Report No. UIUCDCS-R-2006-2747 **UILU-ENG-2006-1787**

Isometric Projection

by

Deng Cai, Xiaofei He, and Jiawei Han

July 2006

Isometric Projection*

Deng Cai† Xiaofei He‡ Jiawei Han†

† Department of Computer Science, University of Illinois at Urbana-Champaign

‡ Yahoo! Research Labs

Abstract

Recently the problem of dimensionality reduction has received a lot of interests in many fields of information processing, including data mining, information retrieval, and pattern recognition. We consider the case where data is sampled from a low dimensional manifold which is embedded in high dimensional Euclidean space. The most popular manifold learning algorithms include Locally Linear Embedding, ISOMAP, and Laplacian Eigenmap. However, these algorithms are nonlinear and only provide the embedding results of training samples. In this paper, we propose a novel linear dimensionality reduction algorithm, called **Isometric Projection**. Isometric Projection constructs a weighted data graph where the weights are discrete approximations of the geodesic distances on the data manifold. A linear subspace is then obtained by preserving the pairwise distances. Our algorithm can be performed in either original space or reproducing kernel Hilbert space, which leads to Kernel Isometric Projection. In this way, Isometric Projection can be defined everywhere. Comparing to Principal Component Analysis (PCA) which is widely used in data processing, our algorithm is more capable of discovering the intrinsic geometrical structure. Specially, PCA is optimal only when the data space is linear, while our algorithm has no such assumption and therefore can handle more complex data space. We present experimental results of the algorithm applied to synthetic data set as well as real life data. These examples illustrate the effectiveness of the proposed method.

1 Introduction

Dimensionality reduction has been a key problem in many fields of information processing, such as data mining, information retrieval, and pattern recognition. When data is represented as points in a high-dimensional space, one is often confronted with tasks like nearest neighbor search. Many methods have been proposed to index the data for fast query response, such as K-D tree, R tree, R* tree, etc [7]. However, these methods can only operate with small dimensionality, typically less than 100. The

* The work was supported in part by the U.S. National Science Foundation NSF IIS-03-08215/IIS-05-13678. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

effectiveness and efficiency of these methods drop exponentially as the dimensionality increases, which is commonly referred to as the "curse of dimensionality".

During the last decade, with the advances in computer technologies and the advent of the World Wide Web, there has been an explosion in the amount and complexity of digital data being generated, stored, analyzed, and accessed. Much of this information is multimedia in nature, including text, image, and video data. The multimedia data is typically of very high dimensionality, ranging from several thousands to several hundreds of thousand. Learning in such a high dimensional in many cases is almost infeasible. Thus, learnability necessitates dimensionality reduction. Once the high-dimensional data is mapped into lower-dimensional space, conventional indexing schemes can then be applied.

One of the most popular dimensionality reduction algorithms might be Principal Component Analysis (PCA) [12]. PCA performs dimensionality reduction by projecting the original n-dimensional data onto the $d(\ll n)$-dimensional linear subspace spanned by the leading eigenvectors of the data's covariance matrix. Its goal is to find a set of mutually orthogonal basis functions that capture the directions of maximum variance in the data so that the pairwise *Euclidean* distances can be best preserved. If the data is embedded in a linear subspace, PCA is guaranteed to discover the dimensionality of the subspace and produces a compact representation. PCA has been widely applied in data mining [16], information retrieval [5], multimedia [14], etc.

In many real world databases, however, there is no evidence that the data is sampled from a linear subspace. For example, it is always believed that the face images are sampled from a nonlinear low-dimensional manifold which is embedded in the high-dimensional ambient space [9]. This motivates us to consider manifold based techniques for dimensionality reduction. Recently, various manifold learning techniques, such as ISOMAP [21], Locally Linear Embedding (LLE) [18] and Laplacian Eigenmap [2] have been proposed which reduce the dimensionality of a *fixed* training set in a way that maximally preserve certain inter-point relationships. LLE and Laplacian Eigenmap are local methods which attempt to preserve local geometry of the data; essentially, they seek to map nearby points on the manifold to nearby points in the low-dimensional representation. ISOMAP is a global method which attempts to preserve geometry at all scales, mapping nearby points on the manifold to nearby points in low-dimensional space, and faraway points to faraway points. One of the major limitations of these methods is that they do not generally provide a functional mapping between the high and low dimensional spaces that are valid both on and off the training data. Moreover, these methods are computationally expensive and may not be able to handle large scale databases.

In this paper, we propose a novel dimensionality reduction algorithm called **Isometric Projection** (IsoProjection), which explicitly takes into account the manifold structure. To model the manifold structure, we first construct a nearest neighbor graph of the observed data. We then compute shortest paths in the graph for all pairs of data points. The shortest-paths computation gives an estimate of the global metric structure. Using techniques from Multi-Dimensional Scaling (MDS) and requiring the mapping function to be linear, we finally obtain Isometric Projection. IsoProjection can operate in either original data space or reproducing kernel Hilbert space (RKHS) which leads to Kernel IsoProjection.

With a nonlinear kernel, kernel IsoProjection is capable of discovering nonlinear structure of the data manifold. More crucially, kernel IsoProjection is defined everywhere.

The points below highlight several aspects of the paper:

- IsoProjection provides an optimal linear approximation to the true isometric embedding of the underlying data manifold. It tends to give a more faithful representation of the data's global structure than PCA does.
- IsoProjection is linear. It is computationally tractable. It can be obtained by solving an eigenvector problem.
- IsoProjection, as well as its nonlinear extension, is defined everywhere. Therefore, query points can also be mapped into the low-dimensional representation space in which retrieval, clustering and classification may be performed.
- IsoProjection is fundamentally based on ISOMAP [21], but ISOMAP does not have properties (2) and (3) above.

The remainder of the paper is organized as follows. In Section 2, we provide some back materials for manifold based dimensionality reduction. Section 3 introduces our proposed IsoProjection algorithm. Section 4 describes its nonlinear extension, kernel IsoProjection. The extensive experimental results are presented in Section 5. Finally, we provide some concluding remarks and suggestions for future work in Section 6.

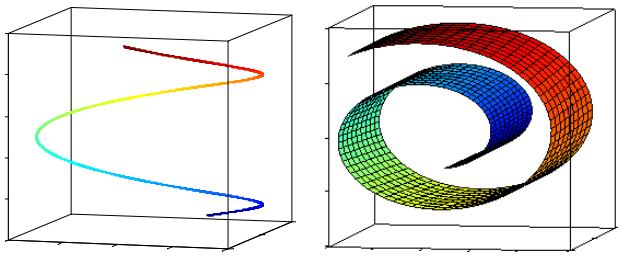
2 Background

In this section, we provide mathematical background of manifold based dimensionality reduction, as well as its effect on some potential applications like retrieval, clustering and classification. For a detailed treatment of manifolds, please see [10].

2.1 Manifold based Dimensionality Reduction

Data are generally represented as points in high-dimensional vector space. For example, a 32×32 image can be represented by a 1024-dimensional vector. Every element of the vector corresponds to a pixel. A text document can be represented by a term vector. In many cases of interests, the data may not fill the whole ambient space, but reside on or near a submanifold embedded in the ambient space. One hopes then to estimate geometrical and topological properties of the submanifold from random samples ("scattered data") lying on this unknown submanifold. The formal definition of manifold is as follows.

Definition An p-dimensional manifold (denoted by $\mathcal{M}^p$) is a topological space that is locally Euclidean. That is, around every point, there is a neighborhood that is topologically the same as the open unit ball in $\mathbb{R}^p$.



(a) (b)

Figure 1: Examples of one-dimensional manifold (a) and two-dimensional manifold (b). Both of them are embedded in the three-dimensional ambient space.

Figure 1 gives examples of manifold with dimensionality 1 and 2. In order to compute distances on the manifold, one needs to equip a metric to the topological manifold. A manifold possessing a metric is called *Riemannian Manifold*, and the metric is commonly referred to as *Riemannian Metric*.

Definition Suppose for every point $\mathbf{x}$ in a manifold $\mathcal{M}$, an inner product $\langle \cdot, \cdot \rangle_x$ is defined on a tangent space $T_x\mathcal{M}$ of $\mathcal{M}$ at $\mathbf{x}$. Then the collection of all these inner products is called the Riemannian metric.

Once the Riemannian metric is defined, one is allowed to measure the lengths of the tangent vectors $\mathbf{v} \in T_x\mathcal{M}$:

$$\|\mathbf{v}\|^2 = \langle \mathbf{v}, \mathbf{v} \rangle$$

For every smooth curve $r : [a, b] \to \mathcal{M}$, we have tangent vectors:

$$r'(t) = \frac{dr}{dt} \in T_{r(t)}\mathcal{M}$$

and can therefore use the Riemannian metric (inner product of the tangent spaces) to define their lengths. We can then define the length of r from a to b:

$$length(r) = \int_a^b \|\frac{dr}{dt}\| dt = \int_a^b \|r'(t)\| dt$$

Note that, a Riemannian metric is not a distance metric on $\mathcal{M}$. However, for a connected manifold, it is the case that every Riemannian metric induces a distance matric on $\mathcal{M}$, i.e. *Geodesic Distance*.

Definition The geodesic distance $d_{\mathcal{M}}(a, b)$ is defined as the length of the shortest curve connecting a and b.

In the plane, the geodesics are straight lines. On the sphere, the geodesics are great circles (like the equator). Suppose $\mathcal{M}^p$ is embedded in a n-dimensional Euclidean space $\mathbb{R}^n$ $(p \leq n)$. Let us consider a low dimensional map, $f : \mathbb{R}^n \to \mathbb{R}^d (d \leq n)$, and the f has a support on a submanifold $\mathcal{M}^p$, i.e.

$supp(f) = \mathcal{M}^p$. Note that, $p \leq d \leq n$, and p is generally unknown. Let d_{R^d} denote the standard Euclidean distance measure in $\mathbb{R}^d$. In order to preserve the intrinsic (invariant) geometrical structure of the data manifold, we seek a function f such that:

$$d_{\mathcal{M}^p}(\mathbf{x}, \mathbf{y}) = d_{R^d}(f(\mathbf{x}), f(\mathbf{y})) \tag{1}$$

In this paper, we are particularly interested in linear mappings, i.e. projections. The reason is for its simplicity. And more crucially, the same derivation can be performed in reproducing kernel Hilbert space (RKHS) which naturally leads to its nonlinear extension.

2.2 Potential Applications

Dimensionality reduction is often considered as a data pre-processing. After that, retrieval, clustering and classification can be performed in the lower dimensional subspace.

In information retrieval, the most commonly used strategy is *query by example*. Give a dataset $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$, the query-by-example process can be formally stated below:

1. The user submits a query $\mathbf{q}$.
2. Compute the distance between $\mathbf{x}_i$ and $\mathbf{q}$ according to some pre-defined distance measure d, $i = 1, \cdots, m$. Sort $d(\mathbf{q}, \mathbf{x}_i)$ in increasing order. Let $r(\mathbf{x}_i)$ be the rank of $\mathbf{x}_i$.
3. Return the top k matches, $\mathcal{R}(k, \mathbf{q}, \mathcal{X}) = \{\mathbf{x}_i | r(\mathbf{x}_i) \leq k\}$.

As can be seen, a key step of the above process is the distance measure. In practice, one can only consider simple distance measures for fast query response, such as Euclidean distance, although it may not reflect the intrinsic geometrical structure. In this paper, however, by using Isometric Projection, the Euclidean distances in the low dimensional subspace provide a faithful approximation to the geodesic distances on the intrinsic data manifold.

Clustering is an unsupervised learning problem. It aims at grouping objects with some common properties. For example, document clustering aims at grouping documents sharing the same topics. The K-means algorithm is one of the most popular iterative descent clustering methods. Let $C(i)$ be an assignment of class label of $\mathbf{x}_i$, $i = 1, \cdots, m$. K-means tries to minimize the following objective function:

$$\min \sum_{k=1}^{K} \sum_{C(i)=k} d(\mathbf{x}_i, \mathbf{m}_k)$$

where $\mathbf{m}_k$ is the center of the k-th cluster. The performance of K-means is essentially determined by the choice of the distance measure. Recently, there has been considerable interest in spectrally based techniques to data clustering due to their good performance [15], [20]. Spectral clustering has very close tie to spectral dimensionality reduction. In fact, spectral clustering can be though of as a combination of spectral dimensionality reduction and traditional clustering algorithms such as K-means. The rationale

behind spectral clustering reside in the fact that, after dimensionality reduction, Euclidean distances in the subspace can better describe the intrinsic relationships between objects than those in the original ambient space. Therefore, it is expected that good clustering performance can be achieved in the subspace obtained by our Isometric Projection algorithm.

3 Isometric Projection

In this section, we introduce a novel dimensionality reduction algorithm, called Isometric Projection. We begin with a formal definition of the problem of dimensionality reduction.

3.1 The Problem

The generic problem of dimensionality reduction is the following. Given a set of points $\mathbf{x}_1, \cdots, \mathbf{x}_m$ in $\mathbb{R}^n$, find a mapping function that maps these m points to a set of points $\mathbf{y}_1, \cdots, \mathbf{y}_m$ in $\mathbb{R}^d$ ($d << n$), such that $\mathbf{y}_i$ "represents" $\mathbf{x}_i$, where $\mathbf{y}_i = f(\mathbf{x}_i)$. Our method is of particular applicability in the special case where $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m \in \mathcal{M}$ and $\mathcal{M}$ is a nonlinear manifold embedded in $\mathbb{R}^n$.

In this section, we consider that f is linear. In the next section, we will describe its nonlinear extension using kernel techniques.

3.2 The Objective Function of Isometric Projection

We define $X = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m)$ and $f(\mathbf{x}) = \mathbf{a}^T\mathbf{x}$. Sometimes the rank of X is less than the number of dimensions (n). In this case, we can apply Singular Value Decomposition (SVD) to project them into a lower dimensional subspace without losing any information. We have:

$$X = U\Sigma V^T$$

where $\Sigma = diag(\sigma_1, \cdots, \sigma_r)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$ are the singular values of X, $U = [\mathbf{u}_1, \cdots, \mathbf{u}_r]$ and $\mathbf{u}_i$ is called left singular vectors, $V = [\mathbf{v}_1, \cdots, \mathbf{v}_r]$ and $\mathbf{v}_i$ is called right singular vectors. We project the data points $\mathbf{x}_i$ ($i = 1, \cdots, m$) into the SVD subspace by throwing away the components corresponding to zero singular value. We denote by W_{SVD} the transformation matrix of SVD, $W_{SVD} = U$. By SVD projection, The rank of the new data matrix is equal to the number of features (dimensions). Note that, this step is used to guarantee that matrix XX^T is non-singular. When the number of data points (m) is large than the number of features (n), XX^T is usually non-singular. In such case, this step is not necessary. For the sake of simplicity, we still use X to denote the data in the SVD subspace in the following.

Let $d_{\mathcal{M}}$ be the geodesic distance measure on $\mathcal{M}$ and d the standard Euclidean distance measure in $\mathbb{R}^d$. Isometric Projection aims to find a Euclidean embedding such that Euclidean distances in $\mathbb{R}^d$ can

provide a good approximation to the geodesic distances on $\mathcal{M}$. That is,

$$f^{opt} = \arg\min_f \sum_{i,j} \Big(d_{\mathcal{M}}(\mathbf{x}_i, \mathbf{x}_j) - d\big(f(\mathbf{x}_i), f(\mathbf{x}_j)\big) \Big)^2 \tag{2}$$

In real life data set, the underlying manifold $\mathcal{M}$ is often unknown and hence the geodesic distance measure is also unknown. In order to discover the intrinsic geometrical structure of $\mathcal{M}$, we first construct a graph G over all data points to model the local geometry. There are two choices[1]:

1. ϵ-graph: we put an edge between i and j if $d(\mathbf{x}_i, \mathbf{x}_j) < \epsilon$.
2. kNN-graph: we put an edge between i and j if $\mathbf{x}_i$ is among k nearest neighbors of $\mathbf{x}_j$ or $\mathbf{x}_j$ is among k nearest neighbors of $\mathbf{x}_i$.

Once the graph is constructed, the geodesic distances $d_{\mathcal{M}}(i,j)$ between all pairs of points on the manifold $\mathcal{M}$ can be estimated by computing their shortest path distances $d_G(i,j)$ on the graph G. The procedure is as follows: initialize $d_G(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_i, \mathbf{x}_j)$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are linked by an edge; $d_G(\mathbf{x}_i, \mathbf{x}_j) = \infty$ otherwise. Then for each value of $p = 1, 2, \cdots, m$ in turn, replace all entries $d_G(\mathbf{x}_i, \mathbf{x}_j)$ by

$$\min \big\{ d_G(\mathbf{x}_i, \mathbf{x}_j),\ d_G(\mathbf{x}_i, \mathbf{x}_p) + d_G(\mathbf{x}_p, \mathbf{x}_j) \big\}.$$

The matrix of final values $D_G = \{d_G(\mathbf{x}_i, \mathbf{x}_j)\}$ will contain the shortest path distances between all pairs of points in G. This procedure is named Floyd-Warshall algorithm [4]. More efficient algorithms exploiting the sparse structure of the neighborhood graph can be found in [8].

In the following, we apply techniques from Multi-Dimensional Scaling (MDS) to convert distances to inner products, which uniquely characterize the geometry of the data in a form that supports efficient optimization [12]. We have the following theorem:

Theorem 1 *Let D be the distance matrix such that D_{ij} is the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Define matrix $S_{ij} = D_{ij}^2$ and $H = I - \frac{1}{m}\boldsymbol{e}\boldsymbol{e}^T$ where I is the identity matrix and $\boldsymbol{e}$ is the vector of all ones. It can be shown that $\tau(D) = -HSH/2$ is the inner product matrix. That is, $D_{ij}^2 = \tau(D)_{ii} + \tau(D)_{jj} - 2\tau(D)_{ij}$, $\forall\ i, j$.*

Proof We have:

$$\begin{aligned} \tau(D) &= -\frac{1}{2}HSH = -\frac{1}{2}(I - \frac{1}{m}\mathbf{e}\mathbf{e}^T)S(I - \frac{1}{m}\mathbf{e}\mathbf{e}^T) \\ &= -\frac{1}{2}(S - \frac{1}{m}\mathbf{e}\mathbf{e}^T S - \frac{1}{m}S\mathbf{e}\mathbf{e}^T + \frac{1}{m^2}\mathbf{e}\mathbf{e}^T S\mathbf{e}\mathbf{e}^T) \end{aligned}$$

and

$$\tau(D)_{ij} = -\frac{1}{2}\left(S_{ij} - \frac{1}{m}\sum_{i=1}^m S_{ij} - \frac{1}{m}\sum_{j=1}^m S_{ij} + \frac{1}{m^2}\sum_{i=1}^m\sum_{j=1}^m S_{ij} \right).$$

[1] Under supervised situation, more restrictions can be imposed that require the edge only be put between data points which share the same label.

Since

$$\begin{aligned}
S_{ij} = D_{ij}^2 &= \|\mathbf{x}_i - \mathbf{x}_j\|^2 = (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) \\
&= \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\mathbf{x}_j + \mathbf{x}_j^T\mathbf{x}_j \\
\sum_{i=1}^{m} S_{ij} &= \sum_{i=1}^{m} \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_j^T\sum_{i=1}^{m}\mathbf{x}_i + m\mathbf{x}_j^T\mathbf{x}_j \\
\sum_{j=1}^{m} S_{ij} &= m\mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\sum_{j=1}^{m}\mathbf{x}_j + \sum_{j=1}^{m}\mathbf{x}_j^T\mathbf{x}_j \\
\sum_{i=1}^{m}\sum_{j=1}^{m} S_{ij} &= m\sum_{i=1}^{m}\mathbf{x}_i^T\mathbf{x}_i - 2\sum_{j=1}^{m}\mathbf{x}_j^T\sum_{i=1}^{m}\mathbf{x}_i + m\sum_{j=1}^{m}\mathbf{x}_j^T\mathbf{x}_j
\end{aligned}$$

and note that

$$\sum_{i=1}^{m}\mathbf{x}_i^T\mathbf{x}_i = \sum_{j=1}^{m}\mathbf{x}_j^T\mathbf{x}_j$$

and

$$\bar{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i = \frac{1}{m}\sum_{j=1}^{m}\mathbf{x}_j,$$

we have:

$$\tau(D)_{ij} = \mathbf{x}_i^T\mathbf{x}_j - \mathbf{x}_i^T\bar{\mathbf{x}} - \mathbf{x}_j^T\bar{\mathbf{x}} + \bar{\mathbf{x}}^T\bar{\mathbf{x}} = (\mathbf{x}_i - \bar{\mathbf{x}})^T(\mathbf{x}_j - \bar{\mathbf{x}})$$

Thus, we have:

$$D_{ij}^2 = \tau(D)_{ii} + \tau(D)_{jj} - 2\tau(D)_{ij}, \forall\ i, j$$

The matrix H is often called "centering matrix". Let D_Y denote the Euclidean distance matrix in the reduced subspace, and $\tau(D_Y)$ be the corresponding inner product matrix. Thus, the objective function (2) becomes minimizing the following:

$$\|\tau(D_G) - \tau(D_Y)\|_{L^2} \tag{3}$$

where $\|A\|_{L^2}$ is the L^2 matrix norm $\sqrt{\sum_{i,j} A_{i,j}^2}$.

3.3 Learning Isometric Projections

Consider a linear function $f(\mathbf{x}) = \mathbf{a}^T\mathbf{x}$. Let $y_i = f(\mathbf{x}_i)$ and $Y = (y_1, \cdots, y_m) = \mathbf{a}^T X$. Thus, we have

$$\tau(D_Y) = Y^T Y = X^T\mathbf{a}\mathbf{a}^T X$$

The optimal projection is given by solving the following minimization problem:

$$\mathbf{a}^* = \min_{\mathbf{a}} \|\tau(D_G) - X^T\mathbf{a}\mathbf{a}^T X\|^2 \tag{4}$$

Following some algebraic steps and noting $tr(A) = tr(A^T)$, we see that:

$$\begin{aligned} & \|\tau(D_G) - X^T\mathbf{a}\mathbf{a}^T X\|^2 \\ = \; & tr\Big(\Big(\tau(D_G) - X^T\mathbf{a}\mathbf{a}^T X\Big)\Big(\tau(D_G) - X^T\mathbf{a}\mathbf{a}^T X\Big)^T\Big) \\ = \; & tr\Big(\tau(D_G)\tau(D_G)^T - X^T\mathbf{a}\mathbf{a}^T X\tau(D_G)^T - \\ & \quad \tau(D_G)X^T\mathbf{a}\mathbf{a}^T X + X^T\mathbf{a}\mathbf{a}^T XX^T\mathbf{a}\mathbf{a}^T X\Big) \end{aligned}$$

Note that, the magnitude of $\mathbf{a}$ is of no real significance because it merely scales y_i. Therefore, we can impose a constraint as follows:

$$\mathbf{a}^T XX^T\mathbf{a} = 1$$

Thus, we have

$$\begin{aligned} & tr\Big(X^T\mathbf{a}\mathbf{a}^T XX^T\mathbf{a}\mathbf{a}^T X\Big) \\ = \; & tr\Big(\mathbf{a}^T XX^T\mathbf{a}\mathbf{a}^T XX^T\mathbf{a}\Big) \\ = \; & 1 \end{aligned}$$

And,

$$\begin{aligned} & \|\tau(D_G) - X^T\mathbf{a}\mathbf{a}^T X\|^2 \\ = \; & tr\Big(\tau(D_G)\tau(D_G)^T\Big) - 2tr\Big(\mathbf{a}^T X\tau(D_G)X^T\mathbf{a}\Big) + 1 \end{aligned}$$

Now, the minimization problem (4) can be written as follows:

$$\arg \max_{\substack{\mathbf{a} \\ \mathbf{a}^T XX^T\mathbf{a}=1}} \mathbf{a}^T X\tau(D_G)X^T\mathbf{a} \tag{5}$$

We will now switch to a Lagrangian formulation of the problem. The Lagrangian is as follows

$$\mathcal{L} = \mathbf{a}^T X\tau(D_G)X^T\mathbf{a} - \lambda\mathbf{a}^T XX^T\mathbf{a}$$

Requiring that the gradient of $\mathcal{L}$ vanish gives the following eigenvector problem:

$$X[\tau(D_G)]X^T\mathbf{a} = \lambda XX^T\mathbf{a} \tag{6}$$

It is easy to show that the matrices $X[\tau(D_G)]X^T$ and XX^T are both symmetric and positive semi-definite. The vectors $\mathbf{a}_i (i = 1, 2, \cdots, l)$ that minimize the objective function are given by the eigenvectors corresponding to the maximum eigenvalues of the generalized eigen-problem. Let $A = [\mathbf{a}_1, \cdots, \mathbf{a}_l]$, the linear embedding is as follows:

$$\mathbf{x} \to \mathbf{y} = W^T\mathbf{x}$$

$$W = W_{SVD}A$$

where $\mathbf{y}$ is a l-dimensional representation of the high dimensional data point $\mathbf{x}$. W is the transformation matrix.

4 Kernel IsoProjection

In this Section, we describe a method to conduct IsoProjection in the reproducing kernel Hilbert space into which the data points are mapped. This gives rise to Kernel IsoProjection.

Suppose $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m\} \in \mathcal{X}$ is the training sample set. We consider the problem in a feature space $\mathcal{F}$ induced by some nonlinear mapping

$$\phi : \mathcal{X} \to \mathcal{F}$$

For a proper chosen ϕ, an inner product $\langle , \rangle$ can be defined on $\mathcal{F}$ which makes for a so-called reproducing kernel Hilbert space (RKHS). More specifically,

$$\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle = \mathcal{K}(\mathbf{x}, \mathbf{y})$$

holds where $\mathcal{K}(.,.)$ is a positive semi-definite kernel function. Several popular kernel functions are: Gaussian kernel $\mathcal{K}(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|^2/\sigma^2)$; polynomial kernel $\mathcal{K}(\mathbf{x}, \mathbf{y}) = (1 + \langle \mathbf{x}, \mathbf{y} \rangle)^d$; Sigmoid kernel $\mathcal{K}(\mathbf{x}, \mathbf{y}) = tanh(\langle \mathbf{x}, \mathbf{y} \rangle + \alpha)$.

Given a set of vectors $\{\mathbf{v}_i \in \mathcal{F} | i = 1, 2, \cdots, d\}$ which are orthonormal ($\langle \mathbf{v}_i, \mathbf{v}_j \rangle = \delta_{i,j}$), the projection of $\phi(\mathbf{x}_i) \in \mathcal{F}$ to these $\mathbf{v}_1, \cdots, \mathbf{v}_d$ leads to a mapping from $\mathcal{X}$ to Euclidean space $\mathbb{R}^d$ through

$$\mathbf{y}_i = \left(\langle \mathbf{v}_1, \phi(\mathbf{x}_i) \rangle, \langle \mathbf{v}_2, \phi(\mathbf{x}_i) \rangle, \cdots, \langle \mathbf{v}_d, \phi(\mathbf{x}_i) \rangle \right)^T$$

We look for such $\{\mathbf{v}_i \in \mathcal{F} | i = 1, 2, \cdots, d\}$ that helps $\{\mathbf{y}_i | i = 1, \cdots, m\}$ preserve geodesic distances on the data manifold. A typical scenario is $\mathcal{X} = \mathbb{R}^n, \mathcal{F} = \mathbb{R}^\theta$ with $d << n < \theta$.

Let $\phi(X)$ denote the data matrix in RKHS:

$$\phi(X) = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \cdots, \phi(\mathbf{x}_m)]$$

Now, the eigenvector problem in RKHS can be written as follows:

$$\left(\phi(X)[\tau(D_G)]\phi^T(X) \right) \mathbf{v} = \lambda \left(\phi(X)\phi^T(X) \right) \mathbf{v} \tag{7}$$

Because the eigenvector of (7) are linear combinations of $\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \cdots, \phi(\mathbf{x}_m)$, there exist coefficients $\alpha_i, i = 1, 2, \cdots, m$ such that

$$\mathbf{v} = \sum_{i=1}^{m} \alpha_i \phi(\mathbf{x}_i) = \phi(X)\boldsymbol{\alpha}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_m)^T \in \mathbb{R}^m$.

Following some algebraic formulations, we get:

$$\begin{aligned}
& \left(\phi(X)[\tau(D_G)]\phi^T(X) \right) \mathbf{v} = \lambda \left(\phi(X)\phi^T(X) \right) \mathbf{v} \\
\Rightarrow \quad & \left(\phi(X)[\tau(D_G)]\phi^T(X) \right) \phi(X)\boldsymbol{\alpha} = \lambda \left(\phi(X)\phi^T(X) \right) \phi(X)\boldsymbol{\alpha} \\
\Rightarrow \quad & \phi^T(X) \left(\phi(X)[\tau(D_G)]\phi^T(X) \right) \phi(X)\boldsymbol{\alpha} \\
& = \lambda \phi^T(X) \left(\phi(X)\phi^T(X) \right) \phi(X)\boldsymbol{\alpha} \\
\Rightarrow \quad & K[\tau(D_G)]K\boldsymbol{\alpha} = \lambda K K \boldsymbol{\alpha}
\end{aligned} \tag{8}$$

where K is the kernel matrix, $K_{ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$. Let the column vectors $\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, \cdots, \boldsymbol{\alpha}^m$ be the solutions of equation (8). For a test point $\mathbf{x}$, we compute projections onto the eigenvectors $\mathbf{v}^k$ according to

$$(\mathbf{v}^k \cdot \phi(\mathbf{x})) = \sum_{i=1}^{m} \alpha_i^k (\phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i)) = \sum_{i=1}^{m} \alpha_i^k \mathcal{K}(\mathbf{x}, \mathbf{x}_i)$$

where α_i^k is the i^{th} element of the vector $\boldsymbol{\alpha}^k$. For the original training points, the map can be obtained by $\mathbf{y} = K\boldsymbol{\alpha}$, where the i^{th} element of $\mathbf{y}$ is the one-dimensional representation of $\mathbf{x}_i$.

In some situations, IsoProjection, kernel IsoProjection and Isomap [21] may give the same embedding results. We have the following proposition.

Proposition 2 *If X in equation (6) is a full rank square matrix, then IsoProjection and Isomap have the same embedding results on training points; and if K in equation (8) is positive definite, then kernel IsoProjection and Isomap have the same embedding results on training points.*

Proof Recall that the eigen-problem of IsoProjection is as follows:

$$X[\tau(D_G)]X^T\mathbf{w} = \lambda XX^T\mathbf{w}. \tag{9}$$

For the original training points, the embedding results can be obtained by $\mathbf{y} = X^T\mathbf{w}$, where the i^{th} element of $\mathbf{y}$ is the one-dimensional embedding of $\mathbf{x}_i$. Replace $X^T\mathbf{w}$ by $\mathbf{y}$, equation (9) can be rewritten as follows:

$$X[\tau(D_G)]\mathbf{y} = \lambda X\mathbf{y} \tag{10}$$

Since X is a full rank square matrix, the inverse of X always exists. Thus, the above equation can be changed to

$$X^{-1}X[\tau(D_G)]\mathbf{y} = \lambda X^{-1}X\mathbf{y}. \tag{11}$$

Finally, we get

$$[\tau(D_G)]\mathbf{y} = \lambda\mathbf{y} \tag{12}$$

which is just the eigen-problem of Isomap.

In kernel IsoProjection, the map of the training points can be obtained by $\mathbf{y} = K\boldsymbol{\alpha}$, where the i^{th} element of $\mathbf{y}$ is the one-dimensional embedding of $\mathbf{x}_i$. Replace $K\boldsymbol{\alpha}$ by $\mathbf{y}$, equation (8) can be rewritten as:

$$K[\tau(D_G)]\mathbf{y} = \lambda K\mathbf{y}. \tag{13}$$

Similarly, if K is positive definite, the above equation can be reduced to

$$[\tau(D_G)]\mathbf{y} = \lambda\mathbf{y} \tag{14}$$

which again is the eigen-problem of Isomap. ∎

This proposition illustrates three interesting points:

1. When the number of features (m) is larger than the number of samples (n), X will be a full rank square matrix after SVD transformation if all the data vectors are linearly independent. In this case, IsoProjection provides the same embedding result on training points as Isomap. However, IsoProjection has the projection functions which can be applied to testing data. In many real world applications such as information retrieval, the dimensionality of the document space is typically much larger than the number of documents. It falls into this case if these document vectors are linearly independent.

2. Kernel IsoProjection with a positive definite kernel matrix yields the same results as Isomap on the training points. Moreover, Kernel IsoProjection is defined everywhere while Isomap is only defined on the training samples. In reality, when the number of samples is much larger than the number of features (such as data in Figure 2), kernel IsoProjection might have more power than IsoProjection to discover the nonlinear manifold structure.

3. Based on (1) and (2), a general guideline for choosing IsoProjection or kernel IsoProjection could be: when the number of features (m) is larger than the number of samples (n), IsoProjection is preferred; otherwise, kernel IsoProjection is preferred.

5 Experimental Results

5.1 A Toy Problem

We first take the synthetic "Swiss roll" data to examine our algorithm. The 1000 data points are sampled from a 2-dimensional manifold which is embedded in 3-dimensional ambient space (Figure 2(a)). Since the number of data points ($m = 1000$) is much larger than the number of features ($n = 3$), we use kernel IsoProjection with Gaussian kernel. The kernel matrix is positive definite thus the embedding result (Figure 2(c)) of training data (1000 points) is the same as that in Isomap [21]. However, kernel IsoProjection provides a mapping function that we can use to project new testing data (Figure 2(d)(e)).

Kernel IsoProjection correctly recover the intrinsic dimensionality and geometric structure of the data. The Euclidean distance in the embedding space (Figure 2(c)) can accurately approximate the geodesic distance on the manifold. For comparison, we also demonstrate the embedding result of kernel PCA [19] on the same data, as shown in Figure 2(f). Clearly, kernel PCA failed to illustrate the low-dimensional manifold structure.

5.2 Experiments on Clustering

In this subsection, we investigate the use of dimensionality reduction algorithms for document clustering. Latent Semantic Indexing (LSI) [5] is the most popular dimensionality reduction algorithm for document analysis. LSI is essentially equivalent to PCA provided that the data points have a zero mean. In this experiment, we compared our IsoProjection with LSI.

5.2.1 Data Corpora

Reuters-21578 corpus[2], which contains 21578 documents in 135 categories, was used in our experiments. In our experiments, we discarded those documents with multiple category labels, and selected the largest 30 categories. It left us with 8,067 documents as described in Table 1. Each document is represented as a term-frequency vector and each document vector is normalized to 1. We simply removed the stop words, and no further preprocessing was done.

5.2.2 2-D Visualization of Document Set

As we described previously, LSI and IsoProjection are different dimensionality reduction algorithms. In this subsection, we use them to project the documents into a 2-dimensional subspace for visualization. We randomly selected four classes for this test. Figure 3 shows the 2D embedding results. As can be seen, LSI fails to distinguish the different classes, and the four classes are mixed together. The four classes can be easily separated in IsoProjection embedding. This illustrative example shows that IsoProjection can have more discriminating power than LSI.

5.2.3 Evaluation Metric of Clustering

We chose K-means as our clustering algorithm and compared three methods. These three methods are listed below:

- K-means on original term-document matrix (Baseline)
- K-means after LSI (LSI)
- K-means after IsoProjection (IsoP)

In IsoProjection, the parameter k (number of nearest neighbors) was set to 15.

We tested these algorithms on several cases. For each case, $K(= 2 \sim 10)$ classes were randomly selected from the document corpus. The documents and the cluster number K are provided to the clustering algorithms. The clustering result is evaluated by comparing the obtained label of each document with that provided by the document corpus. Two metrics, the accuracy (AC) and the normalized mutual information metric ($\overline{MI}$) are used to measure the clustering performance [3]. Given a document $\mathbf{x}_i$, let r_i and s_i be the obtained cluster label and the label provided by the corpus, respectively. The AC is defined as follows:

$$AC = \frac{\sum_{i=1}^{n} \delta(s_i, map(r_i))}{n}$$

where n is the total number of documents and $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and map(r_i) is the permutation mapping function that maps each cluster label r_i

[2]Reuters-21578 corpus is at `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

to the equivalent label from the data corpus. The best mapping can be found by using the Kuhn-Munkres algorithm [11].

Let C denote the set of clusters obtained from the ground truth and C' obtained from our algorithm. Their mutual information metric $MI(C, C')$ is defined as follows:

$$MI(C, C') = \sum_{c_i \in C, c'_j \in C'} p(c_i, c'_j) \cdot log_2 \frac{p(c_i, c'_j)}{p(c_i) \cdot p(c'_j)}$$

where $p(c_i)$ and $p(c'_j)$ are the probabilities that a document arbitrarily selected from the corpus belongs to the clusters c_i and c'_j, respectively, and $p(c_i, c'_j)$ is the joint probability that the arbitrarily selected document belongs to the clusters c_i as well as c'_j at the same time. In our experiments, we use the normalized mutual information $\overline{MI}$ as follows:

$$\overline{MI}(C, C') = \frac{MI(C, C')}{\max(H(C), H(C'))}$$

where $H(C)$ and $H(C')$ are the entropies of C and C', respectively. It is easy to check that $\overline{MI}(C, C')$ ranges from 0 to 1. $\overline{MI} = 1$ if the two sets of clusters are identical, and $\overline{MI} = 0$ if the two sets are independent.

5.2.4 Results

The evaluations were conducted with different numbers of clusters. For each given class number K, K classes were randomly selected from the database. This process were repeated 50 times, and the average performance was computed. For each single test (given K classes of documents), we applied the above three methods. For each method, the K-means step was repeated 10 times with different initializations and the best result in terms of the objective function of K-means was recorded. For IsoProjection and LSI, they both need to estimate the dimensionality of the subspace. In general, their performance varies with the dimensionality of the subspace. Figure 4 show the clustering performance of these algorithms as a function of the dimensionality of the subspace. Table 2 shows the best performance obtained by each algorithm. The *paired T-test* on the 50 random tests are reported in Table 3.

As can be seen, our clustering algorithm consistently outperformed LSI and baseline. LSI learned a compact representation for documents, however, there is no significant performance improvement over baseline. This shows that LSI fails to discover the intrinsic class structure of the document corpus.

5.3 Experiments on Classification

In this subsection, we investigate the performance of our proposed IsoProjection algorithm for classification task, particularly, face recognition. In classification, the label information of training data is available which can be incorporate into the graph construction of our algorithm. The most well known supervised dimensionality reduction method is Linear Discriminate Analysis (LDA)[6]. Both PCA [22]

and LDA [1] are popular linear methods for subspace learning in face recognition. Thus, our algorithm is compared with these two algorithms.

5.3.1 Dataset and Experimental Design

In this study, we use the Yale face database[3]. The Yale face database was constructed at the Yale Center for Computational Vision and Control. It contains 165 gray scale images of 15 individuals. The images demonstrate variations in lighting condition, facial expression (normal, happy, sad, sleepy, surprised, and wink). Figure 6 shows the 11 images of one individual in Yale data base.

In the experiments, preprocessing to locate the faces was applied. Original images were manually aligned (two eyes were aligned at the same position), cropped, and then re-sized to 32×32 pixels, with 256 gray levels per pixel. Each image is represented by a $1,024$-dimensional vector in image space. Different pattern classifiers have been applied for face recognition, such as nearest-neighbor [1], Bayesian [13], Support Vector Machine [17]. In this paper, we apply the nearest-neighbor classifier for its simplicity. The Euclidean metric is used as our distance measure.

In short, the recognition process has three steps. First, we calculate the face subspace from the training samples; then the new face image to be identified is projected into d-dimensional subspace by using our algorithm; finally, the new face image is identified by a nearest neighbor classifier.

5.3.2 Results

A random subset with $l(= 2, 3, 4, 5, 6, 7, 8)$ images per individual was taken with labels to form the training set, and the rest of the database was considered to be the testing set. For each given l, we average the results over 50 random splits. Note that, for LDA, there are at most $c-1$ nonzero generalized eigenvalues and, so, an upper bound on the dimension of the reduced space is $c-1$, where c is the number of individuals [1]. The graph in IsoProjection is built based on the label information.

In general, the performance of all these methods varies with the number of dimensions. We show the best results and the optimal dimensionality obtained by PCA, LDA, IsoProjection and baseline methods in Table 4. The *paired T-test* on the 50 random splits are reported in Table 5. For the baseline method, the recognition is simply performed in the original 1024-dimensional image space without any dimensionality reduction.

As can be seen, our algorithm performed the best in all the cases. There is no improvement over baseline for PCA method. The performance of LDA is very sensitive to the training size. When the training size is small, LDA can be even worse than PCA. As the training sample increases, LDA achieves similar performance to IsoProjection.

[3] http://cvc.yale.edu/projects/yalefaces/yalefaces.html

6 Concluding Remarks and Future Work

In this paper, we propose a new linear dimensionality reduction algorithm called Isometric Projection. It can be performed in either original space or reproducing kernel Hilbert space, which leads to Kernel Isometric Projection. Both IsoProjection and kernel IsoProjection are based on the same variational principle that gives rise to the Isomap [21]. As a result they are capable of discovering the nonlinear degree of freedom that underlie complex natural observations. Our approaches has a major advantage over recent nonparametric techniques for global nonlinear dimensionality reduction such as [18][21][2] that the functional mapping between the high and low dimensional spaces are valid both on and off the training data. Performance improvement of this method over Principal Component Analysis and Linear Discriminant Analysis is demonstrated through several experiments.

There are several interesting problems that we are going to explore in the future work:

1. In this paper, the geodesic distance of two points is approximated by the length of the shortest path on the nearest neighbor graph. It is unclear if there are more efficient and better ways to do it.

2. In most of previous algorithms on manifold learning, either an eigen-problem or a generalized eigen-problem need to be solved. Thus, all these algorithms will be failed to handle extremely large data set. The tradeoff between effectiveness and efficiency might be needed in such case. It is interesting to develop a flexible algorithm which can make the tradeoff under different situations.

References

[1] P. N. Belhumeur, J. P. Hepanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[2] 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, Cambridge, MA, 2001.

[3] D. Cai, X. He, and J. Han. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1624–1637, December 2005.

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

[5] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[6] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2000.

[7] V. Gaede and O. Günther. Multidimensional access methods. *ACM Comput. Surv.*, 30(2):170–231, 1998.

[8] A. Grama, G. Karypis, V. Kumar, and A. Gupta. *An Introduction to Parallel Computing: Design and Analysis of Algorithms*. Addison Wesley, 2nd edition, 2003.

[9] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[10] J. M. Lee. *Introduction to Smooth Manifolds*. Springer-Verlag New York, 2002.

[11] L. Lovasz and M. Plummer. *Matching Theory*. Akadémiai Kiadó, North Holland, Budapest, 1986.

[12] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, 1980.

[13] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710, 1997.

[14] B. Moghaddam, Q. Tian, N. Lesh, and T. S. H. C. Shen. Visualization and user-modeling for browsing personal photo libraries. *International Journal of Computer Vision*, 56, 2004.

[15] A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, Cambridge, MA, 2001.

[16] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 697–708, Trondheim, Norway, 2005.

[17] P. J. Phillips. Support vector machines applied to face recognition. *Advances in Neural Information Processing Systems*, 11:803–809, 1998.

[18] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[19] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, (10):1299–1319, 1998.

[20] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[21] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[22] M. Turk and A. P. Pentland. Face recognition using eigenfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, 1991.

The IsoProjection Algorithm

Input:	Data matrix $X = [\mathbf{x}_1, \cdots, \mathbf{x}_m]$, $\mathbf{x}_i \in \mathbb{R}^n$ ϵ for ϵ-graph or k for kNN-graph
Output:	Transformation matrix $W = [\mathbf{w}_1, \cdots, \mathbf{w}_l]$, $\mathbf{w}_j \in \mathbb{R}^n$
Step 1:	**SVD projection**: $X = U\Sigma V$, $\Sigma = diag(\sigma_1, \cdots, \sigma_r)$ $\sigma_1 \geq \cdots \geq \sigma_r > 0$ and $r \leq m$ $X \leftarrow U^T X$
Step 2:	**Construct neighborhood graph**: Define graph G over all data points by connecting points i ad j if $\|\mathbf{x}_i - \mathbf{x}_j\| < \epsilon$, or if i and j are among the k nearest neighbors of each other. Set edge lengths equal to $\|\mathbf{x}_i - \mathbf{x}_j\|$.
Step 3:	**Compute shortest paths**: Calculate the shortest path distances $d_G(i,j)$ between all pairs of points in G, let $D_G = \{d_G(i,j)\}$.
Step 4:	**Isometric projection** : Define S, $\{S_{ij} = {D_G}^2_{ij}\}$; $H = I - \frac{1}{m}\mathbf{e}\mathbf{e}^T$. Define $\tau(D_G) = -\frac{1}{2}HSH$. Solve the generalized eigen-problem: $X[\tau(D_G)]X^T\mathbf{a} = \lambda XX^T\mathbf{a}$ Suppose $\mathbf{a}_1, \cdots, \mathbf{a}_l$ are the eigenvectors corresponding to the largest l eigenvalues. Let $A = [\mathbf{a}_1, \cdots, \mathbf{a}_l]$. $W = UA$.

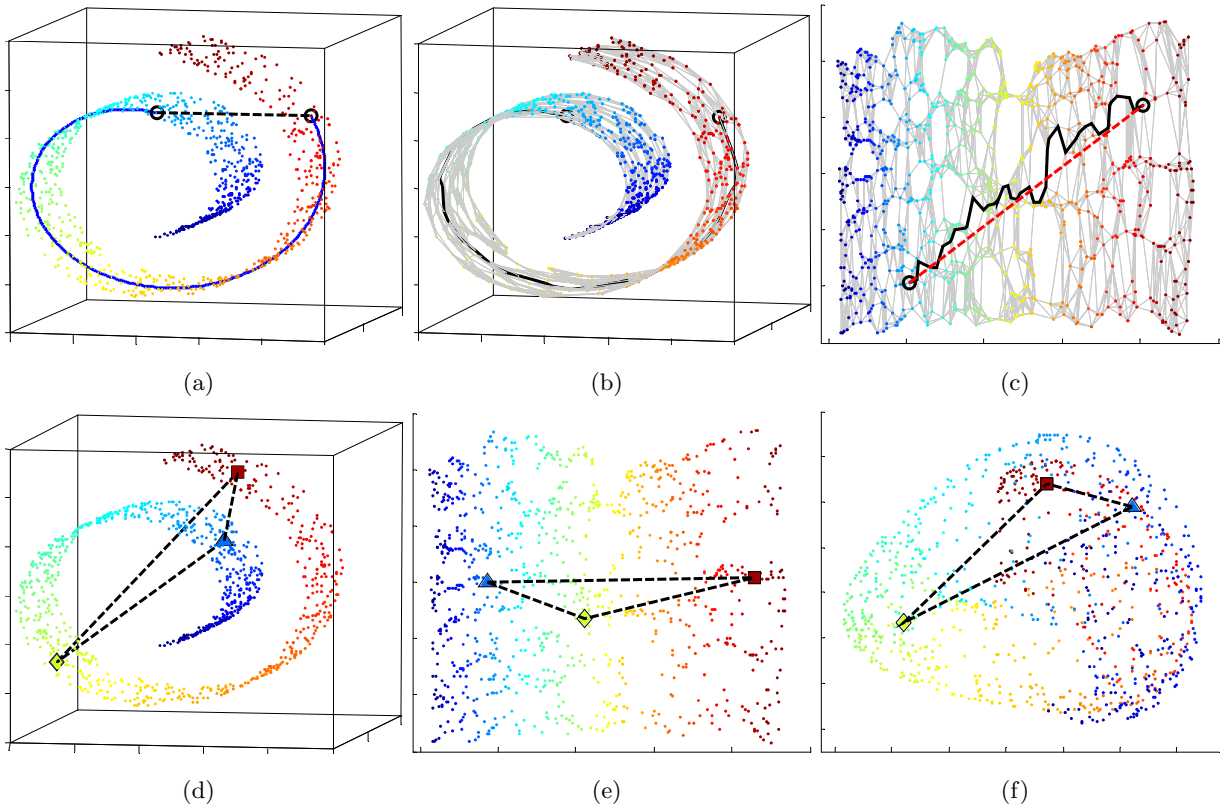
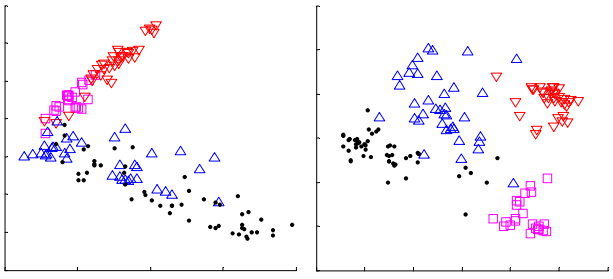


Figure 2: The "Swiss roll" data set, illustrating how Kernel IsoProjection exploits geodesic distance for nonlinear dimensionality reduction as well as provides a projection function which is defined everywhere. (a) For two arbitrary points (circled) on a nonlinear manifold, their Euclidean distance in the high-dimensional input space (length of dashed line) may not accurately reflect their intrinsic similarity, as measured by geodesic distance along the low-dimensional manifold (length of solid blue curve). (b) The neighborhood graph G constructed in IsoProjection (with k=7) allows an approximation (black segments) to the true geodesic path to be efficiently computed as the shortest path in G. (c) The two-dimensional embedding of Kernel IsoProjection best preserves the shortest path distances in the neighborhood graph. The straight dashed line (red) in the lower dimensional Euclidean space is a good approximation to the geodesic on the data manifold (d) Three new points (test points) are injected into the system. Similarly, their Euclidean distances in the high-dimensional input space (length of dashed lines) can not accurately reflect their intrinsic similarity. (e) Using the mapping function learned by Kernel IsoProjection, we map these three test points into the two-dimensional space where the Euclidean distances can accurately reflect their intrinsic relationship. (f) The embedding results of Kernel PCA on the same data set. Clearly, Kernel PCA failed to capture the low-dimensional manifold structure.

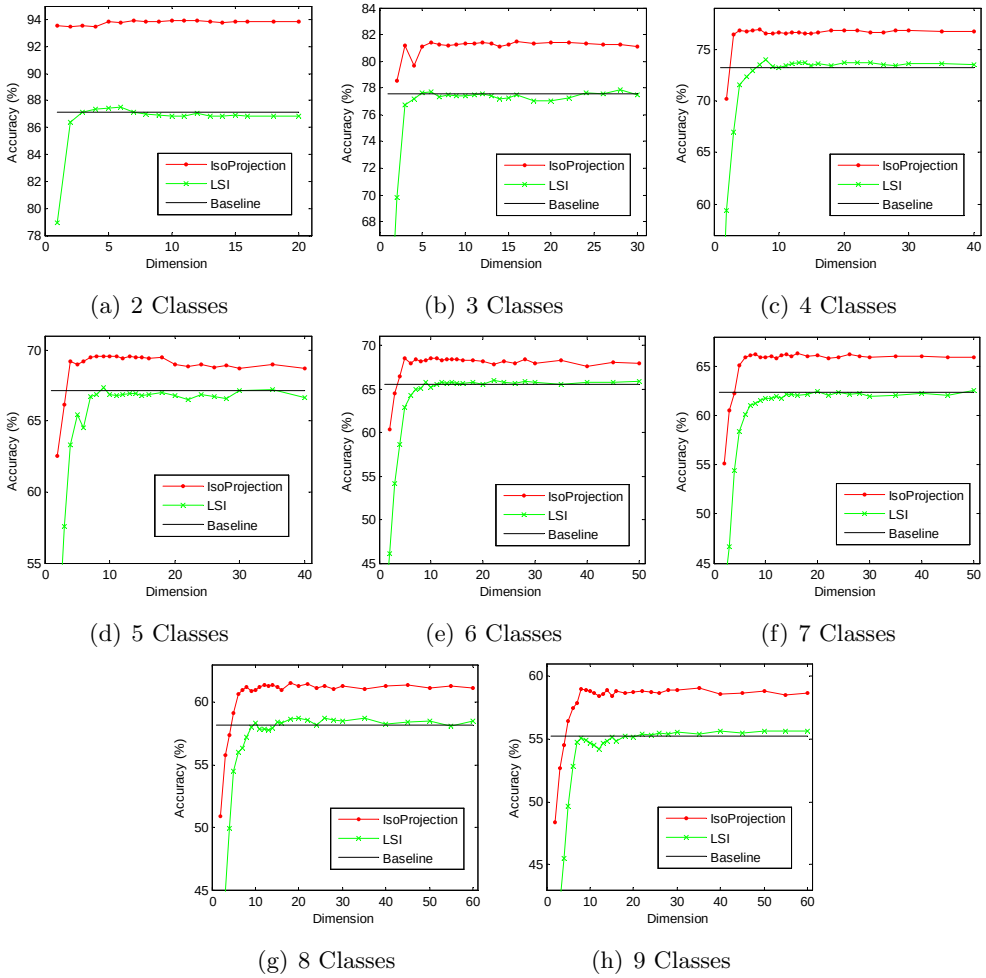
Table 1: 30 semantic categories from Reuters-21578 used in our experiments

category	**num of doc**	**category**	**num of doc**
earn	3713	grain	45
acq	2055	copper	44
crude	321	jobs	42
trade	298	reserves	38
money-fx	245	rubber	38
interest	197	iron-steel	37
ship	142	ipi	36
sugar	114	nat-gas	33
coffee	110	veg-oil	30
gold	90	tin	27
money-supply	87	cotton	24
gnp	63	bop	23
cpi	60	wpi	20
cocoa	53	pet-chem	19
alum	45	livestock	18



(a) LSI (b) IsoProjection

Figure 3: 2D visualization of a document set



(a) 2 Classes (b) 3 Classes (c) 4 Classes

(d) 5 Classes (e) 6 Classes (f) 7 Classes

(g) 8 Classes (h) 9 Classes

Figure 4: The average accuracy over different number of classes. The clustering performance was evaluated at different dimensionality. As can be seen, the clustering performance of both IsoProjection and LSI are not sensitive to the reduced dimensionality. Clustering performances after IsoProjection in all cases are consistently better than baseline, while clustering after LSI does not show any significant improvement over baseline.

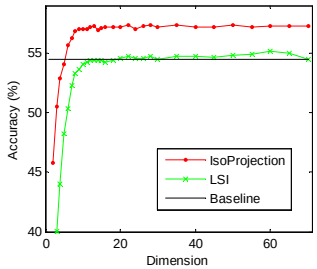


Figure 5: The average accuracy on 10 classes

Table 2: Clustering Results on Reuters-21578

	Accuracy (%)			Mutual Information (%)		
k	Baseline	LSI	IsoP	Baseline	LSI	IsoP
2	87.13	87.50	93.91	59.98	60.75	73.64
3	77.53	77.83	81.50	56.68	56.98	61.60
4	73.23	74.01	76.89	59.80	60.35	62.38
5	67.11	67.34	69.55	56.31	56.22	58.18
6	65.48	65.95	68.54	57.90	58.09	59.63
7	62.31	62.53	66.28	57.27	57.40	60.20
8	58.19	58.75	61.55	55.61	55.89	57.41
9	55.26	55.62	59.02	54.88	55.20	56.87
10	54.50	55.15	57.39	55.16	55.45	56.81
Ave.	66.75	67.18	70.51	57.07	57.37	60.75

Table 3: T-test on clustering

	LSI vs. Baseline		IsoP vs. LSI	
k	Accuracy	Mutual Info.	Accuracy	Mutual Info.
2	$\sim$	$\sim$	$\gg$	$\gg$
3	$\sim$	$\sim$	$\gg$	$\gg$
4	$\sim$	$\sim$	$\gg$	$\gg$
5	$\sim$	$\sim$	$\gg$	$\gg$
6	$\sim$	$\sim$	$\gg$	$\gg$
7	$\sim$	$\sim$	$\gg$	$\gg$
8	$\sim$	$\sim$	$\gg$	$\gg$
9	$\sim$	$\sim$	$\gg$	$\gg$
10	$\sim$	$\sim$	$\gg$	$\gg$

"$\gg$" or "$\ll$" means P-value ≤ 0.01

"$>$" or "$<$" means $0.01 <$ P-value ≤ 0.05

"$\sim$" means P-value > 0.05

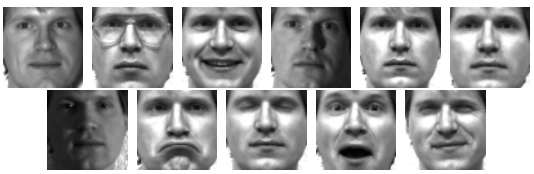


Figure 6: Sample face images from the Yale database. For each subject, there are 11 face images under different lighting conditions with facial expression.

Table 4: Recognition accuracy on the Yale database

Train Num	Baseline	PCA	LDA	IsoProjection
2	0.46	0.46 (29)	0.44 (9)	**0.56** (14)
3	0.52	0.52 (44)	0.61 (14)	**0.67** (14)
4	0.55	0.55 (59)	0.69 (14)	**0.73** (14)
5	0.58	0.58 (74)	0.74 (14)	**0.77** (14)
6	0.61	0.61 (89)	0.77 (14)	**0.79** (14)
7	0.62	0.62 (36)	0.80 (14)	**0.81** (14)
8	0.65	0.65 (116)	0.81 (14)	**0.82** (14)

Table 5: T-test on classification

Train Num	LDA vs. Baseline	IsoProjection vs. LDA
2	$\ll$	$\gg$
3	$\gg$	$\gg$
4	$\gg$	$\gg$
5	$\gg$	$\gg$
6	$\gg$	$\gg$
7	$\gg$	$\sim$
8	$\gg$	$\sim$