Spectral Clustering

Seungjin Choi

Department of Computer Science and Engineering
Pohang University of Science and Technology
77 Cheongam-ro, Nam-gu, Pohang 37673, Korea
seungjin@postech.ac.kr

Spectral Clustering?

- Spectral methods
  - Methods using eigenvectors of some matrices
  - Involve eigen-decomposition (or spectral decomposition)
- Spectral clustering methods: Algorithms that cluster data points using eigenvectors of matrices derived from the data
- Closely related to spectral graph partitioning
- Pairwise (similarity-based) clustering methods
  - Standard statistical clustering methods assume a probabilistic model that generates the observed data points
  - Pairwise clustering methods define a similarity function between pairs of data points and then formulates a criterion that the clustering must optimize

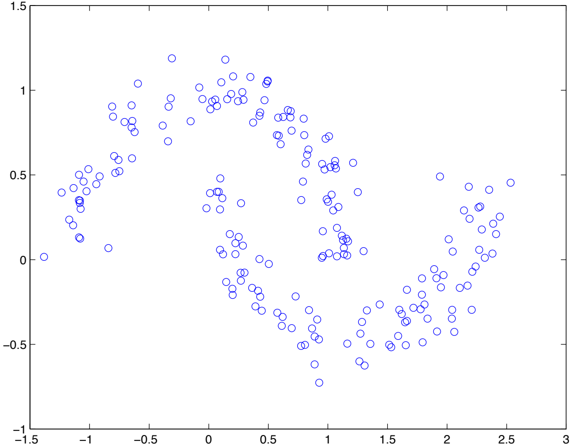
Spectral Clustering Algorithm: Bipartioning

1. Construct affinity matrix

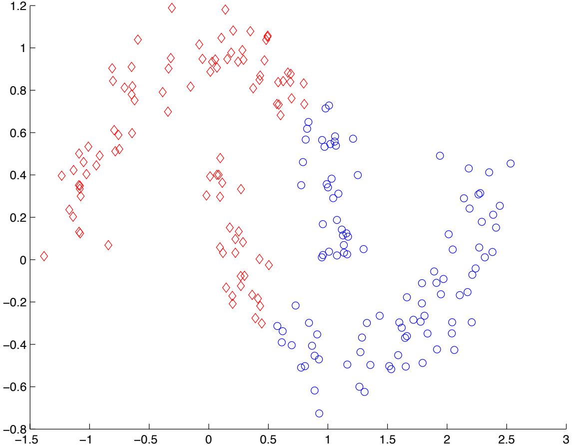
$$W_{ij} = \begin{cases} \exp\{-\beta\|v_i - v_j\|^2\} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}$$

2. Calculate the graph Laplacian $L = D - W$ where $D = \text{diag}\{d_1, \ldots, d_n\}$ and $d_i = \sum_j W_{ij}$ is the degree of node v_i.
3. Compute the second smallest eigenvector of the graph Laplacian (denoted by $u = [u_1 \cdots u_n]^\top$, Fiedler vector)
4. Partition u_i's by a pre-specified threshold value and assign data points v_i to cluster.

Two Moons Data

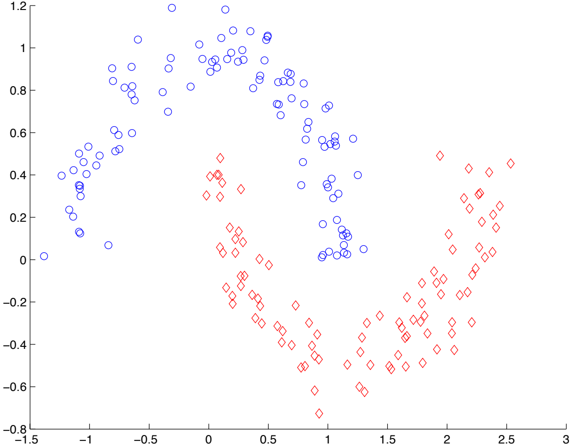


Two Moons Data: *k*-Means



Two Moons Data: Fiedler Vector

Two Moons Data: Spectral Clustering



Graphs

- Consider a connected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, \ldots, v_n\}$ and $\mathcal{E}$ denote a set of vertices and a set of edges, respectively, with pairwise similarity values being assigned as edge weights.
- Adjacency matrix (similarity, proximity, affinity matrix): $W = [W_{ij}] \in \mathbb{R}^{n \times n}$.
- Degree of nodes: $d_i = \sum_j W_{ij}$.
- Volume: $\mathrm{vol}(\mathcal{S}_1) = d_{\mathcal{S}_1} = \sum_{i \in \mathcal{S}_1} d_i$.

Neighborhood Graphs

Gaussian similarity function is given by

$$w(v_i, v_j) = W_{ij} = \exp\left\{-\frac{\|v_i - v_j\|^2}{2\sigma^2}\right\}.$$

- ϵ-neighborhood graph
- k-nearest neighbor graph

Graph Laplacian

(Unnormalized) graph Laplacian is defined as $L = D - W$.

1. For every vector $x \in \mathbb{R}^n$, we have

$$x^\top L x = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}(x_i - x_j)^2 \geq 0. \quad \text{(positive semidefinite)}$$

2. The smallest eigenvalue of L is 0 and the corresponding eigenvector is $\mathbf{1} = [1 \cdots 1]^\top$, since $D\mathbf{1} = W\mathbf{1}$, i.e, $L\mathbf{1} = 0$.
3. L has n nonnegative eigenvalues, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n = 0$.

$$
\begin{aligned}
x^\top L x &= x^\top D x - x^\top W x \\
&= \sum_{i=1}^{n} d_i x_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij} x_i x_j \\
&= \frac{1}{2}\left(\sum_i d_i x_i^2 - 2\sum_i\sum_j W_{ij} x_i x_j + \sum_j d_j x_j^2\right) \\
&= \frac{1}{2}\sum_i\sum_j W_{ij}(x_i - x_j)^2.
\end{aligned}
$$

Normalized Graph Laplacian

Two different normalization methods are popular, including:

- Symmetric normalization:

$$L_s = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}.$$

- Normalization related to random walks:

$$L_{rw} = D^{-1} L = I - D^{-1} W.$$

Remarks

1. For every vector $x \in \mathbb{R}^n$, we have

$$x^\top L_s x = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2 .$$

2. L_{sym} and L_{rw} are positive semidefinite and have n nonnegative real-valued eigenvalues, $\lambda_1 \geq \cdots \lambda_n = 0$.
3. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ is an eigenvalue of L_s with eigenvector $D^{1/2}u$.
4. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ and u solves the generalized eigenvalue problem $Lu = \lambda Du$.
5. 0 is an eigenvalue of L_{rw} with the constant one vector $\mathbf{1}$ as eigenvector. 0 is an eigenvalue of L_s with eigenvector $D^{1/2}\mathbf{1}$.

Unnormalized Spectral Clustering

1. Construct a neighborhood graph with corresponding adjacency matrix W.
2. Compute the unnormalized graph Laplacian $L = D - W$.
3. Find the k smallest eigenvectors of L and form the matrix $U = [u_1 \cdots u_k] \in \mathbb{R}^{n \times k}$.
4. Treating each row of U as a point in $\mathbb{R}^k$, cluster them into k groups using k-means algorithm.
5. Assign v_i to cluster j if and only if row i of U is assigned to cluster j.

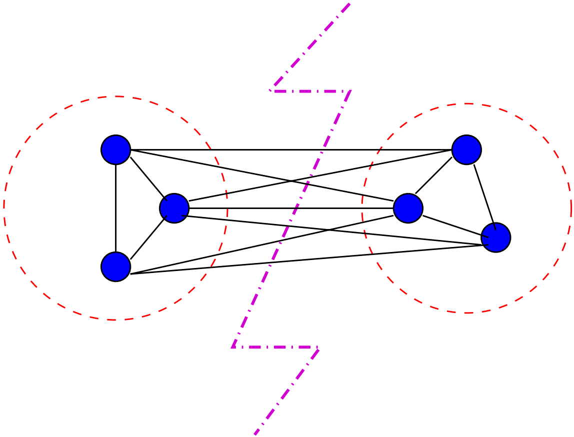
Normalized Spectral Clustering: Shi-Malik

1. Construct a neighborhood graph with corresponding adjacency matrix W.
2. Compute the unnormalized graph Laplacian $L = D - W$.
3. Find the k smallest generalized eigenvectors $u_1, \ldots, u_k$ of the problem $Lu = \lambda Du$ and form the matrix $U = [u_1 \cdots u_k] \in \mathbb{R}^{n \times k}$.
4. Treating each row of U as a point in $\mathbb{R}^k$, cluster them into k groups using k-means algorithm.
5. Assign v_i to cluster j if and only if row i of U is assigned to cluster j.

Normalized Spectral Clustering: Ng-Jordan-Weiss

1. Construct a neighborhood graph with corresponding adjacency matrix W.
2. Compute the normalized graph Laplacian $L_s = D^{-1/2} L D^{-1/2}$.
3. Find the k smallest eigenvectors $u_1, \ldots, u_k$ of L_s and form the matrix $U = [u_1 \, \cdots u_k] \in \mathbb{R}^{n \times k}$.
4. Form the matrix $\widetilde{U}$ from U by re-normalizing each row of U to have unit norm, i.e., $\widetilde{U}_{ij} = U_{ij} / (\sum_j U_{ij})^{1/2}$.
5. Treating each row of $\widetilde{U}$ as a point in $\mathbb{R}^k$, cluster them into k groups using k-means algorithm.
6. Assign v_i to cluster j if and only if row i of $\widetilde{U}$ is assigned to cluster j.

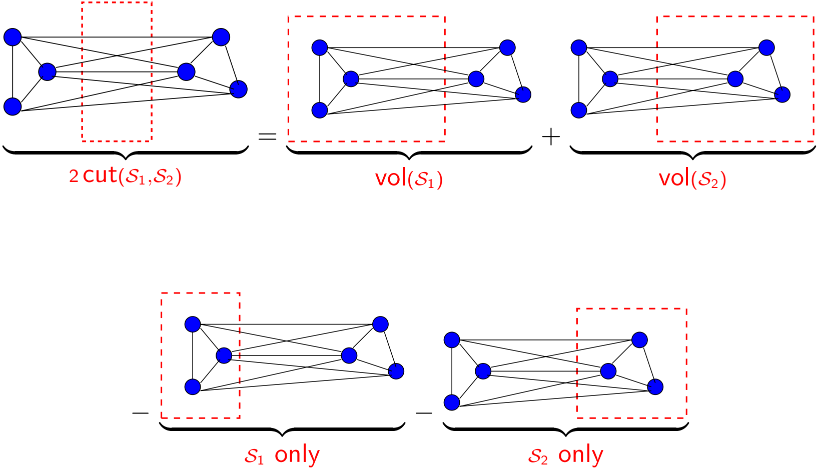
Pictorial Illustration of Graph Partitioning



Graph Partitioning: Bipartitioning

- Consider a connected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, \ldots, v_n\}$ and $\mathcal{E}$ denote a set of vertices and a set of edges, respectively, with pairwise similarity values being assigned as edge weights.
- Graph bipartitioning involves taking the set $\mathcal{V}$ apart into two coherent groups, $\mathcal{S}_1$ and $\mathcal{S}_2$, satisfying $\mathcal{V} = \mathcal{S}_1 \cup \mathcal{S}_2$, ($|\mathcal{V}| = n$), and $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$, by simply cutting edges connecting the two parts
- Adjacency matrix (similarity, proximity, affinity matrix): $W = [W_{ij}] \in \mathbb{R}^{n \times n}$.
- Degree of nodes: $d_i = \sum_j W_{ij}$.
- Volume: $\text{vol}(\mathcal{S}_1) = d_{\mathcal{S}_1} = \sum_{i \in \mathcal{S}_1} d_i$.

Pictorial Illustration: Cut and Volume



Graph Partitioning

The task is to find k disjoint sets, $\mathcal{S}_1, \ldots, \mathcal{S}_k$, given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{S}_1 \cap \cdots \cap \mathcal{S}_k = \phi$ and $\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_k = \mathcal{V}$ such that a certain cut criterion is minimized.

1. Bipartitioning: $\text{cut}(\mathcal{S}_1, \mathcal{S}_2) = \sum_{i \in \mathcal{S}_1} \sum_{j \in \mathcal{S}_2} W_{ij}$.
2. Multiway partitioning: $\text{cut}(\mathcal{S}_1, \ldots, \mathcal{S}_k) = \sum_{i=1}^{k} \text{cut}(\mathcal{S}_i, \overline{\mathcal{S}}_i)$.
3. Ratio cut: $\text{Rcut}(\mathcal{S}_1, \ldots, \mathcal{S}_k) = \sum_{i=1}^{k} \frac{\text{cut}(\mathcal{S}_i, \overline{\mathcal{S}}_i)}{|\mathcal{S}_i|}$.
4. Normalized cut: $\text{Ncut}(\mathcal{S}_1, \ldots, \mathcal{S}_k) = \sum_{i=1}^{k} \frac{\text{cut}(\mathcal{S}_i, \overline{\mathcal{S}}_i)}{\text{vol}(\mathcal{S}_i)}$.

Cut: Bipartitioning

The degree of dissimilarity between $\mathcal{S}_1$ and $\mathcal{S}_2$ can be computed by the total weights of edges that have been removed.

$$
\begin{aligned}
\text{Cut}(\mathcal{S}_1, \mathcal{S}_2) &= \sum_{i \in \mathcal{S}_1} \sum_{j \in \mathcal{S}_2} W_{ij} \\
&= \frac{1}{2} \left\{ \sum_{i \in \mathcal{S}_1} d_i + \sum_{j \in \mathcal{S}_2} d_j - \sum_{i \in \mathcal{S}_1} \sum_{j \in \mathcal{S}_1} W_{ij} - \sum_{i \in \mathcal{S}_2} \sum_{j \in \mathcal{S}_2} W_{ij} \right\} \\
&= \frac{1}{4} \left\{ (q_1 - q_2)^\top L \, (q_1 - q_2) \right\},
\end{aligned}
$$

where $q_j = [q_{1j} \cdots q_{nj}]^\top \in \mathbb{R}^n$ is the indicator vector which represents partitions,

$$
q_{ij} = \begin{cases} 1, & \text{if } i \in \mathcal{S}_j \\ 0, & \text{if } i \notin \mathcal{S}_j \end{cases}, \quad \text{for } i = 1, \ldots, n \text{ and } j = 1, 2.
$$

Note that q_1 and q_2 are orthogonal, i.e., $q_1^\top q_2 = 0$.

Introducing bipolar indicator vector, $x = q_1 - q_2 \in \{+1, -1\}^n$, the cut criterion is simplified as

$$\mathrm{Cut}(\mathcal{S}_1, \mathcal{S}_2) = \frac{1}{4} x^\top L x.$$

The balanced cut involves the following combinatorial optimization problem

$$\arg\min_x x^\top L x$$
$$\text{subject to } \mathbf{1}^\top x = 0, \quad x \in \{1, -1\}.$$

Dropping the integer constrains (spectral relaxation), leads to the symmetric eigenvalue problem. The second smallest eigenvector of L corresponds to the solution, since the smallest eigenvalue of L is 0 and its associated eigenvector is $\mathbf{1}$. The second smallest eigenvector is known as Fiedler vector.

Suggested Further Readings

1. P. K. Chan, M. D. F. Schlag, and J. Y. Zien, "Spectral k-way ratio-cut partitioning and clustering," IEEE Trans. CAD of Integrated Circuits and Systems, 1994.
2. J. Shi and J. Malik, "Normalized cuts and image segmentation," CVPR-1997.
3. Y. Weiss, "Segmentation using eigenvectors: A unifying view," ICCV-1999.
4. M. Meila and J. Shi, "A random walks view of spectral segmentation," AISTATS-2001.
5. C. Ding *et al.*, "A min-max cut algorithm for graph partitioning and data clustering," ICDM-2001.
6. A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," NIPS-2001.
7. U. von Luxburg, "A tutorial on spectral clustering," MPI for Biological Cybernetics, TR-149, 2006.