Adaptive Manifold Learning

Zhenyue Zhang, Jing Wang, and Hongyuan Zha

Abstract—Manifold learning algorithms seek to find a low-dimensional parameterization of high-dimensional data. They heavily rely on the notion of what can be considered as local, how accurately the manifold can be approximated locally, and, last but not least, how the local structures can be patched together to produce the global parameterization. In this paper, we develop algorithms that address two key issues in manifold learning: 1) the adaptive selection of the local neighborhood sizes when imposing a connectivity structure on the given set of high-dimensional data points and 2) the adaptive bias reduction in the local low-dimensional embedding by accounting for the variations in the curvature of the manifold as well as its interplay with the sampling density of the data set. We demonstrate the effectiveness of our methods for improving the performance of manifold learning algorithms using both synthetic and real-world data sets.

Index Terms—Manifold learning, dimensionality reduction, neighborhood selection, bias reduction, classification.

1 Introduction

High-dimensional data are ubiquitous in many real-world applications. Understanding the potential intrinsic low-dimensional structures of those high-dimensional data is an essential preprocessing step for a number of further data analysis processes such as feature analysis, pattern classification, and visualization. Recently in the machine learning and pattern recognition communities, there have been advances in developing effective and efficient algorithms for learning nonlinear low-dimensional manifolds from sample data points embedded in high-dimensional spaces, emphasizing simple algorithmic implementation and avoiding optimization problems prone to local minima. The algorithms also find widespread applications, such as microarray gene expression profiles [16], 3D body pose recovery [7], [17] and face recognition [27], image processing [26], [1], and image-based age estimation [10].

The proposed manifold learning algorithms include Isomap [24], locally linear embedding (LLE) [22] and its variations, manifold charting [2], Hessian LLE [6], and local tangent space alignment (LTSA) [30]. The overall framework of most manifold learning algorithms consists of the following three steps: 1) construct nearest neighbor (NN) graph on the set of sample points, 2) linearly approximate the local manifold geometry within the neighborhood of each sample point, and 3) minimize a global error function to obtain the global embedding which involves the solution of an eigenvalue problem. These algorithms have been successfully applied to several computer vision and pattern recognition problems [12], [26]. Several drawbacks and possible extensions of the algorithms have also been pointed out in [23] and [30].

Three key issues in manifold learning determine the effectiveness of the above-mentioned algorithms. One issue is the notion of what can be considered as local. This issue is essential when imposing the connectivity structure on the given set of sample points. It is important to adaptively select the neighborhood sizes to match the local geometry of the manifold. The second issue is how to accurately capture the local geometry of the low-dimensional manifold. It is important to account for the variation in the curvature of the manifold as well as its interplay with the data sampling density in order to mitigate the distortion of the local geometry resulting from the local linear approximation models.[1] Last, but not least, is the issue of how the local structures can be patched together to produce the global parametrization. The neighborhoods used need to have enough overlap to guarantee an accurate global parameterization in the end. The focus of this paper is the first two issues; for the third issue, the reader is referred to [32].

In the literature, there are two commonly used strategies for selecting the neighborhoods for a given set of samples: the k-nearest-neighborhood (k-NN) and ϵ-neighborhood (ϵ-N) methods [22], [24], using the euclidean distances of the samples. For uniformly distributed sample points, k-NN and ϵ-N strategies are roughly equivalent in the sense that, with certain choices for the values of k and ϵ, the two strategies produce the same set of neighborhoods.[2] Generally, k-NN tends to perform better than ϵ-N since it is scale invariant and can better handle data sets with nonuniformly distributed sample points. The amount of overlap among the neighborhoods, i.e., the size of the intersection sets between the neighborhoods, is important for stably solving the associated eigenvalue problems involved in the third step of the manifold learning algorithms. As shown in [32], larger

- *Z. Zhang is with the Department of Mathematics and State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou 310027, P.R. China. E-mail: zyzhang@zju.edu.cn.*
- *J. Wang is with the School of Computer Science and Technology, Huaqiao University, Xiamen 361021, P.R. China. E-mail: wroaring@yahoo.com.cn.*
- *H. Zha is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30322. E-mail: zha@cc.gatech.edu.*

Manuscript received 12 Oct. 2009; revised 28 Aug. 2010; accepted 10 May 2011; published online 16 June 2011.
Recommended for acceptance by D.D. Lee.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-10-0680.
Digital Object Identifier no. 10.1109/TPAMI.2011.115.

1. For Isomap, by the local linear approximation we mean the approximation of the manifold distances between neighbors using the local Euclidian distances.

2. A difference exists for neighbors locating on the boundary of the ϵ-disk.

overlaps will produce better conditioned eigenspaces used for extracting the global embedding for the manifold, thus favoring larger neighborhoods. However, with larger neighborhoods, the accuracy of the local linear approximation will suffer. Furthermore, the variation of the curvature, the sampling density of the data points, and the noise also interact with the neighborhood size and local linear approximation accuracy and further complicate the problem. These problems will be illustrated more concretely in the next section.

The purpose of this paper is to address the aforementioned two key issues. We will discuss the two problems in the context of the local tangent space alignment algorithm [30]. An analysis for the local linear approximation within a neighborhood will be given first, which leads to a criterion for deciding whether a neighborhood can be well approximated within a given accuracy by a linear fitting. Then, we will propose two algorithms for selecting neighbors that satisfy the criterion by exploring the strategies of neighborhood contraction and neighborhood expansion. These adaptive neighborhood selection methods can be used for constructing the connectivity graphs for other manifold learning methods as well. The second improvement studied in this paper is specially designed for LTSA. We modify the minimization model in the original LTSA algorithm by implicitly taking into account the local curvatures of the manifold. This improvement can reduce the bias in the construction of the global embeddings computed by LTSA. We believe that the basic ideas we propose can be similarly adapted to other manifold learning algorithms.

The rest of this paper is organized as follows: We briefly review the LTSA algorithm in Section 2 and discuss some of its failure modes. Section 3 proposes the criterion for measuring the accuracy of linear approximations and neighborhood contraction and neighborhood expansion algorithms for adaptive neighborhood selection. In Section 4, we discuss the modification of the LTSA minimization model to reduce the bias of the embedding of LTSA. Numerical experiments are reported in Section 5.

2 A Brief Review of LTSA

The basic idea of LTSA is to construct *local* linear approximations of the manifold in the form of a collection of *overlapping* approximate tangent spaces at each sample point, and then align those tangent spaces to obtain a global parametrization of the manifold. This idea comes from the mathematical definition of a manifold.[3] LTSA maps the high-dimensional data points on a manifold to points in a lower dimension euclidean space, and this mapping is isometric if the manifold is isometric to its parameter space [30]. We now describe the algorithm in more precise terms. Computational details and derivation of the algorithm can be found in [30].

Algorithm LTSA. Given a data set $X = [x_1, \ldots, x_N]$ with $x_i \in \mathcal{R}^m$, sampled (possibly with noise) from a d-dimensional manifold ($d < m$), $x_i = f(\tau_i) + \epsilon_i$, where $f : \Omega \subset \mathcal{R}^d \to \mathcal{R}^m$, Ω is an open connected subset, and ϵ_i represents noise. LTSA assumes that d is known and proceeds in the following steps (see Fig. 1 for an illustration):

3. Around every point of a manifold there is a neighborhood that is topologically the same as the open unit ball in $\mathcal{R}^d$.

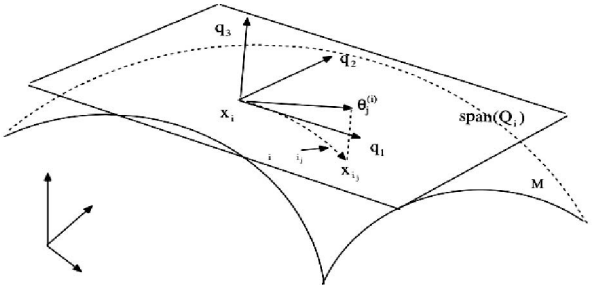

Fig. 1. An illustration of the manifold, tangent space $\mathrm{span}(Q_i)$ at a point x_i, and local coordinate $\theta_j^{(i)}$ of a neighbor x_{i_j}.

1. SETTING NEIGHBORHOODS. For each x_i, $i = 1, \ldots, N$, determine its neighbors $x_{i_1}, \ldots, x_{i_{k_i}}$, for example, its k_i nearest neighbors, including x_i itself.
2. EXTRACTING LOCAL COORDINATES. It can be done, for example, by applying PCA to each set of the selected neighbors. This is an optimal linear fitting to the sample points in the neighborhood, satisfying

$$\sum_{j=1}^{k_i} \left\| x_{i_j} - (\bar{x}_i + Q_i \theta_j^{(i)}) \right\|^2 = \min_{\substack{x, \{\theta_j\}, \\ Q^T Q = I_d}} \sum_{j=1}^{k_i} \|x_{i_j} - (x + Q\theta_j)\|^2. \quad (2.1)$$

   Where $\|\cdot\|$ is the euclidean norm of a vector. The solution of (2.1) is $\bar{x}_i = \frac{1}{k_i}\sum_{j=1}^{k_i} x_{i_j}$ and $\theta_j^{(i)} = Q_i^T(x_{i_j} - \bar{x}_i)$, which are the local coordinates of x_{i_j}.
3. ALIGNING LOCAL COORDINATES. Align the N sets of the *local* coordinates $\Theta_i = [\theta_1^{(i)}, \ldots, \theta_{k_i}^{(i)}]$ to obtain the global coordinates $\tau_1, \ldots, \tau_N$ by minimizing the global reconstruction error,

$$\min_{T,\, TT^T = I_d} \sum_{i=1}^{N} \min_{\substack{c_i \in \mathcal{R}^d, \\ L_i \in \mathcal{R}^{d\times d}}} \frac{1}{k_i} \sum_{j=1}^{k_i} \left\| \tau_{i_j} - \left(c_i + L_i \theta_j^{(i)}\right) \right\|^2, \quad (2.2)$$

   over all row-orthonormal $T = [\tau_1, \ldots, \tau_N] \in \mathcal{R}^{d\times N}$. The error term $\min_{c_i, L_i} \sum \|\tau_{i_j} - (c_i + L_i\theta_j^{(i)})\|^2$ matches the local PCA in (2.1). Alter some algebra, (2.2) leads to an eigenvalue problem:

$$\min_{TT^T = I_d} \mathrm{trace}(T\Phi T^T),$$

   with the symmetric semidefinite matrix $\Phi = \sum_{i=1}^{N} \frac{1}{k_i} S_i \Phi_i S_i^T$, which is called as an alignment matrix, where S_i is the 0-1 selection matrix satisfying $TS_i = [\tau_{i_1}, \ldots, \tau_{i_{k_i}}] = T_i$ and Φ_i is an orthogonal projection with null space $\mathrm{span}([e, \Theta_i^T])$ since $\mathrm{trace}(T_i\Phi_i T_i^T) = \mathrm{trace}(TS_i\Phi_i S_i^T T^T)$.

Generally, LTSA works well if the neighbor sets are well determined [32]. However, the k-NN or ϵ-N with constant k or ϵ may be not suitable for data with variant sample densities or manifold curvatures. In the rest of this section, we give two examples of 1D manifolds to concretely illustrate those problems in neighborhood selection. One has highly varying curvatures and the samples are highly nonuniformly distributed. The other one involves sample points uniformly distributed and the manifold has a unit

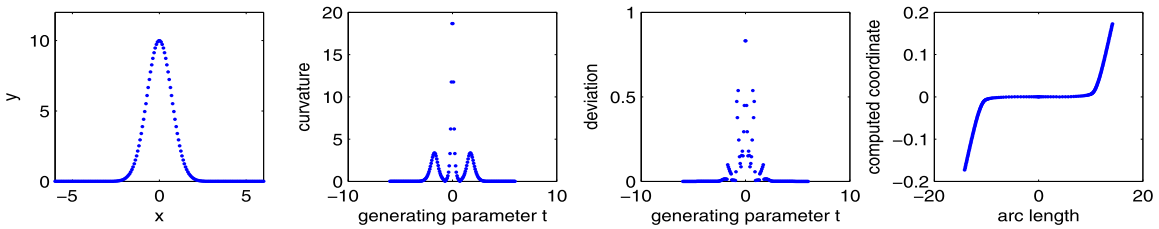

Fig. 2. Example 1 (from left to right). The data points, curvature, and deviation of computed tangent space versus the generating parameters, and the coordinates of LTSA with k-NN neighborhood ($k = 8$) versus the arc lengths.

curvature throughout. LTSA with constant neighborhood size selection fails for both data sets, due to the following reasons: 1) For a small neighborhood size, the constructed neighborhoods do not overlap enough with the nearby neighborhoods. 2) For a large neighborhood size, the sample points within it give rise to poor local linear approximations to the local tangent spaces.

Example 1. The data points are generated as $x_i = f(t_i)$ with the parameterized curve $f(t) = [t, 10e^{-t^2}]^T$ and t_i are equally spaced in the interval $[-6, 6]$. We set $N = 180$. These sample points are of a highly nonuniform density, since the curvature function

$$c(t) = \frac{20|1 - 2t^2|e^{-t^2}}{(1 + 40t^2e^{-2t^2})^{3/2}}$$

varies from 0 to 20 on $[-6, 6]$. LTSA fails to recover the arc length within an affine transformations because the local optimal linear fitting has large deviation to the tangent space at those points x_i near which the curve has large curvatures. The middle two panels of Fig. 2 plot the curvature of the curve and the sine of angle between the linear fitting line and the gradient direction (the deviation of computed tangent space).[4] Isomap and LLE also fail for these data.

Even if a manifold has almost constant curvatures and sample density does not change very much, it may also be not easy to determine a suitable value k or ϵ for all k-NNs or ϵ-Ns. To illustrate this, let us consider the following example:

Example 2. Let $\{x_i\}$ be $N = 500$ points generated as $x_i = [\sin(t_i), \cos(t_i), 0.02t_i]^T$ with t_i randomly chosen from $(0, 4\pi)$. For small k, the deviation of the local optimal linear fitting tends to be small, but the constructed neighborhoods do not have enough overlap. However, increasing the value of k, k-NN will produce worse neighborhoods since the third component of a sample point changes slightly with respect to the generating parameter t_i. This is clearly depicted by the deviations plotted in Fig. 3: As k increases, the neighborhood sizes increase, too, especially for small neighborhoods, while the number of bad neighborhoods also increases. Isomap, LLE, and LTSA fail in this example. We will discuss this example further in Section 5.

The above two examples clearly show that it is desirable to have a better strategy for choosing neighborhood sizes. This issue will be taken up in the next section. We remark that our approaches differ from the simple strategy of varying k or ϵ in k-NN or ϵ-N which may not yield good improvement.

4. The definition of deviation for higher dimensional subspaces will be given in the next section.

3 Adaptive Neighborhood Selection

What can be considered as local in manifold learning algorithms amounts to how to adaptively select neighbor sets. This challenging problem was briefly mentioned in [23] and [30]. However, no detailed solutions have been proposed for solving this problem. In this section, we develop an approach that adheres to the following two requirements:

1. The selected neighbors for each sample point should reflect the local geometric structure of the manifold so that the linear subspace determined by the optimal linear fitting to the neighbor set can approximate the tangent space with high accuracy.
2. Large enough overlaps among the nearby neighborhoods should be maintained to facilitate efficient propagation of local information to obtain the global parameterization [32].

To this end, we first give a criterion for neighbor sets satisfying above requirement 1. A contraction algorithm is then proposed for determining a (potentially smaller) k-NN-based neighborhood that satisfies this criterion. Finally, we discuss how to expand this set as large as possible in order to satisfy requirement 2 while keeping requirement 1 intact as much as possible.

3.1 A Criterion for Tangent Space Approximation

Consider the linear structure of a neighborhood of point $x = f(\tau)$ which can be characterized by the tangent space of

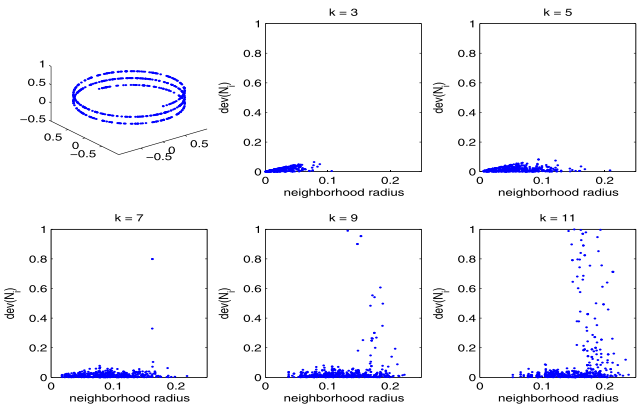

Fig. 3. Example 2. Deviation of computed tangent space plotted versus neighborhood radius for k-NN selection with different k.

the manifold at x. Using first-order Taylor expansion of f at x, a neighbor $\hat{x} = f(\hat{\tau})$ of x can be represented by

$$\hat{x} = x + J_\tau \cdot (\hat{\tau} - \tau) + \varepsilon(\tau, \hat{\tau}). \tag{3.3}$$

Here, $J_\tau \in \mathcal{R}^{m\times d}$ is the Jacobian matrix of f at τ, whose columns span the tangent space,[5] and $\varepsilon(\tau, \hat{\tau})$ is a second-order term of $\hat{\tau} - \tau$ which measures the approximation error of $\hat{x}$ to the tangent space. $\varepsilon(\tau, \hat{\tau})$ is aproximately equal to $\frac{1}{2}(H_\tau(\hat{\tau} - \tau), \hat{\tau} - \tau)$ with the Hessian $H_\tau = [\frac{\partial^2 f}{\partial \tau_i \partial \tau_j}]$ of f and $\|\varepsilon(\tau, \hat{\tau})\| \approx \frac{1}{2}\|H_\tau\|\|\hat{\tau} - \tau\|^2$, depending on the local curvature of the manifold. Based on this, we can select the following neighborhood in the parameter space:

$$\Omega_\tau = \{\hat{\tau} : \|\hat{x} - x - J_\tau(\hat{\tau} - \tau)\| \le \eta\|J_\tau(\hat{\tau} - \tau)\|\} \tag{3.4}$$

or equivalently, the set $\mathcal{N}_x = f(\Omega_\tau)$ in the feature space, with a small constant $\eta \in (0, 1/2)$ for determining the local linear structure. It basically means that the second-order term should be much smaller than the first-order term. Thus, $\mathcal{N}_x$ satisfies requirement 1 with the given accuracy η. Obviously, $\mathcal{N}_x$ depends on the local curvature of the manifold: Smaller curvature near x gives rise to a larger neighborhood, while larger curvature tends to shrink the neighborhood. So, $\mathcal{N}_x$ is generally not equal to the manifold subset contained in an ϵ-ball $\mathcal{S}_x = \{\hat{x} \in \mathcal{R}^m \mid \|\hat{x} - x\| \le \epsilon\}$.

In the case of a set of finite sample points $\{x_i = f(\tau_i)\}$, (3.4) becomes

$$\|x_\ell - x_i - J_{\tau_i}(\tau_\ell - \tau_i)\| \le \eta\|J_{\tau_i}(\tau_\ell - \tau_i)\|. \tag{3.5}$$

For simplifying our analysis shown below, we slightly modify (3.5) by considering the Taylor expansion at the mean $\bar{\tau}_i$ of those points satisfying (3.5):

$$\|x_\ell - \tilde{x}_i - J_{\bar{\tau}_i}(\tau_\ell - \bar{\tau}_i)\| \le \eta\|J_{\bar{\tau}_i}(\tau_\ell - \bar{\tau}_i)\|, \tag{3.6}$$

with $\tilde{x}_i = f(\bar{\tau}_i)$. Let X_i be the matrix of neighbors x_ℓs as its columns and T_i the matrix of τ_ℓs. Equation (3.6) has the following matrix form in Frobenius norm:

$$\|X_i - \tilde{x}_i e^T - J_{\bar{\tau}_i}(T_i - \bar{\tau}_i e^T)\|_F \le \eta\|J_{\bar{\tau}_i}(T_i - \bar{\tau}_i e^T)\|_F, \tag{3.7}$$

where e is a column vector of all 1s. Obviously, (3.6) implies (3.7).

The neighbor set determined by (3.7) may be slightly larger than the neighbor set by (3.6). However, they are mathematically equivalent in the sense that (3.6) holds with an $\tilde{\eta}$ slightly larger than η, if (3.7) is satisfied. In fact, as a column of $E_i = X_i - \tilde{x}_i e^T - J_{\bar{\tau}_i}$ with $\bar{T}_i = T_i - \bar{\tau}_i e^T$, $x_\ell - \tilde{x}_i - J_{\bar{\tau}_i}(\tau_\ell - \bar{\tau}_i)$ has the 2-norm bounded by $\|E_i\|_F$. Thus, (3.7) gives

$$\begin{aligned}\|x_\ell - \tilde{x}_i - J_{\bar{\tau}_i}(\tau_\ell - \bar{\tau}_i)\| &\le \eta\|J_{\bar{\tau}_i}\bar{T}_i\|_F \\ &\le \eta\sqrt{k}\|J_{\bar{\tau}_i}\bar{T}_i\| \le \tilde{\eta}\|J_{\tau_i}\bar{T}_i\|,\end{aligned}$$

where $\tilde{\eta} = \eta\sqrt{k}\kappa(J_{\bar{\tau}_i}(\tau_\ell - \tau_i))$ and $\kappa(\cdot) = \frac{\sigma_{\max}(\cdot)}{\sigma_{\min}(\cdot)}$ denotes the condition number of a matrix with the largest singular value $\sigma_{\max}(\cdot)$ and the smallest singular value $\sigma_{\min}(\cdot)$. The last inequality above is from the fact that $\|A\|_2 = \kappa(A)\sigma_{\min}(A) \le \kappa(A)\|Ay\|_2$ for any square matrix A and vector y. Note that $\kappa(J_{\bar{\tau}_i}\bar{T}_i)$ measures the affine rigidity of the neighbor set generally [32]. The theoretical estimation $\tilde{\eta}$ may be large if $\kappa(J_{\bar{\tau}_i}\bar{T}_i)$ is large since we considered the worst case.

For actual computation, a slight modification of the above model is required since $\tilde{x}_i$ may not be a sample point, and the Jacobian $J_{\bar{\tau}_i}$, τ_ℓ, and τ_i are unknown. Noticing that, by (3.3), the mean $\bar{x}_i$ is a second-order approximation of $\tilde{x}_i$ and $\bar{\varepsilon}_i = \bar{x}_i - \tilde{x}_i$ equals the mean of $\varepsilon(\tau_\ell, \bar{\tau}_i)$, a second-order term, too. On the other hand, $Q_i\theta_{i,\ell}$ with $\theta_{i,\ell} = Q_i^T(x_\ell - \bar{x}_i)$ is the PCA estimation of $J_{\bar{\tau}_i}(\tau_\ell - \bar{\tau}_i)$. So, we can consider the following practical model:

$$\|x_\ell - \bar{x}_i - Q_i\theta_{i,\ell}\| \le \eta\|\theta_{i,\ell}\|$$

without loss of information. Or consider the slightly relaxed but global form:

$$\|X_i - (\bar{x}_i e^T + Q_i\Theta_i)\|_F \le \eta\|\Theta_i\|_F, \tag{3.8}$$

where X_i is the matrix of the neighbors with x_ℓ as its columns and Θ_i is the matrix of $\theta_{i,\ell}$. The condition (3.8) is approximately equivalent to (3.7), as shown in the following lemma.

Lemma 3.1. *If (3.7) holds, then (3.8) is also true with a slightly modified $\tilde{\eta} = \frac{\eta}{1-2\eta}$. On the other hand, let $\alpha_i = \|Q_i\Theta_i - J_{\bar{\tau}}\bar{T}_i\|_F$ and $\beta_i = \frac{k\|\bar{x}_i - \tilde{x}_i\|^2}{\|E_i\|_F^2}$. If (3.8) holds and $\beta_i < 1$, then*

$$\|X_i - \tilde{x}_i e^T - J_{\bar{\tau}_i}(T_i - \bar{\tau}_i e^T)\|_F \le \frac{\eta}{\sqrt{1-\beta_i}}\|J_{\bar{\tau}}\bar{T}_i\| + \frac{(1+\eta)\alpha_i}{\sqrt{1-\beta_i}}.$$

The details of the proof are presented in the Appendix, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2011.115. Based on Lemma 3.1, if the PCA estimation $Q_i\Theta_i$ of $J_{\bar{\tau}_i}\bar{T}_i$ is a good approximation and β_i is negligible compared with $\|E_i\|_F$, then (3.7) with a slightly larger η holds approximately.

In actual computation, we represent the norms $\|\Theta_i\|_F$ and $\|X_i - (\bar{x}_i e^T + Q_i\Theta_i)\|_F$ in (3.8) in terms of the singular values of $X_i - \bar{x}_i e^T$, say $\sigma_1^{(i)} \ge \cdots \ge \sigma_d^{(i)} \ge \cdots \ge \sigma_{k_i}^{(i)}$:

$$\|\Theta_i\|_F = \sqrt{\sum_{j\le d}(\sigma_j^{(i)})^2},$$

$$\|X_i - (\bar{x}_i e^T + Q_i\Theta_i)\|_F = \sqrt{\sum_{j>d}(\sigma_j^{(i)})^2}.$$

Therefore, (3.8) is equivalent to

$$\sqrt{\sum_{j>d}(\sigma_j^{(i)})^2} \le \eta\sqrt{\sum_{j\le d}(\sigma_j^{(i)})^2}. \tag{3.9}$$

The condition (3.8) or (3.9) can be used as a criterion for selecting neighborhoods adaptively. We will propose such an adaptive approach for constructing $\mathcal{N}_i$ in the next two sections. The approach consists of two parts: 1) a contraction step for determining the largest k-NN neighborhood satisfying (3.8), starting from a k-NN neighborhood with large k, and 2) an expansion step for expanding the set in the contraction step as large as possible. We will prove that the resulting set still satisfies the condition (3.8) with a

5. For an isometric manifold, the Jacobian matrix is orthonormal, i.e., $J_\tau^T J_\tau = I$.

slightly increased η. In the last section, we will discuss an adaptive approach of setting η.

3.2 Neighborhood Contraction

Assume that we have a relatively large neighborhood $\mathcal{N}_i = \{x_{i_1}, \ldots, x_{i_{k_{\max}}}\}$ of the sample point x_i in question. For example, it can be determined by the k-NN method. We fix $\eta < 1$. If (3.9) is not satisfied, we contract $\mathcal{N}_i$ by removing the farthest neighbor from their mean $\bar{x}_i$. This contraction step can be repeated until (3.9) holds or until some preset minimal neighborhood size $k_{\min}$ is reached. If no neighborhood satisfying (3.9) can be found, we select a k-NN that minimizes the ratio $r = \sqrt{\sum_{j>d} \sigma_j^2 / \sum_{j \le d} \sigma_j^2}$ among the tested neighborhoods during the contraction process. This contraction process is summarized in the following algorithm:

Algorithm NC: Neighborhood Contraction.

1. Determine the initial $k = k_{\max}$ and the k-NN neighborhood $X_i^{(k)} = [x_{i_1}, \ldots, x_{i_k}]$ for x_i, ordered in non-decreasing distances to x_i, $\|x_{i_1} - x_i\| \le \|x_{i_2} - x_i\| \le \cdots \le \|x_{i_k} - x_i\|$.
2. Compute the singular values $\{\sigma_j^{(k,i)}\}$ of $X_i^{(k)} - \bar{x}_i^{(k)} e^T$, and $r_i^{(k)} = \sqrt{\sum_{j>d} \left(\sigma_j^{(k,i)}\right)^2 \Big/ \sum_{j \le d} \left(\sigma_j^{(k,i)}\right)^2}$.
3. If $r_i^{(k)} < \eta$, then set $X_i = X_i^{(k)}$ and terminate, otherwise do the next step.
4. If $k > k_{\min}$, delete the last column of $X_i^{(k)}$ to obtain $X_i^{(k-1)}$, set $k \leftarrow k - 1$, and return to step step 2, otherwise, go to step 5.
5. Compute $k_i = \arg\min_{k_{\min} \le k \le k_{\max}} r_i^{(k)}$ and set $X_i = X_i^{(k_i)}$.

The main cost of the above algorithm is to compute the singular values of the $m \times k$ matrix $X_i^{(k)} - \bar{x}_i^{(k)} e_k^T$ which has complexity $O(k^2(k+m))$. The whole complexity for N neighborhoods is $O((k_{\max} - k_{\min}) k_{\max}^2 (k_{\max} + m) N)$. There are two ways to accelerate the computation: 1) we can delete more sample points at each step, and therefore reduce the number of times that Step 1 of NC needs to be executed and 2) $\sigma_j^{(k+1,i)}$ can be updated to obtain $\sigma_j^{(k,i)}$, and we do not need to start from scratch. This is the well-known SVD down-dating problem and is discussed in detail in [9]. In our numerical experiments, we set $k_{\min} = d + 1$ or $k_{\min} = d + 2$. At the moment, it is still not clear what the best initial $k_{\max}$ to choose if there is any such best $k_{\max}$. However, in our numerical experiments, the algorithms are not very sensitive to the choice of the initial $k_{\max}$.

DEVIATION OF SUBSPACES. The columns of the orthonormal matrix Q_i determined by the linear fitting span a d-dimensional linear space $\mathrm{span}(Q_i)$ (cf. (2.1)). We will measure its deviation from the ideal tangent space $\mathrm{span}(J_{\tau_i})$ by the distance between $\mathrm{span}(Q_i)$ and $\mathrm{span}(J_{\tau_i})$ defined as follows:

$$\begin{aligned} \mathrm{dev}(\mathcal{N}_i) &= \mathrm{dist}(\mathrm{span}(Q_i), \mathrm{span}(J_{\tau_i})) \\ &= \sqrt{1 - \sigma_{\min}^2(Q_i^T G_i)}, \end{aligned} \tag{3.10}$$

where G_i is an orthonormal basis matrix of the tangent space $\mathrm{span}(J_{\tau_i})$, and $\sigma_{\min}(\cdot)$ is the minimum singular value of a matrix.

3.3 Neighborhood Expansion

For an anisotropic manifold with dimension larger than 1, the curvature of the manifold at a fixed point usually varies much along different directions. The neighborhoods computed by the contraction algorithm have to adapt the largest curvature at each point. Hence, the resulting neighborhood can be unnecessarily small, even though a relatively large neighborhood exists. This is especially true at a point with highly varying curvatures.

Clearly, a neighborhood larger than the contracted one could be obtained by taking advantage of the possible anisotropic shape of the manifold. One possible approach is to add back some of the unselected x_{i_j} in the initial neighborhood of x_i once the contraction step is done, while at the same time keeping the condition (3.8) intact as much as possible. Theorem 3.2 below shows a rule of reselecting those suitable neighbors so that the constraint (3.8) also holds with a slightly increased η.

Theorem 3.2. *Let $\mathcal{N}_i = \{x_{i_1}, \ldots, x_{i_k}\}$ be a neighborhood of x_i satisfying (3.8) with its optimal linear fitting $\{\bar{x}_i + Q_i \theta_j^{(i)}\}$. Assume that we expand $\mathcal{N}_i$ by adding p other neighbors, each satisfying*

$$\left\| x_{i_j} - \bar{x}_i - Q_i \theta_j^{(i)} \right\| \le \eta \left\| \theta_j^{(i)} \right\|, \ \theta_j^{(i)} = Q_i^T (x_{i_j} - \bar{x}_i) \tag{3.11}$$

for $j = k_i + 1, \ldots, k_i + p$. Then, the optimal linear fitting $\{\tilde{x}_i + \tilde{Q}_i \tilde{\theta}_j^{(i)}\}$ to the resulting neighborhood $\tilde{\mathcal{N}}_i$ satisfies

$$\|\tilde{X}_i - \tilde{x}_i e^T - \tilde{Q}_i \tilde{\Theta}_i\|_F \le \eta \alpha_i \|\tilde{\Theta}_i\|_F, \tag{3.12}$$

where $\tilde{\Theta}_i = [\tilde{\theta}_1^{(i)}, \ldots, \tilde{\theta}_{k_i+p}^{(i)}]$ and

$$\alpha_i = \left(1 + \frac{\| \sum_{j=k_i+1}^{k_i+p} \theta_j^{(i)} \|^2}{(k_i + p) \sum_{j=1}^{k_i+p} \|\theta_j^{(i)}\|^2}\right)^{1/2}. \tag{3.13}$$

The proof of Theorem 3.2 is given in the Appendix, available in the online supplemental material.

Generally, $\alpha_i < (1 + \frac{1}{k_i+p})^{1/2}$ since $\| \sum_{j=k_i+1}^{k_i+p} \theta_j^{(i)} \|^2 < \sum_{j=k_i+1}^{k_i+p} \|\theta_j^{(i)}\|^2$. The expanded neighborhood also satisfies (3.8), with $\tilde{\eta} = \alpha_i \eta$ slightly larger than η. Because (3.8) implies a good d-dimensional PCA approximation, Theorem 3.2 also says that the PCA of the contracted neighbors stays the same while adding other neighbors satisfying (3.12). Based on the above analysis, we propose the following neighborhood expansion algorithm with complexity $O(k_{\max}(k_{\max} + d) m N)$ for all the N neighborhoods.

Algorithm NE: Neighborhood Expansion.

1. Compute the optimal linear fitting $\{\bar{x}_i + Q_i \theta_j^{(i)}\}$ to the neighbor set $\mathcal{N}_i$ obtained by the neighborhood contraction step.
2. Compute $\theta_j^{(i)} = Q_i^T (x_{i_j} - \bar{x}_i)$ of all neighbors x_{i_j} out of $\mathcal{N}_i$, $j = k_i + 1, \ldots, k_{\max}$.
3. Add those points x_{i_j} satisfying $\|x_{i_j} - \bar{x}_i - Q_i \theta_j^{(i)}\| \le \eta \|\theta_j^{(i)}\|$ to $\mathcal{N}_i$.

3.4 Adaptively Selecting Parameter η

In this section, we discuss how to adaptively choose η. Clearly, a well-chosen neighbor set should accurately

identify the locally linear structure. According to (3.9), the ratios

$$\rho(\mathcal{N}_i) = \frac{\sqrt{(\sigma_{d+1}^{(i)})^2 + \cdots + (\sigma_k^{(i)})^2}}{\sqrt{(\sigma_1^{(i)})^2 + \cdots + (\sigma_d^{(i)})^2}}, \quad i = 1, 2, \ldots, N,$$

are uniformly bounded by a small constant η if all the neighbor sets $\{\mathcal{N}_i\}$ are well chosen. For a collection of preselected neighbor sets, for example, all k-NN sets $\mathcal{N}_i$ with a given $k = k_0$, some of them are well chosen with the corresponding ratios $\rho(\mathcal{N}_i)$ relatively small. The others have $\rho(\mathcal{N}_i)$ that are relatively large. Hence, these two classes of neighbor sets can be distinguished according to the magnitude of the $\rho(\mathcal{N}_i)$, and the maximum value of the $\rho(\mathcal{N}_i)$ belonging to the class with smaller values can serve as suitable candidate for η. More precisely, we sort $\{\rho(\mathcal{N}_i)\}$ in decreasing order such that $\tilde{\rho}_1 \geq \cdots \geq \tilde{\rho}_N$. The largest gap between two consecutive terms $\tilde{\rho}_j$ and $\tilde{\rho}_{j+1}$ in the sequence provides a partition into the two classes we mentioned before. Specifically, let $\tilde{\rho}_{j+1}/\tilde{\rho}_j = \max_{1 \leq i \leq N-1} \tilde{\rho}_{i+1}/\tilde{\rho}_i$, then those $\mathcal{N}_i$ with $\rho(\mathcal{N}_i) \leq \tilde{\rho}_j$ belong to one class, the rest belong to the other class, and $\tilde{\rho}_j$ also gives a reasonable choice for η. Another possibility is to use

$$\eta = \frac{\tilde{\rho}_j + \tilde{\rho}_{j+1}}{2}. \tag{3.14}$$

We should mention that for a data set with large noise or poor data distribution, it is still very challenging to choose a suitable η since the geometry or the local dimension of the data points is generally uncertain.

4 Adaptive Bias Reduction

The local approximation error $\|x_{i_j} - (\bar{x}_i + Q_i\theta_j^{(i)})\|$ resulting from the *linear* fitting can be large or small, depending on the local curvature, unless we are dealing with a linear manifold. Varying degree of bias exists in the linear fitting, which can seriously impact the accuracy of the computed local coordinates. In this section, we focus on addressing the accuracy issue by proposing a new weighted version of LTSA alignment specifically designed to correct the bias.

We first describe the bias problem using more precise terms. Consider the set of local coordinates $\{\theta_1^{(i)}, \cdots, \theta_{k_i}^{(i)}\}$ of neighborhood $\mathcal{N}_i$. We use $\{\tau_i^*\}$ to denote the ideal global parameter vectors and $\{\tau_i\}$ the estimated ones. We measure the bias of $\theta_j^{(i)}$ as $\beta_j^{(i)} = \|\tau_{i_j}^* - (\bar{\tau}_i^* + L_i^*\theta_j^{(i)})\|$, where L_i^* is the optimal affine transformation for the local coordinate set $\{\theta_1^{(i)}, \ldots, \theta_{k_i}^{(i)}\}$, i.e.,

$$\begin{aligned}\sum_{j=1}^{k_i} \|\tau_{i_j}^* - (\bar{\tau}_i^* + L_i^*\theta_j^{(i)})\|^2 &= \min_{L_i} \sum_{j=1}^{k_i} \|\tau_{i_j}^* - (\bar{\tau}_i^* + L_i\theta_j^{(i)})\|^2 \\ &= \min_{c_i, L_i} \sum_{j=1}^{k_i} \|\tau_{i_j}^* - (c_i + L_i\theta_j^{(i)})\|^2.\end{aligned}$$

Similarly, we can also define the local error with respect to the estimates, $\alpha_j^{(i)} = \|\tau_{i_j} - (\bar{\tau}_i + L_i\theta_j^{(i)})\|$ with the optimal L_i. It is clear that the more accurate the estimates $\{\tau_i,\}$ are to the ideal $\{\tau_i^*\}$, the more closely matched are the errors $\alpha_j^{(i)}$ and $\beta_j^{(i)}$. So, good estimates $\{\tau_i\}$ should ensure that the errors $\{\alpha_j^{(i)}\}$ and $\{\beta_j^{(i)}\}$ are as similar as possible.

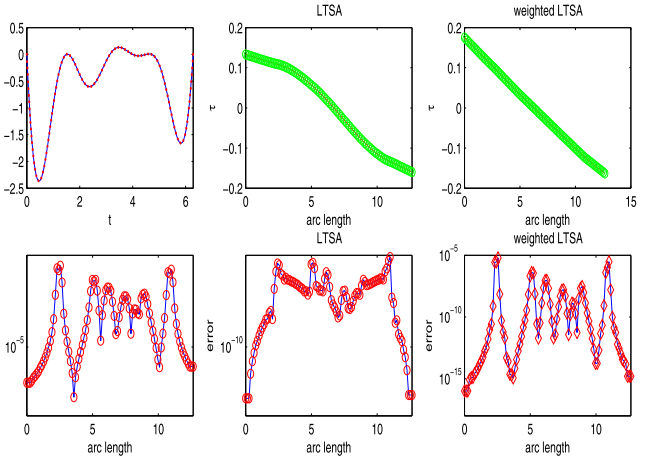

Fig. 4. Example 3. The left column plots the data points and the average local errors α_i. The original LTSA yields a contorted embedding that is plotted in the middle column, together with the local errors of linear fitting. The right column shows the excellent embedding from a modified version of LTSA and the corresponding local errors. The error curve of the modified LTSA matches the curve of α_i very well.

Observe that LTSA treats all the local errors $\alpha_j^{(i)}$ equally in the objective function for the optimization

$$\sum_{i=1}^{N} \frac{1}{k_i} \sum_{j=1}^{k_i} \|\tau_{i_j} - \bar{\tau}_i - L_i\theta_j^{(i)}\|^2 = \sum_{i=1}^{N} \frac{1}{k_i} \sum_{j=1}^{k_i} (\alpha_j^{(i)})^2.$$

Minimizing the above total errors over all feasible T tends to force the $\{\alpha_j^{(i)}\}$ to become similar to each other. This embedding does not take into account the magnitude of the $\{\beta_j^{(i)}\}$. As a result, the $\{\alpha_j^{(i)}\}$ may be very different from $\{\beta_j^{(i)}\}$ if the local curvature is large. We illustrate the phenomenon by an example.

Example 3. Consider the 1D curve $\mathcal{M} = \{(t, s) : s = g(t)\}$ embedded in a 2D plane:

$$g(t) = \frac{1}{5}(t - 1.5)(t - 4)(t - 4.5)\sin(2t), \quad t \in [0, 2\pi].$$

We sample 100 points $\{(t_i,\ g(t_i))\}$ from $\mathcal{M}$ with equal arc lengths between two points and set $x_i = [t_i,\ g(t_i)]^T + \epsilon_i$ with noise vector ϵ_i whose components are uniformly distributed in the interval $[-0.02,\ 0.02]$. Let s_i be the arc length coordinates of the samples. Then, $\beta_j^{(i)} = |s_{i_j} - \bar{s}_i - L_i^*\theta_j^{(i)}|$. For simplicity, we consider the average bias among each neighborhood, $\beta_i = \frac{1}{k_i}\sum_{j=1}^{k_i} \beta_j^{(i)}$, and the average local errors $\alpha_i = \frac{1}{k_i}\sum_{j=1}^{k_i} \alpha_j^{(i)}$ of the 1D embedding T of LTSA. The values of $\{\alpha_i\}$ are quite different from those of $\{\beta_i\}$. Consequently, there are visible deviations of the 1D embedding T of LTSA from the exact arc length coordinates; see the middle column of the top row in Fig. 4.

To confirm our key observation that a good embedding T should have local errors that match the biases of the local coordinates, i.e., $\alpha_j^{(i)} \approx \beta_j^{(i)}$, we plot the coordinates computed by a modified LTSA which imposes such a restriction on the local errors. The resulting average local errors are

shown in the right column of Fig. 4. The computed embedding exhibits good fit to the true arc length.

In order to enforce the constraints

$$\alpha_j^{(i)} \approx \beta_j^{(i)},$$

it is required to provide good estimates of the bias $\beta_j^{(i)}$ for the local coordinate $\theta_j^{(i)}$. In the next sections, we address this problem. The estimates can then be used to penalize the local errors $\alpha_j^{(i)}$ so that they can match $\beta_j^{(i)}$ as much as possible, leading to a modified model of the original LTSA—a weighted version of LTSA.

4.1 Bias Estimation for Local Coordinates

The bias $\beta_j^{(i)}$ can be estimated by the squared error of the linear fitting $\phi_j^{(i)} = \|x_{i_j} - \bar{x}_i - Q_i\theta_j^{(i)}\|$. To show this, let us assume that $\mathcal{M} = f(\Omega)$ is a parameterized manifold with a smooth mapping $f:\ \Omega \subset \mathcal{R}^d \to \mathcal{M} \subset \mathcal{R}^n$. For a finite sample $x_i = f(\tau_i)$, $i = 1, \ldots, N$, with $\{\tau_1, \ldots, \tau_N\}$ the global parameter vectors. To estimate the bias $\beta_j^{(i)} = \|\tau_{i_j} - \bar{\tau}_i - L_i^*\theta_j^{(i)}\|$, let us consider the second-order Taylor expansion of f:

$$\begin{aligned}\hat{x} - x = J_\tau(\hat{\tau} - \tau) + \frac{1}{2}H_\tau(\hat{\tau} - \tau, \hat{\tau} - \tau)\\ + o(\|\hat{\tau} - \tau\|^2),\end{aligned} \tag{4.15}$$

where H_τ is the Hessian tensor of f at τ. Assume that we have a linear subspace $\mathrm{span}(Q_\tau)$ that approximates the tangent space $\mathrm{span}(J_\tau)$ well. Denoting $\theta_x(\hat{x}) = Q_\tau^T(\hat{x} - x)$ and $P_\tau = Q_\tau^T J_\tau$, we see that

$$P_\tau^{-1}Q_\tau^T(\hat{x} - x - J_\tau(\hat{\tau} - \tau)) = -\big(\hat{\tau} - \tau - P_\tau^{-1}\theta_x(\hat{x})\big).$$

Thus, for a neighbor τ of $\hat{\tau}$,

$$\begin{aligned}\|\hat{\tau} - \tau - P_\tau^{-1}\theta_x(\hat{x})\| &= \|P_\tau^{-1}Q_\tau^T(\hat{x} - x - J_\tau(\hat{\tau} - \tau))\|\\ &\le \|P_\tau^{-1}\|\|\hat{x} - x - J_\tau(\hat{\tau} - \tau)\|\\ &\approx \gamma_\tau\|\hat{x} - x - Q_\tau\theta_x(\hat{x})\|,\end{aligned} \tag{4.16}$$

where $\gamma_\tau = \|P_\tau^{-1}\|$. One can verify that $\gamma_\tau \le \frac{\|J_\tau^\dagger\|}{\cos\vartheta_\tau}$ with ϑ_τ the angle between the tangent space $\mathrm{span}(J_\tau)$ and its approximation $\mathrm{span}(Q_\tau)$,[6] where $J_r^\dagger$ is the Moore-Penrose generalized inverse of J_r. So, if the tangent space is well approximated, then $\cos\vartheta_\tau \approx 1$ and $\gamma_\tau \approx \|J_\tau^\dagger\|$. The inequality above clearly indicates that the ideal embedding should have the local errors matching the errors of the local linear fitting. Note that the linear fitting error,

$$\|\hat{x} - x - Q_\tau\theta_x(\hat{x})\| \approx \|\hat{x} - x - J_\tau(\hat{\tau} - \tau)\|,$$

depends on the local curvatures and the squares of the size of the local coordinates, according to (4.15).

For a finite sample possibly with noise, it is better to replace x by the local mean $\bar{x}$. We set $\hat{\tau} = \tau_{i_j}$, $\tau = \bar{\tau}_i$, $J_\tau = J_{\bar{\tau}_i}$, then $\theta_{x_i}(x_{i_j}) = \theta_j^{(i)}$, and (4.16) reads

$$\big\|\tau_{i_j} - \bar{\tau}_i - P_{\bar{\tau}_i}^{-1}\theta_j^{(i)}\big\| \le \gamma_{\bar{\tau}_i}\|x_{i_j} - \bar{x}_i - J_{\tau_i}(\tau_{i_j} - \bar{\tau}_i)\|. \tag{4.17}$$

6. It can be verified by SVD of J_τ and the definition of angle between two subspaces. Refer to [9, Section 2.6].

Roughly speaking,

$$\beta_j^{(i)} = \big\|\tau_{i_j} - \bar{\tau}_i - L_i^*\theta_j^{(i)}\big\| \approx \big\|\tau_{i_j} - \bar{\tau}_i - P_{\bar{\tau}_i}^{-1}\theta_j^{(i)}\big\|,$$

and $\|x_{i_j} - \bar{x}_i - J_{\bar{\tau}_i}(\tau_{i_j} - \bar{\tau}_i)\|$ can be well estimated by $\phi_j^{(i)} = \|x_{i_j} - \bar{x}_i - Q_i\theta_j^{(i)}\|$. Thus, we see that $\beta_j^{(i)} \approx \gamma_{\bar{\tau}_i}\phi_j^{(i)}$, i.e., the values $\{\phi_j^{(i)}\}$ should be similar to those of $\{\beta_j^{(i)}\}$.

4.2 Weighted LTSA

Motivated by the analysis above, and noting that $\|J_\tau^\dagger\|$ is same in magnitude generally, we propose the modified version of LTSA that solves the following modified minimization problem:

$$\min_T \sum_i \frac{1}{k_i}\min_{c_i, L_i}\sum_{j=1}^{k_i}\Big(w_j^{(i)}\|\tau_{i_j} - (c_i + L_i\theta_j^{(i)}\|\Big)^2, \tag{4.18}$$

where

$$w_j^{(i)} = \big(\phi_j^{(i)} + \delta\big)^{-1} = \big(\big\|x_{i_j} - \bar{x}_i - Q_i\theta_j^{(i)}\big\| + \delta\big)^{-1}, \tag{4.19}$$

$\delta > 0$ is a small regularization constant in case $\phi_j^{(i)} = 0$. We generally set $\delta = 10^{-3}$ in our numerical experiments.

The alignment matrix corresponding to the weighted LTSA has a similar construction as that for LTSA. Since

$$\sum_{j=1}^{k_i}\big(w_j^{(i)}\big\|\tau_{i_j} - \big(c_i + L_i\theta_j^{(i)}\big)\big\|\big)^2 = \|T_iD_i - (c_iw_i^T + L_i\Theta_iD_i)\|_F^2,$$

where $D_i = \mathrm{diag}(w_i)$, $w_i = (w_1^{(i)}, \ldots, w_{k_i}^{(i)})^T$, its minimum is achieved when $c_iw_i^T + L_i\Theta_iD_i$ equals the orthogonal projection of T_iD_i in row, i.e., $c_iw_i^T + L_i\Theta_iD_i = T_iD_iP_i$, where P_i is the orthogonal projection onto the column space of $[w_i, D_i\Theta_i^T] = D_i[e, \Theta_i^T]$. Thus,

$$\begin{aligned}&\min_{c_i, L_i}\|T_iD_i - \big(c_iw_i^T + L_i\Theta_iD_i\big)\|_F^2\\ &= \|T_iD_i(I - P_i)\|_2^2 = \mathrm{trace}(T_iD_i\Phi_i^wD_iT_i^T),\end{aligned}$$

where $\Phi_i^w = I - P_i$, whose null space is $\mathrm{span}(D_i[e, \Theta_i^T])$. Denoting the alignment matrix

$$\Phi^w = \sum_i \frac{1}{k_i}S_iD_i\Phi_i^wD_iS_i^T, \tag{4.20}$$

the optimization problem (4.18) can be represented as

$$\min_T \sum_i \frac{1}{k_i}\|T_iD_i\Phi_i^w\|_F^2 = \min_T \mathrm{trace}(T\Phi^wT^T). \tag{4.21}$$

Imposing the normalization condition $TT^T = I$, the unique solution to (4.21) is given by the d eigenvectors corresponding to the second to $(d+1)$st smallest eigenvalues of matrix Φ^w. The above discussion gives a weighted version of LTSA.

4.3 Normalized Weights

For noisy data sets, it is often the case that there are outliers or some samples have relatively large noise. It is known that the local PCA cannot give an acceptable estimate to the tangent space at a point with large noise. Let us consider an outlier whose neighbors are relatively clustered together. The PCA errors are then relatively small and hence the corresponding local weights $w_j^{(i)}$ in (4.19) are very large. The left panel of

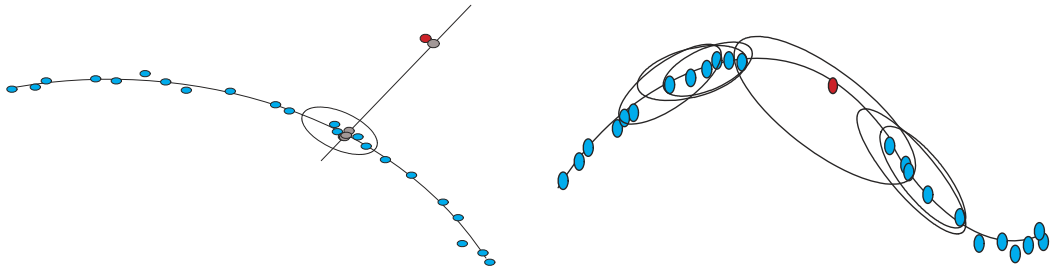

Fig. 5. Left: An outlier (red point) may result in a wrong linear fitting (line) and wrong projections (gray points) of its neighbors (blue points inclosed by an ellipse). Right: An isolator (red point) is seldom selected as a neighbor of other points. Some neighbor sets of points nearby the isolator are marked by ellipses.

Fig. 5 shows such an example. Meanwhile, the local PCA coordinates have large deviations to the true coordinates. Thus, the large local weights compel the resulting embedding to have very small errors corresponding to the wrong coordinates within an affine transformation, resulting in a wrong embedding eventually. Obviously, decreasing those local weights can improve the results significantly.

For the case when the data sampling is far from uniform, there can also be isolated samples and very few neighbor sets contain these isolated points as neighbors (see an illustration shown in the right of Fig. 5). To clearly show it, let us consider a point x_ℓ and let K_ℓ be the index set of those neighborhoods $\mathcal{N}_i$ containing x_ℓ as a neighbor of x_i, i.e., if $i \in K_\ell$, then $x_\ell \in \mathcal{N}_i$. x_ℓ is also denoted as x_{i_j} in the neighbor set $\mathcal{N}_i$, where $j = j(\ell, i)$ is a local index depending on ℓ and i. The sum in (4.18) now reads

$$\sum_{\ell} \sum_{i \in K_\ell} \left(w^{(i)}_{j(\ell,i)} \left\| \tau_\ell - \left(c_i^* + L_i^* \theta^{(i)}_{j(\ell,i)}\right)\right\|\right)^2, \tag{4.22}$$

where c_i^* and L_i^* are the optimal solution to

$$\min_{c_i, L_i} \sum_{j=1}^{k_i} \left(w_j^{(i)} \left\| \tau_{i_j} - \left(c_i + L_i \theta_j^{(i)}\right)\right\|\right)^2.$$

Clearly, if x_ℓ has sparse neighbors, K_ℓ is relatively small, and hence the related restrictions for embedding is much weaker than for others which have large K_ℓs.

Taking into account the observations above, we modify our weight formula by normalizing the weights corresponding to each K_ℓ. To simplify discussion, we change, in this section only, the notation $w_j^{(i)}$ with local index $j = j(\ell, i)$ corresponding to x_{i_j} to $w_\ell^{(i)}$ with global index ℓ. We view $w_\ell^{(i)}$ as the weights for the affine reconstruction errors of τ_ℓ with respect to the neighborhood $\mathcal{N}_i$, and normalize the related weights $w_\ell^{(i)}$ among the indices $i \in K_\ell$ as

$$w_\ell^{(i)} \leftarrow \frac{w_\ell^{(i)}}{\sum_{i \in K_\ell} w_\ell^{(i)}} = \frac{w_\ell^{(i)}}{s_\ell}, \tag{4.23}$$

where $s_\ell = \sum_{i \in K_\ell} w_\ell^{(i)}$. If x_ℓ is an outlier, its local weights can be significantly decreased, whether taking x_ℓ as a neighbor of itself or of other points. Otherwise, the normalization does not change the magnitude of its original local weights. For isolated points, normalizing the weights can increase the information convection through those points.

4.4 Adaptive LTSA

The adaptively weighted LTSA can be used together with variant neighborhood selections such as k-NN or the adaptive neighborhood selection strategy (contraction and expansion). We refer to the weighted LTSA with the normalized adaptive weights as *adaptive* LTSA and summarize it as follows:

Algorithm. Adaptive LTSA.

1. ADAPTIVE NEIGHBORHOOD SELECTION. Determine neighborhoods $\mathcal{N}_i = \{x_{i_1}, \ldots, x_{i_{k_i}}\}$ of x_i, $i = 1, \ldots, N$, using the neighborhood contraction/expansion methods.
2. OPTIMAL LOCAL LINEAR FITTING. For each neighborhood $\mathcal{N}_i$, compute the SVD $Q_i \Sigma_i V_i^T$ of $X_i - \bar{x}_i e_{k_i}^T$. Set $\theta_j^{(i)} = Q_i(:, 1 : d)^T (x_{i_j} - \bar{x}_i)$ corresponding to the first d largest singular values, and $w_j^{(i)} = \|Q_i(:, d+1 : k_i)^T (x_{i_j} - \bar{x}_i)\|$ for $x_{i_j} \in \mathcal{N}_i$, $i = 1, \cdots, N$.
3. NORMALIZING LOCAL ADAPTIVE WEIGHTS. Update $w_j^{(i)}$ by (4.23) for $x_{i_j} \in \mathcal{N}_i$, $i = 1, \cdots, N$.
4. EMBEDDING. Construct the weighted align matrix Φ^w and compute its $d+1$ smallest eigenvectors $u_1, \cdots, u_{d+1}$. Set the global coordinate matrix $T = [u_2, \ldots, u_{d+1}]^T$ corresponding to the 2nd to $d+1$st smallest eigenvalues.

5 Experimental Results

In this section, we present several numerical examples to show the improvements of the three manifold learning methods Isomap, LLE, and LTSA on one toy data set and seven real-world data sets, when the adaptive methods for neighborhood selection as well as bias correction are used. Our goal is to test the methods using a variety of data sets possessing certain challenging properties that tend to break less robust manifold learning algorithms. The data sets include sparse samples from a smooth manifold (toy data), noisy data with relatively identifiable dimensionality in a high-dimensional space (face images), data with a few samples from multiple classes (images of rotary objects), data in multiple classes having noisy local geometry (handwritten digits), and data of frequency features with uncertain dimensionality (speech signals). Three classifiers, the Nearest Neighbor, the Nearest Feature Line (NFL) [15],

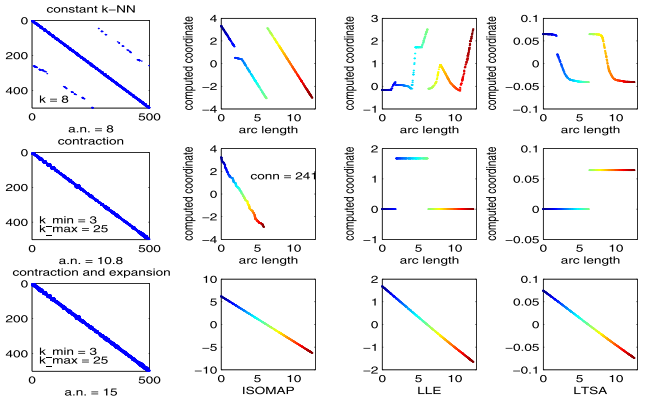

Fig. 6. Comparison of three neighborhood selection methods (k-NN, contraction, and contraction-plus-extension). The left column shows the connection matrices of neighborhoods and the other three columns are the corresponding 1D embeddings of Isomap, LLE, and LTSA. k-NN yields a wrong neighborhood graph that results in bad embeddings of Isomap, LLE, and LTSA. The contraction method gives true but overcautious neighborhoods that lose the necessary connection and the embeddings are breakdown. The bottom row shows excellent improvement on the resulting embeddings by the extension strategy.

and the Support Vector Machine (SVM) [5], are used to show the improvements of our proposed adaptive manifold learning methods for dimensionality reduction used for the classification problems. NFL uses the distance of a testing point to a feature line that connects two pints in the same class. We also use SVM and Gaussian Mixture Model (GMM) [21] in the speech example. The parameter η is always set as (3.14) with $k_0 = k_{\max}$ unless otherwise specified in the experiments.

5.1 A Curve in 3D Space

We show how the neighborhood contraction/expansion approaches improve the neighborhood graphs and the embedding results of Isomap, LLE, and LTSA. The test data set is sampled from a 1D manifold as in Example 2, but noise is added as follows: $x_i = [\sin(t_i), \cos(t_i), 0.02t_i]^T + \epsilon_i$, $i = 1, \ldots, N = 500$, where the elements of noise vectors ϵ_i are uniformly distributed in $[-0.01, 0.01]$.

Because of the data sparsity and the presence of noise, partial neighbor sets determined by k-NN contain wrong neighbors, which mainly result in failure of Isomap, LLE, and LTSA. The top row of Fig. 6 shows the adjacency matrix G of the neighborhoods of k-NN with $k = 8$ and the 1D embeddings of these three learning algorithms. Here, G is defined by $G(i, j) = 1$ for $j \in I_i$ and $G(i, j) = 0$ for $j \notin I_i$. The neighborhood contraction can improve the neighborhood graph, but the required neighborhood overlapping is also reduced, which also results in "breaking" of the three methods. See the second row of the figure. The expansion approach strengthens the true connection between the adjoining neighborhood. The left of the last row shows the improved neighborhood graph. Using the resulting neighborhood graph, Isomap, LLE, and LTSA give excellent embeddings.

5.2 Stanford Face Images

The face image data set [24] consists of 698 64-by-64 images of a statue and three physically meaningful degrees of freedom can be identified: two pose parameters (left-right and up-down) and one lighting parameter. We consider 3D projections of Isomap and LTSA with k-NN or adaptive neighbor strategies. Each image is converted to an $m = 4{,}096$-dimensional image vector in this experiment.

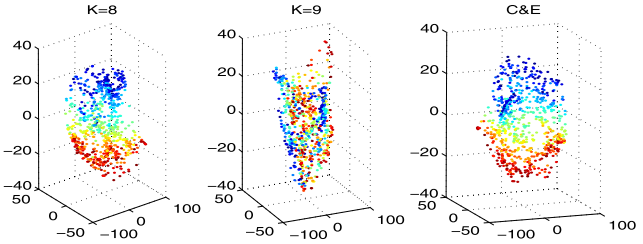

Fig. 7. The Isomap embeddings with 8-NN or 9-NN plotted in the left or the middle panels are quite different. Isomap using adaptive neighborhood (C&E) gives a better embedding. The right panel shows the computed 3D coordinates with $k_{\min} = d + 1$, $k_{\max} = 20$.

It is reported that Isomap can compute the parameter vectors very well if the neighborhood size is set to be $k = 8$ [24]. The 3D embedding has a reasonable distribution and a cube-like shape. However, the Isomap embedding is very sensitive to k; the nonlinear projection will be very bad if one slightly increases the neighborhood size (see the middle panel in Fig. 7 for $k = 9$ or Table 1 for $k = 9, 10$). The adaptive method can select suitable neighbor sets for Isomap. The right panel of Fig. 7 shows the embedding of Isomap with the adaptive neighborhoods ($k_{\min} = 4$ and $k_{\max} = 20$), which looks better than or at least as good as the best one of k-NN achieved at $k = 8$.

The improvement can be measured by the relative affine error of the computed coordinates defined as

$$\epsilon(T) = \min_{c,W} \frac{\|T^* - (ce^T + WT)\|_F}{\|T^*\|_F},$$

where T^* denotes the matrix of the true parameter vectors (two pose parameters and one light parameter) and T the matrix of computed parameter vectors. Table 1 given below shows the affine errors of computed 3D embeddings by Isomap using k-NN neighborhoods with different k and the adaptive neighborhoods, respectively, where C&E is short for the contraction and expansion algorithms. The affine error corresponding to the adaptive neighborhoods is smaller than the lowest one corresponding to the k-NN strategy.

5.3 The coil20 Data Set

Differently from the previous data sets, the data set `coil20` [19] has a small number of samples and many classes (20 classes). Each class contains 72 samples of a rotary object, and the data set has 1,440 points in total. In this example, we demonstrate the improvement of the weighted strategy in LTSA for the NN classification. Low-dimensional nonlinear projections with different setting of dimension d are

TABLE 1
Relative Affine Errors of Isomap Embeddings for the Face Images with Different Neighborhoods

k	6	7	8	9	10	C&E
$\epsilon(T)$	0.0839	0.0818	0.0797	0.2399	0.2388	**0.0774**

TABLE 2
Average Errors (Percent) of NN Classifier on Projections of LTSA and WLTSA of coil20

d	method	k =13	14	15	16	17	18
7	LTSA	17.1	14.5	13.5	15.8	16.0	16.9
	WLTSA	14.0	11.3	12.0	10.9	9.7	14.3
8	LTSA	15.7	13.5	14.0	13.3	14.5	13.5
	WLTSA	14.5	12.8	12.7	10.6	9.9	12.0
9	LTSA	13.5	13.2	11.2	10.5	12.9	14.4
	WLTSA	12.3	11.3	13.8	11.6	10.9	12.8

generated by LTSA and the weighted LTSA using k-NN neighborhoods. Then, we applied the NN classifier on the projected data to recognize the 20 groups. The training set is formed by 200 points that are randomly selected (10 points belonging to each class) from the data set and the remaining data points are used as testing set. We repeated with 100 runs for the NN classifier on both of the projected data set by LTSA or the weighted LTSA with each choice of k and d. Table 2 lists the average errors in percentage of NN classifier on the projected points of LTSA and the weighted LTSA. The improvement obtained by the weighted strategy is significant: For almost each choice of d and k, the weighted LTSA performed better than LTSA in this example and the classification error can be reduced up to 40 percent. We remark that the weighted strategy may fail if the low-dimensional structure is not correctly captured in the neighbor set (for relatively large d and small k).

5.4 Handwritten Digits

We used two data sets semeion and mfeat of handwritten digits ("0"-"9").[7] The data set semeion contains 1,593 binary vectors of dimension 256 that are generated from 1,593 images of handwritten digits written by about 80 people. Because of large noise, LTSA needs a relatively large neighborhood to identify the local geometry for a fixed dimensionality d, while LLE works on relatively small neighborhoods. However, the NN classifier still produces larger classification error when applied to the LLE projection, compared with the error on the LTSA projection. The weighted version of LTSA can further improve the LTSA projection. In Table 3, we list the errors in percentage for the NN classifier on the low-dimensional projections of computed by LLE, LTSA, and wLTSA. The errors are averaged over 100 runs each a random 1:1 training and testing split of the data set. For each dimension d used, wLTSA always achieved the smallest classification error. The classification error of NN on the original data is 10.90 percent.

The data set mfeat consists of 2,000 handwritten digits. Each of the 10 digits has 200 examples. We used the data set represented by the pixel features using 240 pixel averages in 2×3 windows, and removed six duplicate data points. In this experiment, we compared the effectiveness of the adaptive manifold projections in the context of SVM and NN classifications. The multiclass classification problem is converted to 10 two-class problems by the one-against-the-rest split for SVM, and the first 25 percent points in each class are used as the training set and the remaining one as the testing set. Table 4 shows the results of the SVM classification with Gaussian kernel and the NN classification on the original data and the 5D projected data obtained by LLE, LTSA, and WLTSA using k-NN neighborhoods with the best choice of k ranging from $k_{\min} = 7$ to $k_{\max} = 23$ (for SVM, $k = 7$ in LLE and $k = 13$ in LTSA or WLTSA; for NN, $k = 10$ in LLE and $k = 9$ in LTSA or WLTSA), or the adaptively selected neighborhoods with the η determined by (3.14). Compared with the k-NN neighborhoods with the best setting of k in the three manifold learning methods, the adaptive neighborhood selection can produce better results in both of the SVM and NN classifications. In fact, the SVM classifier on the adaptively computed projections achieved higher accuracy than that using the original data. The NN classifier produced slightly larger errors than the SVM classifier but it is very fast computationally.

7. These two data sets can be downloaded from http://archive.ics.uci.edu/ml/datasets/.

5.5 Yale Face Images

The data set used in this experiment is generated from the extended Yale Face Database B which contains 2,431 images from 38 individuals with a fixed pose under 64 different illumination conditions [8], [14]. The original images are of size 192×168. Because of the poor performance of the existing manifold learning algorithms on the original data, we preprocessed the data as follows: First, each image is resized to 32×28 for reducing the computation complexity. Then, each pixel of the small image is represented by an 8D binary vector whose components are set as follows: If the pixel value is larger than a neighboring pixel in one of the eight directions (left-up, up, right-up, and so on), then the corresponding component is 1. Otherwise, it is set to be 0. This preprocessing can reduce the influence of luminance to some extent. The resulting data are a 7,168-dimensional binary vector. As a result, we have 2,431 binary vectors in 7,168-dimensional space.

We first compared the errors of the NN classifier on the nonlinear low-dimensional projections generated by Isomap, LLE, LTSA, and WLTSA with the k-NN neighborhoods or the adaptively selected neighborhoods, using a 1:3 training and testing split of the data points (16 training points and about 48 testing points) in each class. We repeated this experiment in 100 random splits and checked the average error. The classification accuracy is sensitive to the neighborhood size if the k-NN used in Isomap, LTSA, or WLTSA. LLE seems to be relatively less sensitive to k, though it does not achieve the smallest error produced by LTSA or WLTSA with a well-chosen value of k. Table 5 shows the classification errors of the NN classifier on the nonlinear projections of LLE, LTSA, and WLTSA with k ranging from $d+3$ to $d+9$, relatively better than other settings of k, for $d = 8, 9, 10$. We did not list the errors of the NN classifier on the Isomap projection because of its poor performance (the errors are larger than 13 percent for the settings of parameters k and d in Table 5). The weighted strategy can improve the LTSA significantly. For all the settings except $d = 8$ and $k = d+9$, the weighted strategy always gives a better embedding of LTSA. In the right column of Table 5, we also list the classification errors on the LLE projection ($k_{\min} = d+1,\ k_{\max} = d+9$) and the LTSA or WLTSA projection ($k_{\min} = d+2,\ k_{\max} = 25$). The adaptive neighborhood selection method significantly

TABLE 3
Average Errors (Percent) of NN Classifier on Database semeion

	LLE				LTSA				wLTSA			
d	k=10	15	20	25	35	40	45	50	35	40	45	50
12	12.47	12.58	13.52	13.63	11.96	12.03	11.89	12.57	11.10	11.16	**11.09**	11.19
14	12.63	12.01	12.39	13.36	11.75	10.27	10.78	11.03	10.42	**10.12**	10.70	10.92
16	11.67	11.82	12.09	13.17	10.59	11.37	10.62	10.53	**9.89**	9.93	10.66	10.28
18	10.85	11.57	11.98	12.99	11.07	10.38	10.25	10.08	10.15	**9.87**	10.71	10.12
20	10.23	11.02	10.87	12.46	9.93	10.17	10.40	10.41	10.46	10.00	10.11	**9.85**

improved LTSA and WLTSA in this example. Practically, the error of NN classifier on the LTSA projection can be reduced by 45 percent if we use both of the weighting strategy and the adaptive neighborhood selection in LTSA. For LLE, the adaptive neighborhood method worked well for $d = 8$ and slightly worse for $d = 9$ and $d = 10$. Notice that the NN classifier produces 5.05 percent error on the original data.

We also tested the SVM classifier on the original data set and the six projected data sets of LLE, LTSA, and WLTSA with $d = 8$ and the k-NN neighborhoods or the adaptively selected neighborhoods. We converted the multiclass classification problem to a set of two-class problems in the way of one-against-the-rest for SVM. In this experiment, CPU time for SVM is on the order of 10^4 seconds since the data have 38 classes. We are not able to repeat many random training-testing splits as in the NN classification: We used the first 16 points in each class data and others as testing points. The size of the training set is the same as that in the NN classification. We used the Gaussian kernel in SVM because the low-dimensional projection may not preserve the linear separability of the original data (if it is linearly separable). The SVM classifier has 9.49 percent error on the original data set with the CPU time of about 3.5 hours. The low-dimensional projections of LLE, LTSA, or WLTSA can reduce the CPU time of SVM by half. The projections of LLE and LTSA using k-NN strategy did not produce better results of SVM. WLTSA improved LTSA but its SVM error is also larger than the SVM error on the original data. In this experiment, the adaptive neighboring also performs excellently. The SVM classifier achieved a very low error when it is applied on the projection of the adaptive LTSA: About 38 percent of errors of SVM on the original data are reduced and about half time of running is saved. See Table 6 for the details.

5.6 Speech Data

This example uses the Harvard-Haskins database of Regularly Timed Speech.[8] The database contains 264 16-bit wav files from six speakers (three male and three female); each speaker uttered sequences of syllables in four different utterances, respectively. Every speech signal is split into several overlapped frames, each of which is of length 10,000 (a piece of 1 second speech) with one half of frame overlapping with the next frame. Then, each frame is represented by suitable features to highlight the speech characters. In this experiment, we used two kinds of features that are commonly used in speech processing. One is the spectrogram obtained by the Short-Time Fourier Transform (STFT). As is standard in speech processing, logarithms of the Fourier coefficients of a frame form a data point. The resulting data set consists of 2,766 vectors in a 5,001-dimensional space, belonging to six classes corresponding to the different speakers. The other kind of features are the Mel-Frequency Cepstral Coefficients (MFCCs) [20]. In this example, each frame has 924 MFCCs which form a vector. In total, we have 2,766 MFCC vectors of dimension 924. Three classifiers, SVM, GMM, and NN, are used on the STFT or MFCC feature vectors or their low-dimensional projections by LLE, LTSA, and the weighted LTSA for the problem of speaker recognition. In NN, we use the distance of a testing signal to each training point, i.e., the sum of euclidean distances of points generated from the signal to the training point.

First, we show the effectiveness of the approaches of the adaptive neighborhood selection and the weighting strategy on these STFT vectors. We randomly select only one signal from each speaker and the STFT vectors corresponding to the selected files form the training set. The remaining vectors of other speech files are the testing points. The

TABLE 4
Errors and CPU Times of the SVM and NN Classifiers on Database mfeat

	projection (d=5)	LLE		LTSA		WLTSA		original
		k-nn	C&E	k-nn	C&E	k-nn	C&E	data
SVM	error(%)	3.75	3.48	4.22	**3.21**	4.08	3.28	3.55
	CPU(s)	986	1000	966	977	974	966	1273
NN	error(%)	4.35	**3.55**	4.62	3.61	4.69	3.88	3.61
	CPU(s)	0.28	0.28	0.28	0.28	0.28	0.28	1.12

TABLE 5
Average Errors (Percent) of NN Classifier for the Extended Yale Face Database B

d	k	d+3	d+4	d+5	d+6	d+7	d+8	d+9	C&E
8	LLE	7.86	7.44	7.27	7.81	7.70	7.77	7.93	**5.93**
	LTSA	8.48	5.62	6.83	8.27	7.84	7.79	7.71	**3.88**
	WLTSA	8.16	4.65	4.72	5.85	6.04	7.52	8.01	**3.01**
9	LLE	**6.49**	7.16	7.46	7.64	8.11	8.01	7.25	6.85
	LTSA	8.29	6.71	6.32	6.67	5.97	6.70	7.52	**4.40**
	WLTSA	7.33	4.56	5.96	5.22	4.84	5.75	6.14	**3.22**
10	LLE	**6.24**	6.45	7.07	7.38	7.37	6.79	7.94	6.59
	LTSA	11.19	6.86	6.28	6.34	5.44	5.99	7.36	**4.18**
	WLTSA	8.41	6.72	4.74	4.65	4.46	4.09	4.79	**3.13**

TABLE 6
Errors of SVM Classifier for the Extended Yale Face Database B ($d = 8$)

	k-NN neighborhoods			adaptive neighborhoods	
	k	error(%)	cpu(s)	error(%)	cpu(s)
LLE	13	12.07	6723.5	8.89	6960.5
LTSA	12	11.57	6648.7	9.49	6374.8
WLTSA	12	9.54	6414.8	**5.92**	6435.4
original data	error = 9.49%, cpu time = 12362 seconds				

8. http://vesicle.nsi.edu/users/patel/speech_database.html.

TABLE 7
Average Errors (Percent) of NN, SVM, and GMM Based on STFTs (Single Signal of Each Speaker in Training)

NN							SVM	GMM
k	10	15	20	25	C&E	original data	5.74	39.56
LLE	17.69	23.16	26.45	6.95	2.23	LLE+C&E	1.90	14.42
LTSA	5.38	2.35	2.80	2.28	1.31	LTSA+C&E	1.29	11.82
wLTSA	4.60	2.20	2.51	2.01	1.42	wLTSA+C&E	1.32	10.21

recognition rate of NN classifier based on the LLE projection is not satisfactory if we simply use k-NN neighbors, even if the fixed size k of neighborhoods is carefully selected. The LTSA projection performs much better than LLE and the weighted version of LTSA gives smaller recognition errors than LTSA when the same neighbor sets are used. The adaptive neighborhood strategy can further improve the NN recognition on the LLE, LTSA, or wLTSA projections. The left of Table 7 lists the average recognition errors in percentage of the NN classifier based on the 2D projections of LLE, LTSA, and wLTSA using k-NN with several different values of k or the adaptive neighborhoods with $k_{\max} = 25$ based on 100 repeated experiments. We did not show the results based on Isomap projections since the recognition error is much bigger than the LTSA results. The improvement of the adaptive neighborhood strategy on the Isomap projection is limited in this example. PCA projection fails to improve the NN recognition rate on the unreduced data. As a comparison, both SVM and GMM have larger recognition errors. The nonlinear dimensionality reductions can help to improve the recognition performance for SVM and GMM in this experiment. In the right of Table 7, we also give the average errors of SVM with Gaussian kernel and GMM on the original STFT vectors or the nonlinear projections with the adaptive neighbors.

The recognition performance of SVM and GMM on the STFT features increases if more signals of each speaker are available for training, while the nonlinear projections cannot further improve the recognitions of SVM or GMM. However, NN, SVM and GMM still give very low errors on the nonlinear projection with the adaptive neighborhoods. If the adaptive neighborhoods are used, these 2D projections are also suitable for SVM and GMM. See the average errors listed in Table 8, where first column n_{sig} is the number of signals of each speaker in training. GMM and NN were repeated 100 times for each case. SVM with Gaussian kernel was repeated 100, 40, 40, 16, and 10 times for $n_{\text{sig}} = 1, 3, 5, 10, 15$, respectively, since SVM costs much time if the data scale is large. SVM with linear kernel performs excellently if n_{sig} is not small. For example, the errors are $3.39, 0.88, 0.22, 0$ for $n_{\text{sig}} = 1, 3, 5, 10$, respectively.

We also tested the MFCC features. Unfortunately, SVM and NN cannot give reasonable recognition results on MFCC vectors in this speech example, even if multiple signals of each speakers are used in training. Although GMM performs much better than SVM and NN, the errors of GMM (62.60, 17.52, 7.69, 3.44, and 2.06 corresponding to $n_{\text{sig}} = 1, 3, 5, 10, 15$) are relatively large.

6 Concluding Remarks

In this paper, we developed algorithms that address two key issues in manifold learning: 1) the adaptive selection of the neighborhood sizes through neighborhood contraction and expansion and 2) the adaptive bias reduction in embedding by weighting local affine errors in the embedding of the manifold. The adaptive neighborhood selection methods can be used for other neighborhood-based manifold learning methods, while the second improvement is specially designed for LTSA. Though the theoretical analysis is given for ideal smooth and locally isometric manifolds, the proposed adaptive neighborhood selection methods and the modified weighting model of LTSA also work for noisy data sets well. For a data set with small noises, the modified LTSA with NN neighborhoods works as well as the modified LTSA with adaptively selected neighborhoods. However, the adaptive neighborhood selection method is more robust for noisy data.

The adaptive neighborhood selection method and the adaptive bias reduction model of LTSA need low-dimensional geometric structure of manifolds or data sets. For those data sets (such as those in classification) whose local low-dimensional geometric structure is blurry or difficult to determine via a few neighbors from a set of sample points, the improvement by the adaptive neighborhood selection approaches may be less significant. However, the normalizing and weighting strategies still work well. We expect this strategy to improve other existing methods for nonlinear dimensionality reduction. On the other hand, it is very complicated to determine a good embedding when a noisy data set is sampled from a manifold with variable curvatures and the noises are relatively large; curvatures will be obscured if we focus on reducing noise, or inversely, curvatures may be estimated incorrectly if we ignore noises. This is a topic that certainly deserves further investigation.

Acknowledgments

The work of Zhenye Zhang was supported in part by NSFC projects 10771194 and 11071218, and National Basic

TABLE 8
Average Errors (in Percent) of GMM, SVM, and NN on STFT Features Using Multiple Signals in Training

n_{sig}	GMM				SVM				NN			
	original data	projected data			original data	projected data			original data	projected data		
		LLE	LTSA	wLTSA		LLE	LTSA	wLTSA		LLE	LTSA	wLTSA
1	39.56	14.42	11.82	10.21	5.74	1.90	1.29	1.32	4.06	2.23	**1.31**	1.42
3	7.32	7.50	7.45	8.77	1.56	1.41	1.46	1.54	1.74	**1.38**	1.52	1.58
5	2.41	5.93	6.33	7.49	**1.14**	1.17	1.72	1.41	1.21	1.40	1.48	1.44
10	1.15	3.29	3.70	4.36	1.12	**0.80**	1.69	1.09	0.89	0.95	1.32	134
15	0.99	2.43	2.24	2.40	0.68	**0.59**	0.77	0.83	0.70	1.03	1.31	1.20

Research Program of China (973 Program) 2009CB320804, the work of Jing Wang was supported by NSFC for Youth 10901062 and NSF of Fujian Province 2010J01336, and the work of Hongyuan Zha was supported by US National Science Foundation (NSF) grants DMS-0311800, CCF-0305879, and DMS-0736328. A preliminary version of a subset of the results reported in this paper was published without proof in [28].

REFERENCES

[1] M.D. Abramoff, P.J. Magelhaes, and S.J. Ram, "Image Processing with ImageJ," *Biophotonics Int'l,* vol. 11, no. 7, pp. 36-42, 2004.
[2] M. Brand, "Charting a Manifold," *Advances in Neural Information Processing Systems,* S. Becker, S. Thrun, and K. Obermayer, eds., vol. 15, pp. 961-968, MIT Press, 2003.
[3] H. Bunke and K. Riesen, "Graph Classification Based on Dissimilarity Space Embedding," *Proc. Joint IAPR Int'l Workshop Structural, Syntactic, and Statistical Pattern Recognition,* pp. 996-1007, 2008.
[4] M. do Carmo, *Differential Geometry of Curves and Surfaces.* Prentice-Hall, 1976.
[5] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning,* vol. 20, no. 3, pp. 273-297, 1995.
[6] D. Donoho and C. Grimes, "Hessian Eigenmaps: New Locally Linear Embedding Techniques for High-Dimensional Data," *Proc. Nat'l Academy of Sciences USA,* vol. 100, pp. 5591-5596, 2003.
[7] A. Elgammal and C. Lee, "Inferring 3D Body Pose from Silhouettes Using Activity Manifold Learning," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2004.
[8] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman, "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 6, pp. 643-660, June 2001.
[9] G.H. Golub and C.F. Van Loan, *Matrix Computations,* third ed. Johns Hopkins Univ. Press. 1996.
[10] G. Guo, Y. Fu, C. Dyer, and T. Huang, "Image-Based Human Age Estimation by Manifold Learning and Locally Adjusted Robust Regression," *IEEE Trans. Image Processing,* vol. 17, no. 7, pp. 1178-1188, July 2008.
[11] A.J. Izenman, *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning.* Series: Springer Texts in Statistics, Springer, 2008.
[12] A. Jansen and P. Niyogi, "Intrinsic Fourier Analysis on the Manifold of Speech Sounds," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing,* 2006.
[13] M.H. Law and A.K. Jain, "Incremental Nonlinear Dimensionality Reduction by Manifold Learning," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 3, pp. 377-391, Mar. 2006.
[14] K.C. Lee, J. Ho, and D. Kriegman, "Acquiring Linear Subspaces for Face Recognition under Variable Lighting," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 5, pp. 684-698, May 2005.
[15] S. Li, K. Chan, and C. Wang, "Performance Evaluation of the Nearest Feature Line Method in Image Classification and Retrieval," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 11, pp. 1335-1339, Nov. 2000.
[16] L. Li and H. Li, "Dimension Reduction Methods for Microarrays with Application to Censored Survival Data," *Bioinformatics.* vol. 20, no. 18, pp. 3406-3412, 2004.
[17] T.B. Moeslunda, A. Hiltonb, and V. Krgerc, "A Survey of Advances in Vision-Based Human Motion Capture and Analysis," *Computer Vision and Image Understanding,* vol. 104, nos. 2/3, pp. 90-126, 2006.
[18] E. Murphy-Chutorian and M. Trivedi, "Head Pose Estimation in Computer Vision: A Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 4, pp. 607-626, Apr. 2009.
[19] S.A. Nene, S.K. Nayar, and H. Murase, "Columbia Object Image Library (COIL-20)," Technical Report CUCS-005-96, Columbia Univ., 1996.
[20] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition.* Prentice-Hall, 1993.
[21] D. Reynolds and R. Rose, "Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Model," *IEEE Trans. Speech and Audio Processing,* vol. 3, no. 1, pp. 72-83, Jan. 1995.
[22] S. Roweis and L. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science,* vol. 290, pp. 2323-2326, 2000.
[23] L. Saul and S. Roweis, "Think Globally, Fit Locally: Unsupervised Learning of Nonlinear Manifolds," *J. Machine Learning Research,* vol. 4, pp. 119-155, 2003.
[24] J. Tenenbaum, V. De Silva, and J. Langford, "A Global Geometric Framework for Nonlinear Dimension Reduction," *Science,* vol. 290, pp. 2319-2323, 2000.
[25] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski, "Information Retrieval Perspective to Nonlinear Dimensionality Reduction for Data Visualization," *J. Machine Learning Research,* vol. 11, pp. 451-490, 2010.
[26] K.Q. Weinberger and L.K. Saul, "Unsupervised Learning of Image Manifolds by Semidefinite Programming," *Int'l J. Computer Vision,* vol. 70, no. 1, pp. 77-90, 2006.
[27] J. Zhang, S.Z. Li, and J. Wang, "Nearest Manifold Approach for Face Recognition," *Proc. Sixth Int'l Conf. Automatic Face and Gesture Recognition,* May 2004.
[28] Z. Zhang, J. Wang, and H. Zha, "Adaptive Manifold Learning," *Advances in Neural Information Processing Systems,* L.K. Saul, Y. Weiss, and L. Bottou, eds., vol. 17, pp. 1473-1480, MIT Press, 2005.
[29] Z. Zhang and H. Zha, "Structure and Perturbation Analysis of Truncated SVD for Column-Partitioned Matrices," *SIAM J. Matrix Analysis and Applications,* vol. 22, no. 4, pp. 1245-1262, 2001.
[30] Z. Zhang and H. Zha, "Principal Manifolds and Nonlinear Dimensionality Reduction via Tangent Space Alignment," *SIAM J. Scientific Computing,* vol. 26, no. 1, pp. 313-338, 2004.
[31] H. Zha and Z. Zhang, "Continuum Isomap for Manifold Learning," *Computational Statistics and Data Analysis,* vol. 52, pp. 184-200, 2007.
[32] H. Zha and Z. Zhang, "Spectral Properties of the Alignment Matrices in Manifold Learning," *SIAM Rev.,* vol. 51, no. 3, pp. 545-566, 2009.

Zhenyue Zhang received the BS degree in mathematics from Fudan University, Shanghai, China, in 1982 and the PhD degree in scientific computing from the same university in 1989. He was an assistant professor in the Department of Mathematics, Fudan University, from 1982 to 1985, and has been a full professor in the Department of Mathematics, Zhejiang University, since 1998. His current research interests include machine learning and its applications, numerical linear algebra, and recommendation systems.

Jing Wang received the BS degree from the Department of Mathematics, Zhejiang University, China, in 2001, and the PhD degree from the same university in 2006. He is now an associate professor in the School of Computer Science and Technology, Huaqiao University, China. His research interests include manifold learning, data mining, and numerical linear algebra.

Hongyuan Zha received the BS degree in mathematics from Fudan University, Shanghai, in 1984, and the PhD degree in scientific computing from Stanford University in 1993. He was a faculty member of the Department of Computer Science and Engineering at Pennsylvania State University from 1992 to 2006 and he worked from 1999 to 2001 at Inktomi Corporation. He is now a professor in the School of Computational Science and Engineering, College of Computing, Georgia Institute of Technology. His current research interests include web search, recommendation systems, and machine learning applications.