A Numerical Investigation of Laplacian Eigenmaps

Ashka Dalal[1], Bobita Atkins[2], Natalie Dinin[3], Tess McGuinness[4]

[1]Rose-Hulman Institute of Technology
[2]Massachusetts College of Liberal Arts
[3]California State University, Chico
[4]University of Connecticut

August 7, 2023

Outline

How to Construct Eigencoordinates from a Set of Points

Consider a set of points. These points are connected if the distance, d, between them is less than or equal to ε. We can embed these connections in a Laplacian Matrix defined as

$$L = D - W$$

How to Construct Eigencoordinates from a Set of Points

W is the adjacency matrix of the graph. Each edge is given a weight defined by w_{ij}.

$$w_{ij} = \begin{cases} 1 \text{ if } |x_i - x_j| \leq \varepsilon, i \neq j \\ 0 \text{ else} \end{cases} .$$

This graph is undirected, so $w_{ij} = w_{ji}$. Thus,

$$W = (w_{ij})_{i,j=0}^{n-1}$$

How to Construct Eigencoordinates from a Set of Points

D is the degree matrix where the diagonal elements are row sums of the adjacency matrix.

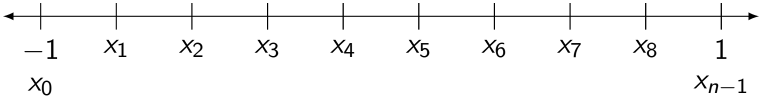
$$D_i = \sum_{j=0}^{n-1} W_{ij}$$

How to Construct Eigencoordinates from a Set of Points

Calculate the eigenvalues and eigenvectors of the Laplacian Matrix. Call each eigenvalue and eigenvector pair λ_i and v_i respectively. Sort the eigenvalues such that $0 = \lambda_0 < \lambda_1 < \lambda_2 < \cdots < \lambda_n$. We then select v_1 and v_2 and plot the two against each other. This plot is known as the eigenmap or eigencoordinates.

Evenly Spaced Points

We begin by choosing evenly spaced points off the interval $[-1, 1]$.



Note that we select n points, each a distance $\delta = \frac{2}{n-1}$ apart from each other. We also set an $\varepsilon = k\delta$ where $k = 1, \ldots, \lfloor \frac{n-1}{2} \rfloor$. Depending on the value of n and k that we select, we construct eigencoordinates based on points that are evenly spaced in the interval $[-1, 1]$.

Eigencoordinates generated from equally spaced points on the interval $[-1, 1]$

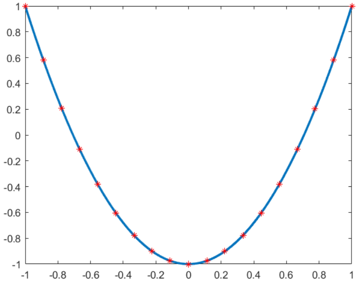


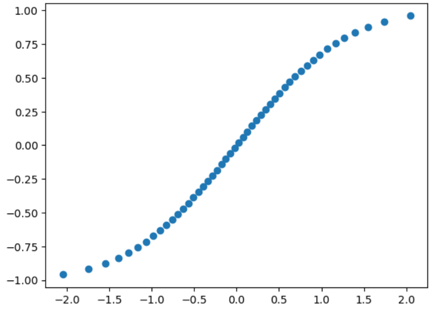
Figure 1: In blue: eigencoordinates for $n = 1000$, $k = 1$.
Red stars: Chebyshev polynomial of the first kind

Evenly Spaced Points Arranged According to a Distribution

Now take *n* evenly spaced points with a chosen value for ε. Then take the points and input them into either the inverse cumulative distribution function of the Gaussian or the inverse cumulative distribution function of the exponential distribution.

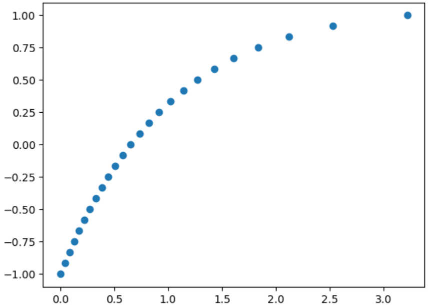
Evenly Spaced Points Arranged According to a Distribution

Inverse CDF of the Gaussian distribution is:



Evenly Spaced Points Arranged According to a Distribution

Inverse CDF of exponential distribution is:



Eigenvectors for points arranged according to a Distribution

We normalize the eigenvectors by dividing each eigenvector by the maximum element of the vector, so that the elements within each eigenvector range from -1 to 1 or 0 to 1.

Eigencoordinates picture generated from n evenly spaced points selected according to the Gaussian distribution

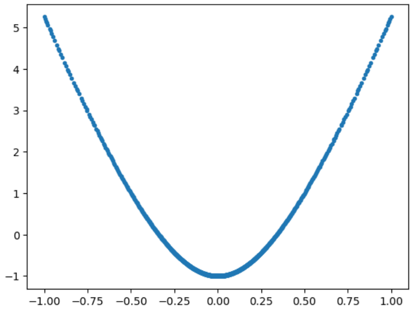


Figure 2: Evenly spaced points selected according to the Gaussian Distribution with $n = 5000, \varepsilon = .1$

Eigencoordinates picture generated from n evenly spaced points selected according to the exponential distribution

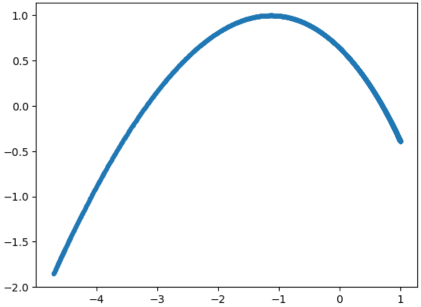


Figure 3: Evenly spaced points selected according to the Exponential Distribution with $n = 5000, \varepsilon = .1$

Eigencoordinates picture generated from n Uniformly Random Points selected according to the Gaussian Distribution

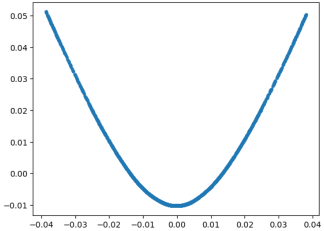


Figure 4: Uniformly Random Points arranged according to the Gaussian distribution with $n = 5000, \varepsilon = .1$

Eigencoordinates picture generated from n Uniformly Random Points selected according to the Exponential Distribution

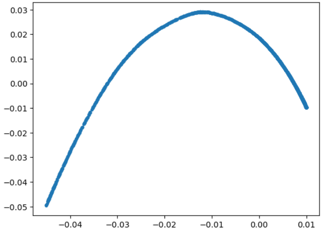


Figure 5: Uniformly Random Points arranged according to the exponential distribution with $n = 5000, \varepsilon = .1$

Random Points from an Interval

We begin by picking n randomly selected points from a distribution on a given interval. Distributions we looked at: uniform, gaussian, and exponential

About selecting points from Uniform Dist.

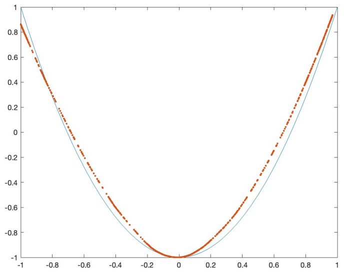
When randomly selecting points from the uniform distribution, we chose points on the interval $[-1, 1]$.

Conjecture

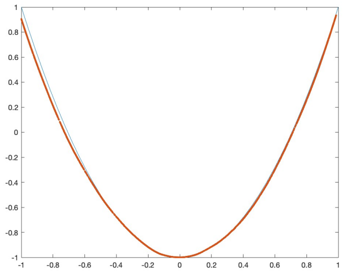
As $n \to \infty$ and $\varepsilon \to 0$, the Laplacian eigencoordinates of uniformly distributed random points converge to the Chebyshev polynomial.

We start with $n = 1000$ points and $\varepsilon = 0.05$, and plot the eigencoordinates against the quadratic Chebyshev polynomial (seen in Figure 6a).

Random Points Sampled directly from Uniform Dist.



(a) $n = 1000, \varepsilon = 0.05$



(b) $n = 5000, \varepsilon = 0.01$

Figure 6: Laplacian eigencoordinates plotted against the Chebyshev polynomial (blue) $T_2(x) = 2x^2 - 1$.

About selecting points from Gaussian Dist.

When randomly selecting points from the Gaussian distribution (also known as the Normal Distribution), in order to avoid the situation where outliers cause our graphs to be disconnected, we removed points outside of the interval $[-2, 2]$.

Random Points Sampled directly from Gaussian Dist.

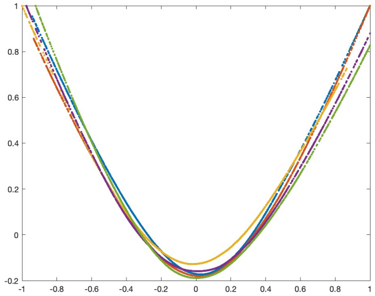


Figure 7: Eigenmap of normally distributed random points, $n = 5000, \varepsilon = 0.08$, 5 runs.

What is going on with the Gaussian eigencoordinates?

We expected the eigencoordinates of normally distributed random points to converge to the Hermite polynomials. However, when plotting the first and third or fourth eigenvectors, we see that the eigencoordinates loosely follow the Legendre polynomials.
Question: Is there theory behind this? Could this be happening perhaps because of our scaling/normalization?

Random Points Sampled directly from Gaussian dist. compared with polynomial

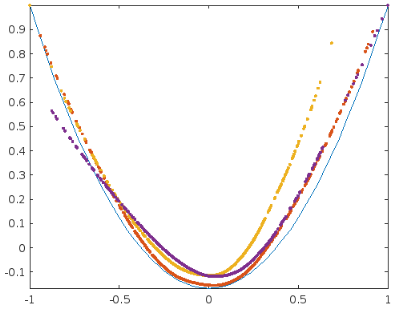


Figure 8: $n = 1000, \varepsilon = 0.3$, polynomial $= \frac{7}{6}x^2 - \frac{1}{6}$

Average of the Sum of Squared Errors for Different Values of Epsilon

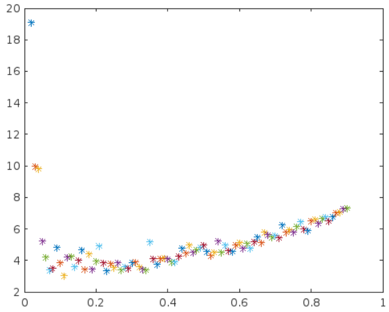


Figure 9: $n = 3000, \varepsilon = .02$ to $.9, 10$ runs

About selecting points from the Exponential Dist.

In order to avoid the situation where outliers cause our graphs to be disconnected, we removed points outside the interval $[0, 5]$.

Random Points Sampled directly from Exponential dist.

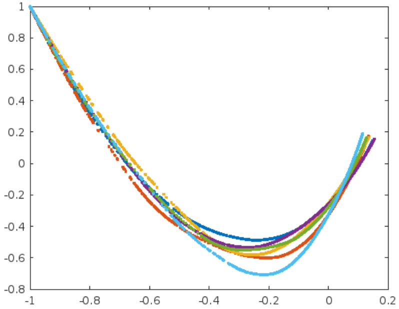


Figure 10: $n = 4000, \varepsilon = 0.1$

Point Selection Methodology

1. Define two variables, x_i and y_i.
2. Assign x_i, y_i values from $[0, 1]$ with a uniform random distribution.
3. Define the point $p_i = (x_i, y_i)$.
4. Complete 1-3 for $i = 1, 2, \ldots, n$.
5. Generate the Laplacian using the following edge weight formula:

$$w_{ij} = \begin{cases} 1 \text{ if } (x_i - x_j)^2 + (y_i - y_j)^2 \leq \varepsilon^2, i \neq j \\ 0 \text{ else} \end{cases}$$

Resulting Eigencoordinates

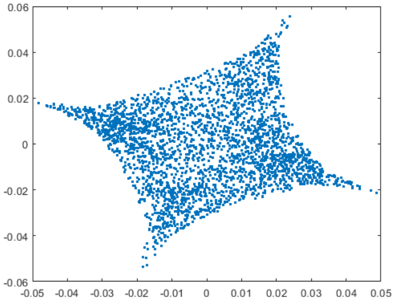


Figure 11: 2-dimensional eigencoordinates: $n = 3000, \varepsilon = 0.7$.

Resulting Eigencoordinates

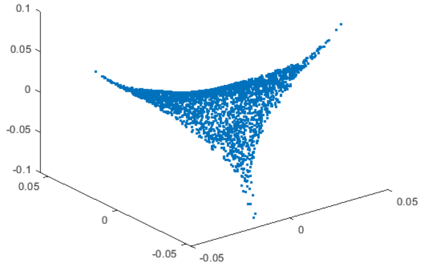


Figure 11: 3-dimensional eigencoordinates: $n = 3000, \varepsilon = 0.7$.

Point Selection Methodology

1. Define two variables, x_i and y_i.
2. Assign x_i, y_i values from $[0, 1]$ with a uniform random distribution.
3. Define the point $p_i = (x_i, y_i)$.

Point Selection Methodology

1. Complete 1-3 for $i = 1, 2, \ldots, n$.
2. Generate the Laplacian using the following edge weight formula:

$$w_{ij} = \begin{cases} 1 \text{ if } (\min(|x_i - x_j|, 1 - |x_i - x_j|))^2 \\ \qquad + (\min(|y_i - y_j|, 1 - |y_i - y_j|))^2 \leq \varepsilon^2, i \neq j \\ 0 \text{ else} \end{cases}$$

Point Selection Methodology

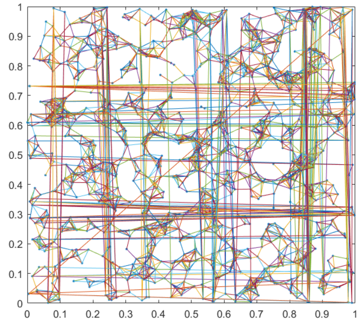
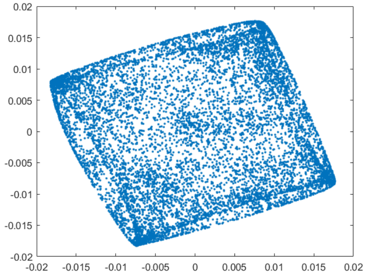
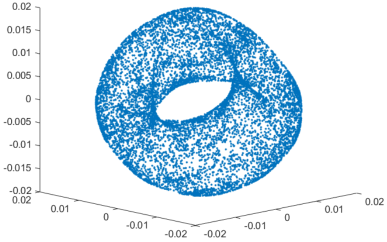


Figure 12: A plot of the resulting points and connections where $n = 1000$ and $\varepsilon = 0.05$.

Resulting Eigencoordinates



(a) Two-dimensional eigenmap.



(b) Three-dimensional eigenmap.

Figure 13: The 2-dimensional and 3-dimensional eigenmaps for a points sample from a torus where $n = 10000$ and $\varepsilon = 0.5$

Point Selection Methodology

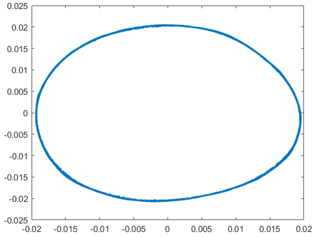
1. Define two variables, x_i and y_i.
2. Assign x_i values from $[0, 1]$ and y_i values from $[0, 5]$ with a uniform random distribution.
3. Define the point $p_i = (x_i, y_i)$.

Point Selection Methodology

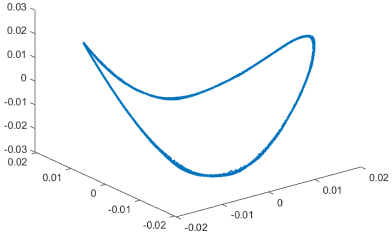
1. Complete 1-3 for $i = 1, 2, \ldots, n$.
2. Generate the Laplacian using the following edge weight formula:

$$w_{ij} = \begin{cases} 1 \text{ if } (\min(|x_i - x_j|, 1 - |x_i - x_j|))^2 \\ \qquad + (\min(|y_i - y_j|, 5 - |y_i - y_j|))^2 \leq \varepsilon^2, i \neq j \\ 0 \text{ else} \end{cases}$$

Resulting Eigencoordinates



(a) 2D eigencoordinates

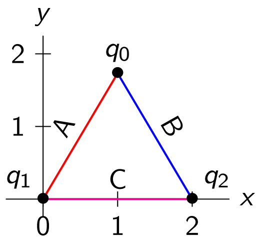


(b) 3D eigencoordinates

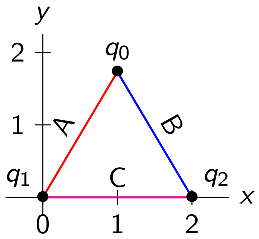
Figure 14: 5000 points uniform randomly chosen from a thin (1×5) torus where $\varepsilon = 0.1$

Point Selection Methodology

- Define an equilateral triangle on $\mathbb{R}^2$.



Point Selection Methodology



- Define a variable X to be uniformly random within $[0, 3]$. Based on the value of X, define x as follows

Point Selection Methodology

- Define a variable X to be uniformly random within $[0, 3]$. Based on the value of X, define x as follows

$$x = \begin{cases} X \text{ if } 0 \leq X < 2 \\ 2X - 4 \text{ if } 2 \leq X \leq 3 \end{cases}$$

Point Selection Methodology

- Use x to calculate y and set $p_i = (x, y)$ as the seed point.
- Define another variable a to be uniform random in $[0, 3]$ and the set of similarities,

$$F = \{F_i(p) := \frac{1}{2}(p - q_i) + q_i \mid i = 0, 1, 2\}$$

- For $j = 1, 2, \ldots m$ apply the following function to calculate p_m

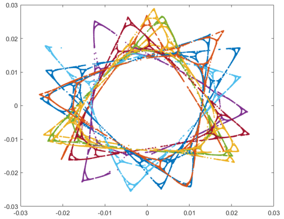
$$p_i, j = \begin{cases} F_0(p_i, j-1) \text{ if } 0 \leq a < 1 \\ F_1(p_i, j-1) \text{ if } 1 \leq a < 2 \\ F_2(p_i, j-1) \text{ otherwise} \end{cases}$$

Point Selection Methodology

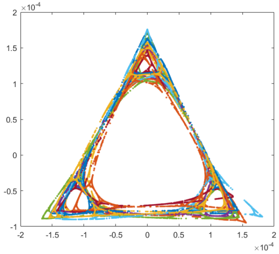
- Set $p_i = p_i, m$
- Complete these steps $i = 1, 2, \dots, n$.
- Generate the Laplacian using the following edge weight formula:

$$w_{ij} = \begin{cases} 1 \text{ if } (x_i - x_j)^2 + (y_i - y_j)^2 \leq \varepsilon^2, i \neq j \\ 0 \text{ else} \end{cases}$$

Resulting Eigencoordinates



(a) Sierpinski Eigenmaps plotted without any rotation.



(b) Sierpinski Eigenmaps plotted with rotation through a linear transformation.

Figure 15: These plots depict 10 runs of the code, plotted over each other. Each time $n = 5000$, $m = 15$, and $\varepsilon = 0.04$.

Weighted Laplacian

Now consider a Laplacian with different edge weights that has previously been investigated by Belkin and Niyogi Also, we get a variance σ^2 that can be changed that we represented as a. Then we times the variance a by 2 and it becomes $2\sigma^2$.

Weighted Laplacian

Let w_{ij} be defined as

$$w_{ij} = \begin{cases} e^{\frac{-\left|x_i - x_j\right|^2}{a}} \text{ if } i \neq j, a > 0 \\ 0 \text{ else} \end{cases}$$

We write $W = (w_{ij})_{i,j=0}^{n-1}$ and call it the adjacency matrix.

Weighted Laplacian

Now the adjacency matrix, the degree matrix and subsequently the Laplacian matrix, eigenvalues, and eigenvectors, can be constructed.

Eigenvectors

We normalize the eigenvectors by dividing each eigenvector by the minimum element of the vector, so that the elements within each eigenvector range from -1 to 1.

Weighted Graphs

We compare the eigencoordinates to Chebyshev polynomials using best fit curves. Polynomials for the quadratic, cubic, quartic, and quintic.

Weighted Graphs

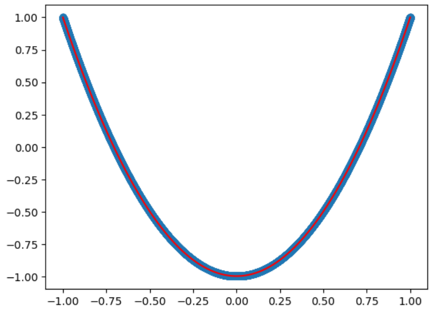


Figure 16: Best Fit Curve:
$y = 2.0036467x^2 + 8.0965123 * 10^{-11}x - 1.000014520$
Compare with $T_2(x) = 2x^2 - 1$

Weighted Graphs

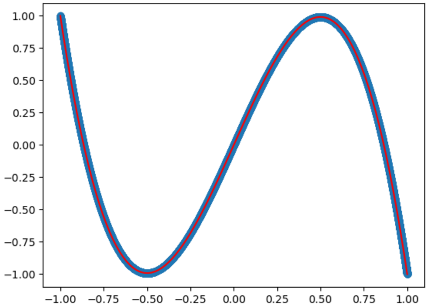


Figure 17: Best Fit Curve:
$y = 3.977825x^3 + 1.13000164 * 10^{-10}x^2 - 2.978068x - 2.52833309 * 10^{-11}$
Compare with $T_3(x) = 4x^3 - 3x$

Weighted Graphs

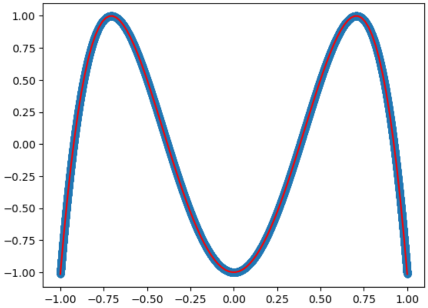


Figure 18: Best Fit Curve: $y = -8.0288566x^4 - 3.2939437 * 10^{-9}x^3 + 8.01490096x^2 + 1.36692 * 10^{-10}x + -1.00016946$

Compare with $T_4(x) = 8x^4 - 8x^2 + 1$

Weighted Graphs

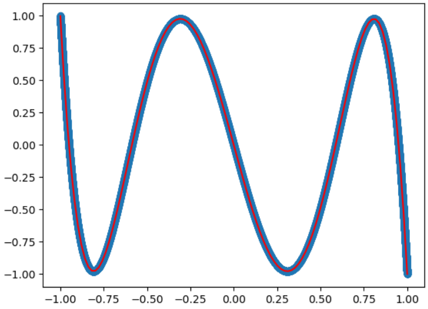


Figure 19: Best Fit Curve: $y = 15.7054676x^5 - 5.9303838 * 10^{-8}x^4 - 19.598658x^3 + 4.38873 * 10^{-8}x^2 + 4.89162x - 1.713723 * 10^{-9}$
Compare with $T_5(x) = 16x^5 - 20x^3 + 5x$