Dimension Reduction of Image Manifolds

Arian Maleki
Department of Electrical Engineering
Stanford University
Stanford, CA, 94305, USA
E-mail: arianm@stanford.edu

I. Introduction

Dimension reduction of datasets is very useful in different application including classification, compression, feature extraction etc.; Linear methods such as principal Component Analysis, have been used for a long time and seem to work very well in many applications. But, there are much more applications in which the dataset doesn't have a linear structure and so the linear methods do not work very well for them. Fig. (1) and Fig. (2) illustrate what linear and nonlinear structures mean. When the points of the dataset lie close to a linear subspace of the ambient space, its structure is called linear and if they are close to a smooth manifold as in Fig. (2), then it is said to have nonlinear structure. Nonlinear dimension reduction methods try to recover the underlying parametrization of scattered data on a manifold embedded in high dimensional Euclidean space. In the next section some of the linear and nonlinear methods will be explained very briefly; But, the focus of this report will be on image processing applications of dimension reduction algorithms. Usually the image databases (each image is considered as a very large dimensional vector) do not have linear structures; Donoho and Grimes [1] have shown that in some image databases nonlinear methods will lead to very reasonable parameters and this proved the efficiency of these methods in processing image databases. In this report, nonlinear methods will be used in three different image processing applications: Image Synthesis,Image Classification and Video Synthesis. It will be shown that the methods which are proposed here can easily outperform the linear methods. The structure of this paper is as follows: In the next section, different algorithms of dimension reduction will be reviewed. In section III two ways are explained for synthesizing Images. In section IV an image classification is proposed and compared with the other methods and finally in section V, I will explain how to synthesize a video by using the manifolds.

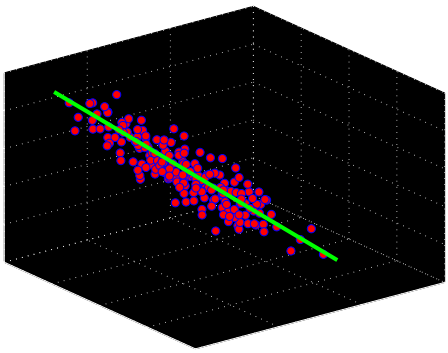

Fig. 1. Linear Data Structure

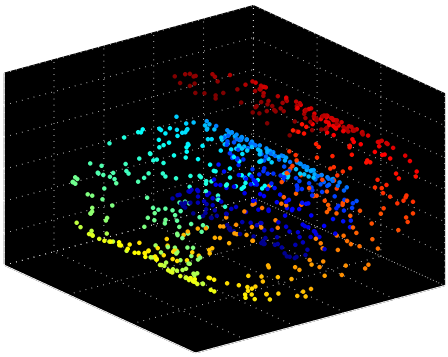

Fig. 2. Nonlinear Data Structure

During this project several different databases have been tested; But in order to use the same database during the whole report, most of the figures in this report are generated from the same database. This image database is shown in Fig. (3).

II. Dimension Reduction Algorithms

In this section some of the linear and nonlinear dimension reduction algorihtms will be explained.

Fig. 3. Some of the patches in the first Database I used

A. Linear Methods

1) Principal Component Analysis(PCA): PCA [2] is a way of changing the coordinates such that the first coordinate is the maximum variance direction (i.e. among all the directions that you can project the data on, projection on this direction may have the maximum variance). The second coordinate of PCA is orthogonal to the first direction and among all the directions that are orthogonal to the first direction it has the maximum variance property. The other directions of PCA can be defined in the same way. For more information please refer to [2].

2) Multidimensional Scaling: Multidimensional Scaling (MDS) deals with the following problem: for a set of observed similarities (or distances) between every pair of N items, find the representation of the items in fewer dimensions such that the similarities or distances of these new items is close to the original similarities. It is obvious that in most of the situations the similarities are not exactly the same as the original similarities and one should just does his/her best to make them as close as possible. Assume that the similarities are Euclidean distances between the points $\underline{x}_1, \underline{x}_2, ..., \underline{x}_n$; These points are embedded in an m dimensional space $\mathbb{R}^m$ and the question is, can these points be embedded in the d-dimensional ($d \ll m$) Euclidean space such that their structure(or the distances between them) doesn't change? Assume that all the vectors are zero mean. Obviously, the solution of this problem is not unique and is invariant to translation and rotation of the data. In order to make the solution unique, some more constraints are imposed.The constraint is that the lower dimensional points are found from an orthogonal projection of the original data on a d-dimensional space. Assume that the data in d-dimensional space is represented by $\underline{y}_1, \underline{y}_2, ..., \underline{y}_n$. Let d_{ij} be the distance between $\underline{x}_i$ and $\underline{x}_j$ and $\hat{d}_{ij}$ shows the distance between $\underline{y}_i$ and $\underline{y}_j$; The closeness of d_{ij} and $\hat{d}_{ij}$ is measured with $Stress$ which is defined in this way:

$$Stress = \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ij}^2 - \hat{d}_{ij}^2) \tag{1}$$

An interesting point about this quantity is that , the elements of summation are not in the absolute value. The reason is that $\underline{y}_i$ is an orthogonal projection of $\underline{x}_i$ and so $\hat{d}_{ij}$ is always less than d_{ij}. The advantage of defining $Stress$ in this way is that the global optimum of this problem can be found explicitly and there is no need for an iterative algorithm. The optimum projection would be the first d principal components of the covariance matrix of $\underline{x}_1, \underline{x}_2, ..., \underline{x}_n$ [3]. In other words, $\underline{y}_1, \underline{y}_2, ..., \underline{y}_n$ can be found by the multiplication of X with the matrix which is found from the principal components of the covariance matrix. For more information refer to [3].

B. Nonlinear Methods

1) ISOMAP: ISOMAP [4] is an algorithm that tries to preserve the geometry of the data. MDS also tries to keep the geometry of the data unchanged; The term geometry was defined by the similarity matrix in section II. It was also mentioned in section II that, the Euclidean distance is usually used as a similarity matrix. But, as it is obvious the Euclidean distance is not at all, representative of the geometry of a Manifold. Instead, another distance called geodesic distance can represent the geometry very well. Geodesic distance between two points on the manifold is the length of the shortest path on the manifold from one point to another. This is a brief description of the ISOMAP algorithm:

- Find some neighbors of each point; Two different criteria are considered for neighborliness . One is epsilon-neighborhood and the other is k-neighborhood. In epsilon neighborhood, a real value epsilon is considered and every point that has euclidean distance of less than epsilon to the point, is recognized as its neighbor. In k-neighborhood algorithm, the k closest points are considered as the neighbors of the point.
- In order to find a geodesic distance between two points we try to connect every two points with a path which is constructed from connecting the neighbors of the points. The geodesic distance is the length of the shortest path.

- apply the MDS to this similarity Matrix and reduce the dimension which has geodesic distances as its elements.

2) Locally Linear Embedding: Locally Linear Embedding(LLE) [5] is another algorithm to retrieve the parameter space of a Manifold. Again LLE is trying to somehow preserve the geometry of the data which is in a high dimensional space. But, the method this algorithm chooses to do so is different from that of ISOMAP. The algorithm is as follows:

- Find the optimum value of w_{ij} such that:

$$w_{ij}^{opt} = \arg\min_{w_{ij}} \sum_{i=1}^{n} |x_i - \sum_{j=1}^{n} w_{ij} x_{ij}|^2 \qquad (2)$$

subject to these constraints:

$$\begin{cases} w_{ij} = 0, & \text{if } x_j \notin N_k(x_i); \\ \sum_j w_{ij} = 1 & \forall i \end{cases} \qquad (3)$$

in which $N_k(x_i)$ shows the k-neighborhood of the point x_i;

- Find $y_1, y_2, ...y_n$ in a d-dimensional space such that this cost function is minimized:

$$Error = \sum_{i=1}^{n} |y_i - \sum_{j=1}^{n} w_{ij}^{opt} y_{ij}|^2 \qquad (4)$$

as can be seen each of these two steps are simple Least Squares Problems and so the global optimum of each of these two problems can be found easily. This algorithm tries to keep the neighborhood of each point (or local structure of manifold) almost unchanged.

III. Image Synthesis

Assume that two images are given (from the same database) and we want to somehow find an average of these two images. How can it be done? It is obvious that, the simple average doesn't work at all and will not generate any reasonable image. For example consider the two images shown in the first row of Fig. (4) are given and the goal is to find the average of these two images.

Considering the average to be the midpoint on the line that connects the vectors of these two images will result in an image which is not a natural image at all(this is equivalent to simple averaging). For example the simple average of these two images is also shown in Fig. (4); Two different algorithms are proposed here to solve this problem.

1) All the images are on the manifold that was found before (By ISOMAP or LLE). There is a path on the manifold that connects these two images and has a shortest length (the geodesic). Find the midpoint of this path and that's the average image. Obviously if the density of the points on the manifold is not enough, the average image will not be very natural but for most practical purposes the density of points is high enough. The result of this algorithm is shown in Fig. (4);

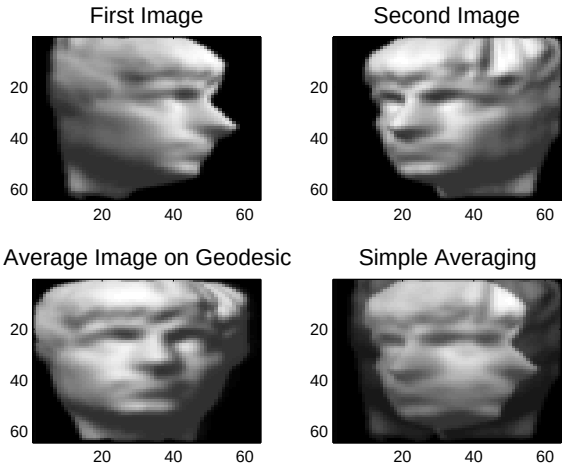

Fig. 4. Two Images and their Geodesic Average as defined in this report and their simple average!

2) Assume that the parameter space of the manifold is convex. Also assume that the parameters that correspond to our two images are θ_1 and θ_2; I define the average image as the image that corresponds to the parameter:

$$\theta_{avg} = \frac{\theta_1 + \theta_2}{2} \qquad (5)$$

But θ_{avg} may correspond to a point which is not in our data base. In order to approximate it with the points in the database, Assume that $\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_K$ are the K nearest neighbors of θ_{avg};

- Calulate w^{opt} in this way:

$$\underline{w}^{opt} = \arg \min_{(w_1, w_2, ..., w_K)} ||\theta_{avg} - \sum w_i \hat{\theta}_i||^2 \qquad (6)$$

- Find the image from this equation:

$$Image_{avg} = \sum w_i^{opt} x_i \qquad (7)$$

in which x_i is the image that corresponds to $\hat{\theta}_i$.

The result of this algorithm when applied to two images is shown in Fig. (5); It is obvious that the second method is more general in the sense that with this method for every value of θ in the parameter space one can synthesize an image corresponding to this parameter.

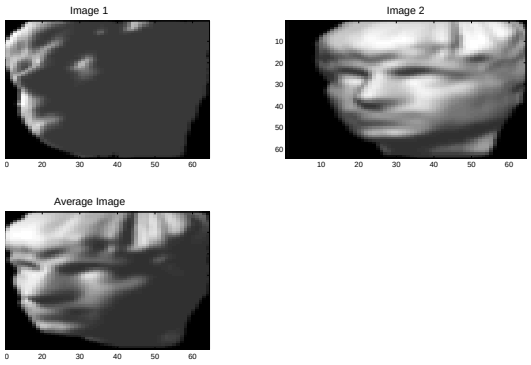

Fig. 5. Two images and their average by the second method

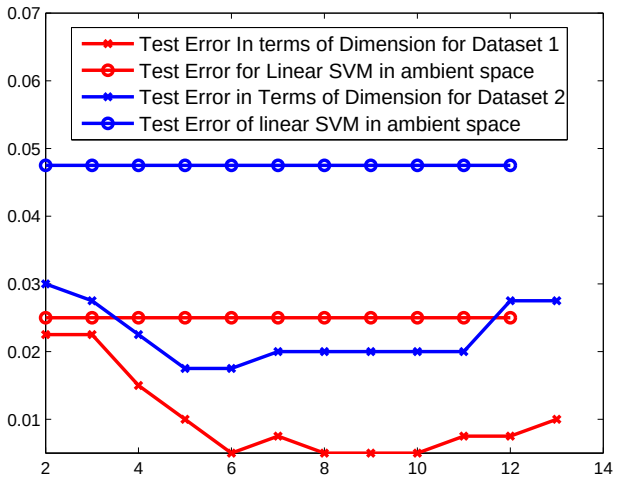

Fig. 6. Test Error of Classification 2s and 3s in Terms of Dimension for two datasets

IV. Image Classification

Support Vector Machine (SVM) ([2] or [6]) uses the fact that by transforming the data to higher dimensional space we can make it more linearly separable. Therefore in SVM method, we first transform the data to a higher dimensional space and then try to find the optimum separating hyperpalne between the two classes. Making the data more separable (in the linear sense) is the main reason behind the success of SVM. Now, assume that the data is on a manifold and is not separable in that high dimensional ambient space. How can we find a good kernel(or equivalently transform) to make this data more suitable for linear classifiers. A very reasonable answer to this problem is to use the $PARAMETER\ SPACE$. There are several advantages in using the parameter space. First, it is usually very low dimensional space and whatever algorithm we use for classification, overfitting is much less likely(Of course we haven't lost much information about the relative distances of the points and so the likelihood of underfitting in parameter space is not much more than that of the ambient space). On the other hand, in many situations when we transform the data to the parameter space because we flatten the curls of the manifold, the data becomes more separable. All these observations lead us to use parameter space for classification. Here is a method for classification of the data:

- Find the lower dimensional coordinates of the training data;
- By using SVM find the best hyperplane that separates the data in lower dimensional space
- Find the nearest neighbors of each test point in the training data;
- Find the optimum linear weights that approximate the test data from the Training data
- Use the same weights to approximate the parameters of the test data
- Use SVM classifier in lower dimensional space

In order to understand the best dimension for classification, one can use cross validation; But as it will be seen later the performance of this algorithm is usually not very sensitive to the dimension we choose, as far as the dimension is reasonable. There are other parameters in the algorithm that can be chosen through cross validations; But, on the datasets that I have done my simulations, the performance of the algorithm was very robust to the small variation of those parameters.

As an example, the classification of 2s and 3s is shown here. A set of 1000 images of handwritten 2s and 3s is used to train the model. The size of each image is 28x28; Two datasets of 200 images are considered as Test sets. The test errors in terms of the dimension of parameter space is shown in the following figure;

As it can be seen, as the dimension changes from 4 to 11 the error doesn't change very much; In terms of cross validation the best dimension is 6, but as mentioned before $4 \leq d \leq 11$ is acceptable. It can also be seen that if the dimension is chosen correctly, this algorithm can easily outperform SVM in the ambient space.

V. Synthesizing Video

Again assume that we have a database of images. For example assume that these images are photos of a face from different horizontal and vertical directions. Also assume that two images are given (I_1 and I_2) from this database. The goal is to synthesize a video to show us

a way that the camera can move very smoothly from one position to another. The idea to achieve this goal is very similar to the idea which was proposed before, for finding the average. Again we find the geodesic between the two patches. Then we divide the geodesic into n points(n is the number of frames that we want in the video)and synthesize these frames from their neighbors. To be more specific assume that $x_1, x_2, ..., x_k$ are the image patches from the dataset that lie on the geodesic of the two images $x_1 = I_1$ and $x_k = I_2$. Also assume that the geodesic distance of x_1 and x_k is equal to D; The distance of each new patch from x_1 is calculated from this formula(distance is calculated on the geodesic from x_1 to x_n):

$$d(x_1, z_j) = \frac{D(j-1)}{n} \tag{8}$$

in which z_j is the j'th synthesized frame and $d(x_1, z_j)$ represent the distance of z_j and x_1 on the geodesic of x_1 and x_k; In order to synthesize z_j assume that i is found such that:

$$d(x_1, x_i) \leq d(x_1, z_j) \leq d(x_1, x_{i+1}) \tag{9}$$

Then use this formula to reconstruct z_j:

$$\begin{aligned} z_j = x_i \frac{d(x_1, x_{i+1}) - d(x_1, z_j)}{d(x_1, x_{i+1}) - d(x_1, x_i)} \\ + x_{i+1} \frac{d(x_1, z_j) - d(x_1, x_i)}{d(x_1, x_{i+1}) - d(x_1, x_i)} \end{aligned} \tag{10}$$

Fig. (7) Shows an example of this algorithm. Leftmost frame and rightmost frame are given and 3 frames are synthesized;

VI. Conclusion and Future Work

In this report some of the methods which are used for linear and nonlinear dimension reduction have been reviewed and applied to some of the simple databases of images. Two algorithms have been proposed for generating some synthetic images that can work well if the manifold of images has been sampled at a high density. This also led us to the video synthesis to show smooth variation of one image to another image. Also, I proposed a method for classifying high dimensional data by using its low dimensional parameter space. This algorithm needs more thought and more simulations; Also it should be tested in different applications of classification and on other databases. In this report only binary classification was considered; Definitely, one of the future directions would be to extend this algorithm to more general classification problems. In addition to classification there are some other interesting questions that shall be answered. For example, is there any better method for synthesizing patches? Another problem that shall be answered is how can someone compare the result of LLE and ISOMAP. In other words, a measure is needed to say which parameters match better to the database. This problem seems very interesting, because there are many methods of Nonlinear dimension reduction and it is not easy to say which one works better than the others.

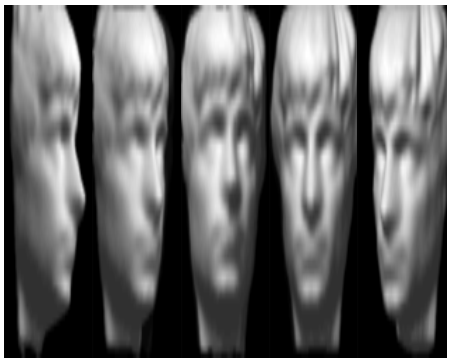

Fig. 7. 5 Frames of Video to Show the Rotation of The Face

References

[1] D.L. Donoho, C. Grimes,"Image Manifolds which are Isometric to Euclidean Space", Journal of mathematics Imaging nad Vision(23),5-24,2005

[2] A. Ng, "Lecture Notes of CS229", Stanford University, http://www.stanford.edu/class/cs229/

[3] T.F. Cox, M.A. Cox, Multidimensional Scaling, Second Edition, 2000.

[4] D.L. Tennenbaum, V. De Silva, "A Global Geometric Framework For Nonlinear dimensionality reduction," Sciencs(290), 2000.

[5] S. Roweis, L. Saul "Nonlinear Dimensionality Reduction by Locally Linear Embedding ," Science(290),2000.

[6] T. Hastie, et. al,The Elements of Statistical Learning, Springer, 2001

[7] D. Donoho, "Wedgelets: Nearly Minimax estimation of Edges,"Anals of Stat., vol. 27, pp.859-897, 1999.