Chapter 4
Spectral transform

We are witnessing an explosion of multimedia content, such as music, images, and video, over the internet and in the world of PDAs (personal data assistants) today. One of the main driving forces behind this phenomenon is the availability of effective *compression* algorithms for these data. What differentiates the compression of visual data, such as an image, from conventional text compression is that while the latter exploits predictabilities in the bit representation of the input text, the former focuses on organizing regularities in the visual perception of the given data.

Typically, high compression ratios are achieved when certain data loss can be tolerated. One of the most popular approaches to lossy compression relies on *transform-based* techniques, where data given in the time or spatial domain are transformed into the spectral or frequency domain and data compression can be achieved by properly lowering the accuracy of the representations in the frequency domain. A prominent example in image compression is JPEG, where high-frequency contents in an image are represented at lower accuracy, since the human eye is less sensitive to high-frequency brightness variations. Although in practice, JPEG image compression involves steps such as color space transformation and downsampling (to accentuate brightness over chrominance), block splitting (dividing an image into small blocks), and quantization (round numbers to integers), the main ingredient of the approach is the use of the discrete cosine transform (DCT) [Jai89] to transform an image into the frequency domain. In Figure 1, we illustrate the effect of eliminating (i.e., zeroing) particular percentages of the high-frequency DCT coefficients, achieving the effect of compression.

As we can see from Figure 1, the compressed images appear to be blurred or smoothed versions of the original with certain sharp features near the edges and high-

Original 80% 90% 95% 98% 99%

Figure 1: The Lena image can be compressed by eliminating the indicated percentages of DCT coefficients. The images shown are the results of reconstruction using the remaining DCT coefficients.

frequency details (e.g., over the hat accessories) removed. Indeed, each compressed image is constructed as a weighted sum of a set of *basis images*. The basis images represent intensity oscillations at varying frequencies, as shown in Figure 2, where we plot all of the 64 DCT basis images for 8×8 grayscale images. For a natural image, such as the Lena in Figure 1, the most significant transform coefficients, which are the weights in the weighted sum, correspond to low-frequency basis images and those which characterize high-frequency contents can typically be eliminated without introducing noticeable quality degradation in the reconstructed image. In general, when high-frequency contents are removed, the image is smoothed.

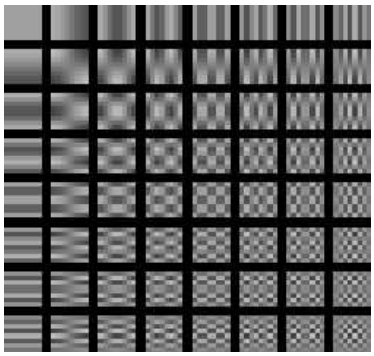


Figure 2: Plots of 64 DCT basis images for 8×8 grayscale images.

As we can see from Figure 1, with only the first 20% of the (low-frequency) basis images used, corresponding to a compression ratio of 5:1, we already obtain a faithful reconstruction of the original image. With the other steps taken by conventional JPEG compression and adding entropy coding to the quantized trasnform coefficients, much higher compression rates can be obtained.

With the recent advances in 3D data acquisition technology, highly detailed geometric models can be obtained via laser scanning. Efficient storage and delivery of these large geometric models motivate the development of geometry compression methods. Similar to image compression, visual perception plays the key role in geometry compression, where intensity variations over an image are replaced by geometric undulations over the surface of a geometric model. It is natural to ask then whether transform-based lossy compression techniques, such as a JPEG-like scheme, can be applied to geometry. This question is answered in this chapter where we present tools for spectral transform of 3D geometric data.

Before going into 3D data, we first take a look at the 2D case in Section 1, where a 2D shape is represented by a contour. This allows us to reduce the 2D shape analysis problem to the study of 1D functions specifying the contour. We introduce the notion of Laplacian smoothing and show how spectral analysis based on 1D discrete Laplacian operators can perform smoothing as well as compression. The relationship between spectral analysis based on Laplacians and the classical Fourier transform is revealed in Section 3. Spectral transforms derived from eigendecomposition of mesh Laplacian operators are defined in Section 4, where we discuss their applications to mesh compression and filtering.

1 Laplacian smoothing and spectral analysis in 1D

Consider the seahorse shape depicted by a closed contour, shown in Figure 3. The contour is represented as a sequence of 2D points (the contour vertices) that are connected by straight line segments (the contour segments), as illustrated by a zoomed-in view.

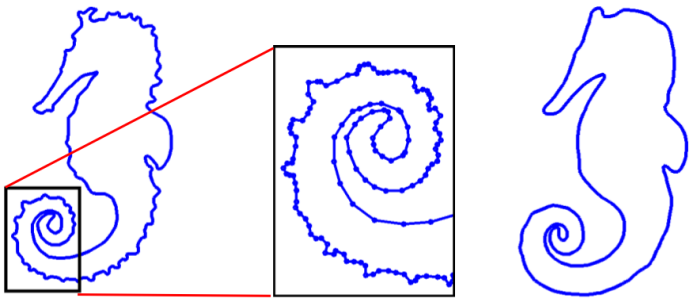


Figure 3: A seahorse shape with rough features (left) and a smoothed version (right).

Now suppose that we wish to remove the rough features over the shape of the seahorse and obtain a smoothed version, such as the one shown in the right of Figure 3.

1.1 Laplacian smoothing

A simple procedure to accomplish this is to repeatedly connect the midpoints of successive contour segments; we refer to this as *midpoint smoothing*. Figures 4(a) illustrates two such steps. As we can see, after two steps of midpoint smoothing, each contour vertex $\mathbf{v}_i$ is moved to the midpoint of the line segment connecting the midpoints of the original contour segments adjacent to $\mathbf{v}_i$. Specifically, let $\mathbf{v}_{i-1} = (x_{i-1}, y_{i-1})$, $\mathbf{v}_i = (x_i, y_i)$, and $\mathbf{v}_{i+1} = (x_{i+1}, y_{i+1})$ be three consecutive contour vertices. Then the new vertex $\hat{\mathbf{v}}_i$ after two steps of midpoint smooothing is given by a *local averaging*,

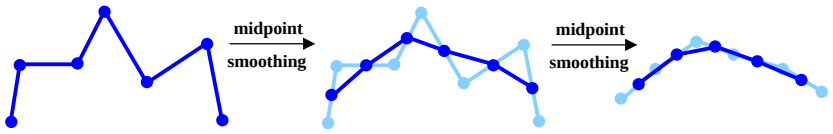
$$\hat{\mathbf{v}}_i = \frac{1}{2}\left[\frac{1}{2}(\mathbf{v}_{i-1} + \mathbf{v}_i)\right] + \frac{1}{2}\left[\frac{1}{2}(\mathbf{v}_i + \mathbf{v}_{i+1})\right] = \frac{1}{4}\mathbf{v}_{i-1} + \frac{1}{2}\mathbf{v}_i + \frac{1}{4}\mathbf{v}_{i+1}. \tag{1}$$

The vector between $\mathbf{v}_i$ and the midpoint of the line segment connecting the two vertices adjacent to $\mathbf{v}_i$, shown as a red arrow in Figure 4(b), is called the *1D disctete Laplacian* at $\mathbf{v}_i$, denoted by $\delta(\mathbf{v}_i)$,

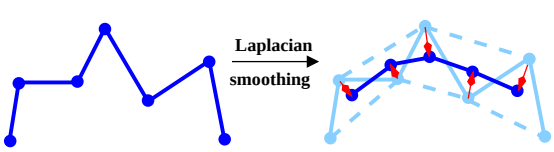
$$\delta(\mathbf{v}_i) = \frac{1}{2}(\mathbf{v}_{i-1} + \mathbf{v}_{i+1}) - \mathbf{v}_i. \tag{2}$$

As we can see, after two steps of midpoint smoothing, each contour vertex is displaced by half of its 1D discrete Laplacian, as shown in Figure 4(c); this is referred to as *Laplacian smoothing*. The smoothed version of the seahorse in Figure 3 was obtained by applying 10 steps of Laplacian smoothing.

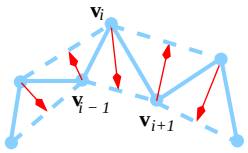
Some natural questions one would ask include: why is Laplacian smoothing removing the rough features (or noise) in the data? What will happen if we apply Laplacian smoothing repeatedly, e.g., what will happen in the limit? To answer these questions, we will first need to express our problem and algorithms algebraically and then apply analysis using the algebraic tools we had developped in previous chapters.



(a) Two steps of midpoint smoothing.



(b) Laplacian smoothing.



(c) 1D discrete Laplacians (red).

Figure 4: One step of Laplacian smoothing (b) is equivalent to two steps of midpoint smoothing (a). The 1D discrete Laplacian vectors are shown in red in (c).

2 Signal representation and spectral transform

Let us denote the contour vertices by a coordinate vector V, which has n rows and two columns, where n is the number of vertices along the contour and the two columns correspond to the x and y components of the vertex coordinates. Let us denote by x_i (respectively, y_i) the x (respectively, y) coordinate of a vertex $\mathbf{v}_i$, $i = 1, \ldots, n$. For analysis purposes, let us only consider the x component of V, denoted by X; the treatment of the y component is similar.

We treat the vector X as a discrete 1D signal. Since the contour is closed, we can view X as a periodic 1D signal defined over uniformly spaced samples along a circle, as illustrated in Figure 5(a). We sort the contour vertices in counterclockwise order and plot it as a conventional 1D signal, designating an arbitrary element in X to start the indexing. Figure 5(b) shows such a plot for the x-coordinates of the seahorse shape ($n = 401$) from Figure 3.

We can now express the 1D Laplacians (2) for all the vertices using an $n \times n$ matrix L, called the *1D discrete Laplacian operator*, as follows:

$$\delta(X) = LX = \begin{bmatrix} 1 & -\frac{1}{2} & 0 & \ldots & \ldots & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & \ldots & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & \ldots & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \ldots & \ldots & 0 & -\frac{1}{2} & 1 \end{bmatrix} X. \tag{3}$$

Laplacian smoothing can then be expressed as applying a smoothing operator S to the signal X, resulting in a new contour represented by $\hat{X} = SX$. The smoothing

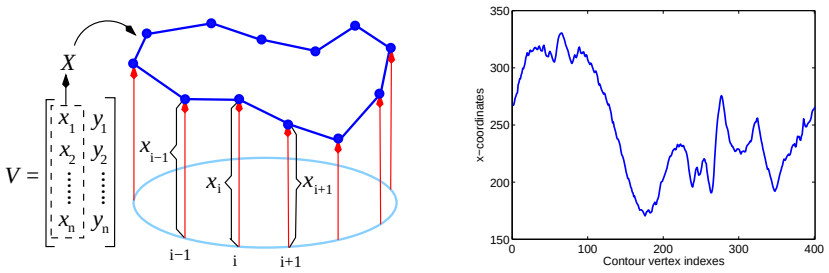


Figure 5: The x-component of the contour coordinate vector V, X, can be viewed as a 1D periodic signal defined over uniform samples along a circle, as shown on the left. Right: it is shown by a conventional 1D plot for the seahorse contour from Figure 3.

operator

$$S = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & 0 & \dots & \dots & 0 & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & 0 & \dots & \dots & 0 & \frac{1}{4} & \frac{1}{2} \end{bmatrix} \tag{4}$$

is related to the Laplacian operator by $S = I - \frac{1}{2}L$.

To analyze the behavior of Laplacian smoothing, in particular what happens in the limit, we utilize the set of basis vectors formed by the eigenvectors of L. This leads to a framework for *spectral analysis* of geometry. From linear algebra, we know that since L is symmetric, it has real eigenvalues and a set of real and orthogonal set of eigenvectors which form a basis. Any vector of size n can be expressed as a linear sum of these basis vectors. We are particularly interested in such an expression for the coordinate vector X. Denote by $\mathbf{e}_1$, $\mathbf{e}_2$,...,$\mathbf{e}_n$ the normalized eigenvectors of L, corresponding to eigenvalues λ_1, λ_2, ..., λ_n, and let E be the matrix whose columns are the eigenvectors. Then we can express X as a linear sum of the eigenvectors,

$$X = \sum_{i=1}^{n} \mathbf{e}_i \tilde{x}_i = \begin{bmatrix} E_{11} \\ E_{21} \\ \vdots \\ E_{n1} \end{bmatrix} \tilde{x}_1 + \ldots + \begin{bmatrix} E_{1n} \\ E_{2n} \\ \vdots \\ E_{nn} \end{bmatrix} \tilde{x}_n = \begin{bmatrix} E_{11} & E_{12} & \dots & E_{1n} \\ E_{21} & E_{22} & \dots & E_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ E_{n1} & E_{n2} & \dots & E_{nn} \end{bmatrix} \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ \vdots \\ \tilde{x}_n \end{bmatrix} = E\tilde{X}. \tag{5}$$

Since the eigenvectors form an orthonormal basis, E is invertible and $E^{-1} = E^{\mathrm{T}}$, the matrix transpose of E. Hence, we can write

$$\tilde{X} = E^{\mathrm{T}} X.$$

This represents a transform from the original signal X to a new signal $\tilde{X}$ in terms of a new basis, the basis given by the eigenvectors of L. We call this a *spectral transform*. For each i, the *spectral transform coefficient*

$$\tilde{x}_i = \mathbf{e}_i^{\mathrm{T}} \cdot X. \tag{6}$$

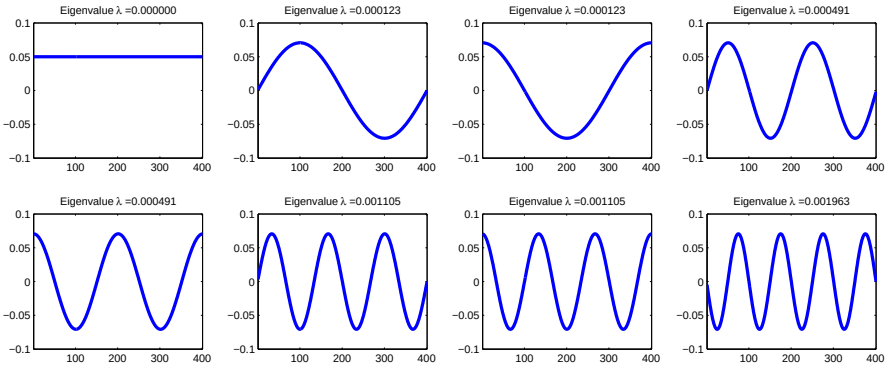


Figure 6: Plots of first 8 eigenvectors of the 1D discrete Laplacian operator ($n = 401$) given in equation (3). They are sorted by eigenvalue λ, shown above each plot.

That is, the spectral coefficient $\tilde{x}_i$ is obtained as a *projection* of the signal X along the direction of the i-th eigenvector $\mathbf{e}_i$. In Figure 6, we plot the first 8 eigenvectors of L, sorted by increasing eigenvalues, where $n = 401$, maching the size of the seahorse shape from Figure 3. The indexing of elements in each eigenvector follows the same contour vertex indexing as X, the coordinate vector, as plotted in Figure 5(b) for the seahorse. It is worth noting that aside from an agreement on indexing, the Laplacian operator L and the eigenbasis vectors do not depend on X, which specifies the geometry of the contour. L, as defined in equation (3), is completely determined by n, the size of the input contour, and a vertex ordering.

As we can see, the eigenvector corresponding to the zero eigenvalue is a constant vector. As eigenvalue increases, the eigenvectors start to oscillate as sinusoidal curves at higher and higher frequencies. Note that the eigenvalues of L repeat (multiplicity 2) after the first one, hence the corresponding eigenvectors of these repeated eigenvalues are not unique. One particular choice of the eigenvectors reveals a connection of our spectral analysis to the classical Fourier analysis; this will be discussed in Section 3.

2.1 Signal reconstruction and compression

With the spectral transform of a coordinate signal defined as in equation (5), we can now look at compression and filtering of a 2D shape represented by a contour. Analogous to image compression using JPEG, we can obtain compact representations of a contour by retaining the leading (low-frequency) spectral transform coefficients and eliminating the rest. Given a signal X as in equation (5), the signal reconstructed by using the k leading coefficients is

$$X^{(k)} = \sum_{i=1}^{k} \mathbf{e}_i \tilde{x}_i, \qquad k \leq n. \tag{7}$$

This represents a compression of the contour geometry as only k out of n coefficients need to be stored to approximate the original shape. We can quantify the information

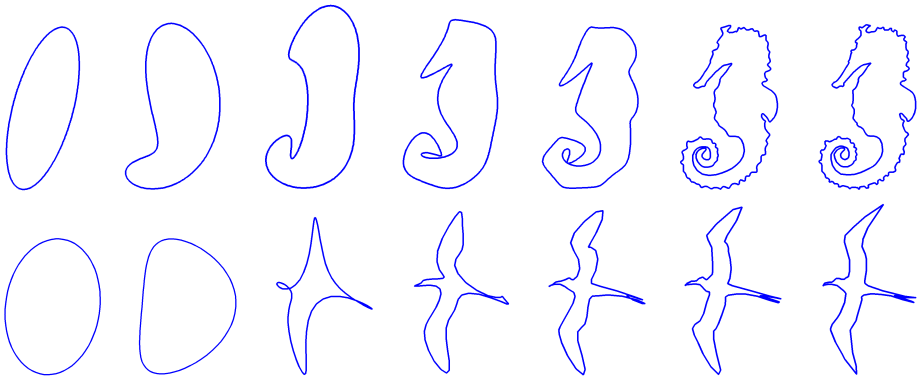


Figure 7: Shape reconstruction via Laplacian-based spectral analysis.

loss by measuring the L_2 error or Euclidean distance between X and $X^{(k)}$,

$$||X - X^{(k)}|| = || \sum_{i=k+1}^{n} \mathbf{e}_i \tilde{x}_i || = \sqrt{\sum_{i=k+1}^{n} \tilde{x}_i^2}.$$

The last equality is easy to obtain if we note the orthogonality of the eigenvectors, i.e., $\mathbf{e}_i^{\mathrm{T}} \cdot \mathbf{e}_j = 0$ whenever $i \neq j$. Also, since the $\mathbf{e}_i$'s are normalized, $\mathbf{e}_i^{\mathrm{T}} \cdot \mathbf{e}_i = 1$.

In Figure 7, we show some results of this type of shape reconstruction (7), with varying k for the seahorse and a bird shape. As more and more high-frequency spectral coefficients are removed, i.e., with decreasing k, we obtain smoother and smoother reconstructed contours. How effectively a 2D shape can be compressed this way may be visualized by plotting the spectral transform coefficients, the $\tilde{x}_i$'s in (6), as done in Figure 8. In the plot, the horizontal axis represents eigenvalue indexes, $i = 1, \ldots, n$, which roughly correspond to frequencies. One can view the magnitude of the $\tilde{x}_i$'s as the energy of the input signal X at different frequencies.

A signal whose energies are sharply concentrated in the low-frequency end can be effectively compressed at a high compression rate, since as a consequence, the total energy at the high-frequency end, representing the reconstruction error, is very low. Such signals will exhibit fast decay in its spectral coefficients. Both the seahorse and the bird models contain noisy or sharp features so they are not as highly compressible as a shape with smoother boundaries. This can be observed from the plots in Figure 8. Nevertheless, at 2:1 compression ratio, we can obtain a fairly good approximatios, as one can see from Figure 7.

2.2 Filtering and Laplacian smoothing

Compression by truncating the vector $\tilde{X}$ of spectral transform coefficients can be seen as a *filtering* process. When a discrete filter function f is applied to $\tilde{X}$, we obtain a new coefficient vector $\tilde{X}'$, where $\tilde{X}'(i) = f(i) \cdot \tilde{X}(i)$, for all i. The filtered signal X' is then

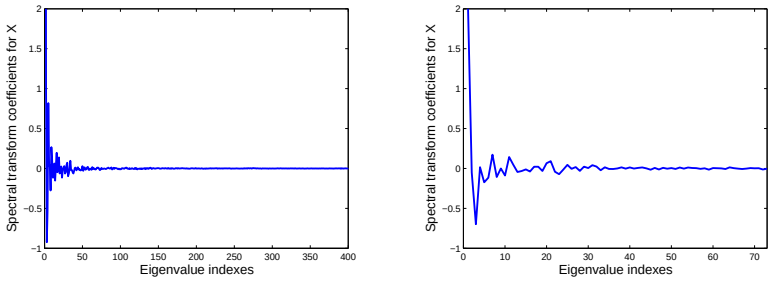


Figure 8: Plot of spectral transform coefficients for the x component of a contour; refer to equation (6). Left: seahorse. Right: bird. The models are shown in Figure 7.

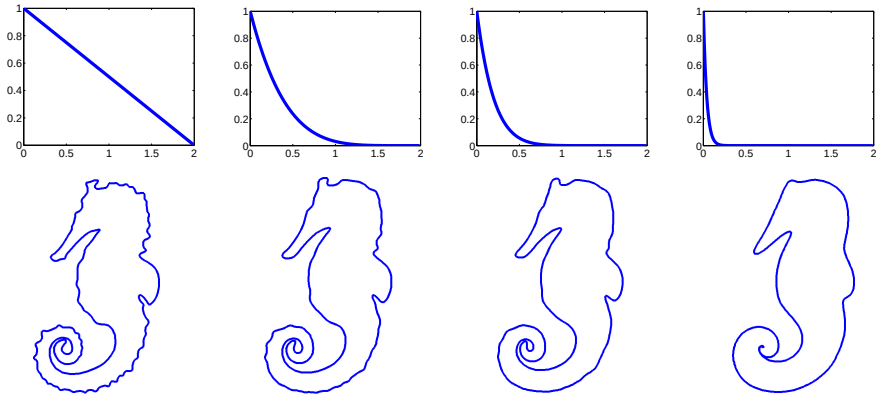


Figure 9: First row: filter plots, $(1-\frac{1}{2}\lambda)^m$ with $m = 1, 5, 10, 50$. Second row: corresponding results of Laplacian smoothing on the seahorse.

reconstructed from $\tilde{X}'$ by $X' = E\tilde{X}'$, where E is the matrix of eigenvectors as defined in equation (5). We next show that Laplacian smoothing is one particular filtering process. Specifically, when we apply the Laplacian smoothing operator S to a coordinate vector m times, the resulting coordinate vector becomes,

$$X^{(m)} = S^m X = (I - \frac{1}{2}L)^m X = \sum_{i=1}^{n} (I - \frac{1}{2}L)^m \mathbf{e}_i \tilde{x}_i = \sum_{i=1}^{n} \mathbf{e}_i (1 - \frac{1}{2}\lambda_i)^m \tilde{x}_i. \qquad (8)$$

The last equality above results from the definition of eigenvalue/eigenvector, $L\mathbf{e}_i = \lambda_i \mathbf{e}_i$. For example, we note that by applying the definition k times, we get $L^k \mathbf{e}_i = \lambda_i^k \mathbf{e}_i$.

Equation (8) provides a characterizaion of Laplacian smoothing in the spectral domain via filtering and the filter function is given by $f(\lambda) = (1 - \frac{1}{2}\lambda)^m$. A few such filters with different m are plotted in the first row of Figure 9. The corresponding Laplacian smoothing leads to attenuation of the high-frequency content of the signal and hence achieves denoising or smoothing, as shown in the second row.

To examine the limit behavior of Laplacian smoothing, let us look at equation (8).

Note that it can be shown (via the Gerschgorin's Theorem [TB97]) that the eigenvalues of the Laplacian operator are in the interval $[0,2]$ and the smallest eigenvalue $\lambda_1 = 0$. Since $\lambda \in [0,2]$, the filter function $f(\lambda) = (1 - \frac{1}{2}\lambda)^m$ is bounded by the unit interval $[0,1]$ and attains the maximum $f(0) = 1$ at $\lambda = 0$. As $m \to \infty$, all the terms in the right-hand side of equation (8) will vanish except for the first, which is given by $\mathbf{e}_1\tilde{x}_1$. Since $\mathbf{e}_1$, the eigenvector corresponding to the zero eigenvalue is a normalized, constant vector, we have $\mathbf{e}_1 = [\frac{1}{\sqrt{n}} \ \frac{1}{\sqrt{n}} \ \ldots \ \frac{1}{\sqrt{n}}]^{\mathrm{T}}$. Now taking the *y*-component into consideration, we get the limit point for Laplacian smoothing as $\frac{1}{\sqrt{n}}[\tilde{x}_1 \ \tilde{y}_1]$. Finally, noting that $\tilde{x}_1 = \mathbf{e}_1^{\mathrm{T}} \cdot X$ and $\tilde{y}_1 = \mathbf{e}_1^{\mathrm{T}} \cdot Y$, we conclude that the limit point of Laplacian smoothing is the *centroid* of the set of original contour vertices.

3 Spectral analysis vs. discrete Fourier transform

In the previous section, we started with a purely geometric treatment of contour smoothing based on 1D discrete Laplacians. We then cast the problem from a signal processing perspective and applied the algebraic tool of spectral analysis to interpret compression, Laplacian smoothing, and general filtering of a signal. For those who are familiar with signal processing and Fourier analysis, a connection appears to be obvious. In particular, by looking at plots of the eigenvectors of the 1D discrete Laplacian operator (3) in Figure 6, one notices their resemblence to the sinosoidal curves of the Fourier or DCT basis functions. We now make that connection explicit.

Typically, one introduces discrete Fourier transform (DFT) in the context of Fourier series expansion. Given a discrete signal $X = [x_1 \ x_2 \ \ldots \ x_n]^{\mathrm{T}}$, its DFT is given by

$$\tilde{X}(k) = \frac{1}{n}\sum_{j=1}^{n} X(k)e^{-i2\pi(k-1)(j-1)/n}, k = 1,\ldots,n.$$

The corresponding inverse DFT is given by

$$X(j) = \sum_{k=1}^{n} \tilde{X}(k)e^{i2\pi(j-1)(k-1)/n}, j = 1,\ldots,n,$$

or

$$X = \sum_{k=1}^{n} \tilde{X}(k)\mathbf{g}_k, \text{ where } \mathbf{g}_k(j) = e^{i2\pi(j-1)(k-1)/n}, k = 1,\ldots,n.$$

We see that in the context of DFT, the signal X is expressed as a linear combination of the complex exponential DFT basis functions, the $\mathbf{g}_k$'s. The coefficients are given by the $\tilde{X}(k)$'s, which form the DFT of X. Fourier analysis, provided by the DFT in the discrete setting, is one of the most important topics in mathematics and has wide-ranging applications in many scientific and engineering disciplines. For a systematic study of the subject, we refer the reader to the classic text by Bracewell [Bra99].

The connection we seek, between DFT and spectral analysis with respect to the Laplacian, is that the DFT basis functions, the $\mathbf{g}_k$'s, form a set of eigenvectors of the 1D discrete Laplacian operator L, give in (3). A proof of this fact can be found in Jain's classic text on image processing [Jai89], where a stronger claim with respect to

circulant matrices was made. A matrix is circulant if each row can be obtained as a shift (with circular wrap-around) of the previous row. It is clear that L is circulant.

Specifically, if we sort the eigenvalues of L in ascending order, then they are

$$\lambda_k = 2\sin^2 \frac{\pi\lfloor k/2 \rfloor}{n}, k = 2, \ldots, n. \tag{9}$$

The first eigenvalue λ_1 is always 0. We can see that every eigenvalue of L, except for the first, and possibly the last, has a multiplicity of 2. That is, it corresponds to an eigensubspace spanned by two eigenvectors. If we define the matrix G of DFT basis as $G_{kj} = e^{i2\pi(j-1)(k-1)/n}$, $1 \leq k, j \leq n$, then the first column of G is an eigenvector corresponding to λ_1 and the k-th and $(n+2-k)$-th columns of G are two eigenvectors corresponding to λ_k, for $k = 2, \ldots, n$. The set of eigenvectors of L is not unique. In particular, it has a set of real eigenvectors as shown in Figure 6.

We remark that increasing eigenvalues of L, given in (9), correspond to increasing frequencies of the associated eigenvectors. Generally, the frequency of a 1D signal can be measured by the number of zero-crossings. In the case of the DFT basis, the k-th and $(n+2-k)$-th bases both have $2(k-1)$ zero-crossings. For a different set of eigenvectors of L (corresponding to the same set of eigenvalues), such as the ones shown in Figure 6, the number of zero-crossings changes but the relationship between eigenvalues and frequencies is still monotonic.

It turns out that the above connection also applies to the DCT and the discrete sine transform and the three transforms can be treated in a unified manner [Jai89]. Consider the following parametric family of matrices,

$$\mathbf{J} = \mathbf{J}(k_1, k_2, k_3) = \begin{bmatrix} 1-k_1\alpha & -\alpha & 0 & 0 & \ldots & 0 & k_3\alpha \\ -\alpha & 1 & -\alpha & 0 & \ldots & 0 & 0 \\ 0 & -\alpha & 1 & -\alpha & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & -\alpha & 1 & -\alpha & 0 \\ 0 & \ldots & 0 & 0 & -\alpha & 1 & -\alpha \\ k_3\alpha & 0 & \ldots & 0 & 0 & -\alpha & 1-k_2\alpha \end{bmatrix}$$

It can be shown that the basis vectors of the DCT, the discrete sine transform, and DFT are respectively the complete and orthogonal sets of eigenvectors of $\mathbf{J}(1,1,0)$, $\mathbf{J}(0,0,0)$, and $\mathbf{J}(1,1,-1)$. These eigenvectors all exhibit sinusoidal behavior. The DCT generally possesses better energy compaction properties compared to the other two transforms. This means that for the same signal, the DCT coefficients exhibit faster decay, leading to better compression. The DCT, DFT, and discrete sine transforms all admit fast constructions, in time $O(n \log_2 n)$, where n is the length of the signal, making them practical to use for a variety of signal and image processing tasks.

Based on the above observation, one way to extend the notion of Fourier analysis to the surface setting, where our signal will represent the geometry of the surfaces, is to define appropriate discrete Laplacian operators for surface meshes and rely on the eigenvectors of the Laplacians to perform Fourier-like analysis. This observation was made by Taubin in his seminal paper on a signal processing approach to mesh smoothing [Tau95] in 1995. This work has inspired a string of developments on spectral mesh processing, as covered in the comprehensive survey by Zhang et al. [ZvKD10].

4 Spectral analysis on meshes

In this section, we extend spectral analysis of 1D signals presented in Section 1 to surfaces modeled by triangle meshes. A triangle mesh is a tessellation of a surface in 3D using triangles; it is formed by a set of triangles pasted along their edges. The main issue to resolve is the irregularity of the neighborhood of a mesh vertex. In our 1D case, every vertex has precisely two neighbors. For a mesh, each vertex can have any number of neighbors. Extending the signal representation to the mesh case requires us to define an appropriate *mesh Laplacian* operator. Once that is done, spectral analysis of a mesh signal with respect to the eigenvectors of the mesh Laplacian works in exactly the same way as for its 1D counterpart.

4.1 Mesh signal representation and mesh Laplacian

A triangle mesh with n vertices is represented as $\mathscr{M} = (\mathscr{G}, P)$, where $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ models the mesh graph, with $\mathscr{V}$ denoting the set of mesh vertices and $\mathscr{E} \subseteq \mathscr{V} \times \mathscr{V}$ the set of edges, and $P \in \mathbb{R}^{n\times 3}$ represents the geometry of the mesh, given by an array of 3D vertex coordinates. Each vertex $i \in \mathscr{V}$ has an associated position vector, denoted by $\mathbf{p}_i = [x_i\ y_i\ z_i]$; it corresponds to the i-th row of P. Any function defined on the mesh vertices can be seen as a discrete mesh signal. Here we focus on P, the coordinate signal for the mesh. As before, we will only deal with the x-component, denoted by X. The treatment of the y and z components is similar.

Discrete mesh Laplacian operators are linear operators that act on discrete signals defined on the vertices of a mesh. If a mesh $\mathscr{M}$ has n vertices, then the mesh Laplacian will be given by an $n \times n$ matrix. Loosely speaking, a mesh Laplacian operator locally takes a weighted average of the differences between the value of a signal at a vertex and its value at the first-order or immediate neighbour vertices. Although there are several ways to define a mesh Laplacian [ZvKD10], we focus our attention on the simplest of them, which is a direct generalization of the 1D discrete Laplacian from Section 1. Specifically, the midpoint between two points adjacent to the point in question in the 1D case is replaced by the centroid of the point's immediate neighbors in the mesh.

Let us denote by W the adjacency matrix of the mesh graph $\mathscr{G}$. Thus

$$W_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \mathscr{E}, \\ 0 & \text{otherwise.} \end{cases}$$

The degree matrix D is a diagonal matrix of the row sums of W,

$$D_{ij} = \begin{cases} d_i = |N(i)| & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

where $N(i)$ is the set of immediate neighbor vertices of i, and d_i is said to be the degree of vertex i. Note that the above W and D matrices are $n \times n$ matrices, where $n = |\mathscr{V}|$.

We define our *mesh Laplacian* matrix T as

$$T = I - D^{-1}W. \tag{10}$$

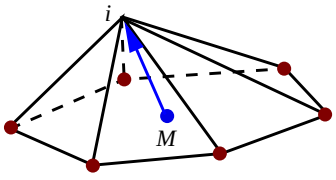
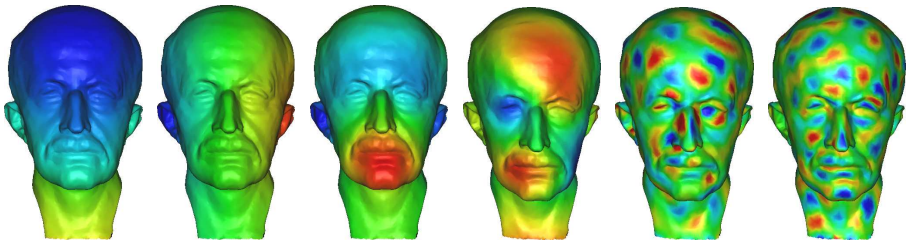


Figure 10: Discrete mesh Laplacian (blue arrow) at vertex i. M is the centroid of i's neighbors.



$i = 2.$ $i = 4.$ $i = 6.$ $i = 16.$ $i = 250.$ $i = 360.$

Figure 11: Color plots of a few eigenvectors (corresponding to the i-th smallest eigenvalues) of the mesh Laplacian of the Max Planck model.

Applying T to a mesh signal X yields the discrete Laplacian at each vertex

$$\delta(x_i) = x_i - \frac{1}{|N(i)|} \sum_{j \in N(i)} x_j. \qquad (11)$$

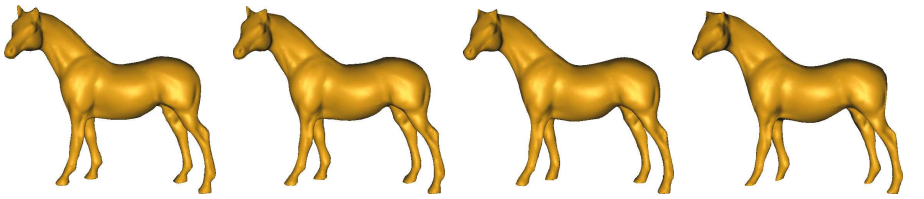
We see that $\delta(x_i)$ is a vector directed from the centroid M of the immediate neighbors of vertex i to vertex i, as illustrated in Figure 10.

4.2 Spectral mesh transform

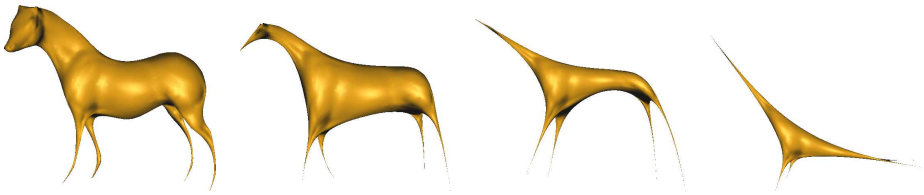
With the mesh Laplacian operator T defined, defining the spectral transform of a mesh signal X with respect to T is exactly the same as the 1D case for L in Section 2. Denote by $\mathbf{e}_1$, $\mathbf{e}_2$,...,$\mathbf{e}_n$ the normalized eigenvectors of T, corresponding to eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$, and let E be the matrix whose columns are the eigenvectors. The vector of spectral transform coefficients $\tilde{X}$ is obtained by $\tilde{X} = E^{\mathrm{T}}X$. And for each i, we obtain the spectral coefficient by $\tilde{x}_i = \mathbf{e}_i^{\mathrm{T}} \cdot X$ via projection.

In Figure 11, we plot several eigenvectors of a mesh model using color coding, where eigenvector entries are used to index into a color map. The oscillatory and low-to-high frequency vibration patterns of the eigenvectors as the eigenvalues increase are evident. In Figure 12, we show spectral reconstruction, as define in (7), of a mesh model with progressively more spectral coefficients added. As we can see, higher-frequency contents of the geometric mesh signal manifest themselves as rough geometric features over the shape's surface; these features can be smoothed out by taking only the low-frequence spectral coefficients in a reconstruction. A tolerance on such loss of geometric features leads to a JPEG-like compression of mesh geometry [KG00].

Just as in the 1D classical case, the oscillatory behavior of the Laplacian eigenvectors and the implied spectral transform of geometric mesh signals allow us to apply "Fourier-like" analysis to mesh models. In addition to spectral geometry compression (Figure 12), any type of filtering can be applied to the spectral coefficients [DMSB99, KR05, Tau95, VL08] to obtain a global editing of a mesh shape. A few examples are



(a) Original model. (b) $k = 300$. (c) $k = 200$. (d) $k = 100$.



(e) $k = 50$. (f) $k = 10$. (g) $k = 5$. (h) $k = 3$.

Figure 12: Shape reconstruction based on spectral analysis using a mesh Laplacian operator, where k is the number of eigenvectors or spectral coefficients used. The original model has 7,502 vertices and 15,000 faces. [[Richard: will change to a better model.]]

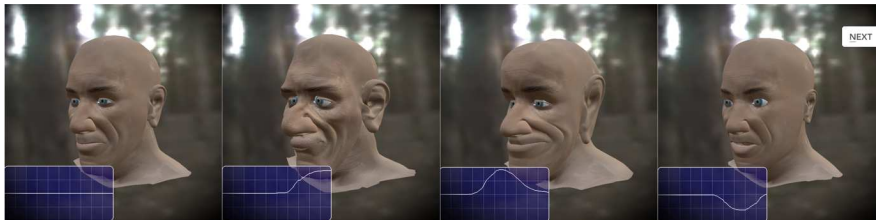


Figure 13: A few examples of mesh filtering. [[Richard: will reproduce; these are copied from Vallet's paper in 2008.]]

shown in Figure 13, mimicking their counterparts in the 1D classical case shown in Figure 9. A variety of other applications which utilize the spectral coefficients or the eigenvalues and eigenvectors themselves, as derived from different variants of the mesh Laplacian operator (10), have been developed over the years. We refer the reader to the recent survey [ZvKD10] for a comprehensive coverage.

4.3 Comparison to classical Fourier analysis

Comparing spectral mesh transforms to classical Fourier analysis, there are at least two crucial differences. First, while the DFT and DCT basis functions are fixed as long as the length of the signals in consideration is determined, the eigenvectors of the mesh Laplacian T, on the other hand, would change with the mesh graph connectivity. Since the same surface can be tessellated in different ways by a triangle mesh, it may be desirable to use a mesh Laplacian operator that is geometry-aware and not dictated by

the particular surface tessellation chosen.

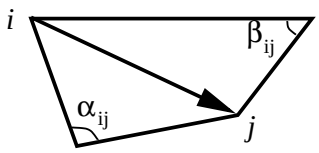


Figure 14: Angles which define the cotangent weights for L^{cot}.

A popular geometric mesh Laplacian is the cotangent operator due to Pinkall and Polthier [PP93]. For a closed triangle mesh, the cotangent Laplacian is given by $L^{\text{cot}} = D^{\text{cot}} - W^{\text{cot}}$, where W^{cot} is defined by the cotangent weights:

$$W_{ij}^{\text{cot}} = \frac{1}{2}(\cot\alpha_{ij} + \cot\beta_{ij}),$$

where α_{ij} and β_{ij} are opposite angles to edge (i, j) in the mesh (see Figure 14) and if (i, j) is not an edge, $W_{ij}^{\text{cot}} = 0$. The matrix D^{cot} is a diagonal matrix of the row sums of W^{cot}. Since the cotangent weights are meant to model the geometry of the mesh surface, the resulting mesh Laplacian L^{cot} and spectral transform are shape-dependent, but are largely insensitive to change of mesh tessellation, in contrast to T.

The second difference between classical DFT and spectral mesh transforms is from the computational aspect. While the former admits fast $O(n\log n)$ computations, e.g., fast DFT or FFT [Bra99], computing the eigenvectors of general mesh Laplacians, such as T or L^{cot}, has cubic time complexity. There are several ways to speed up this process, e.g., via spectral shift and invert [VL08], algebraic multi-grid methods, or settling for approximated results [ZvKD10], but these are beyond the scope of our coverage.

References

[Bra99] Ronald N. Bracewell. *The Fourier Transform And Its Applications*. McGraw-Hill, 1999.

[DMSB99] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proc. of ACM SIGGRAPH*, pages 317–324, 1999.

[Jai89] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, 1989.

[KG00] Z. Karni and C. Gotsman. Spectral compression of mesh geometry. In *Proc. of ACM SIGGRAPH*, pages 279–286, 2000.

[KR05] Byung Moon Kim and Jarek Rossignac. Geofilter: Geometric selection of mesh filter parameters. *Computer Graphics Forum (Special Issue of Eurographics 2005)*, 24(3):295–302, 2005.

[PP93] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, 2(1):15–36, 1993.

[Tau95] G. Taubin. A signal processing approach to fair surface design. In *Proc. of ACM SIGGRAPH*, pages 351–358, 1995.

[TB97] Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra*. SIAM, 1997.

[VL08] B. Vallet and B. Lévy. Spectral geometry processing with manifold harmonics. 27(2):251–260, 2008.

[ZvKD10] Hao Zhang, Oliver van Kaick, and Ramsay Dyer. Spectral mesh processing. *Computer Graphics Forum*, 29(6):1865–1894, 2010.