Multilevel Manifold Learning with Application to Spectral Clustering[*]

Haw-ren Fang
Dept. of Computer Science & Engineering
University of Minnesota
Minneapolis, MN 55455, USA
hrfang@cs.umn.edu

Sophia Sakellaridi
Dept. of Computer Science & Engineering
University of Minnesota
Minneapolis, MN 55455, USA
sakell@cs.umn.edu

Yousef Saad
Dept. of Computer Science & Engineering
University of Minnesota
Minneapolis, MN 55455, USA
saad@cs.umn.edu

ABSTRACT

In the past decade, a number of nonlinear dimensionality reduction methods using an affinity graph have been developed for manifold learning. This paper explores a multilevel framework with the goal of reducing the cost of unsupervised manifold learning and preserving the embedding quality at the same time. An application to spectral clustering is also presented. Experimental results indicate that our multilevel approach is an appealing alternative to standard techniques.

Categories and Subject Descriptors

I.5.3 [**Pattern Recognition**]: Clustering; G.2.2 [**Graph theory**]: Graph algorithms; F.2.1 [**Numerical Algorithms and Problems**]: Computations on matrices

General Terms

Algorithms, Experimentation

Keywords

Multilevel Methods, Manifold Learning, Spectral Clustering

1. INTRODUCTION

Real world high dimensional data can often be represented as points or vectors in a much lower dimensional nonlinear manifold. Examples include face databases, continuous video images, digital voices, microarray gene expression data, and financial time series. The observed dimensions is the size of the number of pixels per image, or generally the number of numerical values per data item, and can be characterized by far fewer features.

In the past decade, a number of algorithms have been developed to 'learn' the low dimensional manifold of high dimensional data sets. Given a set of high dimensional data represented by vectors $x_1, \ldots, x_n$ in $\mathbb{R}^m$, the task is to represent these with low dimensional vectors $y_1, \ldots, y_n \in \mathbb{R}^d$ with $d \ll m$, such that nearby points remain nearby, and distant points remain distant.

Multilevel techniques, which aim at reducing the problem size and improving computational efficiency, have been successfully applied to various scientific problems, such as graph and hypergraph partitioning, e.g., [12, 13]. On the other hand, their incorporation into dimensionality reduction methods is currently under-explored. Inspired by their success in other applications, we presented a graph-based multilevel scheme for linear dimensionality reduction [19]. Here we expand this work to a multilevel framework for *nonlinear dimensionality reduction*. The framework of these methods relies on an affinity graph and so it can be especially useful for affinity-graph-based manifold learning methods.

The multilevel framework proposed in this paper consists of three phases: data coarsening, nonlinear dimension reduction, and data refining. To coarsen the data, we employ a graph coarsening algorithm based on maximum independent sets. After this, we project the coarsened data at the coarsest level using one of several known manifold learning methods. Finally, we recursively refine the data level by level, by solving a linear system to go from a given lower level to a higher level. The linear system comes from a least squares optimization which aims to preserve the closeness of data points between two adjacent levels.

Landmark versions of manifold learning algorithms by random sampling have been proposed to reduce the problem size and therefore the computational cost, e.g., Isomap [8], maximum variance unfolding [30], Locally Linear Embedding and Laplacian Eigenmaps [3]. These methods can be seen as learning eigenfunctions of some kernel. When the cost of a manifold learning algorithm is dominated by the spectral factorization of a symmetric matrix, an alternative way to reduce cost is via low-rank matrix approximation techniques [26].

The method proposed in this paper has three distinct properties from the landmark approach. First, maximum independent sets contain repelled points that potentially provide a better basis of the original data than random samples. Second, by recursive coarsening we obtain a succession of graphs on which our refining scheme is based and this phase is independent of the dimensionality reduc-

[*] This work was supported by NSF grants DMS-0810938 and DMR-0940218 and by the Minnesota Supercomputing Institute.

CIKM'10, October 26–29, 2010, Toronto, Ontario, Canada.
Copyright 2010 ACM 978-1-4503-0099-5/10/10 ...$10.00.

tion method. Third, the multilevel structure propagates the geodesic information into the coarsened graphs and this may be beneficial to some manifold learning algorithms.

Here we give an example showing that bad landmarks may result in an unsatisfactory embedding which can be prevented by our multilevel approach. Figure 1(a) presents a set of 1,000 points sampled on a `Swissroll` in three-dimensional space, with the embedding by Isomap given in Figure 1(e). In all we use a kNN graph with $k = 12$. The results by our multilevel-Isomap are in Figures 1(b) and 1(f) down to $r = 2, 3$ levels, respectively, where n_r is the number of sample points at the bottom level. The results by Landmark-Isomap are in Figures 1(c) and 1(g), where the landmarks are uniformly distributed which can be expected but not guaranteed by random sampling. The number of landmarks is denoted by n_s. All these methods unfolded the `Swissroll` in a reasonable way. However, as shown in Figures 1(d) and 1(h), if the landmarks cluster in the left end, then the embedded points in the right end scatter. Such worst case scenarios can be prevented by our multilevel method. The details will be given in Section 3.

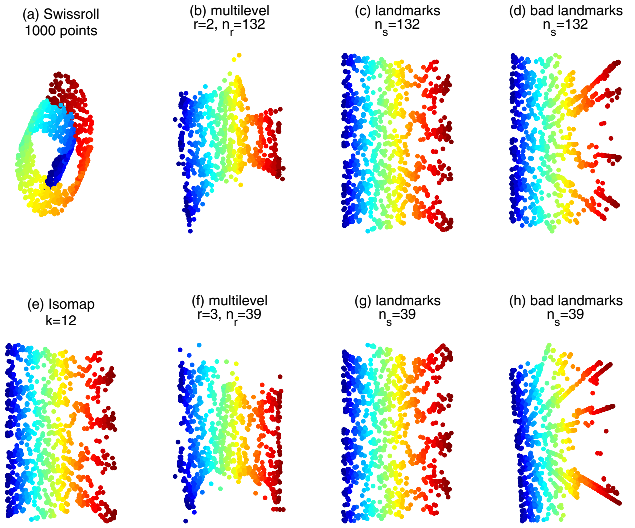

Figure 1: 2D mappings of `Swissroll` data set.

In this paper we consider three manifold learning algorithms: Isomap [27], Locally Linear Embedding (LLE) [18, 21], and Laplacian Eigenmaps [2], which are representative in manifold learning [22]. Note that our multilevel framework is not limited to these methods. It can be applied to virtually all affinity-graph-based manifold learning methods, such as maximum variance unfolding [31], Hessian LLE [10], conformal Isomap [8], diffusion maps [7, 14], conformal eigenmaps [23], and minimum volume embedding [24].

Spectral clustering methods, e.g., [17], perform nonlinear dimensionality reduction on the input data, and apply a vector modeled clustering algorithm to the mapped data in the spectral space for classification. This type of methods is closely related to manifold learning. We will also show the application of our multilevel nonlinear dimensionality reduction technique to spectral clustering.

2. MANIFOLD LEARNING

We say that a given open set $\Psi \in \mathbb{R}^m$ in m-dimensional Euclidean space resides in a lower d-dimensional manifold (typically $d \ll m$), if there is a continuously differentiable function $g : \Omega \to \mathbb{R}^m$ on an open domain $\Omega \in \mathbb{R}^d$, such that $g(\Omega) = \Psi$. The parameterized manifold $\Psi = g(\Omega)$ is called *regular*, if the Jacobian matrix $J(y)$ of $g(y)$ has full rank for all $y \in \Omega$, and $g(y)$ does not self-intersect; i.e., $y_i \neq y_j$ implies $g(y_i) \neq g(y_j)$. A regular manifold mapping $g : \Omega \to \Psi$ has an inverse function $f : \Psi \to \Omega$. Manifold learning methods attempt to find a function f that maps points in $\Psi \in \mathbb{R}^m$ into points of a lower dimension $\mathbb{R}^d$. In practice, we often have a discrete and possibly noisy sampled data $x_1, \ldots, x_n \in \mathbb{R}^m$ of Ψ, and the objective is to find the corresponding low dimensional embedding $y_1, \ldots, y_n \in \mathbb{R}^d$. The goal of the mapping is to preserve the closeness of nearby points, for which an affinity graph $G = (V, E)$, normally a kNN graph, is employed.

In this paper, we use matrices $X = [x_1, \ldots, x_n] \in \mathbb{R}^{m \times n}$ and $Y = [y_1, \ldots, y_n] \in \mathbb{R}^{d \times n}$ $(d < n)$ to denote the original high dimensional data and the mapped low dimensional data, respectively. The column vector of ones is denoted by e. We also use integers $1, \ldots, n$ to denote the vertices of the affinity graph $G = (V, E)$, i.e., $V = \{1, \ldots, n\}$.

2.1 Isomap

Isomap [27] is a nonlinear generalization of the linear multidimensional scaling (MDS). It replaces the Euclidean distances in MDS by the *geodesic* distances approximated by an affinity graph $G = (V, E)$, whose vertices $1, \ldots, n$ in V correspond to the input data $x_1, \ldots, x_n \in \mathbb{R}^m$, and edges in E define the closeness of them. The length of the shortest path between vertices x_i and x_j, denoted by $\tilde{d}_{ij}$, is the approximate geodesic distance between them.

The algorithm can be summarized as follows. It starts by constructing an affinity graph, typically a kNN graph for the data. With this, the all-pair shortest path problem is solved and all the squared approximate geodesic distances $\tilde{d}_{ij}^2$ are saved in a symmetric matrix $\widetilde{D} \in \mathbb{R}^{n \times n}$. The next step is to compute the Grammian matrix $\widetilde{B} = -\frac{1}{2} J \widetilde{D} J \in \mathbb{R}^{n \times n}$, where $J = I - \frac{1}{n} ee^T \in \mathbb{R}^{n \times n}$ with $I \in \mathbb{R}^{n \times n}$ the identity matrix and $e \in \mathbb{R}^n$ a column vector of ones. Then Isomap maps $X = [x_1, \ldots, x_n] \in \mathbb{R}^{m \times n}$ nonlinearly to $Y = [y_1, \ldots, y_n] \in \mathbb{R}^{d \times n}$ by minimizing $\|\widetilde{B} - Y^T Y\|_F$. To be precise, denote by $\lambda_i \in \mathbb{R}$ and $v_i \in \mathbb{R}^n$ the ith eigenvalue and eigenvector of $\widetilde{B}$ in decreasing order. Let $\Sigma_d \in \mathbb{R}^{d \times d}$ be the diagonal matrix formed by $\lambda_1, \ldots, \lambda_d$, and the columns of $V_d \in \mathbb{R}^{n \times d}$ be $v_1, \ldots, v_d$. The mapped data is $Y = \Sigma_d^{1/2} V_d^T \in \mathbb{R}^{d \times n}$.

2.2 Locally Linear Embedding

Locally linear embedding (LLE) [18, 21] maps the high dimensional input data $x_1, \ldots, x_n \in \mathbb{R}^m$ to $y_1, \ldots, y_n \in \mathbb{R}^d$ in a lower dimensional space (i.e., $d < n$) by three steps.

First, a kNN graph is constructed. Second, the reconstruction weights $W = [w_{ij}] \in \mathbb{R}^{n \times n}$ are obtained by minimizing the cost function:

$$\mathcal{E}(W) = \sum_{i=1}^{n} \|x_i - \sum_{j=1}^{n} w_{ij} x_j\|_2^2, \tag{1}$$

subject to that $w_{ij} = 0$ if x_j is not one of k nearest neighbors of x_i, and $\sum_{j=1}^{n} w_{ij} = 1$ for $i = 1, \ldots, n$. Minimizing $\|x_i - \sum_{j=1}^{n} w_{ij} x_j\|_2^2$ in (1) requires solving a constrained least squares problem for each $i = 1, \ldots, n$.

Finally, the high dimensional data $X = [x_1, \ldots, x_n] \in \mathbb{R}^{m\times n}$ is mapped to the low dimensional data $Y = [y_1, \ldots, y_n] \in \mathbb{R}^{d\times n}$ by minimizing the embedding cost function:

$$\begin{aligned}\Phi(Y) &= \sum_{i=1}^{n} \|y_i - \sum_{j=1}^{n} w_{ij} y_j\|_2^2 = \|Y - YW^T\|_F^2 \\ &= \operatorname{trace}[Y(I-W)^T(I-W)Y^T]. \qquad (2)\end{aligned}$$

Two constraints are added for the problem to be well-posed. First, it is required that the projected data be centered, i.e., $\sum_{i=1}^{n} y_i = 0$. Second, the mapped data, subject to scaling, must have unit covariance, i.e., $\sum_{i=1}^{n} y_i y_i^T = YY^T = I$.

Let $M = (I-W)^T(I-W)$. Then $Me = 0$, where e is the column vector of ones. Therefore, e is a eigenvector of M associated with the smallest eigenvector 0. Other eigenvectors v satisfy $v^T e = 0$. The embedding is formed by the d right singular vectors of $I - W$ associated with the second to the $(d+1)$st smallest singular values.

2.3 Laplacian Eigenmaps

In Laplacian eigenmaps, a kNN graph of $X = [x_1, \ldots, x_n] \in \mathbb{R}^{m\times n}$ is also constructed. The low dimensional embedding $Y = [y_1, \ldots, y_n] \in \mathbb{R}^{d\times n}$ is the minimizer of the cost function:

$$\Psi(Y) = \sum_{i,j} w_{ij}\|y_i - y_j\|_2^2 = 2\operatorname{trace}(Y(D-W)Y^T), \quad (3)$$

where $W = [w_{ij}]$ is a symmetric weight matrix, D is a diagonal matrix with $d_{ii} = \sum_{j=1}^{n} w_{ij}$.

A popular weighting scheme is the Gaussian weights:

$$w_{ij} = \exp(-\|x_i - x_j\|_2^2/t) \quad (4)$$

for each pair of neighboring points x_i, x_j, where $t > 0$ is a preset parameter. This weighting scheme is also called *heat kernel* in [2]. In our experiments we set t equal to the median of $\|x_i - x_j\|_2^2$ of all neighboring points x_i, x_j. Setting $\sigma = \infty$ in (4), we obtain the induced binary weighting method.

To make the minimization of (3) well-posed, the constraints $YDY^T = I$ and $YDe = 0$ are imposed, where e is the column vector of ones. The problem is transformed to solving the generalized eigenvalue problem $(D-W)z = \lambda Dz$, whose d generalized eigenvectors corresponding to the second to the $(d+1)$st eigenvalues form Y. The bottom generalized eigenvector e associated with eigenvalue 0 is ignored.

3. MULTILEVEL NONLINEAR DIMENSIONALITY REDUCTION

This section presents our multilevel framework for nonlinear dimensionality reduction for manifold learning. This approach consists of three phases: data coarsening, nonlinear dimension reduction, and data refining. Figure 2 provides an illustration. In a nutshell, a few levels of coarsening are performed leading to a sequence of smaller and smaller graphs. The analysis of the data is done at the coarsest level using a standard dimension reduction technique such as Isomap, LLE, or Laplacian eigenmaps. Then an 'uncoarsening' step of this low dimensional data is performed backing up to the highest level. Details are provided next.

3.1 The Coarsening Phase

Coarsening a graph $G = (V,E)$ means finding a 'coarse' approximation $\widehat{G} = (\widehat{V}, \widehat{E})$ that represents $G = (V,E)$, where $|\widehat{V}| < |V|$. By recursively coarsening we obtain a succession of smaller graphs which approximate the original graph G.

For graph coarsening steps we used maximum independent sets, which have been in use for multilevel graph partitioning [1, 5]. Connectivity of an affinity graph is important to many manifold learning algorithms but coarsening by maximum independent sets does not guarantee that the coarse graph is connected. However, Algorithm 1 visits the vertices in a special order to build the maximum independent set, so that it preserves the connectivity of the graph in the coarsening stage. This is now explained.

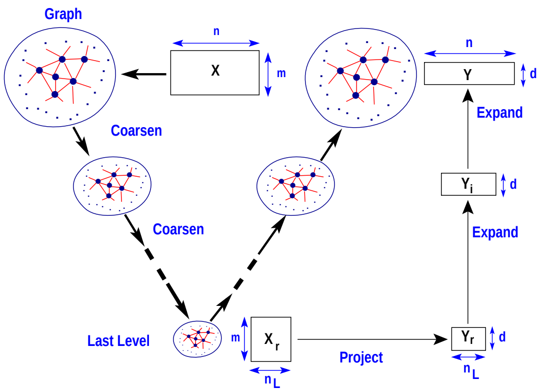

Figure 2: A sketch of the multilevel nonlinear dimensionality reduction.

```
**Input:** Graph $G = (V,E)$ with $V = \{1, \ldots, n\}$.
**Output:** The coarsened graph $\widehat{G} = (\widehat{V}, \widehat{E})$.
$\widehat{V} \leftarrow \emptyset$                          $\triangleright$ maximum independent set
$\widehat{U} \leftarrow \emptyset$                          $\triangleright$ complement set of $\widehat{V}$
Randomly pick $k_0 \in V$; $S \leftarrow \{k_0\}$.          $\triangleright$ (†)
repeat
    Randomly pick $i \in S$; $S \leftarrow S \setminus \{i\}$.   $\triangleright$ (*)
    if $i \notin \widehat{U} \cup \widehat{V}$ then
        $\widehat{V} \leftarrow \widehat{V} \cup \{i\}$                $\triangleright$ (**)
        for all $(i,j) \in E$, $j \notin \widehat{U}$ do
            $\widehat{U} \leftarrow \widehat{U} \cup \{j\}$
            for all $(j,k) \in E$ do
                if $k \notin \widehat{U} \cup \widehat{V}$ then
                    $S \leftarrow S \cup \{k\}$                  $\triangleright$ (††)
                end if
            end for
        end for
    end if
until $S = \emptyset$
$\widehat{E} \leftarrow \emptyset$                          $\triangleright$ edge set of $\widehat{G}$
for all $i, k \in \widehat{V}$ do
    if $\exists j$ such that $(i,j), (j,k) \in E$ then
        $\widehat{E} \leftarrow \widehat{E} \cup \{(i,k)\}$
    end if
end for
```

Algorithm 1: Graph coarsening by a maximum independent set.

Consider the steps of Algorithm 1 to compute the coarse graph $\widehat{G} = (\widehat{V}, \widehat{E})$. We claim that for each vertex k added to $\widehat{V}$, other than the very first element k_0 added to S, there exists a path consisting of edges in $\widehat{E}$ linking vertices k_0 and k. We now prove our claim by induction. All vertices in $\widehat{V}$ are from S in (*) and added in (**). Each element k ever in S, except the very first k_0 in (†), is added to S in (††), where there exist $(i, j), (j, k) \in E$ with i already in $\widehat{V}$. Since there is a path $i \to j \to k$ in the fine graph, if k is added into $\widehat{V}$ in some later iteration, then there will be an edge $(i, k) \in \widehat{E}$ in the coarse graph as instructed by the bottom part of the algorithm. Assuming that previous vertices added to $\hat{V}$ satisfy our claim, there exists a path consisting of edges in $\hat{E}$ linking k_0 and i, unless $i = k_0$. Since $(i, k) \in \hat{E}$, k_0 also links to k via a path in the coarse graph. This proves our claim by induction. Therefore, the coarse graph $\widehat{G} = (\widehat{V}, \widehat{E})$ is guaranteed to be connected under the condition that the original graph is.

Algorithm 1 provides an affinity graph $\widehat{G} = (\widehat{V}, \widehat{E})$ of the coarse level. Therefore, it is not necessary to compute a kNN graph for the graphs obtained at each level. In addition, we need the distances between nearby points in the coarse graph in the following two situations. First, some manifold learning algorithms, such as Isomap, need distances between nearby points to compute the mapping. Second, the multilevel refining stage, to be described later, will require the edge weights, and some weighting schemes, such as Gaussian weights, depend on the distances between nearby points.

We use δ and $\hat{\delta}$ to denote the distances at the fine and coarse levels, respectively. Given $(i, j) \in \widehat{E}$, one can simply use the actual distance $\hat{\delta}(x_i, x_j) = \|x_i - x_j\|_2$ for the coarse level. Alternatively, we define

$$\hat{\delta}(x_i, x_j) = \min_{(i,k),(k,j) \in E} \delta(x_i, x_k) + \delta(x_k, x_j). \tag{5}$$

Then distance computations are avoided at the coarse level. More importantly, if we compute distances by (5) at all levels, the computed distances indeed approximate geodesic distances. This is especially important to Isomap which aims at preserving geodesic distances.

By recursively coarsening the graph, we obtain a succession of graphs $G_1, G_2, \ldots, G_r$, where $G_i = (V_i, E_i)$ is the coarse graph of level i for $i = 1, \ldots, r$, and G_r is the coarsest level graph. The corresponding data sets are denoted by matrices $X_i \in \mathbb{R}^{m \times |V_i|}$ for $i = 1, \ldots, r$.

3.2 The Dimension Reduction Phase

Given a data set $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{m \times n}$, a dimensionality reduction algorithm produces $Y = [y_1, y_2, \ldots, y_n] \in \mathbb{R}^{d \times n}$ $(d < m)$ such that Y preserves certain features of X. In our multilevel framework, presented in Figure 2, we apply a dimensionality reduction method to the data set $X_r \in \mathbb{R}^{m \times |V_r|}$ of the coarsest level (rth level), and obtain a set $Y_r \in \mathbb{R}^{d \times |V_r|}$ $(d < m)$. The dimensionality reduction methods considered for this task are affinity-graph-based, such as Isomap, LLE, and Laplacian eigenmaps, where the graph from the multilevel framework is used. Recall that it is not necessary to build a kNN graph at the coarsest level.

Note that Isomap and Laplacian eigenmaps use an undirected affinity graph (i.e., applying symmetrization to a kNN graph), whereas LLE uses a directed affinity graph (i.e., a kNN graph w/o symmetrization). In our multilevel framework the affinity graph is undirected, regardless of the dimensionality reduction method applied at the bottom level.

3.3 The Refining Phase

The objective of the refining phase is to obtain a reduced representation $Y \in \mathbb{R}^{d \times n}$ of the data $X \in \mathbb{R}^{m \times n}$, where $n = |V_1|$, at the finest level, starting from the reduced representation $Y_r \in \mathbb{R}^{d \times |V_r|}$ of data $X_r \in \mathbb{R}^{m \times |V_r|}$ of the coarsest level (rth level).

We refine the data level by level in the low dimensional space as follows. We denote by $G = (V, E)$ and $\widehat{G} = (\widehat{V}, \widehat{E})$ the two graphs of the kth and $(k+1)$st levels, respectively. For each level $k = r-1, r-2, \ldots, 1$, we recursively build the reduced representation Y of the kth level from $\widehat{Y}$ of the $(k+1)$st level in a low dimensional space, by minimizing

$$E = \sum_{i,j \in V} w_{ij} \|y_i - y_j\|_2^2, \tag{6}$$

where $W = [w_{ij}]$ is a symmetric weight matrix; each entry w_{ij} is nonzero only if the vertices i, j are adjacent (i.e., connected by an edge).

Yet not specified are the weights between nearby data points. We adopt the the Gaussian weighting scheme $w_{ij} = exp(-\delta(x_i, x_j)^2/t)$ for some scalar $t > 0$. The distance function $\delta(x_i, x_j)$ between x_i and x_j can be the Euclidean distance $\|x_i - x_j\|_2$ as that in (4). With our multilevel framework we use the approximate geodesic distance (5) across all levels. When $\sigma = \infty$, we obtain the binary weights, i.e., $w_{ij} = 1$ for all adjacent vertices and otherwise 0.

We denote the vertex set of the coarse level by $\widehat{V} \subset V$, and its complement by $\widehat{U} = V \setminus \widehat{V}$. We rewrite (6) as

$$\begin{aligned} E &= \sum_{i \in \widehat{U}} \sum_{j \in \widehat{U}} w_{ij} \|y_i - y_j\|_2^2 + \sum_{i \in \widehat{V}} \sum_{j \in \widehat{V}} w_{ij} \|y_i - y_j\|_2^2 && (7) \\ &+ 2 \sum_{i \in \widehat{U}} \sum_{j \in \widehat{V}} w_{ij} \|y_i - y_j\|_2^2. && (8) \end{aligned}$$

The first term of (7) can be written as

$$\sum_{i \in \widehat{U}} \sum_{j \in \widehat{U}} w_{ij} \|y_i - y_j\|_2^2 = 2\,\mathrm{trace}[Y_1 (D_1 - W_1) Y_1^T], \tag{9}$$

where $Y_1 \in \mathbb{R}^{d \times |\widehat{U}|}$ includes the points to be determined in Y, $W_1 \in \mathbb{R}^{|\widehat{U}| \times |\widehat{U}|}$ is the matrix of edge weights with edges connecting points in Y_1, and $D_1 \in \mathbb{R}^{|\widehat{U}| \times |\widehat{U}|}$ is the diagonal matrix whose entries are the row/column sums of W_1.

The second term of (7) is a constant, since it depends only on points in $Y_2 \in \mathbb{R}^{d \times |\widehat{V}|}$ that have been already determined at the coarse level.

The term of (8), after some algebra, can be written as

$$\begin{aligned} & 2 \sum_{i \in \widehat{U}} \sum_{j \in \widehat{V}} w_{ij} \|y_i - y_j\|_2^2 \\ = \; & 2\,\mathrm{trace}[Y_1 D_{12} Y_1^T] - 4\,\mathrm{trace}[Y_1 W_{12} Y_2^T] + C, \end{aligned} \tag{10}$$

where $W_{12} \in \mathbb{R}^{|\widehat{U}| \times |\widehat{V}|}$ is the matrix of edge weights with edges connecting points to be determined (i.e., indexed by $\widehat{U}$) and those already determined (i.e., indexed by $\widehat{V}$), $D_{12} \in \mathbb{R}^{|\widehat{U}| \times |\widehat{U}|}$ is the diagonal matrix whose entries are the column sums of W_{12}, and C is a constant.

Putting the expressions (9) and (10) together back into (7), we obtain a quadratic function:

$$2\,\mathrm{trace}[Y_1(D_1 - W_1 + D_{12})Y_1^T] - 4\,\mathrm{trace}[Y_2 W_{12}^T Y_1^T] \quad (11)$$

plus a constant term. To minimize E, we set the partial derivatives of (11) to zero, and obtain the equation

$$Y_1(L_1 + D_{12}) = Y_2 W_{12}^T, \quad (12)$$

where $L_1 = D_1 - W_1 \in \mathbb{R}^{|\widehat{U}|\times|\widehat{U}|}$ is the Laplacian matrix of the points to be determined.

Two remarks must be made. First, L_1 is symmetric and diagonally dominant with a positive diagonal. By the Gershgorin circle theorem, L_1 is positive semidefinite. It also has the smallest eigenvalue 0 associated with eigenvector e, the column vector of ones. D_{12} is diagonal with nonnegative entries. Therefore, the objective function (11) is convex, and Y_1 is the minimizer if and only if (12) holds. Second, our data coarsening method is based on maximum independent sets, and we refine the mapping level by level. Hence each undetermined vertex $i \in \hat{U}$ has at least one determined neighbor $j \in \hat{V}$ associated with a positive weight $w_{ij} > 0$. So D_{12} has a positive diagonal. Recall that L_1 is symmetric positive semidefinite. By a theorem of Weyl [25, Corollary 4.9], stated below, $L_1 + D_{12}$ is positive definite and therefore nonsingular. Thus, the solution to the linear system (12) is unique, and so is the minimizer of (11).

Theorem 1 (Weyl). *Let A, B be two $n\times n$ Hermitian matrices and $\lambda_k(A)$, $\lambda_k(B)$, $\lambda_k(A+B)$ be the eigenvalues of A, B, and $A+B$ arranged in increasing order for $k = 1,\dots,n$. Then for $k = 1,\dots,n$, we have*

$$\lambda_k(A) + \lambda_1(B) \le \lambda_k(A+B) \le \lambda_k(A) + \lambda_n(B).$$

Putting the points as columns of Y_1 (i.e., indexed by $\widehat{U}$) and those already determined as columns of Y_2 (i.e., indexed by $\widehat{V}$) together, we obtain the reduced representation of the finer level (i.e., vertices indexed by $\widehat{U}\cup\widehat{V}$). By recursively refining the data this way, we obtain a reduced representation of the original data.

4. MANIFOLD LEARNING EXPERIMENTS

In this section we illustrate the application of the proposed multilevel manifold learning scheme. We use the three nonlinear dimensionality reduction methods, Isomap [27], LLE [21], and Laplacian eigenmaps [2], and the versions of the multilevel algorithms which incorporate these techniques at the coarsest level as described earlier.

All experiments were performed in Matlab in sequential mode on a PC equipped with a four-core Intel Xeon E5504 @ 2.0GHz processor. The kNN graph construction is by a brute-force algorithm, which can be improved by an approximation algorithm [6]. We used a C/C++ implementation of Dijkstra's algorithm [9] by John Boyer to solve the all-pair shortest path problem, which arises in Isomap and multilevel-Isomap. We also implemented Algorithm 1 for graph coarsening in C/C++. For all eigencomputations, we used the Matlab routine `eigs` which invokes the Fortran library `ARPACK` [15].

Since Algorithm 1 for coarsening the data is randomized, we report the average numbers from 100 random runs for each data set, each method, and each level $r = 2, 3, 4$ in Tables 1 and 2, which display the average number of images at each coarsening level, and the average CPU time used for graph coarsening, processing for dimensionality reduction, and data refining. For all methods, processing time includes the time used for eigencomputation. For Isomap and multilevel-Isomap, processing time also includes the time to compute the geodesic distances; the time for computing the all-pair shortest distances is also reported as the braced number. For LLE, it includes the time to obtain the reconstruction weights.

4.1 Embedding Evaluation

In order to compare the quality of the nonlinearly mapped data, we adopt two embedding evaluation metrics, the *trustworthiness* and *continuity* of the proximity relationships of data entries [28, 29].

Let $x_1,\dots,x_n$ be the points in the high dimensional space, and $y_1,\dots,y_n$ be the mapped points in the low dimensional space. Denote by $r(i,j)$ the rank of x_j in the ordering according to the distance from x_i. The longest vertex x_j from x_i has $r(i,j) = 1$, and the shortest vertex x_j from x_i has $r(i,j) = n-1$. Likewise, denote by $\hat{r}(i,j)$ the rank of y_j in the ordering according to the distance from y_i. The trustworthiness is defined by

$$T(p) = \frac{2}{np(2n-3p-1)} \sum_{i=1}^{n} \sum_{j\in U_p(i)} (r(i,j) - p),$$

where $U_p(i)$ contains the indices of p nearest neighbors of y_i in the low dimensional space. The continuity is defined by

$$C(p) = \frac{2}{np(2n-3p-1)} \sum_{i=1}^{n} \sum_{j\in V_p(i)} (\hat{r}(i,j) - p),$$

where $V_p(i)$ contains the indices of p nearest neighbors of x_i in the high dimensional space.

The higher the trustworthiness or continuity, the better the manifold mapping. Both $T(p)$ and $C(p)$ are bounded above by 1. The upper bound 1 is reached if and only if $U_p(i) = V_p(i)$ for $i = 1,\dots,n$, which means that the p nearest neighbors for each data entry in the high dimensional space coincide with those in the low dimensional space.

4.2 Frey Face Video Frames

The `Frey Face` data set [21][1] contains 1,965 face images of a single person, Brendan Frey, taken from sequential frames of a small video. Each image is of size 20-by-28 in grayscale, and hence in 560-dimensional space after vectorization.

We report the result using a kNN graph with $k = 12$ and embedding dimensions $d = 2$. In our multilevel framework Figure 3 illustrates the two-dimensional mappings of the these images obtained by LLE and multilevel-LLE. We can observe that all plots exhibits two intrinsic attributes, i.e., pose (left-right) and expression (serious-happy), which are correlated with the coordinate axes. This property is also more or less reflected in the plots by Isomap, multilevel-Isomap, Eigenmaps, and multilevel-Eigenmaps, which are not shown to save space.

The average computation time for manifold learning, displayed in Table 1, consists of four parts: the kNN graph construction, embedding process, and for multilevel methods the graph coarsening time and data refining. Our multilevel

[1] `http://cs.nyu.edu/~roweis/data.html`

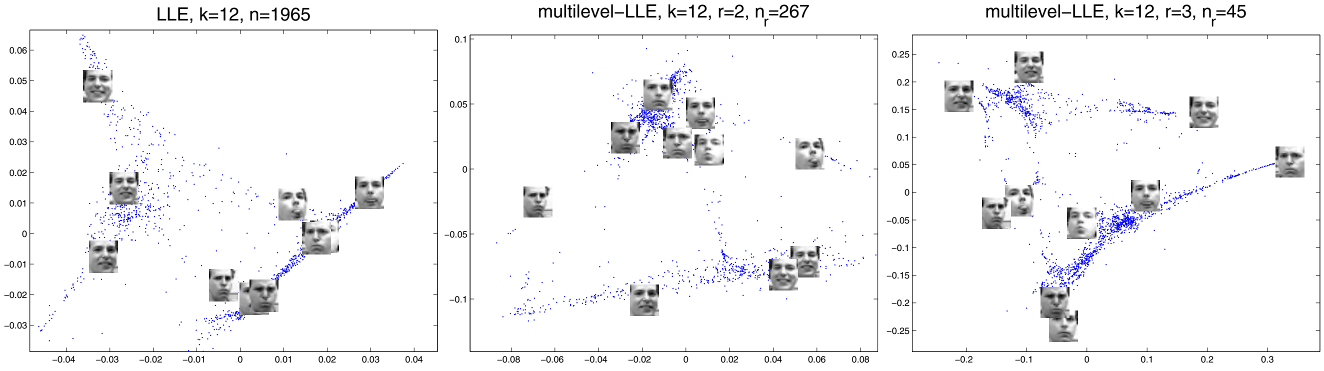

Figure 3: 2D mappings of `Frey Face` database using LLE and multilevel-LLE.

Table 1: Computation time for `Frey Face` data set.

kNN time (secs)	level	average # of images	coarsen. time (secs)	Isomap proc. time	Isomap ref. time	LLE proc. time	LLE ref. time	Eigenmaps proc. time	Eigenmaps ref. time
0.9767	#1	1965	N/A	4.79 (4.27)	N/A	1.1000	N/A	0.2800	N/A
	#2	252.02	0.0012	0.0657 (0.0560)	0.0989	0.1151	0.0986	0.0275	0.0975
	#3	47.06	0.0001	0.0075 (0.0009)	0.0050	0.0204	0.0048	0.0188	0.0044
	#4	12.48	-	0.0039 (0.0001)	0.0008	0.0079	0.0009	0.0126	0.0011

technique reduced the computation time significantly. The savings with $r = 2$ levels for Isomap, LLE, and Eigenmaps are about 80.2%, 42.6%, and 12.2%, respectively. Using more levels resulted in more time savings.

Figure 4 displays the plots of trustworthiness and continuity as a function of p, the size of the neighborhood used in measuring them, where we set the number of levels up to four. Our multilevel technique improved Isomap and LLE in both computation time and embedding quality, while multilevel-Eigenmaps performed comparable to Eigenmaps using this data set.

4.3 Labeled Faces in the Wild

The `Labeled Faces in the Wild` (LFW) data set [11][2] includes 13,233 images of size 250-by-250 in RGB color of 5,749 unique individuals. We resized these images to 50-by-50 and converted them to grayscale for manifold learning experiments. Figure 5 lists the sample images of four individuals: Naoto Kan (1-4 images in row 1), Sally Field (5-8 images in row 1), Helen Clark (1-4 images in row 2), and Gilberto Rodriguez Orejuela (5-8 images in row 2).

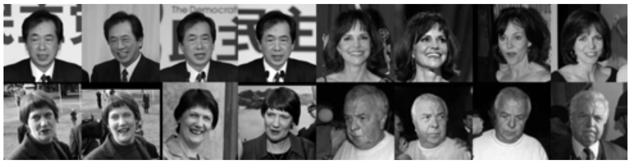

Figure 5: Sample LFW face images.

We report the experimental results using a kNN graph with $k = 6$ and embedding dimensions $d = 2$. In the experiments on this data set, some of the vertices in the coarse

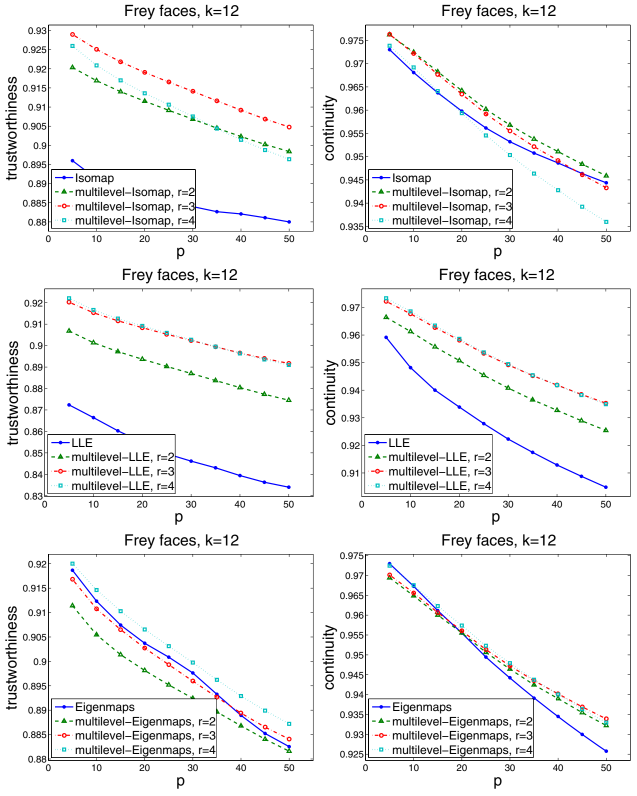

Figure 4: Trustworthiness and continuity of `Frey Face` database.

[2] `http://vis-www.cs.umass.edu/lfw/`

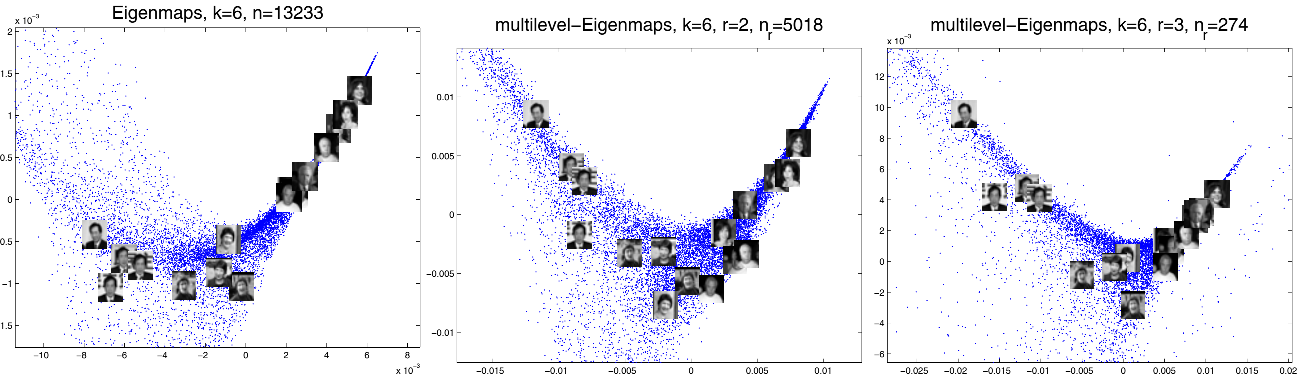

Figure 6: 2D mappings of the LFW data set by Isomap and multilevel-Isomap.

Table 2: Computation time for the LFW data set.

kNN time (secs)	level	average # of images	coarsen. time (secs)	Isomap proc. time	Isomap ref. time	LLE proc. time	LLE ref. time	Eigenmaps proc. time	Eigenmaps ref. time
102.13	#1	13233	N/A	296.78 (270.02)	N/A	19.190	N/A	9.5500	N/A
	#2	5091.14	0.1248	65.954 (62.432)	2.2159	3.5615	2.1905	1.2286	2.1933
	#3	292.22	0.0183	0.1073 (0.0909)	5.5481	0.1450	5.5519	0.0418	5.5908
	#4	29.98	0.0006	0.0909 (0.0014)	0.0209	0.0210	0.0213	0.0223	0.0220

graphs have their numbers of neighbors increased significantly. However for LLE and Eigenmaps, the number of neighbors per vertex is kept modest to preserve the 'locality' in the embedding. Therefore, we trimmed the graph at the coarse level such that each vertex has at most $k = 6$ neighbors, i.e., outgoing edges. This results in an unsymmetric affinity matrix for LLE. For Eigenmaps we applied symmetrization as before. We did not apply this step for the global method Isomap.

Figure 6 illustrates the two-dimensional mappings of the LFW data set, using Eigenmaps and multilevel-Eigenmaps, where sample images listed in Figure 5 are displayed. To some extent, these images are clustered into four groups of the individuals, and our multilevel method preserved their relative locations in the mapping. On the other hand, the mapped data tends to lose cohesiveness when the number of levels increases.

Table 2 reports the average computation time for manifold learning, which consists of four parts: the kNN graph construction, embedding process, and for multilevel methods the graph coarsening time and data refining. Note that our multilevel technique generally achieved significant savings in CPU time. Using $r = 2$ levels, our multilevel technique achieved about 57.3% savings in computation time for Isomap, 11.0% savings for LLE, and 5.4% savings for Eigenmaps. Omitting the time for kNN graph construction, the savings were 77.0%, 69.4%, and 62.9% for Isomap, LLE, and Eigenmaps, respectively.

Figure 7 displays the plots of trustworthiness and continuity as a function of p, the size of the neighborhood used in the measurement, where we set the number of levels up to four. In this experiment our multilevel technique improved LLE and Eigenmaps, and multilevel-Isomap performed comparably to Isomap.

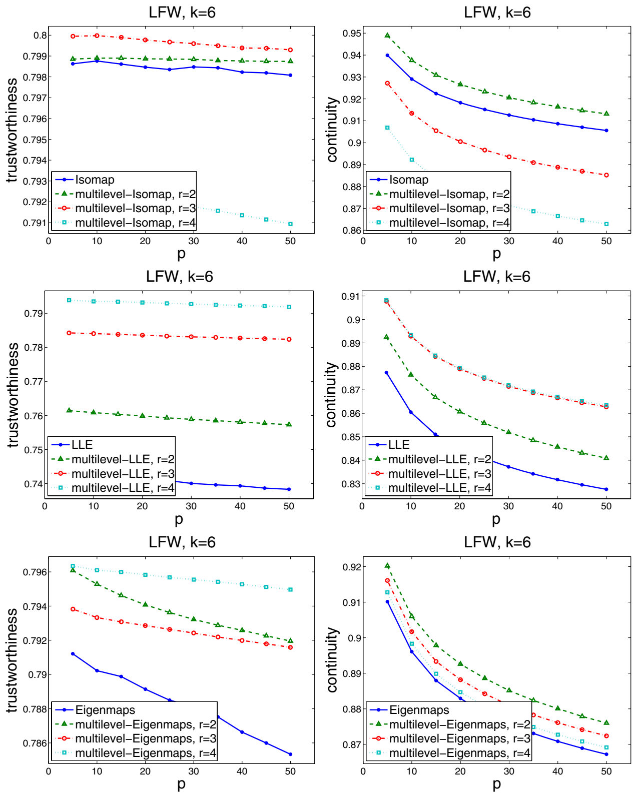

Figure 7: Trustworthiness and continuity of the LFW data set.

5. APPLICATION TO CLUSTERING

Given set of points $X = [x_1, x_2, \dots, x_n]$ in Euclidean space, the objective of clustering is to partition it into a certain number of subsets, called clusters, which are as distinct as possible. The K-means algorithm, as one of the best-known clustering methods available, (locally) minimizes the quantization error:

$$E(s, w) = \sum_{i}^{n} \|x_i - c(s(i))\|_2^2, \qquad (13)$$

where $s(i)$ is the index of the cluster to which x_i belongs, and $c(j)$ is the prototype, e.g., the centroid of cluster j.

The performance of K-means clustering can be improved by a spectral clustering algorithm in [17], which is presented in Algorithm 2.

```
{Given $x_1, \cdots, x_n \in \mathbb{R}^m$, partition them into $K$ clusters.}
Form a symmetric affinity graph $G = (V, E)$ of $x_1, \dots, x_n$.
Compute $W = [w_{ij}] \in \mathbb{R}^{n \times n}$ with $w_{ij} = \exp(-\|x_i - x_j\|_2^2/t)$ if $x_i, x_j$ are neighbors and otherwise 0.
Compute diagonal $D = [d_{ii}] \in \mathbb{R}^{n \times n}$ with $d_{ii} = \sum_{j=1}^{n} w_{ij}$.
Compute the normalized matrix $A = D^{-1/2} W D^{-1/2}$.
Compute the $K$ eigenvectors $v_1, \dots, v_K$ of $A$ corresponding to the $K$ largest eigenvalues.
Form $Y = [y_1, \dots, y_n] = [v_1, \dots, v_K]^T \in \mathbb{R}^{K \times n}$.
Normalize $\bar{y}_i = y_i/\|y_i\|_2$ for $i = 1, \dots, n$.
Apply the K-means clustering, initialized randomly, to $\bar{y}_1, \dots, \bar{y}_n$.
Assign $x_i$ to cluster $j$ if $\bar{y}_i$ was assigned to cluster $j$.
```

Algorithm 2: A spectral clustering algorithm.

Two observations on Algorithm 2 deserve noting. First, $I - A = D^{-1/2} L D^{-1/2}$, where L is the graph Laplacian, and A and $I - A$ share the same eigenvectors with eigenvalues changed form λ_i to $1 - \lambda_i$. Therefore, the mapping to Y is indeed performing normalized Laplacian Eigenmaps. Second, the normalized Laplacian Eigenmaps multiplies each of the Eigenmaps points $y_1, \dots, y_n$ by a constant number. However, Algorithm 2 applies $\bar{y}_i = y_i/\|y_i\|$ for $i = 1, \dots, n$, the previous scaling is ineffective. Therefore, we can simply apply the Laplacian Eigenmaps without dropping the bottom eigenvector, followed by the normalization $\bar{y}_i = y_i/\|y_i\|$ for $i = 1, \dots, n$.

Both K-means clustering and the clustering algorithm in Algorithm 2 have the same drawback, that it is sensitive to initialization. Random initialization could yield poor results in extreme cases, which may be avoided to some extent by a structured initialization scheme utilizing our multilevel technique. The procedure is sketched next.

We first recursively apply the graph coarsening method in Algorithm 1, and obtain a succession of graphs $G_i = (V_i, E_i)$ for $i = 1, \dots, r$, where $V_1 = \{1, \dots, n\}$ is the set of indices of the given data $x_1, \dots, x_n$. Then we cluster the data points at the bottom level by Algorithm 2, where we also obtain the mapped data $\bar{Y}_r$ in the spectral space.

We also refine the data level by level. However, since the mapped data are expected to unit norm. We normalize the newly mapped points from solving (12) in each data refining level. This normalization step indeed matches the relaxation theory for multiclass clustering [32]. We obtain a succession of sets of low dimensional points $\bar{Y}_r, \dots, \bar{Y}_1$ (the column norms are all ones). For each level $i = r-1, r-2, \dots, 1$, we still do the K-means clustering to the normalized $\bar{Y}_i$, initialized by the clustering centroids at $(i+1)$st level. The pseudocode is given in Algorithm 3.

```
{Given $x_1, \cdots, x_n \in \mathbb{R}^m$, partition them into $K$ clusters.}
Form a symmetric affinity graph $G = (V, E)$ of $x_1, \dots, x_n$.
Form a succession of graphs $G_1, \dots, G_r$ by Algorithm 1.
Apply Algorithm 2 to the data points at the bottom level with $G_r$, and obtain mapped points $\bar{Y}_r$ and $K$ clusters.
Obtain $\bar{Y}_r, \bar{Y}_{r-1}, \dots, \bar{Y}_1$ by multilevel data refining. At each level, the newly mapped data are normalized.
for $i = r-1, \dots, 1$ do
    Apply K-means clustering to points in $\bar{Y}_i$, initialized by the $K$ centroids at level $i+1$.
end for
{The mapped data at top level are denoted by $\bar{y}_1, \dots, \bar{y}_n$.}
Assign $x_i$ to cluster $j$ if $\bar{y}_i$ was assigned to cluster $j$.
```

Algorithm 3: A multilevel spectral clustering algorithm.

6. CLUSTERING EXPERIMENTS

We compare empirically the performance of three clustering algorithms discussed in Section 5.

1. K-means clustering.
2. Spectral clustering (Algorithm 2).
3. Multilevel spectral clustering (Algorithm 3).

In all experiments, we assume the number of classes is known in advance which is used as the number of clusterings in the three clustering algorithms. We also used $r = 2$ levels in the multilevel spectral clustering algorithm in all tests.

The quality of clusters are evaluated by *purity* and *entropy* [33]. For each algorithm and each data set, we report the average of the results from 100 random initializations. The purity and entropy are defined as:

$$\text{purity} = \sum_{i=1}^{K} \frac{n_i}{n} \text{purity}(i), \ \text{purity}(i) = \frac{1}{n_i} \max_j \left(n_i^j\right),$$

$$\text{entropy} = \sum_{i=1}^{K} \frac{n_i}{n} \text{entropy}(i), \ \text{entropy}(i) = -\sum_{j=1}^{K} \frac{n_i^j}{n_i} \log_K \frac{n_i^j}{n_i},$$

where K is the number of clusters, n_i^j is the number of entries of class j in cluster i, and n_i is the number of data entries in cluster i.

Both purity and entropy are bounded between 0 and 1. The larger the purity, or the smaller the total entropy, the better the performance. The optimal value 0 of entropy and the optimal value 1 of purity are met, if and only if the clusters match exactly the classes.

We have used several data sets for clustering experiments. Our multilevel spectral clustering algorithm does not always yield better cluster quality than the spectral method. However, we have observed the expected computational savings on clustering larger data sets. For example in clustering the `USPS` digit set[3] which consist of 11,000 digit images, we obtained 22.1% CPU time savings (excluding kNN graph construction time in the comparison).

[3] `http://cs.nyu.edu/~roweis/data.html`

Figure 8: Sample ORL face images.

Figure 9: Sample AR face images.

Figure 10: Sample Yale face images.

Figure 11: Sample USPS digit images.

Now we report the results of our experiments on three face image databases, `Yale` faces, `ORL` faces [20], `AR` faces [16], and one digit image databases, `USPS` digits. These images are all in grayscale. The sample images are displayed in Figures 8-11. The results of clustering experiments are summarized in Table 3. As can be observed, the K-means clustering was outperformed by the spectral clustering which was further improved by our multilevel technique, except for the `USPS` database we got the worse entropy.

The last experiment we report is on a collection of document sets J1-J11 from [4]. All these 11 sets consist of the same 185 documents but differ in the number of key words from 183 to 10,536, where J1 contains all words and forms a matrix of size 10,536-by-185. Each document has been assigned by hand a label according to its topic. There are 10 different labels (topics) in total. The clustering results are presented in Figure 12. In all we normalized the document vectors and used a kNN graph with $k = 4$. Multilevel spectral clustering gave better results than spectral clustering in 7 of the 11 cases. Both methods outperformed the K-means algorithm significantly. The details are omitted.

7. CONCLUSION

The class of multilevel nonlinear dimension reduction methods for manifold learning presented in this paper aim at reducing cost without sacrificing accuracy. Experiments indicate that the proposed multilevel framework usually reduces the computational cost of some existing methods for manifold learning, while yielding comparable or better results.

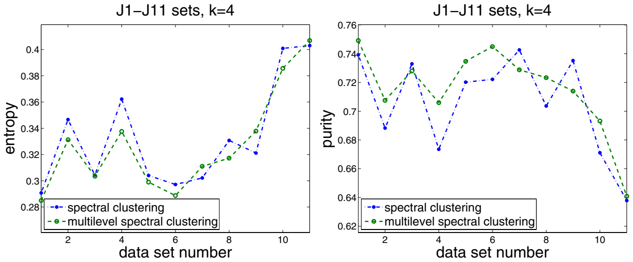

Figure 12: Results of document clustering

We have shown an application of the method to spectral clustering, by incorporating our multilevel technique with a spectral clustering algorithm for structured initialization. Experiments show that this may result in an improvement in clustering quality.

The multilevel nonlinear dimension reduction techniques represented in this paper are based on maximum independent sets, which may result in rapid coarsening that affect the performance. Future work will investigate the use of other coarsening techniques (e.g., based on maximal matching) to alleviate this behavior.

8. REFERENCES

[1] S. T. Barnard and H. D. Simon. A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. *Concurrency: Practice and Experience*, 6:101–107, 1994.

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

[3] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[4] D. L. Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):324–344, 1998.

[5] T. Chan, B. Smith, and J. Zou. Multigrid and domain decomposition methods for unstructured meshes. In *The 3rd International Conference on Advances in Numerical Methods and Applications*, pages 53–62, Sofia, Bulgaria, 1994.

[6] J. Chen, H.-r. Fang, and Y. Saad. Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. *J. of Machine Learning Research*, 10:1989–2012, 2009.

[7] R. R. Coifman and S. Lafon. Diffusion maps. *Appl. Comput. Harmon. Anal.*, 21:5–30, 2006.

[8] V. de Silva and J. B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, Cambridge, MA, 2003.

[9] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

Table 3: Results of image clustering.

clustering method		`ORL` faces	`AR` faces	`Yale` faces	USPS digits
# subjects		40	126	15	10
# images per subject		10	8	11	1100
image size		112×92	38×11	112×92	16×16
clustering method	kNN graph	$k = 4$	$k = 3$	$k = 4$	$k = 4$
K-means clustering	purity	0.637	0.465	0.506	0.469
	entropy	0.209	0.264	0.425	0.561
spectral clustering	purity	0.729	0.571	0.574	0.660
	entropy	0.142	0.202	0.395	**0.347**
multilevel spectral clustering	purity	**0.760**	**0.579**	**0.578**	**0.662**
	entropy	**0.127**	**0.181**	**0.378**	0.353

[10] D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. In *Proceedings of the National Academy of Arts and Sciences*, pages 100:5591–5596, 2003.

[11] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[12] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *J. Parallel Distrib. Comput.*, 48(1):96–129, 1998.

[13] G. Karypis and V. Kumar. Multilevel k-way hypergraph partitioning. *VLSI Design*, 11(3):285–300, 2000.

[14] S. Lafon and A. B. Lee. Diffusion maps and coarse-graining: a unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(9):1393–1403, 2006.

[15] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods.* SIAM Publications, 1998.

[16] A. M. Martínez and A. C. Kak. PCA versus LDA. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(2):228–233, 2001.

[17] A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14 (NIPS)*, 2002.

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

[19] S. Sakellaridi, H.-r. Fang, and Y. Saad. Graph-based multilevel dimensionality reduction with applications to eigenfaces and latent semantic indexing. In *The 7th International Conference on Machine Learning and Applications (ICMLA)*, pages 194–200, Washington, DC, USA, 2008. IEEE Computer Society.

[20] F. S. Samaria and A. C. Harter. Parameterisation of a stochastic model for human face identification. In *The 2nd IEEE Workshop on Applications of Computer Vision*, pages 138–142, 1994.

[21] L. K. Saul and S. T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *J. of Machine Learning Research*, 4:119–155, 2003.

[22] L. K. Saul, K. Q. Weinberger, J. H. Ham, F. Sha, and D. D. Lee. Spectral methods for dimensionality reduction. In *Semisupervised Learning.* MIT Press, Cambridge, MA, 2006.

[23] F. Sha and L. K. Saul. Analysis and extension of spectral methods for nonlinear dimensionality reduction. In *The 22nd international conference on Machine learning (ICML)*, pages 784–791, New York, NY, USA, 2005. ACM.

[24] B. Shaw and T. Jebara. Minimum volume embedding. In *The 11th Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 2, pages 460–467, 2007.

[25] G. W. Stewart and J.-g. Sun. *Matrix Perturbation Theory.* Academic Press, 1990.

[26] A. Talwalkar, S. Kumar, and H. A. Rowley. Large-scale manifold learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

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

[28] J. Venna and S. Kaski. Neighborhood preservation in nonlinear production methods: An experimental study. In *International Conference on Artificial Neural Networks (ICANN)*, pages 485–491, 2001.

[29] J. Venna and S. Kaski. Local multidimensional scaling. *Neural Networks*, 19(6–7):889–899, 2006.

[30] Q. Weinberger, B. D. Packer, and L. K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *The 10th International Workshop on Artificial Intelligence and Statistics (AISTATS)*, pages 381–388, 2005.

[31] K. Q. Weinerger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal on Computer Vision*, 70(1):77–90, 2006.

[32] S. X. Yu and J. Shi. Multiclass spectral clustering. In *The 9th IEEE International Conference on Computer Vision (ICCV)*, pages 313–319, Washington, DC, USA, 2003. IEEE Computer Society.

[33] Y. Zhao and G. Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55(3):311–331, 2004.