TRACE OPTIMIZATION AND EIGENPROBLEMS IN DIMENSION REDUCTION METHODS

E. KOKIOPOULOU*, J. CHEN†, AND Y. SAAD†

Abstract. This paper gives an overview of the eigenvalue problems encountered in areas of data mining that are related to dimension reduction. Given some input high-dimensional data, the goal of dimension reduction is to map them to a low-dimensional space such that certain properties of the initial data are preserved. Optimizing the above properties among the reduced data can be typically posed as a trace optimization problem that leads to an eigenvalue problem. There is a rich variety of such problems and the goal of this paper is to unravel relations between them as well as to discuss effective solution techniques. First, we make a distinction between projective methods that determine an explicit linear projection from the high-dimensional space to the low-dimensional space, and nonlinear methods where the mapping between the two is nonlinear and implicit. Then, we show that all of the eigenvalue problems solved in the context of explicit projections can be viewed as the projected analogues of the so-called nonlinear or implicit projections. We also discuss kernels as a means of unifying both types of methods and revisit some of the equivalences between methods established in this way. Finally, we provide some illustrative examples to showcase the behavior and the particular characteristics of the various dimension reduction methods on real world data sets.

Key words. Linear Dimension Reduction, Nonlinear Dimension Reduction, Principal Component Analysis, Projection methods, Locally Linear Embedding (LLE), Kernel methods, Locality Preserving Projections (LPP), Laplacean Eigenmaps.

1. Introduction. The term 'data mining' refers to a broad discipline which includes such diverse areas as machine learning, data analysis, information retrieval, pattern recognition, and web-searching, to list just a few. The widespread use of linear algebra techniques in many sub-areas of data mining is remarkable. A prototypical area of data mining where numerical linear algebra techniques play a crucial role is that of *dimension reduction* which is the focus of this study. Dimension reduction is ubiquitous in applications ranging from pattern recognition and learning [50] to the unrelated fields of graph drawing [26, 32], materials research [13, 11], and magnetism [42]. Given a set of high-dimensional data, the goal of dimension reduction is to map the data to a low-dimensional space. Specifically, we are given a data matrix

$$X = [x_1, \ldots, x_n] \in \mathbb{R}^{m \times n}, \tag{1.1}$$

for which we wish to find a low-dimensional analogue

$$Y = [y_1, \ldots, y_n] \in \mathbb{R}^{d \times n}, \tag{1.2}$$

with $d \ll m$, which is a faithful representation of X in some sense. As will be seen, many of the dimension reduction techniques lead to optimization problems which typically involve a trace. This in turn leads to eigenvalue problems [38].

It may be helpful to define some terms used in data mining. Among the many problems which arise in data mining, two are of primary importance. One is 'unsupervised clustering', which is the task of finding subsets of the data such that items from the same subset are most similar and items from distinct subsets are most dissimilar. The second is classification (supervised learning), whereby we are given a set of distinct sets that are labeled (e.g. samples of handwritten digits labeled from 0 to 9) and when a new sample is presented to us we must determine to which of the sets is it most likely to belong. For the example of

*Seminar for Applied Mathematics, ETH, HG G J49, Rämistrasse 101, 8092 Zürich, Switzerland. Email: effrosyni.kokiopoulou@sam.math.ethz.ch

†Department of Computer Science and Engineering; University of Minnesota; Minneapolis, MN 55455. Email: (jchen, saad)@cs.umn.edu. Work supported by NSF under grant DMS-0810938 and by the Minnesota Supercomputer Institute.

handwritten digits this is the problem of recognizing a digit given many labeled samples of already deciphered digits available in a given data set. In order to perform these tasks it is common to first process the given datasets (e.g. the database of handwritten digits) in order to reduce its dimension, i.e., to find a dataset of much lower dimension than the original one but which preserves its main features. What is often misunderstood is that this dimension reduction is not done for the sole purpose of reducing cost but mainly for reducing the effect of noise and extracting the main features of the data. For this reason the term "feature extraction" is sometimes used in the literature instead of dimension reduction.

There have been two types of methods proposed for dimension reduction. The first class of methods can be termed "projective". This includes all linear methods whereby the data matrix is explicitly transformed into a low-dimensional version. Then these projective methods find an explicit linear transformation to perform the reduction, i.e., they find a $m \times d$ matrix V and express the reduced dimension data as $Y = V^T X$. This class of methods includes the standard Principal Component Analysis (PCA), the Locality Preserving Projection (LPP) [22], the Orthogonal Neighborhood Preserving Projections, (ONPP) [24] and other variants of these. A second class of methods that do not rely on explicit projections and are inherently nonlinear [27], find directly the low dimensional data matrix Y, by simply imposing that certain locality or affinity between near-by points be preserved. Furthermore, both types of dimension reduction methods can be extended to their supervised versions, where each data point is associated with a class label and the class labels are taken into account when performing the reduction step.

The goal of this paper is to try to unravel some of the relationships between these dimension reduction methods, their supervised counterparts and the optimization problems that they rely upon. The paper will not describe the details of the various applications. Instead these will be summarized and expressed in simple mathematical terms with the goal of showing the objective function that is optimized in each case. In addition, two main observations will be made in this paper. The first is about a distinction between the projective methods and the nonlinear ones. Specifically, the eigenvalue problem solved in the linear case consists of applying a projection technique, i.e., a Rayleigh-Ritz projection method, as it leads to the solution of an eigenvalue problem in the space spanned by the columns of the data matrix X^T. The second is that these two families of methods can be brought together thanks to the use of kernels. These observations will strengthen a few similar observations made in a few earlier papers, e.g., [24, 21, 55]. The observation that kernels can help unify dimension reduction has been made before. Ham et al [21] note that several of the known standard methods (LLE [34], Isomap [46], Laplacean eigenmaps [4, 5] can be regarded as some form of Kernel PCA. In [24], it was observed that linear and nonlinear projection methods are in fact equivalent, in the sense that one can define one from the other with the help of kernels.

Producing a set Y in the form (1.2) that is an accurate representation of the set X in (1.1), with $d \ll m$, can be achieved in different ways by selecting the *type* of the reduced data Y as well as the desirable *properties to be preserved*. By type we mean whether we require that Y be simply a low-rank representation of X, or a data set in a vector space with fewer dimensions. Examples of properties to be preserved may include the global geometry, neighborhood information such as local neighborhoods [5, 34] and local tangent space [60], distances between data points [46, 54], or angles formed by adjacent line segments [43]. The

mapping from X to Y may be implicit (i.e., not known via an explicit function) or explicit. Nonlinear (i.e., implicit) methods make no assumptions about the mapping and they only compute for each x_i its corresponding y_i in the reduced space. On the other hand, linear methods compute an explicit linear mapping between the two.

The rest of this paper is organized as follows. Section 2 summarizes a few well-known results of linear algebra that will be exploited repeatedly in the paper. Then, Sections 3 and 4 provide a brief overview of nonlinear and linear methods respectively for dimension reduction. Section 5 discusses dimension reduction in supervised settings, where the class labels of the data are taken into account. Section 6 provides an analysis of relations between the different methods as well as connections to methods from different areas, such as spectral clustering and projection techniques for eigenvalue problems. Kernelized versions of different linear dimension reduction methods are discussed in Section 7, along with various relationships with their nonlinear counterparts. Finally, Section 8 provides illustrative examples for data visualization and classification of handwritten digits and faces, and the paper ends with a conclusion in Section 10.

2. Preliminaries. First, given a symmetric matrix A, of dimension $n \times n$ and an arbitrary unitary matrix V of dimension $n \times d$ then the trace of $V^T AV$ is maximized when V is an orthogonal basis of the eigenspace associated with the (algebraically) largest eigenvalues. In particular, it is achieved for the eigenbasis itself: if eigenvalues are labeled decreasingly and $u_1, \cdots, u_d$ are eigenvectors associated with the first d eigenvalues $\lambda_1, \cdots, \lambda_d$, and $U = [u_1, \cdots, u_d]$, with $U^T U = I$, then,

$$\begin{cases} \max\limits_{\substack{V \in \mathbb{R}^{n \times d} \\ V^T V = I}} \ \mathrm{Tr}\left[V^T AV\right] = \ \mathrm{Tr}\left[U^T AU\right] = \lambda_1 + \cdots + \lambda_d. \end{cases} \tag{2.1}$$

While this result is seldom explicitly stated on its own in standard textbooks, it is an immediate consequence of the Courant-Fisher characterization, see, e.g., [33, 35]. It is important to note that the optimal V is far from being unique. In fact, any V which is an orthonormal basis of the eigenspace associated with the first d eigenvalues will be optimal. In other words, what matters is the subspace rather than a particular orthonormal basis for it.

The main point is that to maximize the trace in (2.1), one needs to solve a standard eigenvalue problem. In many instances, we need to maximize $\mathrm{Tr}\,[V^T AV]$ subject to a new normalization constraint for V, one that requires that V be B-orthogonal, i.e., $V^T BV = I$. Assuming that A is symmetric and B positive definite, we know that there are n real eigenvalues for the generalized problem $Au = \lambda Bu$, with B-orthogonal eigenvectors. If these eigenvalues are labeled decreasingly, and if $U = [u_1, \cdots, u_d]$ is the set of eigenvectors associated with the first d eigenvalues, with $U^T BU = I$, then we have

$$\begin{cases} \max\limits_{\substack{V \in \mathbb{R}^{n \times d} \\ V^T BV = I}} \ \mathrm{Tr}\left[V^T AV\right] = \ \mathrm{Tr}\left[U^T AU\right] = \lambda_1 + \cdots + \lambda_d. \end{cases} \tag{2.2}$$

In reality, Problem (2.2) often arises as a simplification of an objective function that is

more difficult to maximize, namely:

$$\begin{cases} \max\limits_{V \in \mathbb{R}^{n\times d}} & \dfrac{\text{Tr}\,\left[V^T A V\right]}{\text{Tr}\,\left[V^T B V\right]}\ . \\ V^T C V = I \end{cases} \tag{2.3}$$

Here B and C are assumed to be symmetric and positive definite for simplicity. The matrix C defines the desired orthogonality and in the simplest case it is just the identity matrix. The original version shown above has resurfaced in recent years, see, e.g., [19, 49, 56, 59] among others. Though we will not give the above problem as much attention as the more standard problem (2.2), it is important to give an idea on the way it is commonly solved. There is no loss of generality in assuming that C is the identity. Since B is assumed to be positive definite[1], it is not difficult to see that there is a maximum μ that is reached for a certain (non-unique) orthogonal matrix, which we will denote by U. Then, $\text{Tr}\,[V^T A V] - \mu\,\text{Tr}\,[V^T B V] \leq 0$ for any orthogonal V. This means that for this μ we have $\text{Tr}\,[V^T(A-\mu B)V] \leq 0$ for any orthogonal V, and also $\text{Tr}\,[U^T(A-\mu B)U] = 0$. Therefore we have the following necessary condition for the pair μ, U to be optimal:

$$\max_{V^T V = I}\ \text{Tr}\,[V^T(A-\mu B)V] = \text{Tr}\,[U^T(A-\mu B)U] = 0. \tag{2.4}$$

According to (2.1), the maximum trace of $V^T(A-\mu B)V$ is simply the sum of the largest d eigenvalues of $A-\mu B$ and U is the set of corresponding eigenvectors. If μ maximizes the trace ratio (2.3) (with $C = I$), then the sum of the largest d eigenvalues of the pencil $A-\mu B$ equals zero, and the corresponding eigenvectors form the desired optimal solution of (2.3).

When B is positive definite, it can be seen that the function

$$f(\theta) = \max_{V^T V = I}\ \text{Tr}\,[V^T(A-\theta B)V]$$

is a decreasing function of θ. For $\theta = 0$ we have $f(\theta) > 0$. For $\theta > \lambda_{max}(A,B)$ we have $f(\theta) < 0$, where $\lambda_{max}(A,B)$ is the largest generalized eigenvalue of the pencil (A,B). Finding the optimal solution will involve a search for the (unique) root of $f(\theta)$. In [49] and [19] algorithms were proposed to solve (2.3) by computing this root and by exploiting the above relations. No matter what method is used it appears clear that it will be far more expensive to solve (2.3) than (2.2), because the search for the root μ will typically involve solving several eigenvalue problems instead of just one.

3. Nonlinear dimension reduction. We start with an overview of nonlinear methods. In what follows, we discuss LLE and Laplacean Eigenmaps, which are the most representative nonlinear methods for dimensionality reduction. These methods begin with the construction of a weighted graph which captures some information on the local neighborhood structure of the data. In the sequel, we refer to this graph as the "affinity graph". Specifically, the affinity (or adjacency) graph is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose nodes $\mathcal{V}$ are the data samples. The edges of this graph can be defined for example by taking a certain nearness measure and including all points within a radius ϵ of a given node, to its adjacency list. Alternatively,

[1] We can relax the assumptions: B can be positive semidefinite, but for the problem to be well-posed its null space must be of dimension less than d. Also if A is positive semidefinite, we must assume that $Null(A) \cap Null(B) = \emptyset$.

one can include those k nodes that are the nearest neighbors to x_i. In the latter case it is called the k-NN graph. It is typical to assign weights w_{ij} on the edges $e_{ij} \in \mathcal{E}$ of the affinity graph. The affinity graph along with these weights then defines a matrix W whose entries are the weights w_{ij}'s that are nonzero only for adjacent nodes in the graph.

3.1. LLE. In *Locally Linear Embedding* (LLE), the construction of the affinity graph is based on the assumption that the points lie on some high-dimensional manifold, so each point is approximately expressed as a linear combination of a few neighbors, see [34, 37]. Thus, the affinity matrix is built by computing optimal weights which will relate a given point to its neighbors in some locally optimal way. The reconstruction error for sample i can be measured by

$$\left\| x_i - \sum_j w_{ij} x_j \right\|_2^2 . \tag{3.1}$$

The weights w_{ij} represent the linear coefficients for (approximately) reconstructing the sample x_i from its neighbors $\{x_j\}$, with $w_{ij} = 0$, if x_j is not one of the k nearest neighbors of x_i. We can set $w_{ii} \equiv 0$, for all i. The coefficients are scaled so that their sum is unity, i.e.,

$$\sum_j w_{ij} = 1. \tag{3.2}$$

Determining the w_{ij}'s for a given point x_i is a *local* calculation, in the sense that it only involves x_i and its nearest neighbors. As a result computing the weights will be fairly inexpensive; an explicit solution can be extracted by solving a small linear system which involves a 'local' Grammian matrix, for details see [34, 37]. After this phase is completed we have available a matrix W which is such that each column x_i of the data set is well represented by the linear combination $\sum_j w_{ij} x_j$. In other words, $X \approx XW^T$, i.e., X^T is a set of approximate left null vectors of $I - W$.

The procedure then seeks d-dimensional vectors y_i, $i = 1, \ldots, n$ so that the same relation is satisfied between the matrix W and the y_i's. This is achieved by minimizing the objective function

$$\mathcal{F}_{LLE}(Y) = \sum_i \|y_i - \sum_j w_{ij} y_j\|_2^2. \tag{3.3}$$

LLE imposes two constraints to this optimization problem: i) the mapped coordinates must be centered at the origin and ii) the embedded vectors must have unit covariance:

$$\sum_i y_i = 0; \quad \text{and} \quad \frac{1}{n} \sum_i y_i y_i^T = I \ . \tag{3.4}$$

The objective function (3.3) is minimized with these constraints on Y.

We can rewrite (3.3) as a trace by noting that $\mathcal{F}_{LLE}(Y) = \|Y - YW^T\|_F^2$, and this leads to:

$$\mathcal{F}_{LLE}(Y) = \text{Tr} \left[Y(I - W^T)(I - W)Y^T \right] \ . \tag{3.5}$$

Therefore the new optimization problem to solve is[2]

$$\begin{cases} \min \\ Y \in \mathbb{R}^{d\times n} \\ YY^T = I \end{cases} \quad \text{Tr}\left[Y(I-W^T)(I-W)Y^T\right]. \tag{3.6}$$

The solution of the problem is obtained from the set of eigenvectors associated with the d smallest eigenvalues of $M \equiv (I-W^T)(I-W)$:

$$(I-W^T)(I-W)u_i = \lambda_i u_i; \quad Y = [u_2, \cdots, u_{d+1}]^T. \tag{3.7}$$

Note that the eigenvector associated with the eigenvalue zero is discarded and that the matrix Y is simply the set of bottom eigenvectors of $(I-W^T)(I-W)$ associated with the 2nd to $(d+1)$-th eigenvalues. We will often refer to the matrix $M = (I-W^T)(I-W)$ as the LLE matrix.

3.2. Laplacean Eigenmaps. The *Laplacean Eigenmaps* technique is rather similar to LLE. It uses different weights to represent locality and a slightly different objective function. Two common choices are weights of the heat kernel $w_{ij} = \exp(-\|x_i - x_j\|_2^2/t)$ or constant weights ($w_{ij} = 1$ if i and j are adjacent, $w_{ij} = 0$ otherwise). It is important to note that the choice of the parameter t is crucial to the performance of this method. The name heat 'kernel' is self explaining, since the matrix $[w_{ij}]$ happens to be positive semi-definite.

Once this graph is available, a Laplacean matrix of the graph is constructed, by setting a diagonal matrix D with diagonal entries $d_{ii} = \sum_j w_{ij}$. The matrix

$$L \equiv D - W$$

is the *Laplacean* of the weighted graph defined above. Note that the row-sums of the matrix L are zero by the definition of D, so $L\mathbf{1} = 0$, and therefore L is singular. The problem in Laplacean Eigenmaps is then to minimize

$$\mathcal{F}_{EM}(Y) = \sum_{i,j=1}^{n} w_{ij}\|y_i - y_j\|_2^2 \tag{3.8}$$

subject to an orthogonality constraint that uses the matrix D for scaling:

$$YDY^T = I\ .$$

The rationale for this approach is to *put a penalty for mapping nearest neighbor nodes in the original graph to distant points in the low-dimensional data.*

Compare (3.8) and (3.3). The difference between the two is subtle and one might ask if (3.8) can also be converted into a trace optimization problem similar to (3.6). As it turns out $\mathcal{F}_{EM}$ can be written as a trace that will put the method quite close to LLE in spirit:

$$\mathcal{F}_{EM}(Y) = 2\text{Tr}\,[Y(D-W)Y^T]. \tag{3.9}$$

Therefore the new optimization problem to solve is

$$\begin{cases} \min \\ Y \in \mathbb{R}^{d\times n} \\ YD\ Y^T = I \end{cases} \quad \text{Tr}\left[Y(D-W)Y^T\right]\ . \tag{3.10}$$

[2]The final y_i's are obtained by translating and scaling each column of Y.

The solution Y to this optimization problem can be obtained from the eigenvectors associated with the d smallest eigenvalues of the generalized eigenvalue problem

$$(D - W)u_i = \lambda_i D u_i \ ; \qquad Y = [u_2, \cdots, u_{d+1}]^T \ . \tag{3.11}$$

One can also solve a standard eigenvalue problem by making a small change of variables and this is useful to better see links with other methods. Indeed, it would be useful to standardize the constraint YDY^T so that the diagonal scaling does not appear. For this we set $\hat{Y} = YD^{1/2}$ and $\hat{W} = D^{-1/2}WD^{-1/2}$, and this simplifies (3.10) into:

$$\begin{cases} \min \\ \hat{Y} \in \mathbb{R}^{d \times n} \\ \hat{Y}\ \hat{Y}^T = I \end{cases} \quad \text{Tr} \left[\hat{Y}(I - \hat{W})\hat{Y}^T \right]. \tag{3.12}$$

In this case, (3.11) yields:

$$(I - \hat{W})\hat{u}_i = \lambda_i \hat{u}_i \ ; \qquad Y = [\hat{u}_2, \cdots, \hat{u}_{d+1}]^T D^{1/2} \ . \tag{3.13}$$

The quantity $\hat{L} = I - \hat{W} = D^{-1/2}LD^{-1/2}$ is called the *normalized Laplacean*.

4. Linear dimension reduction. The methods in the previous section do not provide an explicit function that maps a vector x into its low-dimensional representation y in d-dimensional space. This mapping is only known for each of the vectors x_i of the data set X. For each x_i we know how to associate a low-dimensional item y_i. In some applications it is important to be able to find the mapping y for an arbitrary, 'out-of-sample' vector x. The methods discussed in this section have been developed in part to address this issue. They are based on using an explicit (linear) mapping defined by a matrix $V \in \mathbb{R}^{m \times d}$. These projective techniques replace the original data X by a matrix of the form

$$Y = V^T X, \quad \text{where} \quad V \in \mathbb{R}^{m \times d}. \tag{4.1}$$

Once the matrix V has been learned, each vector x_i can be projected to the reduced space by simply computing $y_i = V^T x_i$. If V is a unitary matrix, then Y represents the orthogonal projection of X into the V-space.

4.1. PCA. The best known technique in this category is *Principal Component Analysis* (PCA). PCA computes an orthonormal matrix V such that the variance of the projected vectors is maximized, i.e, V is the maximizer of

$$\begin{array}{l} \max \\ V \in \mathbb{R}^{m \times d} \\ V^T V = I \end{array} \ \sum_{i=1}^{n} \left\| y_i - \frac{1}{n} \sum_{j=1}^{n} y_j \right\|_2^2, \quad y_i = V^T x_i. \tag{4.2}$$

Recalling that $\mathbf{1}$ denotes the vector of all ones, the objective function in (4.2) becomes

$$\mathcal{F}_{\text{PCA}}(Y) = \sum_{i=1}^{n} \left\| y_i - \frac{1}{n} \sum_{j=1}^{n} y_j \right\|_2^2 = \text{Tr} \left[V^T X (I - \frac{1}{n} \mathbf{1}\mathbf{1}^T) X^T V \right] \ .$$

In the end, the above optimization can be restated as

$$\begin{cases} \max \\ V \in \mathbb{R}^{m\times d} \\ V^T V = I \end{cases} \text{Tr}\left[V^T X(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)X^T V\right] . \tag{4.3}$$

In the sequel we will denote by $\bar{X}$ the matrix $X(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)$ which is simply the matrix with centered data, i.e., each column is $\bar{x}_i = x_i - \mu$ where μ is the mean of X, $\mu = \sum x_i/n$. Since the matrix in (4.3) can be written $V^T\bar{X}\bar{X}^T V$, so (4.3) becomes

$$\begin{cases} \max \\ V \in \mathbb{R}^{m\times d} \\ V^T V = I \end{cases} \text{Tr}\left[V^T\bar{X}\bar{X}^T V\right] . \tag{4.4}$$

The orthogonal matrix V which maximizes the trace in (4.4) is simply the set of left singular vectors of $\bar{X}$, associated with the largest d singular values,

$$[\bar{X}\bar{X}]^T v_i = \lambda_i v_i. \tag{4.5}$$

The matrix $V = [v_1, \cdots, v_d]$ is used for projecting the data, so $Y = V^T\bar{X}$. If $\bar{X} = U\Sigma Z^T$ is the SVD of $\bar{X}$, the solution to the above optimization problem is $V = U_d$, the matrix of the first d left singular vectors of X, so, denoting by Σ_d the top left $d \times d$ block of Σ, and Z_d the matrix of the first d columns of Z, we obtain

$$Y = U_d^T\bar{X} = \Sigma_d Z_d^T. \tag{4.6}$$

As it turns out, maximizing the variance on the projected space is equivalent to minimizing the projection error

$$\|\bar{X} - VV^T\bar{X}\|_F^2 = \|\bar{X} - VY\|_F^2.$$

This is because a little calculation will show that

$$\|\bar{X} - VY\|_F^2 = \text{Tr}\left[(\bar{X} - VY)^T(\bar{X} - VY)\right] = \text{Tr}\left[\bar{X}^T\bar{X}\right] - \text{Tr}\left[V^T\bar{X}\bar{X}^T V\right].$$

The matrix VV^T is an orthogonal projector onto the span of V. The points $Vy_i \in \mathbb{R}^m$ are sometimes referred to as *reconstructed points*. PCA minimizes the sum of the squares of the distance between any point in the data set and its reconstruction, i.e., its projection.

4.2. MDS and ISOMAP[3]. In metric *Multi-Dimensional Scaling* (metric MDS) the problem posed is to project data in such a way that distances $\|y_i - y_j\|_2$ between projected points are closest to the original distances $\|x_i - x_j\|_2$. Instead of solving the problem in this form, MDS uses a criterion based on inner products.

It is now assumed that the data is centered at zero so we replace X by $\bar{X}$. An important result used is that one can recover distances from inner products and vice-versa. The matrix of inner products, i.e., the Grammian of $\bar{X}$, defined by

$$G = [\langle \bar{x}_i, \bar{x}_j\rangle]_{i,j=1,\cdots,n} \tag{4.7}$$

[3]These methods are essentially not linear methods; however, they are very closely related to PCA and better be presented in this section.

determines completely the distances, since $\|\bar{x}_i - \bar{x}_j\|^2 = g_{ii} + g_{jj} - 2g_{ij}$. The reverse can also be done, i.e., one can determine the inner products from distances by 'inverting' the above relations. Indeed, under the assumption that the data is centered at zero, it can be shown that [47]

$$g_{ij} = \frac{1}{2}\left[\frac{1}{n}\sum_k (s_{ik} + s_{jk}) - s_{ij} - \frac{1}{n^2}\sum_{k,l} s_{kl}\right] ,$$

where $s_{ij} = \|\bar{x}_i - \bar{x}_j\|^2$. In the matrix form, it is

$$G = -\frac{1}{2}[I - \frac{1}{n}\mathbf{1}\mathbf{1}^T]S[I - \frac{1}{n}\mathbf{1}\mathbf{1}^T] ; \qquad S = [s_{ij}]_{i,j=1,\ldots,n}.$$

As a result of the above equality, in order to find a d-dimensional projection which preserves inter-distances as best possible, we need to find a $d \times n$ matrix Y whose Grammian Y^TY is close to G, the Grammian of X, i.e., we need to find the solution of

$$\min_{Y \in \mathbb{R}^{d\times n}} \|G - Y^TY\|_F^2 . \tag{4.8}$$

Let $G = Z\Lambda Z^T$ be the eigenvalue decomposition of G, where it is assumed that the eigenvalues are labeled from largest to smallest. Then the solution to (4.8) is $Y = \Lambda_d^{1/2} Z_d^T$ where Z_d consists of the first d columns of Z, Λ_d is the $d \times d$ upper left block of Λ. Note that with respect to the SVD of $\bar{X}$ this is equal to $\Sigma_d Z_d^T$, which is identical with the result obtained with PCA, see equation (4.6). *So metric MDS gives the same exact result as PCA*. However it arrives at this result using a different path. PCA uses the covariance matrix, while MDS uses the Gram matrix. From a computational cost point of view, there is no real difference if the calculation is based on the SVD of $\bar{X}$. We should note that the solution to (4.8) is unique only up to unitary transformations. This is because a transformation such as $\hat{Y} = QY$ of Y, where Q is unitary, will not change distances between y-points.

Finally, we mention in passing that the technique of ISOMAP [46] essentially performs the same steps as MDS, except that the Grammian $G = \bar{X}^T\bar{X}$ is replaced by a pseudo-Grammian $\hat{G}$ obtained from *geodesic distances* between the points x_i:

$$\hat{G} = -\frac{1}{2}[I - \frac{1}{n}\mathbf{1}\mathbf{1}^T]\hat{S}[I - \frac{1}{n}\mathbf{1}\mathbf{1}^T] ; \qquad \hat{S} = [\hat{s}_{ij}]_{i,j=1,\ldots,n},$$

where $\hat{s}_{ij}$ is the squared shortest graph distance between x_i and x_j.

4.3. LPP. The *Locality Preserving Projections* (LPP) [22] is a graph-based projective technique. It projects the data so as to preserve a certain *affinity graph* constructed from the data. LPP defines the projected points in the form $y_i = V^Tx_i$ by *putting a penalty for mapping nearest neighbor nodes in the original graph to distant points in the projected data*. Therefore, the objective function to be minimized is identical with that of Laplacean Eigenmaps,

$$\mathcal{F}_{LPP}(Y) = \sum_{i,j=1}^{n} w_{ij}\|y_i - y_j\|_2^2 .$$

The matrix V, which is the actual unknown, is implicitly represented in the above function, through the dependence of the y_i's on V. Writing $Y = V^T X$, we reach the optimization problem,

$$\begin{cases} \min\limits_{V \in \mathbb{R}^{m\times d}} & \text{Tr } \left[V^T X(D-W)X^T V\right] \\ V^T \left(XDX^T\right) V = I \end{cases} \tag{4.9}$$

whose solution can be computed from the generalized eigenvalue problem

$$X(D-W)X^T v_i = \lambda_i XDX^T v_i. \tag{4.10}$$

Similarly to Eigenmaps, the smallest d eigenvalues and eigenvectors must be computed.

It is simpler to deal with the 'normalized' case of LPP, by scaling the set Y as before in the case of Laplacean Eigenmaps (see eq. (3.12)). We define $\hat{Y} = YD^{1/2} = V^T XD^{1/2}$. So, if $\hat{X} = XD^{1/2}$, we have $\hat{Y} = V^T \hat{X}$, and the above problem then becomes

$$\begin{cases} \min\limits_{V \in \mathbb{R}^{m\times d}} & \text{Tr } \left[V^T \hat{X}(I-\hat{W})\hat{X}^T V\right] \\ V^T \left(\hat{X}\hat{X}^T\right) V = I \end{cases} \tag{4.11}$$

where $\hat{W}$ is the same matrix as in (3.12). The eigenvalue problem to solve is now

$$\hat{X}(I-\hat{W})\hat{X}^T v_i = \lambda_i \hat{X}\hat{X}^T v_i. \tag{4.12}$$

The projected data y_i is defined by $y_i = V^T x_i$ for each i, where $V = [v_1, \cdots, v_d]$.

4.4. ONPP. *Orthogonal Neighborhood Preserving Projection* (ONPP) [24, 25] seeks an orthogonal mapping of a given data set so as to best preserve the same affinity graph as LLE. In other words, ONPP is an orthogonal projection version of LLE. The projection matrix V in ONPP is determined by minimizing the same objective function as in (3.5), with the additional constraint that Y is of the form $Y = V^T X$ and the columns of V be orthonormal, i.e. $V^T V = I$. The optimization problem becomes

$$\begin{cases} \min\limits_{V \in \mathbb{R}^{m\times d}} & \text{Tr } \left[V^T X(I-W^T)(I-W)X^T V\right] \\ V^T V = I \end{cases} . \tag{4.13}$$

Its solution is the basis of the eigenvectors associated with the d smallest eigenvalues of the matrix $\tilde{M} \equiv X(I-W^T)(I-W)X^T = XMX^T$.

$$X(I-W^T)(I-W)X^T u_i = \lambda u_i. \tag{4.14}$$

Then the projector V is $[u_1, u_2, \cdots, u_d]$ and results in the projected data $Y = V^T X$.

The assumptions that were made when defining the weights w_{ij} in Section 3.1 imply that the $n \times n$ matrix $I - W$ is singular due to eq. (3.2). In the case when $m > n$ the matrix $\tilde{M}$, which is of size $m \times m$, is at most of rank n and it is therefore singular. In the case when $m \leq n$, $\tilde{M}$ is not necessarily singular but it is observed in practice that ignoring the smallest eigenvalue is helpful [25].

4.5. Other variations on the locality preserving theme. A few possible variations of the methods discussed above can be developed. As was seen ONPP is one such variation which adapts the LLE affinity graph and seeks a projected data which preserves this graph just as in LLE. Another very simple option is to solve the same optimization problem as ONPP but require the same orthogonality of the projected data as LLE, namely: $YY^T = I$. This yields the constraint $V^T XX^T V = I$ instead of the $V^T V = I$ required in ONPP. In [24] we called this *Neighborhood Preserving Projections* (NPP). The resulting new optimization problem is the following modification of (4.13)

$$\begin{cases} \min\limits_{V \in \mathbb{R}^{m\times d}} & \text{Tr } \left[V^T X(I - W^T)(I - W)X^T V\right] . \\ V^T XX^T V = I \end{cases} \tag{4.15}$$

and the new solution is

$$X(I - W^T)(I - W)X^T u_i = \lambda(XX^T)u_i. \tag{4.16}$$

As before, $V = [u_1, \cdots, u_d]$ and $y_i = V^T x_i, i = 1, \cdots, n$.

Another variation goes in the other direction by using the objective function of LPP (using graph Laplaceans) and requiring the data to be orthogonally projected:

$$\begin{cases} \min\limits_{V \in \mathbb{R}^{m\times d}} & \text{Tr } \left[V^T X(D - W)X^T V\right] . \\ V^T\, V = I \end{cases} \tag{4.17}$$

This was referred to as *Orthogonal Locality Preserving Projections* (OLPP), in [24]. Note in passing that a different technique was developed in [10] and named Orthogonal Laplacean faces, which is also sometimes referred to as OLPP. We will not refer to this method in this paper and there is therefore no confusion.

5. Supervised dimension reduction. The problem of classification can be described as follows. We are given a data set consisting of c known subsets (classes or clusters) which are labeled from 1 to c. When a new item is presented to us, we need to determine to which of the classes (clusters) it is the most related in some sense. When the class labels of the data set are taken into account during dimension reduction, the process is called supervised (and unsupervised in the opposite case). It has been observed in general that supervised methods perform better in many classification tasks relative to the unsupervised ones. In what follows, we first describe supervised versions of the above graph-based methods and then we discuss Linear Discriminant Analysis (LDA), which is one of the most popular supervised techniques for linear dimension reduction.

5.1. Supervised graph-based methods. As discussed so far, the above methods do not make use of class labels. It is possible to develop supervised versions of the above methods by taking the class labels into account. Assume that we have c classes and that the data are organized, without loss of generality, as $X_1, \cdots, X_c$ with $X_i \in \mathbb{R}^{m\times n_i}$, where n_i denotes the number of samples that belong to the ith class. In other words, assume that the data samples are ordered according to their class membership.

In supervised methods the class labels are used to build the graph. The main idea is to build the graph in a discriminant way in order to reflect the categorization of the data into

different classes. One simple approach is to impose that an edge $e_{ij} = (x_i, x_j)$ exists if and only if x_i and x_j belong to the same class. In other words, we make adjacent those nodes that belong to the same class. For instance, preserving localities in such a supervised graph, will result in samples from the same class being projected close-by in the reduced space.

Consider now the structure of the induced adjacency matrix H. Observe that the data graph $\mathcal{G}$ consists of c cliques, since the adjacency relationship between two nodes reflects their class membership. Let $\mathbf{1}_{n_j}$ denote the vector of all ones, with length n_j, and $H_j = \frac{1}{n_j}\mathbf{1}_{n_j}\mathbf{1}_{n_j}^T \in \mathbb{R}^{n_j \times n_j}$ be the block corresponding to the jth class. The $n \times n$ adjacency matrix H will be of the following form

$$H = \text{diag}[H_1, H_2, \cdots, H_c]. \tag{5.1}$$

Thus, the (1,1) diagonal block is of size $n_1 \times n_1$ and has the constant entries $1/n_1$, the (2,2) diagonal block is of size $n_2 \times n_2$ and has the constant entries $1/n_2$, and so on. Using the above supervised graph in the graph-based dimension reduction methods yields their supervised versions.

5.2. LDA. The principle used in *Linear Discriminant Analysis* (LDA) is to project the original data linearly in such a way that the low-dimensional data is best separated. Fisher's Linear Discriminant Analysis, see, e.g., Webb [50], seeks to project the data in low-dimensional space so as to maximize the ratio of the "between scatter" measure over "within scatter" measure of the classes, which are defined next. Let μ be the mean of all the data set, and $\mu^{(k)}$ be the mean of the k-th class, which is of size n_k, and define the two matrices

$$S_B = \sum_{k=1}^{c} n_k(\mu^{(k)} - \mu)(\mu^{(k)} - \mu)^T, \tag{5.2}$$

$$S_W = \sum_{k=1}^{c} \sum_{x_i \in X_k} (x_i - \mu^{(k)})(x_i - \mu^{(k)})^T. \tag{5.3}$$

If we project the set on a one-dimensional space spanned by a given vector a, then the quantity

$$a^T S_B a = \sum_{i=1}^{c} n_k |a^T(\mu^{(k)} - \mu)|^2$$

represents a weighted sum of (squared) distances of the projection of the centroids of each set from the mean μ. At the same time, the quantity

$$a^T S_W a = \sum_{k=1}^{c} \sum_{x_i \in X_k} |a^T(x_i - \mu^{(k)})|^2$$

is the sum of the variances of each the projected sets.

LDA projects the data so as to maximize the ratio of these two numbers:

$$\max_a \frac{a^T S_B a}{a^T S_W a} . \tag{5.4}$$

This optimal a is known to be the eigenvector associated with the largest eigenvalue of the pair (S_B, S_W). If we call S_T the total covariance matrix

$$S_T = \sum_{x_i \in X} (x_i - \mu)(x_i - \mu)^T \ , \tag{5.5}$$

then,

$$S_T = S_W + S_B. \tag{5.6}$$

Therefore, (5.4) is equivalent to

$$\max_a \frac{a^T S_B a}{a^T S_T a}, \tag{5.7}$$

or

$$\min_a \frac{a^T S_W a}{a^T S_T a}, \tag{5.8}$$

where an optimal a is known to be the eigenvector associated with the largest eigenvalue of the pair (S_B, S_T), or the smallest eigenvalue of the pair (S_W, S_T).

The above one-dimensional projection is generalized to ones on d-dimensional spaces, i.e., modify the objective function such that the vector a is replaced by a matrix V. A traditional way is to minimize the trace of $V^T S_B V$ while requiring the columns of the solution matrix V to be S_W-orthogonal, i.e., imposing the condition $V^T S_W V = I$. The optimum is achieved for the set of eigenvectors of the generalized eigenvalue problem

$$S_B u_i = \lambda_i S_W u_i \ ,$$

associated with the largest d eigenvalues. Incidentally, the above problem can also be formulated as a generalized singular value problem (see e.g., [23]). Another approach [49] casts the problem as minimizing the ratio of the two traces:

$$\begin{cases} \max \limits_{V \in \mathbb{R}^{n \times d}} & \dfrac{\text{Tr } \left[V^T S_B V\right]}{\text{Tr } \left[V^T S_W V\right]} \ . \\ V^T V = I \end{cases}$$

Approaches for solving this problem were briefly discussed in Section 2.

Note that with simple algebraic manipulations, the matrices S_B, S_W and S_T can be expressed in terms of the data matrix $\bar{X}$:

$$\begin{aligned} S_B &= \bar{X} H \bar{X}^T, \\ S_W &= \bar{X}(I - H)\bar{X}^T, \\ S_T &= \bar{X}\bar{X}^T. \end{aligned}$$

The matrix S_B has rank at most c because each of the blocks in H has rank one and therefore the matrix H itself has rank c. Because the matrix $I - H$ is an orthogonal projector, its range is the null-space of H which has dimension $n - c$. Thus, $I - H$ which plays the role of a Laplacean, has rank at most $n - c$. The corresponding eigenvalue problem to solve for (5.8) is

$$\bar{X}(I - H)\bar{X}^T u_i = \lambda_i (\bar{X}\bar{X}^T) u_i \tag{5.9}$$

6. Connections between dimension reduction methods. This section establishes connections between the various methods discussed in previous sections.

6.1. Relation between the LLE matrix and the Laplacan matrix. A comparison between (3.6) and (3.12) shows that the two are quite similar. The only difference is in the matrix inside the bracketed term. In one case it is of the form $Y(I - \hat{W})Y^T$ where $I - \hat{W}$ is the normalized graph Laplacean, and in the other it is of the form $Y(I - W^T)(I - W)Y^T$ where W is an affinity matrix. Can one just interpret the LLE matrix $(I - W^T)(I - W)$ as a Laplacean matrix? A Laplacean matrix L associated with a graph is a *a symmetric matrix whose off-diagonal entries are non-positive, and whose row-sums are zero* (or equivalently, the diagonal entries are the negative sums of the off-diagonal entries). In other words, $l_{ij} \leq 0$ for $i \neq 0$, $l_{ii} = -\sum_j l_{ij}$. The LLE matrix $M = (I-W)^T(I-W)$ satisfies the second property (zero row sum) but not the first (nonpositive off-diagonals) in general.

PROPOSITION 6.1. *The symmetric matrix $M = (I - W)^T(I - W)$ has zero row (and column) sums. In addition, denoting by $w_{:j}$ the j-th column of W,*

$$m_{jj} = 1 + \|w_{:j}\|^2 \; ; \quad m_{ij} = -(w_{ij} + w_{ji}) + \langle w_{:j}, w_{:i} \rangle, \quad i \neq j \; . \tag{6.1}$$

Proof. Since $(I - W)$ has row sums equal to zero, then $(I - W)\mathbf{1} = 0$ and therefore $M\mathbf{1} = (I - W^T)(I - W)\mathbf{1} = 0$, which shows that the row sums of M are zero. Since M is symmetric, its column-sums are also zero. Since $M = I - W - W^T + W^TW$, a generic entry $m_{ij} = e_i^T M e_j$ of M is given by,

$$\begin{aligned} m_{ij} &= e_i^T e_j - e_i^T W e_j - e_i^T W^T e_j + e_i^T W^T W e_j \\ &= \delta_{ij} - (w_{ij} + w_{ji}) + \langle w_{:i}, w_{:j} \rangle \end{aligned}$$

from which the relations (6.1) follow immediately after recalling that $w_{ii} = 0$. □

Expression (6.1) shows that the off-diagonal entries of M can be positive, i.e., it is not true that $m_{ij} \leq 0$ for all $i \neq j$. In the particular situation when $w_{ij} = w_{ji} = 0$ and $i \neq j$, then $m_{ij} = \langle w_{:i}, w_{:j} \rangle$ and (6.1) implies that $m_{ij} \geq 0$. When w_{ij} and w_{ji} are not both equal to zero but they are both small, then by the same argument it is likely that m_{ij} will be non-negative. It can be observed with randomly generated sparse matrices that in general there are few other instances of positive off-diagonal entries, i.e., in most cases, m_{ij} is positive only when $w_{ij} + w_{ji}$ is zero or small. For example, for the matrix

$$W = \begin{pmatrix} 0 & 0.4 & 0.6 & 0 \\ 0.1 & 0 & 0.3 & 0.6 \\ 0.2 & 0.4 & 0 & 0.4 \\ 0 & 0.5 & 0.5 & 0 \end{pmatrix}$$

one finds that all off-diagonal entries of $(I - W^T)(I - W)$ are negative except the entries (1,4) and (by symmetry) (4,1) whose value, the inner product of columns 1 and 4, equals 0.14.

Among other similarities between the LLE matrix and the graph Laplacean, is the fact that both matrices are symmetric positive semidefinite and that they are both related to the local structure of the data since they relate nearby points by a relation.

Since any matrix $M = (I - W^T)(I - W)$ cannot be a graph Laplacean matrix, one can ask the reverse question: Given a normalized Laplacean matrix which we write as $\hat{L} = I - \hat{W}$, is it possible to find a matrix W such that the matrix M equals $\hat{L}$? One easy answer is obtained by restricting W to being symmetric. In this case, $W = I - \sqrt{I - \hat{W}}$, which is dense and not necessarily positive. There is one important situation where the Graph Laplacean is easily written as an LLE matrix and that is when $I - W$ is a projector. One specific situation of interest is when $L = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, which is the projector used by PCA, see (4.3). In this case $(I - W^T)(I - W) = I - \hat{W}$ which means that the two methods will yield the same result. Yet another situation of the same type in which L is a projector, arises in supervised learning, which brings us to the next connection.

6.2. Connection between LDA, supervised NPP and supervised LPP. Notice that in the supervised setting discussed in Section 5.1 the block diagonal adjacency matrix H (see eq. (5.1)) is a projector. To see why this is true define the characteristic vector g_k for class k as the vector of $\mathbb{R}^n$ whose ith entry is one if x_i belongs to class k and zero otherwise. Then, H can be alternatively written as

$$H = \sum_{k=1}^{c} \frac{g_k g_k^T}{n_k},$$

which shows that H is a projector. Now take $W = \hat{W} = H$ and observe that $(I - W^T)(I - W) = I - W = I - \hat{W} = I - H$ in this case. Next, compare (5.9), (4.12) and (4.16) and note that they are identical.

PROPOSITION 6.2. *LDA, supervised LPP and supervised NPP are mathematically equivalent when $W = \hat{W} = H$.*

6.3. Connection between PCA and LPP. Next we will make other important connections between PCA and LPP. One of these connections was observed in [22], see also [24]. Essentially, by defining the Laplacean graph to be a dense graph, specifically by defining $L = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, one can easily see that the matrix XLX^T is a scaled covariance matrix and thus *ignoring the constraint in LPP*, one would get the projection on the lowest modes instead of the highest ones as in PCA.

Another connection is now considered. Compare the two eigen-problems (4.5) and (4.12) and notice that for PCA we seek the largest eigenvalues whereas for LPP we seek the smallest ones. If we are able to select $\hat{W}$ in (4.12) so that $\hat{X}(I - \hat{W})\hat{X}^T = I$ then we would recover the result of PCA (apart from the diagonal scaling with D). We can restrict the choice by assuming $D = I$ and assume that the data is centered, so $X\mathbf{1} = 0$. Then, it is easy to select such a matrix $\hat{W}$ in the common situation where $m < n$ and X is of full rank. It is the matrix $\hat{W} = I - X^T(XX^T)^{-2}X$. With this, the LPP problem (4.12) becomes $v_i = \lambda_i(XX^T)v_i$ and we are computing the smallest λ_i and associated v_i's, which correspond to the largest eigenpairs of the covariance matrix. Note also that $I - \hat{W} = SS^T$ where $S = X^\dagger$ is the pseudo-inverse of X. We will revisit this viewpoint when we discuss kernels in Section 7.

PROPOSITION 6.3. *When X is $m \times n$ with $m < n$ and full rank, LPP with the graph Laplacean replaced by the matrix $I - \hat{W} = X^T(XX^T)^{-2}X$ is mathematically equivalent to PCA.*

6.4. Connection to projection methods for eigenvalue problems. Comparing the eigenvalue problems (4.12) and (4.16) will reveal an interesting connection with projection methods for eigenvalue problems. Readers familiar with projection methods will recognize in these problems, a projection-type technique for eigenvalue problems, using the space spanned by X^T. Recall that a projection method for computing approximate eigenpairs of a matrix eigenvalue problem of the form

$$Au = \lambda u$$

utilizes a certain subspace $\mathcal{K}$ from which the eigenvectors are extracted. Specifically, the conditions are as follows, where the tildes denote the approximation: Find $\tilde{u} \in \mathcal{K}$ and $\tilde{\lambda} \in \mathbb{C}$ such that

$$A\tilde{u} - \tilde{\lambda}\tilde{u} \perp \mathcal{K} \ . \tag{6.2}$$

This is referred to as an orthogonal projection method. Stating that $\tilde{u} \in \mathcal{K}$ gives k degrees of freedom if $dim(\mathcal{K}) = k$, and condition (6.2) imposes k independent constraints. If V is a basis of the subspace $\mathcal{K}$, then the above conditions become $\tilde{u} = Vy$, for a certain $y \in \mathbb{R}^k$, and (6.2) leads to

$$V^T(A - \tilde{\lambda}I)Vy = 0 \quad \text{or} \quad V^TAVy = \tilde{\lambda}V^TVy.$$

LLE is mathematically equivalent to computing the lowest eigenspace of the LLE matrix $M = (I - W^T)(I - W)$. Eigenmaps seeks the lowest eigenspace of the matrix $I - \hat{W}$.

PROPOSITION 6.4. *LPP is mathematically equivalent to a projection method on Span$\{X^T\}$ applied to the normalized Laplacean matrix $\hat{L} = I - \hat{W}$, i.e., it is a projected version of eigenmaps. It will yield the exact same result as eigenmaps when Span$\{X^T\}$ is invariant under $\hat{L}$. NPP is mathematically equivalent to a projection method on Span$\{X^T\}$ applied to the matrix $(I - W^T)(I - W)$, i.e., it is a projected version of LLE. It will yield the exact same results as LLE when Span$\{X^T\}$ is invariant under $(I - W^T)(I - W)$.*

One particular case when the two methods will be mathematically equivalent is in the special situation of undersampling, i.e., when $m \geq n$ and the rank of X is equal to n. In this case X^T is of rank n and therefore the subspace Span$\{X^T\}$ is trivially invariant under $\hat{L}$.

COROLLARY 6.5. *When the column rank of X is equal to n (undersampled case) LPP is mathematically equivalent to Eigenmaps and NPP is mathematically equivalent to LLE.*

6.5. Connection to spectral clustering/partitioning. It is important to comment on a few relationships with the methods used for spectral clustering (graph partitioning) [45, 31, 14, 28]. Given a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a k-way partitioning amounts to finding k disjoint subsets $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_k$ of the vertex set $\mathcal{V}$, so that the total weights of the edges that cross different partitions are minimized, while the sizes of the subsets are roughly balanced. Formally, a k-way clustering is to minimize the following cost function:

$$\mathcal{F}(\mathcal{V}_1, \ldots, \mathcal{V}_k) = \sum_{\ell=1}^{k} \frac{\sum_{i \in \mathcal{V}_\ell, j \in \mathcal{V}_\ell^c} w_{ij}}{\sum_{i \in \mathcal{V}_\ell} d_i}, \tag{6.3}$$

where $d_i = \sum_{j \in \mathcal{V}} w_{ij}$ is the degree of a vertex i. For each term in the summation of this objective function, the numerator $\sum_{i \in \mathcal{V}_\ell, j \in \mathcal{V}_\ell^c} w_{ij}$ is the sum of the weights of edges crossing

the partition $\mathcal{V}_\ell$ and its complement $\mathcal{V}_\ell^c$, while the denominator $\sum_{i\in\mathcal{V}_\ell} d_i$ is the "size" of the partition $\mathcal{V}_\ell$.

If we define an $n \times k$ matrix Z, whose ℓ-th column is a cluster indicator of the partition $\mathcal{V}_\ell$, i.e.,

$$Z(j,\ell) = \begin{cases} 1/\sqrt{\sum_{i\in\mathcal{V}_\ell} d_i} & \text{if } j \in \mathcal{V}_\ell \\ 0 & \text{otherwise,} \end{cases} \tag{6.4}$$

then the cost function is exactly the trace of the matrix $Z^T L Z$:

$$\mathcal{F}(\mathcal{V}_1, \ldots, \mathcal{V}_k) = \mathrm{Tr}\,(Z^T L Z),$$

with Z satisfying

$$Z^T D Z = I,$$

where L (the graph Laplacean) and D are defined as before. Therefore, the clustering problem stated above can be formulated as that of finding a matrix Z in the form of (6.4) such that $\mathrm{Tr}\,(Z^T L Z)$ is minimum and $Z^T D Z = I$. This being a hard problem to solve, one usually considers a heuristic which computes a matrix Z that is no longer restricted to the form (6.4), so that the same two conditions are still satisfied. With this relaxation, the columns of Z are known to be the k smallest eigenvectors of the generalized eigenvalue problem

$$L z_i = \lambda_i D z_i. \tag{6.5}$$

The above solution Z has a natural interpretation related to Laplacean Eigenmaps. Imagine that there is a set of high dimensional data points which are sampled from a manifold. We perform dimension reduction on these data samples using the Laplacean Eigenmaps method. Then Z is the low dimensional embedding of the original manifold, that is, each sample on the manifold is mapped to a row of Z, in the k-dimensional space. Thus, a good clustering of Z in some sense implies a reasonable clustering of the original high dimensional data.

It is worthwhile to mention that by slightly modifying the cost function (6.3) we can arrive at a similar spectral problem. For this, consider minimizing the objective function

$$\hat{\mathcal{F}}(\mathcal{V}_1, \ldots, \mathcal{V}_k) = \sum_{\ell=1}^{k} \frac{\sum_{i\in\mathcal{V}_\ell, j\in\mathcal{V}_\ell^c} w_{ij}}{|\mathcal{V}_\ell|}. \tag{6.6}$$

Comparing (6.6) with (6.3), one sees that the only difference in the objective is the notion of "size of a subset": Here the number of vertices $|\mathcal{V}_\ell|$ is used to measure the size of $\mathcal{V}_\ell$, while in (6.3) this is replaced by the sum of the degree of the vertices in $\mathcal{V}_\ell$, which is related to the number of edges. Similar to the original problem, if we define the matrix $\hat{Z}$ as

$$\hat{Z}(j,\ell) = \begin{cases} 1/\sqrt{|\mathcal{V}_\ell|} & \text{if } j \in \mathcal{V}_\ell \\ 0 & \text{otherwise,} \end{cases}$$

then we get the following two equations:

$$\hat{\mathcal{F}}(\mathcal{V}_1, \ldots, \mathcal{V}_k) = \mathrm{Tr}\,(\hat{Z}^T L \hat{Z}), \qquad \hat{Z}^T \hat{Z} = I.$$

The cost function (6.6) is again hard to minimize and we can relax the minimization to obtain the eigenvalue problem:

$$L\hat{z}_i = \hat{\lambda}_i \hat{z}_i. \tag{6.7}$$

The partitioning resulting from minimizing the objective function (6.6) approximately via (6.7) is called the *ratio cut* [20]. The one resulting from minimizing (6.3) approximately via (6.5) is called the *normalized cut* [45]. We will refer to the problem of finding the ratio cut (resp. finding the normalized cut), as the *spectral ratio cut problem*, (resp. *spectral normalized cut problem*). Finding the ratio cut amounts to solving the standard eigenvalue problem related to the graph Laplacean L, while finding the normalized cut is equivalent to solving the eigenvalue problem related to the normalized Laplacean $\hat{L} = D^{-1/2}LD^{-1/2}$. This connection results from different interpretations of the "size of a set". The second smallest eigenvector $\hat{z}_2$ (the *Fiedler vector* [15, 16]) of L plays a role similar to that of vector z_2 described above. Since Z is the standard low dimensional embedding of the manifold in the high dimensional ambient space, a natural question is: Is $\hat{Z}$ also a good embedding of this manifold? As will be seen later in Section 7.2, $\hat{Z}$ is the low dimensional embedding of a "kernel" version of PCA that uses an appropriate kernel.

6.6. Unifying Framework. We now summarize the various connections that we have drawn so far. The objective functions and the constraints imposed on the optimization problems seen so far are shown in Table 6.1. As can be seen the methods can be split in two classes. The first class, which can be termed a class of 'implicit mappings', includes LLE, Laplacean Eigenmaps and ISOMAP. Here, the sought low-dimensional data set Y is obtained from solving an optimization problem of the form,

$$\begin{cases} \min\limits_{Y \in \mathbb{R}^{d\times n}} & \text{Tr}\left[YAY^T\right] \\ YBY^T = I \end{cases} \tag{6.8}$$

where B is either the identity matrix (LLE) or the matrix D (Eigenmaps). For LLE the matrix A is $A = (I - W^T)(I - W)$ and for Eigenmaps, A is the Laplacean matrix.

Method	Object. (min)	Constraint
LLE	$\text{Tr}\,[Y(I - W^T)(I - W)Y^T]$	$YY^T = I$
Eigenmaps	$\text{Tr}\,[Y(D - W)Y^T]$	$YDY^T = I$
PCA/MDS	$\text{Tr}\,[-V^TX(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)X^TV]$	$V^TV = I$
LPP	$\text{Tr}\,[V^TX(D - W)X^TV]$	$V^TXDX^TV = I$
OLPP	$\text{Tr}\,[V^TX(D - W)X^TV]$	$V^TV = I$
NPP	$\text{Tr}\,[V^TX(I - W^T)(I - W)X^TV]$	$V^TXX^TV = I$
ONPP	$\text{Tr}\,[V^TX(I - W^T)(I - W)X^TV]$	$V^TV = I$
LDA	$\text{Tr}\,[V^TX(I - H)X^TV]$	$V^TXX^TV = I$
Spect. Clust. (ratio cut)	$\text{Tr}\,[Z^T(D - W)Z]$	$Z^TZ = I$
Spect. Clust. (normalized cut)	$\text{Tr}\,[Z^T(D - W)Z]$	$Z^TDZ = I$

TABLE 6.1
Objective functions and constraints used in several dimension reduction methods.

The second class of methods, which can be termed the class of 'projective mappings' includes PCA/MDS, LPP, ONPP, and LDA, and it can be cast as an optimization problem of the form

$$\begin{cases} \min\limits_{V \in \mathbb{R}^{m\times d}} & \text{Tr}\left[V^T X A X^T V\right] \ . \\ V^T\ B\ V = I \end{cases} \tag{6.9}$$

Here, B is either the identity matrix (ONPP, PCA) or a matrix of the form XDX^T or XX^T. For ONPP the matrix A is the same as the LLE matrix $(I-W)(I-W^T)$ and for LPP, A is a Laplacean graph matrix. For LDA, $A = I - H$. For PCA/MDS the largest eigenvalues are considered so the trace is maximized instead of minimized. This means that we need to take A to be the negative identity matrix for this case. In all cases the resulting V matrix is the projector, so $Y = V^T X$ is the low-dimension data. Figure 6.1 shows pictorially the relations between the various dimension reduction methods.

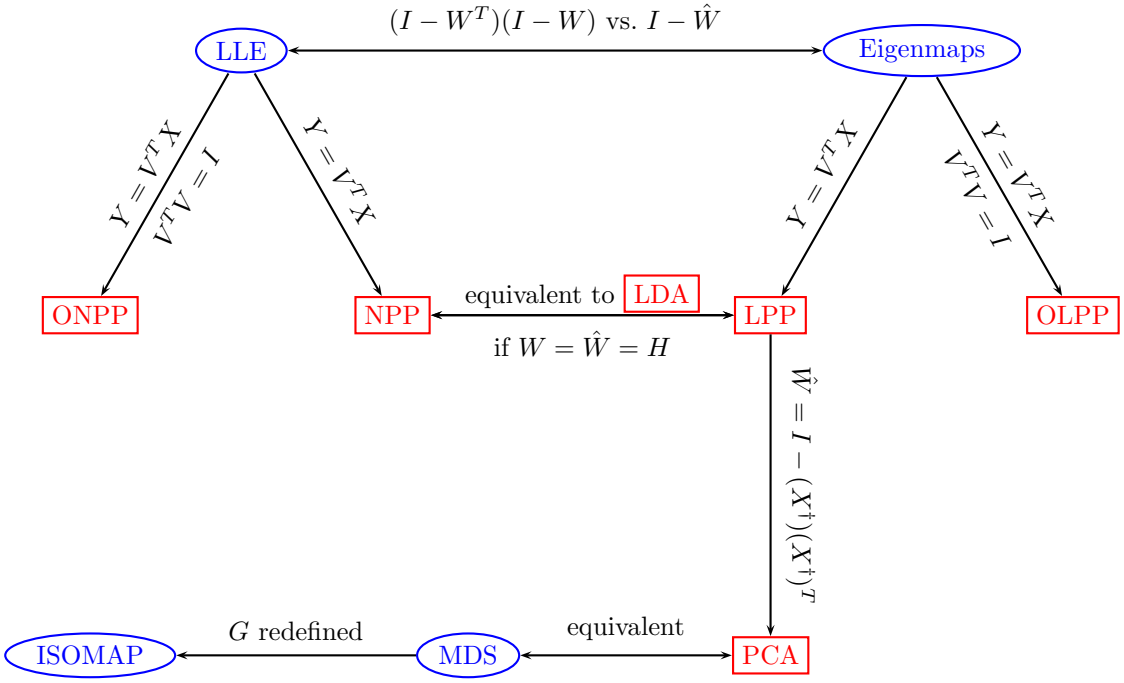

FIG. 6.1. *Relations between the different dimension reduction methods.*

7. Kernels. Kernels have been extensively used as a means to represent data by mappings that are intrinsically nonlinear, see, e.g., [30, 48, 39, 44]. Kernels are based on an implicit nonlinear mapping $\Phi : \mathbb{R}^m \rightarrow \mathcal{H}$, where $\mathcal{H}$ is a certain high-dimensional *feature space*. Denote by $\Phi(X) = [\Phi(x_1), \Phi(x_2), \ldots, \Phi(x_n)]$ the transformed data set in $\mathcal{H}$. We will also use Φ (a matrix) as a shorthand notation of $\Phi(X)$ when there is no risk of confusion with the mapping.

The Moore-Aronszajn theorem [1] indicates that for every symmetric positive definite kernel there is a dot product defined on some Hilbert space. For finite samples X, the

main idea is that the transformation Φ need only be known through its Grammian, which is symmetric positive (semi-)definite, on the data X. In other words, what is known is the matrix K whose entries are

$$K_{ij} \equiv k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle. \tag{7.1}$$

This is the Gram matrix induced by the kernel $k(x, y)$ associated with the feature space. In fact, another interpretation of the kernel mapping is that we are defining an alternative inner product in the X-space, which is expressed through the inner product of every pair (x_i, x_j) as $\langle x_i, x_j \rangle = k_{ij}$.

Formally, any of the techniques seen so far can be implemented with kernels as long as its inner workings require only inner products to be implemented. In the sequel we denote by K the kernel matrix:

$$K \equiv \Phi(X)^T \Phi(X) = [k_{i,j}]_{i,j=1,\cdots,n} = [\Phi(x_i)^T \Phi(x_j)]_{i,j=1,\cdots,n} \ . \tag{7.2}$$

7.1. Explicit mappings with kernels. Consider now the use of kernels in the context of the 'projective mappings' seen in Section 6.6. These compute a projection matrix V by solving an optimization problem of the form (6.9). Formally, if we were to work in feature space, then X in (6.9) would become $\Phi(X)$, i.e., the projected data would take the form $Y = V^T \Phi(X)$. Here $V \in \mathbb{R}^{N \times d}$, where N is the (typically large and unknown) dimension of the feature space.

The cost function (6.9) would become

$$\text{Tr} \left[V^T \Phi(X) A \Phi(X)^T V \right], \tag{7.3}$$

where A is one of the matrices defined earlier for each method. We note in passing that the matrix A, which should capture local neighborhoods, must be based on data and distances between them in the feature space.

Since $\Phi(X)$ is not explicitly known (and is of large dimension) this direct approach does not work. However, as was suggested in [25], one can exploit the fact that V can be restricted (again implicitly) to lie in the span of $\Phi(X)$, since V must project $\Phi(X)$. For example, we can implicitly use an orthogonal basis if the span of $\Phi(X)$, via an implicit QR factorization of $\Phi(X)$ as was done in [25]. In the following this factorization is avoided for simplicity.

7.2. Kernel PCA. Kernel PCA, see, e.g., [41], corresponds to performing classical PCA on the set $\{\Phi(x_i)\}$. Using $\bar{\Phi}$ to denote the matrix $[\Phi(\bar{x}_1), \ldots, \Phi(\bar{x}_n)]$, this leads to the optimization problem:

$$\max \text{Tr} \left[V^T \bar{\Phi} \bar{\Phi}^T V \right] \quad \text{subject to} \quad V^T V = I \ .$$

From what was seen before, we would need to solve the eigenvalue problem

$$\bar{\Phi} \bar{\Phi}^T u_i = \lambda u_i,$$

and the projected data will be $Y = [u_1, \ldots, u_d]^T \bar{\Phi}$.

The above problem is not solvable as is because the matrix $\bar{\Phi} \bar{\Phi}^T$ is not readily available. What is available is the Grammian $\bar{\Phi}^T \bar{\Phi}$. This suggest the following right singular vector approach. We multiply both sides of the above equation by $\bar{\Phi}^T$, which yields:

$$\underbrace{[\bar{\Phi}^T \bar{\Phi}]}_{\bar{K}} \bar{\Phi}^T u_i = \lambda_i \bar{\Phi}^T u_i$$

We stated above that the matrix $\bar{K}$ is available – but in reality since the Φ_i are not explicitly available we cannot recenter the data in feature space. However, there is no real issue because $\bar{K}$ can be expressed easily from K since $\bar{K} = \bar{\Phi}^T\bar{\Phi} = (I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)K(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)$, see [30].

Recall that $Y = V^T\bar{\Phi}$, where $V = [u_1, \cdots, u_d]$, so the vectors $\bar{\Phi}^T u_i$ in the above equation are just the transposes of the rows of the low-dimensional Y. In the end, the rows of Y, when transposed, are the largest d eigenvectors of the Gram matrix. In other words, Y is obtained by solving the largest d eigenvectors of the system

$$\bar{K}z_i = \lambda_i z_i, \qquad [z_1, \ldots, z_d] = Y^T. \tag{7.4}$$

It is interesting to compare this problem with the one obtained for the spectral ratio cut (6.7): the columns of Y^T (n-vectors) are the smallest eigenvectors of the Laplacean matrix L. Hence, it is clear that the spectral ratio cut problem can be interpreted as Kernel PCA with the kernel $K = L^\dagger$ [21, 17]. Figure 7.1 shows pictorially this relation and other ones to be discussed in the sequel.

PROPOSITION 7.1. *The kernel version of PCA, using the kernel $K = L^\dagger$, is mathematically equivalent to the spectral ratio cut problem in feature space.*

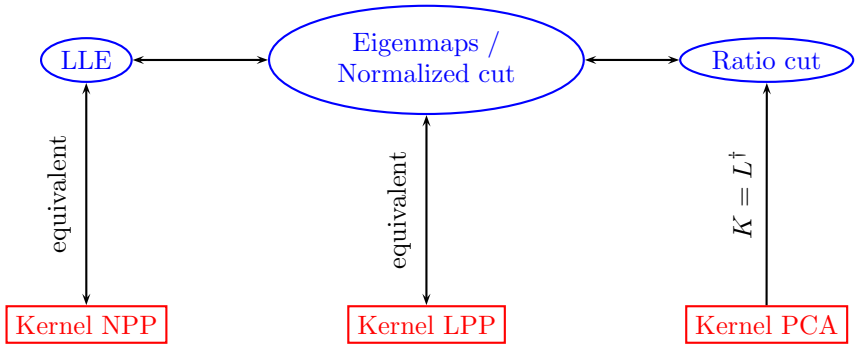

FIG. 7.1. *Kernel methods and their equivalents.*

7.3. Kernel LPP. To define a kernel version of LPP, we can proceed similarly to PCA. Denote again by Φ the system $\Phi \equiv \Phi(X)$, and let $K \equiv \Phi^T\Phi$, which is assumed to be invertible. The problem (4.9) for LPP in feature space is

$$\min_V \text{Tr}\,\left[V^T\Phi L\Phi^T V\right] \quad \text{Subj. to} \quad V^T\Phi D\Phi^T V = I$$

which leads to the eigenvalue problem:

$$\Phi L\Phi^T u_i = \lambda_i \Phi D\Phi^T u_i\ .$$

Again this is not solvable because the matrices $\Phi L\Phi^T$ and $\Phi D\Phi^T$ are not available.

Proceeding in the same was as for PCA, and assuming for simplicity that Φ is of full rank, we can left-multiply by Φ^T, then by K^{-1}, and recalling that $Y = V^T\Phi$, we obtain $Y^T = [z_2, \ldots, z_{d+1}]$ where

$$Lz_i = \lambda_i D z_i. \tag{7.5}$$

One may be puzzled by the remarkable fact that the Grammian matrix K no longer appears in the equation. It is important to recall however, that the information about distances must already be reflected in the Laplacean pair (L, D). In effect this shows the result established in [22].

PROPOSITION 7.2. *The kernel version of LPP is mathematically equivalent to Laplacean eigenmaps in feature space.*

We note that this is in fact a practical equivalence as well, i.e., the computational problems to which the two methods arrive are the same. What appeared to be a nonlinear method (eigenmaps) becomes a linear one using a kernel.

An immediate question is: do we explicitly know the related mapping? In [6] an infinite dimensional operator was used as a means to define out-of-sample extensions of various nonlinear methods. All that is needed is to find a continuous kernel $K(x, y)$ whose discretization gives rise to the discrete kernel $K(x_i, x_j)$.

7.4. Kernel ONPP. The kernel version of ONPP seeks to minimize the function

$$\min_{V \in \mathbb{R}^{L\times d},\ V^TV=I} \quad \left[V^T\Phi M\Phi^T V\right] \tag{7.6}$$

which leads to the eigenvalue problem:

$$\Phi M\Phi^T u_i = \lambda_i u_i. \tag{7.7}$$

We now again multiply by Φ^T to the left and note as before $K = \Phi^T\Phi$, and that the solution Y is such that $Y^T = \Phi^T[u_2, \ldots, u_{d+1}]$. This leads to the eigenvalue problem

$$KMz_i = \lambda_i z_i \quad \text{or} \quad Mz_i^T = K^{-1}z_i^T, \qquad [z_2, \ldots, z_{d+1}] = Y^T \tag{7.8}$$

whose solution is the set of eigenvectors of the matrix M but with a different orthogonality constraint, namely the K^{-1}-orthogonality.

In other words, the rows of the projected data Y can be directly computed as the (transposed) eigenvectors of the matrix KM associated with the smallest d eigenvalues.

Though the matrix KM in (7.8) is nonsymmetric, the problem is similar to the eigenvalue problem $Mz = \lambda K^{-1}z$ and therefore, the eigenvectors are orthogonal with respect to the K^{-1}-inner product, i.e., $z_i^T K^{-1} z_j = \delta_{ij}$. This can also be seen by introducing the Cholesky factorization of K, $K = RR^T$ and setting $\hat{z} = R^{-1}z$. The set of $\hat{z}$'s is orthogonal.

It is also useful to translate the optimization problem corresponding to the eigenvalue problem (7.8) for the Y variable. Clearly Kernel ONPP solves the optimization problem:

$$\begin{cases} \min \\ Y \in \mathbb{R}^{d\times n} \\ YK^{-1}Y^T = I \end{cases} \quad \text{Tr}\left[YMY^T\right]. \tag{7.9}$$

This new problem is again in $\mathbb{R}^n$. In practice, there is still an issue to be resolved with this new setting, namely we need a matrix $M = (I - W^T)(I - W)$ which is determined for the points in feature space. In other words the affinity matrix W should be for the points $\Phi(x_i)$ not the x_i's. Again this is easily achievable because the method for constructing W only requires local grammians which are available from K; see [24] for details.

We now address the same question as the one asked for the relation between LPP and eigenmaps in feature space. The question is whether or not performing LLE in feature

space will yield the kernel version of ONPP. Clearly, the problem (7.9) to which we arrive with kernel ONPP does not resemble the optimization problem of LLE. This is easy to understand: ONPP uses an orthogonal projection while LLE requires the embedded data to be orthogonal. If we were to enforce the same orthogonality on the y_i's as in LLE we might obtain the same result and this is indeed the case.

Recall that we defined this option in Section 4.5 and called it NPP. Consider this alternative way of defining ONPP and referred to as NPP in Section 4.5. Proceeding as above, one arrives at the following optimization problem for Kernel NPP:

$$\min_{V \in \mathbb{R}^{m\times d} \;\; V^T\Phi\Phi^T V=I} \quad \left[V^T\Phi M\Phi^T V\right]$$

which leads to the eigenvalue problem:

$$\Phi M\Phi^T u_i = \lambda_i \Phi\Phi^T u_i.$$

Multiplying by Φ^T and then by K^{-1}, we arrive again at the following problem from which the kernel matrix K has again disappeared:

$$M\Phi^T u_i = \lambda_i \Phi^T u_i \quad \rightarrow \quad Mz_i = \lambda_i z_i \tag{7.10}$$

The projected data is now identical with that obtained from LLE applied to Φ.

PROPOSITION 7.3. *Kernel NPP is equivalent to LLE performed in feature space.*

It is interesting to note that kernel methods tend to use dense kernels – as these are commonly defined as integral operators. Graph Laplaceans on the other hand are sparse and represent – inverses of integral operators. This is just the same situation one has with operators on Hilbert spaces: kernel operators are compact operators which when discretized (Nystrom) yield dense matrices, and their inverses are partial differential operators which when discretized yield sparse matrices.

7.5. What about LLE and Eigenmaps?. In principle, it would be perfectly possible to implement kernel variants of LLE and eigenmaps - since these require constructions of neighborhood matrices which can be adapted by using distances obtained from some Grammian K. However, this would be redundant with the nonlinear nature of LLE/eigenmaps. To understand this it is useful to come back to the issue of the similarity of LLE with Kernel ONPP. Comparing the two methods, one observes that the eigenvalue problems of the projective methods (PCA, LPP, ONPP,..) are $m \times m$ problems, i.e., they are in the data space. In contrast all kernel methods share with LLE and eigenmaps the fact that the eigenproblems are all $n \times n$. Thus, none of the eigenvalue problems solved by Kernel PCA, Kernel LPP, and Kernel ONPP, involves the data set X explicitly, in contrast with those eigenvalue problems seen for the non-kernel versions of the same methods. Compare for example (4.10) for the standard LPP with (7.5) for Kernel LPP or the problems (4.5) and (7.4) for PCA and kernel PCA. In essence, the data is hidden in the Gram matrix K (or its Cholesky factor R) for PCA, and/or the Laplacean pair L, D for LPP. In effect, one can consider that there is only one big class of methods which can be defined using various kernels.

7.6. The kernel effect: A toy example. To illustrate the power of kernels, it is best to take a small artificial example. We take n points ($n = 500$ in this experiment), generated so that half of the points are randomly drawn from a square centered at the origin and with

width 1.5 and the other half are generated so they lie in an annulus surrounding the square. The annulus is the region between the half disk of radius 3.5 centered at [1.0, 0] and the half disk of radius 4.5 centered at [1.0, 0]. This is shown in the first plot of Figure 7.2. The figure is in 2-D. The line shown in this first figure shows how a method based on PCA (called PDDP, see [9]) partitions the set. It fails to see the two distinct parts. In fact any linear separation will do a mediocre job here because the two sets cannot be partitioned by a straight line. What we do next is use kernels to transform the set. In fact the experiment is unusual in that we take the 2-D set and project it into a 2-Dimensional set with Kernel PCA. Recall that this is equivalent to eigenmaps with the Grammian matrix replacing the usual graph Laplacean. The method amounts to simply taking the kernel $\bar{K}$ (see Section 7.2 and equation (7.4)) and computing its largest 2 eigenvectors. This yields two vectors which after transposition yield the projected data Y. Since the dimensions of X and Y are the same there is no dimension reduction per se, but the projection will nevertheless show the effect of kernels and illustrate how they work.

We use a Gaussian kernel which we write in the form $K(x, y) = exp(-\|x - y\|_2^2/\sigma^2)$. This is a very popular kernel, see, e.g., [40]. One of the difficulties with this kernel is that it requires finding a good parameter σ. It is often suggested to select a value of σ equal to half the median of pairwise distances obtained from a large sample of points. In our case, we use all the 500 points for this purpose and call σ_0 the corresponding optimal value. In the experiment we use several values of σ around this pseudo-optimal value σ_0. Specifically we take σ^2 of the form σ_0^2/C where C takes the values: $C = 3, 2, 1, 0.5, 0.2$. The results of the related KPCA projections are shown in Figure 7.2.

When the parameter C takes values of 0.1 ($\sigma^2 \approx 27.46..$) and smaller, the resulting figures start to very much resemble the original picture. These are omitted. This experiment reveals a number of features of kernel methods in general and this particular kernel. When σ is large (C in the experiment is small), then inner-products become basically close to being constant (constant one) and so the Grammian will then be similar to the trivial one seen for PCA. This means we will tend to get results similar to those with standard PCA and this is indeed what is observed. For smaller values of σ the situation is quite different. In this case, large pairwise squared distances $\|x - y\|^2$ are amplified and the negative exponential essentially makes them close to zero. This has the effect of 'localizing' the data. For $\sigma = \sigma_0$, (leftmost figure in second row), the separation achieved between the 2 sets is quite remarkable. Now an algorithm such as K-means (see, e.g., [50]) can do a perfect job at identifying the two clusters (provided we know there are 2 such clusters) and a linear separation can also be easily achieved. This is a major reason why linear methods are not to be neglected. Note that as σ increases, the set corresponding to the annulus expands gradually from a very densely clustered set to one which reaches a better balance with the other set (for σ_0 for example). This can be explained by the fact that pairwise distances between points of the annulus are larger than those of the square.

8. Illustrative examples. The goal of this section is to demonstrate the methods just seen on a few simple examples.

8.1. Projecting digits in 2-D space. Figure 8.1 shows the dimension reduction results of a handwritten digits ('0'–'9') data set [2], which consists of 200 samples per digit. Each sample was originally represented as a 649-dimensional feature vector, including the Fourier coef-

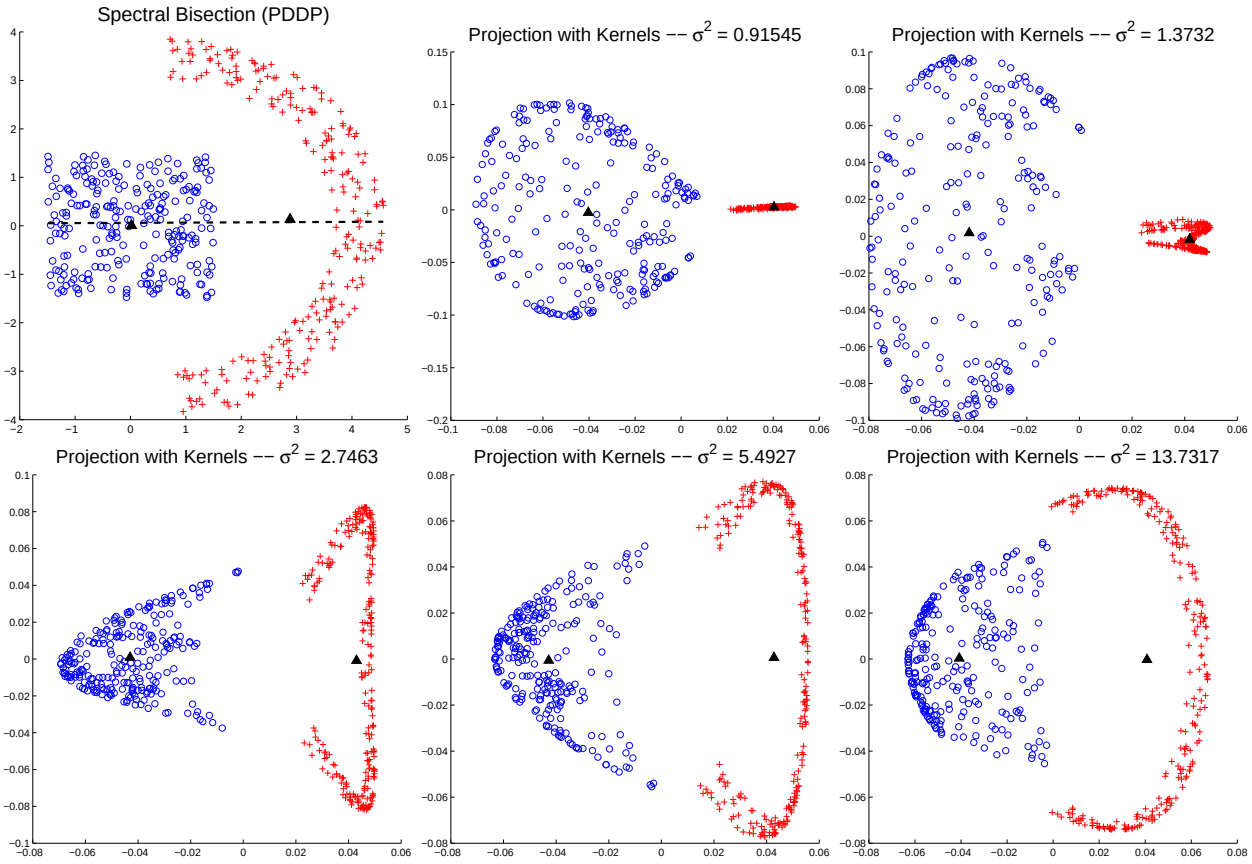

FIG. 7.2. *Original figure (top-left) and results of projections using kernels with different values of* σ

ficients, profile correlations, Karhunen-Love coefficients, pixels averages, Zernike moments, and morphological features. Due to the huge differences between the numeric ranges of the features, we normalize each feature such that the maximum value is one.

Here are the main observations from these plots. First, the supervised method LDA does well in separating the samples of different classes, as compared with the unsupervised method PCA. Both methods take into account the variances of the samples, but LDA makes a distinction between the "within scatter" and "between scatter", and outperforms PCA in separating the different classes. Second, both in theory and in practice, LLE and Eigenmaps share many similarities. For the present data set, both methods yield elongated and thin clusters. These clusters stretch out in the low dimensional space, yet each one is localized and different clusters are well separated. Our third observation concerns NPP and LPP, the linear variants of LLE and Eigenmaps, respectively. The methods should preserve locality of each cluster just as their nonlinear counterparts. They yield bigger cluster shapes instead of the "elongated and thin" ones of their nonlinear counterparts. The fourth observation is that ONPP and OLPP, the orthogonal variants of NPP and LPP, yield poorly separated projections of the data in this particular case. The samples of the same digit are distributed in a globular shape (possibly with outliers), but for different digits, samples just mingle together, yielding a rather undesirable result. Although the orthogonal projection methods OLPP and ONPP do quite a good job for face recognition (see Section 8.3.2, and results

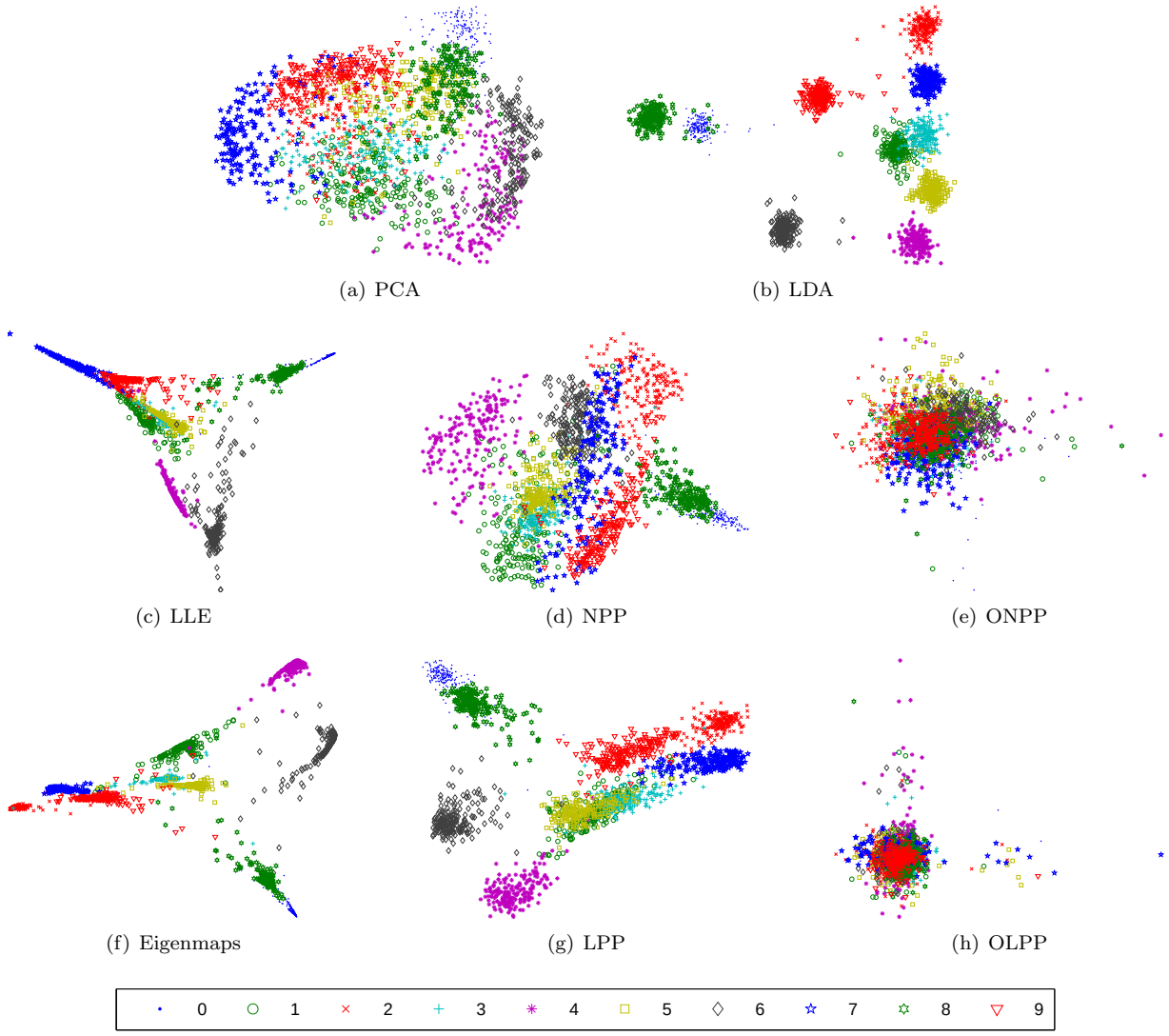

FIG. 8.1. *Low dimensional (2D) representations of handwritten digits.*

in [24]) they yield poor 2-D projections in this case. A possible explanation is that we are projecting on a space of dimension 2 only, from a high dimensional space while face recognition methods utilize much higher dimensions to successfully classify faces. The problem is also intrinsically different. In the current situation we are trying to visualize a clustering of many data items on a 2-D plane surface, whereas in classification we use the projected d-dimension data to compare a test image to other images, which are labeled. The visual clustering of the data when projected in 2-D space does not matter.

8.2. Effect of kernelization. We consider the same dataset as in Section 8.1 but now, fewer digits are taken for each experiment. Specifically we look at digits that are usually more difficult to distinguish, and we select first the 3 digits '5', '8' and '9'. We consider only two methods here, namely PCA and OLPP, and their kernel versions, K-PCA and K-OLPP.

For the kernel version we use the same Gaussian kernel $K(x, y) = exp(-\|x - y\|_2^2/\sigma^2)$ as in Section 7.6. As suggested in Section 7.6, the parameter σ is selected to be half the median

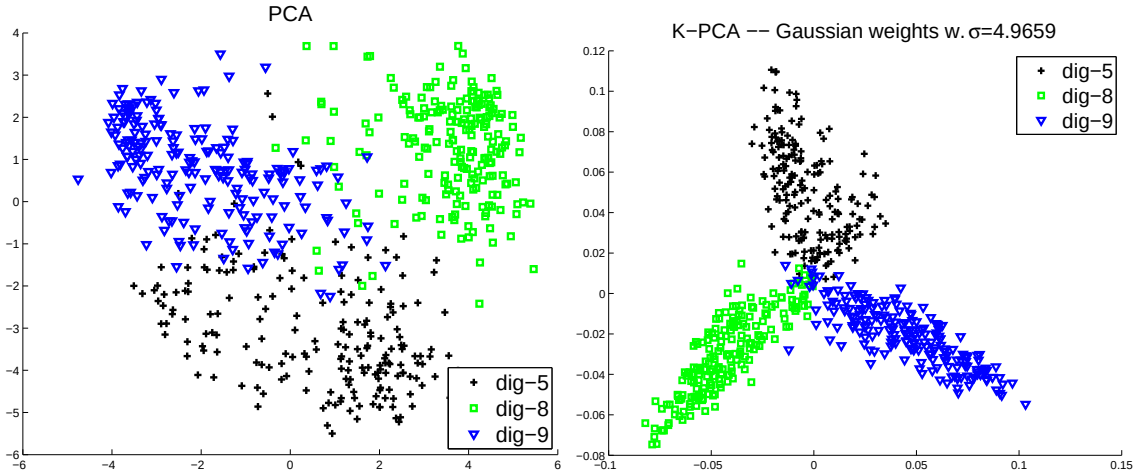

FIG. 8.2. *PCA and K-PCA for digits 5, 8, and 9 of dataset mfeat*

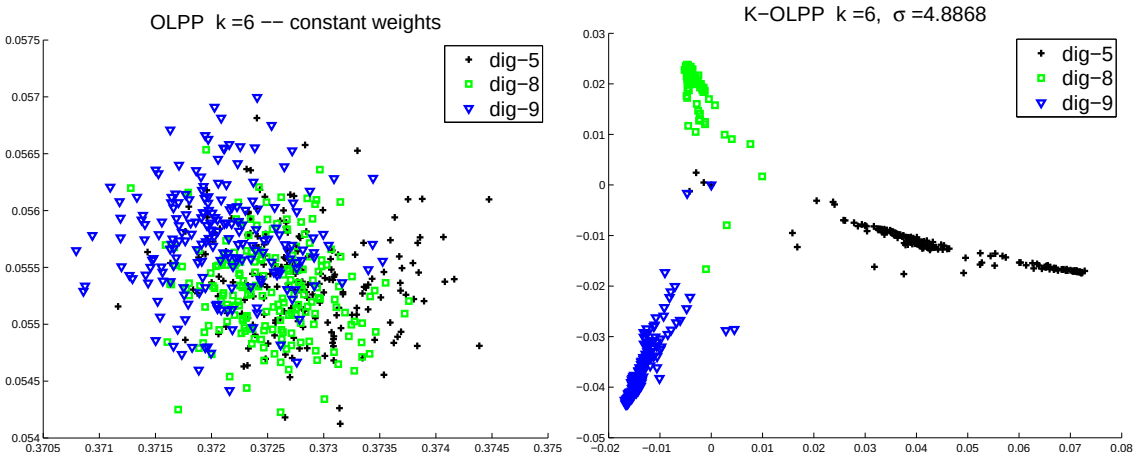

FIG. 8.3. *OLPP and K-OLPP for digits 5, 8, and 9 of dataset mfeat*

of all pairwise distances obtained from a random sample of 1000 points[4]. This typically results in a reasonable estimate of the best σ.

The improvement seen from the standard versions to the kernel versions is striking. Clearly, not all values of σ will yield a good improvement. For example when we tried taking 4 digits, the results for basically any σ were rather poor for this particular dataset.

The next test example uses another digit data set, one which is publicly available from S. Roweis' web page [5]. This dataset contains 39 samples from each class (the digits '0'-'9'). Each digit image sample is represented lexicographically as a vector in space $\mathbb{R}^{320}$ and consists of zeros and ones. Figure 8.4 shows a random sample of 20 such pictures (20 pictures randomly selected out of the whole set of 390 pictures). As can be seen a few of the prints are rather difficult to decipher.

We repeat the previous experiment but this time we select 4 digits: 1, 3, 7, 9. The results are shown in Figures 8.5 and 8.6. The kernel used here is the same as before. Since our set

[4]If the data set contains fewer than 1000 samples then all samples are used.
[5]http://www.cs.toronto.edu/roweis/data.html

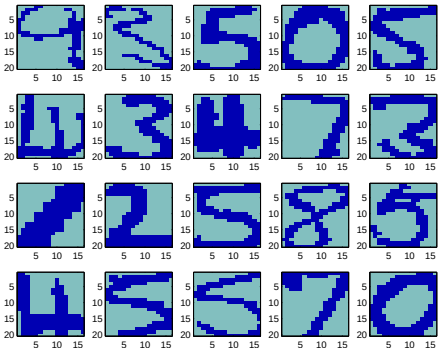

FIG. 8.4. *A sample of 20 digit images from the Roweis data set*

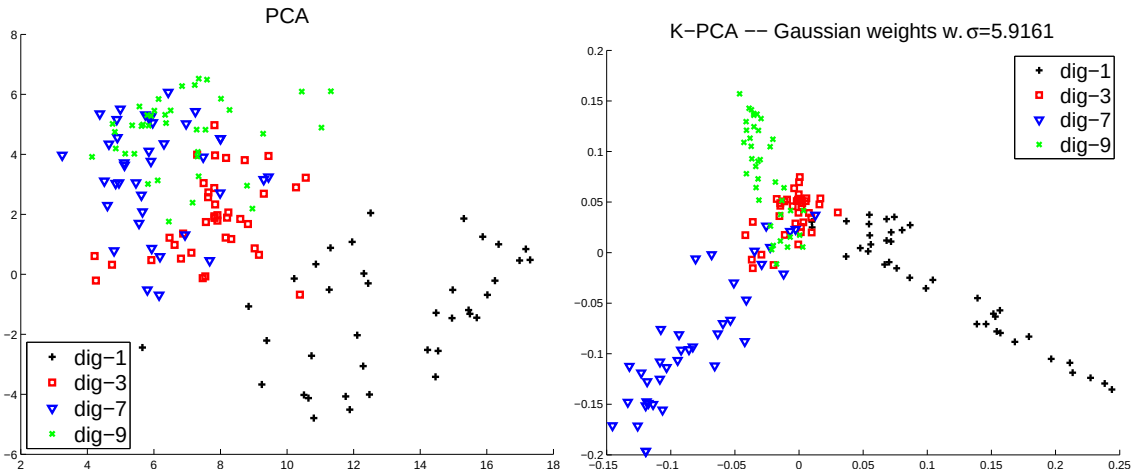

FIG. 8.5. *PCA and K-PCA for digits 1, 3, 7, 9 of the Roweis digits dataset*

is not too large (156 images in all) we simply took σ to be equal to the half the median of all pairwise distances in the set. The value of σ found in this way is shown in the corresponding plots.

The improvement seen from the standard versions to the kernel versions is remarkable. Just as before, not all values of σ will yield a good improvement.

Data set	No of classes	No of samples per class
mfeat	10	200
Roweis	10	39
UMIST	20	19-48
ORL	40	10
AR	126	8

TABLE 8.1
Data sets and their characteristics.

8.3. Classification experiments. In this section we illustrate the methods discussed in the paper on two different classification tasks; namely, digit recognition and face recognition. Recall from Section 5 that the problem of classification is to determine the class of a test sample, given the class labels of previously seen data samples (i.e., training data). Table 8.1

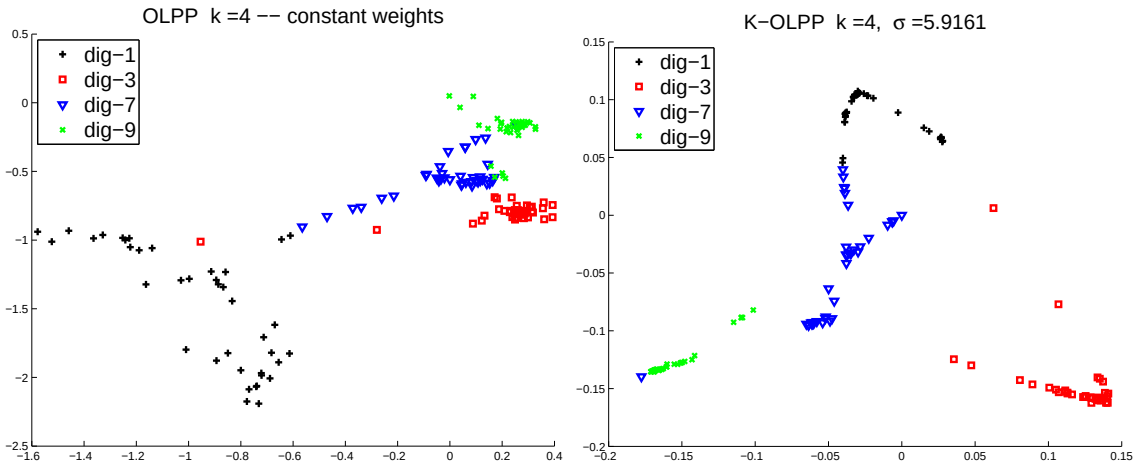

FIG. 8.6. *OLPP and K-OLPP for digits 1, 3, 7, 9 of the Roweis digits dataset*

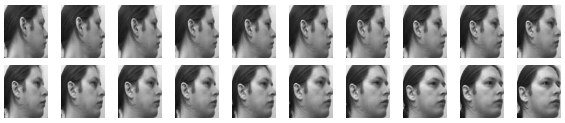

FIG. 8.7. *Sample from the UMIST database.*

summarizes the characteristics of the data sets used in our evaluation. For digit recognition, we use the `mfeat` and Roweis data sets that were previously used in Sections 8.1 and 8.2. For face recognition, we use the UMIST [18], ORL [36] and AR [29] databases. We provide more information below.

- The UMIST database contains 20 people under different poses. The number of different views per subject varies from 19 to 48. We used a cropped version of the UMIST database that is publicly available from S. Roweis' web page[6]. Figure 8.7 illustrates a sample subject from the UMIST database along with its first 20 views.
- The ORL database contains 40 individuals and 10 different images for each individual including variation in facial expression (smiling/non smiling) and pose. Figure 8.8 illustrates two sample subjects of the ORL database along with variations in facial expression and pose.
- The AR face database contains 126 individuals and 8 different images for each individual including variation in facial expression and lighting. Figure 8.9 illustrates two sample subjects of the AR database along with variations in facial expression and lighting.

In all graph-based methods we use supervised graphs, see Section 5.1. In the LPP and OLPP methods we use Gaussian weights, see Sections 7.6 and 8.2. The parameter σ is determined as described in Section 8.2. Finally, we should mention that the above methods have been pre-processed with a preliminary PCA projection step. The PCA projection is used in order to reduce the dimension of the data vectors to $n_{\text{train}} - c$, where n_{train} is the number of training samples (see e.g., [25, 24]). In what follows we discuss first recognition of handwritten digits and then face recognition. In both tasks, recognition is done in the

[6] `http://www.cs.toronto.edu/~roweis/data.html`

FIG. 8.8. *Sample from the ORL database.*

FIG. 8.9. *Sample from the AR database.*

reduced space, after dimension reduction, using nearest neighbor classification.

8.3.1. Handwritten digit recognition. This problem is of great practical importance to postal and delivery services around the world. The number of classes here is $c = 10$. We compare the linear dimension reduction methods discussed in this paper. We use 50 and 15 training samples per class in the mfeat and Roweis data sets respectively. The rest of samples are assigned to the test set.

Figure 8.10 shows the average classification error rate of all methods with respect to the dimension d of the reduced space. The averages are computed over 100 random formations of the training and test sets. Note that for LDA we only report the average performance at $d = c - 1$, as it cannot provide more than $c - 1$ discriminant axes.

First, observe that the performance of LPP parallels that of NPP. This is mostly due to Proposition 6.2, although in this case the relation $W = \hat{W} = H$ is not exactly true, due to the different weights used in each method (i.e., Gaussian weights in LPP and LLE weights in NPP). Then, notice that the orthogonal methods i.e., PCA, ONPP and OLPP offer the best performances and significantly outperform the non-orthogonal ones.

8.3.2. Face recognition. The problem of face recognition is somewhat similar to the one just described for digit recognition. We want now to recognize subjects based on facial images. Face recognition has numerous applications such as surveillance, automated screening, authentication and human-computer interaction, to name just a few.

We use 5, 10 and 5 training samples per class in the ORL, UMIST and AR data sets respectively, while the rest of samples are assigned to the test set. Figures 8.11 and 8.12 show the average classification error rates of all methods on the above three data sets. The averages are computed over 100 random formations of the training and test sets. As was previously done, for LDA we only report the average performances up to $d = c - 1$. Notice again that the orthogonal methods are in general superior to the non-orthogonal ones. Observe also that the orthogonal graph-based methods, ONPP and OLPP, are the best performers for the face recognition task.

One reason why orthogonal projection methods do well for classification may be that distances are not too distorted when projecting data. Indeed $\|V^T(x - y)\| \leq \|x - y\|$, and in fact this distance may be fairly accurate for points belonging to X due to the choice of V (e.g., when columns of V consist of the singular vectors of X as in PCA).

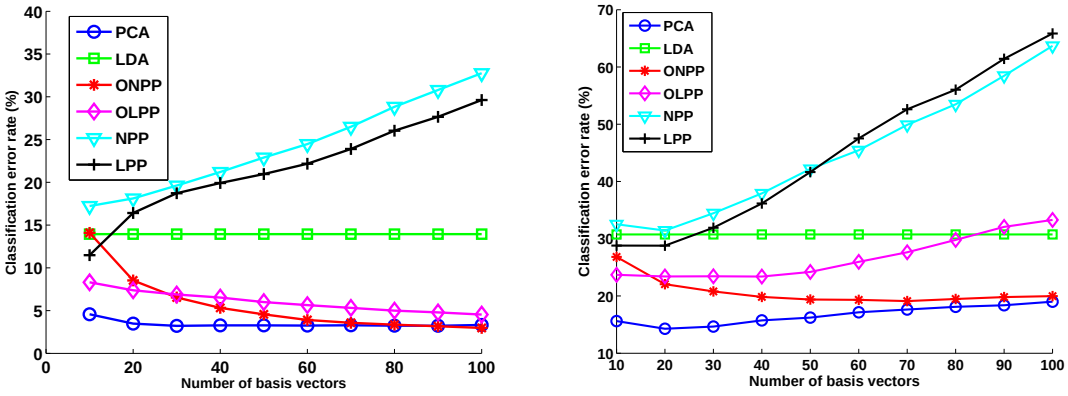

FIG. 8.10. *Handwritten digit recognition. Left panel: mfeat data set and right panel: Roweis data set.*

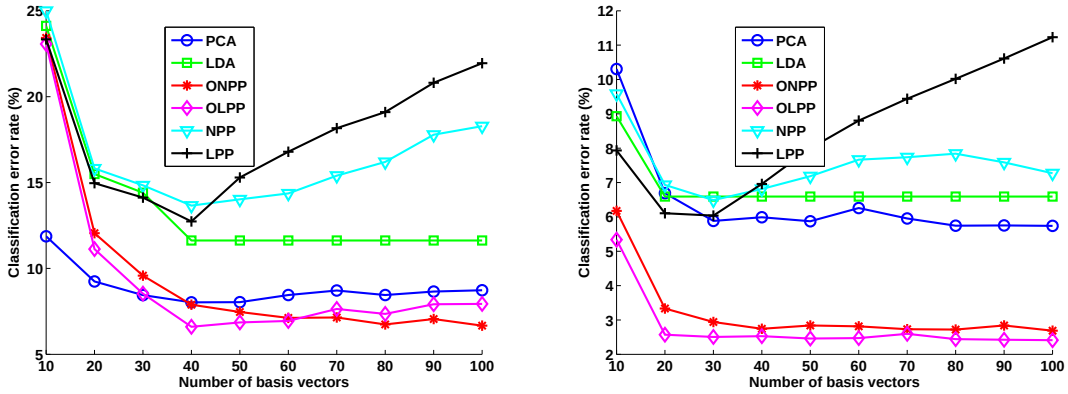

FIG. 8.11. *Face recognition results on the ORL (left) and UMIST(right) datasets.*

9. Beyond spectral methods and trace optimization. While this paper focused on dimension reduction based on spectral techniques and trace optimization, other existing powerful methods rely on convex optimization with constraints. This section briefly describes two examples in this class for illustration purposes. For a recent survey of these techniques see [7] for example.

Possibly the best known technique along these lines in supervised learning is the method of Support Vector Machines (SVM); see [8, 12, 48].

It is in spirit similar to LDA (cf. Section 5.2) in that it finds a one dimensional projection to separate the data in some optimal way. Formally, the SVM approach consists of finding a hyperplane which best separates two training sets belonging to two classes. If the hyperplane is $w^T x + b = 0$, then the classification function would be $f(x) = sign(w^T x + b)$. This will assign the value $y = +1$ to one class and $y = -1$ to the other, and it is capable of perfectly separating the two classes in ideal situations when the classes are linearly separable.

One of the key ingredients used by SVM is the notion of *margin*, which is the distance between two parallel support planes for the two classes. First, observe that the parameters w, b can be normalized by looking for hyperplanes of the form $w^T x + b \geq 1$ to include one set and $w^T x + b \leq -1$ to include the other. With $y_i = +1$ for one class and $y_i = -1$ for

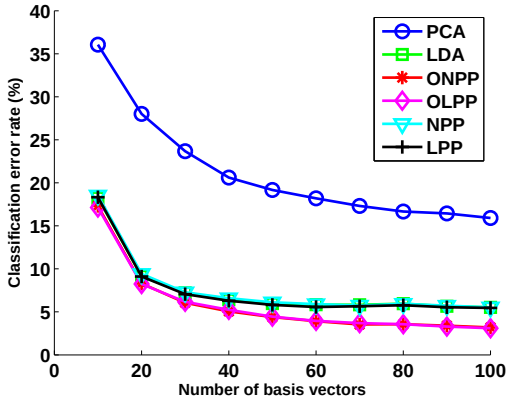

FIG. 8.12. *Face recognition results on the AR dataset.*

the other, we can write the constraints as $y_i(w^T x_i + b) \geq 1$. The margin is the maximum distance between two such planes. SVM finds w, b so that the margin is minimized.

Therefore, SVM finds the best separating hyperplane (middle of the two support planes) by maximizing the margin subjected to the constraint $y_i(w^T x_i + b) \geq 1$. As it turns out the margin is given by $\gamma = 2/\|w\|_2$. (Figure 9.1 shows an illustration.) This leads to the following constrained quadratic programming problem:

$$\begin{aligned} \min_{w.b} \quad & \frac{1}{2}\|w\|_2^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1, \ \ \forall x_i. \end{aligned}$$

Often the dual problem is solved instead of the above primal problem. In case the two classes are not separable, the constraint is relaxed by introducing slack variables. In addition, the problem is often solved in 'feature space', meaning simply that a kernel is used to redefine the inner product to enable a linear separation of the two classes.

There are several other types of optimization problems involving Semi-Definite Programming, in which the optimization problem involves matrices which are constrained to be semi positive definite. *Maximum Variance Unfolding* (MVU) is one such example; see [52, 53]. Assume we have a certain affinity graph available. We could wish to find a set of centered points in low-dimensional space (constraint: $\sum_i y_i = 0$) which maximize the variance $\sum_i \|y_i\|_2^2$ with the constraint that $\|y_i - y_j\|_2 = \|x_i - x_j\|_2$ whenever (x_i, x_j) are linked by an edge. This is a quadratic programming problem with quadratic constraints. It is possible to provide a solution in terms of the matrix Grammian of the low-dimensional data, i.e., $K = Y^T Y$. This then leads to the following semi-definite program:

$$\text{Maximize} \sum_i K_{ii} \quad \text{subject to} \quad \begin{cases} (i) & K_{ii} + K_{jj} - 2K_{ij} = \|x_i - x_j\|_2^2 \text{ if } (x_j, x_j) \in E \\ (ii) & \sum_{ij} K_{ij} = 0 \\ (iii) & K \succ 0 \end{cases}$$

Once the matrix K is found, one computes Y of dimension $d \times n$ such $Y^T Y = K$ and this involves a diagonalization of K.

We have given just two examples (one supervised, one unsupervised) of methods involving more complex techniques (i.e., optimization) than those methods seen in earlier sections,

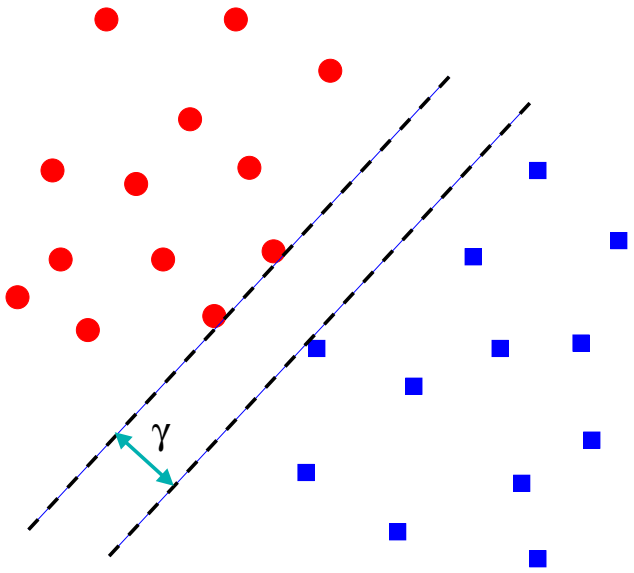

FIG. 9.1. *Illustration of the margin in SVM.*

which were based on (projected) eigenvalue problems. Many other convex optimization formulations have been discussed in, e.g., [57, 58, 3]. We point out that these optimization methods tend to be far more expensive than spectral methods and this limits their capability for handling large scale problems. For this reason, simpler techniques resorting to spectral problems are sometimes preferred. Realistic large scale system can have millions or even billions of variables and constraints and this puts them out of reach of the methods based on these sophisticated optimization techniques. A common alternative in such situations is to perform sampling on the data and reduce the problem size. This is the case for MVU, where a landmark version [51] was proposed if the sample size becomes large. Yet another alternative is to apply heuristics and/or to relax the constraints in order to find approximate solutions. In contrast, as long as the matrix is sparse, eigenvalue problems can still be efficiently solved.

10. Conclusion. This paper gave an overview of spectral problems which arise in dimension reduction methods, with an emphasis on the many interrelations between the various approaches used in the literature. These dimension reduction methods are often governed by a trace optimization problem with constraints, along with some data locality criteria. When viewed from this angle, and with the help of kernels, one can easily define a comprehensive unifying framework for dimension reduction methods. The illustrative examples shown indicate that in spite of their seemingly similar nature, these methods can yield vastly different performances for a given task.

Many challenging issues remain interesting to explore for a linear algebra specialist interested in this topic. For example, although kernels are indeed very powerful, we do not know how to select them (optimally) for a specific dataset and problem. Moreover, kernel methods lead to large $n \times n$ matrices, typically dense problems, which are difficult to handle in practice. This leads to a broader issue that remains a thorn in this area, namely the general question of computational cost. Methods considered in the literature so far have often relied on very expensive matrix factorizations, the most common being the SVD, and

in view of the ever-increasing sizes of practical datasets, it has become critical to now search for less costly alternatives.

REFERENCES

[1] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[2] A. Asuncion and D.J. Newman. UCI machine learning repository (multiple features data set). URL: `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[3] Francis Bach and Zaïd Harchaoui. Diffrac: a discriminative and flexible framework for clustering. In *Advances in Neural Information Processing Systems 20*, 2008.

[4] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, 2001.

[5] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[6] Y. Bengio, J-F Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[7] Kristin P. Bennett and Emilio Parrado-Hernandez. The interplay of optimization and machine learning research. *Journal of Machine Learning Research*, 7:1265–1281, 2006.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[9] Daniel Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, 1998.

[10] D. Cai, X. He, J. Han, and H.-J. Zhang. Orthogonal Laplacianfaces for face recognition. *IEEE Trans. on Image Processing*, 15(11):3608–3614, 2006.

[11] G. Ceder, D. Morgan, C. Fischer, K. Tibbetts, and S. Curtarolo. Data-mining-driven quantum mechanics for the prediction of structure. *MRS Bulletin*, 31:981–985, 2006.

[12] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[13] Stefano Curtarolo, Dane Morgan, Kristin Persson, John Rodgers, and Gerbrand Ceder. Predicting crystal structures with data mining of quantum calculations. *Phys. Rev. Lett.*, 91(13):135503, Sep 2003.

[14] Chris Ding. Spectral clustering. ICML 2004 tutorial, 2004.

[15] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Math. J.*, 23:298–305, 1973.

[16] M. Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its applications to graph theory. *Czechoslovak Math. J.*, 25:619–633, 1975.

[17] François Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.

[18] D. B Graham and N. M Allinson. Characterizing virtual eigensignatures for general purpose face recognition. *Face Recognition: From Theory to Applications*, 163:446–456, 1998.

[19] Yue-Fei Guo, Shi-Jin Li, Jing-Yu Yang, Ting-Ting Shu, and Li-De Wu. A generalized Foley-Sammon transform based on generalized Fisher discriminant criterion and its application to face recognition. *Pattern Recogn. Lett.*, 24(1-3):147–158, 2003.

[20] L. Hagen and A.B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 11(9):1074–1085, 1992.

[21] Jihun Ham, Daniel D. Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 47, New York, NY, USA, 2004. ACM.

[22] X. He and P. Niyogi. Locality preserving projections. *In Proc. Conf. Advances in Neural Information Processing Systems*, 2003.

[23] P. Howland and H. Park. Generalizing discriminant analysis using the generalized singular value decomposition. *IEEE Trans. on Patt. Anal. and Mach. Intel.*, 26(8):995–1006, 2004.

[24] E. Kokiopoulou and Y. Saad. Orthogonal neighborhood preserving projections. In J. Han et al., editor, *IEEE 5th Int. Conf. on Data Mining (ICDM05), Houston, TX, Nov. 27-30th*, pages 234–241. IEEE, 2005.

[25] E. Kokiopoulou and Y. Saad. Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique. *IEEE TPAMI*, 29:2143–2156, 2007.

[26] Yehuda Koren. On spectral graph drawing. In *In COCOON 03, volume 2697 of LNCS*, pages 496–508. Springer-Verlag, 2003.

[27] John A. Lee and Michel Verleysen. *Nonlinear Dimensionality Reduction*. Information Science and Statistics. Springer, 2007.

[28] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[29] A. M. Martinez and R. Benavente. The AR face database. Technical Report 24, CVC, 1998.

[30] K. R. Müller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12:181–201, 2001.

[31] Andrew Y. Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 2002.

[32] Andreas Noack. An energy model for visual graph clustering. In *Proceedings of the 11th International Symposium on*

Graph Drawing (GD 2003), LNCS 2912, pages 425–436. Springer-Verlag, 2004.
[33] B. N. Parlett. *The Symmetric Eigenvalue Problem*. Number 20 in Classics in Applied Mathematics. SIAM, Philadelphia, 1998.
[34] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.
[35] Y. Saad. *Numerical Methods for Large Eigenvalue Problems*. Halstead Press, New York, 1992.
[36] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *2nd IEEE Workshop on Applications of Computer Vision*, Sarasota FL, December 1994.
[37] L. Saul and S. Roweis. Think globally, fit locally: unsupervised learning of nonlinear manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.
[38] L.K. Saul, K.Q. Weinberger, J.H. Ham, F. Sha, and D.D. Lee. Spectral methods for dimensionality reduction. In B. Schöelkopf, O. Chapelle, and A. Zien, editors, *Semisupervised Learning*. 2006.
[39] Bernhard Schlköpf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2001.
[40] B. Schölkopf and A. Smola. *Learning with Kernels*. The MIT press, Cambridge, Massachusetts, 2002.
[41] B. Schölkopf, A. Smola, and K. Muller. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural computation*, 10:1299–1319, 1998.
[42] S. E. Sebastian, N. Harrison, C. D. Batista, L. Balicas, M. Jaime, P. A. Sharma, N. Kawashima, and I. R. Fisher. Dimensional reduction at a quantum critical point. *Nature*, 441:617, 2006.
[43] F. Sha and L. K. Saul. Analysis and extension of spectral methods for nonlinear dimensionality reduction. In *Proceedings of the Twenty Second International Conference on Machine Learning (ICML)*, 2005.
[44] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
[45] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(8):888–905, 2000.
[46] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.
[47] Warren S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4), 1952.
[48] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.
[49] Huan Wang, S.C. Yan, D.Xu, X.O. Tang, and T. Huang. Trace ratio vs. ratio trace for dimensionality reduction. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007*, pages 17–22, 2007.
[50] A. Webb. *Statistical Pattern Recognition, 2nd edition*. J. Wiley & sons, Hoboken, NJ, 2002.
[51] K. Weinberger, B. Packer, and L. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, 2005.
[52] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-04)*, volume 2, pages 988–995, 2004.
[53] K. Q. Weinberger and L. K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *AAAI'06: proceedings of the 21st national conference on Artificial intelligence*, pages 1683–1686. AAAI Press, 2006.
[54] K.Q. Weinberger and L.K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.
[55] Christopher K.I. Williams. On a connection between kernel PCA and metric multidimensional scaling. *Machine Learning*, 46(1-3):11–19, 2002.
[56] Shiming Xiang, Feiping Nie, and Changshui Zhang. Learning a mahalanobis distance metric for data clustering and classification. *Pattern Recognition*, 41(12):3600 – 3612, 2008.
[57] Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems 17*, 2005.
[58] Linli Xu and Dale Schuurmans. Unsupervised and semi-supervised multi-class support vector machines. In *Proceedings of the 20th National Conference on Artificial Intelligence*, 2005.
[59] Shuicheng Yan and Xiaoou Tang. Trace quotient problems revisited. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Proceedings of the European Conference on Computer Vision*, volume 2 of *Lecture Notes in Computer Science, Number 3952*, pages 232–244, Berlin-Heidelberg, 2006. Springer Verlag.
[60] Zhenyue Zhang and Hongyuan Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal on Scientific Computing*, 26(1):313–338, 2005.