag. matrix, $D(i,i) = \sum_j W(i, j)$;
  Let x be a vector in $\{1,-1\}^N$, $x(i) = 1 \Leftrightarrow i \in A$.

- With some simplifications, we can show:

$$\min_x Ncut(x) = \min_y \frac{y^T (D - W) y}{y^T D y}$$

Rayleigh quotient

Subject to: $y^T D1 = 0$ ($\mathbf{y}$ *takes discrete values*)

NP-Hard!

Solving NCut

- Relax the optimization problem into the continuous domain by solving generalized eigenvalue system:

$$\min_{y} y^T(D-W)y \text{ subject to } y^T D y = 1$$

- Which gives:$(D-W)y = \lambda D y$
- Note that $(D-W)1 = 0$, so the first eigenvector is $y_0 = 1$ with eigenvalue 0.
- The second smallest eigenvector is the real valued solution to this problem!!

2-way Normalized Cuts

1. Compute the affinity matrix W, compute the degree matrix (D), D is diagonal and $D(i,i) = \sum_{j \in V} W(i,j)$
2. Solve $(D - W)y = \lambda D y$, where $D - W$ is called the Laplacian matrix
3. Use the eigenvector with the second smallest eigen-value to bipartition the graph into two parts.

Creating Bi-partition Using 2nd Eigenvector

- Sometimes there is not a clear threshold to split based on the second vector since it takes continuous values
- How to choose the splitting point?

a) Pick a constant value (0, or 0.5).

b) Pick the median value as splitting point.

c) Look for the splitting point that has the minimum *Ncut* value:

1. Choose *n* possible splitting points.
2. Compute *Ncut* value.
3. Pick minimum.

K-way Partition?

- Recursive bi-partitioning (Hagen et al.,'91)
  - Recursively apply bi-partitioning algorithm in a hierarchical divisive manner.
  - Disadvantages: Inefficient, unstable
- Cluster multiple eigenvectors
  - Build a reduced space from multiple eigenvectors.
  - Commonly used in recent papers
  - A preferable approach… its like doing dimension reduction then k-means

Spectral clustering

(Ng, Jordan, and Weiss 2001)

- Form the affinity matrix W
  - $W(i,j) = \exp(-\frac{\left|x_i - x_j\right|^2}{2\sigma})\,, W(i,i) = 0$
- Compute the degree matrix $D = diag(W\mathbf{1})$
- Compute the normalized graph Laplacian

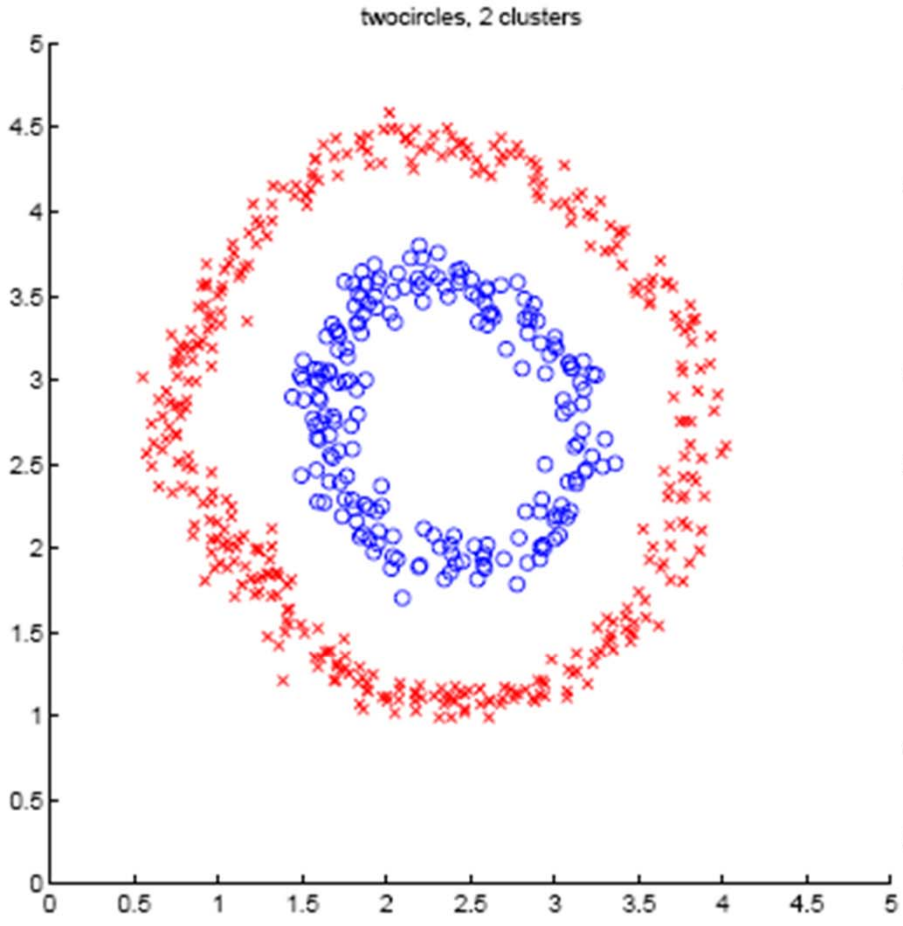
$$L = D^{-\frac{1}{2}} W D \text{^} (-1/2)$$

- Find the k largest eigenvectors, for new data matrix $X'_{n\times k}$
- Normalize the rows to have unit length
- Treating each row as a data point in k-d space and cluster the data into k clusters via kmeans

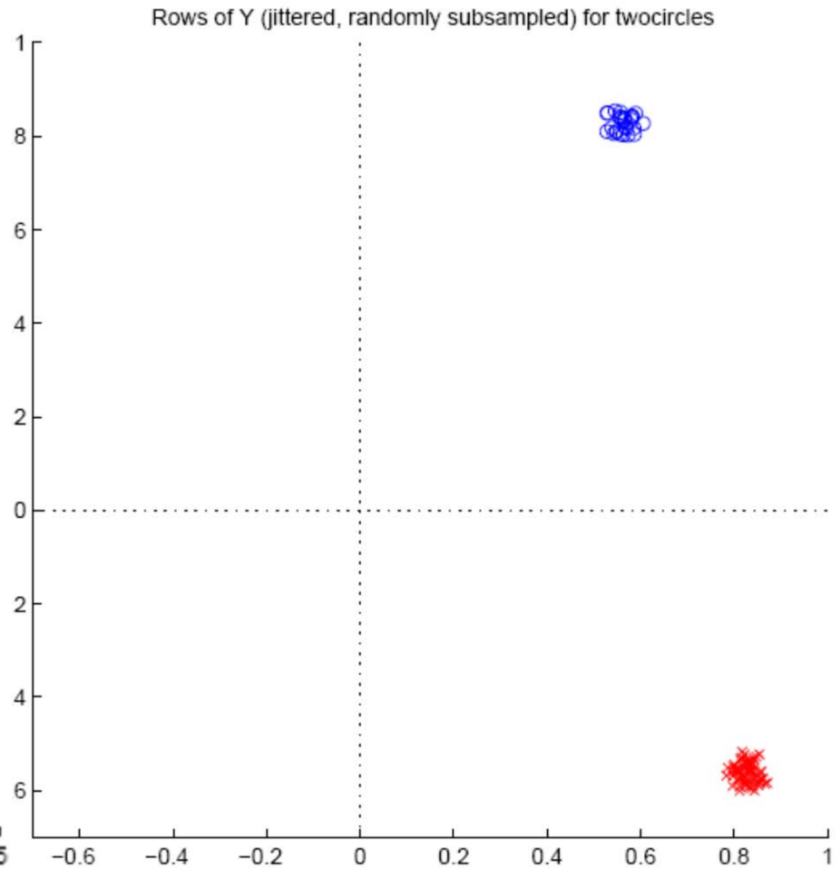
Why?

- If we eventually use K-means, why not just apply K-means to the original data?

- This method allows us to cluster non-convex regions


twocircles, 2 clusters

(e)


Rows of Y (jittered, randomly subsampled) for twocircles

A Random Walk View of Spectral Clustering

- Imagine a random walk from node i on the graph
- Assume that the probability of taking step from node i to node j is given by the transition matrix P: $p_{ij} = \frac{W_{ij}}{D(i,i)}$
- Starting within one cluster and take a random walk governed by P, we will be likely remain in the same cluster for a long time

Property of Random Walk

- If we start at i_0, where will we end up after t steps?

$$i_1 \sim P_{i_0 i_1},$$

$$i_2 \sim \sum_{i_1} P_{i_0 i_1} P_{i_1 i_2} = \left(\mathbf{P}^2\right)_{i_0 i_2}$$

$$i_3 \sim \sum_{i_2} \left(\mathbf{P}^2\right)_{i_0 i_2} P_{i_2 i_3} = \left(\mathbf{P}^3\right)_{i_0 i_3},$$

$$\cdots$$

$$i_t \sim \left(\mathbf{P}^t\right)_{i_0 i_t}$$

Transition Matrix Decomposition

- Recall that $P_{ij} = \frac{W_{ij}}{D(i,i)}$, thus we have: $P = D^{-1}W$
- We will focus on a symmetric variant of this matrix $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ for now
- We can decompose it using its eigen-vectors $z_1, \cdots, z_N$, with $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_N|$

$$D^{-\frac{1}{2}}WD^{-\frac{1}{2}} = \lambda_1 z_1 z_1^T + \cdots + \lambda_N z_N z_N^T$$

- Spectral graph theory states that under mild conditions, we have
  - $\lambda_1 = 1$, and the rest of eigen values are less than 1

Random Walk of Infinite Steps

$$(\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}})^t = (\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}})\cdots(\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}) = \mathbf{D}^{\frac{1}{2}}\mathbf{P}^t\mathbf{D}^{-\frac{1}{2}}$$

- Thus

$$\begin{aligned}\mathbf{P}^t &= \mathbf{D}^{-\frac{1}{2}}\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}\right)^t\mathbf{D}^{\frac{1}{2}} \\ &= \mathbf{D}^{-\frac{1}{2}}\left(\lambda_1\mathbf{z}_1\mathbf{z}_1^T + \ldots + \lambda_N\mathbf{z}_N\mathbf{z}_N^T\right)^t\mathbf{D}^{\frac{1}{2}} \\ &= \mathbf{D}^{-\frac{1}{2}}\left(\lambda_1^t\mathbf{z}_1\mathbf{z}_1^T + \ldots + \lambda_N^t\mathbf{z}_N\mathbf{z}_N^T\right)\mathbf{D}^{\frac{1}{2}}\end{aligned}$$

- Since $\lambda_1 = 1, and\ |\lambda_i| < 1$, when $t \rightarrow \infty$, we have:

$$\mathbf{P}^\infty = \mathbf{D}^{-\frac{1}{2}}\left(\mathbf{z}_1\mathbf{z}_1^T\right)\mathbf{D}^{\frac{1}{2}}$$

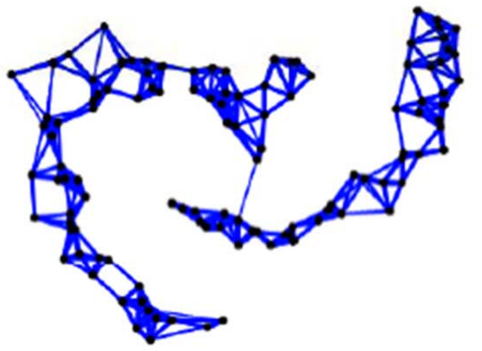
Given infinite time steps, the probability of ending in a particular node is independent of the starting node

Finite Step Random Walk

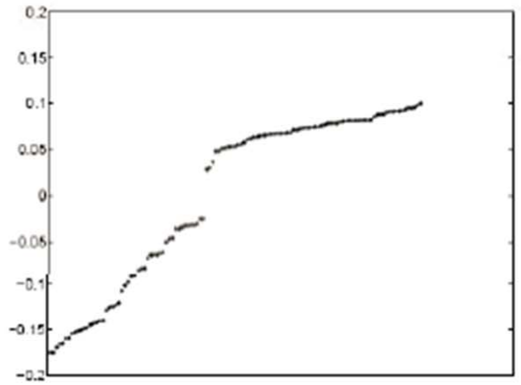
$$\mathbf{P}^t \approx \mathbf{P}^\infty + \mathbf{D}^{-\frac{1}{2}} \left(\lambda_2^2 \mathbf{z}_2 \mathbf{z}_2^T\right) \mathbf{D}^{\frac{1}{2}}$$

- Given large but finite t, we can focus on the second largest eigen-vector
- $(z_2 z_2^T)_{ij} = z_{2i} z_{2j}$, so the probability starting at x_i, and end up at x_j is increased if z_{2i} and z_{2j} have the same sign
- This suggests that we should cluster based on the sign of z_{2i}

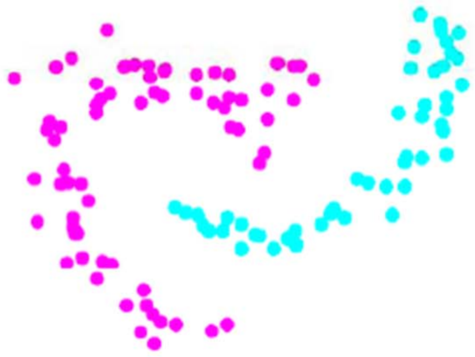
Example



5–NN graph



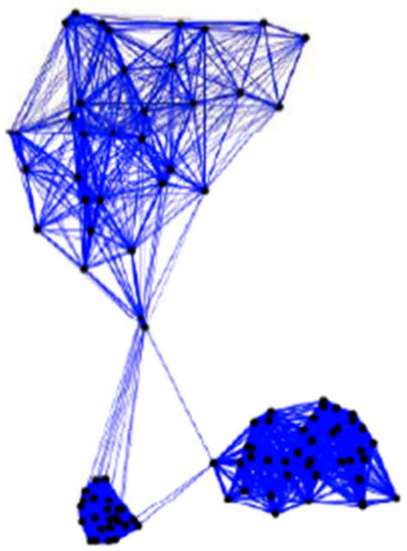
2nd eigen-vector



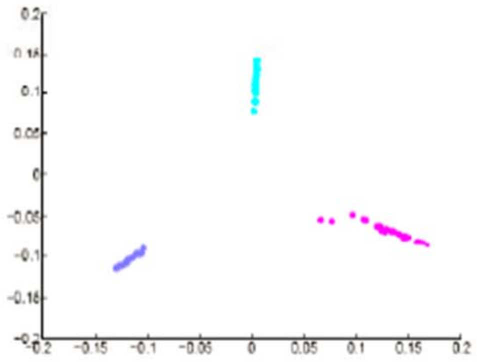
Clustering result

Beyond bi-partition

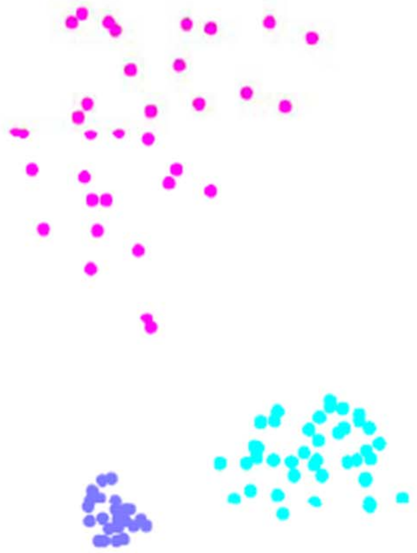
Graph, 20-NN



$\mathbf{Z}$



Clustering



More examples, from [Ng *et al* '01]