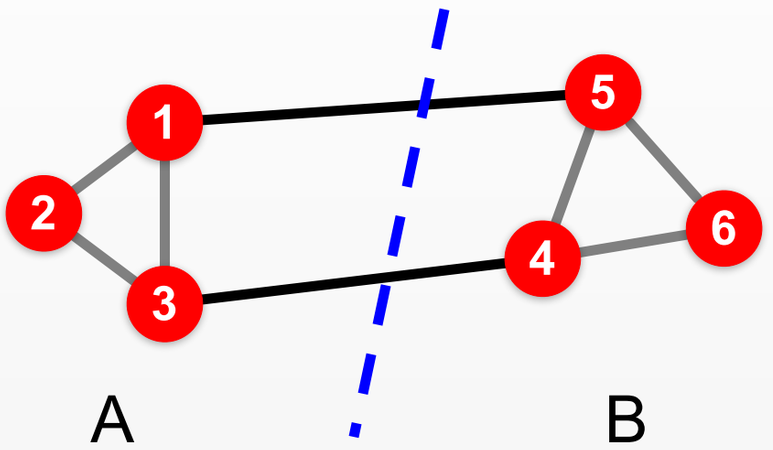
Spectral Clustering

Minimizing normalized cuts

Graph Partitioning

(a.k.a. Community Detection)



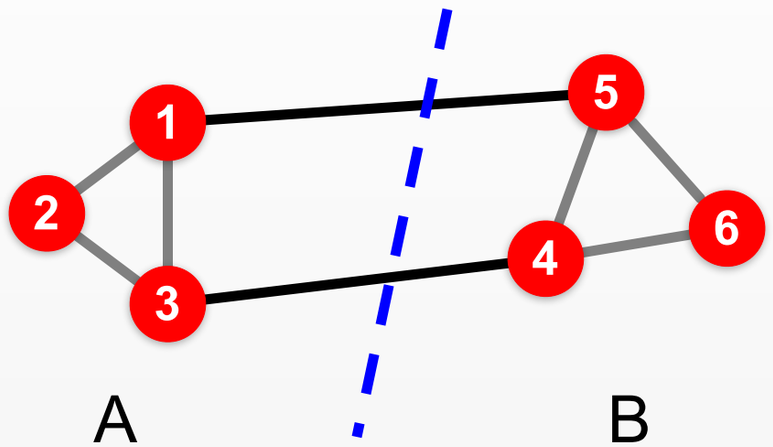
- What makes a good partition?
  - Maximize the within-group connections
  - Minimize the between-group connections

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Graph Cuts

Definition of a graph:
$\mathcal{G} = (V, E)$

- $V = \{1,2,\ldots,N\}$, the set of vertices or nodes
- $E \in V \times V$, the set of edges.



Number of edges across a partition boundary:

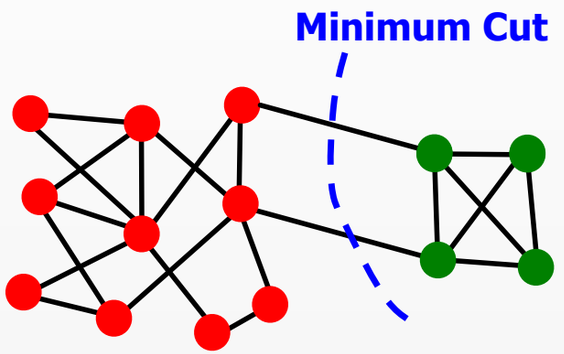
$$\text{cut}(A,B) = \sum_{i \in A, j \in B} I[(i,j) \in E]$$

$$= \frac{1}{4} \sum_{(i,j) \in E} (y_i - y_j)^2 \qquad y_i = \begin{cases} 1 & \text{if } i \in A \\ -1 & \text{if } i \in B \end{cases}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Minimum Cuts

Goal: Find the partition that minimizes the Graph Cut



$$\min_{y \in \{-1,1\}^n} \sum_{(i,j) \in E} (y_i - y_j)^2 \qquad y_i = \begin{cases} 1 & \text{if } i \in A \\ -1 & \text{if } i \in B \end{cases}$$

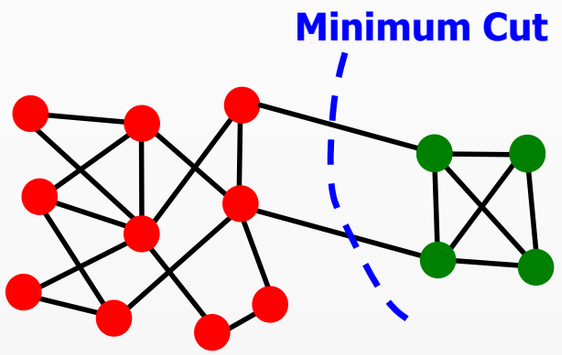
Problem 1: Can't enumerate all choices $y_1, \ldots, y_n$

(same problem as in clustering)

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Minimum Cuts

Goal: Find the partition that minimizes the Graph Cut



$$\min_{y \in \{-1,1\}^n} \sum_{(i,j) \in E} (y_i - y_j)^2$$

$$y = [y_1, y_2, \ldots, y_n]$$

$$y_i = \begin{cases} 1 & \text{if } i \in A \\ -1 & \text{if } i \in B \end{cases}$$

Problem 2: Trivial solution

All vertices in the same community

(same problem as in clustering)

All vertices in the same community

$y_1 = 1, y_2 = 1, \ldots, y_n = 1$

lead to 0 minimum cut

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

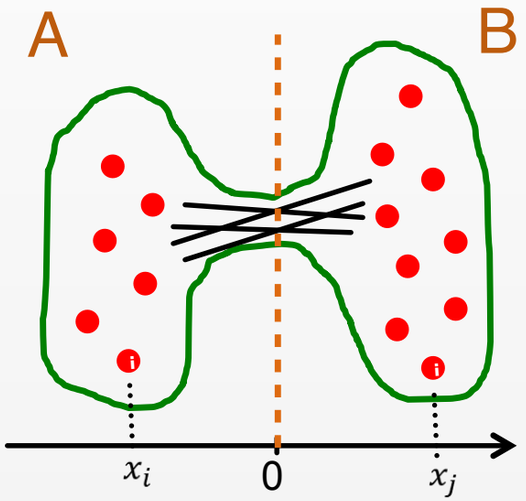
Second Eigenvector (Fiedler Vector)

Minimum Cut

$$\min_{\boldsymbol{y}\in\{-1,1\}^n} \sum_{(i,j)\in E} (\boldsymbol{y}_i - \boldsymbol{y}_j)^2$$

Convert to a continuous optimization problem

$$\min_{x\in\mathbf{R}^n} \sum_{(i,j)\in E} (x_i - x_j)^2$$



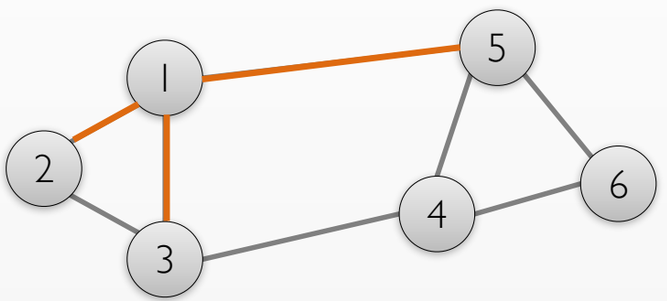
Add the constraint

$$\sum_{i=1}^{n} x_i = 0 \quad \text{and} \quad \| x \| = 1$$

$$y_i = \begin{cases} -1 & \text{if } x_i \le 0 \\ 1 & \text{if } x_i > 0 \end{cases}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Adjacency Matrix



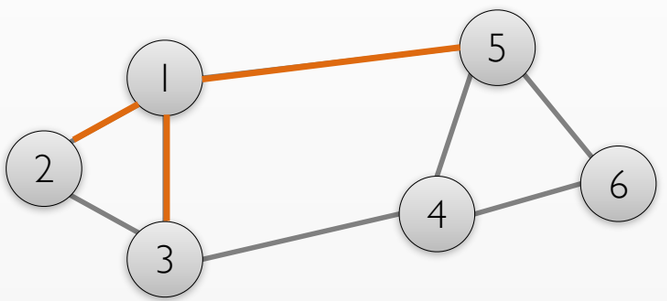
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

Contains 1 iff there is an edge between (i,j):

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Calculating Degrees



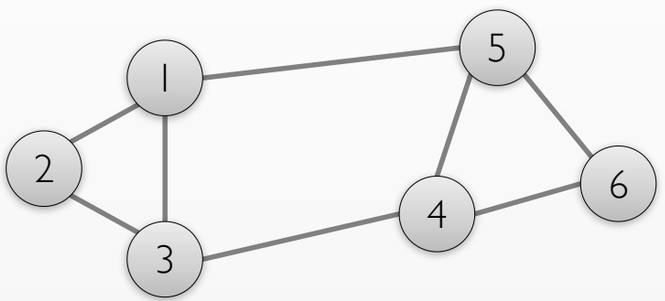
	1	2	3	4	5	6	
1	0	1	1	0	1	0	3
2	1	0	1	0	0	0	2
3	1	1	0	1	0	0	3
4	0	0	1	0	1	1	3
5	1	0	0	1	0	1	3
6	0	0	0	1	1	0	2
	3	2	3	3	3	2	

Degree for each node

$$d_i = \sum_j A_{ij}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Degree Matrix



	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

Matrix with degree of node on diagonal:

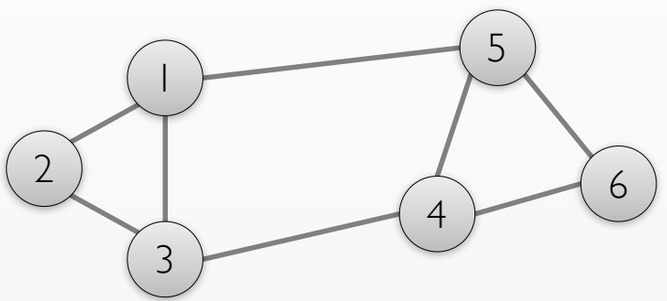
$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

$$D_{ij} = d_i \delta_{ij}$$

$$d_i = \sum_j A_{ij}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Laplacian Matrix



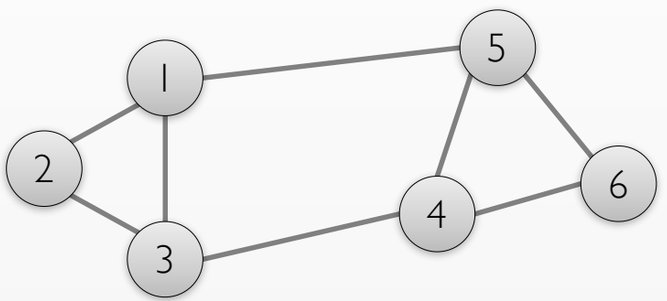
	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

Difference of Degree and Adjacency Matrix

$$L_{ij} = D_{ij} - A_{ij}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Laplacian Matrix



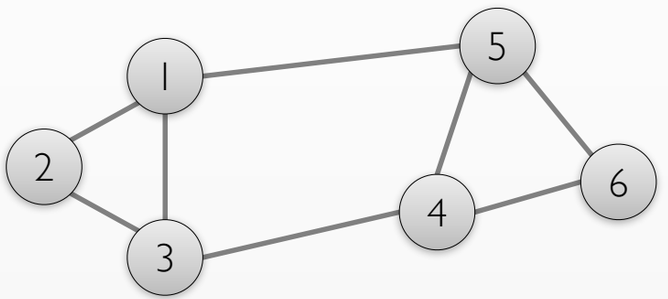
	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

Question: What is the value of the sum over columns or rows?

$$\sum_j L_{ij} = \sum_j (d_i \delta_{ij} - A_{ij}) = 0$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Eigenvectors of the Laplacian



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

Properties of the Laplacian

- Real-valued & Symmetric (eigenvectors orthogonal)

$$L\,\boldsymbol{x}_i = \lambda_i \boldsymbol{x}_i \qquad \boldsymbol{x}_i^{\top}\boldsymbol{x}_j = \delta_{ij}$$

- Rows/columns sum to 0 (constant vector is eigenvector)

$$\boldsymbol{x}_1 = \left[\frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}\right] \qquad L\boldsymbol{x}_1 = \lambda_1 \boldsymbol{x}_1 \qquad \lambda_1 = 0$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Second Eigenvector (Fiedler Vector)

First eigenvector (constant)

$$\boldsymbol{x}_1 = \left[\frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}\right] \propto \mathbf{1} \qquad \lambda_1 = 0$$

Second eigenvector (orthogonal to first vector)

$$\boldsymbol{x}_1^\top \boldsymbol{x}_2 = 0 \quad \rightarrow \quad \sum_{i=1}^{n} \boldsymbol{x}_{2,i} = 0$$

Vector with second smallest eigenvalue

$$\lambda_2 = \min_{\boldsymbol{x} \perp \boldsymbol{x}_1} \frac{\boldsymbol{x}^\top L\, \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}$$

Perpendicular to the first eigenvector

Here we arrange the eigenvalues and eigenvectors in an ascending order. In contrast, in PCA descending order was used

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Second Eigenvector (Fiedler Vector)

Relationship between Fiedler Vector and Graph Cuts

$$\boldsymbol{x}^\top L\boldsymbol{x} = \sum_{i,j}^{n} L_{ij}\,\boldsymbol{x}_i\boldsymbol{x}_j = \sum_{i,j}^{n}(D_{ij} - A_{ij})\,\boldsymbol{x}_i\boldsymbol{x}_j$$

$$= \sum_{i,j}^{n} d_i\,\delta_{ij}\,\boldsymbol{x}_i\boldsymbol{x}_j - \sum_{i,j}^{n} A_{ij}\,\boldsymbol{x}_i\boldsymbol{x}_j$$

$$= \sum_{i}^{n} d_i\,\boldsymbol{x}_i^2 - \sum_{(i,j)\in E} 2\,\boldsymbol{x}_i\boldsymbol{x}_j$$

$$= \sum_{(i,j)\in E} \underbrace{\boldsymbol{x}_i^2 + \boldsymbol{x}_j^2}\, - 2\,\boldsymbol{x}_i\boldsymbol{x}_j = \sum_{(i,j)\in E} (\boldsymbol{x}_i - \boldsymbol{x}_j)^2$$

$$\lambda_2 = \min_{\boldsymbol{x}\perp\boldsymbol{x}_1} \frac{\boldsymbol{x}^\top L\,\boldsymbol{x}}{\boldsymbol{x}^\top\boldsymbol{x}}$$

$$L_{ij} = D_{ij} - A_{ij}$$

$$D_{ij} = d_i\,\delta_{ij}$$

Replace sum over nodes with sum over edges

Sums to degree for each node when we count start *i* and end *j* for each edge

$$\sum_i d_i = 2|E|$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Second Eigenvector (Fiedler Vector)

Minimum Cut

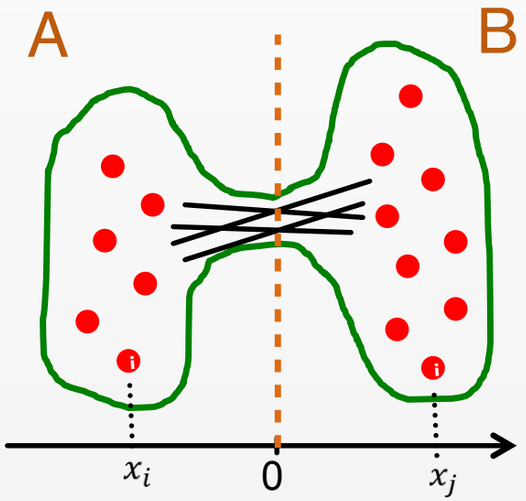
$$\min_{\boldsymbol{y}\in\{-1,1\}^n} \sum_{(i,j)\in E} (\boldsymbol{y}_i - \boldsymbol{y}_j)^2$$

Convert to a continuous optimization problem

$$\min_{x\in\mathbf{R}^n} \sum_{(i,j)\in E} (x_i - x_j)^2$$

Add the constraint

$$\sum_{i=1}^{n} x_i = 0$$



$$y_i = \begin{cases} -1 & \text{if } x_i \le 0 \\ 1 & \text{if } x_i > 0 \end{cases}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Second Eigenvector (Fiedler Vector)

$$\boldsymbol{x}_1^\top \boldsymbol{x}_2 = 0 \quad \rightarrow \quad \sum_{i=1}^{n} \boldsymbol{x}_{2,i} = 0$$

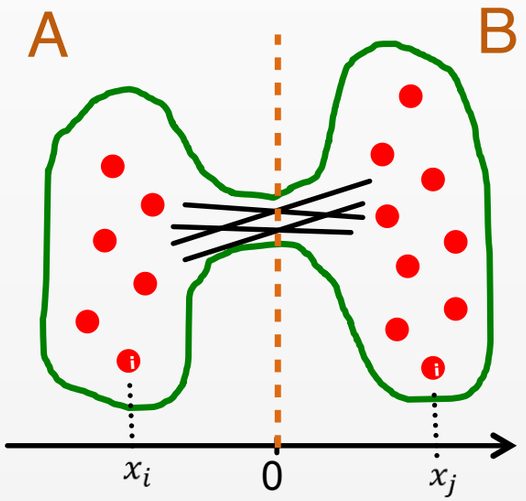
$$\lambda_2 = \min_{\boldsymbol{x} \perp \boldsymbol{x}_1} \frac{\boldsymbol{x}^\top L\, \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}$$

Second Smallest Eigenvalue

$$\lambda_2 \propto \boldsymbol{x}_2^\top L \boldsymbol{x}_2 = \sum_{(i,j)\in E} (\boldsymbol{x}_{2,i} - \boldsymbol{x}_{2,j})^2$$

Minimum Cut

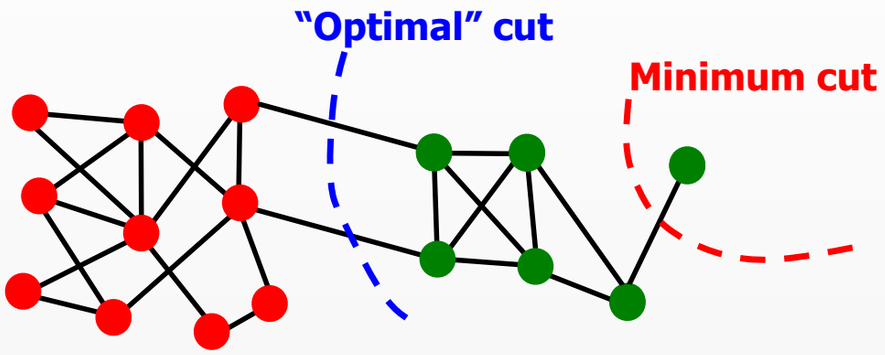
$$\min_{\boldsymbol{y}\in\{-1,1\}^n} \sum_{(i,j)\in E} (\boldsymbol{y}_i - \boldsymbol{y}_j)^2$$



$$\boldsymbol{y}_i = \begin{cases} -1 & \boldsymbol{x}_{2,i} \leq 0 \\ 1 & \boldsymbol{x}_{2,i} > 0 \end{cases}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

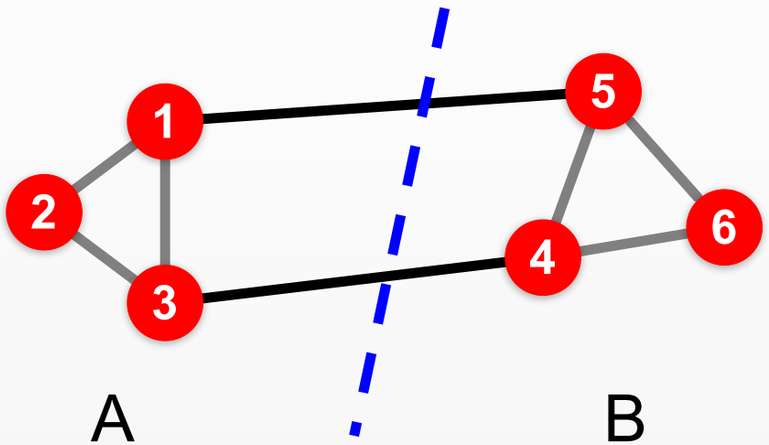
Normalized Cuts



Problem: minimal cut is not necessarily a good splitting criterion

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Normalized Cuts



$$\text{ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{vol}(A)} + \frac{\text{cut}(A,B)}{\text{vol}(B)}$$

Degree

$$d_i = \sum_j \boldsymbol{A}_{ij}$$

Volume

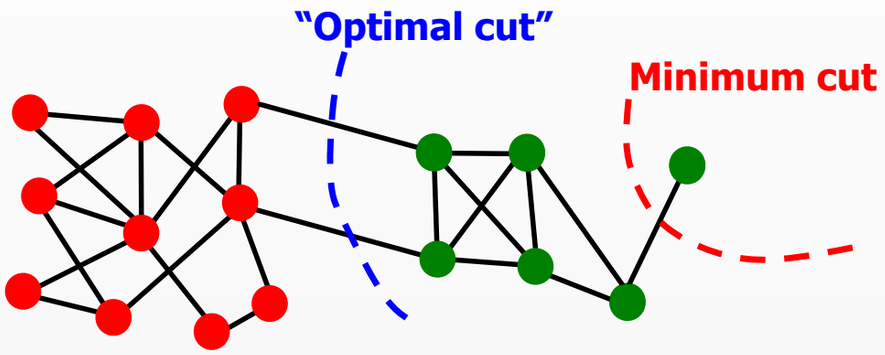
$$\text{vol}(B) = \sum_{j \in B} d_j$$

Cut

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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Solving Normalized Cuts



Solve using Normalized Laplacian

(*for derivation see:* Shi & Malik, IEEE TPAMI, 2000)

$$\lambda = \min_{\boldsymbol{y}} \text{ncut}(\boldsymbol{y}) = \min_{\boldsymbol{x} \perp D\mathbf{1}} \frac{\boldsymbol{x}^\top L \boldsymbol{x}}{\boldsymbol{x}^\top D \boldsymbol{x}} = \min_{\boldsymbol{z} \perp \mathbf{1}} \frac{\boldsymbol{z}^\top L^{\text{sym}} \boldsymbol{z}}{\boldsymbol{z}^\top \boldsymbol{z}}$$

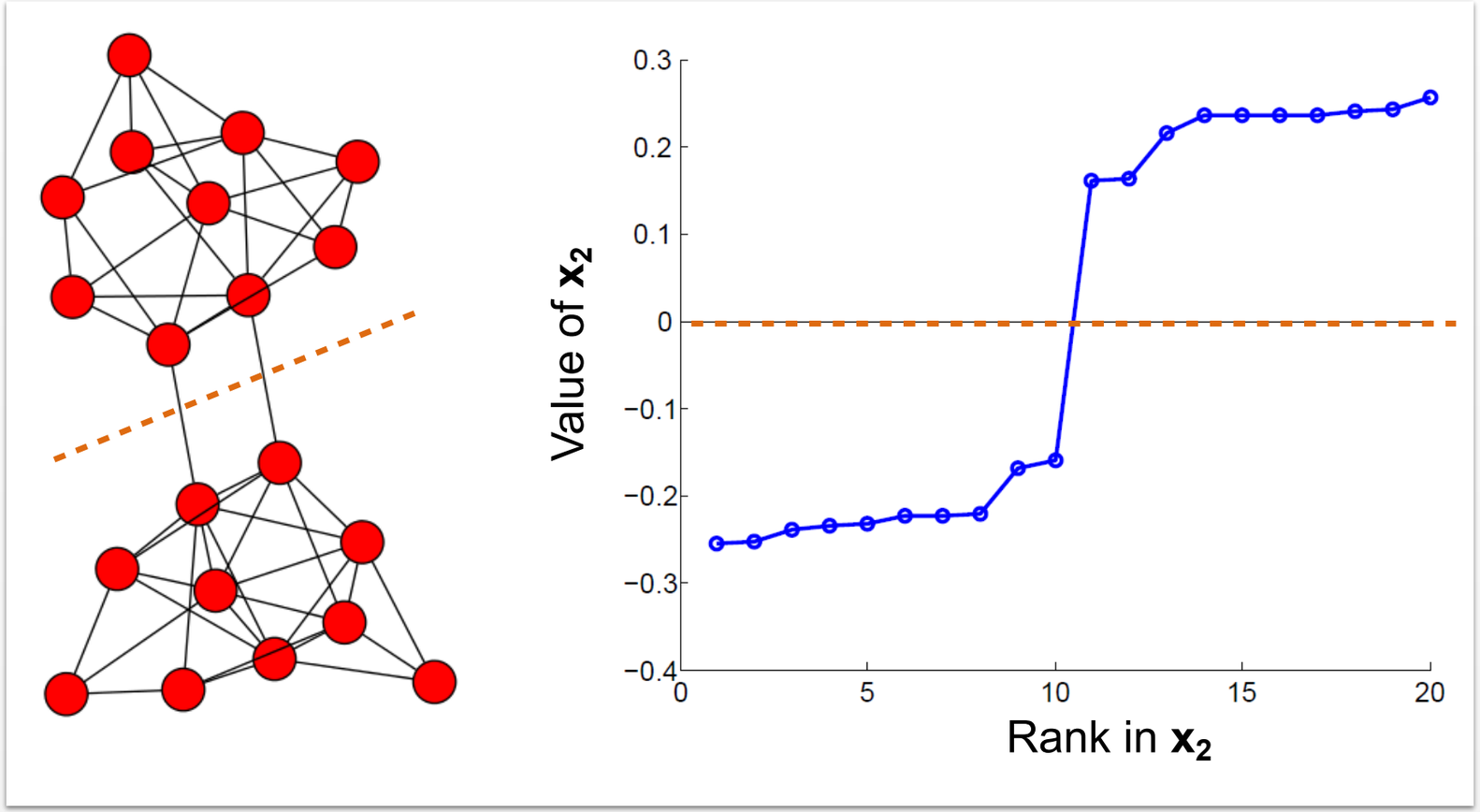
$$D\mathbf{1} = [d_1, d_2, \ldots, d_n]$$

$$z = D^{\frac{1}{2}} x$$

$$L^{\text{sym}} = D^{-1/2} L D^{-1/2}$$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

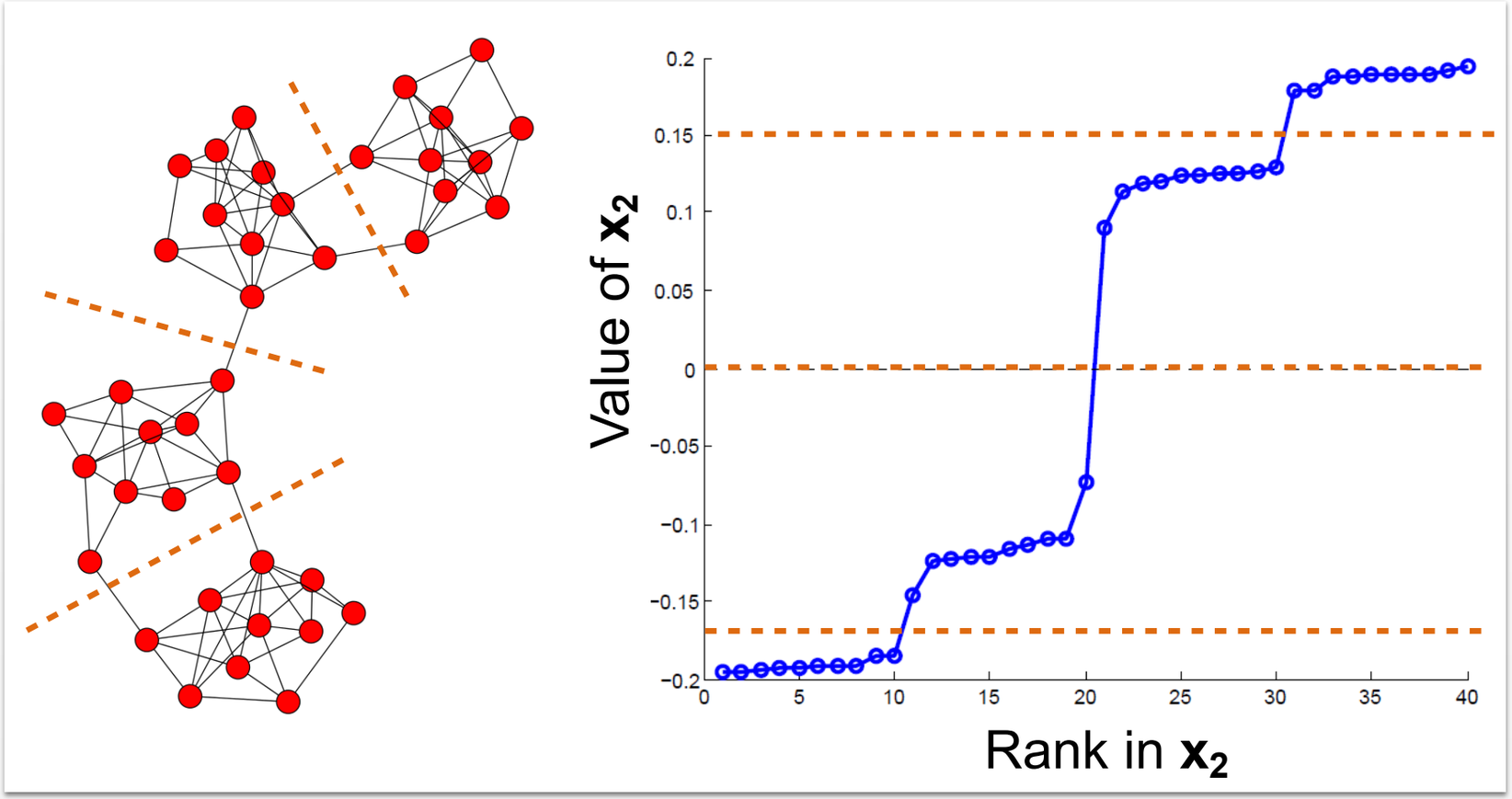
Example: Spectral Partitioning



Assign communities according to Fiedler vector values

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Example: Spectral Partitioning



Multiple communities show up as gaps in coordinate values

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

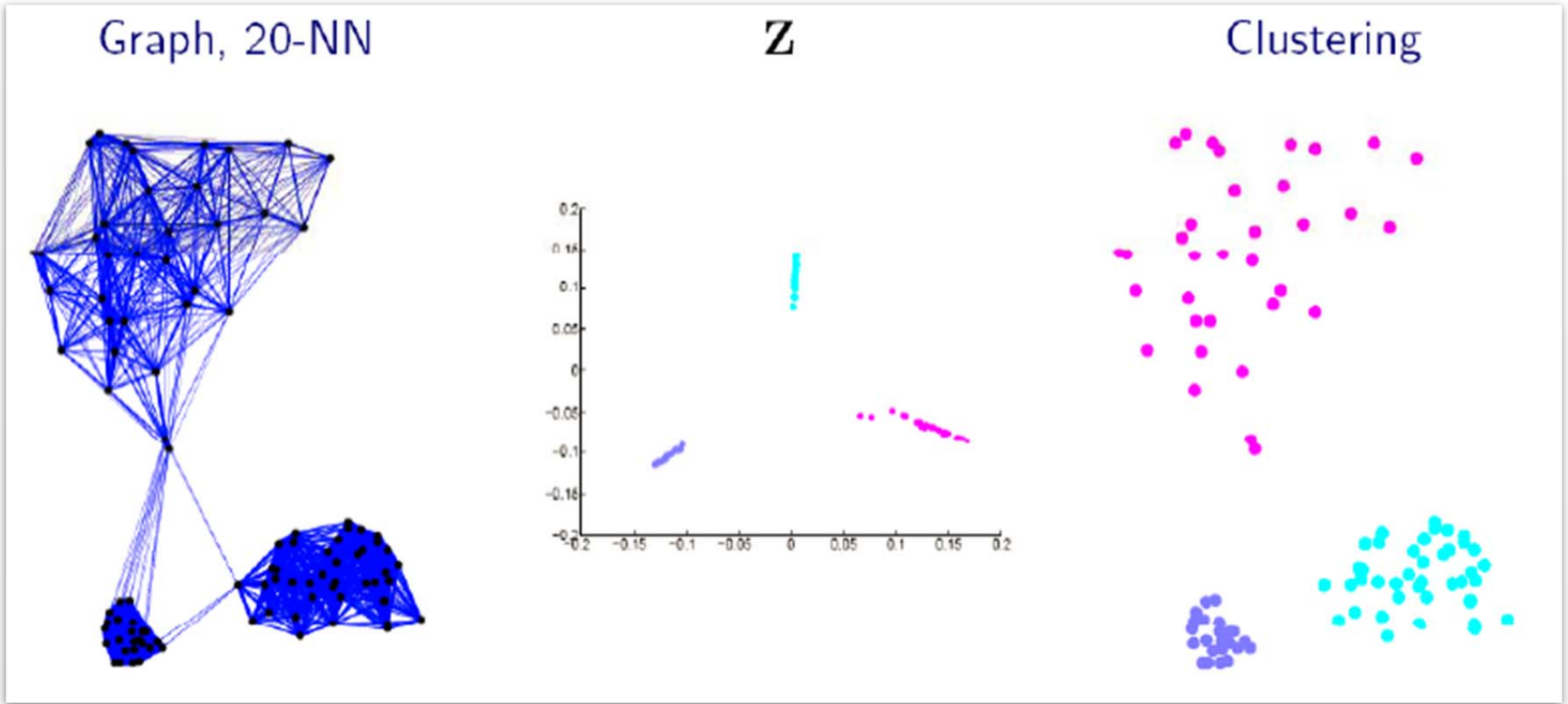
k-Way Spectral Clustering

- How do we partition a graph into k clusters?
- Two basic approaches:
  - Recursive bi-partitioning [Hagen et al., '92]
    - Hierarchical divisional clustering using normalized cut
    - Disadvantages: Inefficient, unstable
  - Cluster multiple eigenvectors [Shi-Malik, '00]
    - Calculate eigenvectors of normalized Laplacian
    - Perform dimensionality reduction (*same as in PCA*)
    - Cluster in reduced space (*e.g. using k-means*)

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

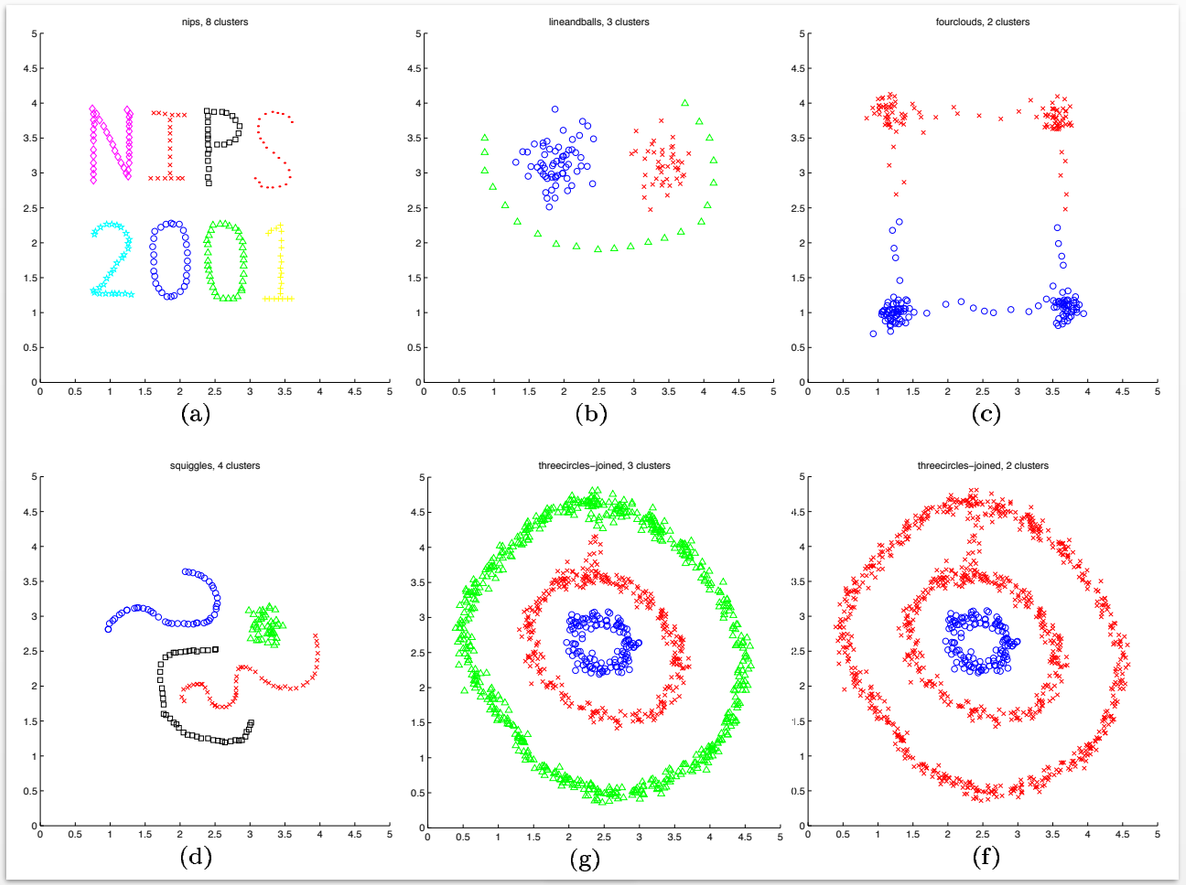
k-Way Spectral Clustering

Example: Clustering with 2 eigenvectors



Spectral Clustering as General-purpose Method

Define adjacency using a proximity/similarity measure
(e.g. a kernel $A_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$)



source: Ng, Jordan and Weiss, NIPS 2001

Spectral Clustering for Image Segmentation

Use RBF kernel: $A_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-|\boldsymbol{x}_i - \boldsymbol{x}_j|^2 / l^2)$