ABSTRACT

Title of dissertation: Extensions of Laplacian Eigenmaps for Manifold Learning

Avner Halevy, Doctor of Philosophy, 2011

Dissertation directed by: Professor John J. Benedetto
Professor Wojciech Czaja
Department of Mathematics

This thesis deals with the theory and practice of manifold learning, especially as they relate to the problem of classification. We begin with a well known algorithm, Laplacian Eigenmaps, and then proceed to extend it in two independent directions. First, we generalize this algorithm to allow for the use of partially labeled data, and establish the theoretical foundation of the resulting semi-supervised learning method. Second, we consider two ways of accelerating the most computationally intensive step of Laplacian Eigenmaps, the construction of an adjacency graph. Both of them produce high quality approximations, and we conclude by showing that they work well together to achieve a dramatic reduction in computational time.

Extensions of Laplacian Eigenmaps for Manifold Learning

by

Avner Halevy

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2011

Advisory Committee:
Professor John J. Benedetto, Chair
Professor Wojciech Czaja, Co-Chair
Professor Radu Balan
Professor Kasso Okoudjou
Professor Alexander Barg

Dedication

To my mother and father, always.

Acknowledgments

It is a pure pleasure to acknowledge the many people who have helped me make it to the end of this long journey.

Most students are lucky to find one decent advisor. For the last three years, I have been incredibly privileged to have three mathematical powerhouses lighting my way.

Truth be told, after getting my M.Sc., I was tired and homesick. It was John who stopped me from making what might have been the biggest mistake of my life, and warmly invited me to join the distinguished Norbert Wiener Center family. Since then, he has extended me nothing but encouraging words (and a rare slap on the wrist when I deserved one). With his broad vision and deep insight, he has inspired me continuously. With his boisterous laughter, he made my many hours in the lab next to his office amusing. And like a godfather and mother hen to all of us, he has seen me through, kindly and elegantly, on to the next stage of my life.

Wojtek has been my coach, guiding me along one hurdle – and, more recently, one comma – at a time. With his quick instincts, he has shown me how to move efficiently from theory to applications and back. Along the way he has challenged me with interesting questions and helped me evaluate and improve the fruit of my labor. He also taught me the truth about deadlines: They do not exist. And at the end of every meeting, he never forgot to ask how I am, as if to say, don't forget to breathe.

Radu has spent countless hours patiently discussing with me a wide spectrum

of ideas, from the purest results of time-frequency analysis and frame theory, down to the very lines of code bringing them to life; from a basic intuition, down to the last epsilon. Although the final structure of my thesis may not reflect it, if I have grown and matured mathematically along the way, I am indebted to Radu for much of it. He has never failed to make time for me when I asked for it, never taken more than seven minutes to reply to an email inquiry, and has consistently encouraged and stimulated my intellectual curiosity.

To all three of you, thanks for watching over me. I am enormously proud to have been your student.

I would like to thank the remaining two members of my committee: Kasso, who has been another warm and supportive member of the family all along, and Prof. Alexander Barg, the Dean's Representative, for taking the time to be involved in the culmination of my academic adventure. Which reminds me: Does everyone know the difference between involvement and commitment? It's like a ham-and-eggs breakfast: The chicken was involved, the pig was committed.

To Prof. Konstantina Trivisa, thank you for being the most positive person I have ever met. Your shining smile and jubilant Greek tones are a steady source of comfort. Everyone at AMSC is lucky to have you leading the way.

To Alverda McCoy, for being there, especially at the last minute, and helping me fight the monsters of bureaucracy, one battle at a time.

To Chris Flake and Kevin Duke, for their help, and for all the good times, especially in the lab.

To the National Geospatial-Intelligence Agency, for funding me for the last

three semesters.

To the University of Maryland, College Park, for its continuous support, financial and other. In particular, to the Department of Mathematics, for providing an exceptionally warm environment in which to work, teach, learn, and grow.

To my best friend, Amir, AAA.

To my sisters, Ronnie and Shelly, for always believing in me, and always being on the other side of the line when I needed support.

To my mother and father, Talma and Yechiam, founders, promoters, dreamers, heroes.

To my Mentor, for life.

Table of Contents

List of Figures

Chapter 1

Introduction

"We must make everything as simple as possible, but no simpler." -Albert Einstein

We are drowning in data. Scientists and engineers are more baffled than ever as we continue to accumulate data at a staggering rate and our capacity for processing and understanding it falls far behind. What we desperately need is intelligent and efficient ways for extracting meaning, structure, useful information and ultimately knowledge from the deluge of meaningless, raw data. Harmonic analysis, broadly speaking, deals with efficient representations of data, and thus offers potential tools for tackling the challenges resulting from an excess of data. In this context, this thesis deals with dimensionality reduction, which is concerned with the design and analysis of algorithms for removing redundancy in data and capturing its true information content.

The problem of analyzing intrinsically low-dimensional data lying in high-dimensional space is common in many areas of science and engineering, such as speech, image, and text processing, and is one of the main concerns of disciplines like machine learning, data mining, and pattern recognition. There is often a reason to believe that high-dimensional signals have only a few degrees of freedom, in which case the intrinsic information they contain may be captured more concisely. We thus look for a mapping from a high-dimensional to a low-dimensional space, which is

faithful to the data, in the sense that little or no information is lost along the way.

No such universally applicable mapping has been found thus far. Mappings that work well in some settings may fail miserably in others. Thus, while designing such a mapping, often a compromise must be made regarding its properties. First, there are purely theoretical considerations, such as determining precisely what type of information shall be preserved and what type may be lost. Second, there are computational considerations, such as the time and space complexity of the resulting implementation. We shall address both of these.

The first and arguably most crucial step in designing algorithms for finding concise descriptions of data is deciding how to model it. For example, in the recently developed theory of Compressed Sensing, the data is assumed to be sparse with respect to some basis, possibly in a transform domain. Manifold learning methods are based on the assumption that the signals lie on or near a low-dimensional manifold, and are designed to recover its structure efficiently. We shall use this latter model in most of what follows.

We now briefly describe the main contributions of this thesis, which are the following:

- Proof of the consistency of Schrödinger Eigenmaps
- A theoretical guarantee for the performance of Laplacian Eigenmaps with random projections, as well as empirical evidence of its advantages
- Empirical evidence of the advantages of a divide-and-conquer algorithm for the construction of the adjacency graph in the context of Laplacian Eigenmaps

- Integration of random projections with divide-and-conquer, optimization of user defined parameters, and evaluation of the performance of the resulting method

1.1 Schrödinger Eigenmaps

One popular manifold learning algorithm, known as Laplacian Eigenmaps (LE) [9], is based on the Laplace-Beltrami operator of the underlying manifold. Eigenfunctions of this operator give rise to smooth mappings, which may be used to produce a low-dimensional representation of the manifold that respects its geometry. However, in practical applications, the manifold is not directly accessible. Instead, we are only given a finite point cloud, which is assumed to have been sampled from the manifold. The first task is then to construct a discrete approximation of the Laplace-Beltrami operator in an efficient manner. Considerable work in this field has been devoted to proving convergence, in various senses, of the discrete approximation to its continuous counterpart, using methods of Riemannian geometry and functional analysis. Such analysis has been used successfully to establish LE on a firm foundation.

As we shall see, LE has been applied to various types of real data with some impressive results. However, this method is unsupervised in the sense that it does not allow for the use of labels associated with the data, when the ultimate goal is classification. In light of the prevalence and importance of this task, it is natural to consider methods for extending and generalizing LE in order to allow for some su-

pervision. Such an extension would leverage the proven capability of LE to identify underlying structure such as clusters in the data, and also allow for the introduction of prior knowledge. One such generalization, an algorithm called Schrödinger Eigenmaps [30], was recently designed and applied to biomedical images. In Chapter 3, we establish its consistency, i.e., we show that under certain mild conditions, the discrete approximations converge to well defined limits as the sample size increases.

1.2 Random Projections

The algorithms discussed above are nonlinear, as well as adaptive in the sense that a mapping is derived from the given data points. In contrast, the theory of Compressed Sensing relies on a linear, non adaptive method for reducing the dimension of signals modeled as sparse with respect to some dictionary. More specifically, random projections are used to stably embed sparse high-dimensional signals in a much lower-dimensional space, and efficient algorithms allow for their recovery. Recently, by replacing the model of sparsity with the model of a manifold, this technique has been extended to a wide class of signals [8]. Using these new and promising techniques, we have been able to reduce the computational cost of algorithms such as LE, without sacrificing accuracy.

LE, as well as many related Laplacian-based methods, relies on the construction of a weighted adjacency graph, which in turn requires the search for nearest neighbors in high-dimensional space. For large data sets, this search can easily become computationally prohibitive. We have shown that using random projections

to map the input data set to a low-dimensional space leads to a dramatic reduction in computational time, while, with high probability, essentially preserving quality of performance. In Chapter 4, we demonstrate this both theoretically and empirically.

1.3 Approximate Neighborhoods

The asymptotic complexity of the search for nearest neighbors depends on the dimension of the ambient space and the number of points. The use of random projections as described in the previous section targets the dimension. In an effort to deal with the number of points, we have implemented and tested an algorithm for the construction of approximate neighborhoods described in [23]. In this algorithm, the set of points is recursively divided into two smaller subsets, using spectral bisection, based on the inexpensive Lanczos algorithm. Once the size of the subset is small enough, the neighborhood graph is constructed using brute-force. Finally, the solutions to the subproblems are assembled in a straightforward manner to yield an approximate solution to the original problem. In chapter 5, we apply this algorithm to artificial as well as real data to show that it yields an approximation of high quality. Our main goal is to show that it works well in the context of LE.

1.4 Integration and Data Analysis

As we shall demonstrate overwhelmingly in Section 2.2.3, the main computational bottleneck in LE consists of the search for nearest neighbors. In Chapter 6, we show that we may readily combine random projections and approximate neigh-

borhoods, as described in the previous two sections, to simultaneously leverage the advantages of both in accelerating this search. Further, we apply the resulting algorithm in a systematic manner, for the purpose of classification, to standard datasets for which ground truth is available, evaluate its performance and show how to optimize the choice of user defined parameters.

Chapter 2

Background

2.1 Dimensionality Reduction

In recent years, our capacity to collect and store data has far outstripped our ability to analyze it. In problem domains as diverse as document retrieval, genetic sequence analysis, face recognition, and remote sensing, we are often given a large dataset in which each observation is associated with a large number of variables. We refer to the number of variables as the dimension of each observation, and in the given setting we say that the data lies in a high-dimensional space, where many common problems are hard or even impossible to solve with limited resources.

It is difficult for humans to visualize and interpret high-dimensional data. But perhaps more crucially, in an age when data analysis is increasingly automated, high-dimensional data is intractable whenever the restrictions of real world computation must be accounted for. The "curse of dimensionality" [15], as it has been called, refers to common situations where the complexity of a problem increases exponentially with the number of dimensions. One example, common to many problems of machine learning and information retrieval, is the search for nearest neighbors.

However, it often turns out that not all the variables are vital for understanding the underlying phenomenon, or that there is a high degree of redundancy in the information they represent. In this case, the structure and information content of

the data may be captured by a much smaller set of variables. For the purposes of processing, computation, communication, visualization, and analysis, it is highly desirable to determine the true, or intrinsic, dimension of the data, and find a faithful representation for it in a space of this lower dimension. Before proceeding into more technical details, let us consider a few intuitive examples to illustrate our goal.

First, consider a set of gray-scale images of a fixed object taken by a moving camera. If each image is $n \times n$, it yields a data point in $\mathbb{R}^{n^2}$. However, the intrinsic dimension of the space of all such images is simply the number of degrees of freedom of the camera, which is usually much lower.

As a more concrete example, consider the dataset in Figure 2.1, consisting of 221 images of the letter "A", each of size 32×32, which have been scaled and rotated. While the ambient space is $\mathbb{R}^{1024}$, the intrinsic dimensionality of the set is two, since only two variables are needed to produce each image. Using Manifold Sculpting, described in [38], to map the dataset to the two-dimensional plane, we obtain the result on the right side of the figure.

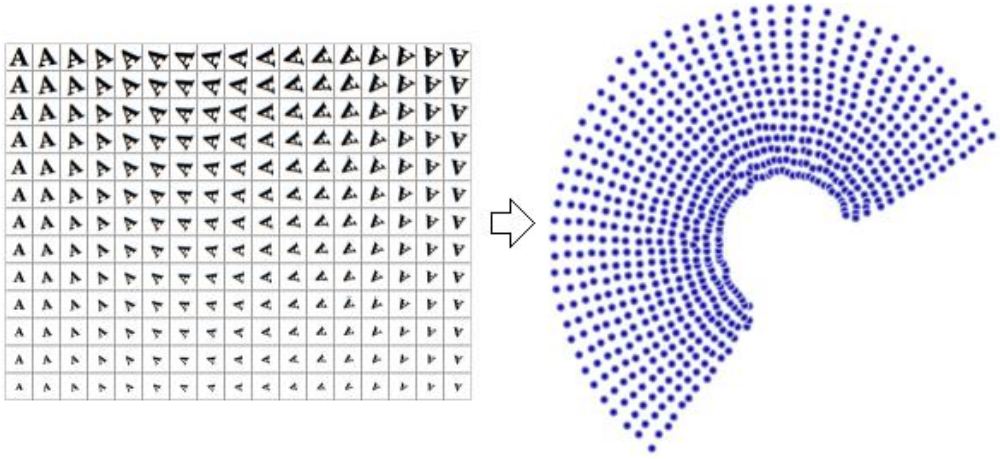


Figure 2.1: Dimensionality Reduction Using Manifold Sculpting

As a slightly more advanced example, taken from [44], we consider a dataset containing 1965 gray-scale face images taken from sequential frames of a short video. Each image has 20×28 pixels and is thus represented as a point in $\mathbb{R}^{560}$. Using Locality Preserving Projections (LPP), described in [44], the images are mapped into the two-dimensional plane. The result is shown in Figure 2.2, where representative images are shown next to some points. As can be seen, the facial expression and the viewing angle change smoothly. For example, the images shown at the bottom correspond to points along the path marked by the solid line. Thus, it appears that the structure in the high-dimensional ambient space has been well preserved in the representation space, whose dimension is drastically lower.

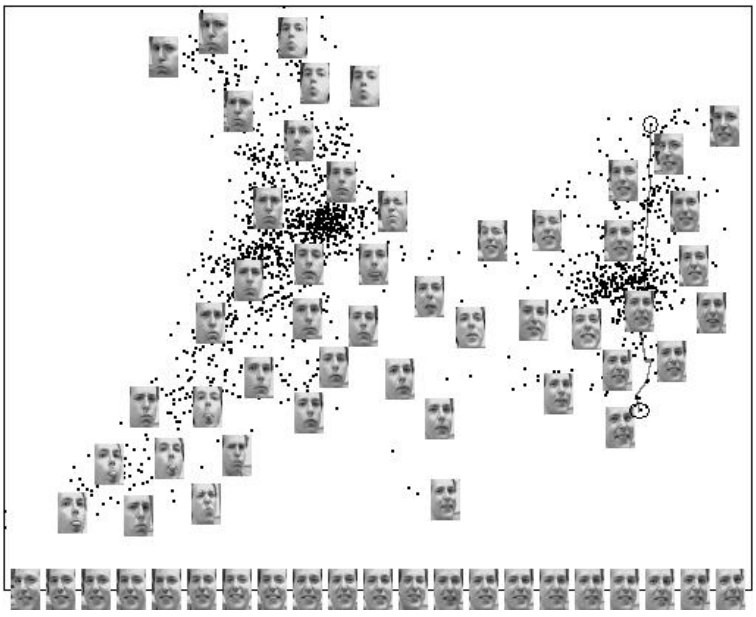


Figure 2.2: Face Recognition with LPP

As an example from another field, consider the following problem, taken from [54]. We are given 1047 articles from Science News, each belonging to one of eight fields. 2036 words are selected to capture the information content of this

particular body of documents. We then represent each article as a vector in $\mathbb{R}^{2036}$, whose ith coordinate is the frequency of the ith word. Using a Laplacian kernel (a common type of which we shall describe in the next section) and Diffusion Wavelets as described in [54], the dataset is then embedded in $\mathbb{R}^6$. Three of the coordinates are then plotted as shown in Figure 2.3. Each circle corresponds to an article whose field is represented by the circle's color. As the figure clearly shows, the structure of the dataset is again well preserved, in the sense that articles from the same field tend to cluster in the same region.

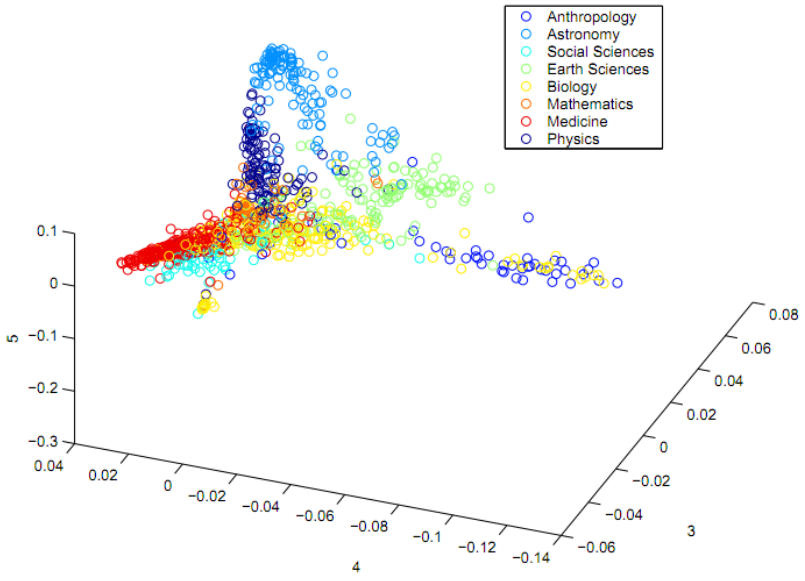
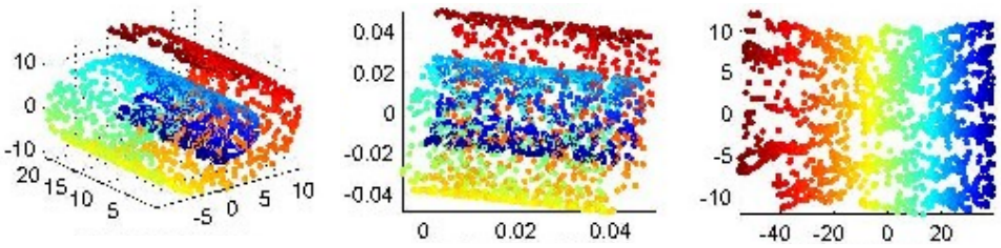


Figure 2.3: Document Classification with Diffusion Wavelets

Many methods of dimensionality reduction (DR) have been developed and successfully applied. An important distinction is made between the linear and the nonlinear techniques [53]. Linear methods assume that the data lies on or near a linear subspace of the high-dimensional ambient space. Nonlinear methods make no assumption of linearity and are designed to identify complex nonlinear manifolds as

well as linear ones. Linear methods have been used for a long time. For example, Principal Component Analysis (PCA) was invented in 1901 and is possibly still the most widely used method of DR. In contrast, nonlinear methods have been the focus of most research in recent times. Many of these methods clearly outperform linear methods when applied to common artificial examples such as the Swiss roll in Figure 2.4, where the performance of PCA is compared to that of ISOMAP, described in [59]. However, applications to real world data are often not as convincing [53].



(a) original set (b) PCA (c) ISOMAP

Figure 2.4: Reducing the Swiss roll

Methods of dimensionality reduction seek to recover the low-dimensional structure of the dataset, which is typically nonlinear, and thus not amenable to classical, linear methods. One class of methods is geometrically motivated and known as manifold learning. The problem can be stated as follows: Given a collection of n data points $x_1, x_2, \ldots, x_n$ (often called the "point cloud") in $\mathbb{R}^N$, which are assumed to lie on (or near) a manifold of dimension $K \ll N$, find n points $y_1, y_2, \ldots, y_n$ in $\mathbb{R}^K$ such that if x_i is mapped to y_i, then the new set of points is a faithful (low-dimensional) representation of the original point cloud. The sense in which the representation is faithful or optimal varies – different algorithms attempt to preserve

different geometric or topological properties of the manifold [49].

In general, nothing is known about the geometry or topology of the manifold, not even its dimension, and this makes manifold learning a notoriously ill-posed problem. Indeed, through any given set of points, it is easy to construct infinitely many different manifolds of varying dimension. The dimension of the manifold is crucial for common tasks relying on neighborhood relations. For example, in Figure 2.5, we are given a set of points on the left. If these points lie on the one-dimensional manifold in the middle, the blue triangle is closer to the red circle than the green square. However, if these points lie on the two-dimensional manifold on the right, the green square is closer. Most algorithms therefore require the intrinsic dimension as an input parameter, and several algorithms exist for estimating it for any given point cloud [48].

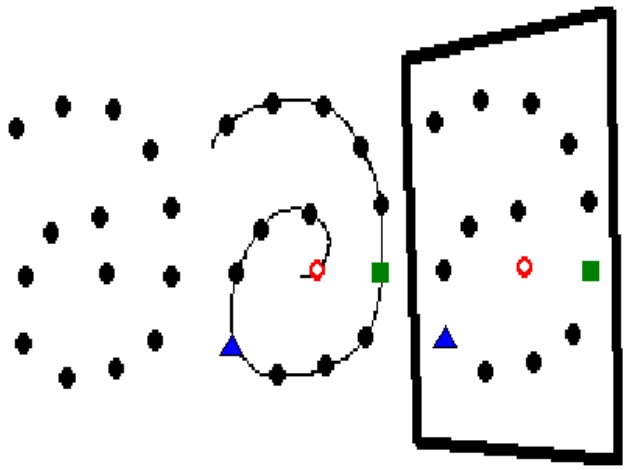


Figure 2.5: One and two-dimensional manifolds

A popular family of manifold learning algorithms is based on the Laplace-Beltrami operator of the underlying manifold. Eigenfunctions of this operator, which

constitute a basis for the space of square integrable functions on the manifold, give rise to optimally smooth mappings which preserve the geometry of the manifold. However, the manifold is not directly accessible. Instead we are only given a finite point cloud which is assumed to have been sampled from the manifold. The main task is then to construct a discrete approximation of the Laplace-Beltrami operator in an efficient manner. Such approximations are typically derived from a graph representation of the data, used in most state-of-the-art learning algorithms, such as Diffusion Wavelets [26], Locally Linear Embedding (LLE) [57], Hessian LLE [33], and Laplacian Eigenmaps [9].

2.2 Laplacian Eigenmaps

Laplacian Eigenmaps (LE) shall be our point of departure in much of what follows, so we now take the time to describe key features of the algorithm and the framework for its analysis. First described in [9], LE is an algorithm for nonlinear dimensionality reduction that attempts to preserve the local geometry of the manifold. It is designed to construct a discrete embedding that approximates the eigenfunctions of the Laplace-Beltrami operator. Using a neighborhood graph allows it to emphasize local information and makes it relatively insensitive to outliers and noise. Furthermore, the emphasis on local information implicitly emphasizes natural clusters in the data, in contrast to global methods such as ISOMAP [59]. In this sense it is closely related to spectral clustering algorithms developed earlier in the fields of machine learning and computer vision [46].

2.2.1 Historical Context

Connections between the spectrum of a manifold, defined as the spectrum of the Laplace-Beltrami operator on the manifold, and its geometry, have been studied for a long time and form the heart of a field known as spectral geometry. The earliest such result was produced in 1911, when Hermann Weyl showed that the dimension and volume of a bounded Euclidean domain are determined by its spectrum. Subsequently, many more geometric spectral invariants were established, leading mathematicians to wonder if in fact the geometry of a manifold is completely determined by its spectrum. The question was settled negatively in 1964 by John Milnor, who constructed two isospectral non-isometric manifolds. Nonetheless, Mark Kac continued to wonder if the answer might be positive for planar domains, and popularized the question in 1966 [47] when he asked: "Can one hear the shape of a drum?" But again the question was answered negatively in 1992 [41] by Gordon, Webb, and Wolpert, who constructed the counter example shown in Figure 2.6: these two regions have identical eigenvalues but different shapes.

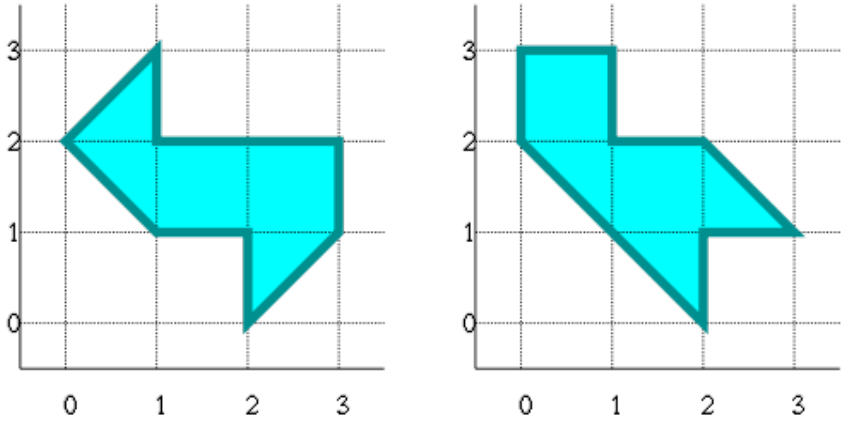


Figure 2.6: One cannot hear the shape of a drum

2.2.2 Eigenfunctions of the Laplace-Beltrami Operator

We would like to show what makes eigenfunctions of the Laplace-Beltrami operator optimal for producing an embedding that respects the manifold's geometry. This will provide motivation for LE, detailed below, at the heart of which is a discrete approximation to the Laplace-Beltrami operator and its eigenfunctions. Before proceeding, we note that while there is a natural analogy between the discrete and continuous settings, the formal and precise connections between the two were established only much later than the formulation of the algorithm, in a series of results we shall discuss in Section 2.2.6 [10, 12].

Let $\mathcal{M}$ be a smooth, compact, k-dimensional manifold embedded in $\mathbb{R}^d$. We are looking for a map $f : \mathcal{M} \to \mathbb{R}$ such that if $x, y \in \mathcal{M}$ are close, then $f(x)$ and $f(y)$ are also close, and assume that f is twice differentiable. Denote by $\nabla f(x)$ the gradient of f at x. Using basic differential geometry, it is straightforward to show that

$$|f(x) - f(y)| \leq \|\nabla f(x)\|\|x - y\| + o(\|x - y\|).$$

Thus, as long as $\|\nabla f(x)\|$ is small, points that are nearby on the manifold are mapped to points nearby on the real line. We therefore look for a map for which this quantity is small on average by finding

$$\underset{\|f\|_{L^2(\mathcal{M})}=1}{\arg\min} \int_{\mathcal{M}} \|\nabla f(x)\|^2. \tag{2.1}$$

Recalling the Laplace-Beltrami operator $\Delta_{\mathcal{M}}(f) = -\mathrm{div}(\nabla f)$, we use Stokes' The-

orem to conclude that

$$\int_{\mathcal{M}} \|\nabla f(x)\|^2 = \int_{\mathcal{M}} \Delta_{\mathcal{M}}(f) f,$$

and thus the functions that minimize (2.1) are the eigenfunctions of the Laplace-Beltrami operator corresponding to the smallest eigenvalues. Furthermore, the spectrum of $\Delta_{\mathcal{M}}$ on a compact manifold $\mathcal{M}$ is known to be discrete [56]. We discard the constant eigenfunction corresponding to eigenvalue 0 and use the next k eigenfunctions to produce an optimally smooth embedding into $\mathbb{R}^k$.

2.2.3 The Algorithm

Given points $x_1, x_2, \ldots, x_n$ in $\mathbb{R}^N$, LE consists of three main steps:

1. Constructing the adjacency graph: put an edge between nodes i and j if x_i and x_j are close. Precisely, given a parameter $m \in \mathbb{N}$, put an edge between nodes i and j if x_i is among the m nearest neighbors of x_j, or vice versa.

2. Computing edge weights: given a parameter $t > 0$, if nodes i and j are connected, set $W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$; otherwise, set $W_{ij} = 0$. Here and throughout we use the Euclidean norm.

3. Computing eigenmaps: Assume the graph constructed in Step 1 is connected, otherwise repeat the following for each connected component. Set $D_{ii} = \sum_j W_{ij}$, and let $L = D - W$. The matrix L is the Laplacian matrix; it is symmetric and positive semidefinite. Solve the generalized eigenvalue problem, $Lf = \lambda Df$ (since D is nonsingular, this problem can be reduced to a

standard eigenvalue problem). Let $f_0, f_1, \ldots, f_K$ be $K+1$ eigenvector solutions corresponding to the first eigenvalues $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_K$. Discard f_0 and use the next K eigenvectors to embed in K-dimensional Euclidean space using the map

$$x_i \to (f_1(i), f_2(i), \ldots, f_K(i)).$$

The asymptotic computational complexity of the algorithm is determined by steps 1 and 3. Assuming $m \ll n$, step 3 requires the solution of a sparse eigenvalue problem. If we denote by p the ratio of the number of nonzero elements to the total number of elements in the matrix, the cost of this step is $O(pn^2)$. However, running time is typically dominated by step 1, which requires a search for nearest neighbors. This fact is decisively illustrated with the following experiment. In Figure 2.7 we see a two-dimensional rectangle embedded in three-dimensional space. We sample the rectangle with increasing density and use LE to map it to the plane. In Figure 2.8, for each of five trials, the height of the bar shows the total running time, with the blue portion representing the time required for step 1. If the ambient dimension is N and the number of points is n, the cost of this step is $O(Nn^2)$. For a large, high-dimensional dataset, this computation may be infeasible. We shall attempt to mitigate this difficulty in Chapters 4 and 5.

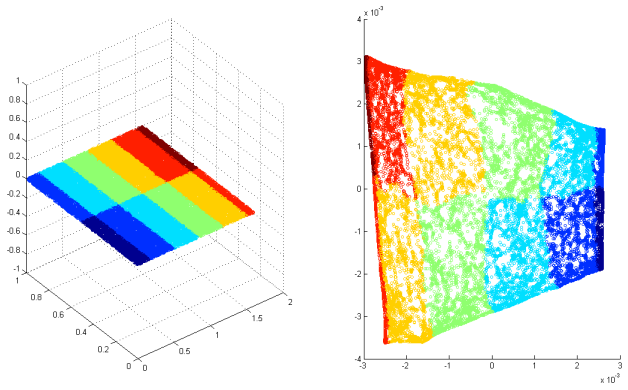


Figure 2.7: Mapping the square to the plane with LE

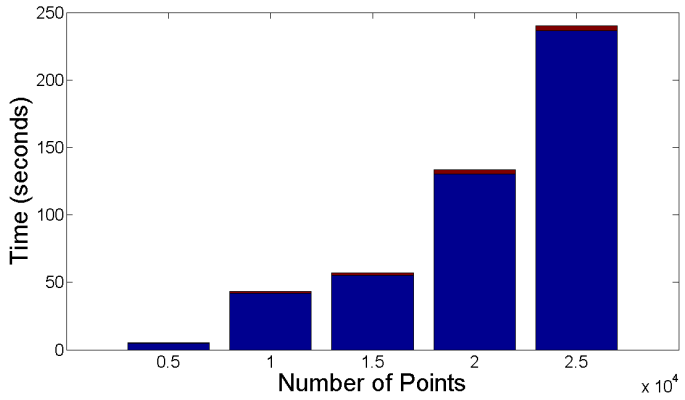


Figure 2.8: Constructing the nearest neighbor graph is a computational bottleneck

2.2.4 The Discrete Mapping

We would like to show that the embedding produced by LE preserves local information optimally in the sense explained below. Suppose the point $x_i \in \mathbb{R}^N$ is mapped to $y_i \in \mathbb{R}^K$. Now consider the following expression:

$$\sum_{i,j} \|y_i - y_j\|^2 W_{ij}.$$

According to the manner in which the weights W_{ij} were defined above, minimizing this expression would ensure that if the points x_i and x_j are close in $\mathbb{R}^N$, then y_i and y_j will be close in $\mathbb{R}^K$. Let $Y = [y_1\ y_2\ \ldots\ y_n]$ and note that

$$\begin{aligned}
\sum_{i,j} \|y_i - y_j\|^2 W_{ij} &= \sum_{i,j} \sum_{m=1}^{K} (y_{im} - y_{jm})^2 W_{ij} \\
&= \sum_{i,j,m} y_{im}^2 + y_{jm}^2 - 2 y_{im} y_{jm} W_{ij} \\
&= \sum_{i,m} y_{im}^2 \sum_{j} W_{ij} + \sum_{j,m} y_{jm}^2 \sum_{i} W_{ij} - 2 \sum_{i,j,m} y_{im} W_{ij} y_{jm} \\
&= \sum_{i,m} y_{im}^2 D_{ii} + \sum_{j,m} y_{jm}^2 D_{jj} - 2 \sum_{i,j,m} y_{im} W_{ij} y_{jm} \\
&= 2 \sum_{i,j,m} y_{im} D_{ij} y_{jm} - 2 \sum_{i,j,m} y_{im} W_{ij} y_{jm} \\
&= 2 \sum_{i,j,m} y_{im} (D_{ij} - W_{ij}) y_{jm} \\
&= 2 \sum_{i,j,m} y_{im} L_{ij} y_{jm} = 2\ \mathrm{tr}(Y^T L Y).
\end{aligned}$$

Therefore, we have

$$\arg\min_{Y^T L Y = I} \sum_{i,j} \|y_i - y_j\|^2 W_{ij} = \arg\min_{Y^T L Y = I} \mathrm{tr}(Y^T L Y).$$

It is well known that the latter expression is minimized by the matrix of eigenvectors corresponding to the smallest eigenvalues of the generalized eigenvalue problem $Ly = \lambda D y$.

As we shall see in the next section, LE performs reasonably well on some real datasets. However, we note that it is easy to construct simple counterexamples, i.e.,

manifolds that cannot be recovered by LE. In fact, in [40] the authors consider a two-dimensional rectangle and show that if the ratio of its sides is greater than two, the output of LE will be a one-dimensional manifold.

2.2.5 Examples

As a toy vision example taken from [9], consider binary images containing exactly one rectangular bar at a random location. Each image is 40×40 and thus corresponds to a point in $\mathbb{R}^{1600}$. We choose 1000 random images, 500 containing a horizontal bar and 500 containing a vertical bar. The space of all vertical bars is a two-dimensional manifold, as is the space of all horizontal bars. In Figure 2.9, the left panel shows what the rectangles look like. The middle panel shows a two-dimensional representation produced with LE, with blue dots corresponding to vertical bars and red plus signs corresponding to horizontal bars. We can see that the two components are well defined. In contrast, the right panel shows the result of using PCA to represent the same data.

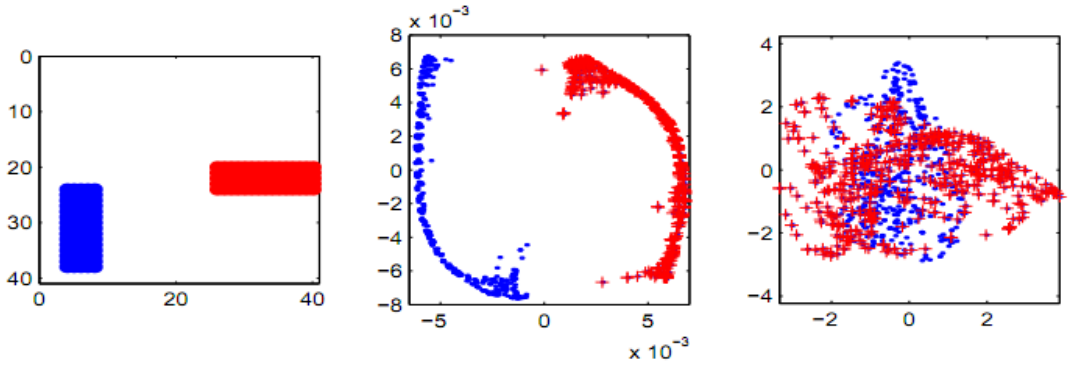


Figure 2.9: DR with toy rectangles: middle - LE, right - PCA

For an example with real data, consider the following data from the world of linguistics, also taken from [9]. Each of the 300 most frequent words in the Brown corpus, a collection of texts containing about 1 million words, is represented by a vector in $\mathbb{R}^{600}$. The first 300 components represent the frequency of each of the 300 words as the word's left neighbor, and the latter 300 components represent the frequency of each of the 300 words as the word's right neighbor. We use LE to obtain a two-dimensional representation of the data, as shown in Figure 2.10, where three clusters are denoted by arrows. These clusters are shown in detail in Figure 2.11, where we can see that words in the same cluster belong to the same syntactic category.

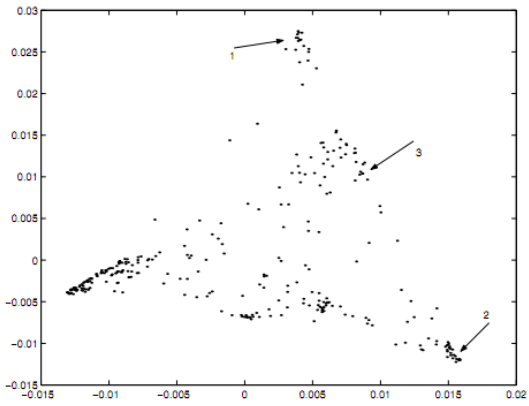


Figure 2.10: LE applied to 300 words

In our final example, taken from [44], we consider the task of recognizing the digits '0'-'9'. The dataset consists of 200 patterns per class, for a total of 2000 binary images. The images are pre-processed and then represented as points in $\mathbb{R}^{649}$. The data is then reduced to two dimensions using both LE and PCA. The results are presented in Figure 2.12, where we can see that LE performs much better

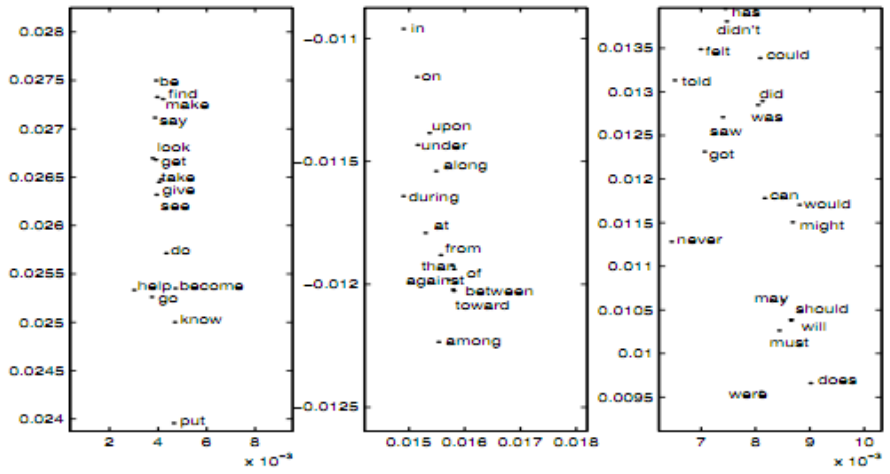


Figure 2.11: Clusters: infinitives of verbs (left), prepositions (middle), modal and auxiliary verbs (right)

at identifying clusters corresponding to the different classes.

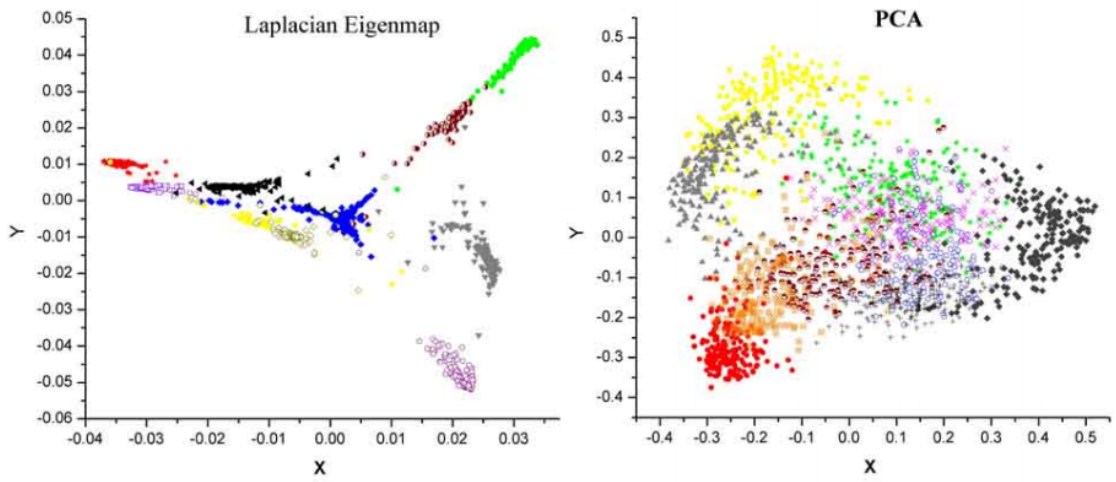


Figure 2.12: Comparison of LE and PCA for digit recognition

2.2.6 Convergence

The theoretical justification for LE relies on the connections between the graph Laplacian and its eigenvectors to the Laplace-Beltrami operator and its eigenfunc-

tions. We will rely on the formal results which make these precise, and now state two of them, as well as briefly mention the main ideas in their proofs.

The main idea behind the construction of the discrete Laplacian comes from the solution to the heat equation on the manifold. Let $f \in C(\mathcal{M})$. Recall that the solution to

$$\frac{\partial}{\partial t}u(x,t) + \Delta u(x,t) = 0, u(x,0) = f(x)$$

is given by

$$u(x,t) = H_t f(x), H_t(f)(x) = \int_{\mathcal{M}} h_t(x,y) f(y) d_\mu(y)$$

where, on $\mathbb{R}^k$, we have the heat kernel

$$h_t(x,y) = (4\pi t)^{-\frac{k}{2}} e^{-\frac{\|x-y\|^2}{4t}}.$$

From this we can readily derive the key to the discrete approximation as follows:

$$\begin{aligned} \Delta f(x) &= -\frac{\partial}{\partial t}u(x,t)\Big|_{t=0} = -\frac{\partial}{\partial t}H_t f(x)\Big|_{t=0} = \lim_{t\to 0}\frac{1}{t}(f(x) - H_t f(x)) \\ &= \lim_{t\to 0}\frac{1}{t}(4\pi t)^{-\frac{k}{2}}\left(\int_{\mathcal{M}} f(x)e^{-\frac{\|x-y\|^2}{4t}}dy - \int_{\mathcal{M}} f(y)e^{-\frac{\|x-y\|^2}{4t}}dy\right). \end{aligned}$$

Thus, we define $\hat{L}_{t,n} : C(\mathcal{M}) \to C(\mathcal{M})$ by

$$\hat{L}_{t,n}f(x) = \frac{1}{t}(4\pi t)^{-\frac{k}{2}}\left(\frac{1}{n}\sum_j f(x)e^{-\frac{\|x-x_j\|^2}{4t}} - \frac{1}{n}\sum_j f(x_j)e^{-\frac{\|x-x_j\|^2}{4t}}\right).$$

The Laplace-Beltrami operator on the manifold $\mathcal{M}$, $\Delta_{\mathcal{M}} : C^2(\mathcal{M}) \to L^2(\mathcal{M})$, is given by $\Delta_{\mathcal{M}}(f) = -\text{div}(\nabla f)$. We are now ready to state the first result, established

in [10].

Theorem 2.2.1 (Pointwise Convergence, Belkin and Niyogi) *Let the data points $x_1, x_2, \ldots, x_n$ be sampled independently from a uniform distribution on a smooth, compact manifold $\mathcal{M} \subset \mathbb{R}^N$. Let $\alpha > 0$, and set $t_n = (\frac{1}{n})^{\frac{1}{k+2+\alpha}}$. For $f \in C^\infty(\mathcal{M})$,*

$$\lim_{n\to\infty} \hat{L}_{t_n,n} f(x) = \frac{1}{vol(\mathcal{M})} \Delta_{\mathcal{M}} f(x) \text{ in probability,}$$

where vol($\mathcal{M}$) is the volume of the manifold with respect to the canonical measure.

In proving the above result, we face several issues. First, in general we do not know the exact form of the heat kernel on the manifold. Furthermore, we do not know the geodesic distance on the manifold, only the (Euclidean) distance in the ambient space.

The proof is broken into two parts, denoted by the following diagram:

$$\hat{L}_{t,n} f(p) \quad \underset{n\to\infty}{\longrightarrow} \quad L_t f(p) \quad \underset{t\to 0}{\longrightarrow} \quad \frac{1}{\text{vol}(\mathcal{M})} \Delta_{\mathcal{M}} f(p),$$

where the operator $L_t : \mathcal{L}^2(M) \to \mathcal{L}^2(M)$ is the functional approximation of $\Delta_{\mathcal{M}}$, given by

$$L_t(f)(x) = \frac{1}{(4\pi t)^{k/2} t} \left(\int_M f(x) e^{-\frac{\|x-y\|^2}{4t}} d\mu_y - \int_M f(y) e^{-\frac{\|x-y\|^2}{4t}} d\mu_y \right).$$

Proving the first part, the empirical approximation, is entirely straightforward. Since $\hat{L}_{t,n}$ is the empirical average of n i.i.d. random variables with expectation

$\mathbb{E}[\hat{L}_{t,n}f(p)] = L_t f(p)$, we can use a concentration inequality:

Lemma 2.2.2 (Hoeffding's Inequality) *Let $X_1, X_2, \ldots, X_n$ be i.i.d random variables such that $|X_i| \leq K$. Then,*

$$\mathbb{P}\left\{\left|\frac{\sum_i X_i}{n} - \mathbb{E}[X_i]\right| > \epsilon\right\} < 2\exp\left(-\frac{\epsilon^2 n}{2K^2}\right).$$

The proof of the second part, the functional approximation, is more intricate, and is divided into three steps:

1. Reduce the integral to a ball B in $\mathcal{M}$, i.e., show that

$$\lim_{t\to 0} L^t f(p) = \lim_{t\to 0} \frac{1}{(4\pi t)^{k/2} t} \int_B (f(p) - f(y)) e^{-\frac{\|p-y\|^2}{4t}} d\mu(y).$$

2. Apply a change of coordinates using the exponential map (see Figure 2.13) to rewrite the integral in $\mathbb{R}^k$.

3. Use a Taylor expansion to analyze the integral in $\mathbb{R}^k$.

Finally, we have the following result relating the eigenvectors of the discrete Laplacian to the eigenfunctions of the Laplace-Beltrami operator on the manifold [12].

Theorem 2.2.3 (Spectral Convergence, Belkin and Niyogi) *Let $\lambda_{t,n}^i$ and $e_{t,n}^i$ be the ith eigenvalue and corresponding eigenfunction of $\hat{L}_{t,n}$. Let λ_i and e_i be the ith eigenvalue and corresponding eigenfunction of $\Delta_{\mathcal{M}}$, respectively. Then there exists a sequence $t_n \to 0$ such that, in probability,*

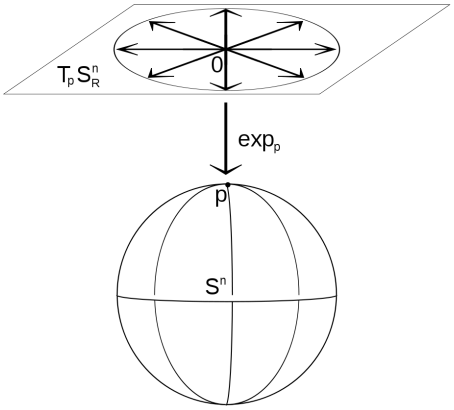


Figure 2.13: The Exponential Map

$$\lim_{n\to\infty} \lambda^i_{t_n,n} = \lambda_i \quad and \quad \lim_{n\to\infty} \|e^i_{t_n,n} - e_i\|_2 = 0.$$

2.3 Classification

In this thesis, we shall apply dimensionality reduction in the context of classification, which falls in the category of supervised machine learning. We now briefly describe the general framework for dealing with this problem, as well as a common algorithm that has been devised for its solution.

In the general setting of a classification task, we are given a training set $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ consisting of n input-output pairs. The input is typically a vector and the output is one of a finite number of classes. The algorithm analyzes the data in a phase called learning, and produces a classifier, also called a discriminant, which is a rule for assigning a category to any valid input vector. The ability to categorize correctly inputs that are not part of the training set is called generalization, and is a central goal of any learning algorithm. Often the

input to the classifier is pre-processed, i.e., the input vector is mapped to a new space of variables. Typically this is done either to improve accuracy or to speed up computation. Dimensionality reduction is one such type of pre-processing.

One of the simplest algorithms for classification is k *nearest neighbors* (kNN), where k is a user defined parameter, and an input is assigned to the class most common among its k nearest neighbors. This method is illustrated in Figure 2.14: If $k = 3$, the green circle, our test sample, is classified as a red triangle, whereas if $k = 5$ it is classified as a blue square. A particular instance of this algorithm is the case $k = 1$, called nearest neighbor classification. In what follows we use nearest neighbor classification, with one important modification: there is only one vector representing each class and its coordinates are the average of all the vectors in the training set that belong to that class. We now describe this procedure in detail, following the presentation in [45].

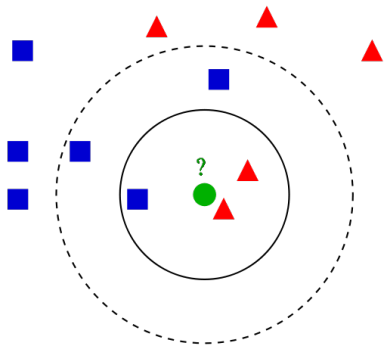


Figure 2.14: kNN classification

Let $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^D$ denote the set of input vectors in our training set. For each i, x_i belongs to exactly one of q classes $C_k, k = 1, 2, \ldots, q$. Assume the input vectors have already been pre-processed, e.g., mapped to a low-dimensional

space. Now construct a representative vector for each class by computing an average. That is, suppose $C_i = \{x_{i,j} : j = 1, 2, \ldots, q_i\}$, where q_i is the number of inputs in class C_i. Then the representative vector for this class is given by

$$\hat{x}_i = \frac{1}{q_i} \sum_{j=1}^{q_i} x_{i,j}.$$

Now suppose we are to classify a new pre-processed vector $x \in \mathbb{R}^D$. We do this by computing the angle between x and each representative vector $\hat{x}_i$, where this angle is given by

$$\theta_{x,\hat{x}_i} = \cos^{-1}\left(\frac{x^T \hat{x}_i}{\|x\|\|\hat{x}_i\|}\right),$$

and choosing the class that minimizes this angle.

2.4 Multispectral and Hyperspectral Data

We will test our algorithms with two kinds of data: multispectral and hyperspectral images. In both cases, the images are produced by sensors that measure the reflected energy in several bands of the electromagnetic spectrum. The idea is illustrated with the cube in Figure 2.15, taken from [2]. This type of imagery has applications in fields such as agriculture, mineralogy, geology, physics, surveillance, and medicine. It is designed to identify materials based on their spectral signature, or "fingerprint", across the electromagnetic spectrum.

Traditional images are produced by combining red, green, and blue images. In multispectral imaging, as the retinal images used in Chapter 3, up to about 10

different bands may be used. Hyperspectral sensors measure energy in narrower and much more numerous bands. Hyperspectral images, such as the land images used in Chapter 4, typically contain over 100 and as many as several hundred contiguous spectral bands, and are thus more sensitive to subtle variations in the reflected energy. Thus, for example, multispectral sensors may be used to detect a forest, while hyperspectral sensors may be used to detect different species of trees within the forest.

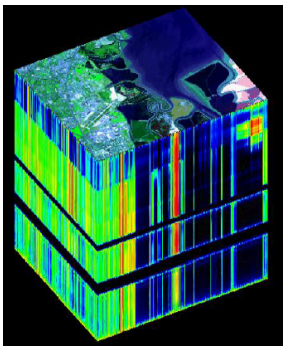


Figure 2.15: Hyperspectral data cube containing 224 bands [2]

In chapter 6, we will use two specific data sets, known as "Urban" and "Smith". Two features of these sets make them popular. First, these sets are publicly available [3–6] [60]. Second, ground truth for the purposes of material classification is available for some of the pixels, a fact that we shall later exploit in order to assess the performance of our algorithms. A color image of each of the regions used in these sets is shown in Figures 2.16 and 2.18. Each of the images in Urban is 307×307 and the set contains 161 bands. Each ground truth pixel belongs to one of 23 classes listed in Table 2.1. Each of the images in Smith is 679×944 and the set contains 110 bands. Each ground truth pixel belongs to one of 22 classes listed in Table 2.2.

Sample bands for each of the sets are shown in Figures 2.17 and 2.19.

Figure 2.16: Urban in color

Figure 2.17: Urban spectral bands 1,51,61, and 161

Table 2.1: Urban class names

1	AsphaltDrk
2	AsphaltLgt
3	Concrete01
4	VegPasture
5	VegGrass
6	VegTrees01a
7	Soil01
8	Soil02
9	Soil03Drk
10	Roof01Wal
11	Roof02A
12	Roof02BGvl
13	Roof03LgtGray
14	Roof04DrkBrn
15	Roof05AChurch
16	Roof06School
17	Roof07Bright
18	Roof08BlueGrn
19	TennisCrt
20	PoolWater
21	ShadedVeg
22	ShadedPav
23	VegTrees01b

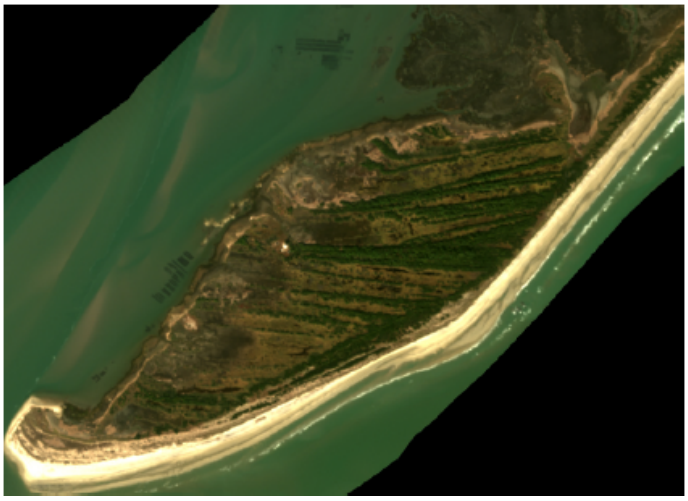


Figure 2.18: Smith in color

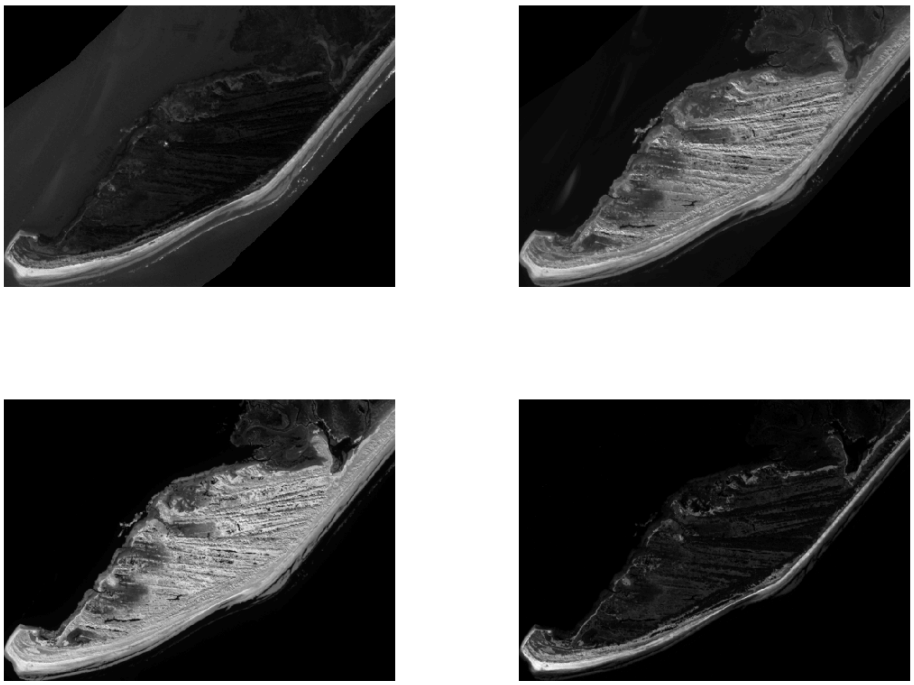


Figure 2.19: Smith spectral bands 1,21,51, and 110

Table 2.2: Smith class names

1	phrag
2	scirpus
3	juncus
4	patens
5	distichlis
6	andropogon
7	ammophila
8	mud
9	alterniora
10	borrichia
11	salicornia
12	iva
13	pine
14	hardwood
15	pond water
16	sand
17	wrack
18	myrica
19	seaoats
20	typha
21	water shore
22	submerged nets

Chapter 3

Schrödinger Eigenmaps

3.1 Introduction

Methods for dimensionality reduction are often applied to data for the purpose of classification. Furthermore, in this setting, we are often provided with some labeled examples. Laplacian Eigenmaps (LE) is an example of a completely unsupervised method, i.e., it does not allow for the use of labels, but only seeks to simplify and expose the underlying structure of the data. It seems natural to extend this method in order to take advantage of additional information and improve the classification process. One framework for doing this, based on regularization in reproducing kernel Hilbert space, is described in [13]. In this chapter, we describe an alternative method called Schrödinger Eigenmaps (SE) [30], based on the Schrödinger operator on the assumed underlying manifold. This simple extension of LE, in which we specify a potential on the adjacency graph, allows the user to take advantage of prior information, and enforce certain relations between points. In particular, certain points may be kept separate, i.e., classified into different classes, while other points may be identified with each other, i.e., classified into the same class. We shall now introduce the method, establish its asymptotic properties, and present the results of experiments with artificial as well as real data.

3.2 Laplacian Eigenmaps

For convenience, we recall the main steps of LE. Given n data points $x_1, x_2, \ldots, x_n$ sampled independently from a uniform distribution on a smooth, compact, K-dimensional manifold $\mathcal{M} \subset \mathbb{R}^d$:

1. Constructing the adjacency graph: Given a parameter $m \in \mathbb{N}$, put an edge between nodes i and j if x_i is among the m nearest neighbors of x_j or vice versa. Given a parameter $t > 0$, if nodes i and j are connected, set $W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$.

2. Constructing the Laplacian matrix: Set $D_{ii} = \sum_j W_{ij}$, and let $L = D - W$.

3. Computing the eigenmaps: Solve the generalized eigenvalue problem, $Lf = \lambda Df$. Let $f_0, f_1, \ldots, f_K$ be $K+1$ eigenvector solutions corresponding to the first eigenvalues $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_K$. Discard f_0 and use the next K eigenvectors to embed in K-dimensional Euclidean space using the map $x_i \to (f_1(i), f_2(i), \ldots, f_K(i))$.

3.3 Schrödinger Eigenmaps

Let $\mathcal{M} \subset \mathbb{R}^d$ be a smooth, compact, K-dimensional manifold. Let $v \in C^\infty(\mathcal{M})$ be a nonnegative potential defined on the manifold. Adding v to the Laplace-Beltrami operator $\Delta_\mathcal{M}$ results in the familiar Schrödinger operator. In order to construct the discrete analogue of $\Delta + v$ we add to L a nonnegative diagonal matrix V. We now repeat step 3 above with L replaced by $L + V$. We may further refine this scheme by considering $L + \alpha V$, where α is a user defined potential parameter

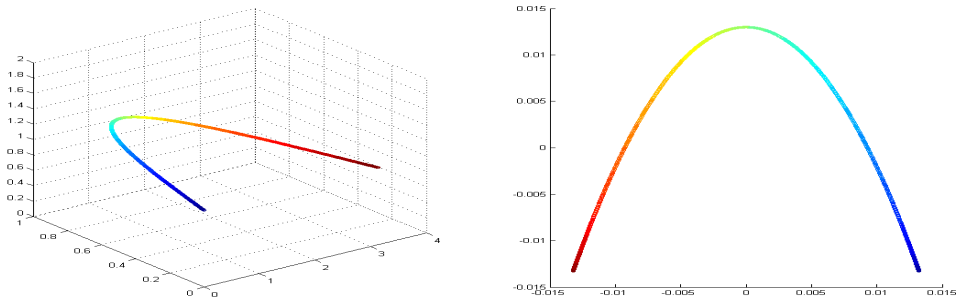


Figure 3.1: Two-dimensional arc recovered with Laplacian Eigenmaps

that allows for more control over the effect of using the potential.

To see this effect, consider an example where we use an artificial dataset. In Figure 3.1, we have a two-dimensional arc in three-dimensional space. When we use LE we obtain a perfect embedding in the plane. In Figure 3.2 we add a potential consisting of zeros along the diagonal except for a 1 in the position corresponding to the point in the middle of the arc. As we gradually increase the value of α, we force the labeled point, as well as its neighbors, to separate from the rest of the points. In Figure 3.3, we use a different matrix V to identify the endpoints of the arc.

For an example with real data, consider the retinal image in Figure 3.4, taken from [30]. By identifying the pixels denoted by the two arrows, the authors classify all the pixels into one class. In contrast, by separating these two pixels, they separate the remaining pixels into two distinct classes.

We would now like to show that the discrete operator converges to the continuous operator as the sample size increases. More precisely, we would like to prove two results that are analogous to the ones established in [10] and [12] for Laplacian Eigenmaps, namely, pointwise and spectral convergence.

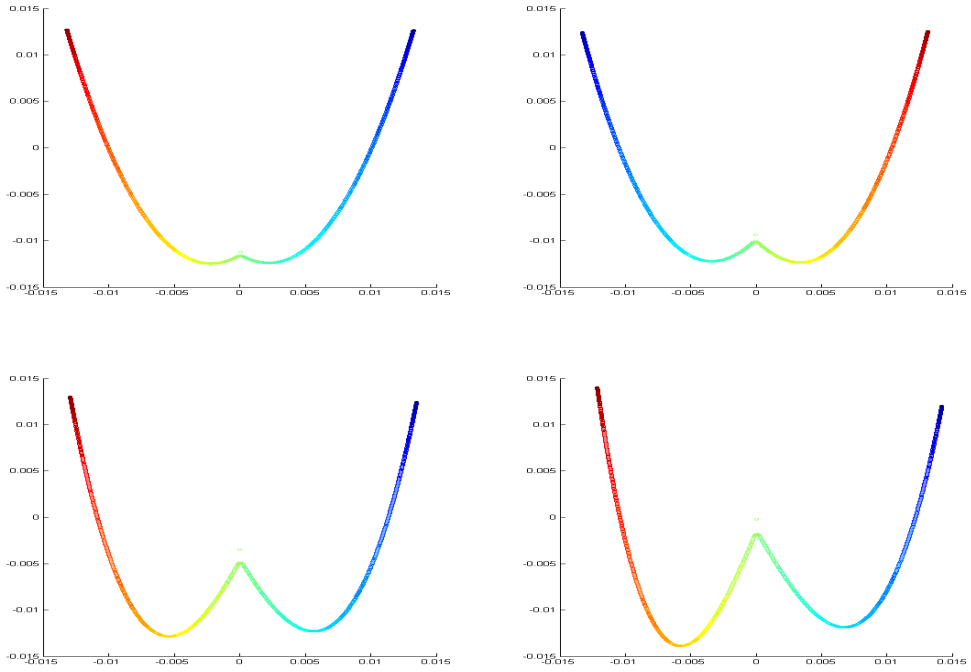


Figure 3.2: Separation with SE: V = diag(0,. . .,0,1,0,. . .,0), alpha = 0.05, 0.1, 0.5, 5

3.4 Pointwise Convergence

Given n data points $x_1, x_2, \ldots, x_n$ sampled independently from a uniform distribution on a smooth, compact, k-dimensional manifold $\mathcal{M} \subset \mathbb{R}^d$, define the operator $\hat{L}_{t,n} : C(\mathcal{M}) \to C(\mathcal{M})$ by

$$\hat{L}_{t,n}(f)(x) = \frac{1}{(4\pi t)^{k/2} t} \left(\frac{1}{n} \sum_j f(x) e^{-\frac{\|x-x_j\|^2}{4t}} - \frac{1}{n} \sum_j f(x_j) e^{-\frac{\|x-x_j\|^2}{4t}} \right).$$

Let $v \in C(\mathcal{M})$ be a potential defined on the manifold. For a point $x \in \mathcal{M}$, let $y_n(x) = \underset{x_1,x_2,\ldots,x_n}{\arg\min} \|x - x_i\|$ and define the operator $V_n : C(\mathcal{M}) \to C(\mathcal{M})$ by $V_n f(x) = v(y_n(x)) f(x)$.

Theorem 3.4.1 (Pointwise Convergence) *Let $\alpha > 0$, and set $t_n = (\frac{1}{n})^{\frac{1}{k+2+\alpha}}$. For*

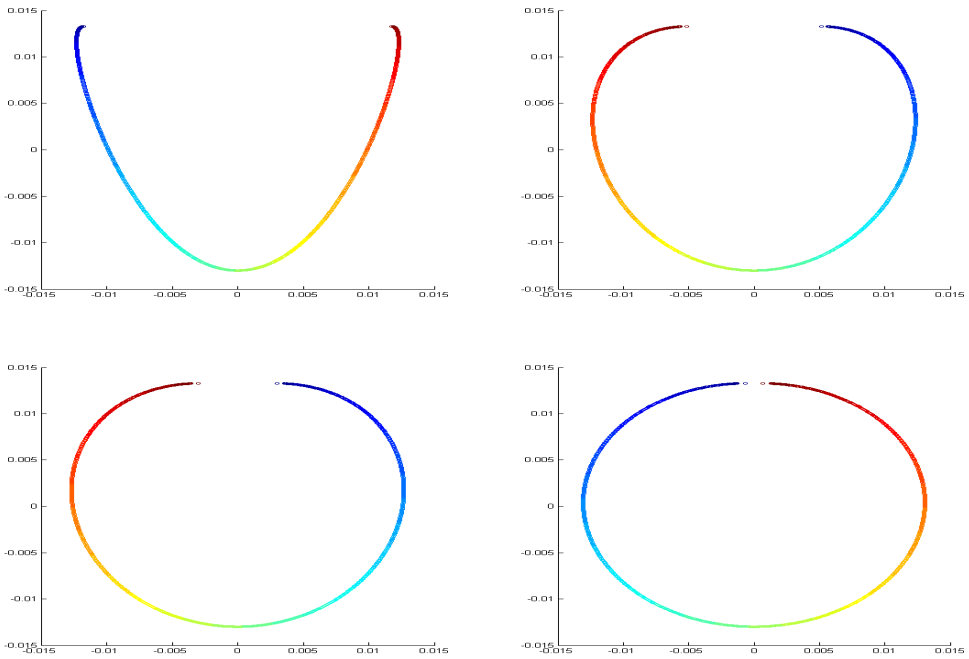


Figure 3.3: Identification with SE: alpha = 0.01, 0.05, 0.1, 1

$f \in C^\infty(\mathcal{M})$,

$$\lim_{n\to\infty} \hat{L}_{t_n,n} f(x) + V_n f(x) = \frac{1}{\mu(\mathcal{M})} \Delta_{\mathcal{M}} f(x) + v(x) f(x) \text{ in probability,}$$

where $\mu(\mathcal{M})$ is the volume of the manifold with respect to the canonical measure μ.

Proof. We are given a point $x \in \mathcal{M}$, a function $f \in C^\infty(\mathcal{M})$, and $\epsilon > 0$. Denote by $\mathbb{P}$ the probability measure obtained by normalizing μ. First, note that

$$\mathbb{P}\left\{\left|\hat{L}_{t_n,n} f(x) + V_n f(x) - (\frac{1}{\mu(\mathcal{M})} \Delta_{\mathcal{M}} f(x) + v(x) f(x))\right| > \epsilon\right\} \leq$$
$$\mathbb{P}\left\{\left|\hat{L}_{t_n,n} f(x) - \frac{1}{\mu(\mathcal{M})} \Delta_{\mathcal{M}} f(x)\right| > \frac{\epsilon}{2}\right\} + \mathbb{P}\left\{|V_n f(x) - v(x) f(x)| > \frac{\epsilon}{2}\right\}.$$

In [10], the first term on the right side of the inequality was shown to be arbitrarily

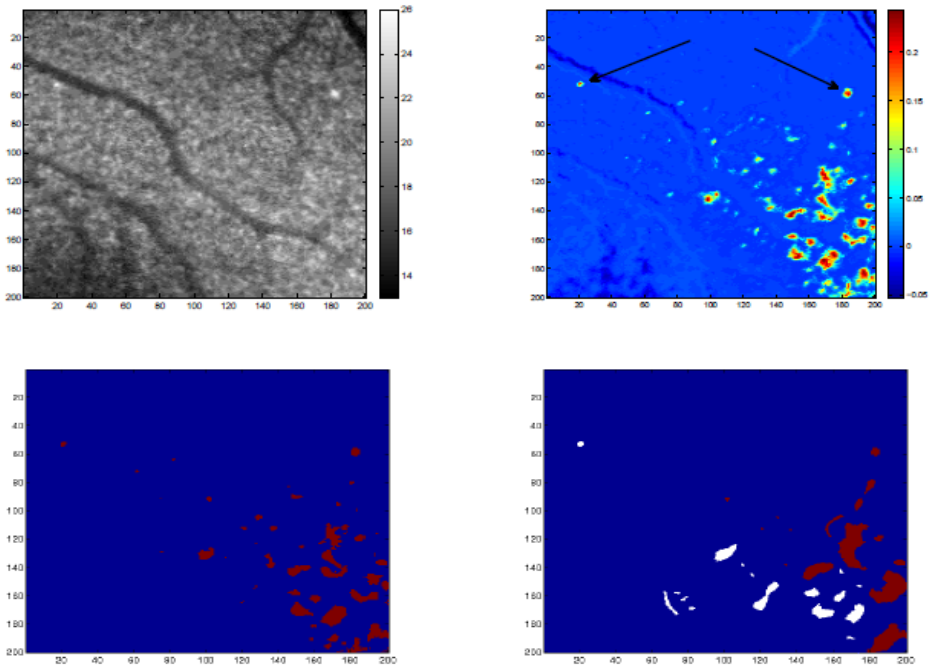


Figure 3.4: Application: classification of retinal Images

small for sufficiently large n. Thus, it suffices to bound the right one. Since f is continuous on a compact manifold, there exists $C > 0$ such that $|f(z)| \leq C$ for all $z \in \mathcal{M}$. Also, since v is continuous, there exists $\delta > 0$ such that if $d(x, z) < \delta$ then $|v(x) - v(z)| < \frac{\epsilon}{C}$, where d is the geodesic distance on the manifold. If at least one of the n points is within δ of x,

$$\begin{aligned} |V_n f(x) - v(x)f(x)| &= |v(y_n(x))f(x) - v(x)f(x)| \\ &= |f(x)||v(y_n(x)) - v(x)| \\ &\leq C\frac{\epsilon}{C} = \epsilon. \end{aligned}$$

Since the points $x_1, x_2, \ldots, x_n$ are sampled independently from a uniform distribution on a manifold with volume $\mu(\mathcal{M})$, the probability that none of the n points is

within δ of x is $\left(\frac{\mu(\mathcal{M})-\mu(B_\delta(x))}{\mu(\mathcal{M})}\right)^n$. Therefore,

$$\mathbb{P}\left\{|V_n f(x) - v(x)f(x)| > \epsilon\right\} \leq \left(\frac{\mu(\mathcal{M}) - \mu(B_\delta(x))}{\mu(\mathcal{M})}\right)^n \underset{n\to\infty}{\longrightarrow} 0.$$

□

3.5 Spectral Convergence

We now wish to establish a stronger convergence result that ensures the validity of Schrödinger Eigenmaps. We wish to show that as the sample size n increases, and the parameter t decreases, the eigenvectors of the discrete Schrödinger operator converge to the eigenfunctions of the continuous Schrödinger operator, in a sense that will be made precise.

Given n data points $x_1, x_2, \ldots, x_n$ sampled from a manifold $\mathcal{M} \subset \mathbb{R}^d$, and a parameter $t > 0$, the unnormalized graph Laplacian (see [9]) is constructed as before, but now we stress the dependence on n and t:

- Construct a symmetric weight matrix $(W_{t,n})_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$
- Construct the diagonal matrix $(D_{t,n})_{ii} = \sum_j (W_{t,n})_{ij}$
- Let $L_{t,n} = D_{t,n} - W_{t,n}$

Given a function f defined on the data points, we have

$$L_{t,n}(f)(x_i) = \sum_j f(x_i)e^{-\frac{\|x_i - x_j\|^2}{4t}} - \sum_j f(x_j)e^{-\frac{\|x_i - x_j\|^2}{4t}}.$$

We may extend this construction to functions defined on the entire manifold, and normalize to obtain the operator $\widehat{L}_{t,n} : C(\mathcal{M}) \to C(\mathcal{M})$ defined by

$$\widehat{L}_{t,n}(f)(x) = \frac{1}{(4\pi t)^{k/2} t} \left(\frac{1}{n} \sum_j f(x) e^{-\frac{\|x-x_j\|^2}{4t}} - \frac{1}{n} \sum_j f(x_j) e^{-\frac{\|x-x_j\|^2}{4t}} \right).$$

Recall the Laplace-Beltrami operator on the manifold $\mathcal{M}$, $\Delta_{\mathcal{M}} : C^2(\mathcal{M}) \to L^2(\mathcal{M})$, given by $\Delta_{\mathcal{M}}(f) = -\text{div}(\nabla f)$. For convenience, we repeat the following result from Section 2.2.6, established in [12].

Theorem 3.5.1 (Spectral Convergence, Belkin and Niyogi) *Let $\lambda^i_{t,n}$ and $e^i_{t,n}$ be the ith eigenvalue and corresponding eigenfunction of $\widehat{L}_{t,n}$. Let λ_i and e_i be the ith eigenvalue and corresponding eigenfunction of $\Delta_{\mathcal{M}}$. Then there exists a sequence $t_n \to 0$, such that*

$$\lim_{n\to\infty} \lambda^i_{t_n,n} = \lambda_i \quad \textit{and} \quad \lim_{n\to\infty} \|e^i_{t_n,n} - e_i\|_2 = 0 \quad \textit{in probability.}$$

Let $v \in C(\mathcal{M})$ be a bounded potential defined on the manifold, and let $\hat{V}_n = \text{diag}(v(x_1), \ldots, v(x_n))$. Given a function f defined on the data points, we have

$$\hat{V}_n(f)(x_i) = v(x_i) f(x_i).$$

Define the multiplication operator $V : C(\mathcal{M}) \to C(\mathcal{M})$ by $Vf(x) = v(x)f(x)$. We define the extension V_n of $\hat{V}_n$ to functions defined on the entire manifold by setting $V_n = V$, i.e., V_n is independent of n. We wish to generalize Theorem 3.5.1

to Schrödinger operators of the form $S_{t,n} = L_{t,n} + V_n$. Precisely, we would like to establish the following.

Theorem 3.5.2 (Spectral Convergence of Schrödinger Operator) *Let $\lambda^i_{t,n}$ and $e^i_{t,n}$ be the ith eigenvalue and corresponding eigenfunction of $S_{t,n}$. Let λ_i and e_i be the ith eigenvalue and corresponding eigenfunction of $\Delta_{\mathcal{M}} + V$. Then there exists a sequence $t_n \to 0$, such that*

$$\lim_{n\to\infty} \lambda^i_{t_n,n} = \lambda_i \quad \textit{and} \quad \lim_{n\to\infty} \|e^i_{t_n,n} - e_i\|_2 = 0 \quad \textit{in probability.}$$

The argument splits into two parts, represented by the following diagram:

$$\mathrm{Eig}S_{t,n} \quad \overrightarrow{n\to\infty} \quad \mathrm{Eig}S_t \quad \overrightarrow{t\to 0} \quad \mathrm{Eig}\Delta_{\mathcal{M}} + V.$$

The first convergence is in probability, and the second one is deterministic. However, we observe that V_n does not depend on the parameter t. Thus it suffices to show the first convergence. To do so, we adapt the arguments in [52] to establish convergence of the eigenvalues and eigenvectors of the empirical Schrödinger operator, under certain conditions.

3.5.1 Overview of Method

We are given n data points $x_1, x_2, \ldots, x_n$ sampled from a manifold $\mathcal{M} \subset \mathbb{R}^d$. An eigenvector $v^n = (v_1, v_2, \ldots, v_n)$ of the discrete operator (matrix) $\hat{S}_n$ can be thought of as a function g_n on these points, by defining $g_n(x_i) = v_i$. As $n \to \infty$,

we would like these functions to converge to a continuous function on the entire manifold. The technical difficulty in proving this convergence is that, for different n, the vectors v^n live in different spaces. To overcome this, let the restriction operator $\rho_n : \mathcal{M} \to \mathbb{R}^n$ map a function to its values on the first n points, that is, $\rho_n f = (f(x_1), f(x_2), \ldots, f(x_n))$. We will construct operators S_n and S on $\mathcal{M}$ with corresponding eigenfunctions f_n and f such that $v^n = \rho_n f_n$ and f_n converges to f.

3.5.2 Preliminaries

From now on we assume the following:

Assumption 3.5.3 *$\mathcal{M} \subset \mathbb{R}^d$ is a compact manifold, $\mathcal{B}$ is the Borel σ-algebra on $\mathcal{M}$, and P is a probability measure on $(\mathcal{M}, \mathcal{B})$. The points $X_1, X_2, \ldots, X_n$ are sampled independently according to P. The similarity function $k : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is symmetric, continuous, and bounded away from 0 by a positive constant. The potential $v \in C(\mathcal{M})$ is bounded above.*

Definition 3.5.4 *(Convergence of Operators) Let $(E, \|\cdot\|_E)$ be a Banach space, B its unit ball, and $(S_n)_n$ a sequence of bounded linear operators on E.*

- $(S_n)_n$ converges pointwise *to S, denoted $S_n \xrightarrow{p} S$, if $\|S_n x - Sx\|_E \to 0$ for all $x \in E$.*
- $(S_n)_n$ converges compactly *to S, denoted $S_n \xrightarrow{c} S$, if it converges pointwise and if for every sequence $(x_n)_n$ in B, the sequence $(S - S_n)x_n$ is relatively compact (has compact closure) in $(E, \|\cdot\|_E)$.*

Compact convergence will ensure the convergence of spectral properties in the following sense, see [22].

Proposition 3.5.5 (Perturbation) *Let $(E, \|\cdot\|_E)$ be a Banach space and $(T_n)_n$ and T bounded linear operators on E with $T_n \xrightarrow{c} T$. Let $\lambda \in \sigma(T)$ be an isolated eigenvalue with finite multiplicity m, and $M \subset C$ an open neighborhood of λ such that $\sigma(T) \cap M = \{\lambda\}$. Then:*

1. *Convergence of eigenvalues: There exists an $N \in \mathbb{N}$ such that, for all $n > N$, the set $\sigma(T_n) \cap M$ is an isolated part of $\sigma(T_n)$, consists of at most m different eigenvalues, and their multiplicities sum up to m. Moreover, the sequence of sets $\sigma(T_n) \cap M$ converges to the set $\{\lambda\}$ in the sense that every sequence $(\lambda_n)_n$ with $\lambda_n \in \sigma(T_n) \cap M$ satisfies $\lim \lambda_n = \lambda$.*

2. *Convergence of spectral projections: Let Pr be the spectral projection of T corresponding to λ, and for $n > N$ let Pr_n be the spectral projection of T_n corresponding to $\sigma(T_n) \cap M$. Then $Pr_n \xrightarrow{p} Pr$.*

3. *Convergence of eigenvectors: If, additionally, λ is a simple eigenvalue, then there exists some $N \in \mathbb{N}$ such that, for all $n > N$, the sets $\sigma(T_n) \cap M$ consist of a simple eigenvalue λ_n. The corresponding eigenfunctions f_n converge up to a change of sign, i.e., there exists a sequence $(a_n)_n$ of signs $a_n \in \{-1, 1\}$ such that $a_n f_n$ converges.*

3.5.3 Proving Convergence

Define the matrices,

$$(\hat{K}_n)_{ij} = k(X_i, X_j), \quad (\hat{D}_n)_{ii} = \sum_{j=1}^{n} k(X_i, X_j),$$

$$\hat{L}_n = \hat{D}_n - \hat{K}_n, \quad \hat{S}_n = \frac{1}{n}\hat{L}_n + \hat{V}_n,$$

the empirical distributions,

$$P_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i},$$

the degree functions,

$$d_n(x) = \int k(x, y) dP_n(y) \in C(\mathcal{M}),$$

$$d(x) = \int k(x, y) dP(y) \in C(\mathcal{M}),$$

the multiplication operators,

$$D_n : C(\mathcal{M}) \to C(\mathcal{M}), \quad D_n f(x) = d_n(x) f(x),$$

$$D : C(\mathcal{M}) \to C(\mathcal{M}), \quad Df(x) = d(x) f(x),$$

$$V_n : C(\mathcal{M}) \to C(\mathcal{M}), \quad V_n f(x) = v(x) f(x),$$

$$V : C(\mathcal{M}) \to C(\mathcal{M}), \quad Vf(x) = v(x) f(x) = V_n f(x),$$

$$M_n : C(\mathcal{M}) \to C(\mathcal{M}), \quad M_n f(x) = V_n f(x) + D_n f(x),$$

$$M : C(\mathcal{M}) \to C(\mathcal{M}), \quad Mf(x) = Vf(x) + Df(x),$$

the integral operators,

$$K_n : C(\mathcal{M}) \to C(\mathcal{M}), \quad K_n f(x) = \int k(x,y) f(y) dP_n(y),$$

$$K : C(\mathcal{M}) \to C(\mathcal{M}), \quad K f(x) = \int k(x,y) f(y) dP(y),$$

and the corresponding sums and differences,

$$L_n : C(\mathcal{M}) \to C(\mathcal{M}), \quad L_n f(x) = D_n f(x) - K_n f(x),$$

$$L : C(\mathcal{M}) \to C(\mathcal{M}), \quad L f(x) = D f(x) - K f(x).$$

Now define the Schrödinger operators, $S_n : C(\mathcal{M}) \to C(\mathcal{M})$,

$$\begin{aligned} S_n f(x) &= V_n f(x) + L_n f(x) \\ &= V_n f(x) + D_n f(x) - K_n f(x) \\ &= M_n f(x) - K_n f(x), \end{aligned}$$

and the limit operator, $S : C(\mathcal{M}) \to C(\mathcal{M})$,

$$\begin{aligned} S f(x) &= V f(x) + L f(x) \\ &= V f(x) + D f(x) - K f(x) \\ &= M f(x) - K f(x). \end{aligned}$$

These definitions ensure that when a function on $\mathcal{M}$ is restricted to the sample

points, the operators on $C(\mathcal{M})$ behave like their matrix analogues. In particular, $\hat{S}_n \circ \rho_n = \rho_n \circ S_n$.

Lemma 3.5.6 (Relating the Spectra of $\hat{S}_n$ and S_n) *1. If $f \in C(\mathcal{M})$ is an eigenfunction of S_n with eigenvalue λ, then the vector $v = \rho_n f \in \mathbb{R}^n$ is an eigenvector of the matrix $\hat{S}_n$ with the same eigenvalue.*

2. If $u = (u_1, u_2, \ldots, u_n)$ is an eigenvector of the matrix $\hat{S}_n$ with eigenvalue $\lambda \notin$ range$(d_n + v)$, then f is an eigenfunction of S_n with the same eigenvalue, where

$$f(x) = \frac{\frac{1}{n}\sum_j k(x, X_j)u_j}{v(x) + d_n(x) - \lambda}.$$

3. If $\lambda \notin$ range$(d + v)$ is an eigenvalue of S, then λ is isolated with finite multiplicity.

Proof.

1. This follows directly from the relation $\hat{S}_n \circ \rho_n f = \rho_n \circ S_n f$: $\hat{S}_n v = \hat{S}_n \rho_n f = \rho_n S_n f = \rho_n(\lambda f)) = \lambda \rho_n f = \lambda u$.

2. Since u is an eigenvector of $\hat{S}_n$, we have

$$\lambda u_k = (\hat{S}_n u)_k = ((\frac{1}{n}\hat{L}_n + V_n)u)_k = \frac{1}{n}\sum_j k(X_k, X_j)(u_k - u_j) + v(X_k)u_k,$$

or

$$u_k(v(X_k) + \frac{1}{n}\sum_j k(X_k, X_j) - \lambda) = \frac{1}{n}\sum_j k(X_k, X_j)u_j.$$

Thus,

$$u_k = \frac{\frac{1}{n}\sum_j k(X_k, X_j)u_j}{v(X_k) + \frac{1}{n}\sum_j k(X_k, X_j) - \lambda} = f(X_k).$$

Now,

$$S_n f(x) = M_n f(x) - K_n f(x) = (v(x) + d_n(x))f(x) - \frac{1}{n}\sum_j k(x, X_j)f(X_j).$$

From the definition of f we have

$$(v(x) + d_n(x))f(x) = \lambda f(x) + \frac{1}{n}\sum_j k(x, X_j)u_j.$$

So

$$S_n f(x) = \lambda f(x) + \frac{1}{n}\sum_j k(x, X_j)u_j - \frac{1}{n}\sum_j k(x, X_j)f(X_j) = \lambda f(x),$$

since $u_j = f(X_j)$.

3. S is the difference of the multiplication operator M and the compact operator K. The essential spectrum is not affected by a compact perturbation [22], so the essential spectrum of S coincides with the essential spectrum of M, which is precisely range$(d + v)$. Thus, if $\lambda \notin$ range$(d + v)$, it is isolated with finite multiplicity.

□

Lemma 3.5.7 *S_n converges compactly to S.*

Proof. By definition, $S_n = V_n + L_n$, and $S = V + L$. In proposition 23 of [52], it was shown that L_n converges compactly to L. Since $V_n = V$ for every n, the claim follows immediately.

□

Putting everything together, we have our main result:

Theorem 3.5.8 (Convergence of Schrödinger Operator) *Let $\lambda \notin range(d+v)$ be an eigenvalue of S and $M \subset \mathbb{C}$ an open neighborhood of λ such that $\sigma(S) \cap M = \{\lambda\}$. Then:*

1. *The eigenvalues in $\sigma(S_n) \cap M$ converge to λ in the sense that every sequence $(\lambda_n)_n$ with $\lambda_n \in \sigma(S_n) \cap M$ satisfies $\lambda_n \to \lambda$ almost surely.*

2. *Convergence of spectral projections: There exists some $N \in \mathbb{N}$ such that, for every $n > N$, the sets $\sigma(S_n) \cap M$ are isolated in $\sigma(S_n)$. For $n > N$, let Pr_n be the spectral projection of S_n corresponding to $\sigma(S_n) \cap M$, and Pr the spectral projection of S for λ. Then $Pr_n \xrightarrow{p} Pr$ almost everywhere.*

3. *Convergence of eigenvectors: If λ is a simple eigenvalue of S and f the corresponding eigenfunction, then the eigenvectors of S_n converge a.s. up to a change of sign, i.e., if v_n is the eigenvector of S_n with eigenvalue λ_n, and $v_{n,i}$ its ith coordinate, then there exists a sequence $(a_n)_n$ with $a_n \in \{-1, 1\}$ such that $sup_{i=1,\dots,n}|a_n v_{n,i} - f(x_i)| \to 0$ almost everywhere.*

Proof. Lemma 3.5.6 established a one-to-one correspondence between the eigensystems of $\hat{S}_n$ and S_n and showed that an eigenvalue $\lambda \notin \text{range}(d + v)$ of S is isolated

with finite multiplicity. Lemma 3.5.7 established that $S_n \xrightarrow{c} S$. These two facts, together with Proposition 3.5.5, imply convergence of eigenvalues and spectral projections, and further, if λ is simple, convergence of eigenvectors, up to a change of sign.

□

3.6 Conclusion

In this chapter, we introduced Schrödinger Eigenmaps. By adding a potential to the graph Laplacian, we allow for the introduction of prior knowledge, thus turning an unsupervised algorithm into a semi-supervised one. This generalization of LE is ideal for classification tasks where user input may be used to label some of the points, as in the example of retinal imagery given above. We have seen that the mapping produced by the algorithm converges to a well defined limit as the sample size increases.

Chapter 4

Laplacian Eigenmaps with Random Projections

As described in Section 2.2.3, LE relies on the construction of a weighted adjacency graph corresponding to the point cloud, and this requires a search for nearest neighbors. If the dimension of the space is high and the dataset is large, this search is expensive or even impossible. In this chapter, we wish to build on a result inspired by the theory of Compressed Sensing (CS) in order to reduce this cost without significantly compromising accuracy. We use random projections as a preliminary step to map the input dataset to a low-dimensional space, thus gaining a dramatic reduction in computational time while, with high probability, essentially preserving the output of the original algorithm. We provide theoretical guarantees as well as numerical evidence of reliability and efficiency.

4.1 Connection to Compressed Sensing

One of the most fundamental elements in the development of useful algorithms for data processing is the model characterizing the expected behavior or structure of the signals of interest. One model that has been the focus of much recent attention is that of sparse signals. Given a basis for the ambient (potentially high-dimensional) space $\mathbb{R}^N$, a signal is called K-sparse if it can be represented using this basis with at most K nonzero coefficients. The theory of CS [18–21, 34, 35] exploits this model in

order to maintain a low-dimensional representation of the signal from which a faithful approximation to the original signal can be recovered efficiently. Dimensionality reduction in CS is linear and nonadaptive, i.e., the mapping does not depend on the data. CS has many promising applications in signal acquisition, compression, and medical imaging [36, 51, 58].

CS theory states that, with high probability, every K-sparse signal $x \in \mathbb{R}^N$ can be recovered from just $M = O(K \log(N/K))$ linear measurements $y = \Phi x$, where Φ is an $M \times N$ measurement matrix drawn randomly from an acceptable distribution. For example, Φ may have i.i.d Gaussian entries. These ideas are illustrated in Figure 4.1. Note that M is linear in the "information level" K and logarithmic in the ambient dimension N. M is taken high enough to ensure that all K-sparse signals remain well-separated when embedded in $\mathbb{R}^M$. CS theory applies equally well to signals that are not strictly sparse but compressible, i.e., if the coefficients in the signal's representation decay fast enough. Furthermore, near optimal recovery is guaranteed even in the presence of noise.

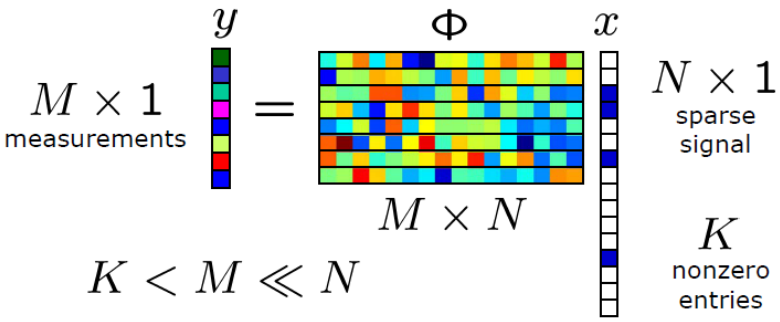


Figure 4.1: Model for measurement in Compressed Sensing

The notion of using a random projection for dimensionality reduction is not

new. Long before the present wave of interest, the Johnson-Lindenstrauss Lemma (JL) [32] used a random projection for a stable embedding of a finite point cloud.

Lemma 4.1.1 (Johnson-Lindenstrauss) *Given* $0 < \epsilon < 1$*, a set* X *of* n *points in* $\mathbb{R}^N$*, and a number* $M \geq O(\ln N)/\epsilon^2$*, there is a Lipschitz function* $f : \mathbb{R}^N \to \mathbb{R}^M$ *such that, for all* $u, v \in X$*,*

$$(1 - \epsilon)\|u - v\| \leq \|f(u) - f(v)\| \leq (1 + \epsilon)\|u - v\|.$$

In [7] a fundamental connection was identified between CS theory and the JL Lemma, despite the fact that the former allows for the embedding of an uncountable number of points.

We note that computing random projections is relatively cheap: projecting n points from N to M dimensions costs $O(NMn)$. To see them in action consider the example shown in Figure 4.2. Here, 2000 points in $\mathbb{R}^{1000}$ are randomly projected to $\mathbb{R}^{20}$. We compute the relative error in the norm of the projected points and plot a histogram. We can see that for the vast majority of points the error is negligible.

Manifold models generalize the notion of sparsity beyond bases. These models arise whenever a signal in $\mathbb{R}^N$ is a continuous function of a K-dimensional parameter. For example, a pure sinusoid is completely determined by its amplitude, phase, and frequency. So a class of signals consisting of pure sinusoids would form a three-dimensional manifold in $\mathbb{R}^N$. The dimension of the manifold under this model is analogous to the sparsity level in the CS model. In [8] the authors extend the CS theory by demonstrating that random linear projections can be used to map the

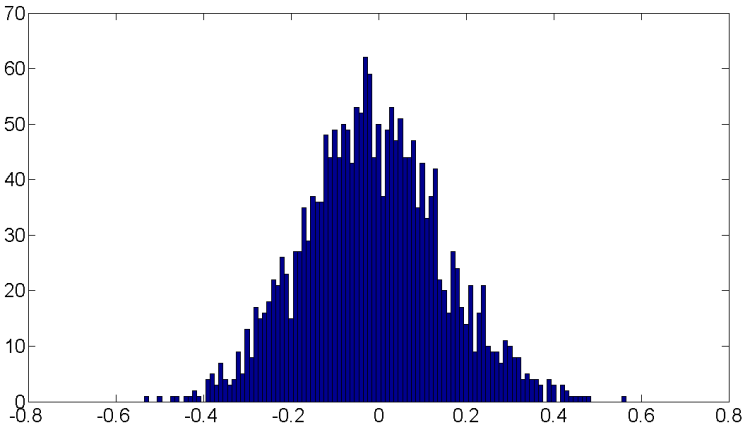


Figure 4.2: Relative error in norm of randomly projected points

high-dimensional manifold-modeled data to a low-dimensional space while, with high probability, approximately preserving all pairwise distances between the points. We use this technique as a preliminary step in LE and show that the resulting algorithm is still provably reliable but considerably faster than the original.

4.2 Preliminaries

4.2.1 Random projections of Smooth Manifolds

First, we recall the main result from [8], which shall be our main tool in establishing a theoretical guarantee on the reliability of our algorithm. The result concerns the effect of a random linear projection $\Phi : \mathbb{R}^N \to \mathbb{R}^M$ on a smooth K-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^N$. Here Φ is a random orthogonal projection, or orthoprojector, constructed by orthonormalizing the rows of an $M \times N$ matrix having i.i.d. Gaussian entries. The authors establish a sufficient number M to guarantee that, with high probability, all pairwise distances between points on $\mathcal{M}$ are

well preserved under the mapping Φ. In their analysis, the authors make several assumptions about regularity of the manifold. In particular, they define the *condition number* of the manifold, which controls both local properties of the manifold (such as curvature) and global properties (such as self-avoidance), and the *geodesic covering regularity* which describes a natural bound on the number of balls of a given radius needed to cover the manifold. Before stating the main result, we state the precise definitions as given in [8].

Definition 4.2.1 *Let $\mathcal{M}$ be a compact Riemannian submanifold of $\mathbb{R}^N$. The condition number is defined as $1/\tau$, where τ is the largest number having the following property: The open normal bundle about $\mathcal{M}$ of radius r is embedded in $\mathbb{R}^N$ for all $r < \tau$.*

Definition 4.2.2 *Let $\mathcal{M}$ be a compact Riemannian submanifold of $\mathbb{R}^N$. Given $T > 0$, the geodesic covering number $G(T)$ of $\mathcal{M}$ is defined as the smallest number such that there exists a set A of points on $\mathcal{M}$, $|A| = G(T)$, so that for all $x \in \mathcal{M}$,*

$$\min_{a \in A} d_{\mathcal{M}}(x, a) \leq T.$$

Definition 4.2.3 *Let $\mathcal{M}$ be a compact Riemannian submanifold of $\mathbb{R}^N$ having volume V. We say that $\mathcal{M}$ has geodesic covering regularity R for resolutions $T \leq T_0$ if*

$$G(T) \leq \frac{R^K V K^{K/2}}{T^K},$$

for all $0 < T \leq T_0$.

Theorem 4.2.4 (Baraniuk and Wakin [8]) *Let $\mathcal{M}$ be a compact K-dimensional Riemannian submanifold of $\mathbb{R}^N$ having condition number $1/\tau$, geodesic covering regularity R, and volume V. Fix $0 < \epsilon < 1$ and $0 < \rho < 1$, and let Φ be a random orthoprojector from $\mathbb{R}^N$ to $\mathbb{R}^M$, where*

$$M \geq \frac{4 - 2\ln(1/\rho)}{\epsilon^2/200 + \epsilon^3/3000} K \ln\left(\frac{1900KNRV}{\epsilon\tau^{1/3}}\right).$$

If $M \leq N$, then, with probability at least $1 - \rho$, the following holds: For every pair of points $x, y \in \mathcal{M}$,

$$(1-\epsilon)\sqrt{\frac{M}{N}} \leq \frac{\|\Phi x - \Phi y\|_2}{\|x - y\|_2} \leq (1+\epsilon)\sqrt{\frac{M}{N}}.$$

The proof proceeds by first specifying a high resolution sampling of points on the manifold, and on the tangent spaces to these points. The JL Lemma is invoked to produce a satisfactory embedding for these points. The embedding is then extended to the entire manifold based on the notions of regularity discussed above.

4.2.2 Approximation of Eigenvectors

We will need a standard result on the approximation of eigenvectors (see e.g. [12]).

Theorem 4.2.5 *Let L and $\hat{L}$ be two symmetric, positive semidefinite matrices, with nondecreasing simple eigenvalues $\{\lambda_j\}$ and $\{\hat{\lambda}_j\}$, respectively. Let v_k be a normalized*

eigenvector of L associated with λ_k. If $r > 0$ satisfies

$$r \leq \min_{i,j} |\lambda_i - \lambda_j| \quad \textit{and} \quad \|L - \hat{L}\| \leq r/2,$$

then,

$$\|v_k - \hat{v}_k\| < \frac{4}{r}\|L - \hat{L}\|,$$

where $\hat{v}_k$ is a normalized eigenvector of $\hat{L}$ associated with $\hat{\lambda}_k$.

4.2.3 Laplacian Eigenmaps

We recall the main steps of LE. Given points $x_1, x_2, \ldots, x_n$ in $\mathbb{R}^N$:

1. Constructing the adjacency graph: Given a parameter $m \in \mathbb{N}$, put an edge between nodes i and j if x_i is among the m nearest neighbors of x_j or vice versa. Given a parameter $t > 0$, if nodes i and j are connected, set $W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$.

2. Constructing the Laplacian matrix: Set $D_{ii} = \sum_j W_{ij}$, and let $L = D - W$.

3. Computing the eigenmaps: Solve the generalized eigenvalue problem, $Lf = \lambda Df$. Let $f_0, f_1, \ldots, f_K$ be $K + 1$ eigenvector solutions corresponding to the first eigenvalues $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_K$. Discard f_0 and use the next K eigenvectors to embed in K-dimensional Euclidean space using the map $x_i \to (f_1(i), f_2(i), \ldots, f_K(i))$.

4.3 Main Result

We begin with a point cloud in $\mathbb{R}^N$ assumed to lie on a K-dimensional submanifold that we wish to learn. We use a random linear projection to map the points to $\mathbb{R}^M$. We then use LE on the projected set, rather than the original. Our goal is to show that, under the standard regularity assumptions on the manifold, if M is sufficiently high (yet logarithmic in N and linear in K), then with high probability, the difference in the resulting output is negligible. This amounts to showing that the eigenvectors computed in step 3, as described in section 4.2.3 above, remain essentially the same. We now state this result precisely.

Theorem 4.3.1 *Given a data set $\{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^N$, sampled from a compact K-dimensional Riemannian manifold , assume $\|x_i - x_j\| \leq A$ for all i, j and some $A > 0$. Let $0 < \lambda_1 < \lambda_2 < \cdots < \lambda_K$ be the first K nonzero eigenvalues computed by LE, assumed simple, with $r = \min_{i,j} |\lambda_i - \lambda_j|$, and let f_j be a normalized eigenvector corresponding to λ_j. Use a random orthoprojector Φ (as described above) to map the points to $\mathbb{R}^M$. Let $\hat{f}_j$ be the jth eigenvector computed by LE for the projected data set. Fix $0 < \alpha < 1$ and $0 < \rho < 1$. If*

$$M \geq \frac{4 - 2\ln(1/\rho)}{\epsilon^2/200 + \epsilon^3/3000} K \ln\left(\frac{CKN}{\epsilon}\right), \text{ where } \epsilon = \frac{r\alpha}{4An(n-1)},$$

then, with probability at least $1 - \rho$,

$$\|f_j - \hat{f}_j\| < \alpha.$$

The constant C depends on properties of the manifold. Precisely, $C = \frac{1900RV}{\tau^{1/3}}$, where R, V, and $1/\tau$ are the geodesic covering regularity, volume, and condition number, respectively, as described in [8].

Proof. Let $d_{ij} = \|x_i - x_j\|$, $\hat{d}_{ij} = \|\Phi x_i - \Phi x_j\|$. The construction in section 4.2.3 leads to an eigenvalue problem for a matrix L whose elements L_{ij} are continuous functions of the interpoint distances d_{ij}. More precisely, for $i \neq j$, $L_{ij} = e^{-d_{ij}^2/t}$ (for convenience we shall assume $t = 1$), and L_{ii} is a sum of $n - 1$ terms of this form. Thus, given $\beta > 0$, there is a $\delta_{ij} > 0$ such that if $|d_{ij} - \hat{d}_{ij}| < \delta_{ij}$, then $|L_{ij} - \hat{L}_{ij}| < \beta$. In fact, since the derivative of e^{-x^2} is bounded by 1, we can let $\delta_{ij} = \beta$, for $i \neq j$, and $\delta_{ii} = \beta/(n-1)$. Let $\beta = (r\alpha)/(4n)$ and let $\delta = \min_{i,j} \delta_{ij} = \beta/(n-1) = (r\alpha)/[4n(n-1)]$. We may choose M as prescribed by Theorem 4.2.4, so that, with probability at least $1 - \rho$, for all i, j, $|\hat{d}_{ij}/d_{ij} - 1| < \delta/A$. Since $d_{ij} \leq A$, we obtain

$$|d_{ij} - \hat{d}_{ij}| < \delta \text{ and } |L_{ij} - \hat{L}_{ij}| < \beta.$$

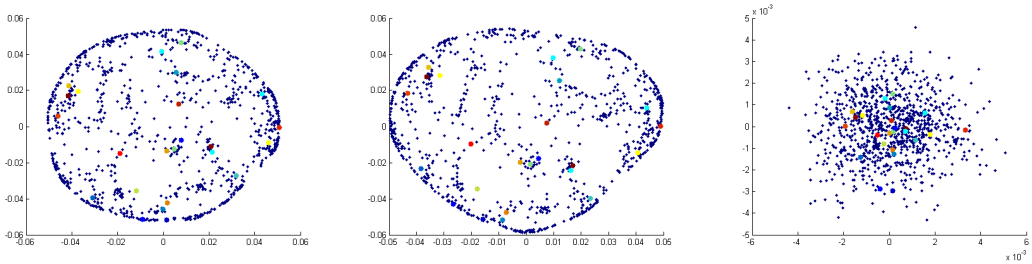
This establishes a bound on the maximum norm of the difference between the matrices, which is equivalent to the operator norm. In particular, for a matrix $E \in \mathbb{M}_n(\mathbb{C})$, $\|E\| \leq n\|E\|_{\max}$. Thus, we have $\|L - \hat{L}\| \leq n\|L - \hat{L}\|_{\max} \leq n\beta = r\alpha/4$. The claim now follows using Theorem 4.2.5.

□

4.4 Numerical Experiments

4.4.1 Artificial Data

We now offer numerical evidence of the advantage of using LE with random projections (LERP). First, we construct an artificial two-dimensional manifold embedded in high-dimensional space ($\mathbb{R}^{200}$). We compare the results of running LE on the original data set, with LE on the projection of the data set to a low-dimensional space ($\mathbb{R}^{20}$). Furthermore, we compare the result with the output produced by the state-of-the-art out-of-sample extension algorithm, Improved Nystrom (IN), described in [63]. Figure 4.3 shows the results. We clearly see, as suggested by the colored marks, that even when the general shape of the image is somewhat altered with LERP, the relative positions of the mapped points, which are crucial for the purposes of classification, are well preserved. This property holds for IN to a much lesser extent.

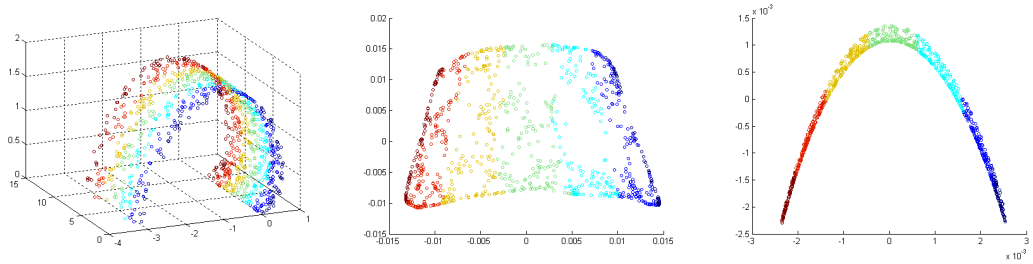


(a) LE (b) LERP (c) Improved Nystrom

Figure 4.3: Reducing a two-dimensional manifold embedded in $\mathbb{R}^{200}$

As a more drastic example, we compare the performance of LERP with IN on a piece of a Swiss roll, commonly used in this setting. Figure 4.4 shows that

while LERP correctly identifies the two-dimensional structure, IN collapses it onto a one-dimensional structure.



(a) original set (b) LERP (c) Improved Nystrom

Figure 4.4: Reducing a Swiss roll

4.4.2 Real Data

To further demonstrate the utility of our method, we now test it with real data, using the hyperspectral dataset Urban described in section 2.4. Recall that the image contains 307×307 pixels, and ground truth is available for 1058 of these, which are classified into one of 23 classes. We use a 248×253 rectangular piece of the image which contains all the ground truth pixels. Table 4.1 presents a comparison of LE, LERP, and IN, in terms of running times and accuracy of classification, averaged over 15 trials: LERP outperforms IN by both measures. Figure 4.5 shows a comparison of accuracy of classification for LE and LERP by class. Figures 4.6 - 4.12 show a comparison of class maps, which are virtually identical. Figures 4.13 - 4.15 show a comparison of eigenvectors corresponding to the smallest eigenvalues, where we can see that LE and LERP produce a very similar separation of materials in the scene.

Table 4.1: Comparison of performance on Urban

Method	Time (min)	Accuracy (percent)
LE	15.26	79.05
LERP	11.78	78.44
IN	12.02	72.09

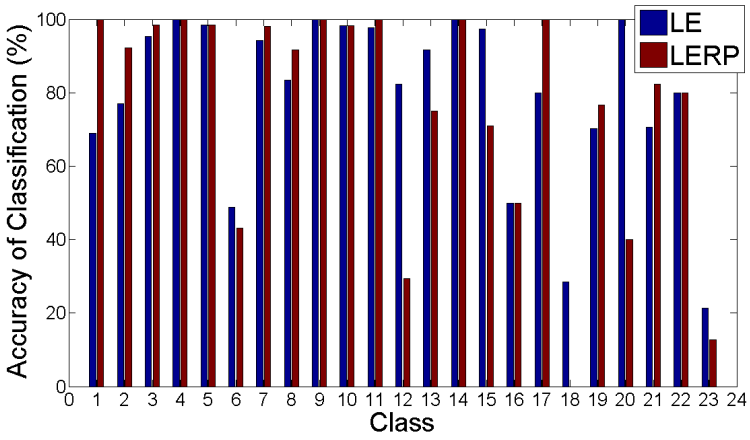
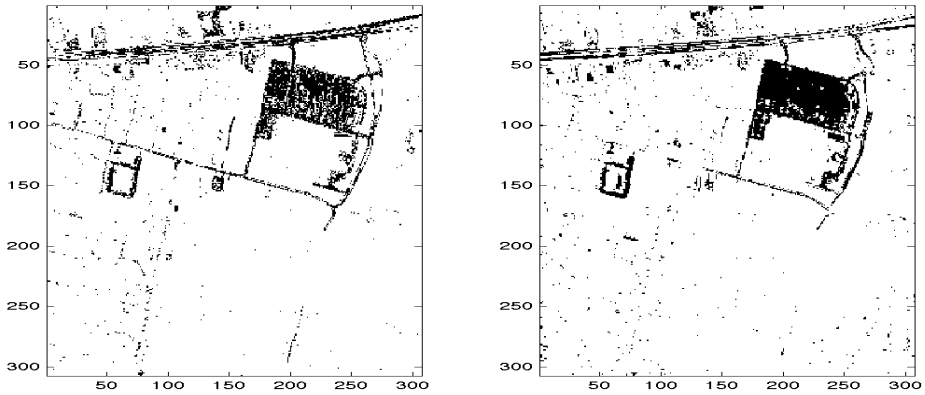


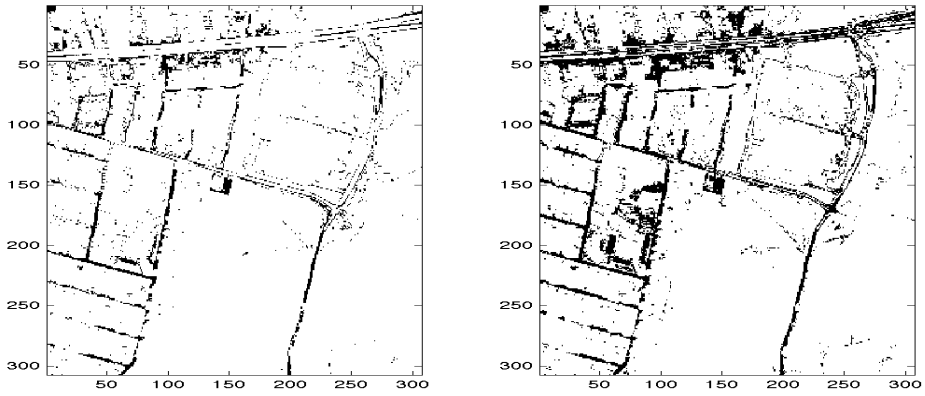
Figure 4.5: Classification of Urban using LE and LERP, comparison by class

4.5 Conclusion

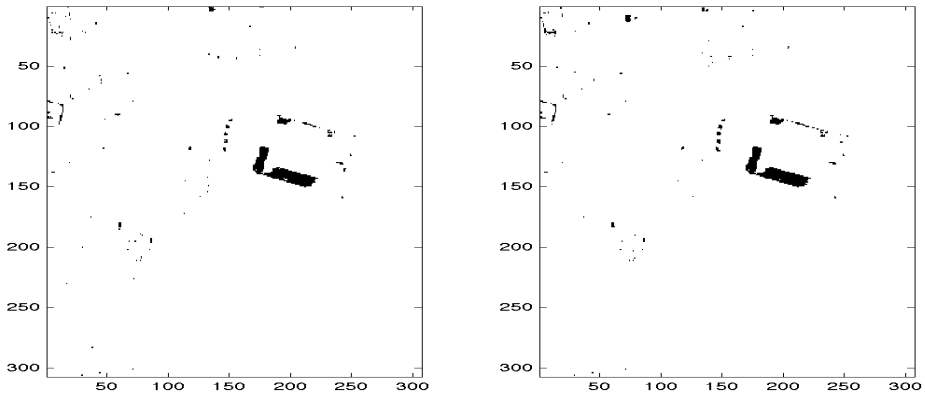
We have shown, both theoretically and empirically, that using a random linear projection as a preliminary step in LE preserves the essential properties of the mapping. At the same time, by accelerating the search for nearest neighbors, it allows for a dramatic reduction in computational time. Thus, using a preliminary random projection makes the algorithm more attractive for applications in general, and for the classification of high-dimensional data, in particular. Finally, we remark that using random projections with other algorithms, which similarly rely on the construction of neighborhoods, is likely to have similar benefits.



(a) Class 1

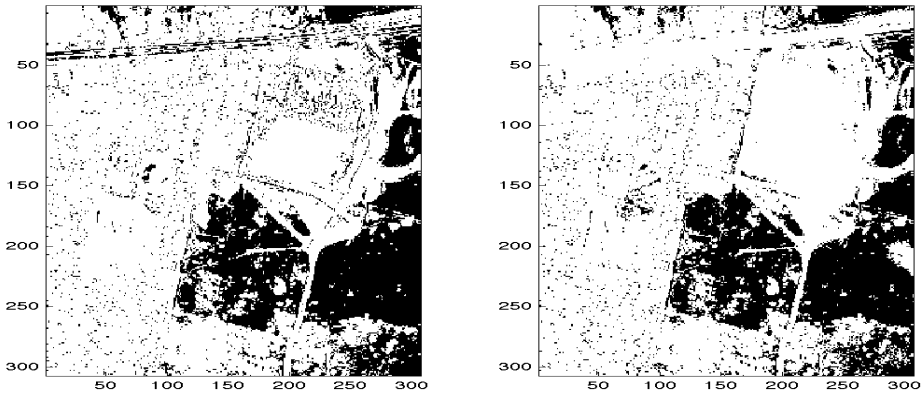


(b) Class 2

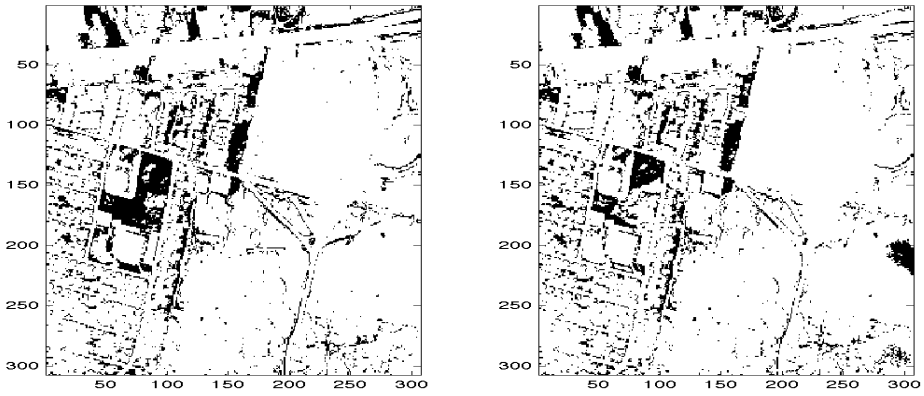


(c) Class 3

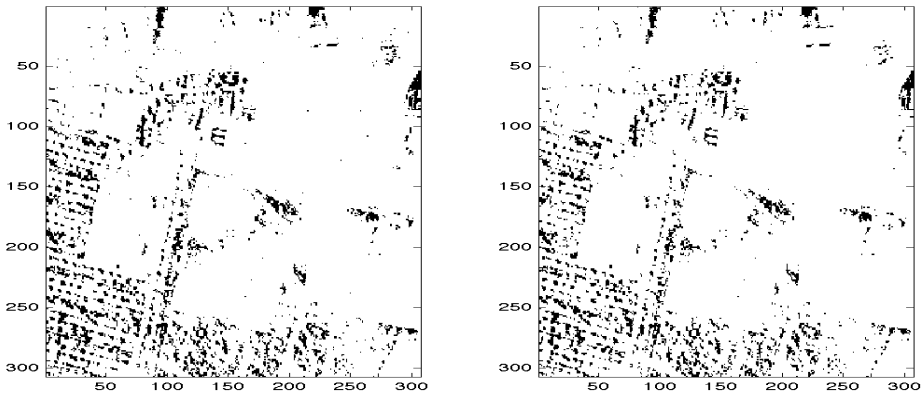
Figure 4.6: Urban classes: left - LE, right - LERP



(a) Class 4

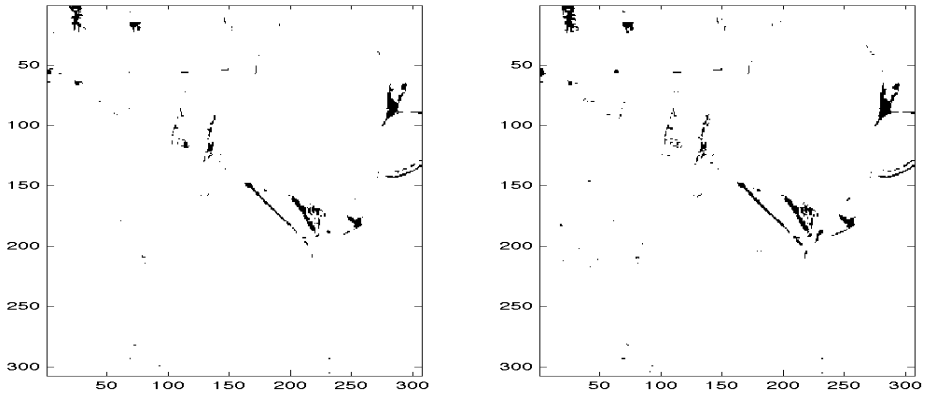


(b) Class 5

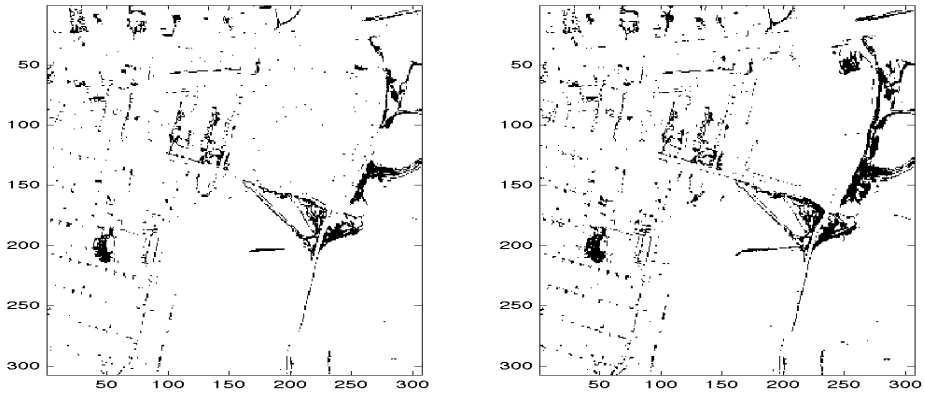


(c) Class 6

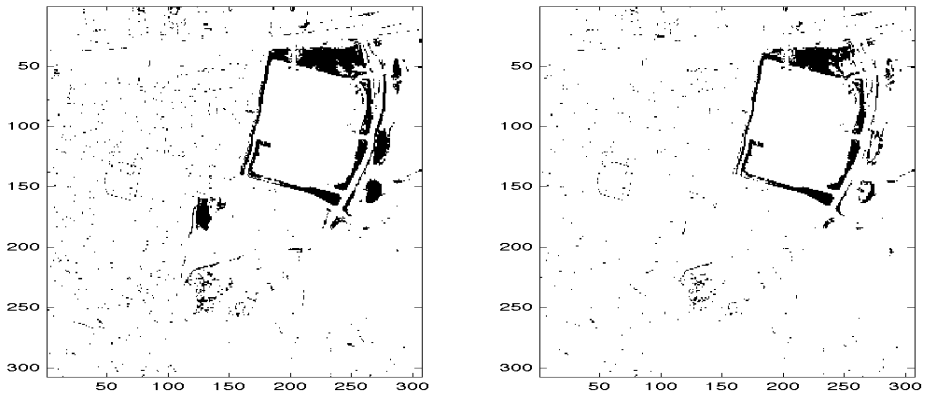
Figure 4.7: Urban classes: left - LE, right - LERP



(a) Class 7

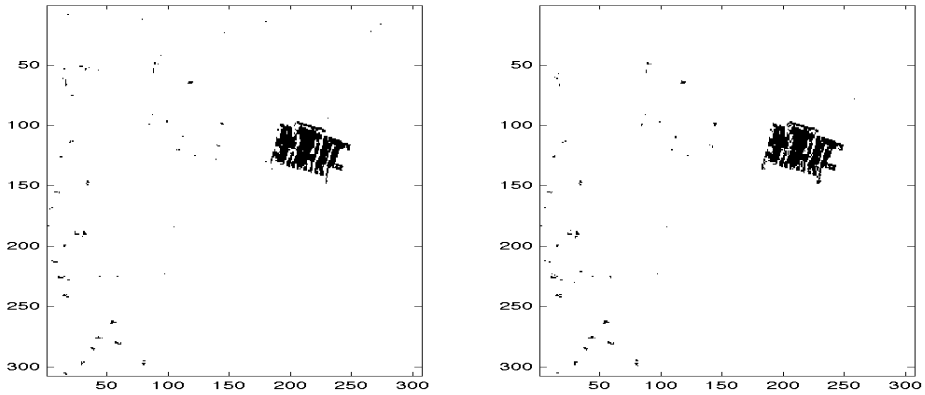


(b) Class 8

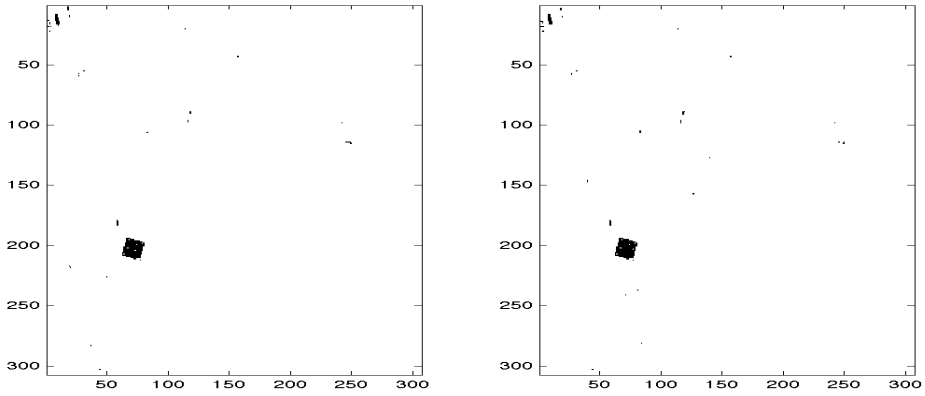


(c) Class 9

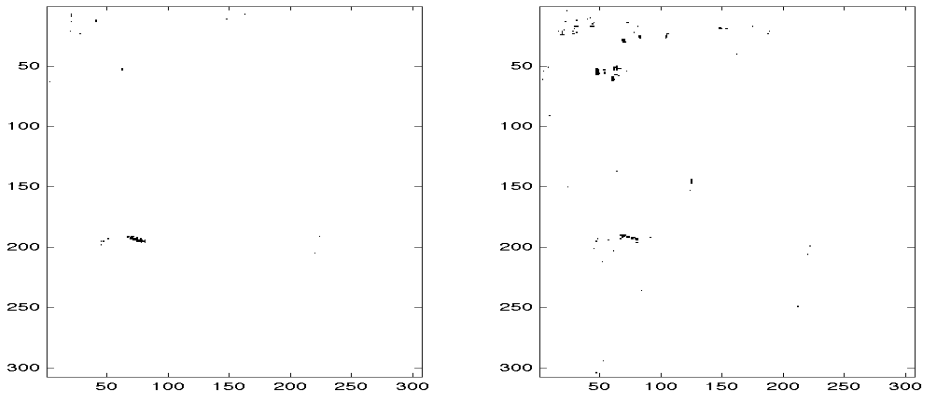
Figure 4.8: Urban classes: left - LE, right - LERP



(a) Class 10

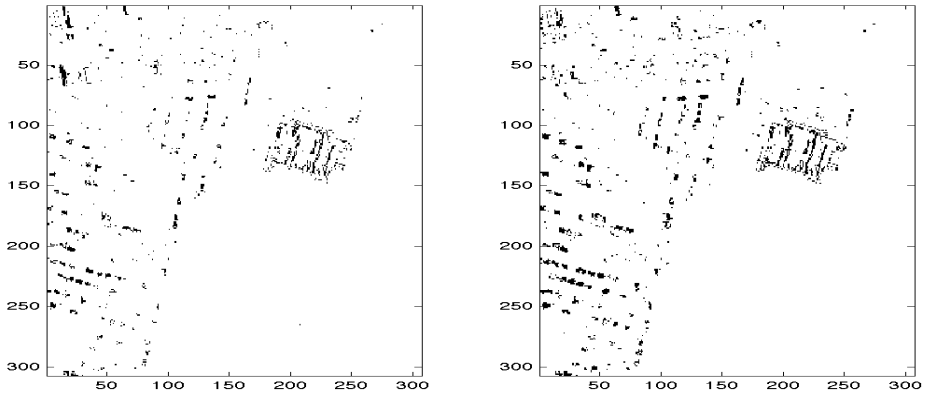


(b) Class 11

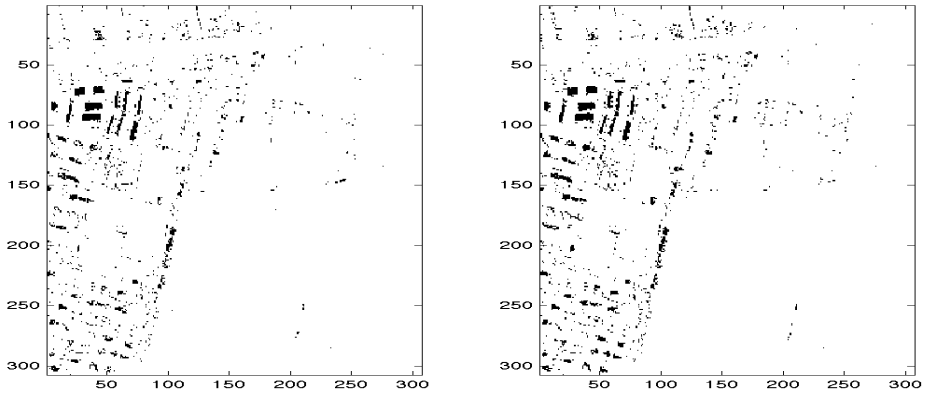


(c) Class 12

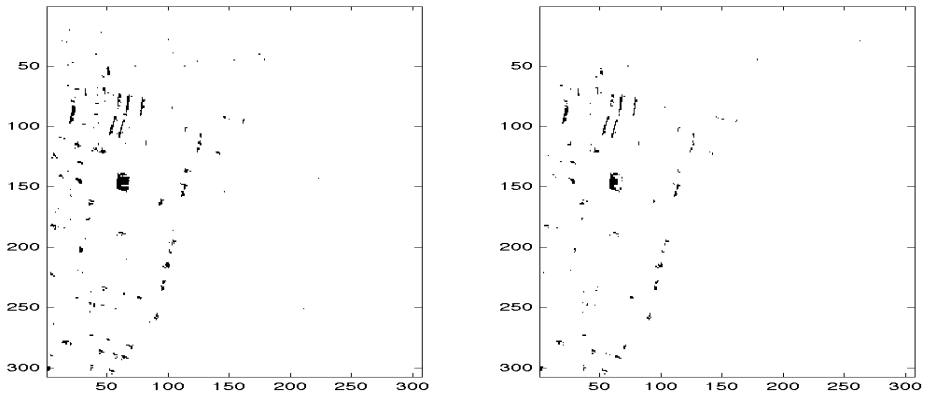
Figure 4.9: Urban classes: left - LE, right - LERP



(a) Class 13

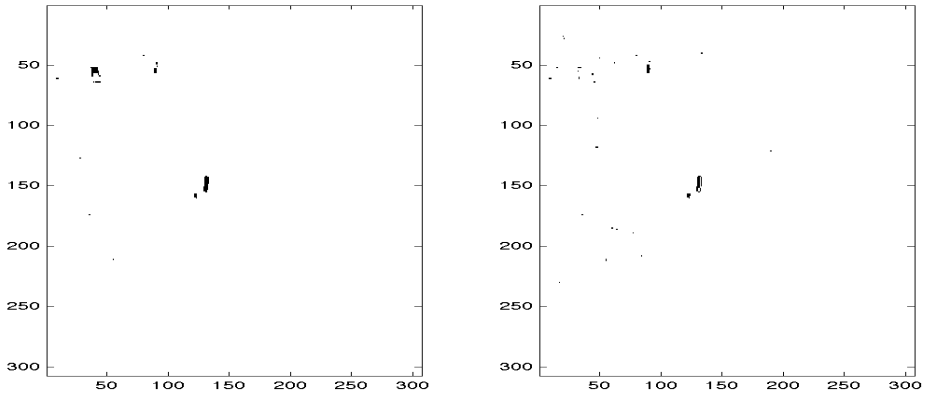


(b) Class 14

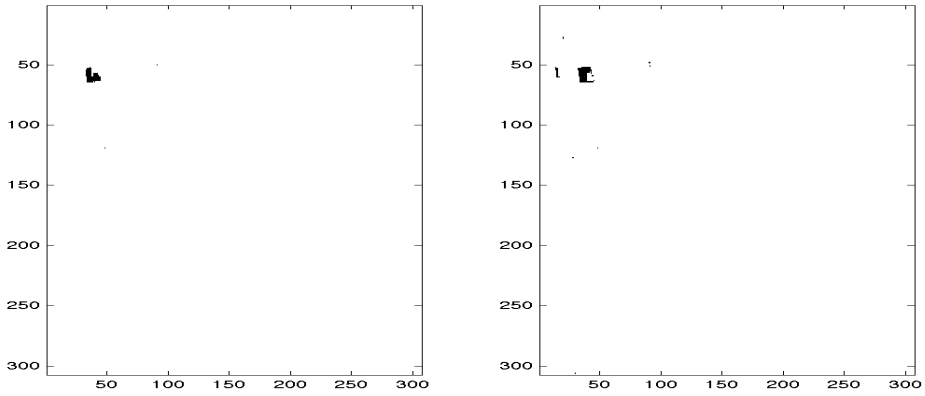


(c) Class 15

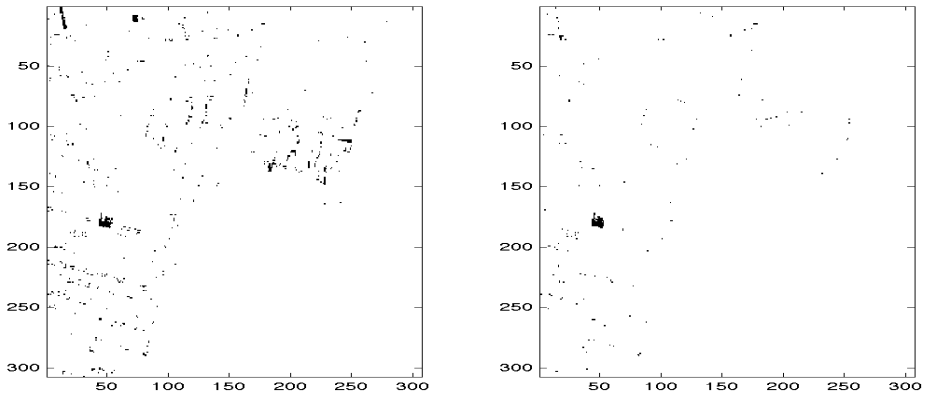
Figure 4.10: Urban classes: left - LE, right - LERP



(a) Class 16

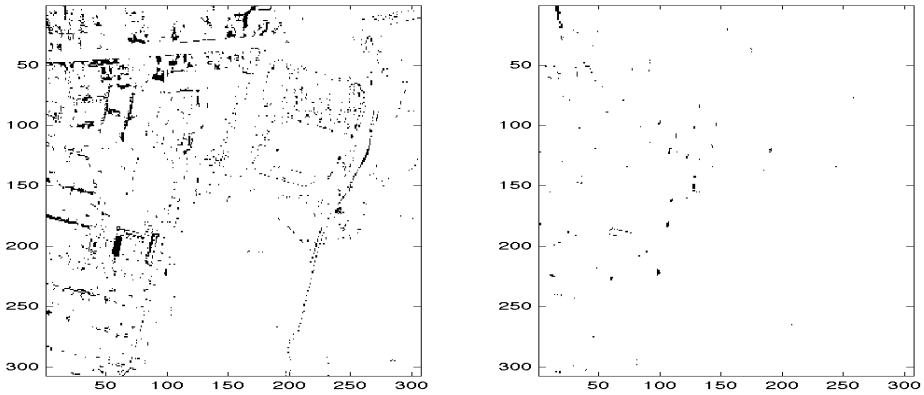


(b) Class 17

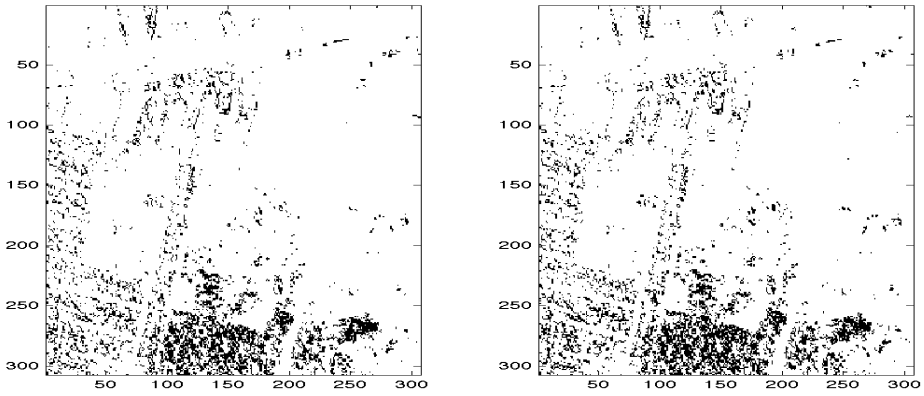


(c) Class 18

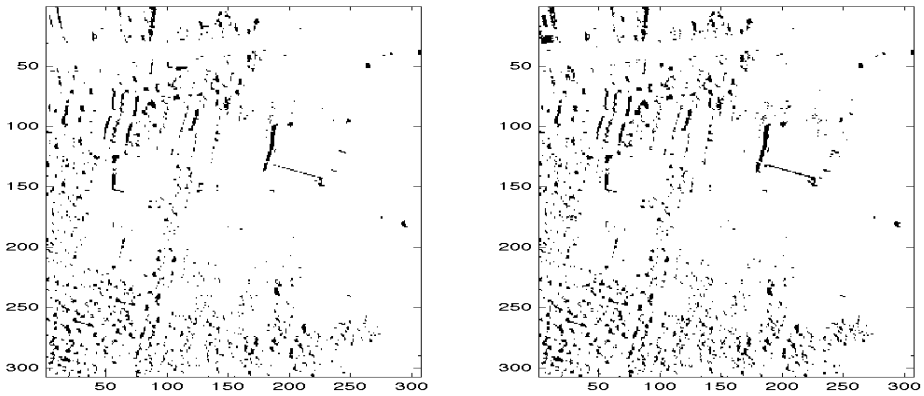
Figure 4.11: Urban classes: left - LE, right - LERP



(a) Class 19



(b) Class 20



(c) Class 21

Figure 4.12: Urban classes: left - LE, right - LERP

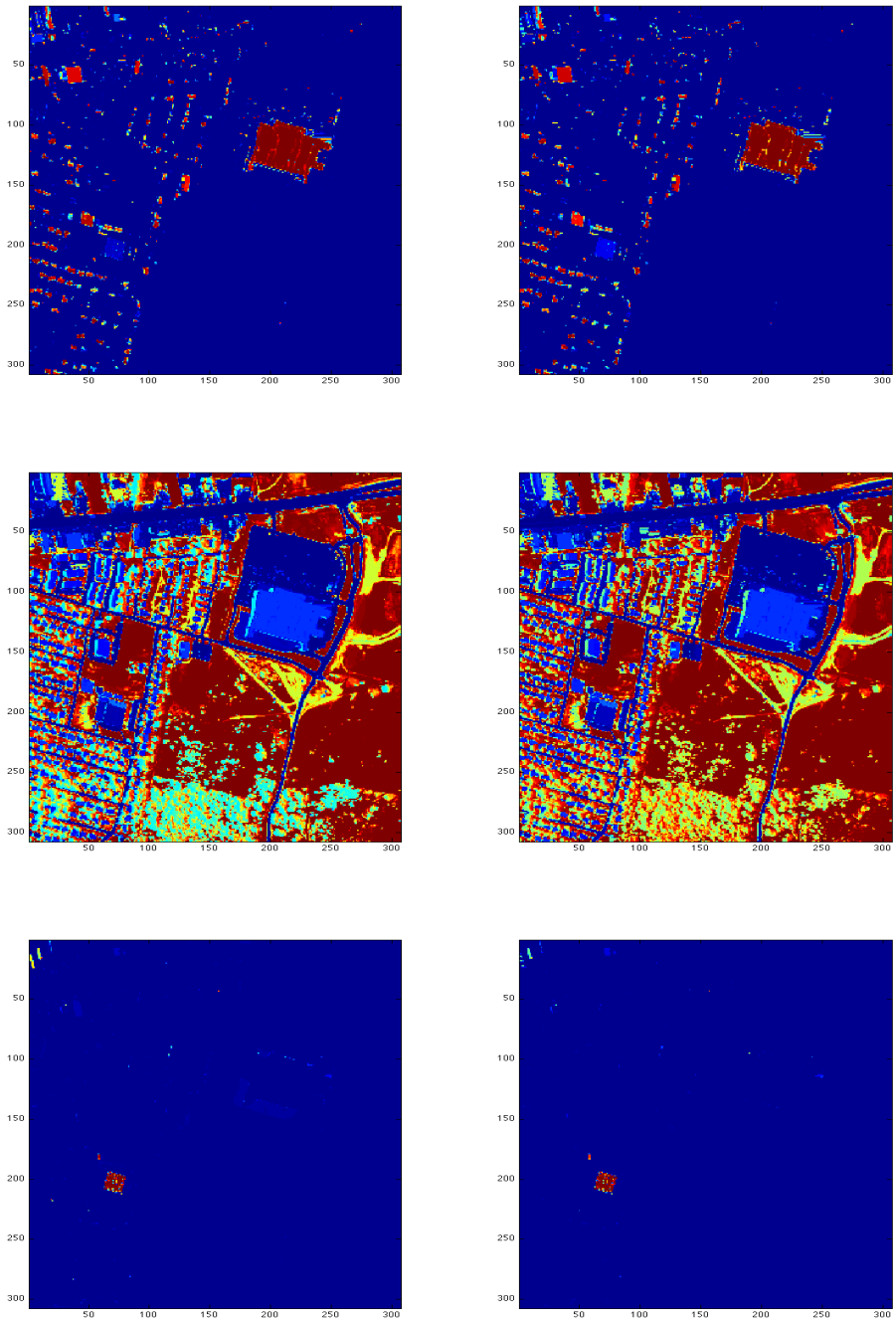


Figure 4.13: Urban eigenvectors: left - LE, right - LERP

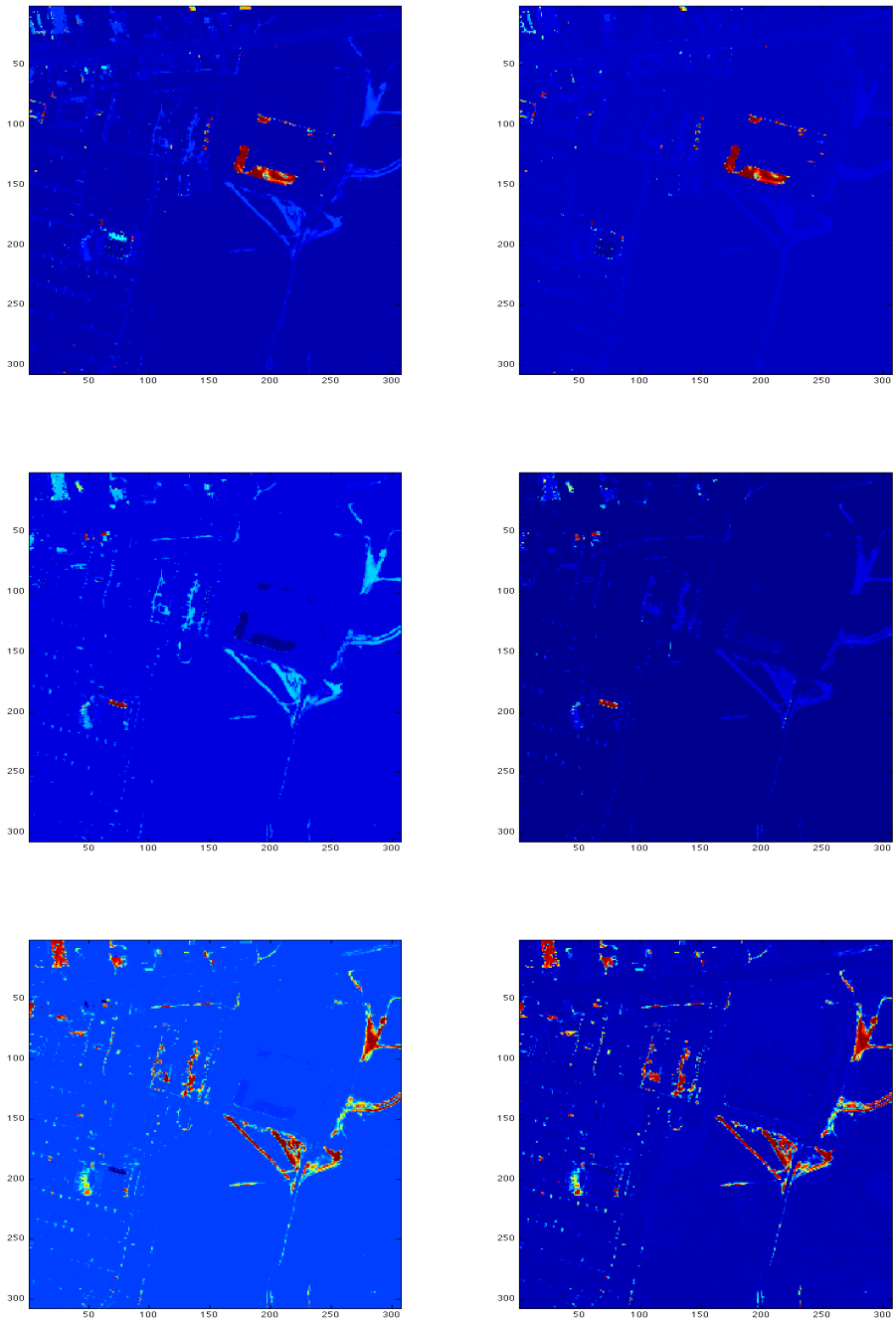


Figure 4.14: Urban eigenvectors: left - LE, right - LERP

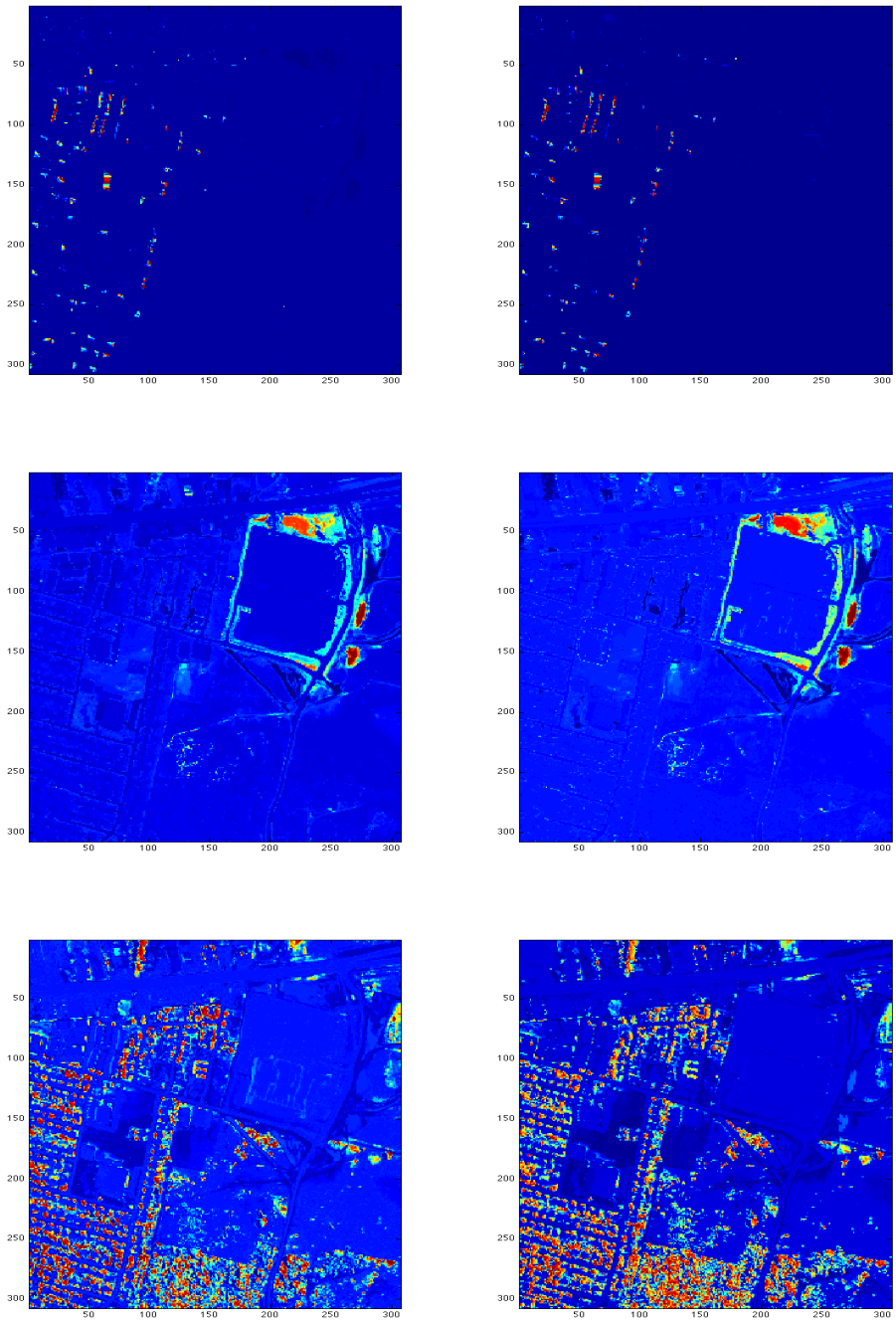


Figure 4.15: Urban eigenvectors: left - LE, right - LERP

Chapter 5

Fast Approximate Neighborhoods

5.1 Introduction

As described in previous sections, at the heart of Laplacian Eigenmaps, as well as many related Laplacian-based methods, is the construction of the k nearest neighbor (kNN) graph, from which a discrete approximation to the manifold Laplacian is derived. The time complexity of the brute-force construction depends linearly on the dimension d of the ambient space, and quadratically on the number n of points. For large datasets, the computation can thus become impractical or even impossible. In order to remedy this situation, we have implemented and tested a recursive algorithm described in [23], which allows for a significant speed-up over the brute-force method, virtually without compromising accuracy.

5.2 Background

The problem of searching for nearest neighbors has attracted much attention in recent years, in light of its importance in numerous applications in domains such as pattern recognition, data mining, machine learning, computer vision, and computational statistics. Many algorithms, deterministic and randomized, exact and approximate, have been proposed, see, e.g., [23], and references therein. However,

most of these algorithms perform poorly in high dimensions, require a significant amount of pre-processing, or fail to provide a guarantee of asymptotic time complexity [1,55]. The algorithm we have chosen to implement and use with LE requires no pre-processing, is very effective in high dimensions, and comes complete with a detailed analysis of time complexity.

5.3 The Algorithm

Our problem can be stated as follows. For each of n data points $x_1, x_2, \ldots, x_n$ in $\mathbb{R}^d$, find its k nearest neighbors (kNN), where for a measure of proximity we use the Euclidean norm. The brute-force method for computing the exact kNN graph requires $\Theta(dn^2)$ time. We now describe a divide-and-conquer method for computing an approximate kNN graph in $\Theta(dn^t)$ time. The exponent t is larger than 1, but as we shall see, experiments show that a small value, close to 1, is sufficient for a high quality graph.

The set of points is recursively divided into two overlapping subsets, as in Figure 5.1, where the size of the overlap is controlled by a parameter $0 < \alpha < 1$ (which determines the exponent t, as we shall soon see). The division is accomplished using spectral bisection, based on the inexpensive Lanczos algorithm. Spectral bisection uses the largest singular triplet of the centered data to produce a hyperplane that separates the points into two sets. The separation is optimal in the sense that the sum of the squared distances between the points and the hyperplane is maximized. To see this, let $\hat{X}$ denote the centered data, and let (σ, u, v) denote the largest sin-

gular triplet of $\hat{X}$ with $u^T\hat{X} = \sigma v^T$. Then, for any hyperplane $w^T x = 0$, the sum is

$$\sum_{i=1}^{n} (w^T \hat{x}_i)^2 = \|w^T \hat{X}\|_2^2 \leq \|\hat{X}\|_2^2 = \sigma^2,$$

while setting $w = u$ achieves equality.

Once the size of a subset is less than a threshold r, the kNN graph is computed using brute-force. The solutions to the small subproblems are then assembled in a simple conquer step: If a data point belongs to more than one of the subsets, its k nearest neighbors are selected from the neighbors found in each of the subsets. Due to the nature of the divide-and-conquer approach, only a small portion of the n^2 distances are actually computed. Memory requirements can be kept modest by using a hash table to store them.

If we denote by $f(n)$ the time needed for the divide-and-conquer steps, then the time complexity T of this algorithm satisfies the following recurrence relation:

$$T(n) = 2T((1+\alpha)\frac{n}{2}) + f(n).$$

It is straightforward to show that $f(n) = O(dn)$. Using the Master Theorem [28] we then have the solution:

$$T(n) = \Theta(dn^t), t = \frac{1}{1 - \log_2(1+\alpha)}.$$

For example, in the experiments below we use $\alpha = 0.1$, in which case $t = 1.16$.

The following is pseudo code for the main functions implemented:

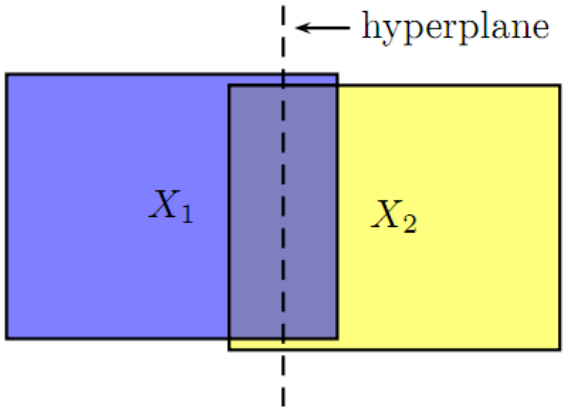


Figure 5.1: Approximate kNN

function $G = k\text{NN}(X, k, \alpha)$
    if $|X| < r$ **then**
        $G = k$NN-Brute-Force(X, k)
    else
        $(X_1, X_2) =$ Divide(X, α)
        $G_1 = k\text{NN}(X_1, k, \alpha)$
        $G_2 = k\text{NN}(X_2, k, \alpha)$
        $G =$ Conquer(G_1, G_2)
    end if
end function

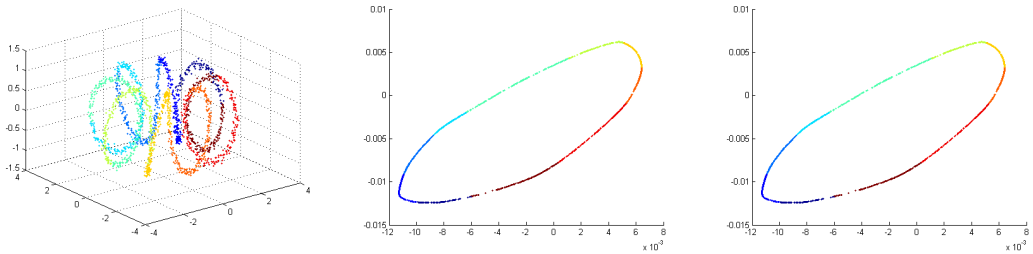
function $[X_1, X_2] =$ Divide(X, α)
    $\hat{X} =$ centered X
    $v =$ largest right singular vector of $\hat{X}$
    $X_1 = \{x_i | v_i \geq (100\alpha)\%$ of absolute values of elements in $v\}$
    $X_2 = \{x_i | v_i < (100\alpha)\%$ of absolute values of elements in $v\}$
end function

function $G =$ Conquer(G_1, G_2)
    for each point in G_1, G_2
        if point is only in G_1
            place its k neighbors in G
        else if point is only in G_2
            place its k neighbors in G
        else
            find its k nearest neighbors among the $2k$ neighbors in G_1 and G_2
            place its k neighbors in G
        end if
    end for
end function

function $G = k$NN-Brute-Force(X, k)
    compute all pairwise distances for points in X
    for each point in X
        find its k nearest neighbors
        place its k neighbors in G
    end for
end function

5.4 Experimental Results

To investigate the effect of using the approximate neighborhoods in Laplacian Eigenmaps, we have conducted several tests. First, we used artificial data sets, such as the helix shown in figure 5.2. As can be seen, the resulting maps, using the exact and approximate neighborhoods, are virtually indistinguishable. Furthermore, the relative error incurred in the norms of the resulting matrices is typically less than 1%. We obtain the same result using a two dimensional roll embedded in $\mathbb{R}^3$, as shown in Figure 5.3. We note in passing that the mapping produced when using the approximate algorithm is consistently better, at least visually, as suggested by this figure.



(a) Helix (b) Exact (c) Approximate

Figure 5.2: Mapping a one-dimensional helix embedded in $\mathbb{R}^3$

Finally, we use LE to reduce the dimension of Urban and classify its pixels.

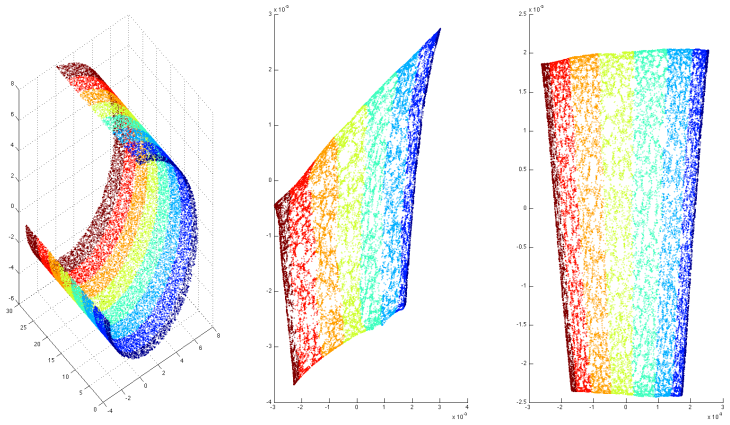


Figure 5.3: Left: two dimensional roll, center: exact, right: approximate

We do it twice: First, using the exact neighborhood construction, and second, using the approximate construction. Table 5.1 presents a comparison of running time and accuracy of classification: The approximate algorithm outperforms the exact one by both measures. Figure 5.4 shows a comparison of accuracy by class. Figures 5.5 - 5.11 show a comparison of class maps, most of which are very similar.

Table 5.1: Comparison of performance on Urban

Method	Time (min)	Accuracy (percent)
LE - exact neighborhood	35.3	77.25
LE - approximate neighborhood	5.33	80.01

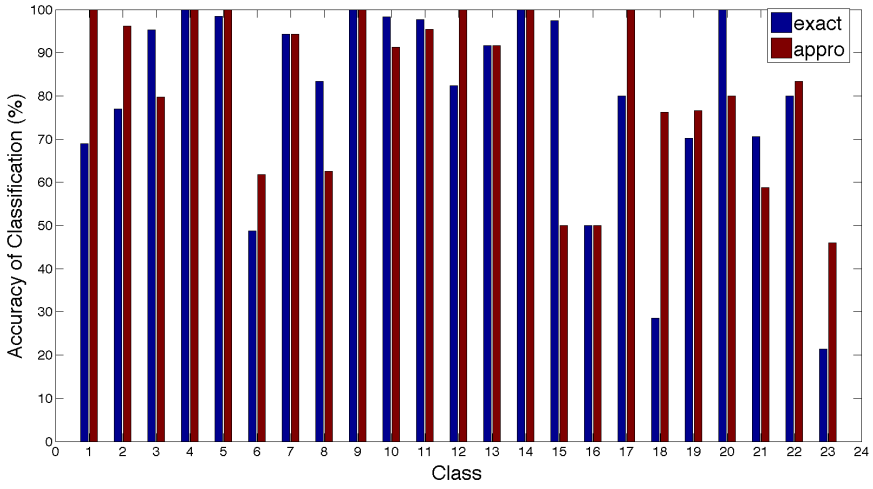
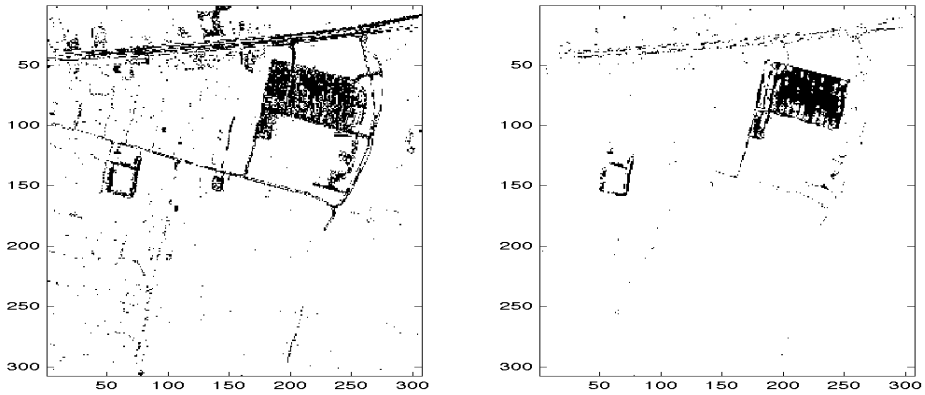
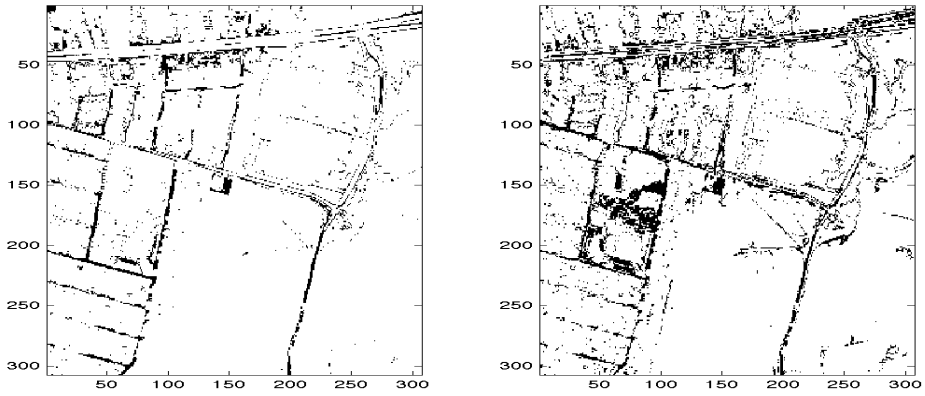


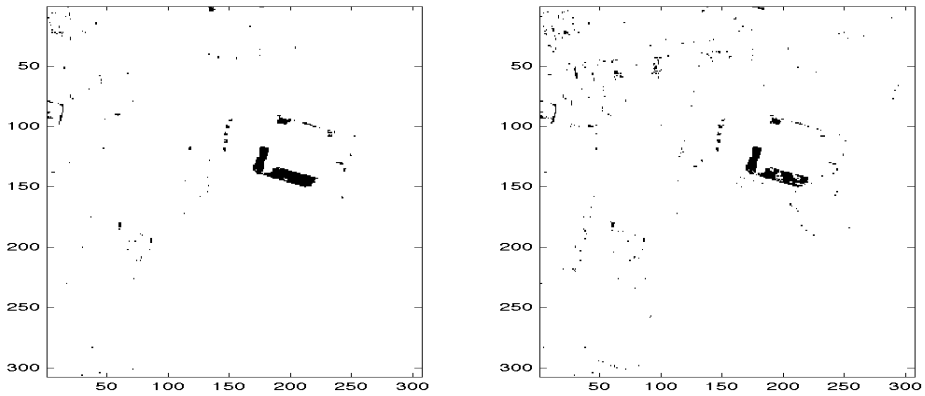
Figure 5.4: Classifying Urban using LE with exact and approximate neighborhoods



(a) Class 1

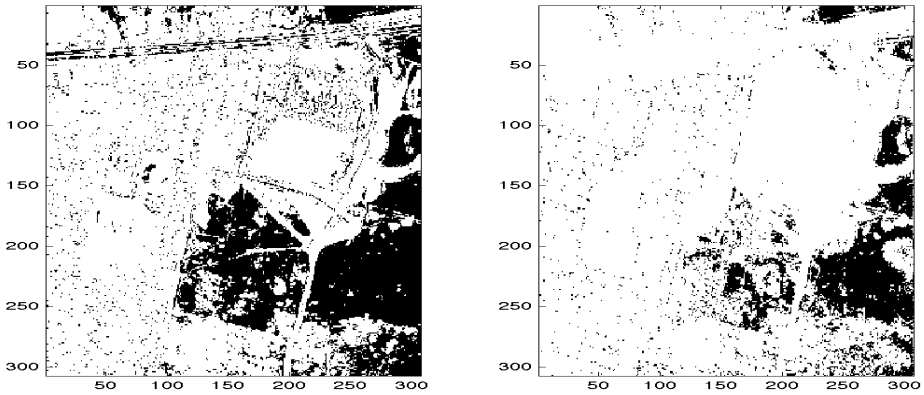


(b) Class 2

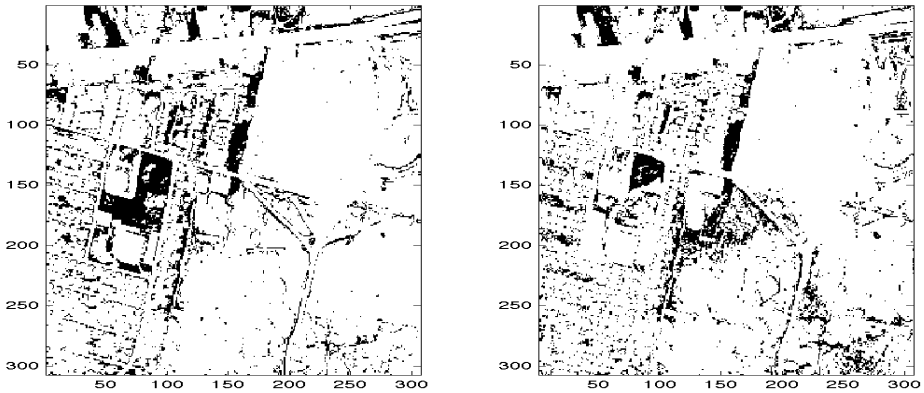


(c) Class 3

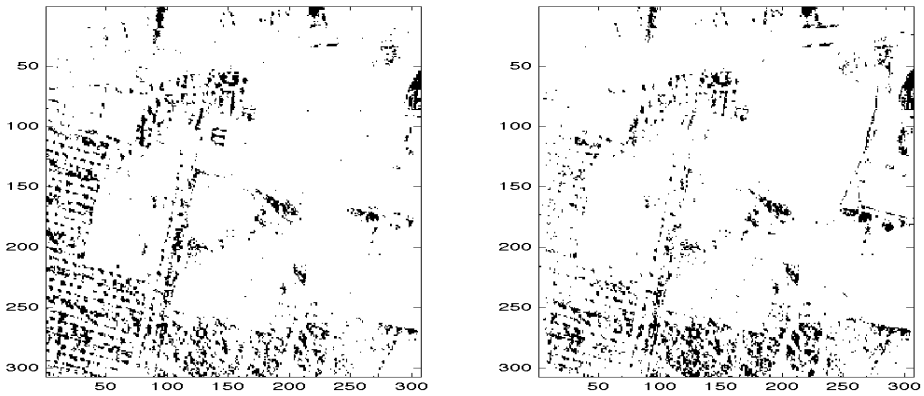
Figure 5.5: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 4

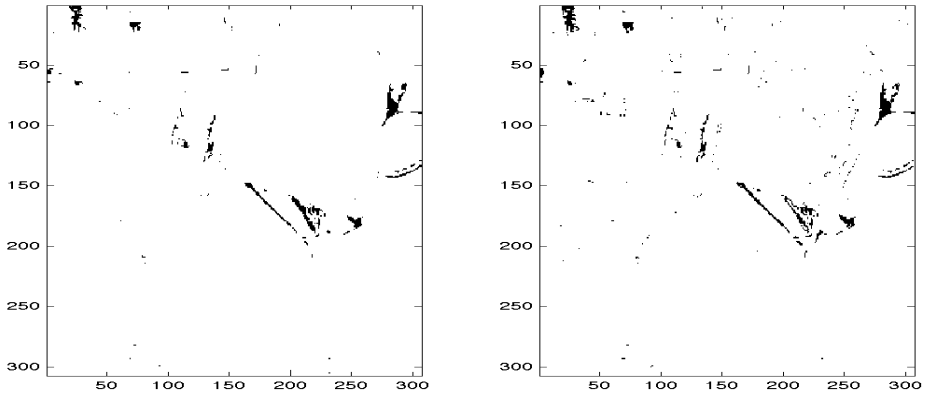


(b) Class 5

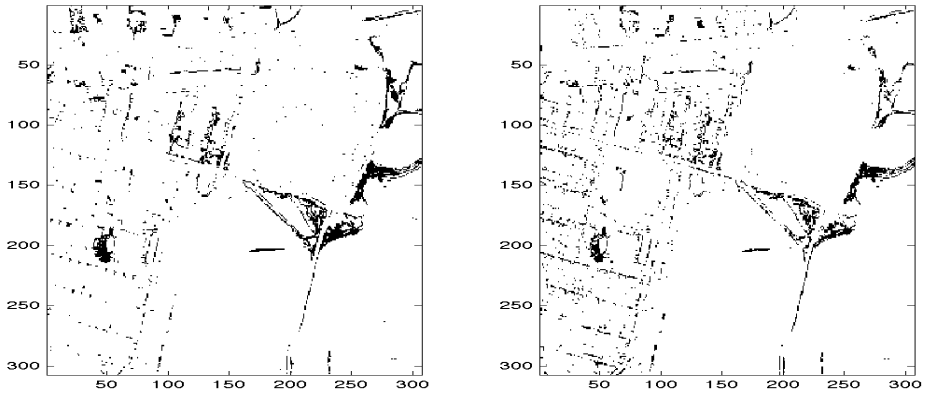


(c) Class 6

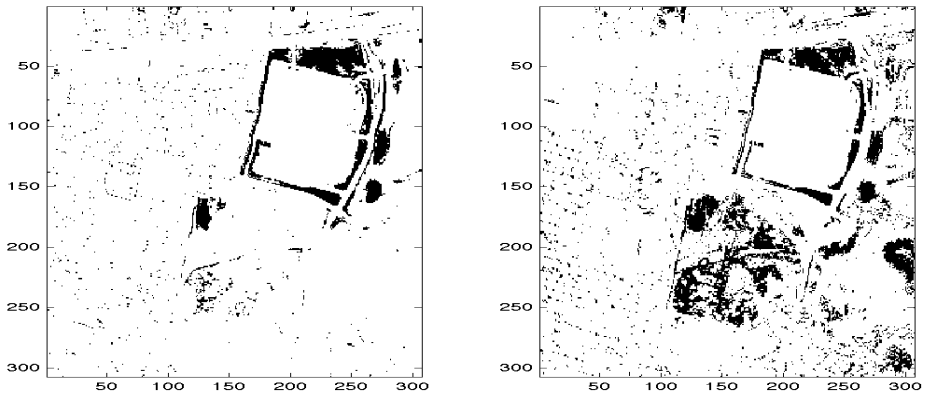
Figure 5.6: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 7

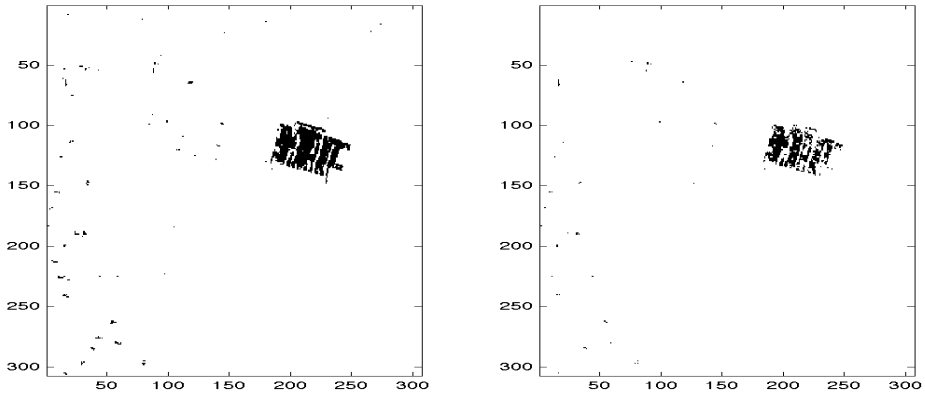


(b) Class 8

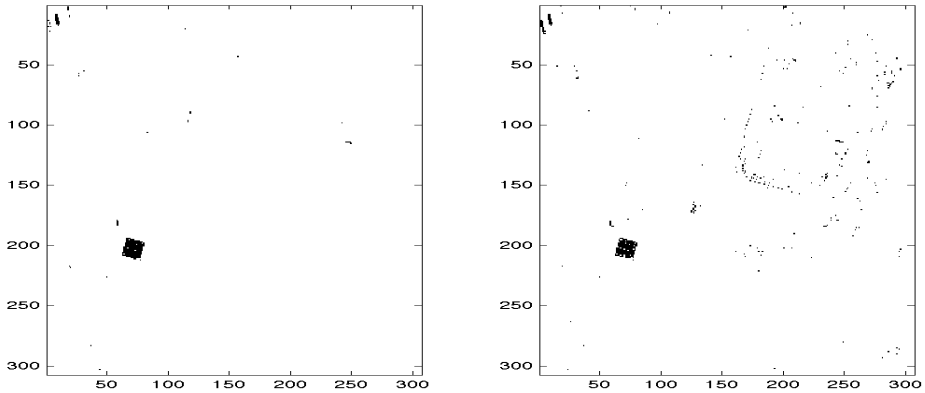


(c) Class 9

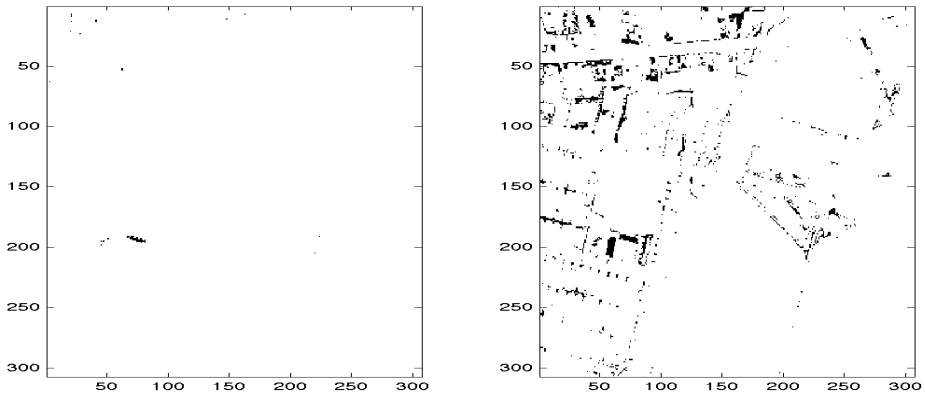
Figure 5.7: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 10

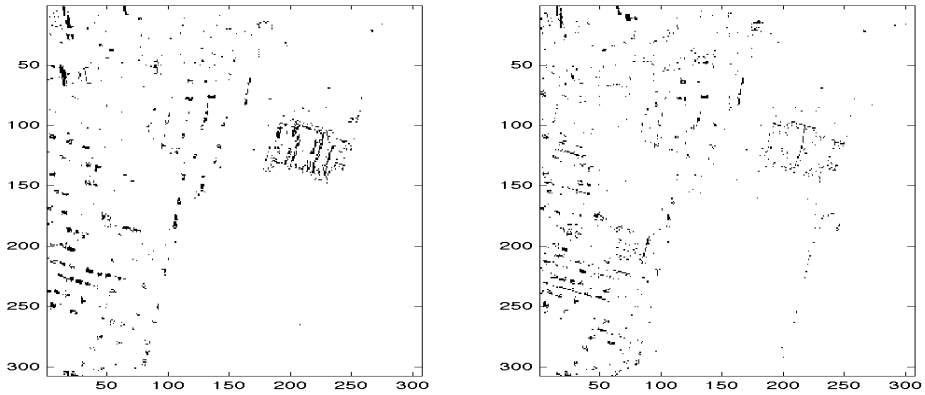


(b) Class 11

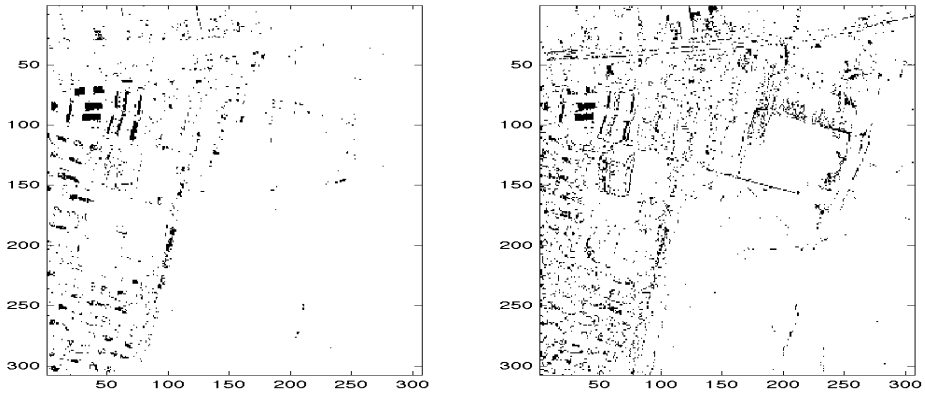


(c) Class 12

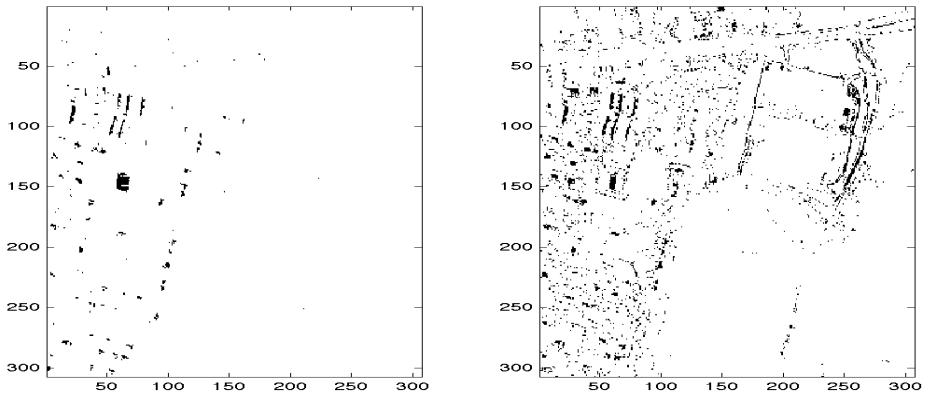
Figure 5.8: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 13

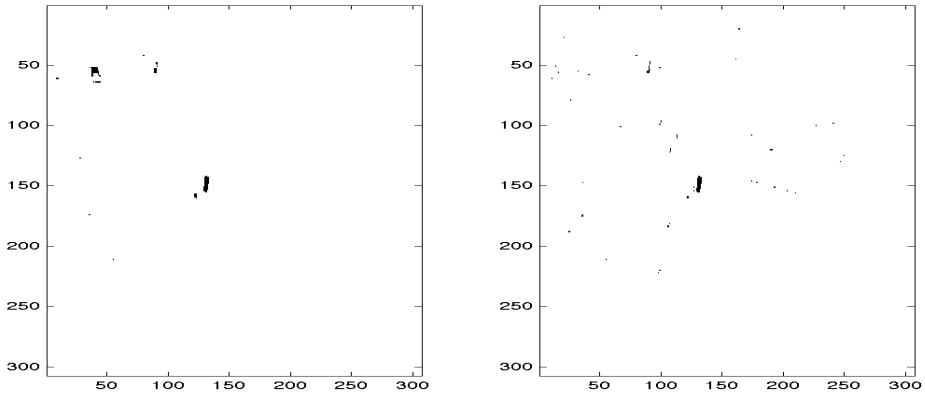


(b) Class 14

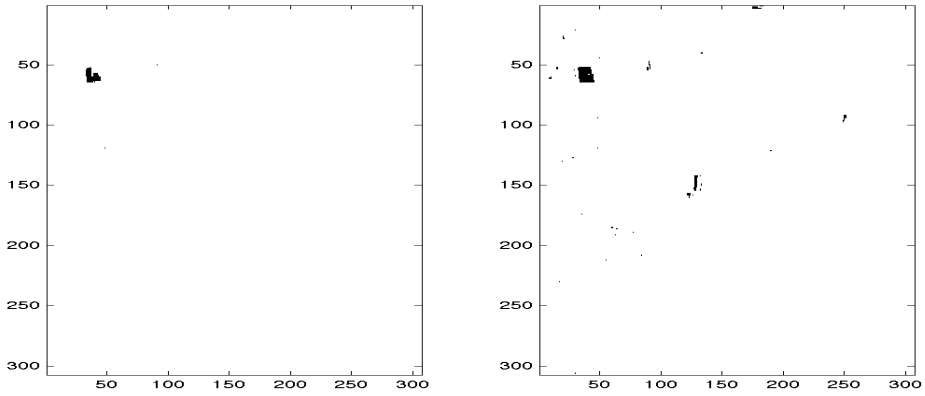


(c) Class 15

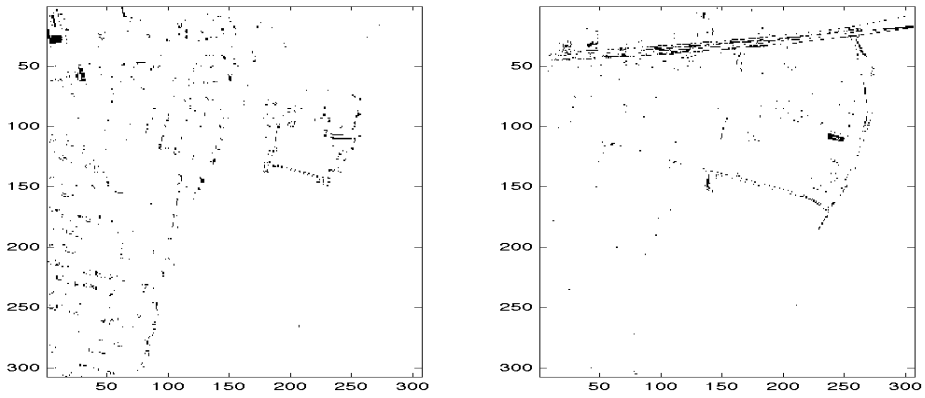
Figure 5.9: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 16

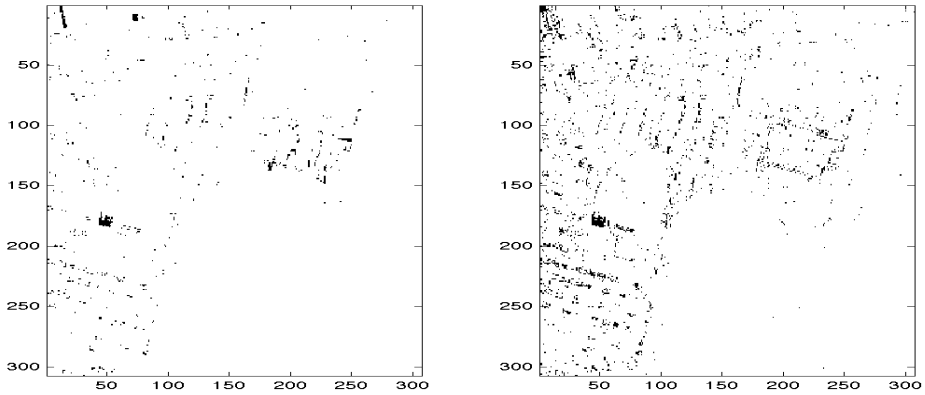


(b) Class 17

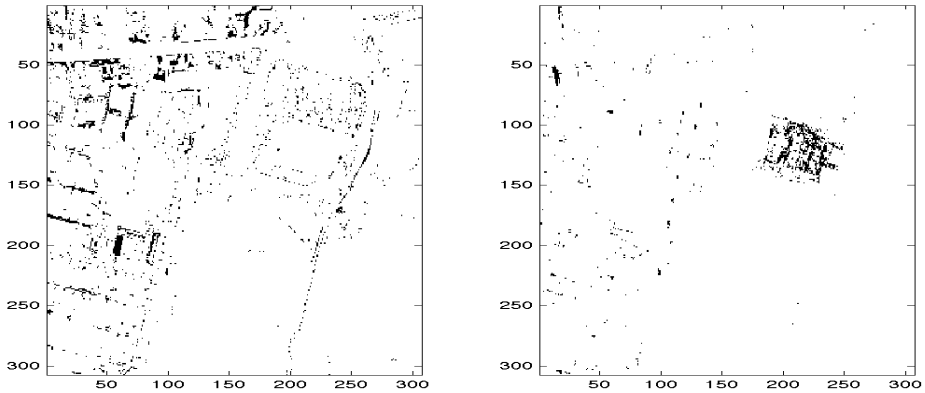


(c) Class 18

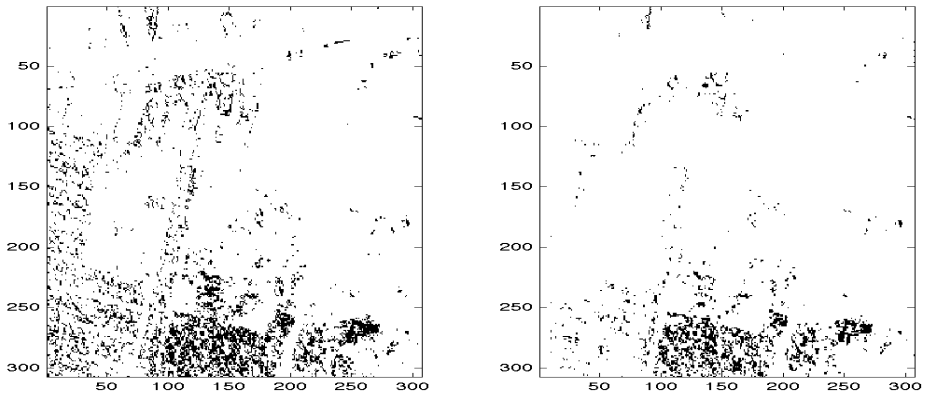
Figure 5.10: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 19



(b) Class 20



(c) Class 21

Figure 5.11: Urban: left - exact neighborhoods, right - approximate neighborhoods

Chapter 6

Integration and Data Analysis

In the last two chapters we established, theoretically and empirically, the advantages of using random projections and approximate neighborhoods with Laplacian Eigenmaps. We now wish to integrate these two methods and show that they work well together and yield a dramatic reduction in computational time with no significant reduction in accuracy. In fact, as our experiments will show, using the combined approximate algorithm, surprisingly, sometimes results in increased accuracy. We shall demonstrate this across a wide choice of parameter settings using the two sets of hyperspectral imaging data described in section 2.4.

We recall that the brute force method for constructing nearest neighbor graphs has asymptotic complexity of $\Theta(dn^2)$, where d is the dimension of the ambient space and n is the number of points. Using random projections, as described in Chapter 4, allows us to reduce d by a significant factor. For example, as we shall see in our experiments with the Urban dataset, whose points lie in $\mathbb{R}^{161}$, there seems to be no significant loss of accuracy if the points are projected down to as few as 20 dimensions. Furthermore, our experiments show that the price of computing the projection is much more than offset by the savings in the construction of the neighborhood graph. Finally, using the recursive approximate neighborhood construction introduced in Chapter 5, we may replace the exponent 2 with a much

smaller one (about 1.16 in most of the trials described below.) Putting these two methods together, we obtain the maximum possible reduction in running time while preserving accuracy, as we shall soon see.

All of our experiments will consist of material classification based on the Urban and Smith hyperspectral datasets. We will use LE to map the high-dimensional set to a lower-dimensional space, and study the effect of using the fast neighborhood construction instead of the exact method. Namely, we shall compare accuracy of classification and running time. Note that we are not trying to establish a new and competitive method for classification, but only to show that the performance of LE does not degrade by using the fast method to construct the nearest neighbor graph, and that the computational savings are substantial. Our methodology of classification will be the same as the one established in [45] and detailed in Section 2.3. In short, after mapping the vectors to a low-dimensional space, we construct a representative vector for each class by averaging all the ground truth vectors in that class. We classify each pixel by computing the angle between the vector it was mapped to and each of the class representatives, and choosing the class that minimizes this angle.

6.1 Hardware Specifications

All of the computations were performed on a Mac OS X with the following specifications:

- Processor: 2 x 2.26 GHz Quad Core Intel Xeon,

- Memory: 16 GB 1066 MHz DDR3.

6.2 Urban

We begin our empirical evaluation with the Urban data set. Recall from Section 2.4 that this set consists of $307 \times 307 = 94249$ pixels and 161 bands. After optimizing each of the algorithms (exact and approximate classification as described above) using an exhaustive search over a grid within a reasonable range, we obtain the results in Table 6.1. The parameters used for this comparison are the following:

Exact algorithm:

- number of neighbors = 12,
- kernel bandwidth = 0.5,
- reduced dimension = 55.

Approximate algorithm:

- number of neighbors = 8,
- kernel bandwidth = 0.5,
- reduced dimension = 55,
- dimension of random projection = 80,
- overlap = 0.1.

Table 6.1: Comparison of performance on Urban

Method	Time (min)	Accuracy (percent)
exact	35.3	77.25
approximate	5.8	78.54

As we can see, the reduction in computational time is dramatic and the accuracy of classification does not degrade. It seems that the slight increase in accuracy can be explained as follows. Computing the adjacency graph using the approximate algorithm incurs small, random errors. These errors mean that, occasionally, points which are distant will be designated as neighbors. This makes the largest connected component in the graph, which is the only one for which an embedding is actually computed, consistently larger than it is when the exact method is used. In fact, typically, for the Urban dataset, it contains all the points, whereas about 2700 are omitted when the exact method is used. Therefore, we obtain a better approximation of the Laplace-Beltrami operator on the assumed underlying manifold, which, in turn, leads to a mapping that is apparently better in preserving geometry.

In Figure 6.1 we can see a comparison of classification results by class. Figures 6.2 - 6.8 show a comparison of maps by class. We can see that most classes look very similar. In Figure 6.9 we see a few of the eigenfunctions corresponding to the smallest eigenvalues, and how well they separate the different materials present in the scene.

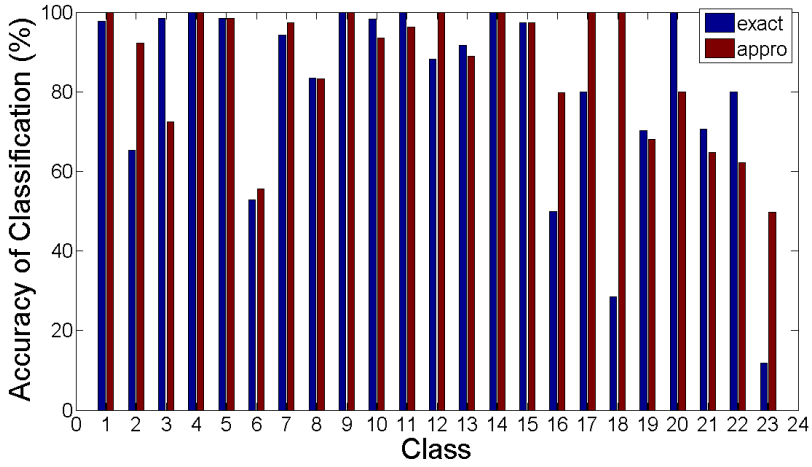


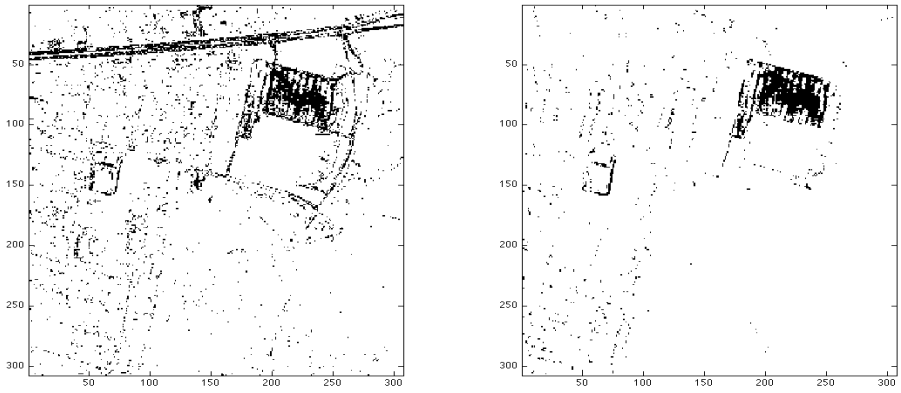
Figure 6.1: Classifying Urban using LE with exact and approximate neighborhoods

6.3 Numerical Analysis of Parameters

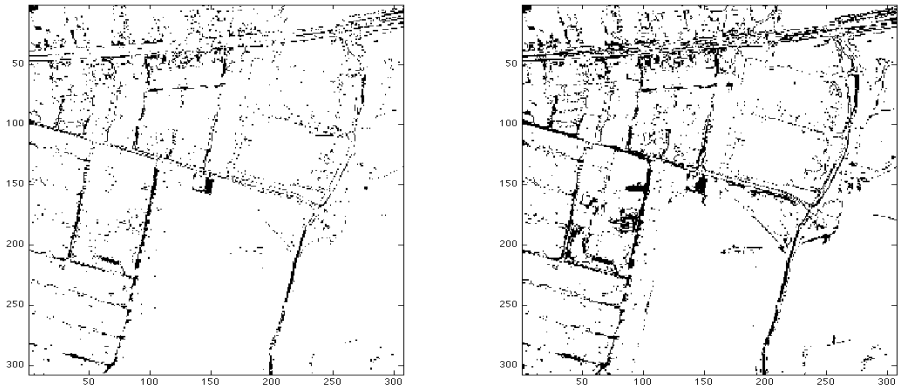
We now isolate each of the user specified parameters that may affect the algorithm's performance in order to gain a deeper understanding of its significance and identify optimal working values. Since the algorithm involves a random element, namely, the use of a random matrix for projection, for each fixed set of parameters, at least three trials are performed and their results are averaged. In this section we continue to work with the Urban data set.

6.3.1 Number of Neighbors

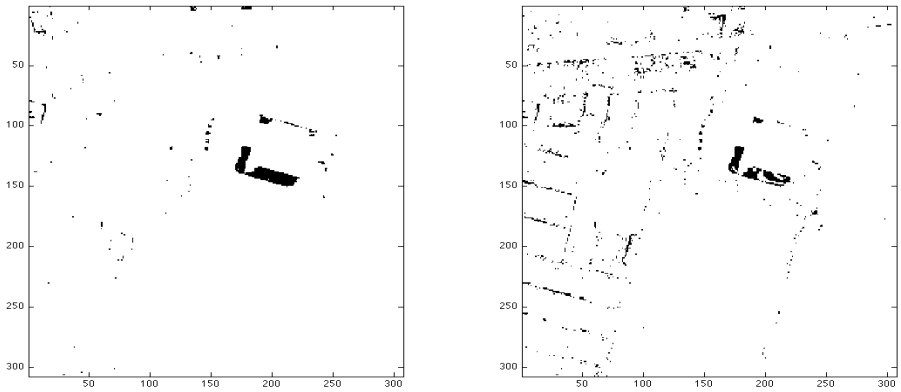
We begin with the number of neighbors that is used in constructing the nearest neighbor graph. We consider values in the range 4 - 24, and the results are presented in Figure 6.10. As far as accuracy of classification, we can see that the algorithm is not very sensitive to the value of this parameter. However, running time increases at least linearly with this parameter and more than doubles in this range. We



(a) Class 1

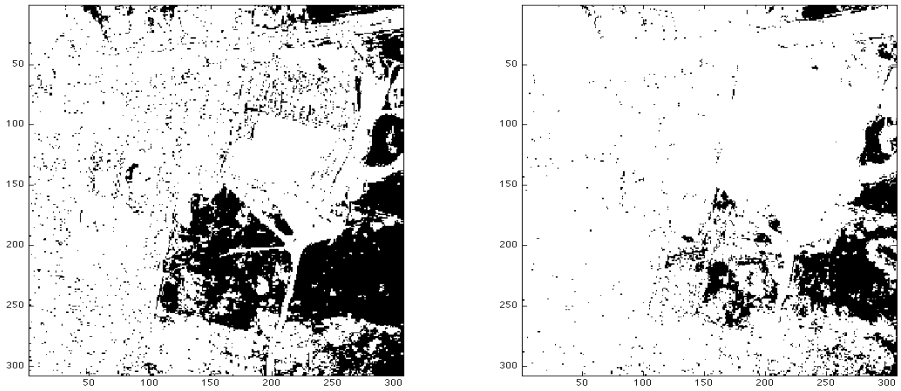


(b) Class 2

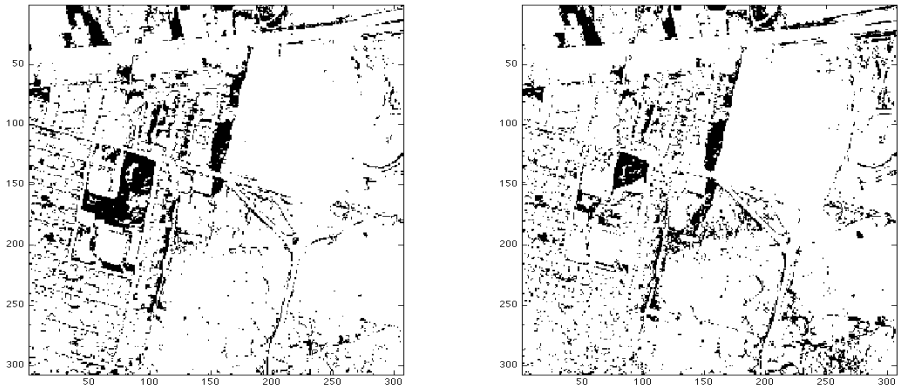


(c) Class 3

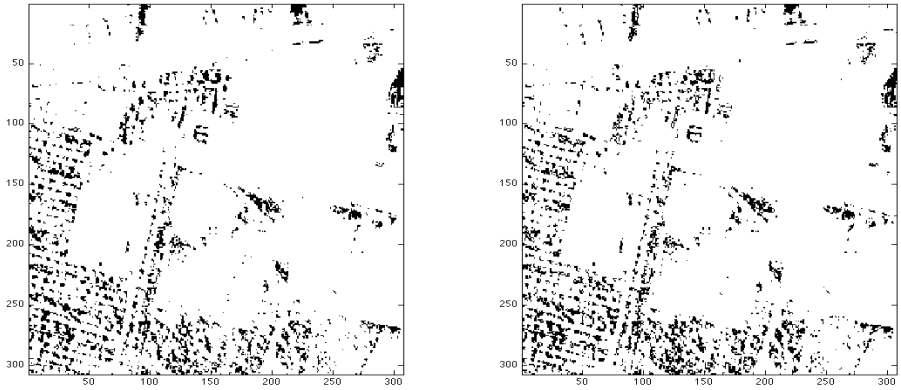
Figure 6.2: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 4

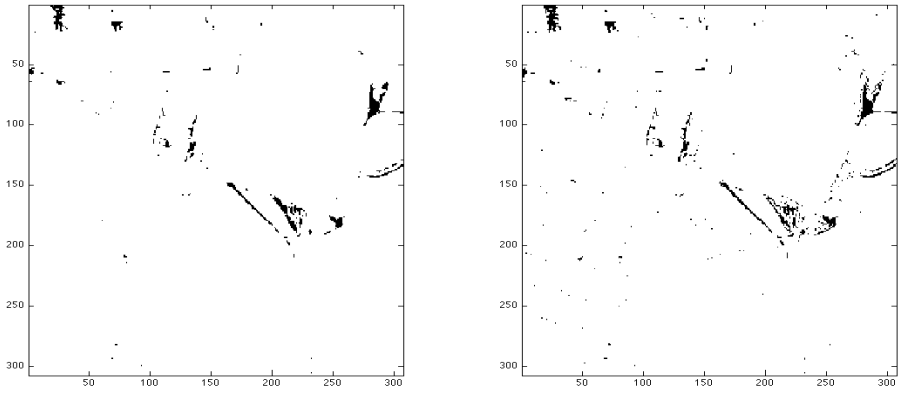


(b) Class 5

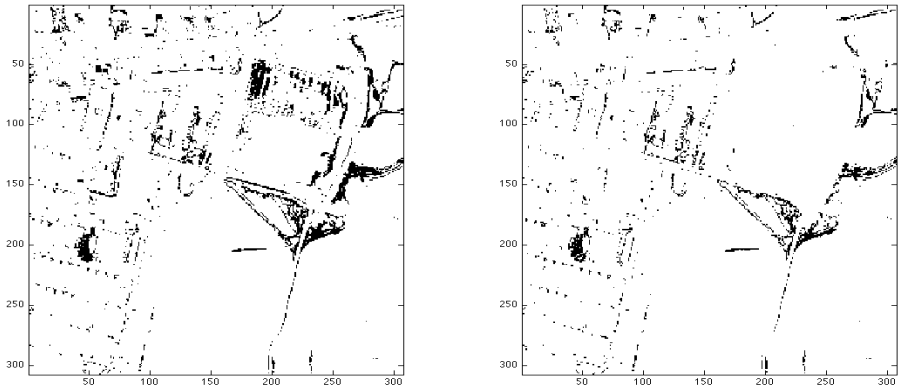


(c) Class 6

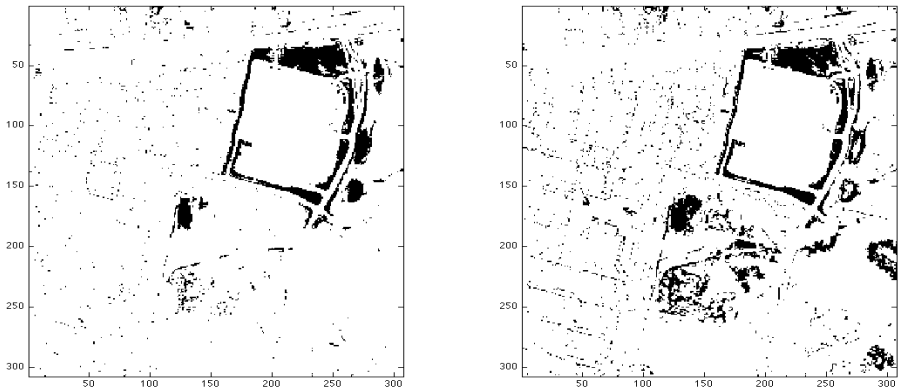
Figure 6.3: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 7

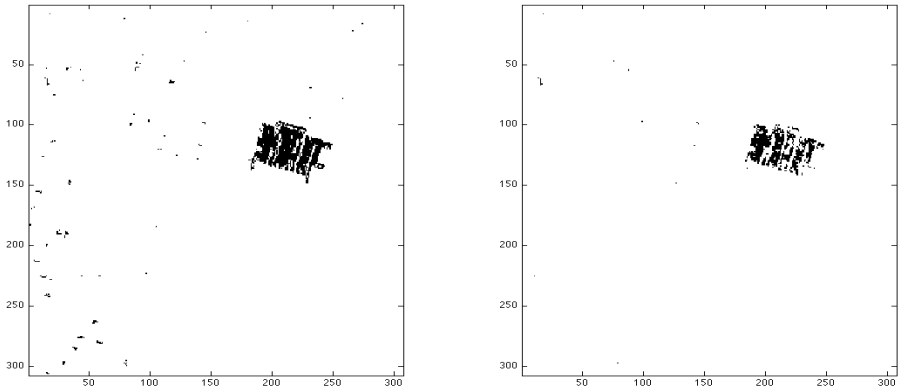


(b) Class 8

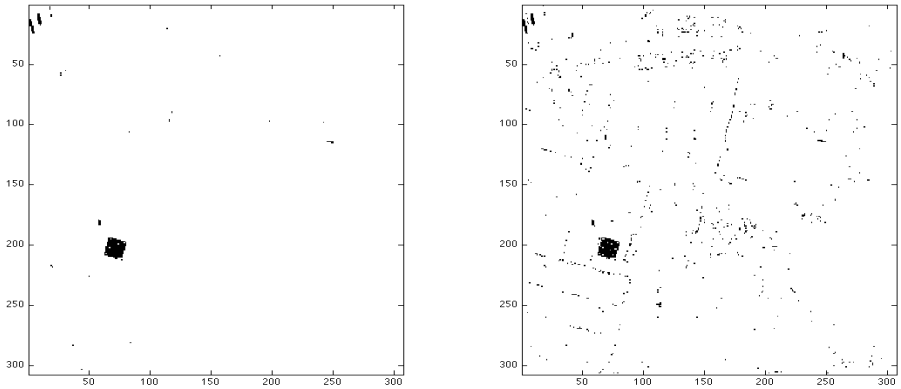


(c) Class 9

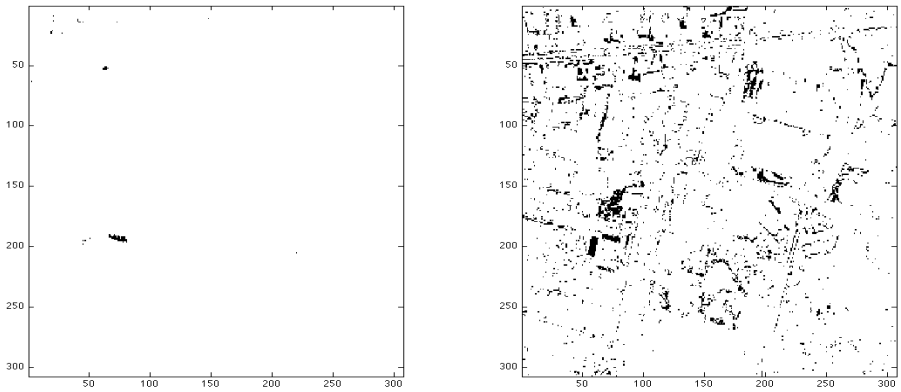
Figure 6.4: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 10

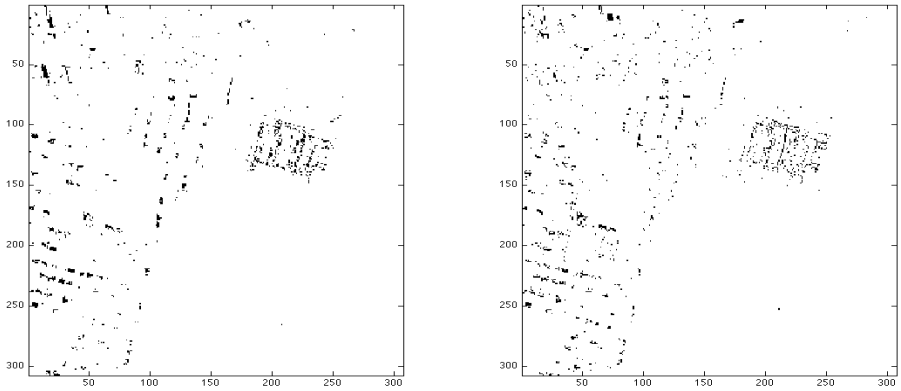


(b) Class 11

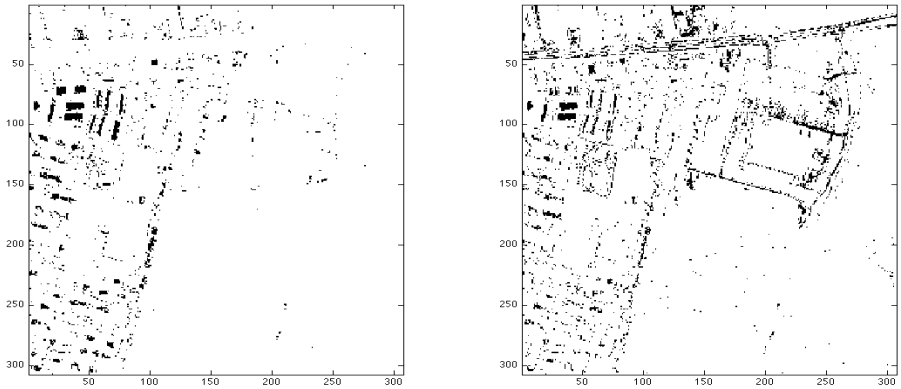


(c) Class 12

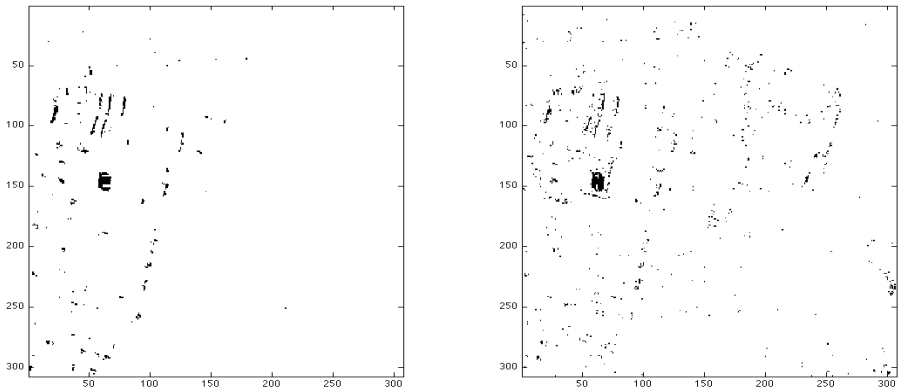
Figure 6.5: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 13

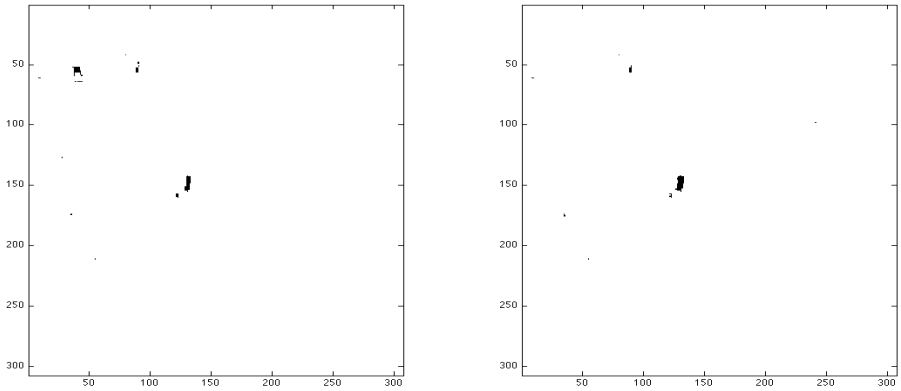


(b) Class 14

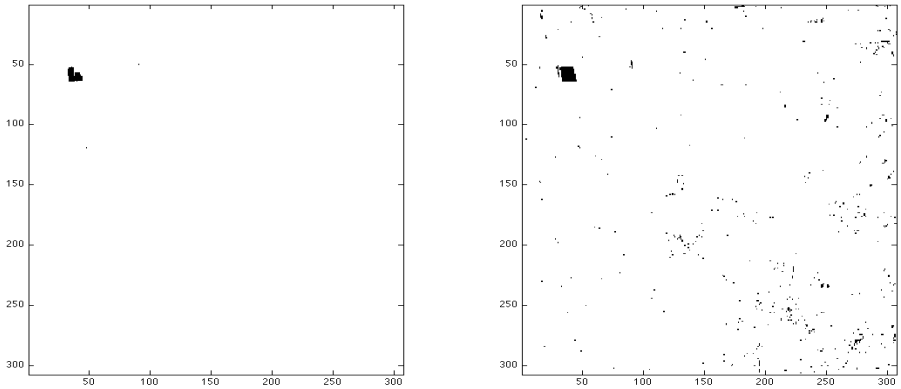


(c) Class 15

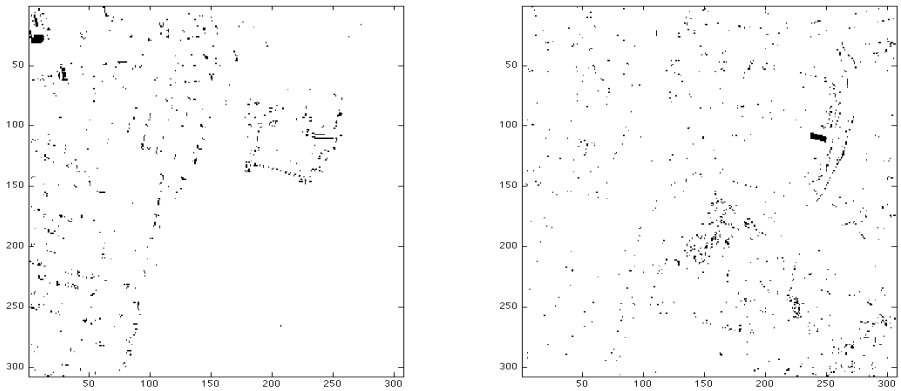
Figure 6.6: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 16

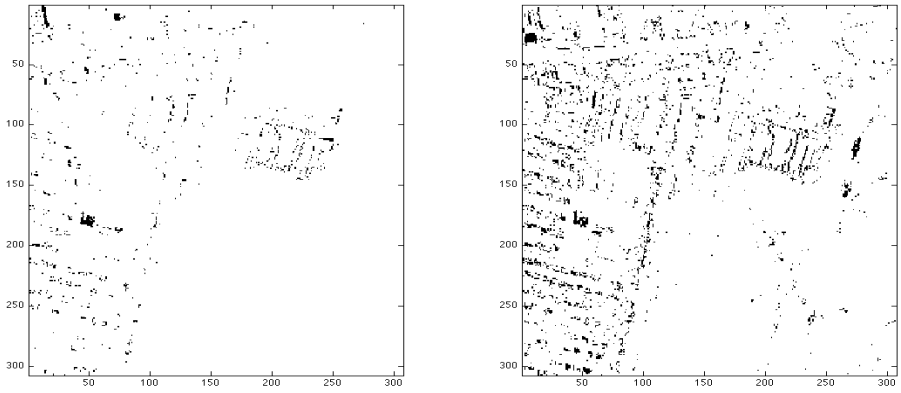


(b) Class 17

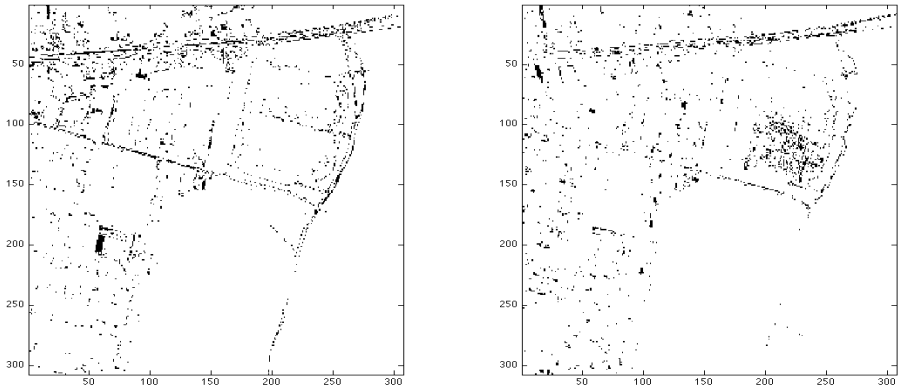


(c) Class 18

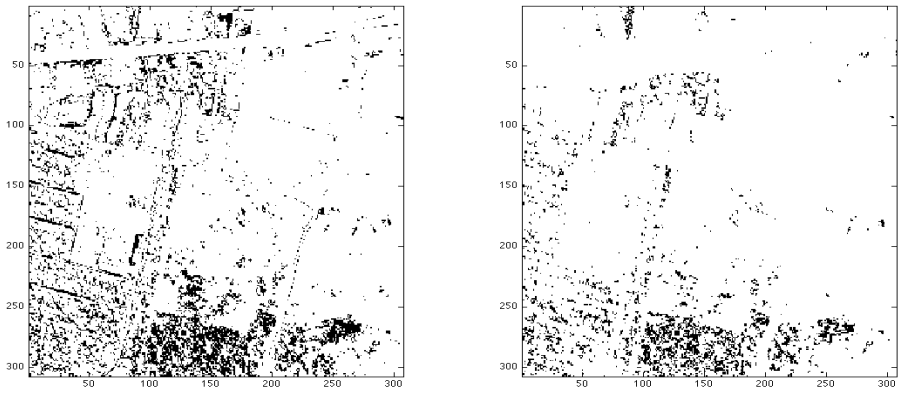
Figure 6.7: Urban: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 19



(b) Class 20



(c) Class 21

Figure 6.8: Urban: left - exact neighborhoods, right - approximate neighborhoods

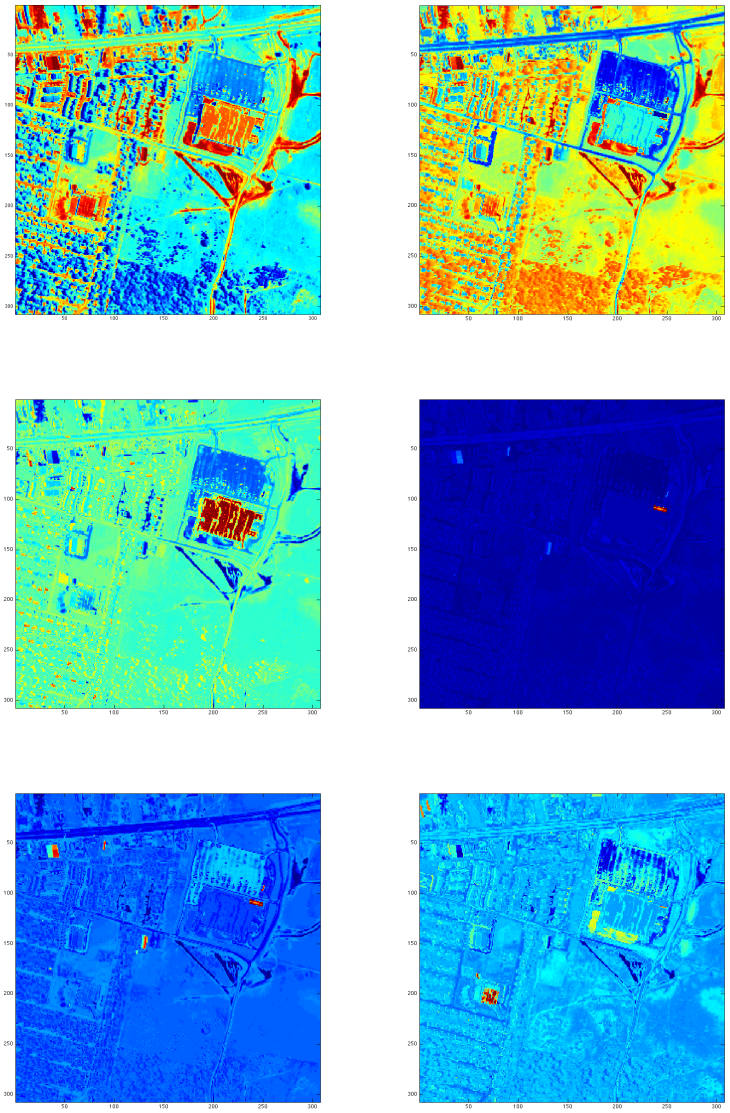


Figure 6.9: Urban: a few eigenvectors

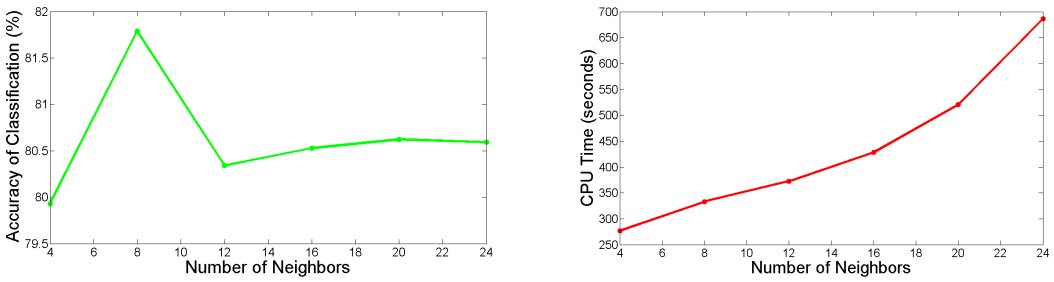


Figure 6.10: Accuracy and running time as functions of the number of neighbors

recall that this number affects not only the time needed to construct the adjacency graph, but also how sparse it is, which, in turn, affects the time needed to solve the associated eigenvalue problem.

Fixed parameters:

- kernel bandwidth = 0.5,
- reduced dimension = 55,
- dimension of random projection = 80,
- overlap = 0.3.

6.3.2 Kernel Bandwidth

The kernel bandwidth (denoted by t in section 2.2.3) determines the penalty associated with the distance between the neighbors. When this parameter is assigned a small value, the effect of distant neighbors on the computed mapping will be negligible. As Figure 6.11 shows, beyond a value of 0.1 both accuracy and time

fluctuate in a relatively small range and the exact value assigned to this parameter does not seem to have a significant effect on either one.

Fixed parameters:

- number of neighbors = 8,
- reduced dimension = 40,
- dimension of random projection = 80,
- overlap = 0.3.

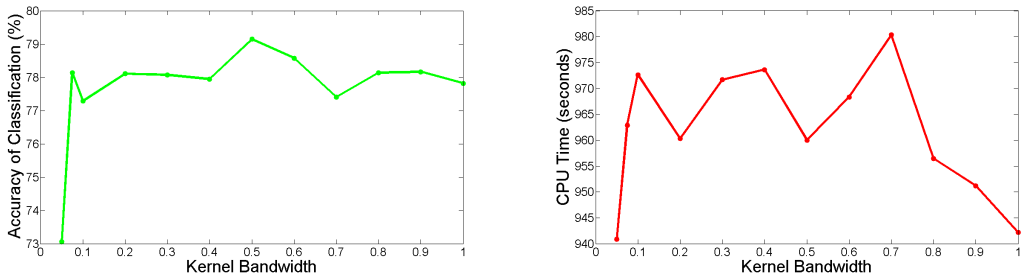


Figure 6.11: Accuracy and running time as functions of the kernel bandwidth

6.3.3 Reduced Dimension

Since we have no way of knowing what the dimension of the assumed underlying manifold is, we must specify the dimension of the space to which we map the points. We test values in the range 5 - 70 and present the results in Figure 6.12. While there is an expected increase in computational time arising from the necessity of computing more eigenvectors, it is not significant. In contrast, we note a steady

increase in accuracy, which tapers off around 55. This may suggest that this is roughly the intrinsic dimension of the manifold.

Fixed parameters:

- number of neighbors = 8,
- kernel bandwidth = 0.5,
- dimension of random projection = 80,
- overlap = 0.3.

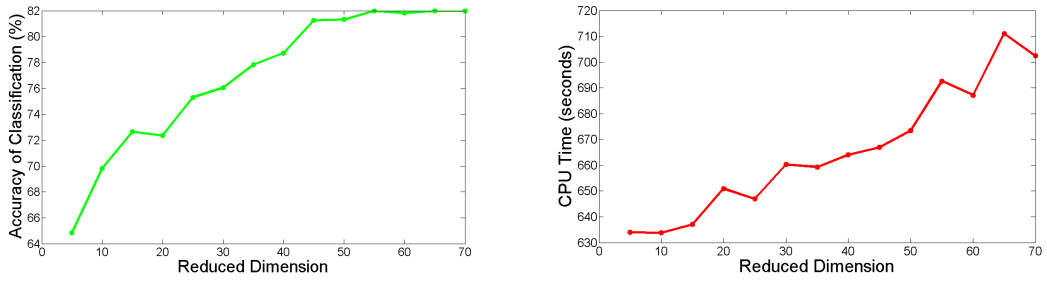


Figure 6.12: Accuracy and running time as functions of the reduced dimension

6.3.4 Dimension of Projection

We now study the role of the dimension of the random projection, denoted by M in Section 4.3, using values in the range 2 - 50. Figure 6.13 shows that once we reach a value of about 10, two things seem to happen. First, further running times are hardly affected (the marginal cost of computing the projection is negligible compared with the rest of the computation). Second, the effect on accuracy seems

to be negligible, except for what seems like random fluctuation. This is interesting in as far as it seems to contradict our previous conclusion: If the intrinsic dimension is really around 55, we would expect to lose a significant amount of information when projecting to a much lower dimension such as 10.

Fixed parameters:

- number of neighbors = 8,
- kernel bandwidth = 0.5,
- reduced dimension = 40,
- Overlap = 0.3.

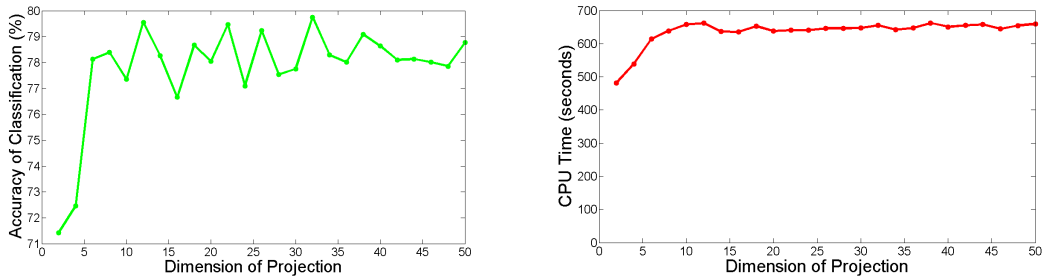


Figure 6.13: Accuracy and running time as functions of dimension of projection

6.3.5 Overlap

The final parameter we consider determines the amount of overlap between the two sets that arise after each bisection, as detailed in section 5.3, where it was denoted by α. Naturally, we expect both the running time and the accuracy to

increase with α, as confirmed by Figure 6.14. We recall that α determines the exponent a in the overall complexity $\Theta(dn^a)$. In fact, Table 6.2 gives the precise values predicted by the theoretical analysis in [23]. Compared with the steep (exponential) increase in running time, we note only a moderate improvement in accuracy. In fact, the low value of 0.1 seems to offer an excellent tradeoff, especially for a large dataset like Smith, to which we turn now.

Table 6.2: The exponent a for different values of α

α	0.05	0.1	0.15	0.2	0.25	0.3	0.35
a	1.08	1.16	1.25	1.36	1.47	1.61	1.76

Fixed parameters:

- number of neighbors = 8,
- kernel bandwidth = 0.5,
- reduced dimension = 40,
- dimension of projection = 80.

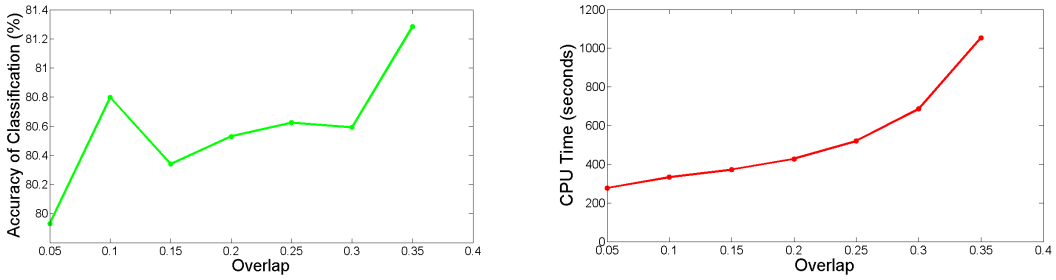


Figure 6.14: Accuracy and running time as functions of the overlap

6.4 Smith

We now present the results of a comparison using the Smith dataset. We recall that this data set is much larger than Urban and consists of $679 \times 944 = 640976$ pixels and 110 wavelengths. However, in order to bring down the running times from many hours to more reasonable periods, only a wide diagonal band was used, consisting of 399,611 pixels. The following parameters were used:

Exact algorithm:

- number of neighbors = 12,
- kernel bandwidth = 0.5,
- reduced dimension = 55.

Approximate algorithm:

- number of neighbors = 8,
- kernel bandwidth = 0.5,
- reduced dimension = 55,
- dimension of random projection = 80,
- overlap = 0.1.

The results are presented in Table 6.3. Once again, the approximate algorithm yields better accuracy in drastically lower time. Figure 6.15 shows a comparison of

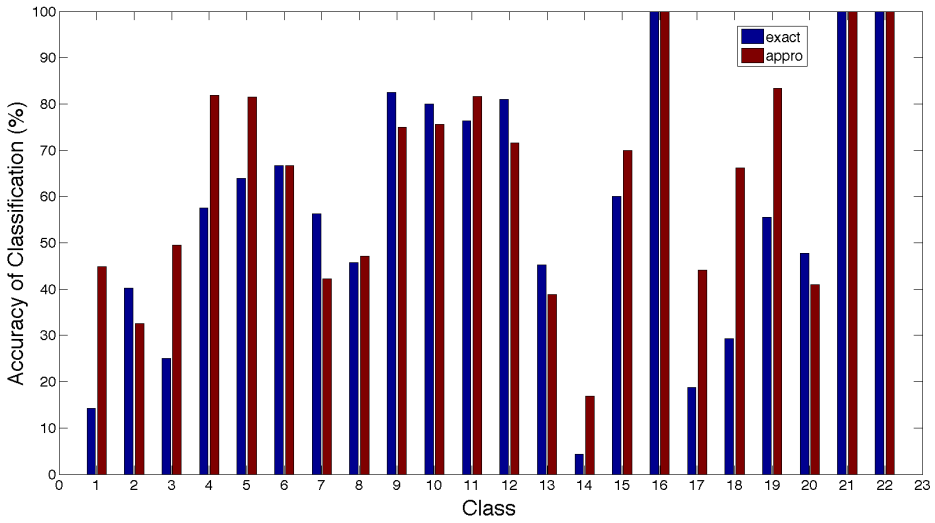
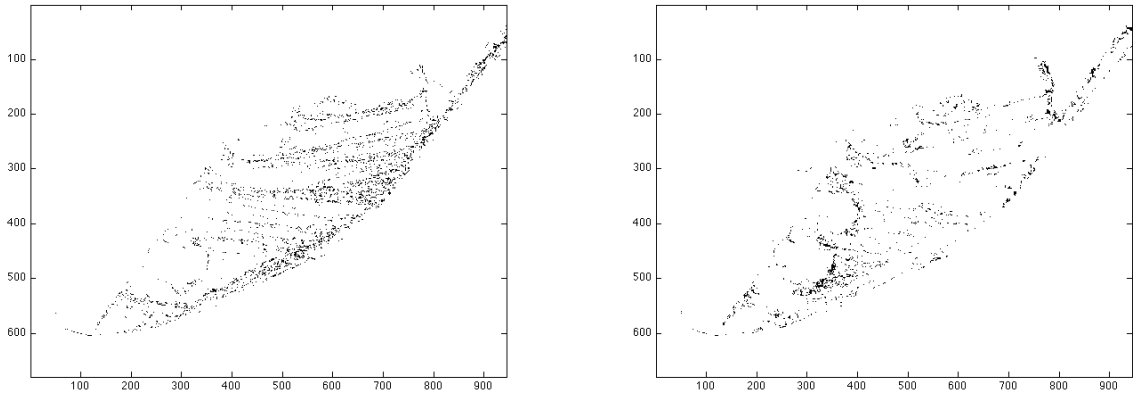


Figure 6.15: Classifying Smith using LE with exact and approximate neighborhoods

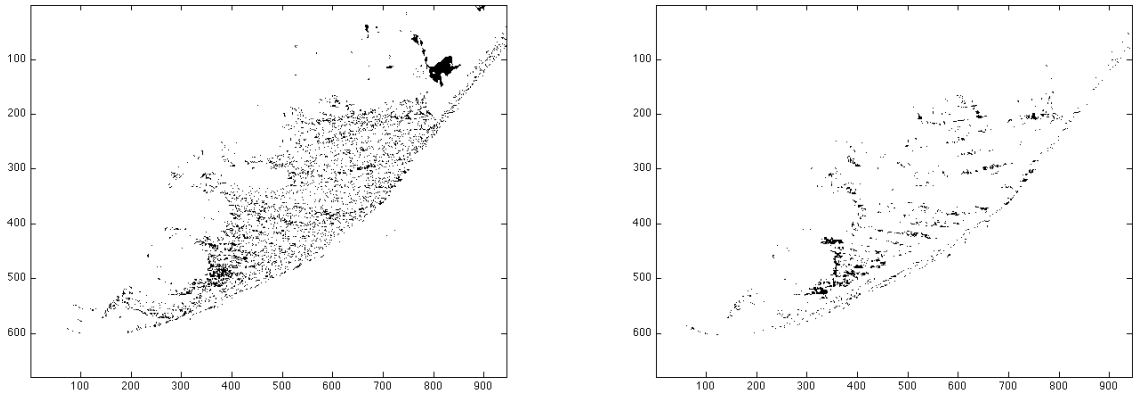
accuracy by class. Figures 6.16 - 6.22 show a comparison of class maps. Figure 6.23 shows a few of the eigenvectors corresponding to the lowest eigenvalues, suggesting a reasonable separation of materials in the scene.

Table 6.3: Comparison of performance on Smith

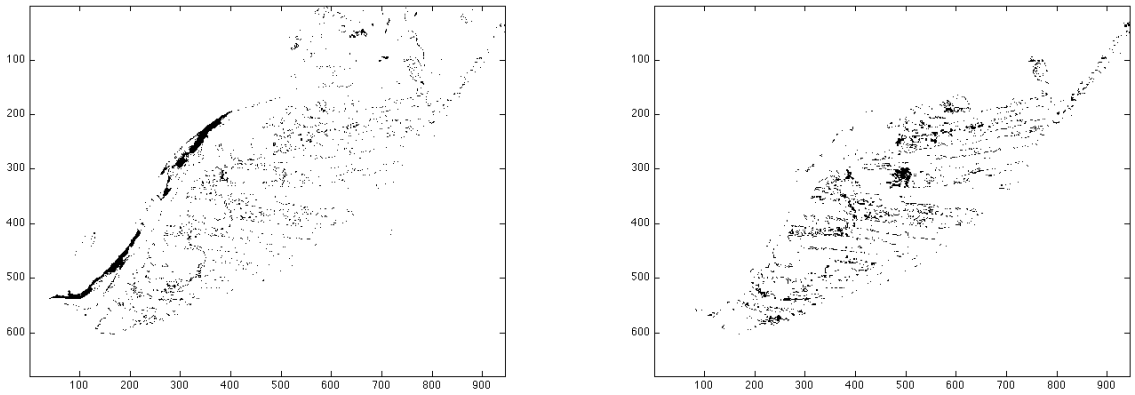
Method	Time (min)	Accuracy (percent)
LE - exact neighborhood	534.3	50.20
LE - approximate neighborhood	69.4	58.24



(a) Class 1

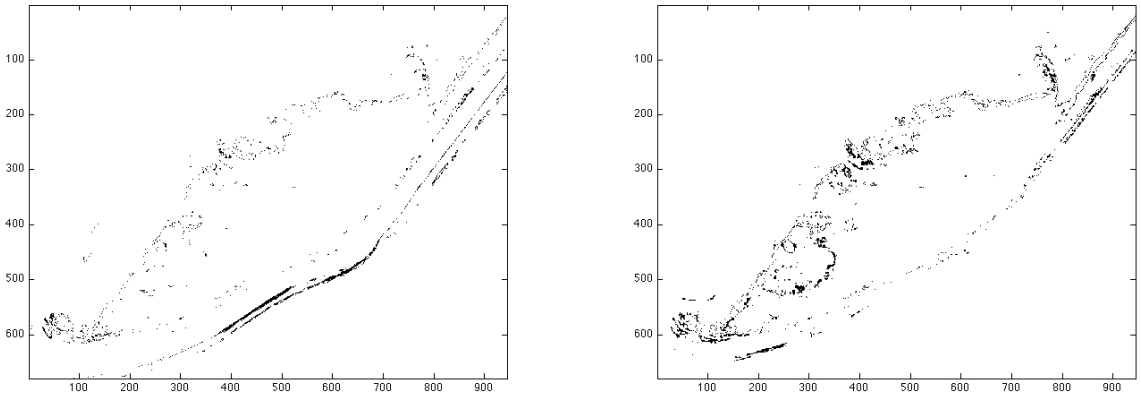


(b) Class 2

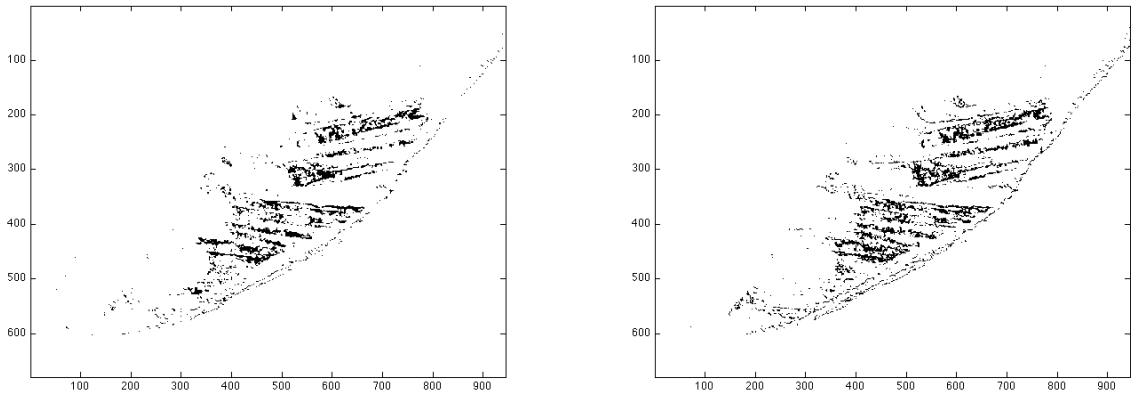


(c) Class 3

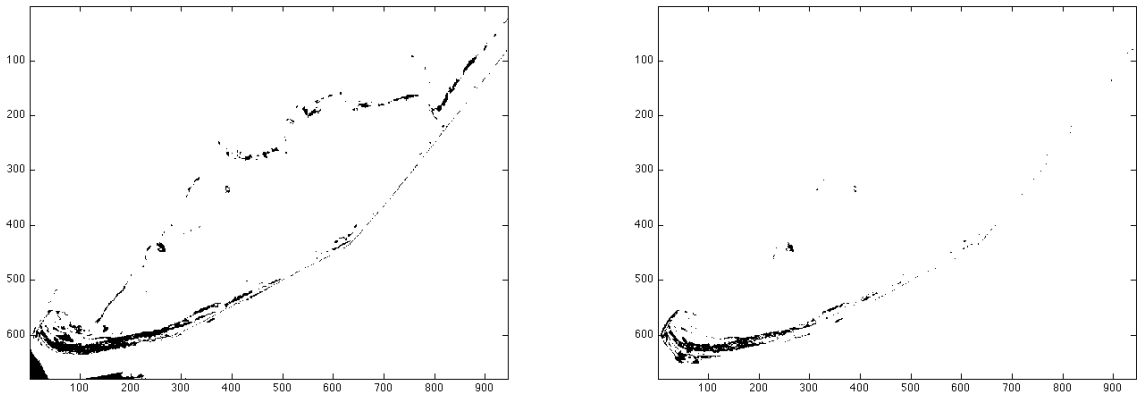
Figure 6.16: Smith: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 4

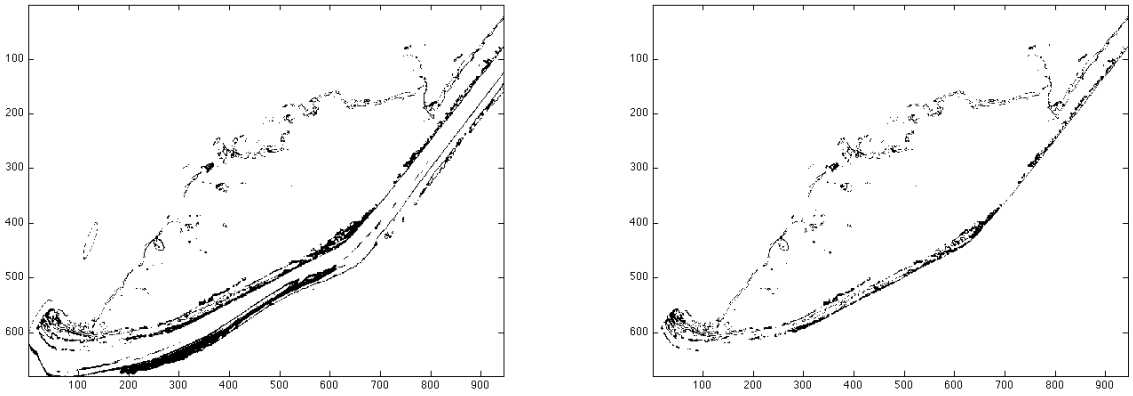


(b) Class 5

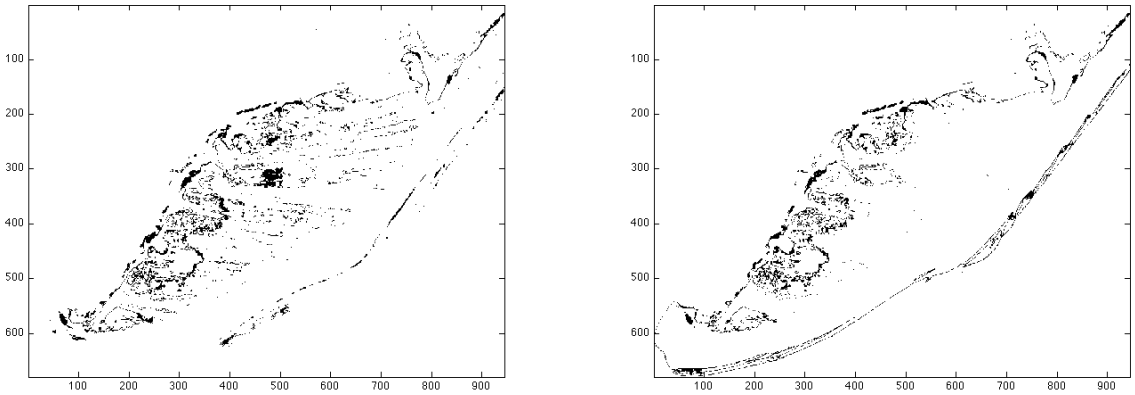


(c) Class 6

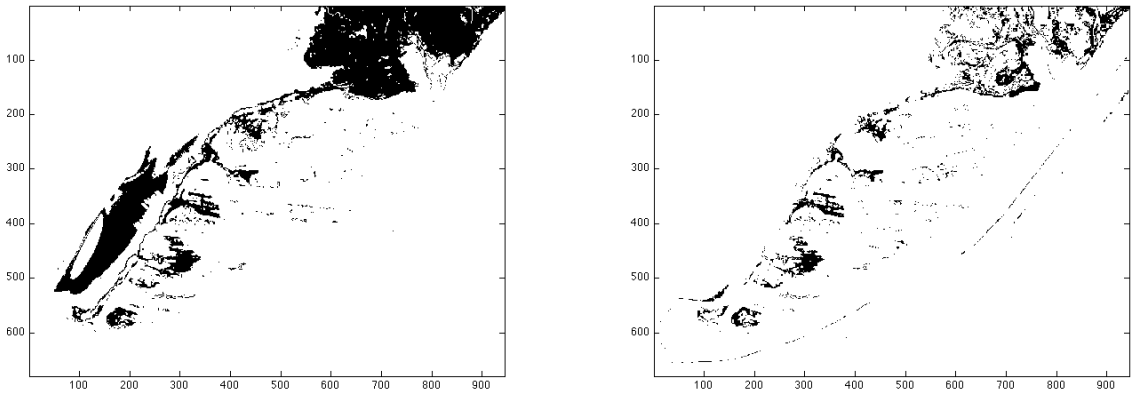
Figure 6.17: Smith: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 7

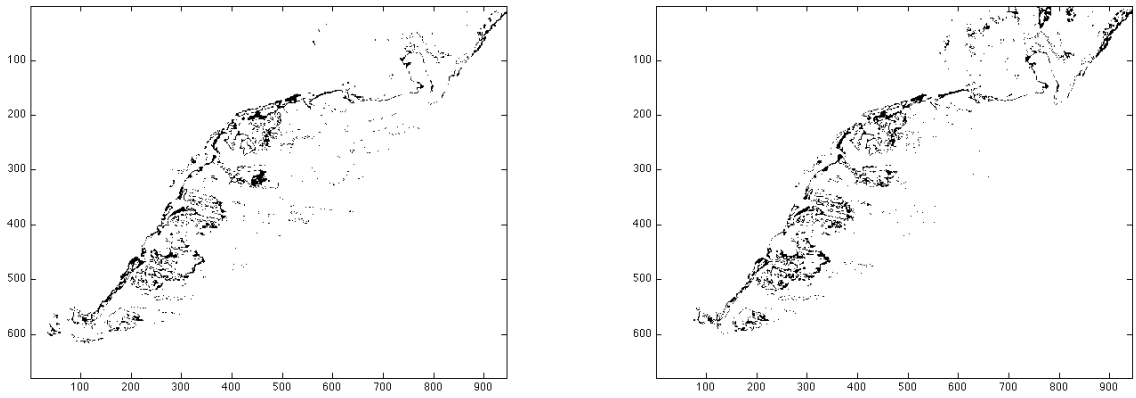


(b) Class 8

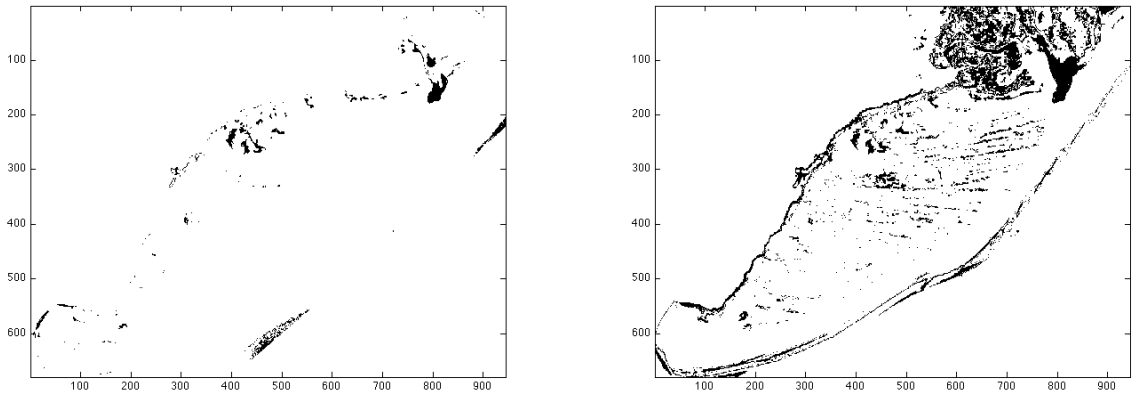


(c) Class 9

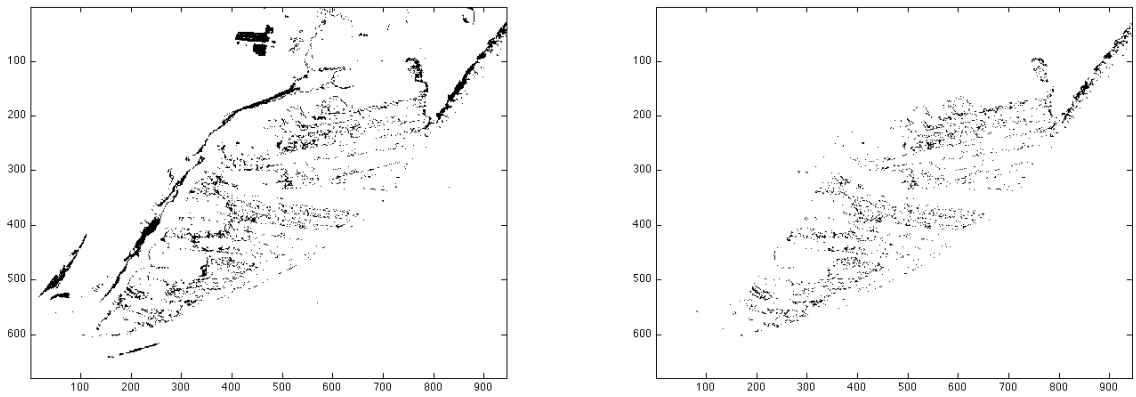
Figure 6.18: Smith: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 10

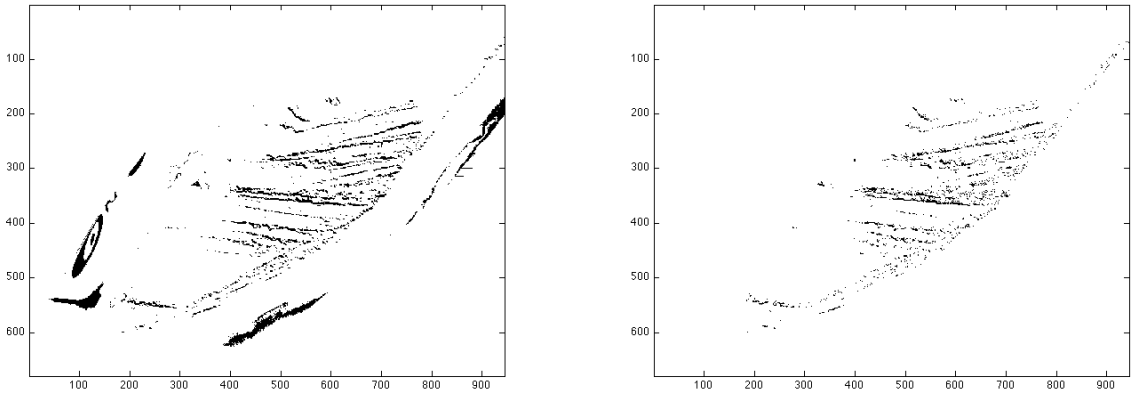


(b) Class 11

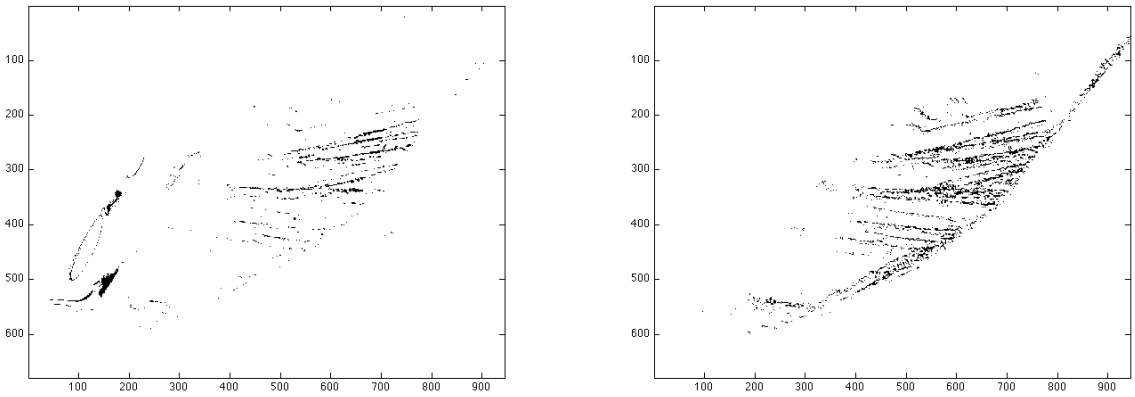


(c) Class 12

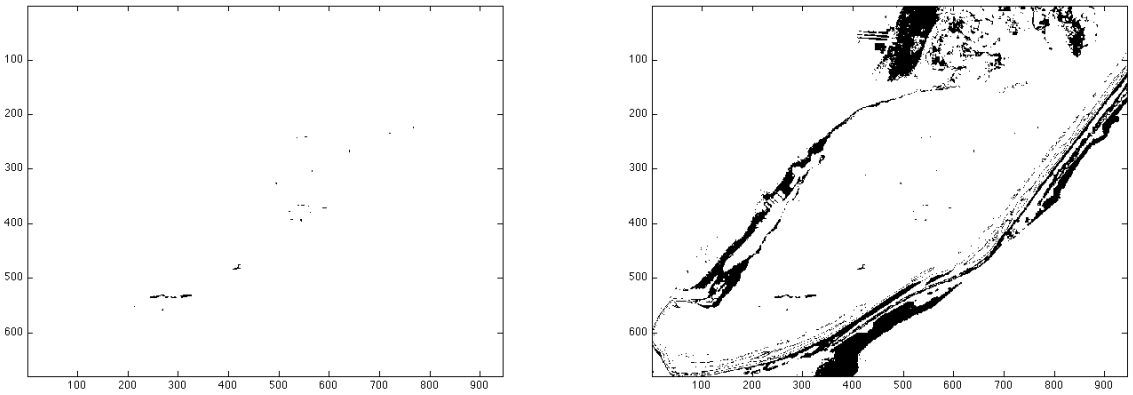
Figure 6.19: Smith: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 13

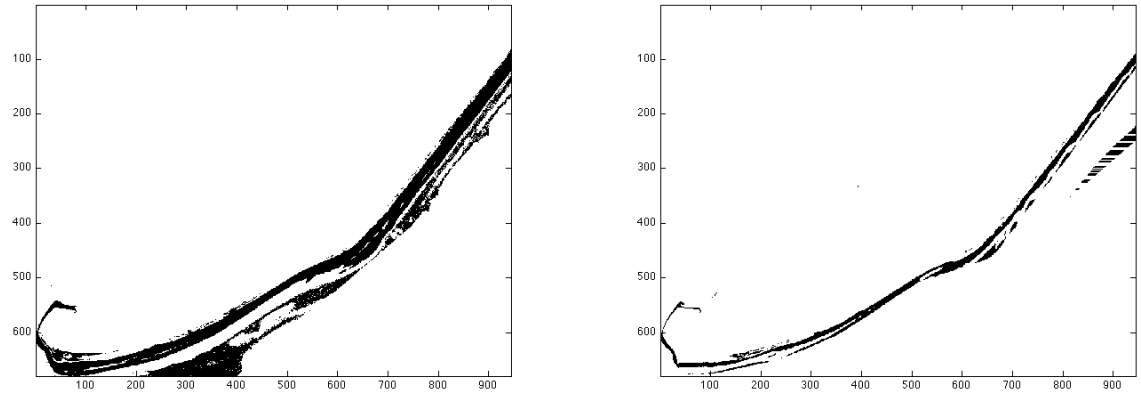


(b) Class 14

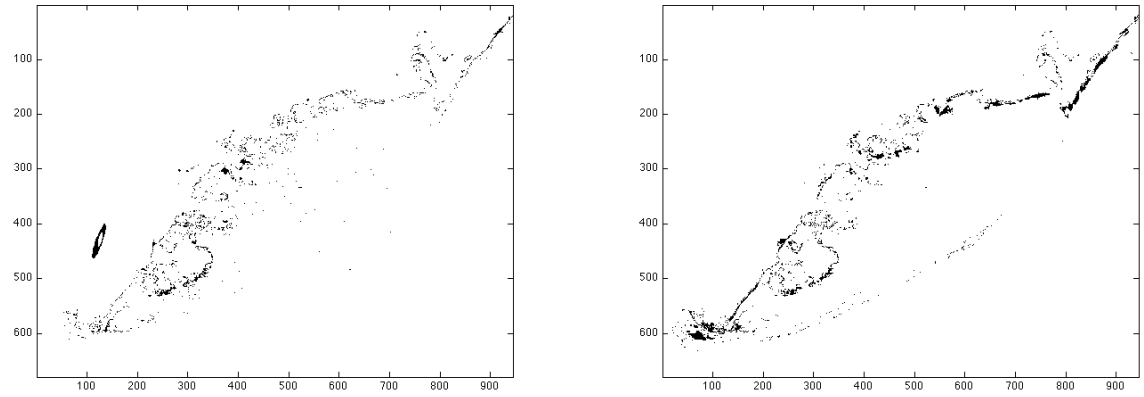


(c) Class 15

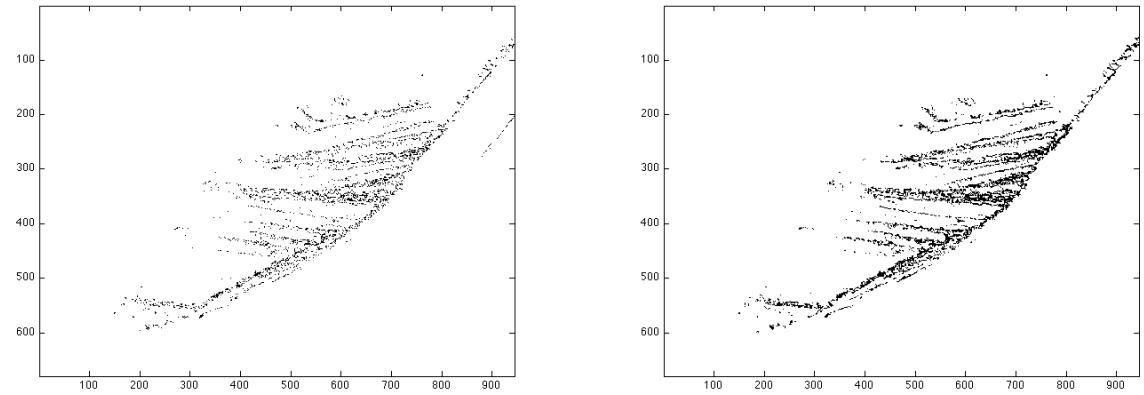
Figure 6.20: Smith: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 16

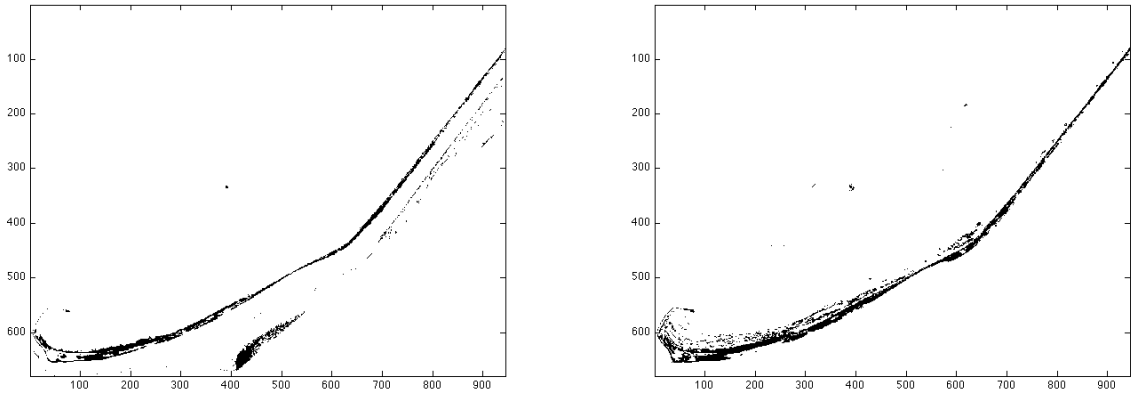


(b) Class 17

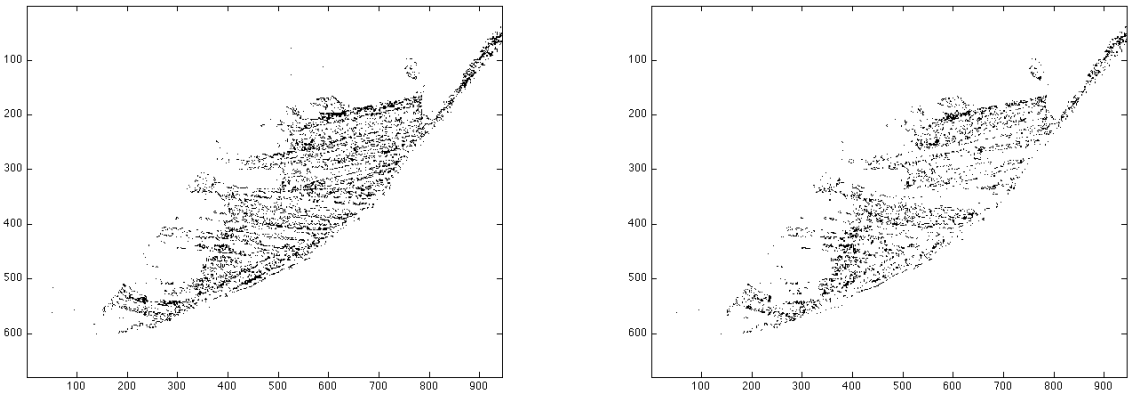


(c) Class 18

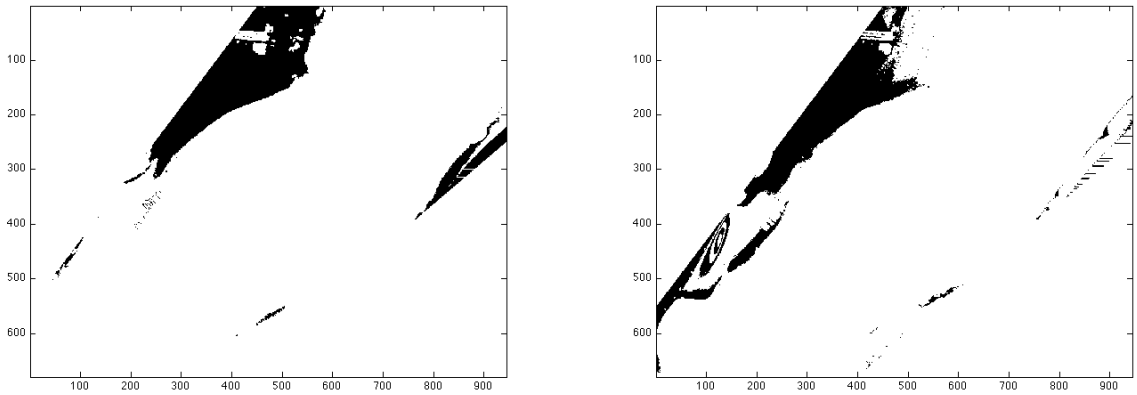
Figure 6.21: Smith: left - exact neighborhoods, right - approximate neighborhoods



(a) Class 19



(b) Class 20



(c) Class 21

Figure 6.22: Smith: left - exact neighborhoods, right - approximate neighborhoods

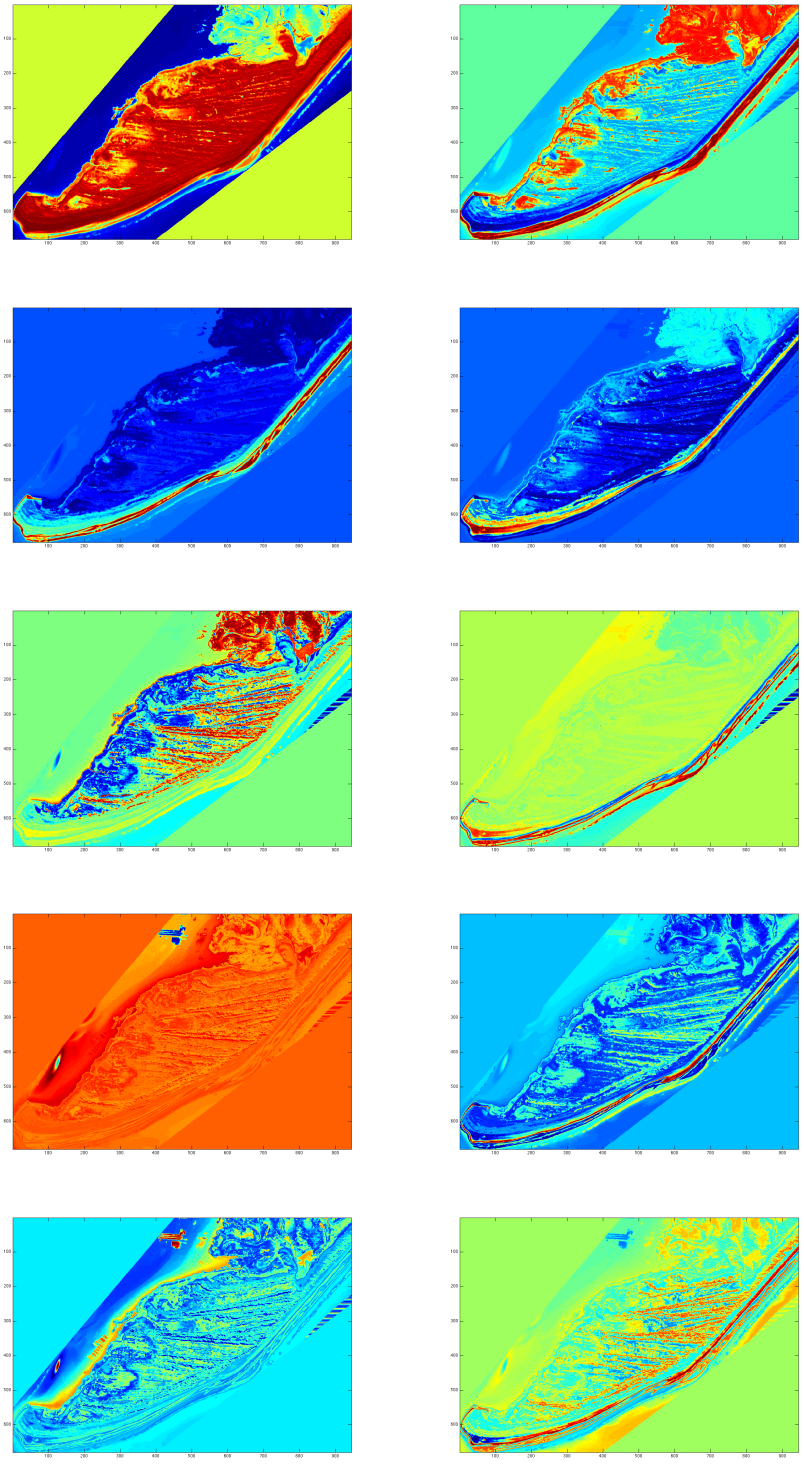


Figure 6.23: Smith: selected eigenvectors

Bibliography

[1] Arya, S., Mount, D., Netanyahu, N., Silverman, R., and Wu, A. *An optimal algorithm for approximate nearest neighbor searching in fixed dimensions*, Journal of the ACM, 45(6), 891-923, 1998.

[2] AVIRIS Free Data, Jet Propulsion Lab, California Institute of Technology. Available: http://aviris.jpl.nasa.gov/html/aviris.freedata.html.

[3] Bachmann, C. M., Ainsworth, T. L., and Fusina, R. A. *Improved manifold coordinate representations of large scale hyperspectral imagery*, IEEE Trans. on Geoscience and Remote Sensing, 44(10), 27862803, 2006.

[4] Bachmann, C. M., Ainsworth, T. L., and Fusina, R. A. *Exploiting manifold geometry in hyperspectral imagery*, IEEE Trans. on Geoscience and Remote Sensing, 43(3), 441454, 2005.

[5] Bachmann, C. M. *Improving the performance of classifiers in high-dimensional remote sensing applications: An adaptive resampling strategy for error-prone exemplars (aresepe)*, IEEE Trans. on Geoscience and Remote Sensing, 41(9), 21012112, 2003.

[6] Bachmann, C. M., Donato, T. F., Lamela, G. M., Rhea, W. J., Bettenhausen, M. H., Fusina, R. A., Du Bois, K., Porter, J. H., and Truitt, B. R. *Automatic classification of land-cover on smith island, va using hymap imagery*, IEEE Trans. on Geoscience and Remote Sensing, 40(10), 23132330, 2002.

[7] Baraniuk, R., Davenport, M., DeVore, R., and Wakin, M. *A simple proof of the restricted isometry property for random matrices*, Constr. Approx., 28(3), 253-263, 2007.

[8] Baraniuk, R., Wakin, M. *Random Projections of Smooth Manifolds*, Foundations of Computational Mathematics, 9(1), 51-77, 2009.

[9] Belkin, M., Niyogi, P. *Laplacian Eigenmaps for Dimensionality Reduction and Data Representation*, Neural Computation, 13, 1373-1396, 2003.

[10] Belkin, M., Niyogi, P. *Towards a Theoretical Foundation for Laplacian-Based Manifold Methods*, COLT, 486-500, 2005.

[11] Belkin, M., Sun, J., and Wang, Y. *Discrete Laplace Operator on Meshed Surfaces*, Proceedings of the twenty-fourth annual symposium on computational geometry, 278-287, 2008.

[12] Belkin, M., Niyogi, P. *Convergence of Laplacian Eigenmaps*, Proceedings of NIPS, 129-136, 2006.

[13] Belkin, M., Niyogi, P., and Sindhwani, V. *Manifold regularization: A geometric framework for learning from labeled and unlabeled examples*, Journal of Machine Learning Research, 7, 23992434, 2006.

[14] Belkin, M., Sun, J., and Wang, Y. *Constructing the Laplace Operator from Point Clouds*, SODA, 1031-1040, 2009.

[15] Bellman, R. E. *Dynamic Programming*, Courier Dover Publications, 2003.

[16] Bernstein, M., de Silva, M., Langford, J., and Tenenbaum, J. *Graph approximations to geodesics on embedded manifolds*, Technical report, Stanford University, 2000.

[17] Camastra, F. *Data dimensionality estimation methods: a survey*, Pattern Recognition, 36, 2945-2954, 2003.

[18] Candes, E., and Tao, T. *Near optimal signal recovery from random projections: Universal encoding strategies?*, IEEE Trans. Inform. Theory, 52(12), 54065425, 2006.

[19] Candes, E., Romberg, J., and Tao, T. *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Trans. Inform. Theory, 52(2), 489509, 2006.

[20] Candes, E., Romberg, J., and Tao, T. *Stable signal recovery from incomplete and inaccurate measurements*, Comm. Pure Appl. Math., 59(8), 12071223, 2006.

[21] Candes, E., and Tao, T. *Decoding via linear programming*, IEEE Trans. Inform. Theory, 51(12), 42034215, 2005.

[22] Chatelin, F. *Spectral Approximation of Linear Operators*, Academic Press, New York, 1983.

[23] Chen, J., Fang, H., and Saad, Y. *Fast Approximate kNN Graph Construction for High Dimensional Data via Recursive Lanczos Bisection*, Journal of Machine Learning Research, 10, 1989-2012, 2009.

[24] Chung, F. R. K. *Spectral Graph Theory*, CBMS Regional Conference Series in Mathematics 92, 1997.

[25] Coifman, R. R., and Lafon, S. *Diffusion maps*, Appl. Comput. Harmon. Anal., 21(1), 5-30, 2006.

[26] Coifman, R. R., Maggioni, M. *Diffusion Wavelets*, Appl. Comp. Harmon. Anal. 21(1), 53-94, 2006.

[27] Coifman, R. R., Lafon, S., Lee, A. B., Maggioni, M., Nadler, B., Warner, F. J., and Zucker, S. W. *Geometric diffusions as a tool for harmonic analysis and structure definition of data. part i: Diffusion maps*, Proc. Nat. Acad. Sci. 102, 74267431, 2005.

[28] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to Algorithms*, The MIT Press, 2nd edition, 2001.

[29] Cox, T., and Cox, M. *Multidimensional scaling*, Chapman and Hall, 1994.

[30] Czaja, W., and Ehler, M. *Schroedinger Eigenmaps for the Analysis and Classification of Multispectral Data in Bio-Medical Imaging* (Submitted).

[31] Czaja, W., and Halevy, A. *Convergence of Schroedinger Eigenmaps* (In Preparation).

[32] Dasgupta, S., and Gupta, A. *An elementary proof of the Johnson-Lindenstrauss lemma*, Technical Report TR-99-006, Berkeley, CA, 1999.

[33] Donoho, D. L., and Grimes, C. *Hessian Eigenmaps: New Locally Linear Embedding Techniques for High Dimensional Data*, Proc. Nat. Acad. Sci. 100, 5591-5596, 2003.

[34] Donoho, D. L. For most large underdetermined systems of linear equations, the minimal L1-norm solution is also the sparsest solution, Comm. Pure Appl. Math., 59(6), 2006.

[35] Donoho, D. L. *Compressed sensing*, IEEE Trans. Inform. Theory, 52(4), 2006.

[36] Duarte, M., Davenport, M., Takhar, D., Laska, J., Sun, T., Kelly, K. and Baraniuk, R. *Single-pixel imaging via compressive sampling*, IEEE Signal Processing Magazine, 25(2), 83-91, 2008.

[37] Flake, J. C. *The Multiplicative Zak Transform, Dimension Reduction, and Wavelet Analysis of LIDAR Data*, Ph.D. Thesis, University of Maryland, College Park, 2010.

[38] Gashler, M., Ventura, D., and Martinez, T. *Iterative Non-linear Dimensionality Reduction with Manifold Sculpting*, Advances in NIPS 20, 513-520, 2008.

[39] Gine, E., and Koltchinskii, V. *Empirical graph Laplacian approximation of Laplace-Beltrami operators: large sample results*, IMS Lecture Notes Monograph Series, 51, 238-259, 2006.

[40] Goldberg, Y., Kushnir, D., Zakai, A., and Ritov, Y. *Manifold Learning: The Price of Normalization*, Journal of Machine Learning Research, 9, 1909-1939, 2008

[41] Gordon, C., Webb, D. L., Wolpert, S. *One cannot hear the shape of a drum*, Bulletin of the American Mathematical Society, 27, 134-138, 1992.

[42] Halevy, A. *Laplacian Eigenmaps with Random Projections* (In Preparation).

[43] Halevy, A. *Accelerating Laplacian Eigenmaps Using Fast Approximate Neighborhood Constructions* (In Preparation).

[44] He, X., and Niyogi, P. *Locality Preserving Projections*, Proc. Conf. Advances in NIPS, 2003.

[45] Hirn, M., *Enumeration of Harmonic Frames and Frame Based Dimension Reduction*, Ph.D. Thesis, University of Maryland, College Park, 2009.

[46] Jain, A. K., Murty, M. N., and Flynn, P. J. *Data Clustering: a Review*, ACM Comput. Surv., 31, 264-323, 1999.

[47] Kac, M. *Can one hear the shape of a drum?*, American Mathematical Monthly, 73(4, part 2), 1-23, 1966.

[48] Kegl, B. *Intrinsic dimension estimation using packing numbers*, Advances in NIPS, 14, MIT Press, 2002.

[49] Lee, J.A., and Verleysen, M. *Nonlinear Dimensionality Reduction*, Springer, New York, NY, 2007.

[50] Levina, E., and Bickel, P. J. *Maximum Likelihood Estimation of Intrinsic Dimension*, NIPS, 2005.

[51] Lustig, M., Donoho, D. L., and Pauly, J. M. *Sparse MRI: The application of compressed sensing for rapid MR imaging*, Magnetic Resonance in Medicine, 58(6), 1182-1195, 2007.

[52] Luxburg, U., Belkin, M., Bousquet, O. *Consistency of Spectral Clustering*, Max Planck Institute for Biological Cybernetics Technical Report TR, 134, 2004.

[53] Maaten, L. J. P., Postma, E. O., and Herik, H. J. *Dimensionality Reduction: A Comparative Review*, Tilburg University Technical Report, TiCC-TR, 2009.

[54] Maggioni, M., and Coifman, R. *Multiscale analysis of data sets with diffusion wavelets*, 7th SIAM International Conference on Data Mining, Minneapolis, MN, 2007.

[55] Paredes, R., Chavez, E., Figueroa, K., and Navarro, G. *Practical construction of k-nearest neighbor graphs in metric spaces*, Proc. 5th Workshop on Efficient and Experimental Algorithms, 2006.

[56] Rosenberg, S. *The Laplacian on a Riemannian Manifold*, Cambridge Univeristy Press, Cambridge, 1997.

[57] Roweis, S. T., and Saul, L. K. *Nonlinear Dimensionality Reduction by Locally Linear Embedding*, Science, 290(5500), 2323-2326, 2000.

[58] Takhar, D., Laska, J., Wakin, M., Duarte, M., Baron, D., Sarvotham, S., Kelly, K., and Baraniuk, R. *A new compressive imaging camera architecture using optical-domain compression*, Computational Imaging IV at SPIE Electronic Imaging, San Jose, California, 2006.

[59] Tenenbaum, J.B., de Silva, V., and Langford, J. C. *A global geometric framework for nonlinear dimensionality reduction*, Science, 290(5500), 23192323, 2000.

[60] Urban Hyperspectral Dataset, HYDICE sensor imagery. Available: http://www.agc.army.mil/Hypercube.

[61] Vapnik, V. N. *Statistical learning theory*, Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications and Control, Wiley-Interscience, 1998.

[62] Widemann, D. P. *Dimensionality Reduction for Hyperspectral Data*, Ph.D. Thesis, University of Maryland, College Park, 2008.

[63] Zhang, K., Tsang, I., and Kwok, J. *Improved Nystrom Low Rank Approximation and Error Analysis*, Proceedings of the 25th International Conference on Machine Learning, 1232-1239, 2008.