Spectral Methods for Dimensionality Reduction

Jay Paek
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, California
jpaek@ucsd.edu

***Abstract*—Dimensionality reduction is a pivotal technique in data science and machine learning, addressing the challenges posed by high-dimensional datasets. This report provides an in-depth analysis of key dimensionality reduction methods, focusing on spectral analysis and its applications. Specifically, we explore Laplacian Eigenmaps and Diffusion Maps, two powerful methods that leverage the spectral properties of graphs to uncover the intrinsic geometry of data. These techniques are particularly effective in preserving the local and global structures of datasets, enabling more efficient data representation and visualization.**

Laplacian Eigenmaps utilize the graph Laplacian to map high-dimensional data points to a lower-dimensional space while maintaining neighborhood relationships. Diffusion Maps extend this concept by employing a diffusion process to capture the connectivity and structure of the data manifold. The theoretical foundations, algorithmic implementations, and practical applications of these methods are discussed in detail.

The report also examines the integration of these spectral methods into deep learning frameworks, highlighting their role in enhancing feature extraction, reducing computational complexity, and improving model performance. Case studies and numerical experiments demonstrate the efficacy of Laplacian Eigenmaps and Diffusion Maps in various applications, including image recognition, clustering, and manifold learning. Through this comprehensive review, we aim to elucidate the significance of spectral dimensionality reduction techniques in advancing the capabilities of modern deep learning algorithms.

***Index Terms*—Manifold learning, dimensionality reduction, spectral analysis, Markov chains**

I. Introduction

Dimensionality reduction is a fundamental aspect of data science and machine learning, rooted in the need to simplify high-dimensional datasets without losing critical information. The history of dimensionality reduction techniques dates back to the early 20th century, with the advent of Principal Component Analysis (PCA) by Karl Pearson in 1901. PCA revolutionized the field by providing a method to reduce the number of variables in a dataset while preserving as much variance as possible. This technique laid the groundwork for many subsequent developments in statistical and computational methods for handling complex, high-dimensional data.

As data science evolved, so did the need for more sophisticated techniques capable of uncovering intricate structures within high-dimensional spaces. The curse of dimensionality, a term coined by Richard Bellman in the 1960s, highlighted the challenges posed by high-dimensional data, such as increased computational complexity and sparsity of data points. These challenges spurred the development of methods like Linear Discriminant Analysis (LDA), Multidimensional Scaling (MDS), and Isomap, each addressing specific aspects of dimensionality reduction and manifold learning. These techniques enabled researchers to project data into lower-dimensional spaces where patterns and relationships become more apparent and computationally manageable.

Manifold learning emerged as a powerful framework within the broader context of dimensionality reduction. It is based on the idea that high-dimensional data often lies on a lower-dimensional manifold embedded within the higher-dimensional space. Techniques such as Locally Linear Embedding (LLE), Laplacian Eigenmaps, and Diffusion Maps leverage this principle to uncover the intrinsic geometry of the data. These methods rely on spectral graph theory and nonlinear dimensionality reduction to preserve the local and global structures of the data, offering significant advantages over traditional linear methods in capturing complex, nonlinear relationships.

The implications of manifold learning for data science are profound. By revealing the underlying manifold structure, manifold learning techniques enhance the interpretability and visualization of high-dimensional data. They facilitate more efficient data compression, noise reduction, and feature extraction, which are critical for various applications, including image and speech recognition, bioinformatics, and social network analysis. Moreover, these techniques have been instrumental in advancing machine learning algorithms, particularly in deep learning, where they help in pretraining neural networks, reducing overfitting, and improving generalization by effectively capturing the essential features of the data.

The evolution of dimensionality reduction techniques has been pivotal in addressing the challenges of high-dimensional data in data science. From the foundational principles of PCA to the advanced methodologies of manifold learning, these techniques have transformed the way we analyze and interpret complex datasets. By preserving the intrinsic structure of data, manifold learning methods provide a robust framework for various applications, driving innovation and enhancing the capabilities of modern machine learning and deep learning algorithms. As data continues to grow in volume and complexity, the importance of dimensionality reduction and manifold learning will only increase, underscoring their critical role in the future of data science.

In the realm of data science and machine learning, handling high-dimensional data is a significant challenge. High-

dimensional datasets often contain many variables, which can complicate analysis due to the so-called "curse of dimensionality." Dimensionality reduction techniques are essential to simplify these datasets while retaining their intrinsic properties. This report delves into two prominent spectral methods for dimensionality reduction: Laplacian Eigenmaps and Diffusion Maps. Both methods aim to uncover the underlying manifold structure of high-dimensional data, thereby enabling more efficient and effective data analysis.

II. The Curse of Dimensionality

A. Understanding the Challenge

The curse of dimensionality refers to various phenomena that arise when analyzing and organizing data in high-dimensional spaces. These challenges include increased computational complexity, data sparsity, and the difficulty of visualizing and interpreting high-dimensional data. Traditional dimensionality reduction techniques, such as Principal Component Analysis (PCA), often fall short when dealing with complex nonlinear structures inherent in high-dimensional datasets.

B. Motivating Example 1: Uniform Sampling from an ℓ_∞-Ball

Consider the unit-norm ball defined by the ℓ_∞ norm in $\mathbb{R}^d$. Let $[\mathbf{x}]_i$ denote the ith entry of $\mathbf{x}$.

$$S = \{\mathbf{x} \in \mathbb{R}^d : \max_{1 \le i \le d} |[\mathbf{x}]_i| < 1\}$$

Sampling from this space under a uniform distribution, each coordinate is independent. What is the probability of sampling a point such that $|[\mathbf{x}]_i| < 0.99$ for all $1 \le i \le d$? Let $X : \mathcal{F} \to \mathbb{R}^d$ be a random vector. The probability is given by:

$$P(||X||_\infty < 0.99) = 0.99^d$$

As $d \to \infty$, $P(||X||_\infty < 0.99) \to 0$. This demonstrates how high-dimensional spaces can behave counterintuitively, with most points lying near the boundary of the space rather than its interior.

C. Motivating Example 2: Principle Component Analysis

Principal Component Analysis (PCA) is a widely used linear dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional space by projecting it onto the directions of maximum variance. This method is essential for reducing the complexity of datasets while preserving as much information as possible.

Consider a clustering task with two clusters. Each cluster has a sample mean and covariance matrix, denoted as $(\bar{\mathbf{x}}_1, \bar{\Sigma}_1)$ and $(\bar{\mathbf{x}}_2, \bar{\Sigma}_2)$, respectively. The objective is to determine the optimal projection direction that maximizes the separation between these clusters.

In PCA, the data is centered by subtracting the mean and then projected onto the eigenvectors of the covariance matrix, which correspond to the principal components. These principal components are the directions along which the variance of the data is maximized. Mathematically, given a dataset $X \in \mathbb{R}^{n \times d}$ with n samples and d dimensions, the covariance matrix is computed as:

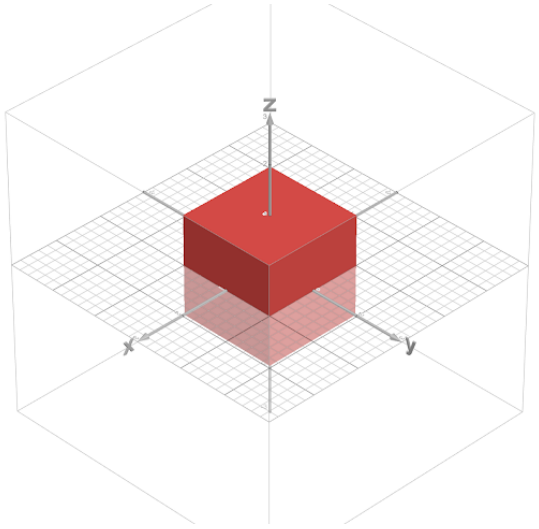

Fig. 1: ℓ_∞ ball in $\mathbb{R}^3$

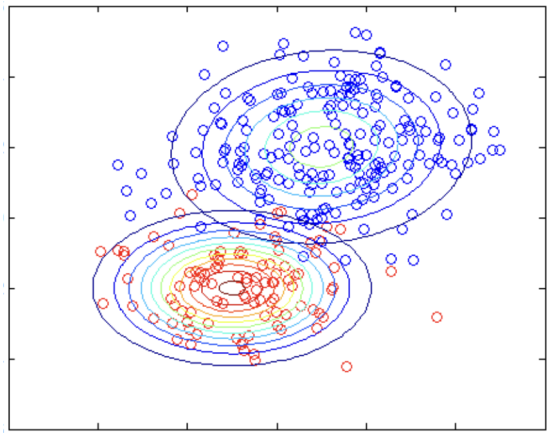

Fig. 2: 2D Gaussian mixture.

$$\Sigma = \frac{1}{n} X^T X$$

The eigenvectors of Σ represent the principal components, and the eigenvalues indicate the amount of variance captured by each principal component.

For a dataset with two clusters, the principal components can be used to identify the direction that best separates the clusters. This is particularly useful in classification tasks, where projecting the data onto the principal components can improve the performance of classifiers by reducing noise and highlighting the most discriminative features.

Consider a subset of the MNIST handwritten digit dataset representing the digit "3". By computing the sample mean $\bar{\mathbf{x}}$ and covariance matrix $\bar{\Sigma}$, and taking the eigenvectors associated with the largest eigenvalues, we can obtain the best features for classification. The principal components effectively capture the variations in the handwritten digits,

making it easier to distinguish between different samples of the digit "3".

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

Eigenvectors: $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_d\}$

Principal Coordinates: $X\mathbf{V}$

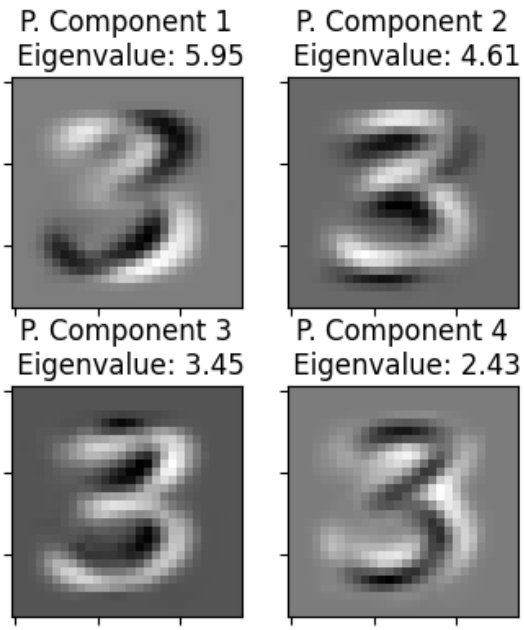

Fig. 3: Principal components for digit "3".

PCA is a useful technique to understand since the methods described in this paper are very similar in logic.

III. Main Algorithms

A. Laplacian Eigenmaps

Laplacian Eigenmaps is a method that uses the graph Laplacian to perform dimensionality reduction. It involves constructing a graph from the data points, where each point is connected to its nearest neighbors, and assigning weights to the edges based on a similarity function.

1) Graph Construction: Form an edge between $\mathbf{x}_i$ and $\mathbf{x}_j$ if they are "close" based on a chosen distance metric, either by neighborhood relation or k-nearest neighbors. For Laplacian Eigenmaps, the weight matrix W is defined as:

$$[W]_{i,j} = \begin{cases} \exp\left\{-\dfrac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\epsilon}\right\} & \mathbf{x}_i, \mathbf{x}_j \text{ connected} \\ 0 & \text{otherwise} \end{cases}$$

with a selected parameters ϵ. Notice that if we send $\epsilon \to \infty$, then the weights become simpler, assign 1 to connected points, and 0 to disconnected points.

2) Spectral Decomposition: Assuming that the graph is connected, construct a diagonal matrix D where $[D]_{i,i} = \sum_{j=1}^{n} W_{j,i}$. This matrix captures the degree of each data point. Make the graph Laplacian $L = D - W$. Solve the generalized eigenvalue problem:

$$L\psi = \lambda D\psi$$

Select the eigenvectors corresponding to the smallest nonzero eigenvalues to form the lower-dimensional embedding.

$$\mathbf{x}_i \mapsto [\psi_1(i), \ldots, \psi_m(i)]$$

B. Diffusion Maps

Diffusion Maps are based on the construction of a Markov chain on the dataset, where the transition probabilities reflect the local geometry of the data. The steps to construct a diffusion map is very similar to that of Laplacian eigenmaps, but the intuition behind the diffusion map is a random walk on the data.

1) Graph Construction: The transition probability matrix P is constructed as:

$$[P]_{i,j} = \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{\mathbf{z}\in X} K(\mathbf{x}_i, \mathbf{z})}$$

where $K(\mathbf{x}_i, \mathbf{x}_j)$ is a Gaussian kernel.

2) Spectral Decomposition: Solve the eigenvalue problem for the transition matrix P:

$$P\psi_k = \lambda_k \psi_k$$

The eigenvectors ψ_k with non-zerp eigenvalues λ_k provide the diffusion coordinates. The diffusion distance between two points $\mathbf{x}$ and $\mathbf{y}$ at time t is:

$$d_t(\mathbf{x}, \mathbf{y}) = \left(\sum_{l\geq 1} \lambda_l^{2t}(\psi_l(\mathbf{x}) - \psi_l(\mathbf{y}))^2\right)^{1/2}$$

It can be seen that both of these methods are very similar to that of PCA. In all of the methods, we perform spectral analysis on the matrix that encodes the similarity between features of the data.

IV. Theoretical Guarantees

In this section, we will go over one of the fundamental proofs that encapsulate the idea behind diffusion maps.

A. Spectral Analysis of the Markov Chain

The diffusion distance is a metric that captures the intrinsic geometry of the data by measuring the connectivity between points via a diffusion process. This section provides a detailed proof of how the diffusion distance can be estimated using the eigenvectors of the transition matrix constructed from the data. Diffusion distance is defined to be as follows:

$$\begin{aligned} d_t(x, y) &:= ||p_t(x, .) - p_t(y, .)||^2_{L^2(X, d\mu/\pi)} \\ &= \int_X (p_t(x, u) - p_t(y, u))^2 \frac{d\mu(u)}{\pi(u)} \end{aligned}$$

At first glance, this formulation may be daunting, but the integral term is just a summation over all of the datapoints since we are working in a finite measure space. Essentially, the diffusion distance characterized by parameter t is a metric that describes the probability of two datapoints "meeting" at another datapoint after traversing t steps of the random walk.

First of all, it is common to ask why it is important to examine the eigenvectors of the probability matrix. Let P be a transition probability matrix for a Markov chain

$$P_{i,j} = p(\mathbf{x}_i, \mathbf{x}_j)$$

$P^t_{i,j}$ is the probability of transition from the i to j in t steps. Since we want to examine the probabilities of a Markov chain after t steps, analyzing entries of repeated matrix multiplication requires the analysis of eigenvalues and eigenvectors.

B. Definition of the Diffusion Distance

Given a dataset $X = \{x_i\}_{i=1}^M$, we construct a weighted graph where each node represents a data point and the edges are weighted by a similarity measure $k(x_i, x_j)$. The transition matrix P is derived from these weights and represents the probabilities of transitioning between nodes in a random walk.

The diffusion distance between two points x_i and x_j at time t can be seen as a simpler formulation:

$$d_t(x_i, x_j) = \left(\sum_{x_k \in X} \left(p_t(x_i, x_k) - p_t(x_j, x_k)\right)^2 \pi(x_k) \right)^{\frac{1}{2}}$$

where $p_t(x_i, x_k)$ is the probability of transitioning from x_i to x_k in t steps, and ϕ_0 is the stationary distribution.

Before diving into the proof, it is first important to change the kernel to something that is more well-behaved. Define a new kernel

$$a(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l \geq 1} \lambda_l \phi_l(\mathbf{x}_i)\phi_l(\mathbf{x}_j)$$

and with this kernel arises an operator A. This operator will act on the entire metric space and restructure it in a sense. This can be directly connected to the shifted Dirac delta function as a kernel in signal processing or convolution kernels in convolutional neural networks. Without loss of generality, we often define kernel operators to be in the form

$$\int_X f(x)g(x,y)dy$$

However, it is realized that our operator is symmetric and compact i.e. it maps finite points to finite points. This means that we can perform spectral analysis on the operator and reformulate the diffusion distance in terms of the eigenfunctions. Spectral analysis on A is equivalent to spectral analysis on the probability matrix:

$$\frac{\sqrt{\pi(\mathbf{x}_i)}}{\sqrt{\pi(\mathbf{x}_j)}} p(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l \geq 1} \lambda_l \phi_l(\mathbf{x}_i)\phi_l(\mathbf{x}_j)$$

$$p(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l \geq 1} \lambda_l \frac{\phi_l(\mathbf{x}_i)}{\sqrt{\pi(\mathbf{x}_i)}} \Big(\sqrt{\pi(\mathbf{x}_j)}\phi_l(\mathbf{x}_j) \Big)$$

$$p^t(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l \geq 1} \lambda_l^t \frac{\phi_l(\mathbf{x}_i)}{\sqrt{\pi(\mathbf{x}_i)}} \Big(\sqrt{\pi(\mathbf{x}_j)}\phi_l(\mathbf{x}_j) \Big)$$

With this redefinition, we can alter the form of $D_t(x, y)$

$$\begin{aligned}
&= \int_X (p_t(x,u) - p_t(y,u))^2 \frac{d\mu(u)}{\pi(u)} \\
&= \int_X \Big(\sum_{l \geq 1} \lambda_l^t \frac{\phi_l(x)}{\sqrt{\pi(x)}} \Big(\sqrt{\pi(u)}\phi_l(u) \Big) \\
&- \sum_{l \geq 1} \lambda_l^t \frac{\phi_l(y)}{\sqrt{\pi(y)}} \Big(\sqrt{\pi(u)}\phi_l(u) \Big) \Big)^2 \frac{d\mu(u)}{\pi(u)} \\
&= \int_X \Big(\sum_{l \geq 1} \lambda_l^{2t} \Big(\frac{\phi_l(x)}{\sqrt{\pi(x)}} - \frac{\phi_l(y)}{\sqrt{\pi(y)}} \Big)^2 \Big(\sqrt{\pi(u)}\phi_l(u) \Big)^2 \frac{d\mu(u)}{\pi(u)} \\
&= \sum_{l \geq 1} \lambda_l^{2t} \Big(\frac{\phi_l(x)}{\sqrt{\pi(x)}} - \frac{\phi_l(y)}{\sqrt{\pi(y)}} \Big)^2 \int_X \Big(\phi_l(u) \Big)^2 d\mu(u) \\
&= \Big(\sum_{l \geq 1} \lambda_l^{2t} \Big(\psi_l(x) - \psi_l(y) \Big)^2
\end{aligned}$$

By the construction of ϕ_l, it is left as an exercise to the viewer to show that $\psi(x) = \frac{\phi(x)}{\sqrt{\pi(x)}}$. We can see that ϕ_l are eigenfunctions that send the datapoints to the diffusion coordinate space.

C. Approximation Using the Leading Eigenvectors

In practice, the eigenvalues decay rapidly, and the diffusion distance can be approximated using only the first d dominant eigenvalues and their corresponding eigenvectors. Thus, we have:

$$d_t(x_i, x_j) \approx \left(\sum_{\ell=1}^{d} \lambda_\ell^{2t} (\psi_\ell(x_i) - \psi_\ell(x_j))^2 \right)^{\frac{1}{2}}$$

This approximation effectively captures the significant structure of the data, as the leading eigenvectors correspond to the directions of maximum variance in the diffusion process.

The proof shows that the diffusion distance, which measures the intrinsic connectivity of the data, can be efficiently estimated using the eigenvectors and eigenvalues of the transition matrix. By focusing on the leading eigenvectors, we can reduce the computational complexity while preserving the essential geometric properties of the data. This approach is particularly useful in manifold learning and data representation, where understanding the low-dimensional structure is crucial.

V. Numerical Experiments

A. Laplacian Eigenmaps on the Swiss Roll Dataset

We apply Laplacian Eigenmaps to a Swiss roll dataset with 10,000 points (Fig 4). Using a Gaussian similarity kernel applied to the nearest 200 points, we can effectively recover the 2D manifold structure of the dataset (Fig 5).

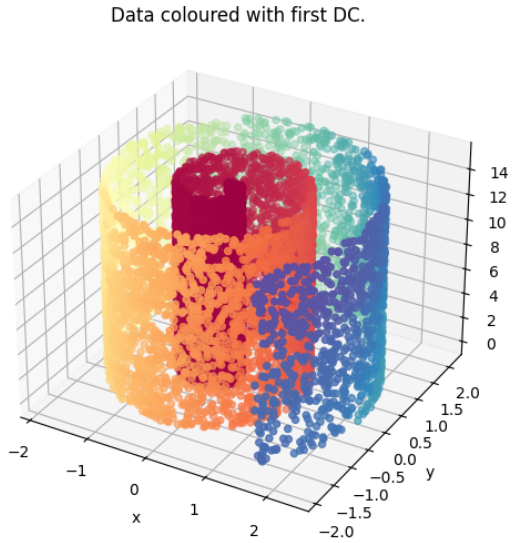

Fig. 4: The Swiss roll dataset

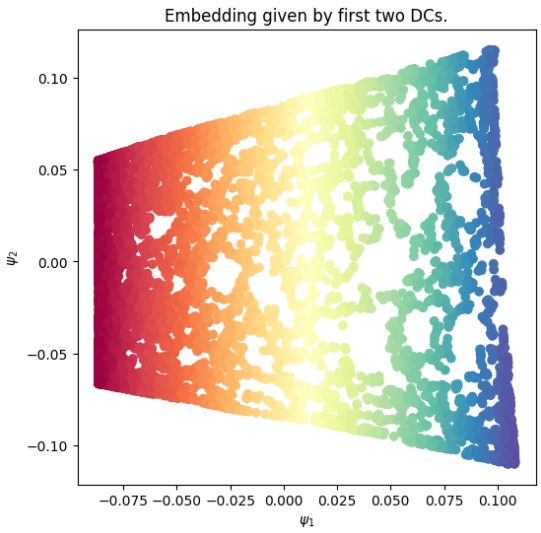

Fig. 5: Diffusion embedding

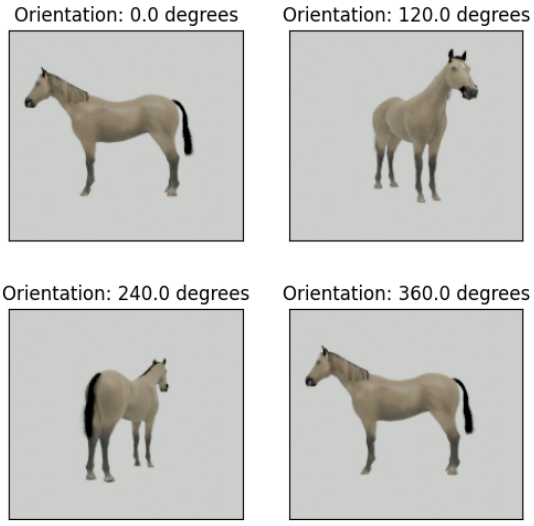

Fig. 6: The horse orientation dataset

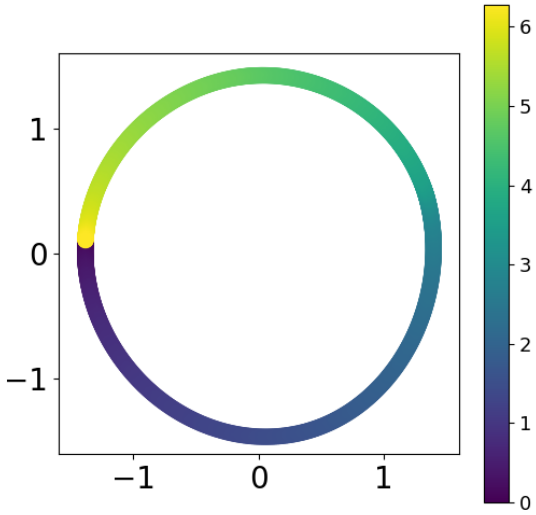

Fig. 7: Diffusion embedding (left bar is the orientation of horse in radians)

B. Diffusion Maps for Orientation Learning

For a dataset of 1,000 images of horses with varying orientations (Fig. 6), we use Diffusion Maps with a Gaussian kernel to construct the transition matrix. The resulting diffusion coordinates reveal the underlying orientation of the images (Fig. 7). Fig. 8 shows the distance of sample sample from the 0° orientation data point. The sine curve showing the change in diffusion distance shows that the embedding is effective.

However, it is important to note that these low dimensional embeddings are computational inefficient to compute. Its uses in real-time learning would not be adequate, but applications in learning for larger pre-made datasets would be useful.

VI. Applications to Deep Learning

Diffusion maps have a considerable application towards deep learning and Stochastic Gradient Descent (SGD). SGD is widely recognized for its computational efficiency in training deep neural networks, yet the reasons behind its superior performance compared to full batch gradient descent are not fully understood. Empirical observations indicate that the Hessian of the loss functions in over-parameterized networks often has many near-zero eigenvalues, suggesting that the optimization process effectively operates within a lower-dimensional subspace defined by the significant eigenvalues of the Hessian. This indicates that despite the high-dimensional parameter space, SGD dynamics are constrained to a lower-dimensional manifold.

Diffusion maps can help data scientists delve deeper into

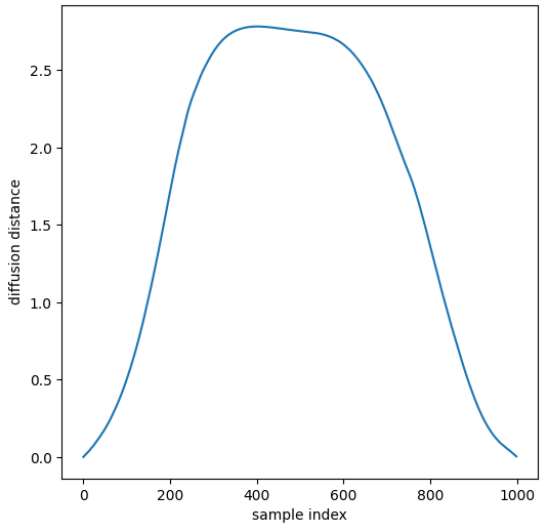

Fig. 8: Diffusion distance of sample points from orientation 0° data point

the geometry of the optimization landscape traced by SGD, this paper employs and facilitate the discovery of local low-dimensional representations of high-dimensional data by analyzing the data generated during the optimization process. This approach not only helps in understanding the SGD dynamics but also in identifying the slow variables and meta-stable states within the optimization landscape.

The analysis includes constructing a graph using the data points from SGD paths, with weights derived from diffusion kernels, and then performing spectral decomposition to obtain the principal components that capture the underlying geometry.

VII. Proof Sketches

A. Proof of Covariance Approximation (Equation 2.6)

First, note that the stochastic gradient $\nabla \tilde{f}_e(x)$ is an unbiased estimator of the full gradient $\nabla f(x)$:

$$E[\nabla \tilde{f}_e(x)] = E\left[\frac{1}{n}\sum_{i\in\Omega}\nabla f_i(x)\right] = \nabla f(x)$$

where $\Omega \subset \{1,\ldots,N\}$ is a random subset of data indices and n is the batch size.

The covariance of the stochastic gradient is given by:

$$C(x) = E[\nabla \tilde{f}_e(x)\nabla \tilde{f}_e(x)^T] - \nabla f(x)\nabla f(x)^T$$

Expanding the expectation term: $E\left[\nabla \tilde{f}_e(x)\nabla \tilde{f}_e(x)^T\right]$ is equal to

$$E\left[\left(\frac{1}{n}\sum_{i\in\Omega}\nabla f_i(x)\right)\left(\frac{1}{n}\sum_{i\in\Omega}\nabla f_i(x)\right)^T\right]$$

Since Ω is randomly sampled:

$$E\left[\nabla \tilde{f}_e(x)\nabla \tilde{f}_e(x)^T\right] = \frac{1}{n^2}\sum_{i,j}E[\nabla f_i(x)\nabla f_j(x)^T\delta_{ij}] \quad (1)$$

Considering δ_{ij}:

$$E[\delta_{ij}] = \begin{cases} \frac{n}{N} & \text{if } i = j \\ \frac{n(n-1)}{N(N-1)} & \text{if } i \neq j \end{cases}$$

Therefore (1) becomes:

$$= \frac{1}{Nn}\sum_i \nabla f_i(x)\nabla f_i(x)^T + \frac{n-1}{n(N-1)}\sum_{i\neq j}\nabla f_i(x)\nabla f_j(x)^T$$

And the covariance estimate becomes evident.

$$= \frac{N-n}{n(N-1)}\left(\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x)\nabla f_i(x)^T - \nabla f(x)\nabla f(x)^T\right)$$

This provides the desired covariance approximation used in the Mahalanobis distance for the diffusion maps.

VIII. Conclusion

Dimensionality reduction techniques are indispensable tools in the realm of data science and machine learning, facilitating the analysis and visualization of high-dimensional datasets. This report delved into the spectral methods of Laplacian Eigenmaps and Diffusion Maps, highlighting their capabilities in uncovering the intrinsic geometric structures of data. By leveraging the spectral properties of graphs, these methods efficiently preserve local and global data relationships, offering significant advantages over traditional linear techniques.

Laplacian Eigenmaps employ the graph Laplacian to project high-dimensional data into lower-dimensional spaces while maintaining neighborhood relationships, making it particularly effective for tasks where local structure is paramount. Diffusion Maps extend this approach by incorporating a diffusion process that captures the connectivity and manifold structure of the data, thus providing a robust framework for manifold learning and data representation. Through theoretical analysis and empirical demonstrations, this report illustrated how these spectral methods can be integrated into deep learning frameworks to enhance feature extraction, reduce computational complexity, and improve model performance.

In conclusion, the study of spectral methods for dimensionality reduction underscores their critical role in modern data science. As datasets continue to grow in size and complexity, the ability to simplify and interpret high-dimensional data remains a vital challenge. Spectral methods such as Laplacian Eigenmaps and Diffusion Maps offer powerful solutions, revealing the underlying structures that govern data distributions. The insights gained from this report not only advance our understanding of these techniques but also pave the way for their broader application in various domains, including image recognition, clustering, and beyond. As research progresses, further exploration of these methods will undoubtedly contribute to the ongoing evolution of data analysis and machine learning methodologies.

REFERENCES

[1] Belkin, M., & Niyogi, P. (2003). Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation*, 15(6), 1373-1396.

[2] Coifman, R. R., & Lafon, S. (2006). Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1), 5-30.

[3] Lederman, R., & Talmon, R. (2023). Manifold learning with density information. *Journal of Machine Learning Research*, 24, 1-50.

[4] Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4), 395-416.

[5] Fjellström, Carmina et al. (2022). Deep learning, stochastic gradient descent and diffusion maps*Statistics and Computing*