# Spectral Graph Theory and Applications

**CS 270 Project Report**

Albert Zheng and Cory Cheung

May 2017

## 1 Introduction

In class, we saw several ways in which the *spectral gap* of a graph can be used to study $k$-regular graphs. The goal of our project was to understand further extensions and applications of these results. To this end, we read [2] "Spectral Theory of Unsigned and Signed Graphs Applications to Graph Clustering: a Survey," a 2016 survey paper by Jean Gallier explaining the mathematics behind the spectral graph drawing and spectral graph clustering algorithms. We also experimented with both the graph drawing and graph clustering algorithms ourselves. This paper contains a summary of our findings.

In Section 2 of this report, we establish the definitions that we will work with throughout the rest of the paper. These definitions allow us to work study weighted, nonregular graphs. In Section 3, we prove basic properties of the spectral gap in this more general setting. Next, in Section 4 we describe the *graph drawing problem* and prove that it is solved cleanly by the eigenvectors of the Laplacian matrix. We implemented the spectral graph drawing algorithm in Sage and produced a variety of examples. These examples are presented in Section 5. Finally, in Section 6 we describe the *graph clustering problem* and compare the spectral algorithm with several other image segmentation techniques.

## 2 Definitions

### 2.1 Unweighted Graphs

Throughout this paper, we will get $G = (V, E)$ refer to a graph on $n$ vertices. We will also write $i \sim j$ to denote that there is an edge between vertices $v_i$ and $v_j$. First, we recall the definitions of the adjacency and degree matrices of $G$ and use this to define the *Laplacian* matrix of $G$.

**Definition 2.1.** *The* ***adjacency matrix*** *of $G$ is the $n \times n$ matrix $A = (a_{ij})$, where $a_{ij}$ is 1 if there is an edge between vertex $i$ and vertex $j$ and 0 otherwise.*

**Definition 2.2.** *The **degree matrix**, $D$, is the $n \times n$ diagonal matrix where $d_{ii}$ is equal to the degree of vertex $i$. The **Laplacian matrix** is then defined as*

$$L := D - A$$

Since $A$ and $L$ are real, symmetric matrices, the spectral theorem implies that both have $n$ real eigenvalues (up to multiplicity) and an orthonormal basis of corresponding eigenvectors. These eigenvalues and the corresponding eigenvectors will be our main object of study. We will let

$$\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$$

denote the eigenvalues of $L$ and

$$\mu_1, \mu_2, \ldots, \mu_n$$

the corresponding eigenvectors.

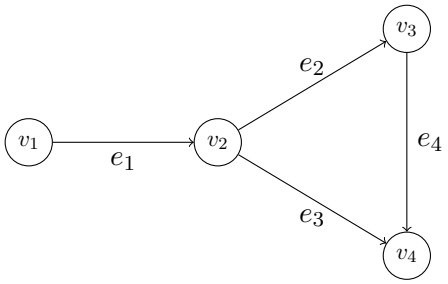
**Remark 2.3.** *In class, we studied $k$-regular graphs through the eigenvalues of $A$. In the case of $k$-regular graphs, we have $D = kI$, so $L = kI - A$. This means it is easy to go from eigenvalues of $A$ to eigenvalues of $L$ and vice versa, since the eigenvalues of $L$ are exactly $k$ minus the eigenvalues of $A$. Hence the study of the eigenvalues of $L$ in general graphs generalizes the study of eigenvalues of $A$ in $k$-regular graphs.*

We introduce one more matrix, which helps us understand the Laplacian.

**Definition 2.4.** *Given an orientation to the edges of $G$ (so each edge $(v_1, v_2)$ points from one vertex to another), the (oriented) **incidence matrix** of $G$, is a matrix $U$ with rows indexed by vertices and columns indexed by edges. The entries $u_{ie}$ corresponding to vertex $i$ and edge $e = (v_1, v_2)$ are defined by*

$$u_{ie} = \begin{cases} 1 & i = v_1 \\ -1 & i = v_2 \\ 0 & i \neq v_1, v_2 \end{cases}$$

**Example 2.5.** *$A$, $D$, $L$, and $U$ (for the depicted orientation) for a simple graph are given below.*




$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

$$L = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix} \quad U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & -1 \end{bmatrix}$$

*The eigenvalues of $L$ are $(\lambda_1, \lambda_2, \lambda_3, \lambda_4) = (0, 1, 3, 4)$.*

The incidence matrix is important to the study of the Laplacian mainly because of the following result.

**Proposition 2.6.** *For any orientation of $G$,*

$$L = UU^\top.$$

*Proof.* Let $U_i$ refer to the $i$th row of $U$. Note that the entries of $U_i$ are 1 or $-1$ at the indices where $v_i$ is adjacent to another vertex and 0 elsewhere. Hence $U_i U_i^\top$ is equal to the number of vertices adjacent to $v_i$, i.e. the degree of the vertex. Now consider $U_i U_j^\top$ for $i \neq j$. Recall that each column corresponds to an edge $e$. The only way that the entries in $U_i$ and $U_j$ at column $e$ are both nonzero is if $e = (v_i, v_j)$ or $e = (v_j, v_i)$. Then one of the entries of 1 and the other if $-1$, so $U_i U_j^\top = -1$ if there is an edge between $i$ and $j$ and 0 otherwise.

So the diagonal entries of $UU^\top$ are the degrees of the vertices and the off-diagonal entries are $-1$ if there is an edge and 0 otherwise. Hence $UU^\top = D - A = L$. □

As a corollary, we see that $L$ is positive semidefinite, so $\lambda_1$ and all greater eigenvalues are nonnegative. In fact, $\lambda_1 = 0$ since the $n \times 1$ vector consisting of all ones is always in the kernel of $L$. We will let $\mathbf{1}$ denote this vector, so $\mu_1 = \mathbf{1}$.

## 2.2 Weighted Graphs

The preceding definitions generalize easily to the study of graphs with edge weights. The Laplacian is defined similarly, with the adjacency matrix replaced by weight matrix.

**Definition 2.7.** *The **weight matrix** of $G$ is the $n \times n$ matrix $W = (w_{ij})$, where $w_{ij}$ is the weight of the edge between vertices $v_i$ and $v_j$ (or 0 if there is no edge).*

**Definition 2.8.** *The **degree matrix**, $D$, is the $n \times n$ diagonal matrix where $d_{ii}$ is equal to the sum of the weights of all edges adjacent to $v_i$. **Laplacian matrix** is then defined as*

$$L := D - W$$

As before, we let $\lambda_1 \leq \ldots \leq \lambda_n$ and $\mu_1, \ldots, \mu_n$ denote the eigenvalues and eigenvectors of $L$, respectively. In fact, it still holds that $lambda_1 = 0$ and $\mu_1 = \mathbf{1}$. We show $L$ is still positive semidefinite by revising the definition of the incidence matrix.

**Definition 2.9.** *The (oriented)* ***incidence matrix*** *of $G$, is a matrix $U$ with rows indexed by vertices and columns indexed by edges. The entries $u_{ie}$ corresponding to vertex $i$ and edge $e = (v_1, v_2)$ are defined by*

$$u_{ie} = \begin{cases} \sqrt{w_{v_1 v_2}} & i = v_1 \\ -\sqrt{w_{v_1 v_2}} & i = v_2 \\ 0 & i \neq v_1, v_2 \end{cases}$$

Then the same argument as before shows that $L = UU^\top$.

# 3 The Spectral Gap

## 3.1 Definition

Recall that in class we defined the spectral gap to be the difference between the two largest eigenvalues of $A$. When working with the eigenvalues of the Laplcian, this becomes the difference between the two smallest eigenvalues of $L$. Since $\lambda_1 = 0$, the spectral gap is just $\lambda_2$.

**Definition 3.1.** *$\lambda_2$ is the* ***spectral gap*** *(or* ***Fiedler value*** *or* ***algebraic connectivity****) of the graph $G$.*

Below are examples of $\lambda_2$ values for some families of graphs. The intuition is that $\lambda_2$ measures connectivity of the graph.

**Example 3.2.** *[3] Let $C_n$ be the graph consisting of a size $n$ cycle. As usual, $K_n$ is the complete graph on $n$ vertices and $K_{m,n}$ is the complete bipartite graph on $m$ and $n$ vertices. Then*

$$\lambda_2(C_n) = 2\left(1 - \cos(\frac{2\pi}{n})\right)$$
$$\lambda_2(K_n) = n$$
$$\lambda_2(K_{m,n}) = \min(m, n).$$

**Example 3.3.** *[4] For a tree $T$, $a(T) \leq 1$ with equality exactly when $T$ is the star graph $S_n$, the tree consisting of one root vertex and $n-1$ leaves. The tree $P_n$, the graph consisting of a line of $n$ vertices, has*

$$\lambda_2(P_n) = 2\left(1 - \cos(\frac{\pi}{n})\right)$$

The highest $\lambda_2$ value out of these examples comes from $K_n$, the densest graph on $n$ vertices. In contrast, trees (which are the sparsest connected graphs on $n$ vertices) have $\lambda_2 \leq 1$. The star graph $S_n$ is the most well-connected tree in the sense that the path length between any two vertices is at most 2. Indeed, $S_n$ has the highest $\lambda_2$ value for trees. Meanwhile, the path graph $P_n$ requires a path length of up to $n$. The given formula shows that $\lambda_2(P_n)$ goes to 0 as $n$ goes to infinity. The cycle graph has one more edge than $P_n$; likewise, its $\lambda_2$ value is slightly higher than that of $C_n$.

## 3.2 Rayleigh Quotients

Before proving results about the spectral gap, we need some tools from linear algebra.

**Definition 3.4.** *Given an $n \times n$ real symmetric matrix $M$ and an $n \times 1$ vector $x$, the **Rayleigh quotient** is defined to be*

$$R(M, x) := \frac{x^\top M x}{x^\top x}$$

The importance of Rayleigh quotients is that they are minimized by eigenvectors. Specifically, we have

$$\begin{aligned}
\lambda_1 &= \min \frac{x^\top M x}{x^\top x} \\
\lambda_2 &= \min_{x \perp \mathbf{1}} \frac{x^\top M x}{x^\top x} \\
\lambda_3 &= \min_{x \perp \mathbf{1}, \mu_2} \frac{x^\top M x}{x^\top x} \\
&\vdots \\
\lambda_i &= \min_{x \perp \mathbf{1}, \ldots, \mu_{i-1}} \frac{x^\top M x}{x^\top x} \\
&\vdots
\end{aligned}$$

Note that since scaling $x$ does not change $R(M, x)$, these expressions can also be written

$$\begin{aligned}
\lambda_1 &= \min_{|x|=1} x^\top L x \\
\lambda_2 &= \min_{\substack{|x|=1, \\ x \perp \mathbf{1}}} x^\top L x \\
\lambda_3 &= \min_{\substack{|x|=1, \\ x \perp \mathbf{1}, \mu_2}} x^\top L x \\
&\vdots
\end{aligned}$$

By considering Rayleigh quotients for $-M$, we can see that similar expressions also hold for the largest eigenvalues of $M$. We have

$$\begin{aligned}
\lambda_n &= \max \frac{x^\top M x}{x^\top x} \\
\lambda_{n-1} &= \max_{x \perp \mu_n} \frac{x^\top M x}{x^\top x} \\
&\vdots \\
\lambda_{n-i} &= \max_{x \perp \mu_n, \ldots, \mu_{n-i+1}} \frac{x^\top M x}{x^\top x} \\
&\vdots
\end{aligned}$$

All three of the above characterizations of $\lambda_i$ will be important to understanding the Laplcian. The following proposition shows that Rayleigh quotients play nicely with the Laplacian matrix.

**Proposition 3.5.** *For any $n \times 1$ vector $x$,*

$$x^\top Lx = \sum_{i \sim j} w_{ij}(x_i - x_j)^2.$$

*Proof.* Recall that $L = UU^\top$. Hence

$$\begin{aligned} x^\top Lx &= x^\top UU^\top x \\ &= |U^\top x|^2 \\ &= \sum_{(i,j) \in E} w_{ij}(v_i - v_j)^2. \end{aligned}$$

□

## 3.3 Basic Properties of $\lambda_2$

We now formalize the intuition that $\lambda_2$ measures connectivity. The following result relates the connected components of $G$ to the eigenvalues.

**Proposition 3.6.** *The multiplicity of 0 as an eigenvalue of $L$ is equal to the number of connected components of $L$. In particular, $\lambda_2 > 0$ if and only if $G$ is connected.*

*Proof.* Let the connected components of $G$ be $C_1, \ldots, C_k$. For $i = 1, \ldots, k$ we define the $n \times 1$ vector $x^{(i)}$ by $x^{(i)}_j = 1$ if $v_j \in C_i$ and $x^{(i)}_j = 0$ otherwise. We show that $x^{(1)}, \ldots, x^{(k)}$ form a basis for the kernel of $L$.

Clearly, the $x^{(i)}$ are linearly independent, so we want to show they span the kernel of $L$. Suppose $Lx = 0$. Then we also have $x^\top Lx = x^\top UU^\top x = |U^\top x| = 0$. The entries of $U^\top x$ are exactly $x_i - x_j$ for each edge $i \sim j$. Hence any two vertices $v_i$ and $v_j$ connected by an edge, we have $x_i = x_j$. It follows that $x_i = x_j$ if $i$ and $j$ belong to the same connected component, so $x$ is in the span of the $x^{(i)}$.

Alternatively, begin by noting that we have $Lx = 0$ if and only if $U^\top x = 0$, so $\ker L = \operatorname{coker} U$. Next, note that $G$ can be seen as a one dimensional combinatorial simplicial complex. In this context, the incidence matrix $U$ is exactly the boundary map from the space of edges to the space of vertices. Then $\operatorname{coker} U$ is exactly the 0-th homology group, so the dimension is equal to the number of connected components. □

The next results show that adding edges increases $\lambda_2$. These results follow easily from the Rayleigh quotient characterization of $\lambda_2$.

**Proposition 3.7.** *Let $e$ be an edge in $G$. Then $\lambda_2(G \setminus e) \leq \lambda_2(G)$.*

*Proof.* Since the Rayleigh quotient for $G$ will be equal to the Rayleigh quotient for $G \setminus e$ plus one more square term for the edge $e$, the Rayleigh quotient for $G$ will be greater. Hence $\lambda_2(H) \leq \lambda_2(G)$. □

**Corollary 3.8.** *[3] Let $H$ be a spanning subgraph $G$. Then $\lambda_2(H) \leq \lambda_2(G)$.*

*Proof.* Induction on the number of edges in $G$ not in $H$. □

The spectral gap is bounded both above and below by various measures of connectivity. For example, [4] gives various bounds for $\lambda_2$ in terms of the diameter, edge connectivity, vertex connectivity, and genus of the graph. In class, we discussed one such bound, *Cheeger's inequality*. To close this section, we prove on direction of this inequality for general graphs.

**Definition 3.9.** *[7] For some subset of vertices $S$, let $\partial S$ denote the set of edges $(i, j)$ where $i \in S$ and $j \notin S$. The* ***isoperimetric ratio*** *of $S$ is*

$$\theta(S) = \frac{|\partial S|}{|S|}$$

**Definition 3.10.** *[4] The* ***isoperimetric number*** *of $G$ is*

$$h(G) = \min_{0<|S|\leq \frac{n}{2}} \theta(S)$$

**Theorem 3.11.** *(Cheeger's inequality) Let $d$ be the maximum degree of a vertex. Then*

$$\frac{\lambda_2}{2} \leq h(G) \leq \sqrt{\lambda_2(2d - \lambda_2)}$$

*Proof.* We prove the lower bound on $h(G)$ and omit the proof of the upper bound (interested readers should refer to [1], [6], and [7]). Let $S$ be a set of vertices with $|S| \leq \frac{n}{2}$ so that $\theta(S)$ is minimal. Let $|S| = k$. Now let $x$ be the vector defined by $x_i = n - k$ if the vertex $v_i$ is in $S$ and $x_i = k$ otherwise. Note that $x \perp \mathbf{1}$, so to show $\lambda_2 \leq 2h(G)$ it suffices to show that

$$\frac{x^\top Lx}{|x|^2} \leq 2h(G).$$

First, we compute the numerator. Recall that

$$x^\top Lx = \sum_{i\sim j}(x_i - x_j)^2.$$

Since $x_i = x_j$ if $i, j \in S$ or $i, j \notin S$, this sum is equal to

$$\sum_{(i,j)\in\partial S}(x_i - x_j)^2 = |\partial S|n^2.$$

Next, we compute $|x|^2$. Using the definition of $x$, this is

$$k(n - k)^2 + (n - k)k^2 = kn(n - k).$$

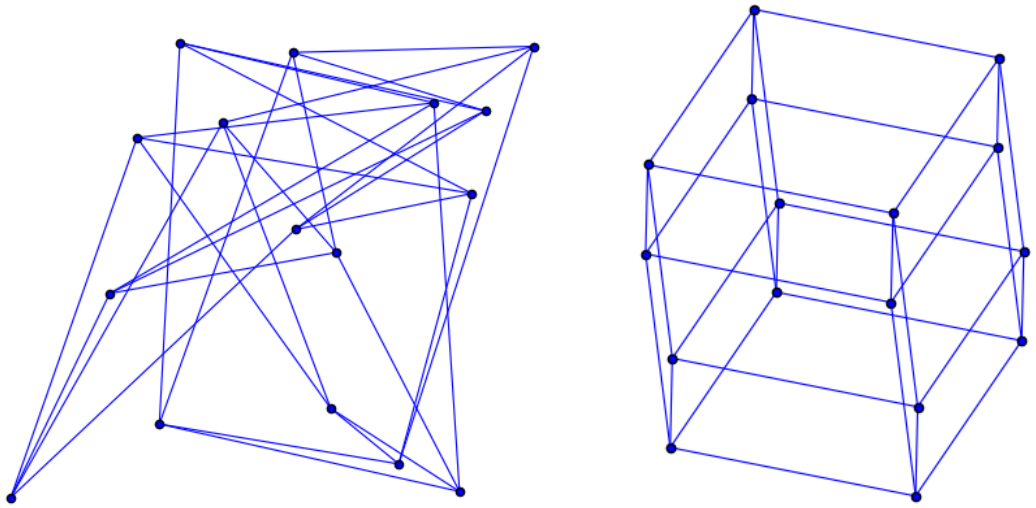
Now putting our expressions together gives us

$$\frac{x^\top Lx}{|x|^2} = \frac{|\partial S|n^2}{kn(n - k)} = h(G)\frac{n}{n - k}.$$

Since $k \leq \frac{n}{2}$ we have $\frac{n}{n-k} \leq 2$, so we are done. □

# 4 The Graph Drawing Problem

Consider the following two drawings of the 4-cube.



Clearly, the drawing on the right is better at conveying information about the structure in the graph. The goal of the *graph drawing problem* is to a produce meaning visualization of a given graph. We show in this section that the problem is solved cleanly by the studying the eigenvectors of $L$.

## 4.1 Defining the Problem

First we, formalize the notion of a graph drawing.

**Definition 4.1.** *A **graph drawing in dimension** $d$ is a function $\rho : V \to \mathbb{R}^d$.*

The information of a graph drawing can equivalently be encoded by arranging the points $\rho(v_1), \ldots, \rho(v_n)$ in a matrix.

**Definition 4.2.** *The **graph drawing matrix**, $R$, associated with the graph drawing $\rho$ is the $n \times d$ matrix with $\rho(v_i)$ as its $i$th row.*

Next, we define a cost function. Intuitively, we want nodes to be drawn close to each other when possible. Also, edges with greater weight should be assigned greater importance in the drawing. This motivates the following definition.

**Definition 4.3.** *The **energy** of a graph drawing $R$ is*

$$\mathcal{E}(R) := \sum_{i \sim j} w_{ij} |\rho(v_i) - \rho(v_j)|^2.$$

This is just the weighted sum of the squared lengths of each edge in our graph drawing. The underlying physical intuition is that if the edges of a graph are seen as a collection of springs in $d$-space attached to the nodes, then $\mathcal{E}(R)$ is literally the energy of the system (with spring constants corresponding to the weights $w_{ij}$).

Notice that by choosing a graph drawing where each $\rho(v_i)$ is the same, energy of 0 can always be obtained. We will impose some constraints on $R$ to avoid such degenerate solutions.

**Definition 4.4.** *A graph drawing $R$ is **balanced** if $\mathbf{1}^\top R = 0$.*

**Definition 4.5.** *A graph drawing $R$ is **orthogonal** if $R^\top R = I$.*

The condition that $R$ is balanced says that each of the $d$ coordinates should be centered around 0. Since any graph drawing can be made balanced by applying an appropriate translation, we consider only balanced graph drawings.

The condition that $R$ is orthogonal can be broken into two conditions: the norm of every column is 1, and columns of $R$ are pairwise orthogonal. It is reasonable to consider only drawings where each column of $R$ is nonzero, since if a column is 0 the we could have taken the drawing in a lower dimension. Then given that each column is nonzero, we can always scale our space so that the columns have unit norm.

As for the condition that columns are pairwise orthogonal, note that given an invertible $d \times d$ matrix we can get another graph drawing by applying the matrix to each point $\rho(v_i)$. This gives an action of $GL_d(\mathbb{R})$ (the group of invertible $d \times d$ matrices) on the set of graph drawings. If we consider graph drawings up to this action, then the graph drawing is determined by the column space. Hence each drawing can be represented by a matrix where the columns are orthogonal.

Now the graph drawing problem can be stated formally as the following optimization problem.

**Problem 4.6.** *Given a graph $G$, find the balanced, orthogonal graph drawing with minimal energy. That is, find $R$ minimizing $\mathcal{E}(R)$ subject to $\mathbf{1}^\top R = 0$ and $R^\top R = I$.*

## 4.2 Dimension One

First, we consider the problem in the one dimensional case $d = 1$. Then a graph drawing matrix is just an $n$-dimensional vector $x$. The balanced condition means $x$ is orthogonal to $\mathbf{1}$. The orthogonal condition means $x$ has norm 1. Finally, note that the energy is now $\sum_{i \sim j} w_{ij}(x_i - x_j)^2$. This is exactly $x^\top L x$.

Thus the graph drawing problem in dimension one is equivalent to finding

$$\min_{x \perp \mathbf{1}, |x|=1} x^\top L x$$

This is exactly the Rayleigh quotient expression that gives $\lambda_2$. The minimum value of $\lambda_2$ is attained by the eigenvector $\mu_2$. Hence the best drawing of a graph on the line is given by the vector corresponding to the second eigenvalue.

## 4.3 Higher Dimensions

In dimension one, we saw that $\mathcal{E}(x) = x^\top Lx$. The expression of the energy as a Rayleigh quotient is generalized to higher dimensions by the following proposition.

**Proposition 4.7.** *The energy of $R$ can be written*

$$\mathcal{E}(R) = \operatorname{tr}(R^\top LR).$$

*Proof.* Let $x^{(k)}$ denote the $k$th column of $R$. This is the $n \times 1$ vector containing $k$th coordinate assigned to each vertex. Note $Lx^{(k)}$ gives the $k$th column of $LR$ and $(x^{(k)})^\top$ is the $k$th row of $R^\top$. Hence $(x^{(k)})^\top Lx^{(k)}$ is the $k$th diagonal entry of $R^\top LR$. Taking the summation over all the diagonal entries, we find

$$\operatorname{tr}(R^\top LR) = \sum_{k=1}^{n} (x^{(k)})^\top Lx^{(k)}.$$

Now recall that for any vector $x$, the expression $x^\top Lx$ is equal to

$$\sum_{i \sim j} w_{ij}(x_i - x_j)^2.$$

Hence our sum can be written

$$\sum_{k=1}^{d} \sum_{i \sim j} w_{ij}(x_i^{(k)} - x_j^{(k)})^2.$$

Interchanging the summations, we get

$$\operatorname{tr}(R^\top LR) = \sum_{i \sim j} \sum_{k=1}^{d} w_{ij}(x_i^{(k)} - x_j^{(k)})^2.$$

Finally, note that $x_i^{(k)} = \rho(v_i)_k$. Thus our summation can be written

$$\begin{aligned}
\sum_{i \sim j} \sum_{k=1}^{d} w_{ij}(x_i^{(k)} - x_j^{(k)})^2 &= \sum_{i \sim j} w_{ij} \sum_{k=1}^{d} (\rho(v_i)_k - \rho(v_j)_k)^2 \\
&= \sum_{i \sim j} w_{ij} |\rho(v_i) - \rho(v_j)|^2 \\
&= \mathcal{E}(R)
\end{aligned}$$

so we are done. □

From this proposition, we can show that the result that the first eigenvalue solves the graph drawing problem in one dimension also generalizes in a natural way to higher dimensions. First, we need a lemma from linear algebra.

**Lemma 4.8.** *(Poincaré separation theorem) Let $A$ be an $n \times n$ real symmetric matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ (in ascending order). Let $B$ be an $n \times d$ matrix such that $B^\top B = I$. Let $\lambda'_1, \ldots, \lambda'_d$ denote the eigenvalues of $B^\top AB$. Then $\lambda_i \leq \lambda'_i$ for $i = 1, \ldots d$.*

*Proof.* Recall that $\lambda_i$ can be expressed as the Rayleigh quotient

$$\lambda_i = \min_{x \perp U_{i-1}} \frac{x^\top Ax}{x^\top x}.$$

where $U_{i-1}$ is the span of the first $i-1$ eigenvectors. Our goal will be to relate this Rayleigh quotient for $A$ to a Rayleigh quotient for $B^\top AB$. We will do so by choosing a certain $v \in \mathbb{R}^d$ and considering the Rayleigh quotients for $v$ and $Bv$ with respect to $B^\top AB$ and $A$ respectively.

First we establish some notation. Let $u_1, \ldots, u_n$ be an orthonormal basis of eigenvectors for $A$ and $v_1, \ldots, v_d$ an orthonormal basis of eigenvectors for $B^\top AB$. We will let $U_i$ denote the span of $u_1, \ldots, u_i$ and $V_j$ denote the span of $v_1, \ldots, v_j$.

Now note that $U_{i-1}$ is a dimension $i-1$ subspace of $\mathbb{R}^n$, so $B^\top U_{i-1}$ is a subspace of $\mathbb{R}^d$ of dimension at most $i-1$. Then taking the orthogonal complement, we see $(B^\top U_{i-1})^\perp$ has dimension at least $d - i + 1$. Since $V_i$ is a subspace of $\mathbb{R}^d$ of dimension $i$, the sum of $\dim(V_i)$ and $\dim(B^\top U_{i-1})^\perp$ is greater than $d$. Hence these two subspaces have nontrivial intersection. Let $v \in \mathbb{R}^d$ be some nonzero vector in this intersection.

Note that $Bv \perp U_{i-1}$, since $(Bv)^\top U_{i-1}$ can be written $v^\top (B^\top U_{i-1})$. Since $v \in (B^\top U_{i-1})^\perp$, this is just 0. Then considering the Rayleigh quotient of $Bv$ and simplifying with $B^\top B = I$, we see that

$$\lambda_i \leq \frac{(Bv)^\top ABv}{(Bv)^\top Bv} = \frac{v^\top B^\top ABv}{v^\top v}.$$

On the other hand, $\lambda'_i$ can also be expressed as a Rayleigh quotient as follows.

$$\begin{aligned} \lambda'_i &= \max_{x \perp \{v_{i+1}, \ldots, v_n\}} \frac{x^\top B^\top ABx}{x^\top x} \\ &= \max_{x \in V_i} \frac{x^\top B^\top ABx}{x^\top x} \end{aligned}$$

Since $v \in V_i$, we get

$$\lambda'_i \geq \frac{v^\top B^\top ABv}{v^\top v}.$$

Hence

$$\lambda_i \leq \frac{v^\top B^\top ABv}{v^\top v} \leq \lambda'_i.$$

so $\lambda_i \leq \lambda'_i$. □

Now we are ready to prove our main result on spectral graph drawing.

**Theorem 4.9.** *The minimal energy of a balanced, orthogonal graph drawing of $G$ in dimension $d$ is $\lambda_2+\lambda_3+\cdots+\lambda_{d+1}$. This minimal energy is attained by the matrix where the columns are unit eigenvectors corresponding to these eigenvalues.*

*Proof.* We wish to find $R$ minimizing $\mathcal{E}(R)$ such that $R^\top R = I$ and $\mathbf{1}^T R = 0$. First, we consider the problem without the constraint $\mathbf{1}^T R = 0$. Since $\mathcal{E}(R) = \operatorname{tr}(R^\top LR)$ is the sum of the eigenvalues of $R^\top LR$, by applying the preceding lemma to each eigenvalue of $R^\top LR$ we have $\mathcal{E}(R) \geq \lambda_1 + \lambda_2 + \cdots + \lambda_d$. This lower bound is attained by letting the columns of $R$ be $\mu_1 = \mathbf{1}/\sqrt{n}, \mu_2, \ldots, \mu_d$.

We show now that the minimal energy over *balanced*, orthogonal graph drawings in dimension $d$ is the same as the minimal energy over *all* orthogonal graph drawings in dimension $d+1$. Applying the above result to dimension $d+1$, we have that the minimal energy in dimension $d+1$ is given by orthogonal eigenvectors $\mu_1, \ldots, \mu_{d+1}$ as the columns of the matrix. The corresponding energy is $\lambda_1 + \cdots + \lambda_{d+1}$. Since $\lambda_1 = 0$, this sum is equal to $\lambda_2 + \cdots + \lambda_{d+1}$. Hence the matrix given by the eigenvectors $\mu_2, \ldots, \mu_{d+1}$ as columns gives an orthogonal graph drawing in dimension $d$ with the same energy. Since each of $\mu_2, \ldots, \mu_{d+1}$ is orthogonal to $\mu_1$, this drawing is balanced. Conversely, given a balanced drawing in dimension $d$ we can get a drawing in dimension $d+1$ with the same energy by adding $\mu_1$ as one of the columns. Hence the two minima are equal.

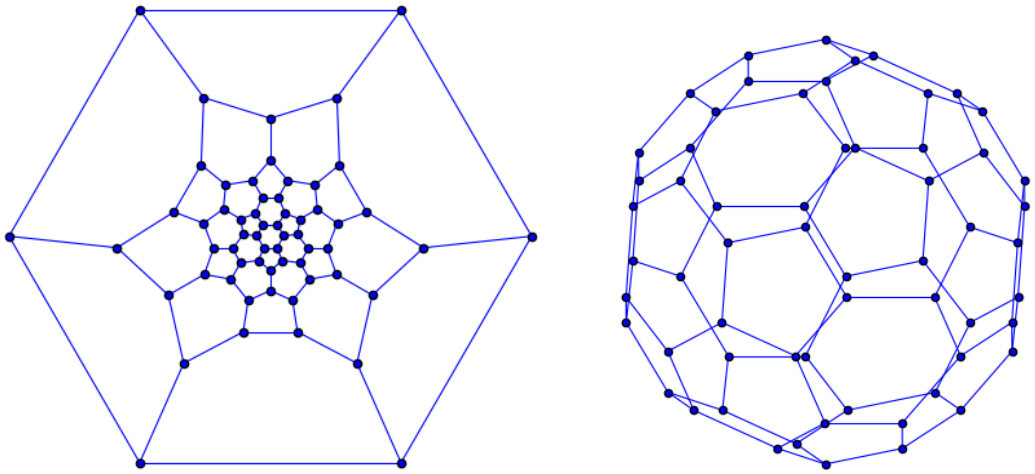
As a result, the minimal energy over balanced, orthogonal graph drawings in dimension $d$ is $\lambda_2 + \cdots + \lambda_{d+1}$. This energy is attained by using eigenvectors $\mu_2, \ldots, \mu_{d+1}$ as columns of the graph drawing. □

## 5 Experiments with Spectral Graph Drawing

We implemented the spectral graph drawing algorithm in Sage and used it produce a large number of two and three dimensional graph drawings. The examples below have also been made available on the SageMath cloud here (the page make take a while to load as it opens the Sage project and plots the graphs). The Sage workesheet also contains three dimensional plots that can be rotated.

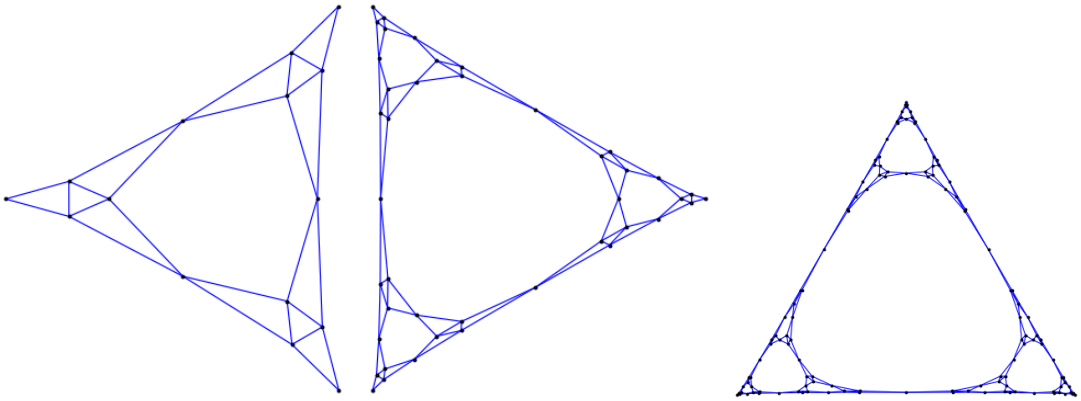
## 5.1 Graph Drawing Gallery

**Example 5.1.** *(Buckyball) Below are two drawings of the buckyball in* $\mathbb{R}^2$.



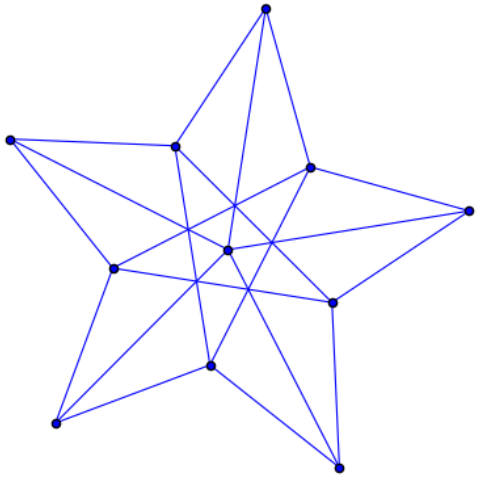
*The drawing on the left is a planar drawing given by Sage. The right is the drawing given by the spectral drawing algorithm. Note that the spectral drawing is a projection of a three dimensional drawing. This is because the two dimensional drawing is just the first two coordinates of the three dimensional drawing.*

**Example 5.2.** *(Sierpinski's Triangle) The following are spectral drawings for 3, 4, and 5 iterations of Sierpinski's triangle. It appears the spectral drawings are converging. In these and other spectral drawings, the tendency is that topological features such as large holes are emphasized.*



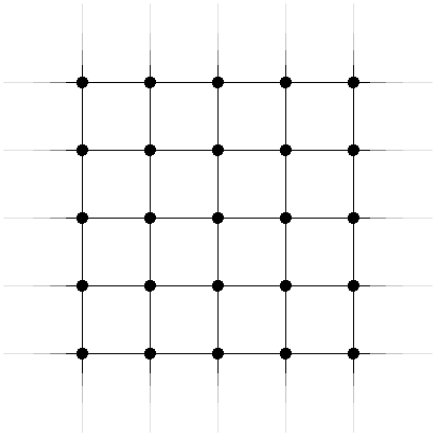
**Example 5.3.** *(Mycielski Graphs) The Mycielski graphs are a recursively defined family of graphs that demonstrate triangle free graphs can have arbitrarily large chromatic number. The following is a spectral drawing of the fourth iteration. The drawing makes*

*apparent the five-way symmetry in the graph, which is not obvious from the definition of the graph.*



*A three dimensional plot of the fifth iteration can be found in the SageMath worksheet.*

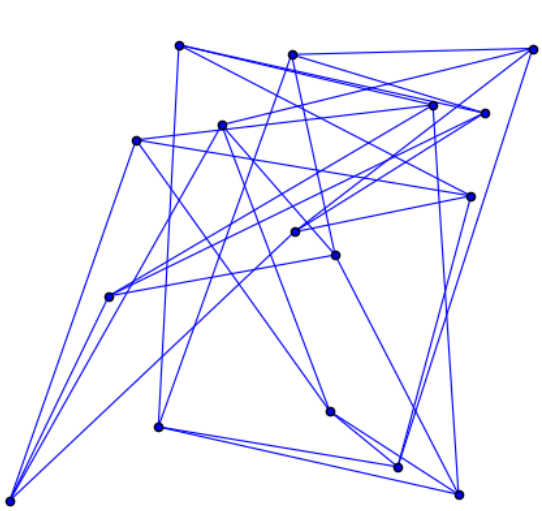
**Example 5.4.** *(Toroidal Grid) The $n \times m$ toroidal grid is the 4-regular graph on $nm$ vertices given by an $n \times m$ grid plus extra edges for points on the boundary. The picture below illustrates a $5 \times 5$ toroidal grid (the edges on the left wrap around to become edges on the right, and edges going off the top wrap around to become the edges on the bottom).*

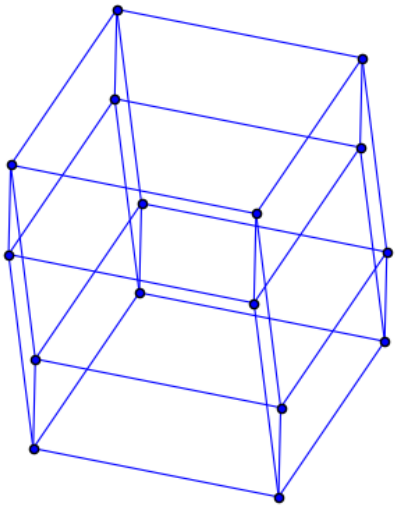


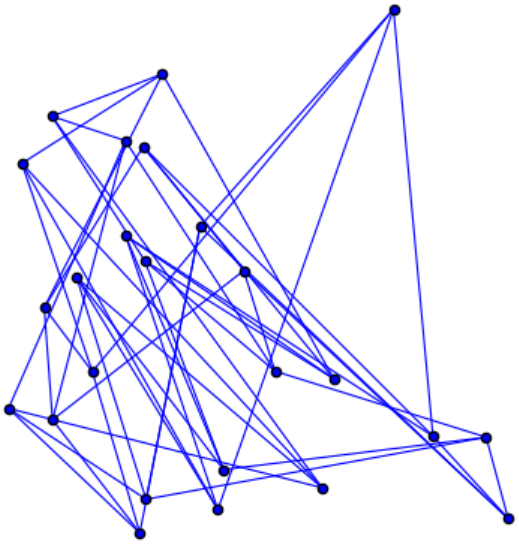
*The toroidal grid cannot be embedded in the plane but can be naturally embedded on the torus, since the torus can be seen as a quotient of the unit square.*

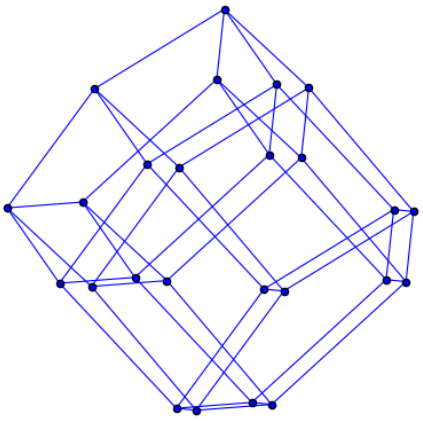
*Below are six drawings of toroidal grids. The dimensions are $4 \times 4, 5 \times 5$, and $15 \times 15$. For interested readers, we also recommend looking at the three dimensional plots in Sage worksheet. Fascinatingly, the spectral drawing in three dimensions seems to converge to the surface of a torus as we increase the size of the graph. In the drawings below, the*

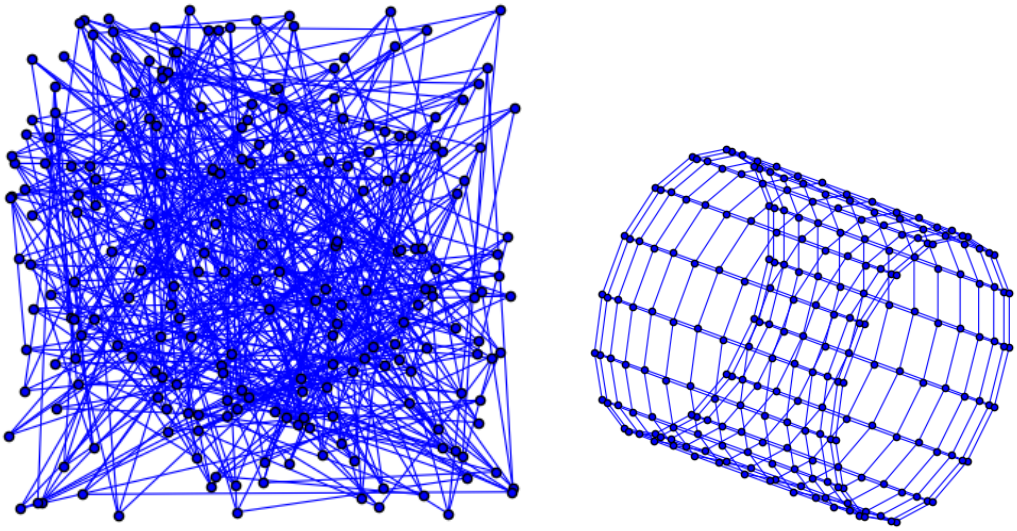
*left column contains pictures produced by the default Sage algorithm. The pictures in the right column are given by the spectral graph drawing algorithm in dimension two.*





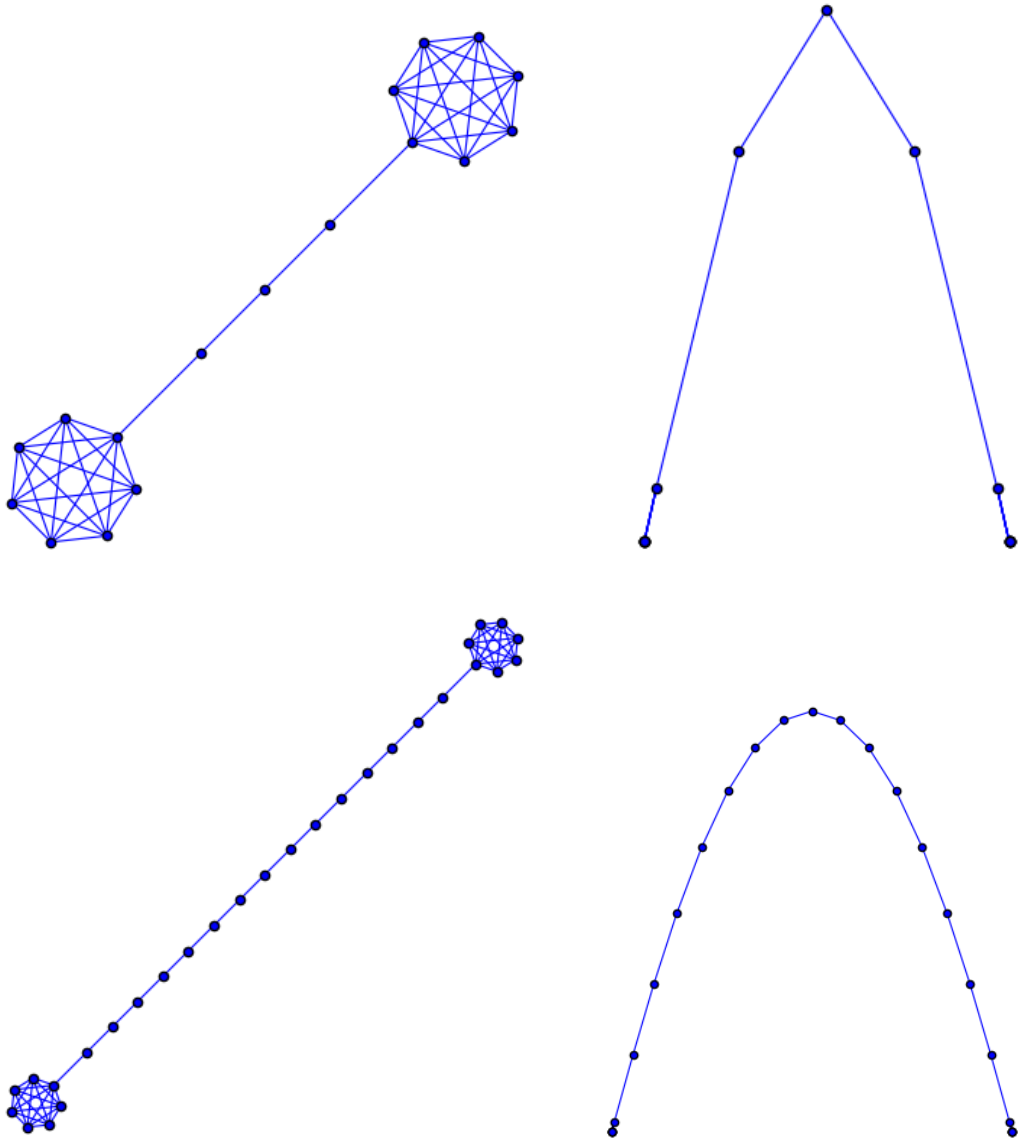






*Note that the* $4 \times 4$ *toroidal grid is the same example from the beginning of Section 4; as it so happens, the* $4 \times 4$ *toroidal grid is isomorphic to the* 4*-cube. Overall, the pattern is that while the default Sage algorithm completely fails to provide meaningful drawings, the spectral drawing algorithm produces orderly drawings that accurately depict the symmetries in the graph. Also, the inherently toroidal nature of the graphs can be seen in the spectral drawings. For example, the* $15 \times 15$ *example appears to be a see-through side view of a torus, where the denser part in the middle of the graph corresponds to the hole. In the three dimensional drawings (in the Sage worksheet), it is apparent that the spectral drawings actually converge to the surface of a torus.*

**Example 5.5.** *(Barbell Graphs) A barbell graph is two copies of $K_n$ connected by a path. Below are two barbell graphs. On the left column are drawings given by the default Sage algorithm. On the right are drawings produced by the spectral graph drawing algorithm.*



*In this case, the spectral drawing is clearly worse; the $K_7$'s are collapsed to a single point. In general, in small examples and examples such as these where the vertices have a natural clustering, we saw that the spectral graph algorithm tends to map many vertices to the same point.*

### 5.2 Conclusion

Overall, the spectral graph drawing algorithm outperforms other methods in some cases but does worse in others. The spectral algorithm does particularly well for graphs that are nonplanar but topological in nature (e.g. the toroidal grid, which is most naturally drawn on a nonplanar surface). In these cases, the Sage default algorithm performed very poorly in comparison. This makes sense, since Sage used a force-directed algorithm to draw graphs. Force-directed drawing methods (algorithms that iteratively improve the drawing by moving vertices and edges in the plane away from each other) may have a hard time handling graphs that are not inherently planar, since it only moves vertices within the plane. Also, we saw in examples such as the buckyball and toroidal grids that the spectral algorithm does well at drawing high dimensional data, since the two dimensional spectral drawing will be a projection of a good high dimensional drawing. In contrast, since the force-directed algorithm can only move points in the plane, it may have trouble capturing this higher dimensionality. The downside to spectral graph drawing is that often many vertices are mapped to the same point, losing important information in the graph. This seems to happen especially for graphs where the vertices have natural clusterings.

## 6 Experiments with Spectral Graph Clustring

The *graph clustering problem* is the problem of dividing the vertices of $G$ into $k$ similar sets. One real-world application of this problem is image segmentation, where one divides an image into several components. The application of spectral graph theory to this problem was first explored by Shi and Malik in their paper[5]. Shi and Malik proposed converting an image to graph with each node representing a pixel with outgoing edges to neighboring pixels and edge weight determined by their similarity.

In Section 4, we saw that the eigenvectors of the Laplacian $L$ give the optimal drawing of $G$ in $d$ dimensional space. Spectral graph clustering works by taking these embeddings of vertices into $d$-space, normalizing, and then using $k$-means clustering to give a partitioning of the the vertices. For example, the second eigenvector $\mu_2$ can be used to divide the graph into two set. To obtain a further finer segmentation, the process can be applied recursively.

In this section we examine image segmentation using spectral clustering using the implementation available in scikit-image library. We then compare the results against Otsu's method and watershed techinque found in scikit-image and opencv2 respectively.

### 6.1 Methods for Image Segmentation

#### 6.1.1 Spectral Clustering

The scikit-image implementation of spectral clustering segmentation transforms the image to a graph with edge weights determined by pixel to pixel gradient. Then the algorithm described above is run. By default, once the algorithm compute the eigenvectors it uses $k$-means to get the clustering.

### 6.1.2 Otsu's Method

Otsu's method is a histogram based method that divides each pixel into one of two groups based on their intensity. The groupings are decided so the intraclass variance is minimized (or equivalently, interclass variance is maximized). The algorithm exhaustively iterates through all possible thresholds to find the one that gives maximal variance according to the equation

$$\sigma_b^2(t) = w_0(t)w_1(t)[\mu_0(t) - \mu_1(t)]^2$$

where weights $w_0$ and $w_1$ are the class probabilities at a threshold $t$ and $\mu_0$, $\mu_1$ the corresponding class means.

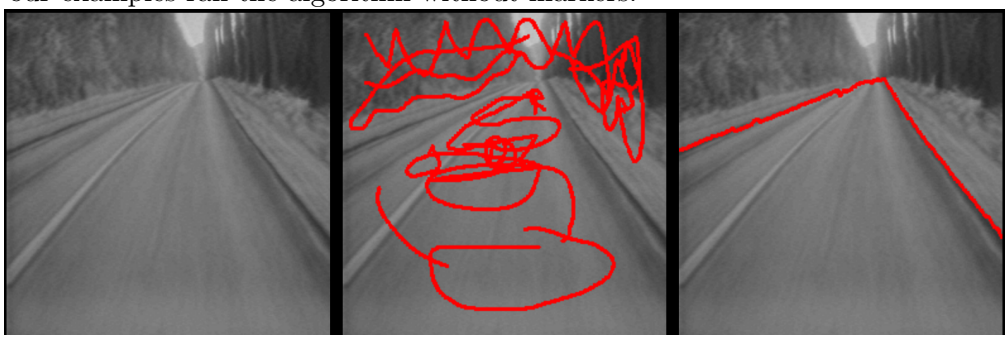
### 6.1.3 Watershed Technique

The watershed technique is based off the idea that any grayscale image can be viewed as a topographic surface. Each region of the image that is considered a valley is filled with water by some specified footprint size (in pixels). When regions being filled by water overlap we build a barrier which corresponds to the pixel production that segments the image.

## 6.2 Testing Methodology

We ran all python code on a dual core 2016 MacBook. All images are preprocessed before being fed into the respective algorithms and benchmarked for real world runtime. Part of the preprocessing is converting the images to greyscale, since all three algorithms use the pixel intensity as the gradient to determine the appropriate segmentation.

It is known that the watershed technique is susceptible to over-segmenting or under-segmenting the image, depending on the input image. Generally, the technique greatly benefits from human added markers as seen in the middle image below. The figure below shows an example of how markers might be used to enhance the segmentation. However, in our examples ran the algorithm without markers.

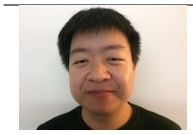
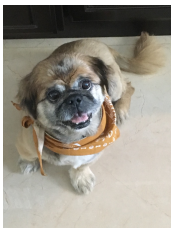


We found that running spectral based image segmentation was computationally expensive for large images. As such, we downsampled our images to around $200 \times 200$ pixels when feeding it to the clustering algorithm. We find this acceptable, as it is common place to downscale the image and apply label assignment on the original image. We ran

all three algorithms on four different images (headshot, nature, cartoon, animal). All associated code and images are available here.

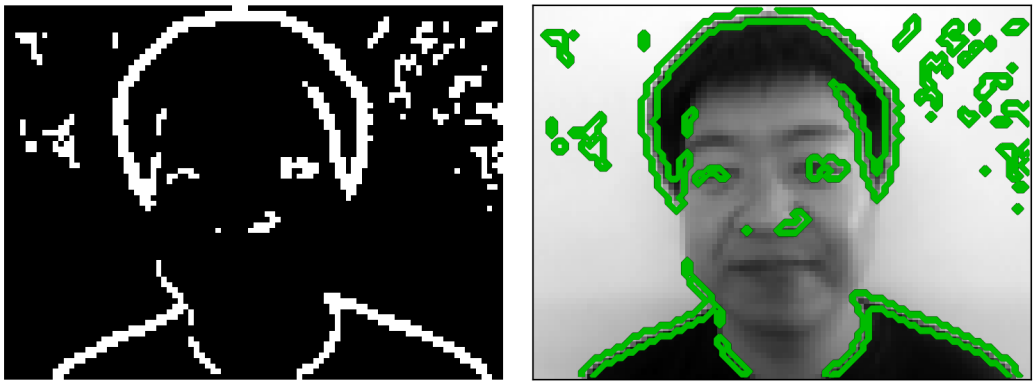
## 6.3 Results

### 6.3.1 Real world runtime

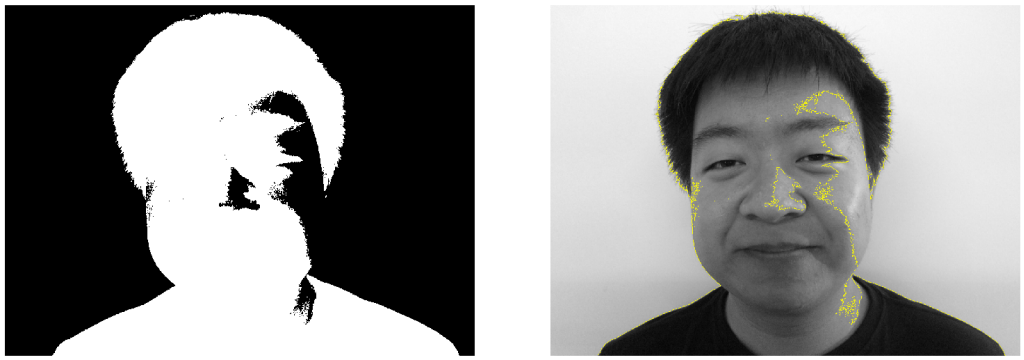
| Image | Spectral | Otsu | Watershed |
|---|---|---|---|
|  | 4.16s | 0.10s | 0.15s |
|  | 6.86s | 0.004s | 0.01s |
|  | 116.02s | 0.03s | 0.03s |
|  | 44.49s | 0.09s | 0.11s |

### 6.3.2 Outputs

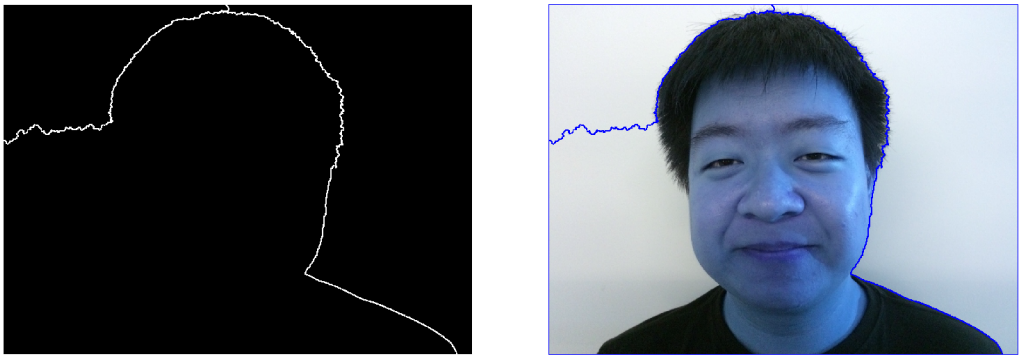
For convenience we show the resultant mask and it overlayed over the original image.



Spectral clustering: 4.16s

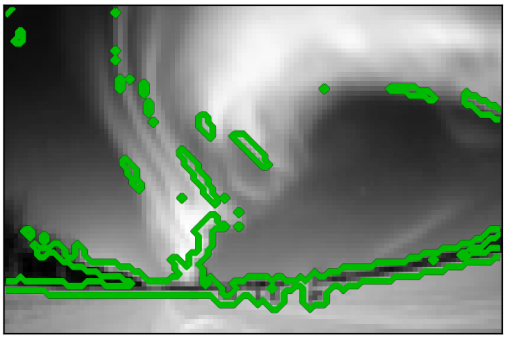


Histogram-based: 0.10s

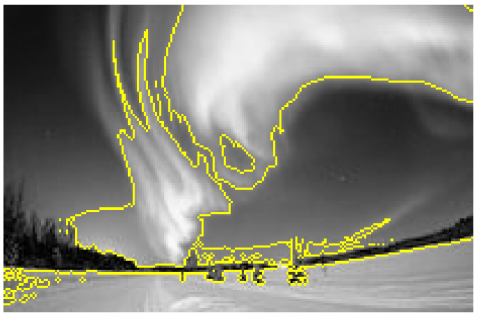


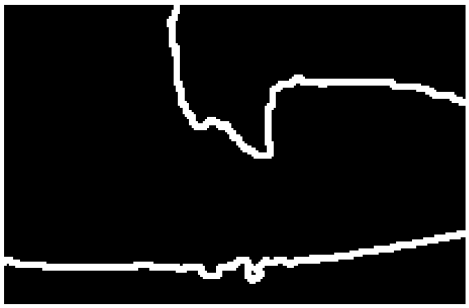
Watershed: 0.15s

Spectral clustering: 6.86s



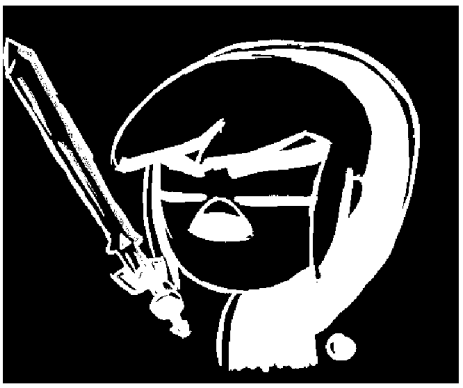
Histogram-based: 0.00s



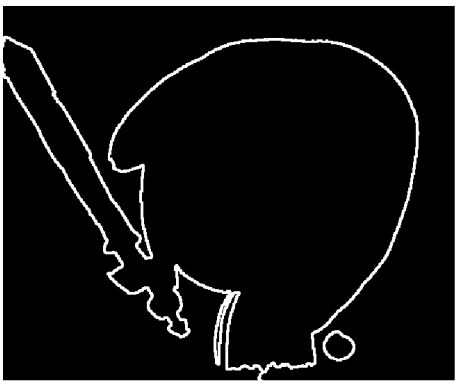


Watershed: 0.01s

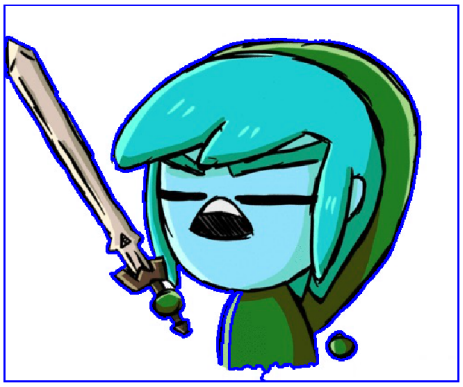
Spectral clustering: 116.02s



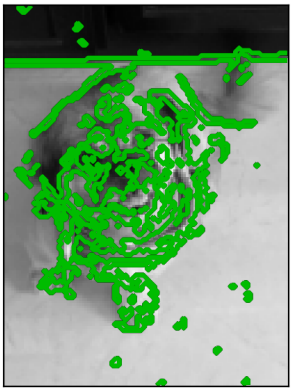
Histogram-based: 0.03s



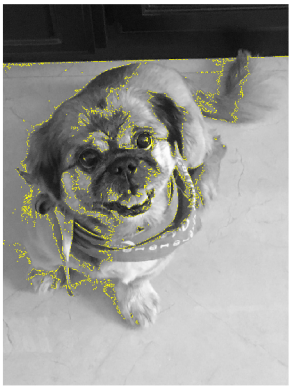
Watershed: 0.03s

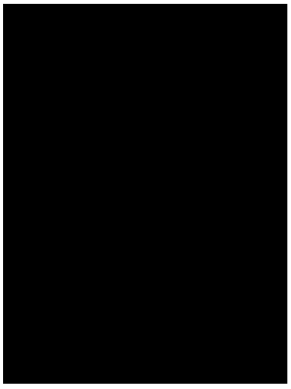


Spectral clustering: 44.49s

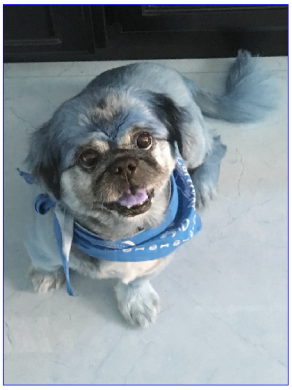


Histogram-based: 0.09s





Watershed: 0.11s



### 6.3.3 Headshot

Of the three methods, the watershed technique performed the poorest. While it captured a partial outline of the head, the resulting cut included a large portion of the background. Otsu's method and spectral clustering both provide interesting results that make the headshot apparent from the mask. The spectral clustering actively targets areas where the change in gradient is most drastic in the original image (black hair and shirt to background/skin). It captures the hairline, eyes, nose, and shirt. Unfortunately there is some noise in the background from the non uniform color background. The histogram based methodology effectively removes the headshot from the background with some minor noise from light reflected off the face.

### 6.3.4 Nature

Again we have an interesting outline from the watershed technique. It segments the largest pseudo-contiguous white segment of snow and aurora borealis from the rest of the image. Spectral clustering and Otsu's method neither provide a clear image from the mask. Otsu's method groups the snow and the aurora borealis together due to the near equivalent pixel intensity. Spectral clustering captures the field of trees with some artifacts in the dark spots of the aurora borealis. In terms of usability of images though spectral clustering provides the a more informative distinction of what is captured in its group relative to human perception.

### 6.3.5 Cartoon

The image of Lonk is an example where direct application of the watershed method without markers still produced favorable results. Specifically, it succeeded in capturing the complete silhouette of the portrait. We suspect this is because the image has a near uniform background. Otsu's method also captures much of the detail from the original image and puts any green part of the image in a single cluster. Spectral clustering provides a middle ground between the two images by capturing both the silhouette and some level of detail without forcing all the green into a single group. However, compared to the other two methods the spectral algorithm took several orders of magnitude longer.

### 6.3.6 Animal

In this case, the watershed method fails completely, as it considers the entire image a basin. We suspect this is because the image is pseudo-uniform; the dog blends into the background. The complexity of the shades of the dogs fur causes Otsu's method to cluster the darker patches of fur with the cabinetry at the top of the image. Spectral clustering may provide a more interpretable result as the cabinetry is not binned entirely together but rather where it intersects the floor. The spectral clustering also shows a moderate amount of detail in the face of the dog. The downsampling makes it difficult to make out the finer details, but the more pronounced features are more visible than in the histogram-based segmentation.

### 6.4 Conclusion

Across the three algorithms, the pattern was that there was a tradeoff between speed and quality of segmentation. Since the algorithm requires operations on large matrices, spectral clustering took by far the longest to run. As such, we do not recommend the algorithm for online purposes. In terms of the quality of segmentation, it is hard to interpret quality of the results in a computer vision context without feeding the images through some application. However, from a human perspective spectral provided the most consistent segmentation.

## References


[1] Fan Chung. Laplacians of graphs and cheeger inequalities.

[2] Jean Gallier. Spectral theory of unsignedand signed graphsapplications to graph clustering: a survey. 2016.

[3] Jonathan Gross, Jay Yellen, and Ping Zhang. Handbook of graph theory. page 327 and 584, 2003.

[4] Bojan Mohar. The laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, 1991.

[5] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[6] Daniel Spielman. Conductance, the normalized laplacian, and cheeger's inequality, from notes on spectral graph theory. 2015.

[7] Daniel Spielman. The laplacian, from notes on spectral graph theory. 2015.