CS168: The Modern Algorithmic Toolbox
Lectures #11: Spectral Graph Theory, I

Tim Roughgarden & Gregory Valiant*

May 6, 2019

Spectral graph theory is the powerful and beautiful theory that arises from the following question:

> What properties of a graph are exposed/revealed if we 1) represent the graph as a matrix, and 2) study the eigenvectors/eigenvalues of that matrix.

Most of our lectures thus far have been motivated by concrete problems, and the problems themselves led us to the algorithms and more widely applicable techniques and insights that we then went on to discuss. This section has the opposite structure: we will begin with the very basic observation that graphs can be represented as matrices, and then ask "what happens if we apply the linear algebraic tools to these matrices"? That is, we will begin with a technique, and in the process of gaining an understanding for what the technique does, we will arrive at several extremely natural problems that can be approached via this technique.

Throughout these lecture notes we will consider undirected, and unweighted graphs (i.e. all edges have weight 1), that do not have any self-loops. Most of the definitions and techniques will extend to both directed graphs, as well as weighted graphs, though we will not discuss these extensions here.

1 Graphs as Matrices

Given a graph, $G = (V, E)$ with $|V| = n$ vertices, there are a number of matrices that we can associate to the graph, including the adjacency matrix. As we will see, one extremely natural matrix is the *Laplacian* of the graph:

Definition 1.1 Given a graph $G = (V, E)$, with $|V| = n$, the *Laplacian* matrix associated to G is an $n \times n$ matrix $L_G = D - A$, where D is the degree matrix—a diagonal matrix where

$D(i,i)$ is the degree of the ith node in G, and A is the adjacency matrix, with $A(i,j) = 1$ if and only if $(i,j) \in E$. Namely,

$$L_G(i,j) = \begin{cases} deg(i) & \text{if } i = j \\ -1 & \text{if } (i,j) \in E \equiv 1 \\ 0 & \text{otherwise.} \end{cases}$$

For ease of notation, in settings where the underlying graph is clear, we will omit the subscript, and simply refer to the Laplacian as L rather than L_G.

To get a better sense of the Laplacian, let us consider what happens to a vector when we multiply it by L: if $Lv = w$, then

$$w(i) = deg(i)v(i) - \sum_{j:(i,j)\in E} v_j = \sum_{j:(i,j)\in E} (v(i) - v(j)) .$$

That is the ith element of the product Lv is the sum of the differences between $v(i)$ and the indices of v corresponding to the neighbors of i in the graph G.

Building off this calculation, we will now interpret the quantity $v^t Lv$, which will prove to be the main intuition that we come back to when reasoning about the eigenvalues/eigenvectors of L:

$$\begin{aligned} v^t Lv &= \sum_i v(i) \sum_{j:(i,j)\in E} (v(i) - v(j)) \\ &= \sum_{(i,j)\in E} v(i)\,(v(i) - v(j)) \\ &= \sum_{i<j:(i,j)\in E} v(i)\,(v(i) - v(j)) + v(j)\,(v(j) - v(i)) \\ &= \sum_{i<j:(i,j)\in E} (v(i) - v(j))^2 . \end{aligned}$$

In other words, if one interprets the vector v as assigning a number to each vertex in the graph G, the quantity $v^t Lv$ is exactly the sum of the squares of the differences between the values of neighboring nodes. Phrased alternately, if one were to place the vertices of the graph on the real numberline, with the ith node placed at location $v(i)$, then $v^t Lv$ is precisely the sum of the squares of the lengths of all the edges of the graph.

2 The Eigenvalues and Eigenvectors of the Laplacian

What can we say about the eigenvalues of the Laplacian, $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$? Since L is a real-valued symmetric matrix, all of its eigenvalues are real numbers, and its eigenvectors are orthogonal to eachother. The calculation above showed that for any vector v, $v^t Lv = \sum_{i<j:(i,j)\in E} (v(i) - v(j))^2$. Because this expression is the sum of squares, we conclude that $v^t Lv \geq 0$, and hence all the eigenvalues of the Laplacian must be nonnegative.

The eigenvalues of L contain information about the structure (or, more precisely, information about the extent to which the graph has structure). We will see several illustrations of this claim; this is a very hot research area, and there continues to be research uncovering new insights into the types of structural information contained in the eigenvalues of the Laplacian.

2.1 The zero eigenvalue

The laplacian always has at least one eigenvalue that is 0. To see this, consider the vector $v = (1/\sqrt{n}, \ldots, 1/\sqrt{n})$, and recall that the ith entry of Lv is $\sum_{j:(i,j)\in E} v(i) - v(j) = \sum(1/\sqrt{n} - 1/\sqrt{n}) = 0 = 0 \cdot v(i)$. The following theorem shows that the multiplicity of the zeroth eigenvalue reveals the number of connected components of the graph.

Theorem 2.1 *The number of zero eigenvalues of the Laplacian L_G (i.e. the multiplicity of the 0 eigenvalue) equals the number of connected components of the graph G.*

Proof: We first show that the number of zero eigenvalues is at least the number of connected components of G. Indeed, assume that G has k connected components, corresponding to the partition of V into disjoint sets $S_1, \ldots, S_k$. Define k vectors $v_1, \ldots, v_k$ s.t. $v_i(\mathrm{j}) = 1/\sqrt{|S_i|}$ if $j \in S_i$, and 0 otherwise. For $i = 1, \ldots, k$, it holds that $||v_i|| = 1$. Additionally, for $i \neq j$, because the sets S_i, S_j are disjoint, $\langle v_i, v_j \rangle = 0$. Finally, note that $Lv_i = 0$. Hence there is a set of k orthonormal vectors that are all eigenvectors of L, with eigenvalue 0.

To see that the number of 0 eigenvalues is at most the number of connected components of G, note that since $v^t L v = \sum_{i<j,(i,j)\in E}(v(i) - v(j))^2$, this expression can only be zero if v is constant on every connected component. To see that there is no way of finding a $k+1st$ vector v that is a zero eigenvector, orthogonal to $v_1, \ldots, v_k$ observe that any eigenvector, v must be nonzero in some coordinate, hence assume that v is nonzero on a coordinate in set S_i, and hence is nonzero and constant on all indices in set S_i, in which case v can not be orthogonal to v_i., and there can be no $k+1st$ eigenvector with eigenvalue 0. ■

2.2 Intuition of lowest and highest eigenvalues/eigenvectors

To think about the smallest eigenvalues and their associated eigenvectors, it is helpful to come back to the fact that $v^t L v = \sum_{(i,j)\in E, i<j}(v(i) - v(j))^2$ is the sum of the squares of the distances between neighbors. Hence the eigenvectors corresponding to the lowest eigenvalues correspond to vectors for which neighbors have similar values. Eigenvectors with eigenvalue 0 are constant on each connected component, and the smallest nonzero eigenvector will be the vector that minimizes this squared distance between neighbors, subject to being a unit vector that is orthogonal to all zero eigenvectors. That is, the eigenvector corresponding to the smallest *non-zero* eigenvalue will be minimizing this sum of squared distances, subject to the constraint that the values are somewhat spread out on each connected component.

Analogous reasoning applies to the maximum eigenvalue. The maximum eigenvalue $\max_{||v||=1} v^t L v$ will try to maximize the discrepancy between neighbors' values of v.

To get a more concrete intuition for what the small/large eigenvalues and their associated eigenvectors look like, you should try to work out (either in your head, or using a computer) the eigenvectors/values for lots of common graphs (e.g. the cycle, the path, the binary tree, the complete graph, random graphs, etc.) See the example/figure below [After the assignment is due, I will add the examples from the pset to these lecture notes.]

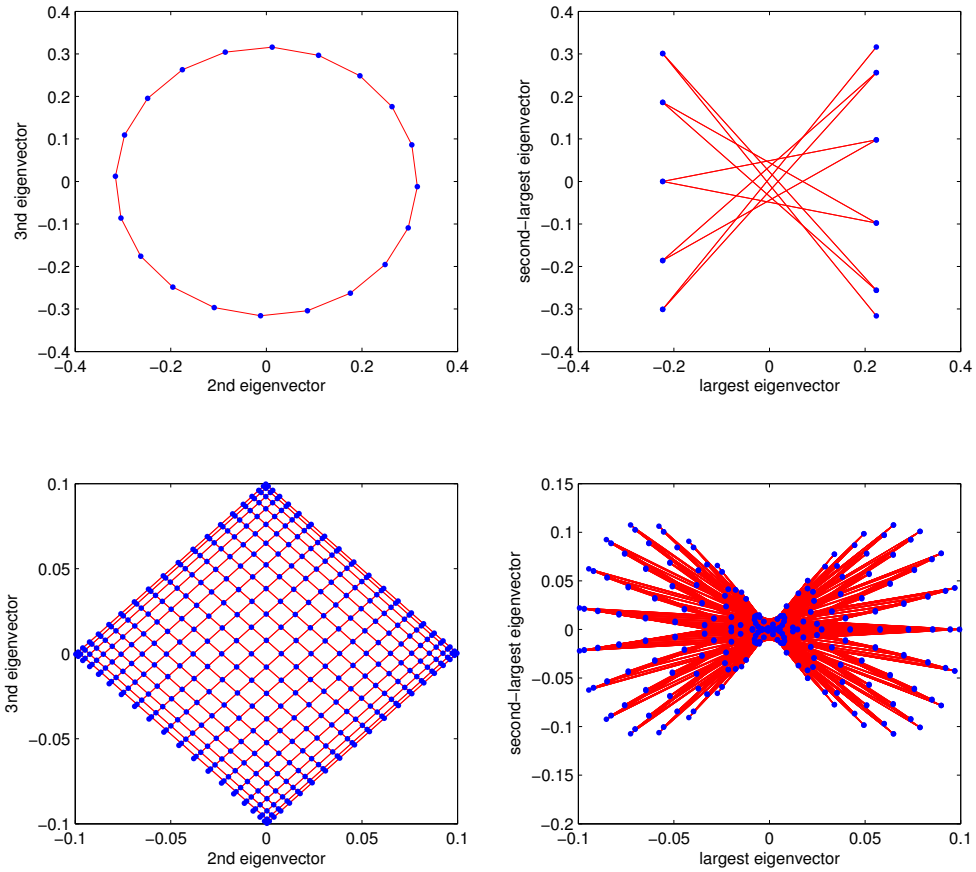

Figure 1: Spectral embeddings of the cycle on 20 nodes (top two figures), and the 20×20 grid (bottom two figures). Each figure depicts the embedding, with the red lines connecting points that are neighbors in the graph. The left two plots show the embeddings onto the eigenvectors corresponding to the second and third smallest eigenvalues; namely, the ith node is plotted at the point $(v_2(i), v_3(i))$ where v_k is the eigenvector corresponding to the kth largest eigenvalue. The right two plots show the embeddings onto the eigenvectors corresponding to the largest and second–largest eigenvalues. Note that in the left plots, neighbors in the graph are close to each other. In the right plots, most points end up far from all their neighbors.

3 Applications of Spectral Graph Theory

We briefly describe several problems for which considering the eigenvalues/eigenvectors of a graph Laplacian proves useful.

3.1 Visualizing a graph: Spectral Embeddings

Suppose one is given a list of edges for some graph. What is the right way of visualizing, or drawing the graph so as to reveal its structure? Ideally, one might hope to find an embedding of the graph into $\mathbb{R}^d$, for some small $d = 2, 3, \ldots$. That is, we might hope to associate some point, say in 2 dimension, to each vertex of the graph. Ideally, we would find an embedding that largely respects the structure of the graph. What does this mean? One interpretation is simply that we hope that most vertices end up being close to most of their neighbors.

Given this interpretation, embedding a graph onto the eigenvectors corresponding to the small eigenvalues seems extremely natural. Recall that the small eigenvalues correspond to unit vectors v that try to minimize the quantity $v^t L v = \frac{1}{2}\sum_{(i,j)\in E}(v(i) - v(j))$, namely, these are the vectors that are trying to map neighbors to similar values. Of course, if the graph has a single connected component, the smallest eigenvector $v_1 = (1/\sqrt{n}, \ldots, 1/\sqrt{n})$, which is not helpful for embedding, as all points have the same value. In this case, we should start by considering v_2, then v_3,et.

While there are many instances of graphs for which the spectral embedding does not make sense, it is a very natural first thing to try. As Figure 1 illustrates, in some cases the embedding onto the smallest two eigenvectors really does correspond to the "natural"/"intuitive" way to represent the graph in 2-dimensions.

3.2 Spectral Clustering/Partitioning

How can one find large, insular clusters in large graphs? How can we partition the graph into several large components so as to minimize the number of edges that cross between the components? These questions questions arise naturally in the study of any large networks (social networks, protein interaction networks, semantic/linguistic graphs of word usage, etc.)

In the next lecture, we will discuss some specific metrics for the quality of a given cluster or partition of a graph, and give some quantitative bounds on these metrics in terms of the second eigenvalue of the graph Laplacian.

For the time being, just understand the intuition that the eigenvectors corresponding to small eigenvalues are, in some sense, trying to find good partitions of the graph. These low eigenvectors are trying to find ways of assigning different numbers to vertices, such that neighbors have similar values. Additionally, since they are all orthogonal, each eigenvectors is trying to find a "different"/"new" such partition. This is the intuition why it is often a good idea to look at the clusters/partitions suggested by a few different small eigenvectors. [This point is especially clear on the next homework.]

3.3 Graph Coloring

Consider the motivating example of allocation one of k different radio bandwidths to each radio station. If two radio stations have overlapping regions of broadcast, then they cannot be assigned the same bandwidth (otherwise there will be interference for some listeners). On the other hand, if two stations are very far apart, they can broadcast on the same bandwidth without worrying about interference from each other. This problem can be modeled as the problem of k-coloring a graph. In this case, the nodes of the graph will represent radio stations, and there will be an edge between two stations if their broadcast ranges overlap. The problem then, is to assign one of k colors (i.e. one of k broadcast frequencies) to each vertex, such that no two adjacent vertices have the same color. This problem arises in several other contexts, including various scheduling tasks (in which, for example, tasks are mapped to processes, and edges connect tasks that cannot be performed simultaneously).

This problem of finding a k-color, or even deciding whether a k-coloring of a graph exists, is NP-hard in general. One natural heuristic, which is motivated by the right column of Figure 1, is to embed the graph onto the eigenvectors corresponding to the *highest* eigenvalues. Recall that these eigenvectors are trying to make vertices as different from their neighbors as possible. As one would expect, in these embeddings, points that are close together in the embedding tend to *not* be neighbors in the original graph. Hence one might imagine a k-coloring heuristic as follows: 1) plot the embedding of the graph onto the top 2 or 3 eigenvectors of the Laplacian, and 2) locally partition the points in this space into k regions, e.g. using k-means, or a kd-tree, and 3) assign all the points in each region the same color. Since neighbors in the graph will tend to be far apart in the embeddings, this will tend to give a decent coloring of the graph.