Spectral Clustering

Li, Jia

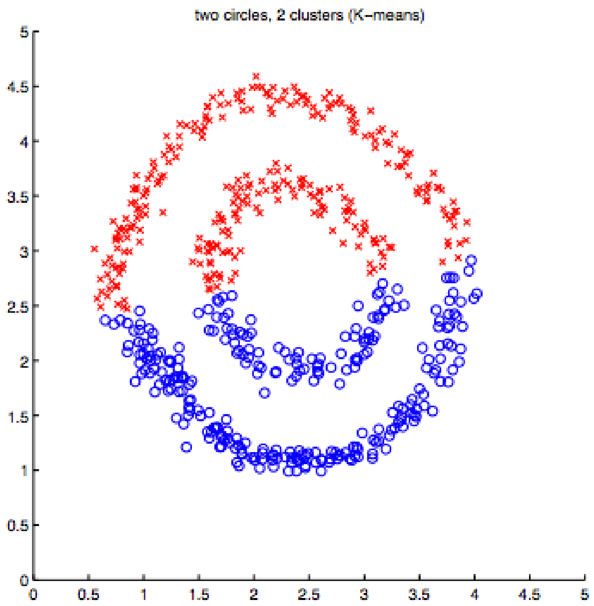
DSAA 5002
HKUST Guangzhou

2025 Fall

Oct 20

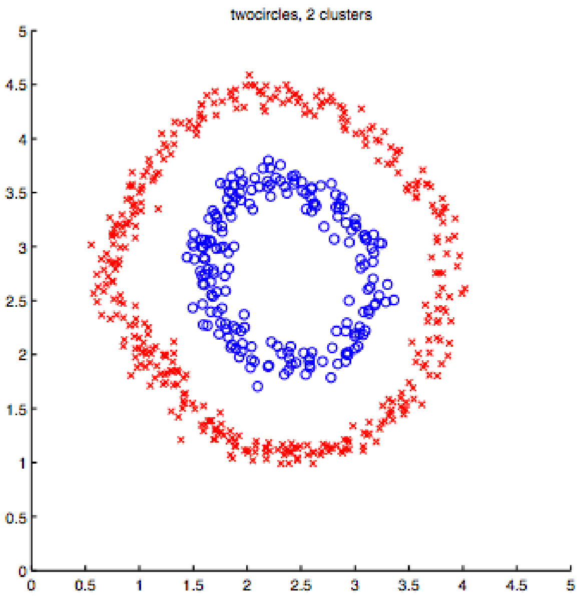
Clustering a 2-D ring

K-means



(i)

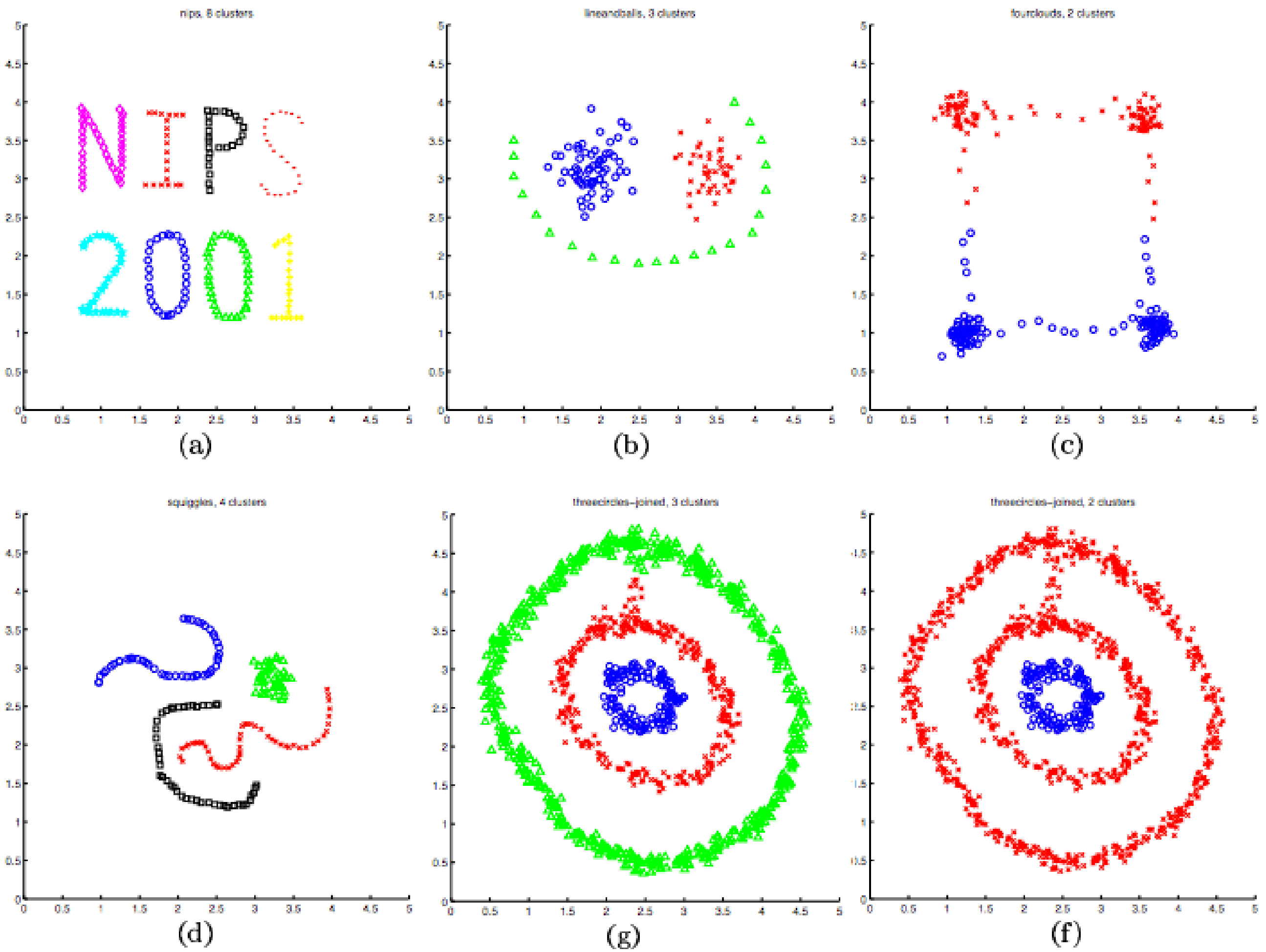
Spectral clustering



(e)

[Shi & Malik ‘00; Ng, Jordan, Weiss NIPS ‘01]

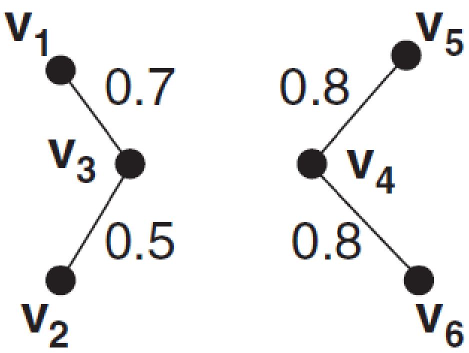
Spectral Clustering



[Figures from Ng, Jordan, Weiss NIPS ‘01]

Spectral Clustering

- Spectral clustering is a graph-based clustering approach
  - Does a graph partitioning of the proximity graph of a data set
  - Breaks the graph into components, such that
    - The nodes in a component are strongly connected to other nodes in the component
    - The nodes in a component are weakly connected to nodes in other components
    - See simple example below (W is the proximity matrix)



$$W = \begin{bmatrix} 0 & 0 & 0.7 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0.7 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.8 & 0.8 \\ 0 & 0 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & 0 & 0 \end{bmatrix}$$

Spectral Clustering

- For the simple graph below, the proximity matrix can be written as

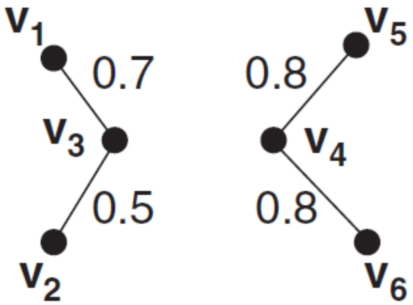
$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 \end{pmatrix}$$

  - Because the graph consists of two connected components finding clusters is easy.
  - More generally, we need an automated approach
    - Must be able to handle graphs where the components are not completely separate
    - Spectral graph partitioning provides such an approach
    - Based on eigenvalue decomposition of a slight modification of the proximity matrix.

Spectral Clustering

- Uses an eigenvalue based approach to do the graph portioning
  - Based on the Laplacian matrix (L) of a graph, which is derived from the proximity matrix (W)
    - W is also known as the weighted adjacency matrix

- Define a diagonal matrix D
  - $D_{ij} = \begin{cases} \sum_k W_{ij}, & if\ i = j \\ 0, & otherwise \end{cases}$
- k^{th} diagonal entry of D is the sum of the edges of the k^{th} node of W
  - See example below



$$W = \begin{bmatrix} 0 & 0 & 0.7 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0.7 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.8 & 0.8 \\ 0 & 0 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & 0 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 0.7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8 \end{bmatrix}$$

Spectral Clustering Algorithm

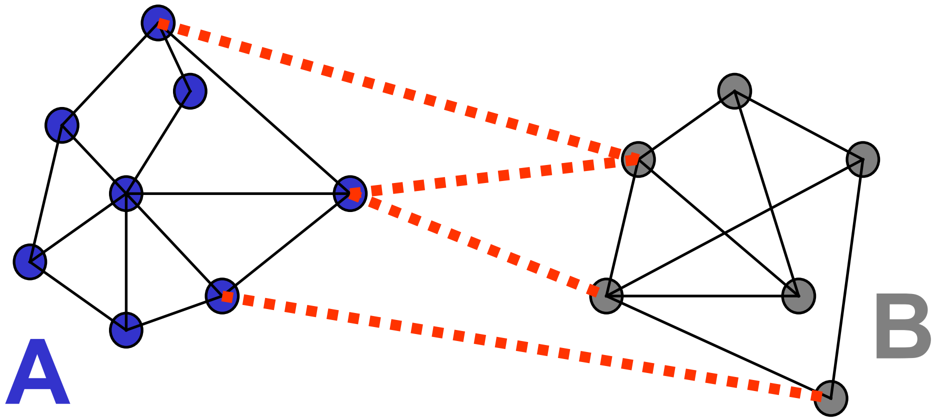
- Given the Laplacian of a graph, it is easy to define a spectral graph clustering algorithm
- We simply apply k-means to the matrix consisting of the first k eignenvectors of L
- Note that we cluster the rows of that matrix

Algorithm 8.10 Spectral clustering algorithm.

1: Create a sparsified similarity graph $\mathcal{G}$.
2: Compute the graph Laplacian for $\mathcal{G}$, $\mathbf{L}$
3: Create a matrix $\mathbf{V}$ from the first k eigenvectors of $\mathbf{L}$.
4: Apply K-means clustering on $\mathbf{V}$ to obtain the k clusters.

Spectral Clustering

- Group points based on the links in a graph



- How do we create the graph?
  - Weights on the edges based on similarity between the points
  - A common choice is the Gaussian kernel

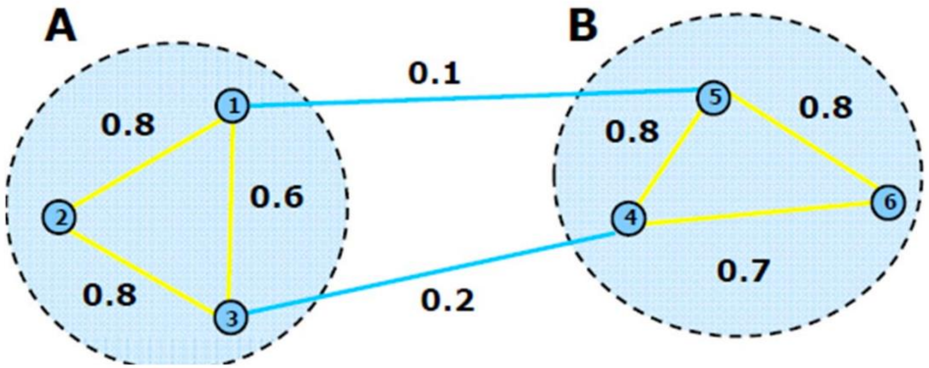
$$W(i,j) = \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right)$$

- One could create
  - A fully connected graph
  - k-nearest graph (each node is connected only to its k-nearest neighbors)

slide credit: Alan Fern

Graph Cut

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



- **Cut(A, B) is the weight of all edges that connect the two groups**

$$\mathrm{Cut}(A,B) = \sum_{i\in A, j\in B} W(i,j) = 0.3$$

- **An intuitive goal is to find a partition that minimizes the cut**
  - min-cuts in graphs can be computed in polynomial time
  - Recap the Max-flow min-cut theorem

Problem with Min Cut

- The weight of a cut is proportional to number of edges in the cut; tends to produce small, isolated components.

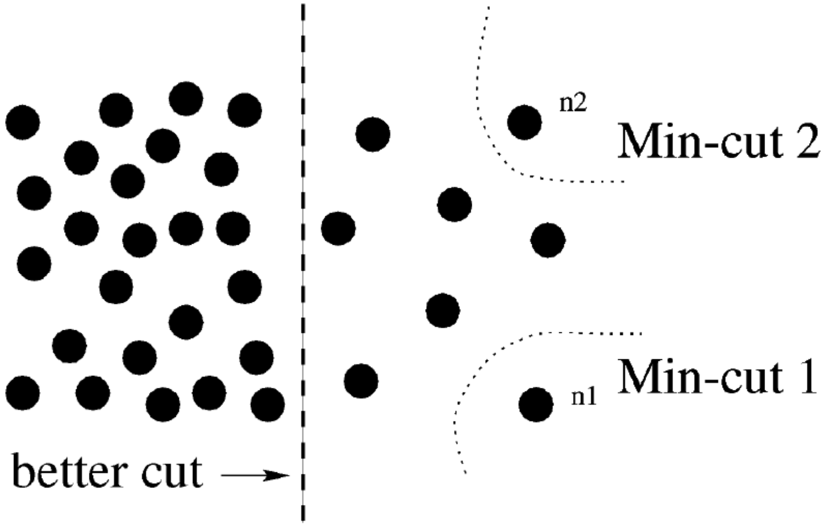


Fig. 1. A case where minimum cut gives a bad partition.

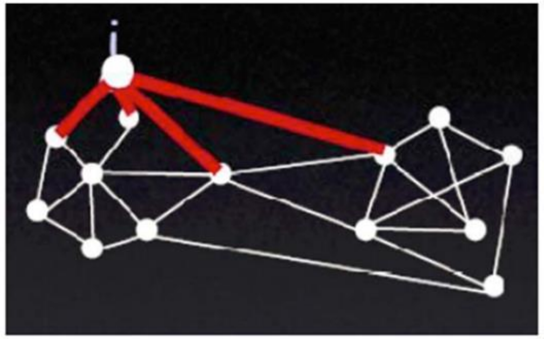
We would like a balanced cut

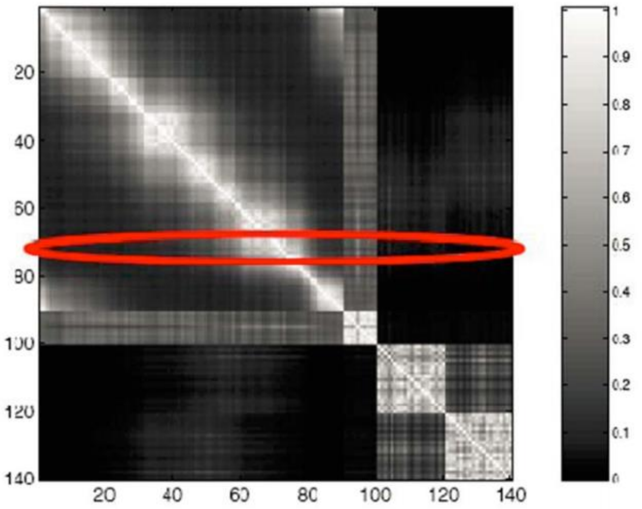
[Shi & Malik, 2000 PAMI]

Graphs as Matrices

- Let *W(i, j)* denote the matrix of the edge weights
- The degree of node in the graph is:

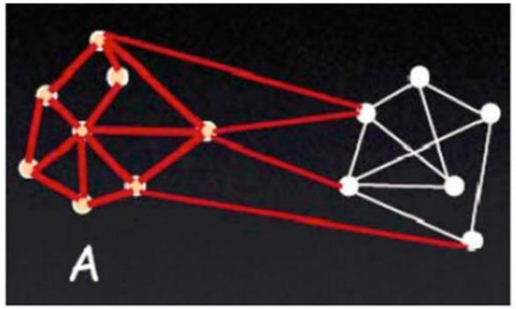
$$d(i) = \sum_j W(i,j)$$

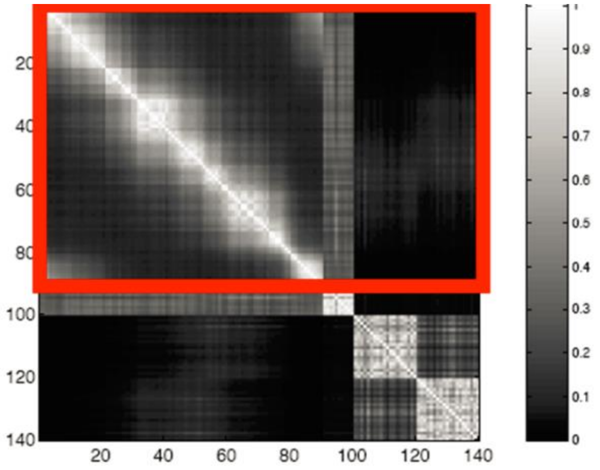




- The volume of a set A is defined as:

$$\mathrm{Vol}(A) = \sum_{i \in A} d(i)$$





Normalized Cut

- Intuition: consider the connectivity between the groups relative to the volume of each group:

$$\mathrm{NCut}(A,B) = \frac{\mathrm{Cut}(A,B)}{\mathrm{Vol}(A)} + \frac{\mathrm{Cut}(A,B)}{\mathrm{Vol}(B)}$$

$$\mathrm{NCut}(A,B) = \mathrm{Cut}(A,B)\left(\frac{\mathrm{Vol}(A)+\mathrm{Vol}(B)}{\mathrm{Vol}(A)\mathrm{Vol}(B)}\right)$$

  - minimized when Vol(A) = Vol(B)
  - encouraging a balanced cut

- Unfortunately minimizing normalized cut is NP-Hard even for planar graphs [Shi & Malik, 00]

Solving Normalized Cut

- We will formulate an optimization problem
  - Let W be the similarity matrix
  - Let D be a diagonal matrix with D(i,i) = d(i) — the degree of node i
  - Let **x** be a vector $\{1, -1\}^N$, x(i) = 1 ↔ i ∈ A
  - The matrix *(D-W)* is called the Laplacian of the graph
- With some simplification we can show that the problem of minimizing normalized cuts can be written as:

$$\min_{\mathbf{x}} \mathrm{NCut}(\mathbf{x}) = \min_{\mathbf{y}} \frac{\mathbf{y}^T(D-W)\mathbf{y}}{\mathbf{y}^T D\mathbf{y}}$$

$$\text{subject to: } \mathbf{y}^T D\mathbf{1} = 0$$

$$\mathbf{y}(i) \in \{1, -b\}$$

Solving Normalized Cut

- Normalized cut objective:

$$\min_{\mathbf{x}} \mathrm{NCut}(\mathbf{x}) = \min_{\mathbf{y}} \frac{\mathbf{y}^T(D-W)\mathbf{y}}{\mathbf{y}^T D\mathbf{y}}$$

$$\text{subject to: } \mathbf{y}^T D\mathbf{1} = 0$$

$$\mathbf{y}(i) \in \{1, -b\}$$

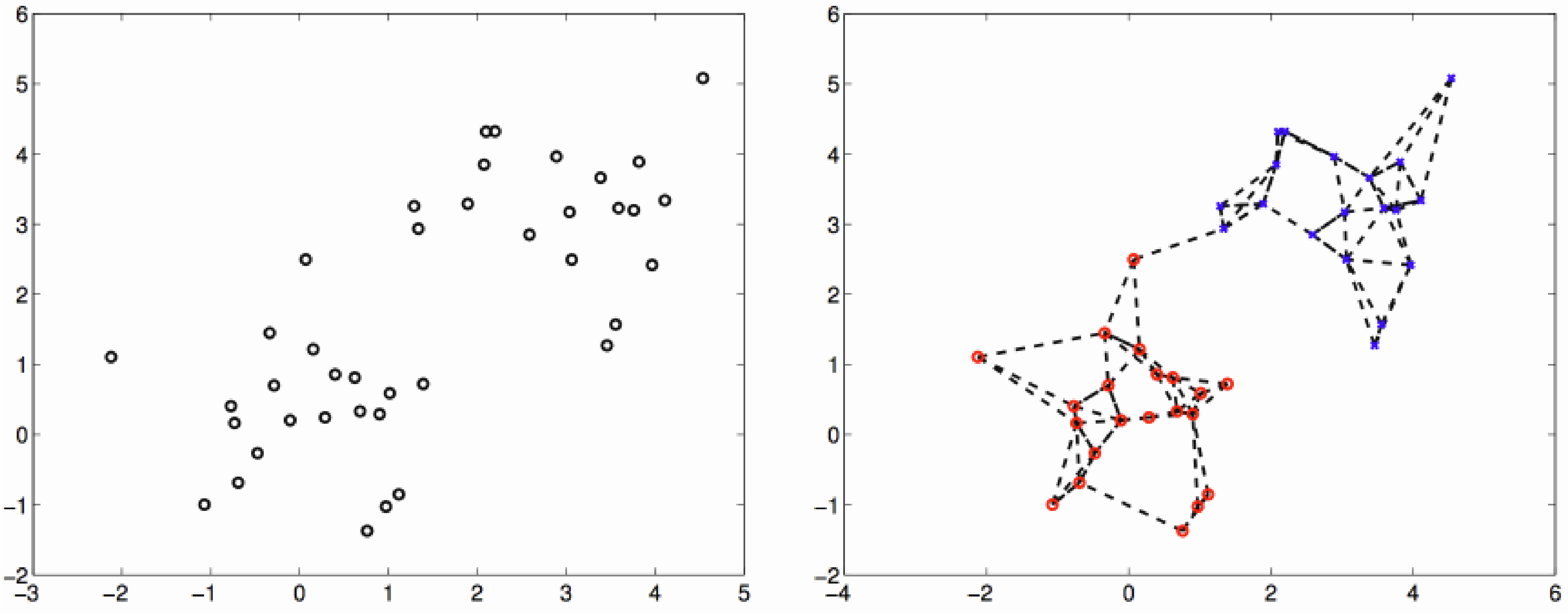
- Relax the integer constraint on **y**:

$$\min_{\mathbf{y}} \mathbf{y}^T(D-W)\mathbf{y}; \text{ subject to: } \mathbf{y}^T D\mathbf{y} = 1, \mathbf{y}^T D\mathbf{1} = 0$$

- Same as: $(D-W)\mathbf{y} = \lambda D\mathbf{y}$ (Generalized eigenvalue problem)
- Note that $(D-W)\mathbf{1} = 0$, so the first eigenvector is $\mathbf{y}_1 = \mathbf{1}$, with the corresponding eigenvalue of 0
- The eigenvector corresponding to the second smallest eigenvalue is the solution to the relaxed problem

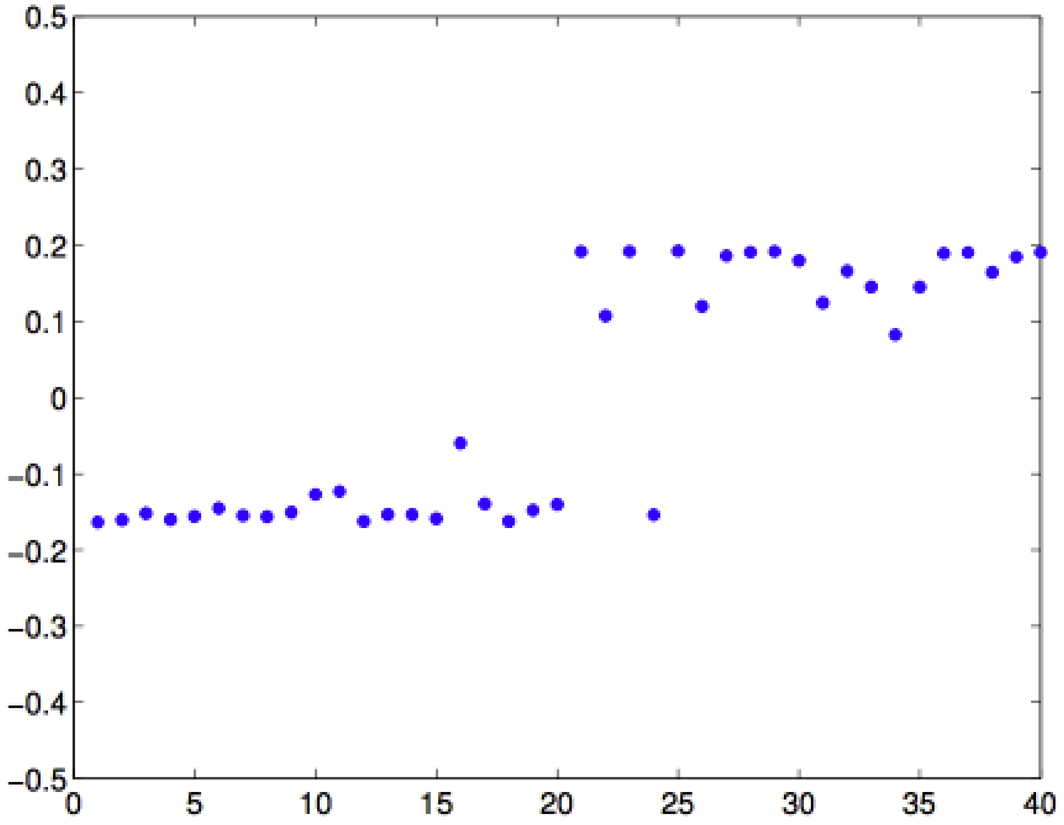
Spectral Clustering Example

- **Data:** Gaussian weighted edges connected to 3 nearest neighbors



Spectral Clustering Example

- Components of the eigenvector corresponding to the second smallest eigenvalue



Strengths and Limitations

- Can detect clusters of different shape and sizes

- Sensitive to how graph is created
- Sensitive to outliers
- Time complexity depends on the sparsity of the data matrix
  - Improved by sparsification

How to create a graph?

- Priors, e.g., friendship in a social network, spatial closeness, semantic similarities
- Embedded in data itself, e.g., kernels

Slides Credit

[1] Tan et al. Introduction to Data Mining.
[2] Subhransu Maji. Clustering in CMPSCI689