Local Smoothing for Manifold Learning

JinHyeong Park[1], Zhenyue Zhang[2], Hongyuan Zha[1] and Rangachar Kasturi [3]

[1]Dept. of Computer Science and Engineering, The Pennsylvania State University,
[2]Dept. of Mathematics, Zhejiang University, Yuquan Campus, Hangzhou, P.R. China
[3]Dept. of Computer Science and Engineering, University of South Florida,
[1]{jhpark,zha}@cse.psu.edu, [2]zyzhang@zju.edu.cn,[3]chair@csee.usf.edu

Abstract

We propose methods for outlier handling and noise reduction using weighted local linear smoothing for a set of noisy points sampled from a nonlinear manifold. Weighted PCA is used as a building block for our methods and we suggest an iterative weight selection scheme for robust local linear fitting together with an outlier detection method based on minimal spanning trees to further improve robustness. We also develop an efficient and effective bias-reduction method to deal with the "trim the peak and fill the valley" phenomenon in local linear smoothing. Synthetic examples along with several image data sets are presented to show that manifold learning methods combined with weighted local linear smoothing give more accurate results.

1. Introduction

Manifold learning represents a novel set of methods for nonlinear dimension reduction emphasizing simple algorithmic implementation and avoiding optimization problems prone to local minima [1, 2, 3]. It has applications in several areas in computer vision such as image interpolation and tracking. At the pixel level, for example, a set of images such as a collection of faces can be represented as high-dimensional vectors, but their intrinsic dimension is usually much smaller than the ambient feature space. Capturing this intrinsic dimension from a set of samples is of paramount importance in computer vision applications which can be clearly seen by the increasing interests in subspace-based methods used in computer vision: Principal Component Analysis (PCA) [4], Independent Component Analysis (ICA) [5], Non-linear Matrix Factorization (NMF) [6], just to name a few. Furthermore, the above generally linear methods have recently been joined by several non-linear dimensionality reduction methods such as Isomap (Isometric feature mapping) [1, 7], LLE (Local Linear Embedding) [2] and LTSA(Local Tangent Space Alignment)[3]. One chief advantage of the newer methods is that they can successfully compute dimension reduction for data points sampled from a *nonlinear* manifold. Similar to PCA, however, the various manifold learning methods are still quite sensitive to outliers. The focus of this paper is the development of an outlier handling and noise reduction method that can be used by those nonlinear methods as a preprocessing procedure to obtain a more accurate reconstruction of the underlying nonlinear manifold. More importantly, outlier handling and noise reduction tend to reduce the chances of having *short-circuit* nearest neighbor connections and therefore help to better preserve the topological/geometric structures of the manifold in a neighborhood graph constructed from a finite set of noisy sample points.

In this paper we use the basic idea of weighted local linear smoothing for outlier handling and noise reduction, and the techniques we used are similar in spirit to local polynomial smoothing employed in non-parametric regression [8]. However, since in our context, we do not have the response variables, local smoothing needs to employ techniques other than least squares fitting. Furthermore, we apply local smoothing in an iterative fashion to further improve accuracy.

We assume that $\mathcal{F}$ is a d-dimensional manifold in an m-dimensional space with *unknown* generating function $f(\tau),\ \tau \in \mathcal{R}^d$, and we are given a data set of N vectors $x_i \in \mathcal{R}^m$, $i = 1, 2, \ldots, N$, generated from the following model,

$$x_i = f(\tau_i) + \epsilon_i, \quad i = 1, \ldots, N, \tag{1}$$

where $\tau_i \in \mathcal{R}^d$ with $d < m$, and ϵ_i's represent noise. The goals of nonlinear manifold learning are to 1) construct the τ_i's from the x_i's; and 2) construct an approximation of the generating function $f(\tau)$ [3]. Before applying a manifold learning method, we propose to carry out a local smoothing procedure as follows. For each sample point $x_i, i = 1, 2, \ldots, N$, we compute its k nearest neighbor sample points. The sample point x_i is then examined to see if it is located in the middle of two or more patches of the manifold using outlier detection based on Minimum Spanning Tree (MST). If it is we move x_i to one of the patches,

Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)
1063-6919/04 $20.00 © 2004 **IEEE**

otherwise we compute an affine subspace using Weighted PCA (WPCA) from the k nearest neighbors, and project the x_i to this affine subspace. After projecting (or moving) all the sample points, we correct their bias, and then the above steps are iterated several times.

The rest of the paper is organized a follows: WPCA and the weight selection are discussed in Section 2 and Section 3 respectively. The MST-based outlier handling follows in Section 4. We propose a new bias correction method in Section 5. Experimental results are discussed in Section 6 with conclusions and future works presented in Section 7.

2 Weighted PCA

A number of robust methods exists in the literature to deal with outlier problems, especially in the statistics community: M-estimators [9], Least Median of Squares [10] etc. These methods were applied to computer vision problems, and a number of variations were published to compute robust linear subspaces despite the presence of outliers [11, 12, 13]. The weighted PCA (WPCA) we present next is similar to the ideas of robust M-estimation. Since we only consider object-level outliers, our version of WPCA has a closed-form solution. (With both object-level and feature-level outliers, the WPCA problem requires an optimization problem to be solved in an iterative fashion, see [12] for example.)

Now consider an arbitrary sample point, say x_1, and without lose of generality, assume its k nearest neighbors including x_1 in terms of Euclidean distance are $x_1, \ldots, x_k$. We want to fit these k neighbors using an affine subspace parameterized as $c + Ut$, where $U \in \mathcal{R}^{m\times d}$ forms the orthogonal basis of the affine subspace, $c \in \mathcal{R}^m$ gives the displacement of the affine subspace, and $t \in \mathcal{R}^d$ represents the local coordinate of a vector in the affine subspace. To this end, we consider the following weighted least squares problem

$$\min_{c,\,\{t_i\},\,U^TU=I_d} \sum_{i=1}^{k} w_i \|x_i - (c + Ut_i)\|_2^2, \qquad (2)$$

where $w_1, \cdots, w_k$ are a given set of weights the choice of which will be discussed in the next section. We denote $X = [x_1, \cdots, x_k], T = [t_1, \cdots, t_k]$, and form the weight vector and the corresponding diagonal matrix, respectively, as

$$w = [w_1, \cdots, w_k]^T, \quad D = \mathrm{diag}(\sqrt{w_1}, \cdots, \sqrt{w_k})$$

Let $\bar{x}_w = \sum_i w_i x_i / \sum_i w_i$ be the weighted mean of $x_1, \cdots, x_k$, and let $u_1, \cdots, u_k$ be the largest left singular vectors of $(X - \bar{x}_w e^T)D$, where e is the k-dimensional column vector of all ones. Then, the optimal solution of the problem (2) is given by

$$c = \bar{x}_w, \quad U = [u_1, \cdots, u_d], \quad t_i = U^T(x_i - \bar{x}_w).$$

The sample point x_1 in question is then projected to x_1^* as

$$x_1^* = \bar{x}_w + UU^T(x_1 - \bar{x}_w).$$

This projection process is done for each of the sample points in $\{x_1, \ldots, x_N\}$.

3. Selecting Weights for WPCA

In this section, we consider how to select the weights to be used in WPCA suggested in the previous section. Since the objective of introducing the weights is to reduce the influence of the outliers as much as possible when fitting the points to an affine subspace, ideally the weights should be chosen such that w_i is small if x_i is considered as an outlier. Specifically, we let the weight w_i be inversely proportional to the distance between x_i and an ideal center x^*. Here x^* is defined as the mean of a *subset* of the points in which outliers have been removed. The ideal center x^*, however, is unknown. We will use an iterative algorithm, as we now show, to approximate x^* starting with the mean of the sample points.

For concreteness we consider weights based on isotropic Gaussian density function defined as

$$w_i = c_0 \exp(-\gamma \|x_i - \bar{x}\|_2^2), \qquad (3)$$

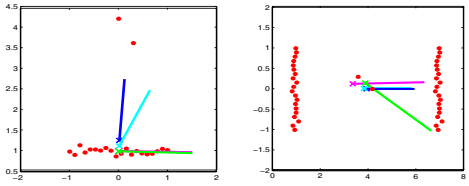
where $\gamma > 0$ is a constant, $\bar{x}$ an approximation of x^*, and c_0 the normalization constant such that $\sum_{i=1}^n w_i = 1$. Other types of weights discussed in [8] can also be used.

Our iterative weight selection procedure for a given set of sample points $x_1, \ldots, x_k$ is structured as follows: Choose the initial vector $\bar{x}_{w^{(0)}}$ as the mean of the k vectors $x_1, \ldots, x_k$, iterate until convergence,

1. Compute the current normalized weights, $w_i^{(j)} = c_j \exp(-\gamma \|x_i - \bar{x}_{w^{(j-1)}}\|_2^2)$.
2. Compute a new weighted center $\bar{x}_{w^{(j)}} = \sum_{i=1}^k w_i^{(j)} x_i$.

Space limit will not allow us to discuss the convergence of the algorithm , however, we now present an informal analysis to illustrate why the iterative weight selection process can be effective at down-weighting the outliers. Consider the simple case where the data set of $k + \ell$ samples has several outliers $x_1, \cdots, x_\ell$ that are far away from the remaining set $x_{\ell+1}, \cdots, x_{\ell+k}$ which are relatively close to each other. We also assume $\ell \ll k$. Denote $\bar{x}_0$ the mean of all $\ell + k$ sample points and $\bar{x}$ the mean of the sample points $x_{\ell+1}, \cdots, x_{\ell+k}$. It is easy to see that

$$\bar{x}_0 = \frac{1}{\ell + k}\Big(\sum_{i=1}^{\ell} x_i + k\bar{x}\Big) = \bar{x} + \delta, \quad \delta = \frac{1}{\ell + k}\sum_{i=1}^{\ell}(x_i - \bar{x}).$$



(a) (b)

Figure 1: WPCA results for various γ's. Blue line denotes the first principal component direction of the original PCA, and the other color lines denote those of WPCA with different γ's (cyan:γ=0.1, magenta:γ=0.5 and green:γ=1). Magenta line is overlapped with green line in (a), and cyan line is overlapped with blue line in (b). A "X" mark denotes the weighted center after applying WPCA.

If $\|\delta\| \gg \|x_j - \bar{x}\|$ for $j = \ell+1, \cdots, \ell+k$, we have $\|x_j - \bar{x}_0\|_2 \approx \|\delta\|_2$. Therefore, for $j = \ell+1, \cdots, \ell+k$, $w_j = c_0 \exp(-\gamma\|x_j - \bar{x}_0\|_2^2) \approx c_0 \exp(-\gamma\|\delta\|_2^2)$. On the other hand, since $\|x_i - \bar{x}\|_2 \gg \|\delta\|_2$ for $i \leq \ell$, we have $w_i = c_0 \exp(-\gamma\|x_i - \bar{x}_0\|_2^2) \approx 0$. If $\|\delta\|$ is not large compared with the distances between $\bar{x}$ and the clustered points $x_{\ell+1}, \cdots, x_{\ell+k}$, we also have $\|x_i - \bar{x}_0\| \approx \|x_i - \bar{x}\| \gg \|x_j - \bar{x}\| \approx \|x_j - \bar{x}_0\|$ for $i \leq \ell < j$. This implies $w_i \approx 0$ and $w_j \approx constant$. Therefore, recalling that the weight set is normalized such that $\sum w_i = 1$, we conclude that $w_0 \approx 0$, $i \leq \ell$, $w_i \approx \frac{1}{k}$, $i > \ell$. Therefore, $\bar{x}_w \approx \bar{x}$ and the updated weights w_i have the effect of down-weighting the outliers.

4. Further Improvement for Outlier Handling using MST

In most situations, WPCA with the above weight selection scheme can handle noise and outliers quite well. However, it is not easy for WPCA to handle outliers located between two or more patches of a manifold (Figure 1-(b)).

Figure 1 depicts two examples of WPCA for various γ's. In case (a), we present a set of outliers under which WPCA performs quite well. The weights of the outliers become close to zero when we increase the value of γ and the first principal axis is computed correctly using WPCA (the green line or the red line). In Figure 1-(b), the weight selection method yields non-negligible weights for the two outliers lying between the two patches of a manifold and almost zero for weights associated with the other points of the patches for a large value of γ. The first principal axis becomes the green line with $\gamma = 1$ see Figure 1-(b).

We now show that we can improve the performance of local smoothing further by tackling this problem using Minimum Spanning Tree (MST). The key observation is that if a point x_i is an outlier as shown in Figure 1-(b), the distance between the outlier point and the nearest patch is much greater than the distances between two near-by points in a patch. Based on this observation, we suggest an outlier detection method using Weighted Minimum Spanning Tree (MST).

A spanning tree for a connected undirected weighted graph G is defined as a sub-graph of G that is an undirected tree and contains all the vertices of G [14]. A minimum spanning tree (MST) for G is defined as a spanning tree with the minimum sum of edge weights. Suppose that we examine an arbitray sample point, say x_1, and without loss of generality let $X = \{x_1, ..., x_k\}$ denote the set of k-nearest neighbors of x_1 including x_1 itself. The MST-based outlier detection procedure proceeds as follows. First, we compute a weighted Minimum Spanning Tree (MST) using $X = \{x_1, ..., x_k\}$. We define an undirected, weighted, fully connected graph $G = (V, E, W)$, where $V = \{x_1, ..., x_k\}$, $E = \{e_{12}, e_{13}, ...e_{(k-1)k}\}$ and $W = \{w(e_{12}), w(e_{13}), ..., w(e_{(k-1)k})\}$. An element in E, e_{ab} , stands for an edge associated with x_a and x_b, and $w(e_{ab})$ is the Euclidean distance between the two vertices.

An MST algorithm such as Prim's algorithm [14] provides a $k-1$ edge list, $E^{'} = \{..., e^{'}_{1a}, e^{'}_{1b}, ...\}$ where we assume $e^{'}_{1a}$ and $e^{'}_{1b}$ are the two edges associated with x_1, which spans all the points in X with the minimum total edge weights[1]. After obtaining the MST, we compute the average weight ($mean_w$) of the MST. If the maximum edge weight associated with x_1 is much greater than the average edge weight, we divide the MST into two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. This is done by disconnecting one of the edges, $e^{'}_{1a}$ or $e^{'}_{1b}$, whose weight is greater than the other as shown in Figure 2. In the next step, we examine if the point is far from the other patch. Assume G_1 contains x_1. We compare the maximum edge weight in G_1 to the average edge weight ($mean_w$). If the edge weight is also much greater than the average weight, the point is determined as the outlier between the two patches.

If a point, x_1 is diagnosed as an outlier between two local patches, it is moved in the direction of one of the patchs according to the following equation, without projecting it onto its tangent space.

$$\begin{aligned} x_1^* &= \alpha x_1 + (1-\alpha)x_t, \quad 0 < \alpha < 1 \\ t &= \text{argmax}\{W(e^{'}_{1a}), W(e^{'}_{1b})\} \end{aligned}$$

Figure 2 illustrates an example where a point x_i (red circle) is an outlier located between the two local patches. The circles represent data points and the lines stand for edges computed using MST. The two blue circles are two points connected by the point, x_1. In the figure, the MST is divided

[1] In this section, we only consider the outliers between two patches of a manifold (Figure 1-(b)) because WPCA can handle the other type of outliers shown in Figure 1-(a) quite well.

into two subgraphs, G_1 and G_2 by disconnecting the edge, e_{1a}, because the edge weight is much greater than the average edge weight of the MST . The two dashed-ellipsoids show the two sub-graphs. The point, x_1, is determined as an outlier between two patches because the maximum weight,$W(e_{bc})$, in graph G_1 is also much greater than the average edge weight, $mean_w$. The outlier is moved to, x_1^* in the direction of x_a because $W(e_{1a})$is less than the distance between x_1 and x_c.

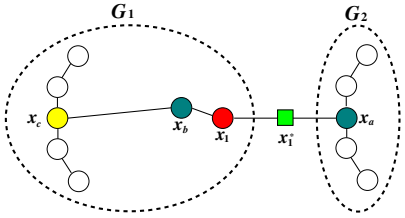


Figure 2: An example of Minimum Spanning Tree (MST). The circles represent data points and lines MST edges. The two blue circles, x_a and x_b are two neighbor points of x_1 composing of two MST edges associated with the point. Green square is the new position of x_1

5. Bias Reduction

Local-smoothing based on linear local fitting does suffer from the well-known *trim the peak* and *fill the valley* phenomenon for regions of the manifold where the curvature is large [8]. When we increase the number of iterations in local linear smoothing, the net effect of this is that the set of projected sample points tend to shrink (Figure 3-(b)(c)), and eventually converge to a single point. One remedy is to use high-order polynomials instead of the linear one [8, 15], but this can be very expensive for high-dimensional data because we need to approximate quantities such as the Hessian matrix of $f(\tau)$. Let x_i be an arbitrary sample point and denote x_i^* as the updated point of $x_i, i = 1, \ldots, N$ by applying the above smoothing algorithm (WPCA and MST). Clearly, if we know an approximation δ_i for the difference $f(\tau_i) - x_i^*$ of the updated point x_i^* of x_i, x_i^* can be improved by adding δ_i to x_i^* to reduce the bias in x_i^*,

$$x_i^* \leftarrow x_i^* + \delta_i$$

The effectiveness of a bias correction method depends on the bias estimation of $f(\tau_i) - x_i^*$. We propose to estimate the bias as $f(x_i) - x_i^* \approx x_i^* - \hat{x}_i^*$, where $\{\hat{x}_i^*\}$ is the updated points of $\{x_i^*\}$ obtained by the same smoothing procedure (WPCA and MST) (Similar ideas have been applied in nonlinear time series analysis [15]). This idea suggests the following bias estimation

$$\delta_i = \big(\bar{x}_i^w + U_i U_i^T (x_i - \bar{x}_i^w)\big) - \big(\bar{x}_i^{*w} + U_i^* U_i^{*T} (x_i^* - \bar{x}_i^{*w})\big),$$

where $\bar{x}_i^w$ and U_i are the weighted mean and orthonormal basis matrix of the affine subspace constructed from WPCA using the k nearest neighbors of x_i from the data set $\{x_i\}$, and $\bar{x}_i^{*w}$ and U_i^* are the weighted mean and orthonormal basis matrix of the affine subspace constructed from WPCA using the k nearest neighbors of x_i^* from the data set $\{x_i^*\}$. This gives the following updating formula:

$$x_i^* = 2\big(\bar{x}_i^w + U_i U_i^T (x_i - \bar{x}_i^w)\big) - \big(\bar{x}_i^{*w} + U_i^* U_i^{*T} (x_i^* - \bar{x}_i^{*w})\big). \quad (4)$$

The proposed smoothing algorithm along with bias correction is summarized as follows. Given a set of sample points $\{x_1, \ldots, x_N\}$:

1. (**Local Projection**) For each sample point, x_i
   - Compute the k-nearest neighbors of x_i, say $x_{i1}, \cdots, x_{ik}$
   - Compute MST using $x_{i1}, \cdots, x_{ik}$
   - Determine if x_i is an outlier bwtween two patches in the manifold using MST
   - If it is, move x_i in the direction of one of the two patches. Otherwise, project x_i onto an affine subspace which is computed by WPCA to obtain x_i^*.
2. (**Bias Correction**) For each x_i^*
   - Correct bias using Eq. 4
3. (**Iteration**) Go to 1 until convergence

In our experiments with noise-corrupted data, it was observed that the results converge to the original manifold very fast, but no formal proof of convergence has been worked out.

6. Experimental Results

In this section, we present experimental results for our local smoothing algorithms using a 2D spiral curve data set and three image data sets. We will use k to represent the number of the nearest neighbors, and d the dimension of affine subspaces used for local smoothing. We use k_{Isomap} as the number of the nearest neighbors used for Isomap. We also use γ as the parameter used in Eq. 3 to compute weights.

6.1 2D Spiral Curve Data

For the 2D spiral curve, a Gaussian noise corrupted sample is generated with 500 points and is plotted in Figure 3-(a). We also used an outlier overlaid spiral curve sample with 500 points: 400 points sampled from the curve with Gaussian noise and 100 points for outliers (Figure 4-(a)). The outlier points are generated uniformly in the range of $x \in [-15, 15]$ and $y \in [-15, 10]$ which includes the spiral curve.

Figure 4 and 5 depict the smoothing results of the Gaussian noise corrupted spiral curve sample and the outlier overlaid spiral curve sample, respectively. In these experiments, different γ values are used by fixing k to be 30. The

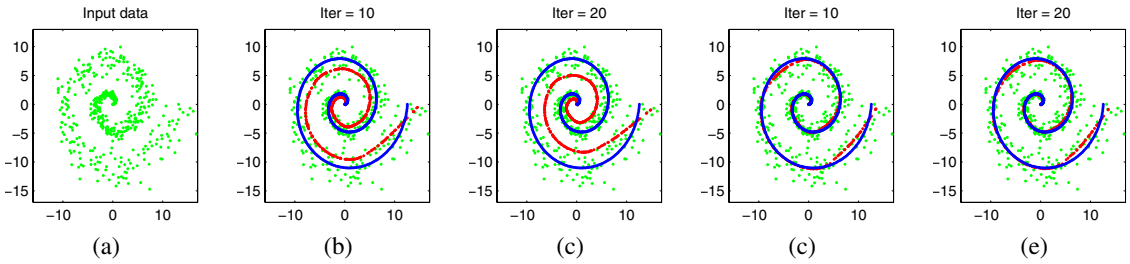


Figure 3: Shrinking Example. Green dots, red dots and blue dots are the original input data points, smoothing results at the iteration and target manifold to estimate, respectively. Parameters: k=30, γ=0.1; (a) Original input image. (b)&(c) are the results without bias correction. (d)&(e) are results with bias correction.

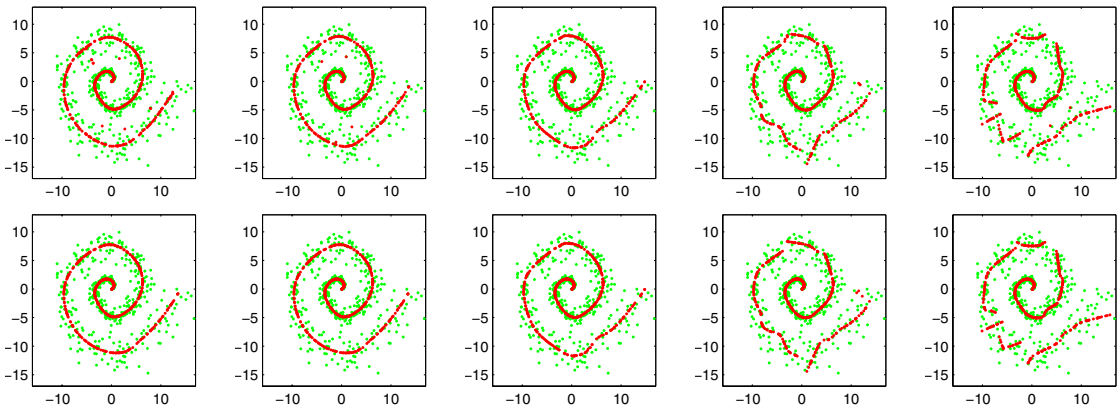
first row in Figure 4 illustrates smoothing results without applying the MST outlier handling method, and the second row illustrates those after applying MST outlier handling. The results in the figure show that the MST outlier handling method makes the smoothing method less sensitive to γ. Figure 4 and 5 illustrate that the relatively small γ produces better results for the Gaussian noise data while relatively large γ yields better results for the outlier-overlaid data. These can be explained as follows. In the outlier-overlaid case, most points are close to the curve, and a relatively small number of points are located far from the curve. When we compute a tangent space for an outlier point, most of the points in its k-nearest neighbor are close to the curve. In this situation, large γ yields a better tangent space than small γ as shown in Figure 1. This is because the weight of outliers is close to zero and that of the other points on a manifold is non-zero for a large γ. In the Gaussian noise case, all the points on the spiral curve are contaminated by zero mean Gaussian noise. When we observe a local region of the curve, its tangent space tends to be close to that of a noise-free spiral curve at the region, and therefore low γ works better in this instance.

Figure 6 presents smoothing results for different ks by fixing γ to be 0.1 which seems to be the best γ based on our empirical experience. In general, the parameter k should be chosen such that k nearest neighbor points should represent the linearity locally in the manifold dimension d well. For this smooth manifold the results are relatively insensitive to the choice of the parameter k as depicted in Figure 6. We obtained similar results for the outlier overlaid case.

6.2 Face Images and Two Video Sequences (Human Walking and Ballet)

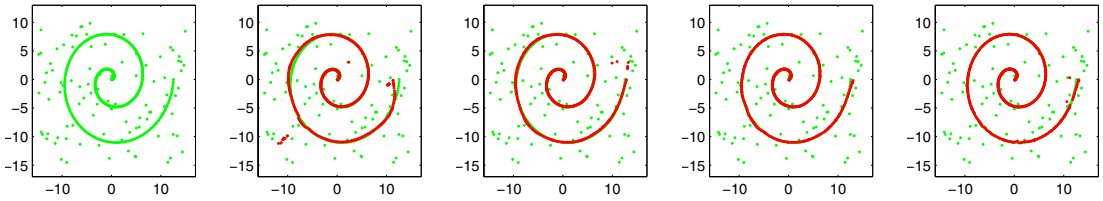
Experiments were also conducted using three image data sets: face images (total 698), a human walking video sequence (109 frames) and a ballet video sequence (166 frames). The face images lie essentially on a 3D manifold which has been shown in [7, 1]. The three dimensions can be interpreted as follows: one dimension corresponds to the lighting variation and the other two dimensions correspond to horizontal and vertical pose variations, respectively. The gray-scale images are 64×64. The human walking video sequence and ballet video sequence are digitized in the sizes of 240×352 and 240×320 respectively. We generate images of the video sequences by cropping only the walking human using simple vision techniques. The cropped images of the walking clip are down-sized to 60×40, and the ballet images are down-sized to 50×45. They are also gray-scaled from 0 to 1. We artificially added noise and outliers to the image sets by adding occlusions to selected images. We randomly selected images from each data set and overlaid a constant intensity noise patch at a random location. The intensity value of the patch is also selected randomly. The examples of the occlusion-overlaid images are shown in Figure 8-(a) for the face images, Figure 10-(a) for the human walking images and Figure 12-(a) for the ballet images. We converted each image to a vector by concatenating each row: a 4096 dimensional vector for a face image, a 2400 dimensional vector for a human walking image and a 2250 dimensional vector for a ballet image. For the purpose of decreasing time complexity, we reduced the dimension of the vectors of each data set by projecting them onto an affine subspace which is computed using PCA with 99% variance preserved. The reduced dimension of the face data, walking data and ballet data is 216, 90 and 134 respectively. Again, this dimensionality reduction is just for time complexity reduction, and is not related to the smoothing performance.

The proposed smoothing method using WPCA and MST is applied to each occlusion data set. We apply Isomap to each occlusion data set and its smoothing results to verify if the smoothing method can handle the noise and outliers properly. The best Isomap result for a smoothing data set is computed by applying a different combination of parameters such as k, d and γ. We define *the best Isomap result* as the one that has the least residual variance in the intrinsic dimension (nonlinear manifold dimension). In almost



(a) γ=0 (b) γ=0.1 (c) γ=0.2 (c) γ=0.5 (e) γ=1.0

Figure 4: Smoothing results for Gaussian noise data with different γ's. Green dots are the input data, and red dots are smoothing results. The first row shows the results without MST outlier handling, and the second row shows those with MST otlier handling. The parameter *k* is set to 30 and the number of iteration to 10.



(a) Outlier Spiral (b) γ=0 (c) γ=0.1 (d) γ=1.0 (e) γ=10.0

Figure 5: Smoothing results for outlier-overlaid data with different γ's. The parameter *k* is set to 30 and the number of iteration to 10.

all our experiments we observed that the residual variances of the best result were smaller than the corresponding residual variances in other results. This could be a useful cue for computing the best result when we do not know the intrinsic dimension.

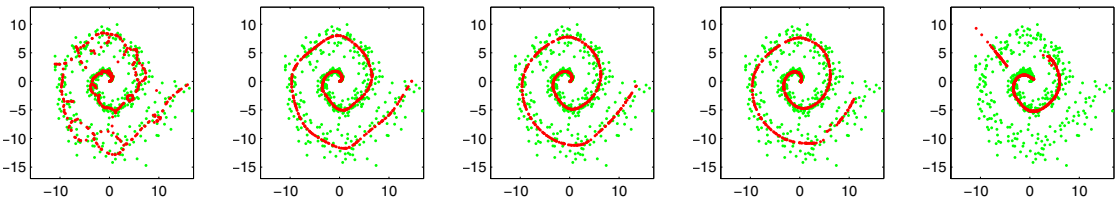
We used the same residual variance used in [7] [1] which is defined as below:

$$Residual\_Variance = 1 - corrcoef(D, D')^2$$

where $corrcoef(D, D')$ is the correlation coefficient between two vectors D and D'. D has the pairwise geodesic distances of the points in an original dimension and D' has those after reducing the dimension using Isomap. We want to emphasize that for the walking data set and ballet data set, the time order is not used in the construction of the nearest neighbors used by our local smoothing algorithm and Isomap. The dynamics (time-ordering) of the frames in these data sets are *automatically* discovered by the Isomap method.

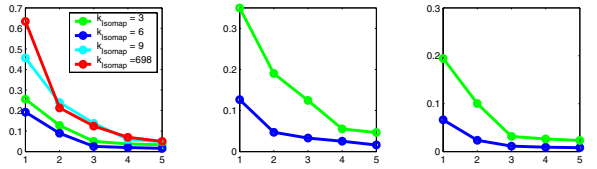
Figure 7 shows the residual variance for the noise-free face data set in (a), and those for the occlusion face data set and the smoothed results in (b) and (c). In Figure 7 (a), k_{Isomap} denotes the number of neighbors for Isomap computation. It shows that Isomap gives the best result for the noise-free face data set when k_{Isomap} is set to 7 (blue line). When we set k_{Isomap} to 698 (the total number of face images), Isomap yields the same results as PCA because all the pair-wise geodesic distances become Euclidean distances. Therefore, the red line in Figure 7-(a) shows residual variances of original PCA.

In Figure 7-(b) and (c), the green line depicts the residual variances for the occlusion face data set, and the blue line shows the residual variance after applying the proposed smoothing method. Figure 8 illustrates the first 42 occlusion-overlaid face images (in the size of 20×20 to 20% of the face image) and their smoothing results after applying the smoothing method, which are the results corresponding to Figure 7-(c). As shown in Figure 8-(b), most occlusion images are successfully projected onto the face manifold by removing the occlusion parts after applying the proposed smoothing method to them. However, it is possible that some outliers are not projected to their original



(a) k=10 (b) k=20 (c) k=30 (c) k=50 (e) k=100

Figure 6: Smoothing results with different ks. γ is set to 0.1 and the number of iteration to 10.



(a) Original images (b) 20×20, 20% (c) 20×20, 30%

Figure 7: Residual variances of face images along the dimension after applying Isomap to face images. (a) shows residual variances for original images without noise. k_{Isomap} in (a) stands for the number of neighbors for Isomap. (b) and (c) show the residual variance for occlusion-overlaid cases (green line) and their smoothing results (blue line). $s \times s$, $r\%$ means a $s \times s$ size of noise patch is overlaid to $r\%$ of the data

(a) (b)

Figure 8: (a) shows occlusion-overlaid face images in the size of 20×20 to 20% of the data set. (b) shows the smoothing results of (a).

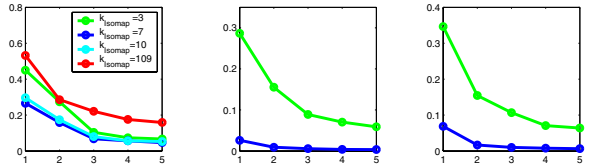
position correctly in the face manifold. A possible explanation is as follows. When a occlusion patch is overlaid to an image it can be displaced far from the original position, and close to the other patch in the manifold. Actually, it depends on the intensity of the occlusion patch. In this situation, the neighbors of the outlier for WPCA come from the patch rather than around its original location. Then, the tangent space is computed using the neighbors from the patch, and the outlier is projected to the patch rather than to the original position. Those examples are shown in the 2^{nd} row & 2^{nd} column and 3^{rd} row & 1^{st} column in Figure 8. Figure 9 and 10 show the experiment results using the human walking data set. These results show that the proposed smoothing method handles the occlusion-overlaid images quite well.

7. Conclusions and Future Work

The focus of this paper is outlier handling and noise reduction for a set of noisy sample points using local linear smoothing. It is shown that the smoothing method is very effective not only for reducing Gaussian noise but also for effective handling of outliers. The performance of the method, however, depends on two parameters: the number of neighbors k used in local smoothing and the γ used for weight computation. As far as γ is concerned, experiments show that relatively low value of γ yields good results for the case when most points in the data are corrupted by Gaussian noise. If many points are involved in a manifold and some outliers are added to the data, a relatively high γ tends to produce better results. For choosing the parameter k, the neighbor points should be selected approximately form a linear affine subspace. The experiments using the image data sets also show that the proposed local smoothing method has a potential to remove occlusions overlaid on images. Several issues deserve further investigation: 1) automatic selection of parameters such as the number of neighbors and γ for weight computation; 2) exploiting RPCA [12] for image data which assigns weights to the pixels and/or images. 3) exploiting different metrics such as tangential distance to compute k nearest neighbors for image data sets rather than the Euclidean distance.

Acknowledgments

The work was supported in part by NSF grants CCF-0305879 and DMS-0311800.

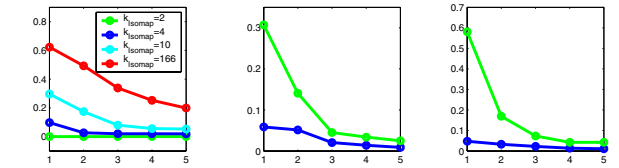


(a) Original images (b) 10×10, 30% (c) 10×10, 40%

Figure 9: Residual variances for each dimension after applying Isomap. (a) shows residual variances for original images without noise. (b) and (c) show the residual variances for occlusion-overlaid cases.

(a) (b)

Figure 10: (a) shows occlusion-overlaid human walking images in the size of 10×10 to 40% of the data set. (b) shows the smoothing results of (a).



(a) Original images (b) 15×15, 30% (c) 15×15, 40%

Figure 11: Residual variances for each dimension after applying Isomap. (a) show residual variances for original images without noise. (b) and (c) show the residual variance for noise contamination cases.

(a) (b)

Figure 12: (a) shows occlusion-overlaid images in the size of 15×15 to 30% of the data set. (b) shows the smoothing results of (a).

References

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

[2] S. Roweis and L. Saul. Nonlinear dimension reduction by locally linear embedding. *Science*, 290: 2323–2326, 2000.

[3] Z. Zhang and H. Zha. Principal Manifolds and Nonlinear Dimension Reduction via Local Tangent Space Alignment. To appear in SIAM J. Scientific Computing.

[4] I. T. Jolliffe. *Principal Component Analysis*. Springer-Berlag, New York, 1986.

[5] A. Hyvärinen. Survey on independent component analysis. *Neural Computing Surveys*, 2:94–128, 1999.

[6] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. *Adv. Neural Info. Proc. Syst. 13*, pages 556–562, 2001.

[7] J. B. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in Neural Information Processing Systems*, 1998.

[8] C. Loader. Local Regression and Likelihood. Springer, New york, 1999.

[9] K. Gabriel and S. Zamir. Lower rank approximation of matrices by least squares with any choice of weights. *Technometrics*, 21(21):489–498, 1979.

[10] P.J. Rousseeuw. Least median of squares regression. *J. of Amer. Stat. Assoc.*, 79:871–880, 1984.

[11] L. Xu and A. Yuille. Robust principal component analysis by self-organizing rules based on statistical physics approach. *IEEE Trans. Nueral Networks*, 6(1):131–143, 1995.

[12] F. De la Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(Issue 1-3):117–142, Aug-145 2003.

[13] D. Skocaj and Leonardis A. Weighted incremental subspace learning. In *Proc. workshop on Cognitive Visioin*, 2002.

[14] S. Baase and A.V. Gelder. *Computer Algorithms, 3rd Ed.* Addison-Wesley, 2000.

[15] T. Schreiber and M. Richter. Nonlinear projective filtering in a data stream. *Int. J. Bifurcat. Chaos*, 9:2039 1999.