EECS 553: Machine Learning (ECE) **University of Michigan**

Spectral Clustering

Winter 2023 **Clayton Scott**

1 Overview

Spectral clustering is a method for clustering that, unlike k-means and GMMs, is able to infer nonconvex clusters. The unlabeled data are represented by a similarity graph, and the clusters are inferred from the spectral decomposition of a matrix associated to the similarity graph known as the graph Laplacian. There are several ways to derive spectral clustering, a sign that it is a fundamental algorithm.

Spectral clustering can be viewed as first performing nonlinear dimensionality reduction (NLDR) in a certain way, and then applying k-means to the resulting embedding of the training data. When spectral clustering works well, this NLDR creates an embedding such that the clusters are well-separated blobs, suitable for clustering with k-means.

Let the data to be clustered be denoted $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$. Below k and K are both used to denote the number of clusters (this needs to be fixed).

2 Similarity Graphs

A *similarity graph* is a graph whose vertices are the n data points, and whose edges have nonnegative weights that reflect the similarity of their endpoints. A similarity graph can be represented by an $n \times n$ symmetric matrix

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & \ldots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \ldots & w_{nn} \end{bmatrix}$$

where $w_{ij} \geq 0$, $w_{ij} = w_{ji}$, and $w_{ij} > 0$ if and only if there is an edge between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. If $w_{ij} > 0$ we say $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are *adjacent*, and $\boldsymbol{W}$ is called the *weighted adjacency matrix*.

A similarity graph is typically defined by stipulating a graph structure (unweighted adjacency matrix) together with a similarity (weighting) function. Common examples of graph structures are

- *k-nearest neighbor graph*: every $\boldsymbol{x}_i$ is adjacent to its k nearest neighbors
- *ϵ-ball graph*: every $\boldsymbol{x}_i$ is adjacent to every $\boldsymbol{x}_j$ within a radius of ϵ
- *complete graph*: every $\boldsymbol{x}_i$ is adjacent to every other $\boldsymbol{x}_j$

The graph structures define which data points are adjacent/connected and therefore have nonzero weights between them. Two examples of similarity measures are

- *Constant*:
$$w_{ij} = \begin{cases} 1, \text{if } \boldsymbol{x}_i, \boldsymbol{x}_j \text{ adjacent} \\ 0, \text{otherwise} \end{cases}$$
- *Gaussian*:
$$w_{ij} = \begin{cases} \exp(-\frac{1}{2\sigma^2}||\boldsymbol{x}_i - \boldsymbol{x}_j||^2), & \text{if } \boldsymbol{x}_i, \boldsymbol{x}_j \text{ adjacent} \\ 0, & \text{otherwise} \end{cases}$$

Note that it would not makes sense to use the complete graph and constant weight function together, but the other combinations are all reasonable. A sample data set is shown in Figure 1, and a similarity graph for the same data set is shown in Figure 2.

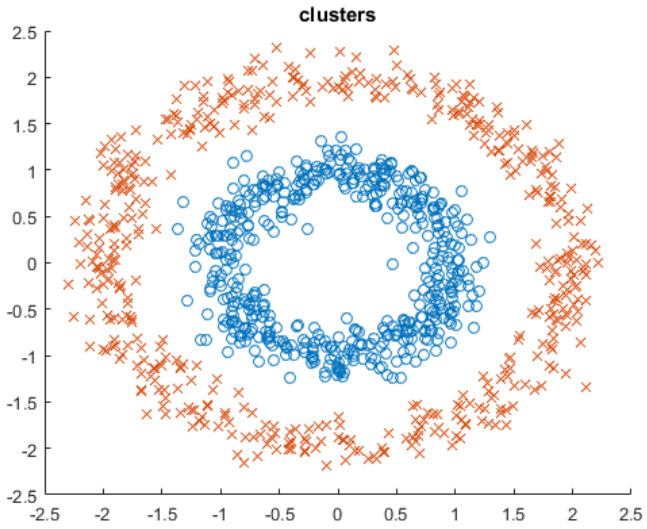

Figure 1: A data set with nonconvex clusters.

3 Graph Laplacians

Spectral clustering involves a matrix associated to the similarity graph called a graph Laplacian. In the section we define two types of graph Laplacians and cover some basic properties.

The *weighted degree* of a node $\boldsymbol{x}_i$ is

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

The *degree matrix* is the diagonal matrix

$$\boldsymbol{D} = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix}.$$

The *unnormalized graph Laplacian* is $\boldsymbol{L} \doteq \boldsymbol{D} - \boldsymbol{W}$. Note that $\boldsymbol{L}$ is independent of the self-similarity weights w_{ii} because

$$\boldsymbol{L}_{ii} = d_i - w_{ii} = \sum_{j \neq i}^{n} w_{ij}.$$

Let $\boldsymbol{f} = [f_1 \ \cdots \ f_n]^T$ denote an arbitrary vector in $\mathbb{R}^n$.

Proposition 1. *$\boldsymbol{L}$ satisfies the following properties.*

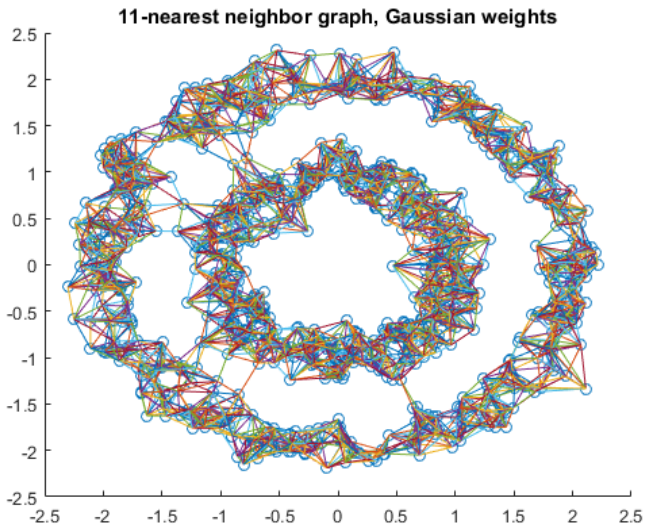

Figure 2: A similarity graph for the data in Figure 1. Only the graph structure is depicted.

1. *For every* $\boldsymbol{f} \in \mathbb{R}^n$

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

2. *The smallest eigenvalue of* $\boldsymbol{L}$ *is* 0*, and* $\mathbf{1} = [1, 1, \ldots, 1]^T \in \mathbb{R}^n$ *is a corresponding eigenvector.*

3. $\boldsymbol{L}$ *is symmetric and PSD.*

Proof. The first part follows from

$$\begin{aligned}
\boldsymbol{f}^T \boldsymbol{L} \boldsymbol{f} &= \boldsymbol{f}^T \boldsymbol{D} \boldsymbol{f} - \boldsymbol{f}^T \boldsymbol{W} \boldsymbol{f} \\
&= \sum_{i=1}^{n} d_i f_i^2 - \sum_{i,j} w_{ij} f_i f_j \\
&= \frac{1}{2} \left(\sum_{i=1}^{n} d_i f_i^2 - 2 \sum_{i,j} w_{ij} f_i f_j + \sum_{j=1}^{n} d_j f_j^2 \right) \\
&= \frac{1}{2} \sum_{i,j} w_{ij} (f_i^2 - 2 f_i f_j + f_j^2) \\
&= \frac{1}{2} \sum_{i,j} w_{ij} (f_i - f_j)^2.
\end{aligned}$$

The second property follows from the observation

$$\boldsymbol{L}\mathbf{1} = \boldsymbol{D}\mathbf{1} - \boldsymbol{W}\mathbf{1} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = 0 \cdot \mathbf{1}$$

To see the third property, note that $\boldsymbol{L}$ is symmetric because $\boldsymbol{D}$ and $\boldsymbol{W}$ are, and PSD by the first property and because $w_{ij} \geq 0$.

□

Let us assume that all vertex degrees are positive, which is equivalent to the graph not having any isolated points. The *normalized graph Laplacian* associated to the similarity graph $\boldsymbol{W}$ is defined to be

$$\tilde{\boldsymbol{L}} = \boldsymbol{D}^{-1}\boldsymbol{L}.$$

Proposition 2. *$\tilde{\boldsymbol{L}}$ satisfies the following properties.*

1. *For every $\boldsymbol{f} \in \mathbb{R}^n$*
$$\boldsymbol{f}^T\tilde{\boldsymbol{L}}\boldsymbol{f} = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2.$$
2. *The smallest eigenvalue of $\tilde{\boldsymbol{L}}$ is 0, and $\mathbf{1} = [1, 1, \ldots, 1]^T \in \mathbb{R}^n$ is a corresponding eigenvector.*
3. *$\tilde{\boldsymbol{L}}$ is PSD??? The von Luxborg tutorial says this is true but I now don't think it's correct. I don't think the property was needed so it's not a big deal. Counterexample:*
$$\boldsymbol{W} = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix}$$
and $\boldsymbol{f} = [-5\ 4]^T$. Then $\boldsymbol{f}^T\tilde{\boldsymbol{L}}\boldsymbol{f} < 0$. Feel free to check me on this and post to Piazza.

The proof is similar to the previous proposition and is left as an exercise.
Finally we state the following result whose proof is deferred to Section 6.

Proposition 3. *For both $\boldsymbol{L}$ and $\tilde{\boldsymbol{L}}$, the dimension of the zero eigenspace (i.e., the set of all eigenvectors with eigenvalue 0) is 1 if and only if the graph is connected.*

4 Spectral Clustering

We state the full spectral clustering algorithm below. This algorithm has not yet been motivated but is stated here for reference.

> Input: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$, desired number of clusters k, similarity graph parameters
> Construct similarity graph, and form the associated (unnormalized or normalized) graph Laplacian
> Determine k smallest eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_k$ of the graph Laplacian
> Let $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k \in \mathbb{R}^n$ be the k smallest eigenvectors
> Set $\boldsymbol{Y}^T = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k]$
> Let $\boldsymbol{y}_i$ be the i^{th} column of $\boldsymbol{Y}$, $i \in \{1, \ldots, n\}$ (in other words, $\boldsymbol{y}^T$ is the ith row of $\boldsymbol{Y}^T$)
> Cluster $\{\boldsymbol{y}_i\}_{i=1}^n$ using a clustering algorithm such as k-means clustering, and assign $\{\boldsymbol{x}_i\}_{i=1}^n$ to the corresponding clusters

In the next three sections, we present three different perspectives that motivate this algorithm. A plot of the quantities $\boldsymbol{y}_i$ is shown in Figure 3.

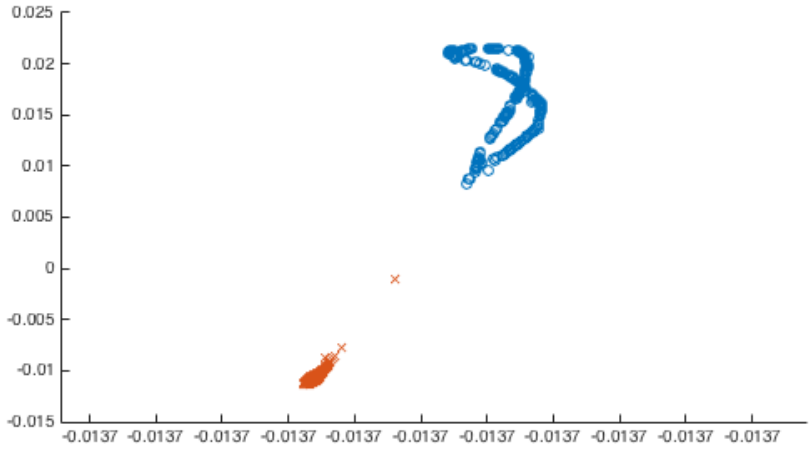

Figure 3: A scatter plot of the quantites $\boldsymbol{y}_i$ for the "rings" data shown earlier.

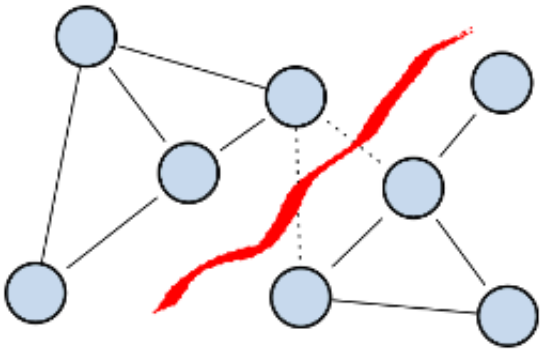

Figure 4: Illustration of a graph cut. Source: https://www.ml.uni-saarland.de/code/oneSpectralClustering/oneSpectralClustering.htm).

5 Spectral Clustering: Graph Cuts Perspective

A partition of $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is a collection of sets $A_1, \ldots, A_M$ such that every $\boldsymbol{x}_i$ is in one and only one A_m. Suppose we believe there to be K clusters. Given a similarity graph $\boldsymbol{W}$, we want to find a partition $A_1, \ldots, A_K$ of $\{1, 2, \ldots, n\}$, where each A_k is viewed as a cluster, such that:

- w_{ij} is large if $\boldsymbol{x}_i, \boldsymbol{x}_j$ are in the same cluster
- w_{ij} is small if $\boldsymbol{x}_i, \boldsymbol{x}_j$ are in different clusters.

To find a good partition, we will employ the idea of graph cuts. A *graph cut* consists of a set of edges such that, when they are removed from a graph, the nodes of the graph are partitioned into two or more sets. Every partition of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ can be described as a graph cut. See Figure 4.

Our approach is to define a numerical score that assesses the quality of an arbitrary graph cut, and then optimize this score over the set of all possible graph cuts. A first attempt to define a numerical score is the

so-called *cut* function,

$$\text{cut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{k=1}^{K} W(A_k, \overline{A_k})$$

where $W(A, B) = \sum_{i \in A} \sum_{j \in B} w_{ij}$ and $\overline{A}$ = complement of A. The problem of minimizing the cut function is called the *mincut* problem. Unfortunately mincut leads to small and often singleton clusters. Therefore some modifications have been proposed:

- *RatioCut* (Hagen and Kahng, 1992)

$$\text{RatioCut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{k=1}^{K} \frac{W(A_k, \overline{A_k})}{|A_k|}$$

  where $|A|$ = # of nodes in A.

- *Normalized Cut (Ncut)* (Shi and Malik, 2000)

$$\text{Ncut}(A_1, \ldots, A_K) = \frac{1}{2} \sum_{k=1}^{K} \frac{W(A_k, \overline{A_k})}{\text{vol}(A_k)}$$

  where $\text{vol}(A) = \sum_{i \in A} \sum_{j \in V} w_{ij}$ and V is the set of all nodes (vertices) in the graph.

Unfortunately, introducing these "balancing" terms causes these problems to be NP-hard. However, in both cases it is possible to approximately solve these problems in ways that we will make precise, and those approximate solutions correspond to spectral clustering with the unnormalized graph Laplacian (RatioCut) and normalized graph Laplacian (NCut).

5.1 Approximate solution to RatioCut, $K = 2$

We wish to solve:

$$\min_A \text{RatioCut}(A, \overline{A}) = \min_A \left[\frac{\text{cut}(A, \overline{A})}{|A|} + \frac{\text{cut}(A, \overline{A})}{|\overline{A}|} \right]$$

Given $A \subseteq \{1, 2, \ldots, n\}$, define $\boldsymbol{f}_A = (f_{A_1}, \ldots, f_{A_n})^T$ by:

$$f_{A_i} = \begin{cases} +\sqrt{|\overline{A}| \,/\, |A|} & : i \in A \\ -\sqrt{|A| \,/\, |\overline{A}|} & : i \notin A \end{cases}$$

Then:

$$\boxed{\boldsymbol{f}_A^T \boldsymbol{L} \boldsymbol{f}_A = n \cdot \text{RatioCut}(A, \overline{A})}$$

To see this, observe:

$$\begin{aligned}
\boldsymbol{f}_A^T \boldsymbol{L} \boldsymbol{f}_A &= \frac{1}{2} \sum_{i,j} w_{ij} \left(f_{A_i} - f_{A_j} \right)^2 \\
&= \frac{1}{2} \sum_{i \in A, j \in \overline{A}} w_{ij} \left(\sqrt{\frac{|\overline{A}|}{|A|}} + \sqrt{\frac{|A|}{|\overline{A}|}} \right)^2 + \sum_{i \in \overline{A}, j \in A} w_{ij} \left(-\sqrt{\frac{|A|}{|\overline{A}|}} - \sqrt{\frac{|\overline{A}|}{|A|}} \right)^2 \\
&= \frac{1}{2} \mathrm{cut}(A, \overline{A}) \left(\frac{|\overline{A}|}{|A|} + \frac{|A|}{|\overline{A}|} + 2 \right) + \frac{1}{2} \mathrm{cut}(A, \overline{A}) \left(\frac{|A|}{|\overline{A}|} + \frac{|\overline{A}|}{|A|} + 2 \right) \\
&= \mathrm{cut}(A, \overline{A}) \left(\frac{|\overline{A}|}{|A|} + \frac{|A|}{|\overline{A}|} + 2 \right) \\
&= \mathrm{cut}(A, \overline{A}) \left(\frac{|A| + |\overline{A}|}{|\overline{A}|} + \frac{|\overline{A}| + |A|}{|A|} \right) \\
&= n \left(\frac{\mathrm{cut}(A, \overline{A})}{|\overline{A}|} + \frac{\mathrm{cut}(A, \overline{A})}{|A|} \right) \\
&= n \cdot \mathrm{RatioCut}(A, \overline{A}).
\end{aligned}$$

Furthermore, $\boldsymbol{f}_A$ satisfies the following two properties:

1) $\mathbf{1}^T \boldsymbol{f}_A = 0$

2) $\|\boldsymbol{f}_A\|^2 = n$

Therefore, RatioCut can be written as the following combinatorial optimization problem:

$$\begin{aligned}
\min_{A \subset \{1, \ldots, n\}} \quad & \boldsymbol{f}_A^T \boldsymbol{L} \boldsymbol{f}_A \\
s.t. \quad & \mathbf{1}^T \boldsymbol{f}_A = 0 \\
& \|\boldsymbol{f}_A\| = \sqrt{n}.
\end{aligned}$$

Note that it would still be RatioCut without the above two constraints, but we include them to keep the relaxation close to the original problem. A *relaxation* of this problem is:

$$\begin{aligned}
\min_{\boldsymbol{f} \in \mathbb{R}^n} \quad & \boldsymbol{f}^T \boldsymbol{L} \boldsymbol{f} \\
s.t. \quad & \mathbf{1}^T \boldsymbol{f} = 0 \\
& \|\boldsymbol{f}\| = \sqrt{n}.
\end{aligned}$$

The key difference between the relaxation and the original problem is that the original problem is a combinatorial optimization problem over a discrete set of vectors, while the relaxation optimizes over all length n vectors and is therefore a continuous optimization problem. The reason for studying continuous relaxations is that they are often easier to solve numerically, e.g., by gradient-based methods. In our case, the relaxation is a well-known eigenvalue problem.

Let us assume that the underlying graph is connected. Otherwise, we could just cluster each connected component separately. Then the solution of the relaxation is an eigenvector of $\boldsymbol{L}$ corresponding to the *second* smallest eigenvalue of $\boldsymbol{L}$. The reason is that the all ones vector is an eigenvector corresponding to eigenvalue

0, and by Prop. 3, the second smallest eigenvalue is > 0. We may therefore apply the generalized Rayleigh quotient theorem from the PCA notes.

Having solved the relaxation, the final step is to recover a solution of the original RatioCut problem. For $k = 2$, there are several ways to do this. One simple strategy is to simply take the sign of the entries of the solution $\boldsymbol{f}^* = [f_1^* \ \cdots \ f_n^*]^T$ to the relaxation. A better option is to apply the 2-means algorithm to the scalar values f_i^*. This is equivalent to applying 2-means to the two-dimensional points

$$\boldsymbol{y}_i := [1 \ f_i^*].$$

This is the recovery technique that generalizes naturally to $k > 2$.

For $k = 2$, we can actually do something better than 2-means. Consider a threshold and declare each data point $\boldsymbol{x}_i$ with f_i^* above the threshold to be in one cluster, and those with f_i^* below the threshold to be in the other cluster. The threshold may be swept from $\max f_i^*$ to $\min f_i^*$, and the threshold with smallest RatioCut value may be selected.

5.2 Other Approximations

For RatioCut with $k > 2$, a similar relaxation can be applied. In this case there is an indicator vector for each cluster $\ell = 1, \ldots, k$. Collecting these indicator vectors into a matrix $\boldsymbol{F}$, and using similar calculations as above, the relaxation of RatioCut is

$$\begin{aligned} \min_{\boldsymbol{F} \in \mathbb{R}^{n \times k}} \quad & \operatorname{tr}(\boldsymbol{F}^T \boldsymbol{L} \boldsymbol{F}) \\ s.t. \quad & \boldsymbol{F}^T \boldsymbol{F} = \boldsymbol{I}_{k \times k}. \end{aligned}$$

See von Luxborg (2007) for details. The solution to this problem is again given by the generalized Rayleigh quotient, and is equal to the $\boldsymbol{F}$ whose columns are the k smallest eigenvectors of $\boldsymbol{L}$. To recover a solution to the original RatioCut problem, let the solution of the relaxation be $\boldsymbol{F} = [f_{i\ell}^*]$, and set

$$\boldsymbol{y}_i = [f_{i1}^* \ \cdots \ f_{ik}^*]^T.$$

A clustering may now be obtained by applying k-means to $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$.

For NCut, a similar argument shows that a relaxation-based solution is given by spectral clustering with the normalized graph Laplacian. See von Luxborg (2007) for details.

Remark 1. There is no guarantee that the solution to the relaxation is a good approximate solution to the original problem. The gap between the optimal value of RatioCut/Ncut and the optimal value of its relaxation can be arbitrarily large.

6 Spectral Clustering: Perturbation Perspective

This derivation of spectral clustering first examines the ideal setting where the similarity graph perfectly captures the clusters, in which case the zero-eigenspace of the graph Laplacian perfectly encodes the clusters, and then argues that that encoding is still approximately true in a realistic setting. The encoding is described by the following result, which extends Prop. 3.

For $A \subseteq \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, define the indicator vector

$$\mathbf{1}_A = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \in \mathbb{R}^n \quad \text{where} \quad f_i = \begin{cases} 1 & \boldsymbol{x}_i \in A \\ 0 & \boldsymbol{x}_i \notin A \end{cases}.$$

Also note that the 0-eigenspace of a matrix, i.e., the subspace of all eigenvectors associated with the eigenvalue 0, is just the nullspace of that matrix.

Proposition 4. *If the graph has connected components $A_1, \ldots, A_k$, then the nullspace of both $\boldsymbol{L}$ and $\tilde{\boldsymbol{L}}$ has dimension k and is spanned by $\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}$.*

Proof. We focus on the unnormalized graph Laplacian $\boldsymbol{L}$. The proof for $\tilde{\boldsymbol{L}}$ is similar. The nullspace of $\boldsymbol{L}$ is $N(\boldsymbol{L}) = \{\boldsymbol{f} : \boldsymbol{L}\boldsymbol{f} = \mathbf{0}\}$. It suffices to show:

- $\mathbf{1}_{A_\ell} \in N(\boldsymbol{L})$ for each $\ell = 1, \ldots, k$
- If $\boldsymbol{f} \in N(\boldsymbol{L})$, then:

$$\boldsymbol{f} = \sum_{\ell=1}^{k} \alpha_\ell \mathbf{1}_{A_\ell}$$

  for some $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$.

The two properties imply that $\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}$ span $N(\boldsymbol{L})$. Since $\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_k}$ are clearly linearly independent, it follows that $dim(N(\boldsymbol{L})) = k$.

First, consider the case $k = 1$. In this case, we have established $\mathbf{1} \in N(\boldsymbol{L})$ previously. To show the spanning property, suppose $\boldsymbol{f} \in N(\boldsymbol{L})$. Then $\boldsymbol{L}\boldsymbol{f} = \mathbf{0}$, and so

$$0 = \boldsymbol{f}^T \boldsymbol{L} \boldsymbol{f} = \frac{1}{2} \sum_{i,j} w_{i,j} (f_i - f_j)^2.$$

If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are adjacent, then $w_{ij} > 0$ which implies $f_i = f_j$. More generally, since $k = 1$, any two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are connected by a path, and therefore $f_i = f_j$. Thus all f_i are equal to a constant, and therefore $\boldsymbol{f}$ is a multiple of $\mathbf{1}$.

If $k > 1$, let us suppose the data are enumerated such that $\boldsymbol{L}$ is block diagonal:

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_1 & & & \\ & \boldsymbol{L}_2 & & \\ & & \ddots & \\ & & & \boldsymbol{L}_k \end{bmatrix}.$$

Notice that $\boldsymbol{L}_\ell$ is the graph Laplacian on A_ℓ. Applying the previous case, we deduce that $\boldsymbol{L}\mathbf{1}_{A_\ell} = \mathbf{0}$ for each ℓ, and if $\boldsymbol{L}\boldsymbol{f} = \mathbf{0}$, then $\boldsymbol{f}$ is piecewise constant on each A_k. This implies that $\boldsymbol{f} = \sum_{k=1}^{K} \alpha_k \mathbf{1}_{A_k}$. □

A simple corollary motivates the spectral clustering algorithm.

Corollary 1. *If $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k\} \subset \mathbb{R}^n$ is a basis of $N(\boldsymbol{L})$ and we define*

$$\boldsymbol{y}_i = [u_{1,i} \ \cdots \ u_{k,i}]^T \in \mathbb{R}^k,$$

then $\boldsymbol{y}_i = \boldsymbol{y}_j$ iff $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are in the same connected component.

Rather than attempt a formal proof, it's more instructive to illustrate the idea. Consider the graph depicted in Figure 5. For this example,

$$\boldsymbol{L} = \begin{bmatrix} \bullet & & & & & \bullet & \\ & \bullet & & \bullet & & \bullet & \bullet \\ & & \bullet & \bullet & & & \\ & & \bullet & \bullet & \bullet & & \\ & \bullet & & & \bullet & & \bullet \\ & & & \bullet & & \bullet & \\ \bullet & \bullet & & & & \bullet & \\ & \bullet & & \bullet & & & \bullet \end{bmatrix}$$

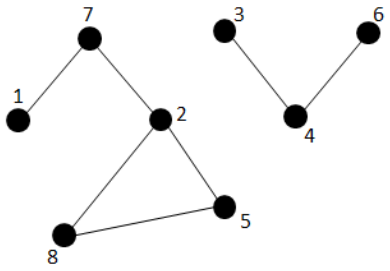

Figure 5: A simple graph to illustrate Cor. 1.

where the •s denote nonzero entries. Then

$$\mathbf{1}_{A_1} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{1}_{A_2} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}.$$

Now suppose $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ are a basis of $N(\boldsymbol{L})$. Write:

$$\boldsymbol{u}_1 = \alpha_1 \mathbf{1}_{A_1} + \beta_1 \mathbf{1}_{A_2},$$

$$\boldsymbol{u}_2 = \alpha_2 \mathbf{1}_{A_1} + \beta_2 \mathbf{1}_{A_2}.$$

Then

$$\begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \\ \boldsymbol{y}_3 \\ \boldsymbol{y}_4 \\ \boldsymbol{y}_5 \\ \boldsymbol{y}_6 \\ \boldsymbol{y}_7 \\ \boldsymbol{y}_8 \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_2 \\ \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \\ \beta_1 & \beta_2 \\ \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \\ \alpha_1 & \alpha_2 \\ \alpha_1 & \alpha_2 \end{bmatrix}$$

We can apply the above idea to develop a clustering algorithm. In practice, we cannot hope for the connected components of the similarity graph to coincide with the clusters. Instead, we're more likely to have a situation like

$$\begin{aligned} \boldsymbol{L} &= \begin{bmatrix} \boldsymbol{L}_1 & & & \\ & \boldsymbol{L}_2 & & \\ & & \ddots & \\ & & & \boldsymbol{L}_k \end{bmatrix} + \begin{bmatrix} & & & \\ & \text{noise} & & \\ & & & \end{bmatrix} \\ &= \quad \boldsymbol{L}_{\text{ideal}} \quad + \quad \Delta \end{aligned}$$

where Δ accounts for edges between nodes in different clusters (we focus here on $\boldsymbol{L}$ but similar reasoning applies to $\tilde{\boldsymbol{L}}$). If we have constructed a decent similarity graph, the entries of Δ should be much smaller than

the entries of $\boldsymbol{L}_1, \boldsymbol{L}_2, ..., \boldsymbol{L}_k$, and can be thought of as "noise." We can then appeal to matrix perturbation theory, which says that the k smallest eigenvalues of $\boldsymbol{L}$ should still be close to zero, and their corresponding eigenvectors should still roughly span the nullspace of $\boldsymbol{L}_{\text{ideal}}$. In other words, to recover the clusters from the similarity graph, we should compute a basis of the nullspace of $\boldsymbol{L}_{\text{ideal}}$. Since we don't have access to $\boldsymbol{L}_{\text{ideal}}$, we will approximate its nullspace using the k smallest eigenvectors of $\boldsymbol{L}$. Then the vectors $\boldsymbol{y}_i$ should be approximately equal for data points in the same cluster. Since they won't be exactly equal because of noise, we can group them using k-means. This procedure is precisely spectral clustering.

7 Spectral Clustering: NLDR Perspective

Spectral clustering can be viewed as first performing a type of nonlinear dimensionality reduction (NLDR), followed by applying a simple clustering algorithm, such as k-means, to the resulting low-dimensional points.

For now let us temporarily forget about spectral clustering and consider an approach to NLDR based on graph Laplacians. Let $\boldsymbol{W}$ be a weighted similarity graph based on $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$. Let $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \in \mathbb{R}^p$ be the reduced dimensionality versions to be learned. Viewing the $\boldsymbol{y}_i$ as column vectors now, consider choosing $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ to minimize

$$\frac{1}{2} \sum_{i,j} w_{i,j} \left\| \boldsymbol{y}_i - \boldsymbol{y}_j \right\|^2.$$

This objective captures the intuition that, if $w_{i,j}$ is large, meaning $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ have high similarity, then the reduced versions $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ should be closer. Using Proposition 1 it can be shown (as an exercise) that

$$\frac{1}{2} \sum_{i,j} w_{i,j} \left\| \boldsymbol{y}_i - \boldsymbol{y}_j \right\|^2 = tr(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T),$$

where $\boldsymbol{Y} = [\boldsymbol{y}_1 \ \cdots \ \boldsymbol{y}_n] \in \mathbb{R}^{p \times n}$, and $\boldsymbol{L}$ is the unnormalized graph Laplacian.

To obtain a unique solution, an energy constraint is added (otherwise setting $\boldsymbol{y}_i$ equal to a constant $\forall i$ is optimal). There are two common energy constraints. The first is

$$\boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{I}.$$

The solution to

$$\min_{\boldsymbol{Y}} \operatorname{tr}(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T) \text{ s.t. } \boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{I}$$

is

$$\boldsymbol{Y}^T = [\boldsymbol{u}_1 \cdots \boldsymbol{u}_p]$$

where $\boldsymbol{L} = \boldsymbol{U}\Lambda\boldsymbol{U}^T$ is the spectral decomposition of $\boldsymbol{L}$, and $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_p$ are the columns of $\boldsymbol{U}$ associated to the p smallest eigenvalues. This follows from the generalized Rayleigh quotient result from the PCA notes.

The second energy constraint is

$$\boldsymbol{Y}\boldsymbol{D}\boldsymbol{Y}^T = \boldsymbol{I}.$$

The solution to

$$\min_{\boldsymbol{Y}} \operatorname{tr}(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T) \text{ s.t. } \boldsymbol{Y}\boldsymbol{D}\boldsymbol{Y}^T = \boldsymbol{I}$$

is

$$\boldsymbol{Y}^T = [\tilde{\boldsymbol{u}}_1 \ \cdots \ \tilde{\boldsymbol{u}}_p],$$

the smallest p eigenvectors of $\tilde{\boldsymbol{L}} = \boldsymbol{D}^{-1}\boldsymbol{L}$, the normalized graph Laplacian. This can be shown by substituting $\boldsymbol{Z} = \boldsymbol{Y}\boldsymbol{D}^{\frac{1}{2}}$, and reducing the problem to the generalized Rayleigh quotient (exercise).

Figure 3 shows the embedding for the rings data shown earlier. In the figure, you will notice that one axis has a very small scale. Along this axis, all $\boldsymbol{y}_i$ have the same coordinate up to numerical precision of the eigenvalue solver. This is because the smallest eigenvalue of both $\boldsymbol{L}$ and $\tilde{\boldsymbol{L}}$ is 0, and the ones vector $\mathbf{1}$

is a corresponding eigenvalue. Furthermore, if the graph is connected, it can be shown that multiples of $\mathbf{1}$ are the only eigenvectors for the eigenvalue zero. Therefore, the first coordinate of all $\boldsymbol{y}_i$ will have the same value. (The numerical value in the figure is not necessarily meaningful because the Matlab eigenvalue solver that produced it does not necessarily return unit-norm eigenvectors.)

Since the first coordinate is predictable, the optimization problems can be reformulated to omit this coordinate and solve only for the $\boldsymbol{Y} \in \mathbb{R}^{(p-1)\times n}$ minimizing $\mathrm{tr}(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T)$, subject to both the energy constraint and an orthogonality constraint. In particular, for the energy constraint $\boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{I}$, the optimization problem becomes

$$\begin{aligned} \min_{\boldsymbol{Y}} \ & \mathrm{tr}(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T) \\ \text{s.t. } & \boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{I} \\ & \boldsymbol{Y}\mathbf{1} = \mathbf{0}. \end{aligned}$$

For the energy constraint $\boldsymbol{Y}\boldsymbol{D}\boldsymbol{Y}^T = \boldsymbol{I}$, the optimization problem becomes

$$\begin{aligned} \min_{\boldsymbol{Y}} \ & \mathrm{tr}(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T) \\ \text{s.t. } & \boldsymbol{Y}\boldsymbol{D}\boldsymbol{Y}^T = \boldsymbol{I} \\ & \boldsymbol{Y}\boldsymbol{D}\mathbf{1} = \mathbf{0}, \end{aligned}$$

where the second constraint is equivalent to requiring orthogonality to the smallest eigenvector in the transformed variable $\boldsymbol{Z}$ (exercise). These two formulations are equivalent to the original ones wherein the first coordinate is discarded. Because of this phenomenon, it is common to choose $p-1$ to be the desired embedding dimension.

The version with the $\boldsymbol{Y}\boldsymbol{D}\boldsymbol{Y}^T = \boldsymbol{I}$ energy constraint is known as Laplacian Eigenmaps. It was introduced in Belkin and Nyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation", *Neural Computation*, vol. 15, no. 6, 1373-1396 (2003). Figure 6 is taken from this paper and shows a $p-1=2$ dimensional embedding where the orthogonality constraint is employed.

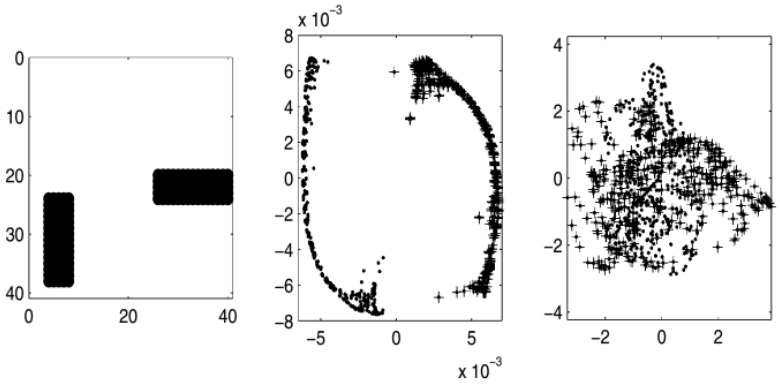

Figure 3: (Left) A horizontal and a vertical bar. (Middle) A two-dimensional representation of the set of all images using the Laplacian eigenmaps. (Right) The result of PCA using the first two principal directions to represent the data. Blue dots correspond to images of vertical bars, and plus signs correspond to images of horizontal bars.

Figure 6: Figure from Belkin and Nyogi.

Spectral clustering can be viewed as first performing dimensionality reduction to $p-1 = k-1$ dimensions and then clustering the resulting lower dimensional representation with k-means. Even though k means would not be a good choice to directly cluster the original data, after nonlinear dimensionality reduction the clusters are (hopefully) better separated making k-means appropriate.

Remark 2. In the NLDR derivation, there is no particular reason why p should be chosen to equal k. One could embed into any dimension and then perform k-means.

8 Closing thoughts

This section highlights some additional points of interest. For a very thorough tutorial on spectral clustering see Ulrike von Luxburg, "A Tutorial on Spectral Clustering", 2007.

8.1 Model selection

Figure 7 shows a simple one-dimensional datasets with four clusters. If there is a jump in the sorted eigenvalues, from "near zero" to "not near zero", that probably indicates the number of clusters. This is based on the perturbation perspective in Section 6.

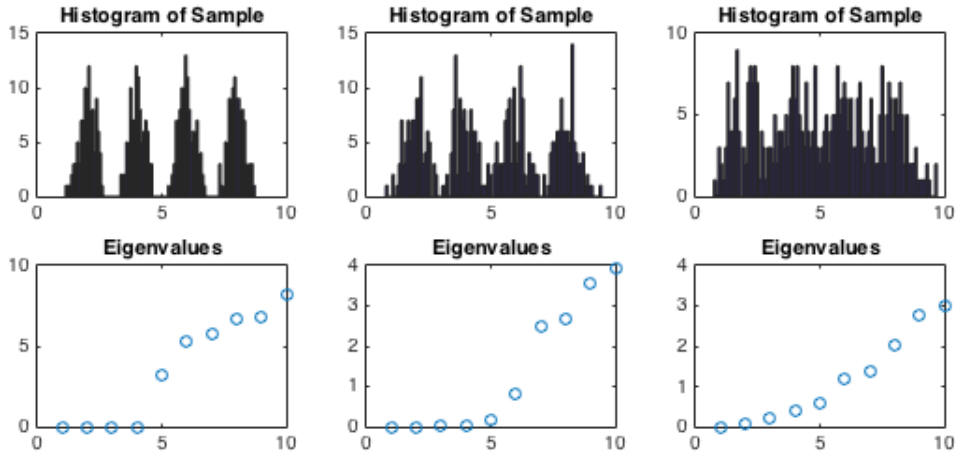

Figure 7: Data histograms (top row) and Laplacian eigenvalues (bottom row) for three different datasets. In the top row, the variance of the datasets increase from left to right, so the clusters become more overlapped. In the bottom row, the jump from zero to nonzero eigenvalues becomes less obvious from left to right.

8.2 Another normalized graph Laplacian

There is yet a third graph Laplacian defined as

$$\widetilde{\boldsymbol{L}} = \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{\frac{1}{2}}$$

It has properties similar to $\boldsymbol{L}$ and $\widetilde{\boldsymbol{L}}$, but the spectral clustering algorithm needs to be tweaked.

8.3 Semi-Supervised Learning

Graph Laplacians also come up in *semi-supervised learning*. Consider a regression problem where you have both

- labeled data $\Big(\boldsymbol{x}_i, y_i\Big)_{i=1}^{m}$
- unlabeled data $\Big(\boldsymbol{x}_i\Big)_{i=m+1}^{m+n}$

The unlabeled data can potentially help to improve a regression estimate. If f denotes a regression function and we solve

$$\min_f \ \frac{1}{2}\sum_{i=1}^{m}\Big(y_i - f(\boldsymbol{x}_i)\Big)^2 + \underbrace{\frac{\lambda}{2}\sum_{i,j=m+1}^{m+n} w_{ij}\Big(f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\Big)^2}_{=\lambda \boldsymbol{f}_n^T \boldsymbol{L} \boldsymbol{f}_n, \quad \boldsymbol{f}_n = [f(\boldsymbol{x}_{m+1}),\ldots,f(\boldsymbol{x}_{m+n})]^T}$$

where w_{ij} come from a similarity graph defined on the unlabeled data, then the second term regularizes the solution to be more smooth. This improves the interpolation between the labeled data points. Here it should be understood that f belongs to some prescribed family of functions, such as linear functions.

Exercises

1. Several steps in the notes are left as an exercise. Complete these steps.
2. (★★) Assuming $K = 2$, show that a relaxation of the Ncut problem is solved by normalized spectral clustering, i.e., spectral clustering with the normalized graph Laplacian $\tilde{\boldsymbol{L}} = \boldsymbol{D}^{-1}L$. *Hint:* First, define $\boldsymbol{f}_A$ in an analogous way to the treatment of RatioCut. Verify analogous formulas for $\boldsymbol{f}_A^T \boldsymbol{L} \boldsymbol{f}_A$, $\boldsymbol{1}^T \boldsymbol{D} \boldsymbol{f}_A$, and $\boldsymbol{f}_A^T \boldsymbol{D} \boldsymbol{f}_A$ to formulate the relaxation. Then make the substitution $\boldsymbol{g} = \boldsymbol{D}^{1/2}\boldsymbol{f}$ and reformulate the relaxation with g as the variable. Once you solve for $\boldsymbol{g}$, don't forget to transform back to $\boldsymbol{f}$.

References

[1] Hagen, Lars, and Andrew B. Kahng. "New spectral methods for ratio cut partitioning and clustering." IEEE transactions on computer-aided design of integrated circuits and systems 11.9 (1992): 1074-1085.

[2] Shi, Jianbo, and Jitendra Malik. "Normalized cuts and image segmentation." IEEE Transactions on pattern analysis and machine intelligence 22.8 (2000): 888-905.