A Tutorial on Spectral Clustering

Ahmad Mousavi

Department of Mathematics and Statistics
UMBC
Baltimore, MD 21250
smousav1@umbc.edu

November 2015

Advantages of Spectral Clustering

1. Can be efficiently implemented and solved by standard linear algebra software.
2. Very often outperforms traditional clustering algorithms such as k-means algorithm.

Disadvantages of Spectral Clustering

1. It is not clear why it works and what it does.
2. Not necessarily finds the best partition.

Goal of This Presentation

To give some intuition about this method.

Similarity Graphs

Given a set of data $x_1, x_2, \ldots, x_n$ and some notion of similarity $s_{ij} \geq 0$ between x_i and x_j, the intuitive goal of clustering is:
To divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar.

Idea:
Using a similarity graph $G = (V, E)$ to reformulate the problem.

Graph Notation:
Let $G = (V, E)$ be an undirected weighted graph such that $V = \{v_1, v_2, \ldots, v_n\}$ is the vertex set and $W = (w_{ij})$ is the weight matrix. Then, degree of vertex v_i

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

Similarity Graphs

For two not necessarily disjoint sets $A, B \subset V$

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

Two different ways of measuring the size of a subset $A \subset V$:

$$|A| := \textit{the number of vertices in } A$$

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

Definition

A subset A is called a connected component if it is connected and if there is no connection between sets A and $\bar{A}$.

Different Similarity Graphs

Several constructions to transform a given set $x_1, x_2, \ldots, x_n$ of data points with pairwise similarities s_{ij} or pairwise distances d_{ij} into a graph. When constructing similarity graphs, the goal is **to model local neighborhood relationships between the data points**.

The ϵ-neighborhood graph
Connect all points whose pairwise distances are smaller than ϵ.
Considered as unweighted graph.
The k-nearest neighbor graphs
Connect vertex v_i to v_j if v_j is among k-nearest neighbors of v_i. Directed graph. Two ways of making this graph undirected.
1. Simply remove directions: k-nearest neighbor graph.
2. Connect vertices v_i and v_j if both are in the k-nearest neighbors of each other: mutual k-nearest neighbor.
In both cases: after connecting the appropriate vertices, we weight the edges by similarity of their endpoints.

Similarity Graph and Graph Laplacians

The fully connected graph

Simply connect all the points with positive similarity and weight all of the edges by s_{ij}. Here, similarity function itself should represent local neighborhood relationships. An example is $s(x_i, x_j) = exp(\frac{-||x_i - x_j||^2}{2\sigma^2})$, where σ controls the width of the neighborhoods.

Note: How does the choice of the similarity graph influence the spectral clustering result? Open Question.

Graph Laplacians and their basic properties

In the following we always assume that G is an undirected, weighted graph with weight matrix W, where $w_{ij} = w_{ij} \geq 0$. Eigenvalues will always be ordered increasingly, respecting multiplicities. By "the first k eigenvectors" we refer to the eigenvectors corresponding to the k smallest eigenvalues.

The unnormalized graph Laplacian

The unnormalized graph Laplacian is defined as $L = D - W$.

Properties of L:

The matrix L satisfies the following properties:

1. For every $f \in \mathbb{R}^n$ we have $f^T L f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$.
2. L is symmetric and positive semidefinite.
3. The smallest eigenvalue of L is 0, the corresponding eigenvector is the constant one vector $\vec{1}$.
4. L has n nonnegative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

Theorem

Let G be an undirected graph with nonnegative weights. Then, the multiplicity k of the eigenvalue 0 of L equals the number of connected components $A_1, A_2, \ldots, A_k$ in the graph. The eigenspace of eigenvalue 0 is spanned by the indicator vectors $\vec{1}_{A_1}, \vec{1}_{A_2}, \ldots, \vec{1}_{A_k}$ of those components.

The normalized graph Laplacians

There are two matrices are called normalized graph Laplacian, which are closely related to each other and are defined as

$$L_{sym} := D^{\frac{-1}{2}} L D^{\frac{-1}{2}} = I - D^{\frac{-1}{2}} W D^{\frac{-1}{2}};$$

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

Proposition (Properties of L_{sym} and L_{rw}):

1. For every $f \in \mathbb{R}^n$ we have $f^T L_{sym} f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (f_i - f_j)^2$.

2. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ is an eigenvalue of L_{sym} with eigenvector $w = D^{\frac{1}{2}} u$.

3. λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ and u solve the generalized eigenvalue problem $Lu = \lambda Du$.

The normalized Graph Laplacians

4. 0 is an eigenvalue of L_{rw} with the constant one vector $\vec{1}$ as eigenvector. 0 is an eigenvalue of L_{sym} with eigenvector $D^{\frac{1}{2}}\ \vec{1}$.

5. L_{sym} and L_{rw} are positive semi-definite and have n nonnegative real-valued eigenvalues $0 = \lambda_1 \leq \ldots \lambda_n$.

Theorem

Let G be an undirected graph with non-negative weights. Then, the multiplicity k of the eigenvalue 0 of both L_{rw} and L_{sym} equals the number of connected components $A_1, A_2, \ldots, A_k$ in the graph. For L_{rw}, the eigenspace of 0 is spanned by the indicator vectors $\vec{1}_{A_i}$, of those components. For L_{sym}, the eigenspace of 0 is spanned by the vectors $D^{\frac{1}{2}}\vec{1}_{A_i}$.

Spectral Clustering Algorithms

We assume that our data consists of n "points" $x_1, x_2, \ldots, x_n$, which can be arbitrary objects and their similarity matrix $S = (s_{ij})$ is available.

Unnormalized spectral clustering

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

1. Construct a similarity graph by one of the ways described. Let W be its weighted adjaceny matrix.
2. Compute the unnormalized Laplacian L.
3. Compute the first k eigenvectors $u_1, u_2, \ldots, u_k$ of L.
4. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, u_2, \ldots, u_k$ as columns.
5. For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
6. Cluster the points $(y_i)_{i=1,\ldots,n} \in \mathbb{R}^k$ with the k-means algorithm into clusters $C_1, C_2, \ldots, C_k$.

Output: Clusters $A_1, A_2, \ldots, A_k$ with $A_i = \{j | y_j \in C_i\}$.

Normalized spectral clustering according to Shai and Malik (2000).
Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.
1. Construct a similarity graph by one of the ways described. Let W be its weighted adjaceny matrix.
2. Compute the unnormalized Laplacian L.
3. **Compute the first k eigenvectors $u_1, u_2, \ldots, u_k$ of the generalized eigenprolem** $Lu = \lambda Du$.
4. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, u_2, \ldots, u_k$ as columns.
5. For $i = 1, \ldots n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U.
6. Cluster the points $(y_i)_{i=1,\ldots n} \in \mathbb{R}^k$ with the k-means algorithm into clusters $C_1, C_2, \ldots, C_k$.
Output: Clusters $A_1, A_2, \ldots, A_k$ with $A_i = \{j | y_j \in C_i\}$.

Note that this algorithm uses the generalized eigenectors of L, which corresponds to the eigenvectors of the matrix L_{rw}.

Normalized spectral clustering according to Ng, Jordan and Weiss (2002).

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.
1. Construct a similarity graph by one of the ways described. Let W be its weighted adjaceny matrix.
2. Compute the normalized Laplacian L.
3. **Compute the first k eigenvectors $u_1, u_2, \ldots, u_k$ of L_{sym}.**
4. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, u_2, \ldots, u_k$ as columns.
5. Form the matrix $T \in \mathbb{R}^{n \times k}$ from U by normalizing the rows to norm 1.
6. For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of T.
7. Cluster the points $(y_i)_{i=1,\ldots n} \in \mathbb{R}^k$ with the k-means algorithm into clusters $C_1, C_2, \ldots C_k$.
Output: Clusters $A_1, A_2, \ldots, A_k$ with $A_i = \{j | y_j \in C_i\}$.

Graph Cut Point of View

Note:
In these algorithms, the main trick is to change the representation of the **abstract** data points x_i to points $y_i \in \mathbb{R}^k$. It is due to properties of the graph Laplacians that this change of representation is useful. This enhances the cluster-properties in the data, so that clusters can be trivially detected in the new representation using k-means.

Here, we will see how spectral clustering can be derived as an approximation to our partitioning problem. Given a similarity graph with adjacency matrix W, the simplest and most direct way to construct a partition is to solve

$$cut(A_1, A_2, \ldots, A_k) = \frac{1}{2} \sum_{i=1}^{k} W(A_i, \bar{A}_i).$$

In the case of $k = 2$, mincut problem is easy, although it is not practical!?

Graph cut point of view

One way to modify this problem is to request that clusters be reasonably large. The two most common functions to encode this are Ratiocut and the normalized Ncut. The definitions are

$$RatioCut(A_1, A_2, ..., A_k) := \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i, \bar{A}_i)}{|A_i|} = \sum_{i=1}^{k}\frac{cut(A_i, \bar{A}_i)}{|A_i|};$$

$$Ncut(A_1, A_2, \ldots, A_k) := \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i, \bar{A}_i)}{vol(A_i)} = \sum_{i=1}^{k}\frac{cut(A_i, \bar{A}_i)}{vol(A_i)}.$$

Note that the minimum of function $\sum_{i=1}^{k}\frac{1}{|A_i|}$ is achieved if all $|A_i|$ coincide. So, both objective functions try to achieve "balanced" clusters. Unfortunately, balancing condition makes the new problems NP-hard. In fact, spectral clustering is a way to solve relaxed versions of these problems.

Approximating RatioCut for k=2

Our goal is to solve the optimization problem

$$min \quad RatioCut(A, \bar{A}) \quad s.t. \;\; A \subset V. \tag{1}$$

Given a subset $A \subset V$, we define the vector $f = (f_1, f_2, \ldots, f_n)^T \in \mathbb{R}^n$ s.t.

$$f_i = \begin{cases} \sqrt{\frac{|\bar{A}|}{|A|}} & if \;\; v_i \in A \\ -\sqrt{\frac{|A|}{|\bar{A}|}} & if \;\; v_i \in \bar{A} \end{cases}$$

Now, we have

$$f^T L f = \frac{1}{2} \sum_{i,j=1} w_{ij} (f_i - f_j)^2$$

$$= \frac{1}{2} \sum_{i \in A, j \in \bar{A}} w_{ij} \left(\sqrt{\frac{|\bar{A}|}{|A|}} + \sqrt{\frac{|A|}{|\bar{A}|}} \right)^2$$

$$+ \frac{1}{2} \sum_{i \in \bar{A}, j \in A} w_{ij} \left(-\sqrt{\frac{|\bar{A}|}{|A|}} - \sqrt{\frac{|A|}{|\bar{A}|}} \right)^2$$

$$= cut(A, \bar{A}) \left(\frac{|\bar{A}|}{|A|} + \frac{|A|}{|\bar{A}|} + 2 \right)$$
$$= cut(A, \bar{A}) \left(\frac{|A| + |\bar{A}|}{|A|} + \frac{|A| + |\bar{A}|}{|\bar{A}|} \right)$$
$$= |V| RatioCut(A, \bar{A}).$$

Additionally, we have

$$f^T \vec{1} = \sum_{i=1}^{n} f_i = \sum_{i \in A} \sqrt{\frac{|\bar{A}|}{|A|}} - \sum_{i \in \bar{A}} \sqrt{\frac{|A|}{|\bar{A}|}} = |A| \sqrt{\frac{|\bar{A}|}{|A|}} - |\bar{A}| \sqrt{\frac{|A|}{|\bar{A}|}} = 0$$

Also, note that we have

$$\|f\|^2 = \sum_{i=1}^{n} f_i^2 = |A| \frac{|\bar{A}|}{|A|} + |\bar{A}| \frac{|A|}{|\bar{A}|} = n.$$

So, problem (1) can be equivalently written as

$$\begin{aligned} \underset{A \subset V}{\text{minimize}} \quad & f^T L f \\ \text{s.t.} \quad & f^T \vec{1} = 0, \ f \text{ defined as above}, \ \|f\| = \sqrt{n}. \end{aligned}$$

This is discrete optimization problem and still NP. So, the most obvious relaxation would be

$$\begin{aligned} \underset{f \in \mathbb{R}^n}{\text{minimize}} \quad & f^T L f \\ \text{s.t.} \quad & f^T \vec{1} = 0, \ \ \|f\| = \sqrt{n}. \end{aligned}$$

By Rayleigh-Ritz theorem, it can be shown that the solution of this problem is f the eigenvector corresponding to the second smallest eigenvalue of L. So, we can approximate RatioCut by f. To get a partition though, we can use k-means to cluster $f_i \in \mathbb{R}$ into two groups of C and $\bar{C}$ and then

$$\begin{cases} v_i \in A & \text{if } f_i \in C \\ v_i \in \bar{A} & \text{if } f_i \in \bar{C} \end{cases}$$

This is exactly unnormalized spectral clustering algorithm for $k = 2$.

Approximating RatioCut for arbitrary k

Given a partition of V into k sets $A_1, A_2, \ldots, A_k$, we define k indicator vectors $h_j = (h_{1j}, h_{2j}, \ldots, h_{nj})^T$ by (for $i = 1, \ldots, n; j = 1, \ldots, k$)

$$h_{ij} = \begin{cases} \frac{1}{\sqrt{|A_j|}} & \text{if } v_i \in A_j \\ 0 & \text{otherwise} \end{cases}$$

Then, set $H \in \mathbb{R}^{n \times k}$ containing h_j's as columns. Note that $H^T H = I$. Also, we can check that

$$h_i^T L h_i = \frac{cut(A_i, \bar{A}_i)}{|A_i|} = (H^T L H)_{ii}.$$

This results in

$$RatioCut(A_1, A_2, \ldots, A_k) = \sum_{i=1}^{k} h_i^T L h_i = \sum_{i=1}^{k} (H^T L H)_{ii} = Tr(H^T L H)$$

So, the problem of RatioCut can be rewritten as

$$\begin{aligned} \underset{A_1,A_2,\ldots,A_k}{\text{minimize}} \quad & Tr(H^T L H) \\ \text{s.t.} \quad & H^T H = I, \ H \text{ defined as above.} \end{aligned}$$

By letting entries of matrix H to also take real values, we can relax this problem and get

$$\begin{aligned} \underset{H \in \mathbb{R}^{n \times k}}{\text{minimize}} \quad & Tr(H^T L H) \\ \text{s.t.} \quad & H^T H = I \end{aligned}$$

Again, a version of Rayleigh-Ritz tells us that the solution of this problem is given by choosing H as the matrix containing the first k eigenvectors of L as columns. To get a partition, use k-means on the rows of H. This leads to the general unnormalized clustering algorithm.

Approximating Ncut

Similar techniques can be used to drive normalized clustering as relaxation of minimizing Ncut. In the case $k = 2$, we define cluster indicator vector f by

$$f_i = \begin{cases} \sqrt{\frac{vol(\bar{A})}{vol(A)}} & if \ v_i \in A \\ -\sqrt{\frac{vol(A)}{vol(\bar{A})}} & if \ v_i \in \bar{A} \end{cases}$$

In the same manner as before, we can show $(Df)^T\vec{1} = 0$, $f^T Df = vol(V)$ and $f^T Lf = vol(V)Ncut(A, \bar{A})$. Thus, we can rewrite the problem of minimizing Ncut by the equivalent problem

$$\begin{aligned} \underset{A \subset V}{\text{minimize}} \quad & f^T Lf \\ \text{s.t.} \quad & (Df)^T\vec{1} = 0, \ f \ as \ defined \ above, \ f^T Df = vol(V). \end{aligned}$$

Again, we relax the problem by allowing f to take arbitrary real values

$$\begin{aligned} \underset{f \in \mathbb{R}^n}{\text{minimize}} \quad & f^T L f \\ \text{s.t.} \quad & (Df)^T \vec{1} = 0, \ f^T D f = vol(V). \end{aligned}$$

By substituting $g := D^{\frac{1}{2}} f$ we would have

$$\begin{aligned} \underset{g \in \mathbb{R}^n}{\text{minimize}} \quad & g^T D^{\frac{-1}{2}} L D^{\frac{-1}{2}} g \\ \text{s.t.} \quad & g^T (D^{\frac{1}{2}} \vec{1}) = 0, \ \|g\|^2 = vol(V). \end{aligned}$$

Note that $D^{\frac{-1}{2}} L D^{\frac{-1}{2}} = L_{sym}$ and $D^{\frac{1}{2}} \vec{1}$ is the first eigenvector of L_{sym}. Hence, by Rayleigh-Ritz, the solution is given by the second eigenvector of L_{sym}. This means that $f = D^{\frac{-1}{2}} g$ is the second eigenvector of L_{rw} or equivalently the generalized eigenvector of $Lu = \lambda Du$.

In the case of $k > 2$ we define the indicator vectors $h_j = (h_{1j}, h_{2j}, \ldots, h_{nj})^T$ by (for $i = 1, \ldots n; j = 1, \ldots k$)

$$h_{ij} = \begin{cases} \frac{1}{\sqrt{vol(A_j)}} & if\ v_i \in A_j \\ 0 & otherwise \end{cases}$$

Then, set $H \in \mathbb{R}^{n \times k}$ containing h_j's as columns. Note that $H^T H = I$. Also, we can check that

$$h_i^T L h_i = \frac{cut(A_i, \bar{A}_i)}{vol(A_i)} = (H^T L H)_{ii}.$$

This results in

$$\begin{aligned} \underset{A_1, A_2, \ldots, A_k}{\text{minimize}} \quad & Tr(H^T L H) \\ \text{s.t.} \quad & H^T D H = I,\ H \text{ defined as above.} \end{aligned}$$

Relaxing discreteness condition and substituting $T = D^{\frac{1}{2}}H$ we obtain

$$\begin{aligned} \underset{T \in \mathbb{R}^{n \times k}}{\text{minimize}} \quad & Tr(T^T D^{\frac{-1}{2}} L D^{\frac{-1}{2}} T) \\ \text{s.t.} \quad & T^T T = I. \end{aligned}$$

Again, the solution is T containing the first k eigenvectors of L_{sym} as columns. So, $H = D^{\frac{-1}{2}} T$ consists of the first k eigenvectors of L_{rw} or the first k generalized eigenvectors of problem $Lu = \lambda Du$. This yields to normalized spectral clustering algorithm according to Shi and Malik.

Note:
Assume $A_1, A_2, \ldots, A_k$ is the exact solution of minimizing RatioCut and $B_1, B_2, \ldots, B_k$ is the solution of unnormalized spectral algorithm then RatioCut$(B_1, B_2, \ldots, B_k)$-RatioCut$(A_1, A_2, \ldots, A_k)$ can be arbitrarily large.

The similarity function itself
Induced local neighborhoods should be meaningful.
Depends on the domain of data.

Which type of similarity graph
There is no theoretical discussion on this, although authors suggest to use k-nearest neighbor graph as the first choice as it is simple to work with, results in a sparse adjacency matrix and empirically is less vulnerable to unsuitable choice of parameters than the other graphs.

The parameters of similarity graph

In general, experience shows that spectral clustering can be quite sensitive to the choice of parameters. Again, there is no theoretical study on this for finite samples. However, we would like to make sure that similarity graph is connected or only consists of few connected components and few or no isolated vertices since if the number of connected components is more than the clusters required then spectral clustering will return them which is not necessarily the best partition.

Note:

For n data points drawn i.i.d. from some underlying density in $\mathbb{R}^d$, when $n \to \infty$, the k-nearest neighbor and ϵ-neighborhood graphs will be connected if we choose k on the order of $log(n)$ and parameter ϵ as $(\frac{log(n)}{n})^d$; respectively.

Computing the eigenvectors

One has to compute the first k eigenvectors of a potentially large (and sparse if use k-nearest or ϵ-neighborhood graphs) graph Laplacian matrix . Most popular methods are Krylov subspace methods such as Lanchoz method. The speed of convergence of these algorithms mostly depends on $eigengap = |\lambda_k - \lambda_{k+1}|$. This gives a heuristic idea to choose the number of clusters.

The number of clusters

Choose the number of clusters k such that all the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$ are small enough and λ_{k+1} is relatively large. The justification is that in the ideal case we have k completely disconnected clusters, the eigenvalue 0 has multiplicity k and then $\lambda_{k+1} > 0$. However, this may not necessarily result in a good number of clusters and the type of data is important.

The k-means step

In fact, there is nothing principled about using k-means in the last step and people may use other methods as well. For example, some people use hyperplanes for this purpose.

Which graph Laplacian should be used

Here, we bring two reasons explaining why choosing normalized Laplacians are better:

Assume that $k = 2$. In general, clustering has two different objectives:

1. We want to find a partition such that points in different clusters are dissimilar to each other, that is we want to minimize the between-cluster similarity. In the graph setting, this means to minimize $cut(A, \bar{A})$.

2. We want to find a partition such that points in the same cluster are similar to each other, that is we want to maximize the within-cluster similarity. This means that $\sum_{i \in A, j \in A} w_{ij} = W(A, A)$ and $\sum_{i,j \in \bar{A}} w_{ij} = W(\bar{A}, \bar{A})$ should be maximized.

Both RatioCut and Ncut directly implement the first point by explicitly incorporating $cut(A, \bar{A})$ in the objective function. However, concerning the second point, both algorithms behave differently. Note that

$$W(A, A) = W(A, V) - W(A, \bar{A}) = vol(A) - cut(A, \bar{A})$$

So, within-cluster similarity is maximized if $vol(A)$ is large and and if $cut(A, \bar{A})$ is small, which is obtained in the case of Ncut. This does not happen in the case of RatioCut as we are dealing with $|A|$ and $|\bar{A}|$. Note that $|A|$ is not related to within-cluster similarity.

Some consider MinMaxCut criterion introduced by Ding, He, Zha, Gu, and Simon (2001) defined as

$$MinMaxCut(A, B) = \sum_{i=1}^{k} \frac{cut(A_i, \bar{A}_i)}{W(A_i, A_i)}$$

to consider our two goals at the same time. In practice, Ncut and MinMaxCut are often minimized by similar cuts. Also, relaxing MinMaxCut leads to exactly the same optimization problem as relaxing Ncut, namely to normalized spectral clustering with the eigenvectors of L_{rw}.

Note:
Between L_{sym} and L_{rw}, we advocate for using L_{rw} as using L_{sym} does not have any computational advantages. Note that that the eigenvectors of L_{rw} are cluster indicator vectors $\vec{1}_{A_i}$, while the eigenvectors of $L_{sym}$are additionally multiplied with $D^{\frac{1}{2}}$, which might lead to undesired artifacts.

References:

1. A tutorial on spectral clustering, Ulrike von Luxburg,Statistics and Computing December 2007, Volume 17, Issue 4, pp 395-416.

2. On Spectral Clustering: Analysis and an algorithm, Y. Ng , Michael I. Jordan , Yair Weiss, Advances in neural information processing systems, 2002.