NONLINEAR MANIFOLD LEARNING

VIKAS CHANDRAKANT RAYKAR

Abstract. The aim of this report is to study the two nonlinear manifold learning techniques recently proposed (Isomap [12] and Locally Linear Embedding (LLE) [8]) and suggest directions for further research. First the Isomap and the LLE algorithm are discussed in detail. Some of the areas that need further work are pointed out. A few novel applications which could use these two algorithms have been discussed.

1. Problem Statement

Manifold learning can be viewed as implicity inverting a generative model for a given set of observations. Let Y be a d dimensional domain contained in a Euclidean space R^d. Let $f : Y \rightarrow R^D$ be a smooth embedding for some $D > d$. The goal is to recover Y and f given N points in R^D. Isomap [12] and LLE [8] provide implicit description of the mapping f. Given $X = \{x_i \in R^D \mid i = 1 \dots N\}$ find $Y = \{y_i \in R^d \mid i = 1 \dots N\}$ such that $\{x_i = f(y_i) \mid i = 1 \dots N\}$.

For example, consider the swiss roll data set which is used in many of the papers. The swiss roll is generated by the following equations $x_1 = y_1 cos\, y_1$; $x_2 = y_1 sin\, y_1$; $x_3 = y_2; y_1 \in [3\pi/2, 9\pi/2]; y_2 \in [0, 15]$. Based on a set of x_1,x_2 and x_3 we have to find y_1 and y_2. Note that we are implicitly inverting the generative model without explicit parametrization of the generative function f.

2. Types of Embedding

Without imposing any restrictions of f the problem is ill-posed. The simplest case is a linear isometry i.e. f is a linear mapping from $R^d \rightarrow R^D$ where $D > d$. In this case Principal Component Analysis (PCA) recovers the d significant dimensions of the observed data. Classical Multidimensional Scaling (MDS) produces the same results but uses the pairwise distance matrix instead of the actual coordinates.

Two other possibilities are considered in [6, 5]. f can be either a isometric embedding or a conformal embedding. An isometric embedding preserves infinitesimal lengths and angles while a conformal embedding preserves only infinitesimal angles (it does not preserve lengths). In case of conformal embedding at every point $y \in Y$ there is a scalar $s(y) > 0$ such that the infinitesimal vector at y gets magnified by a factor $s(y)$. In case of isometric embedding $s(y) = 1$. One way to visualize these two embeddings is to consider a 2D rubber sheet. The rubber sheet can be curved and folded such that it gets embedded in a 3D space e.g. the rubber sheet could be folded into a S shaped curve. If we fold the rubber sheet without stretching it we have an isometric embedding. If we stretch the rubber sheet with stretching

---

This report was written as a part of the course CMSC828J: Approaches to representing and recognizing objects offered in FALL 2003 by Dr. David Jacobs.

differing at different places we have a conformal embedding. The swiss roll data set is an isometric embedding.

The Isomap algorithm can recover an isometric embedding. The LLE can recover both isometric as well as conformal embeddings. Note that isometric is a special case of conformal embedding where $s(y) = 1$.

3. Isomap Algorithm [12]

One important property of isometric embedding is that the manifold can be regarded as a metric space under geodesic distance i.e. when we unfold the manifold we get a Euclidean space (An isometric embedding is said to be intrinsically flat.). If f is an isometric embedding then the geodesic distance is an invariant under the mapping f. The distance between an two points in Y is equal to the geodesic distance between the two points in $f(Y)$. Isomap algorithm as proposed in [12] uses this invariance and constructs a geodesic metric based on the observed data alone without any knowledge of the underlying metric. If Y is a convex domain in R^d and the data points are sufficiently dense that Isomap can successfully recover the original Euclidean structure.

The crux of the Isomap algorithm is finding an efficient way to compute the true geodesic distance between observations, given only their Euclidean distances in the higher dimensional observation space. The idea is that Euclidean distance is approximately equal to the geodesic distance for closeby points. For points which are faroff the geodesic distance has to be computed by a series of hops. The Isomap algorithm as proposed in [12] consists of three main steps.

(1) Construct the neighborhood graph G over all observation points. Connect points i ad j if they are closer than ϵ or if i is one of the K nearest neighbors of j. Set the edge lengths equal to distance between i and j. The distance could be either Euclidean or other domain specific distance metric.
(2) Compute shortest paths in the graph between every two points using either the Floyd's or the Djkstra's algorithm.
(3) Apply MDS to the resulting geodesic distance matrix to find a d-dimensional embedding.

4. Locally Linear Embedding (LLE) [8]

LLE takes a slightly different approach than Isomap. It eliminates the need to estimate pairwise distances between widely separated data points. The LLE algorithm can be described as below:

(1) For each data point X_i find its K nearest neighbors.
(2) We expect each data point and its neighbors to lie on or close to a locally linear patch of the manifold. Each point can be written as a linear combination of its neighbors. Compute the weights W_{ij} that best linearly reconstructs X_i from its neighbors.
(3) If the data lie on or near a smooth nonlinear manifold of lower dimensionality then there exists a linear mapping (consisting of translation, rotation and rescaling) that maps the higher dimensional coordinates of each neighborhood to global internal coordinates on the manifold. By design, the weights that minimize the reconstruction errors are invariant to rotation, rescaling and translation of the data points. Invariance to translation is enforced by adding the constraint that the weights sum to one. *Hence the*

same weights that reconstruct the data points in D dimensions should reconstruct it in the manifold in d dimensions. The weights characterize the intrinsic geometric properties of each neighborhood. Compute the lower dimensional embedding vectors Y_i best reconstructed by W_{ij}.

See Appendix I for a detailed derivation of the LLE algorithm.

5. Finer Points and Further Directions

In this Section I discuss some of the more subtler points with an untoward bias towards where it would fail. Some of the theoretical issues that need to be resolved are also discussed.

5.1. **The Case of the Conformal Embedding.** Conformal mappings are locally isometric only to a scale $s(y)$. Since the reconstruction weights in LLE are invariant to *rescaling*, it is successful in recovering the lower dimensional embedding. However Isomap can recover only an isometric embedding.

However recently a new version of the algorithm called the C-isomap [6] has been proposed by the same authors for the case of conformal embeddings. The algorithm is based on estimating the scale factor at each point. The method requires the assumption that the original sampling is uniform. A typical example quoted for conformal embedding is the stereographic fishbowl data (also called stereographic projection). Points are uniformly samples on a disk and are projected on a fish bowl lying below it. Uniformly sample points on the disk bunches up non-uniformly near the rim of the fish bowl.

Since conformal maps are locally isometric up to a scale factor $s(y)$, we first estimate $s(y)$ at each point in the observed data. By rescaling, we can then restore the original metric structure of the data and proceed as in Isomap. We can do this by noting that a conformal map rescales local volumes by a factor $s(y)^d$. If the hidden data are sampled uniformly in Y then the density in the observed space will be $1/s(y)^d$. Hence if the original sampling density is uniform then we can estimate the scale factor $s(y)$ at each point based on the density in the observed space. The algorithm is same as the isomap but with the modification that in the graph G edge lengths are weighted by $1/\sqrt{M(i)M(j)}$. $M(i)$ is some estimate of the density at the point. In [5] the mean distance to its k nearest neighbors is used. The authors mention that in asymptotic analysis the exact form of the weighting function is not that critical.

5.1.1. *Non uniform sampling.* The C-isomap requires the assumption that data is uniformly sampled in the original latent space Y. Typical examples include the stereographic and the mercator projection. In a stereographic projection, points are uniformly samples on a disk and are projected on a fish bowl lying below it. Uniformly sample points on the disk bunches up non-uniformly near the rim of the fish bowl. However consider the case of a uniform fishbowl where the data is sampled uniformly on the surface of the fish bowl. Sice there is no bunching around effect we cannot estimate the scale factor. C-isomap would behave very much like isomap since our rescaling factor would be constant. See Figure 1 in [6] for results of the C-isomap algorithm on the uniform fish bowl data set. In the plots the LLE algorithm gave better results than the c-isomap for uniform fishbowl data. First of all is is not clear what to expect out of the algorithm in the case of points uniformly embedded on the top of a sphere.

5.2. **Are there other types of embeddings?** The natural question that arises is whether there are other types of embeddings? Are these formally studied in math literature somewhere? For example like in conformal embeddings where the local neighborhood is scaled the local neighborhood could be sheared? LLE weights are not invariant to shearing. Isomap also fails in this case. Can the local neighborhood be characterized by nonlinear weights which are invariant to the particular kind of embedding?

5.3. **What are the manifolds of natural images?** In the papers the authors show their results on natural images which are generated by smooth variations of certain parameters. One thing that needs to be studied is what is the nature of these manifolds. To start these studies could be done for simple images like circle moving around in an image. What sort of embedding do these images get embedded in the higher dimensional space? What kind of natural images can be successfully recovered by the above two algorithms.

Manifold learning is ideal for images which are produced by smoothly varying some parameters. For example faces with different poses may naturally exhibit manifold geometry. In [12] the algorithm is applied to had written digits. In this case it is not clear whether they lie on a smooth manifold. They show that Isomap still finds some globally meaningful coordinates.

5.4. **The problem of disconnected manifolds.** These methods fail if data lie on disconnected manifolds or connected manifolds each with different dimensionality. it would be beneficial if these algorithms were applied separately on different manifolds. So this involves figuring out whether there are manifolds of different dimensions connected.

5.5. **What is wrong with being non-euclidean?** In our original problem statement we assumed that Let Y be a d dimensional domain contained in a Euclidean space R^d. It is not exactly clear how to handle non-Euclidean domain R^d. What should be the desired output in such cases? Based on the problem at hand the Isomap algorithm can be used with different distance metric instead of the Euclidean distance metric. Need to understand whether this is beneficial and how does it affect the algorithm. In [12] for the 2-data set a tangent distance metric is used. Say instead of using euclidean distance metric we use procrustes distance metric which gives the distance on the manifold. Then how to embed points on the surface of the manifold. Procrustes distance directly gives the distance on the spherical manifold. So we need to derive an version of the MDS on the spherical manifold. Note that classical MDS is designed for euclidean manifold. Classical MDS takes a matrix of pairwise Euclidean distances and gives their corresponding coordinates which are consistent with the given pairwise distances. The main step in metric MDS is where the pairwise distance matrix is converted to the dot-product matrix. This is based on the cosine law and for it to be valid the distance *has to be Euclidean*. For flat manifolds like the swiss roll or spiral this is true since the geodesic distance is equal to the Euclidean distance in the original space Y. But if the distance is non-Euclidean classical MDS does not exactly recreate the original configuration. What about points on the surface of a sphere or torus? The sphere are the toroid are slightly tricky. I have a rough idea on how to do MDS on a spherical manifold. Also need to explore what exactly ordinal MDS does? In our applications is the metric structure important or just the ordering?

Second case is when 2D points get embedded on the surface of a sphere. How do you flatten a sphere or a toroid? In this case the embedding has to be on the surface of the spheres? can you flatten out a sphere? It does not make sense to embed it on a plane. The embedding has to be on the surface of the sphere.

5.6. **Explicit parametrization of f.** The two techniques described implicity parameterize the function f. For each observation point we get the corresponding point in the lower dimensional space. However, for some applications we would like to have an explicit parametrization of the function f or f^{-1}. For example interpolation or extrapolation would require to know the form of f^{-1}. Give pairs of y_i and x_i we can find f based on techniques employing neural networks and radial basis functions. If we know the explicit parametrization then if we have a new observation we can immediately get its corresponding lower dimensional feature vector without redoing the entire procedure. Note that all these methods use all the available data to build a global representation.

5.7. **Short circuit problem.** The only parameter which needs to be tuned is K or ϵ depending on the method used. This step is vulnerable to short-circuit errors if the neighborhood is too large with respect to folds in the manifold on which the data points lie or if noise in the data moves the points slightly off the manifold. Even a single short-circuit error can alter many entries in the geodesic distance matrix, which in turn can lead to a drastically different (and incorrect) low-dimensional embedding [2]. [2] demonstrate this by adding noise to the swiss roll data set. In such cases appropriately selecting K is very essential for the algorithm. Choosing a very small neighborhood is not a satisfactory solution, as this can fragment the manifold into a large number of disconnected regions. How to find appropriate K is an open issue.

5.8. **LLE and K.** In LLE we first find the K nearest neighbors. The algorithm depends on the value of K chosen. Two questions can be addressed here.

- How to automatically choose K? In LLE K is closely related to the intrinsic dimensionality of the data. First the algorithm can recover embeddings whos dimensionality is strictly less than K. How are K and the resulting embedding related?
- The unusual case where $K > D$ like in the swissroll the number of neighbors was greater than the 3 dimensional space into which it wad embedded. In this case the local reconstruction weights are not uniquely defined. So the authors add some regularization term to break the degeneracy. The regularizer favors weights that are uniformly distributed in magnitude.
- What about variable K i.e. the number of neighbors need not be same for each point? What strategy should be used to choose variable K? If so does it give any benefits?

5.9. **Convergence issues and manifold curvature.** How curved can be manifold be? How densely should the manifold be sampled? The asymptotic convergence issues which relate convergence with the manifold curvature and the sampling density can be see in [4]. It gives various converge proofs. Studying that paper could give a idea on what sort of manifolds can the algorithm handle.

5.10. **Intrinsic Dimensionality.** In the LLE algorithm we choose the eigen vectors corresponding to the bottom most $d+1$ eigen values of M. I expected these eigen values to reflect the intrinsic dimensionality of the dataset. however some examples showed in the paper do not reveal the intrinsic dimensionality except for a few examples. This effect is discussed in detail in [9]. The authors also give a method as to how to enforce the intrinsic dimensionality if it is known aproiri.

5.11. **Local vs Global.** Local approaches like LLE and Laplacian Eigenmaps [3] (this is a new technique which I have not yet studies) attempt to preserve the local geometry of the data; essentially, they seek to map nearby points on the manifold to nearby points in the low-dimensional representation. Global approaches like Isomap attempt to preserve geometry at all scales, mapping nearby points on the manifold to nearby points in low-dimensional space, and faraway points to faraway points. The principal advantages of the global approach are that it tends to give a more faithful representation of the datas global structure. The local approaches have two principal advantages: (1) computational efficiency: they involve only sparse matrix computations which may yield a polynomial speedup; (2) representational capacity: they may give useful results on a broader range of manifolds, whose local geometry is close to Euclidean, but whose global geometry may not be.

5.12. **Are these kernel methods?** There is a paper [7] which shows that all these algorithms are different versions of kernel PCA.

5.13. **Differential Geometry.** The problem of manifold Learning involves a lot of concepts from differential geometry. It would be highly beneficial to take a math course on Differential Geometry and have an understanding of the Reimannian geometry, differentiable manifolds and sub-manifolds.

6. Applications

In this section we discuss different applications where manifold learning techniques could be applied to give novel solutions to different applications. The two applications are related to the work I am familiar with.

6.1. **Localization of sensor networks.** I think manifold learning techniques are a great tool for applications in sensor network localization. The problem is that we have thousands of sensors which are ad-hoc deployed. These sensors have wireless capabilities by which they can communicate to each other (not necessarily to each other but atleast to its neighbors). The goal is to localize these sensors automatically i.e find out their positions. Localization is done in two steps: ranging and multilateration. In ranging we measure the pairwise distances between all the sensor nodes. Based on these pairwise distances we estimate the position of the nodes. If we know the pairwise distances between all the nodes we can use MDS to localize these sensors. However when there are thousands of sensors the best we can hope is that we know the pairwise distances between each sensors and its neighbors. So we can build local coordinate systems at each neighborhood. These local neighborhoods are overlapping. The idea is to combine these local coordinate systems into a global one. The LLE algorithm is very much suited for this application. In LLE we knew the actual coordinates of the points in the higher dimensional space. However the LLE can also be derived if we know just the pairwise distances between all points in small neighborhoods. The same trick used in MDS

of converting the distance matrix to dot product matrix is being used here. Here we know the dimensionality of the data set i.e three. We can enforce this by doing a MDS in the local neighborhoods and doing a projection of the points on the three principal components and the expressing each point as a linear combination of the other points in this MDS space.

6.2. **Perceptually motivated Interpolation.** Interpolation or extrapolation of data is a important problem for high dimensional data. Most of the data are sampled over a certain parameter set. We would like to know the data at a set of parameters not sampled. Blind interpolation would be justified only when the sampling rate is very high as given by the Nyquist sampling rate. Ideally interpolation should be perceptually motivated i.e. the perceptually relevant features should be interpolated rather than interpolating the actual data. Manifold learning provides a canonical set of globally meaningful features which could possibly recover the perceptually important/relevant features. The reason for this being most high dimensional data is produced by smoothly varying a few parameters. This smooth variation traces out a manifold in the higher dimensional space. Since manifold learning techniques respect the geodesic distance on the manifold these features are usually extracted. Simple linear operations in the feature space can give rise to highly non-linear transformations in the original observation space. One interesting part would be to figure out whether manifold learning gives perceptually important features? In some cases where the data depends on a large number of parameters we can find out which parameters ar most relevant.

One problem which I have been working on is interpolation of Head Related Impulse response (HRIR). A HRIR is a function of time, elevation, azimuth, and the individual. Measuring a HRIR experimentally for a fine sampling grid of elevation and azimuth is a time consuming and laborious process. In most spatial audio systems some sort of interpolation of the HRIR with respect to elevation and azimuth is needed. Interpolation is a very deceptive word here since any trivial interpolation of the HRIRs without taking into consideration the physical significance of the various features of the HRIRs and their dependency on elevation,azimuth and individuals would not make any sense. For interpolation with respect to elevation and azimuth different approaches used include direct time domain interpolation of HRIRs, direct frequency domain interpolation of HRTFs and interpolation of principal components in the PCA domain. These methods do not take into account the perceptual significance of the different features in the HRIR or the HRTF. For example if we just use linear interpolation in the frequency domain we are just averaging the magnitude of the spectrum of the closest sampling points which may distort the perceptually significant features. Ideally interpolation should be done in the perceptually important feature domain. Also a set of generic HRIR would not work satisfactorily since it has been shown that HRIRs are specific to an individual and if we use HRIRs of some other person the elevation perception will be very poor. Due to the difference in the anatomy of the humans and also the shape of the pinna the HRIRs for different persons will be quite different. I would like to try out the techniques discussed in the paper for HRIR interpolation and customization.

6.3. **Misc.**

6.3.1. *Dimensionality Reduction.* The problem of manifold learning fits into the broader category of non-linear dimensionality reduction. Many fields like computer

vision, bio-informatics, economics, weather prediction, astronomy are faced with the problem of finding relevant low-dimensional patters in large amounts of high-dimensional data. Linear dimensionality reduction techniques may not capture the perceptually relevant features if the data is embedded non linearly in the higher dimensional space.

6.3.2. *Measure of similarity.* Say we would like to find a measure of similarity between two face images of different poses. In this case the geodesic distance on the manifold would be a ideal choice.

6.3.3. *Classfication/Recognition.* All recognition and classification algorithm (like pose or gait recognition) which used to use the linear features (based on PCA/MDS) now could be cast in the framework of the non-linear features extracted and see if we get better results.

6.3.4. *Information Visualization.* Manifold learning could be a great tool for information visualization where experts would like to find patterns in high dimensional data.

6.3.5. *Manifolds of perception.* [10] suggests that there could be a connection between the neural manifolds and the image manifolds. There is a hypothesis that a visual memory is stored as a manifold of stable states. Manifold could prove to be crucial for understanding how perception arises from the dynamics of neural networks in the brain [10]. In auditory neuroscience localization of a sound source by the human ears is a active field. Could the signal spectrum for different directions around the head be stored as a manifold.

7. Conclusion

A good place to start exploring on the field on manifold learning is the website Resources on manifold learning maintained by Martin Law at MSU [1].

8. Appendix I:LLE Derivation

Here I give a detailed derivation of the LLE algorithm. I have slightly modified the derivation given in the paper [9] in that, I have derived it completely in terms of vectors and matrices so that it can be easily implemented in MATLAB.

8.1. **Compute the reconstruction weights.** Compute the weights W_{ij} that best linearly reconstruct X_i from its neighbors. Let W be an $N \times N$ matrix where W_{ij} is the weight by which the j^{th} point should be multiplied to reconstruct X_{ij}.

$$W = arg\min_{W} \sum_{i=1}^{N} \|X_i - \sum_{j=1}^{N} W_{ij} X_j\|^2 \tag{8.1.1}$$

This minimization has to be done subject to two constraints:

- *The sparseness constraint* $W_{ij} = 0$ if X_j is not one of the K neighbors of X_i.
- *The invariance constraint* $\sum_{i=1}^{N} W_{ij} = 1$ i.e. the weights should sum to 1.

Consider one point a $D \times 1$ vector X. Let X_1, X_2, ...,X_K be the K nearest neighbors of x. Define:

$$Z = [X_1|X_2|\dots|X_K] \tag{8.1.2}$$

Let W (slight change of notation forget the old matrix W) be a $K \times 1$ vector be the reconstruction weight vector. Then the reconstructed vector can be written as:

$$\hat{X} = ZW \tag{8.1.3}$$

8.2. **Invariance properties of W.** The constrained weights for any particular data point are invariant to rotations, rescalings and translations of that data point and its neighbors.

8.2.1. *Scale Invariance.* Let s be the scaling factor.

$$\begin{aligned} Z^{new} &= sZ \\ \hat{X}^{new} &= (sZ)W = s(ZW) = s\hat{X} \end{aligned} \tag{8.2.1}$$

8.2.2. *Rotation Invariance.* Let R be the rotation matrix.

$$\begin{aligned} Z^{new} &= RZ \\ \hat{X}^{new} &= (RZ)W = R(ZW) = R\hat{X} \end{aligned} \tag{8.2.2}$$

8.2.3. *Translation Invariance.* Let t be the translation vector.

$$\begin{aligned} Z^{new} &= Z + t\mathbf{1}^t \\ \hat{X}^{new} &= (Z + t\mathbf{1}^t)W = ZW + t\mathbf{1}^tW = \hat{X} + t \end{aligned} \tag{8.2.3}$$

The translation invariance is *enforced* by the sum to one constraint i.e $\mathbf{1}^tW = 1$.

8.3. **Derivation of reconstruction weights.**

$$W = arg\min_W \frac{1}{2}\|X - ZW\|^2 \tag{8.3.1}$$

subject to the constraint that $\mathbf{1}^TW = 1$. Let

$$J(W) = \frac{1}{2}\|X - ZW\|^2 = \frac{1}{2}(X - ZW)^T(X - ZW) \tag{8.3.2}$$

The constrained minimization can be done using the method of lagrange multipliers.

$$J_L(W) = \frac{1}{2}(X - ZW)^T(X - ZW) + \lambda(\mathbf{1}^TW - 1) \tag{8.3.3}$$

where λ is the lagrange multiplier.

$$\frac{\partial J_L(W)}{\partial W} = -Z^TX + Z^TZW + \lambda\mathbf{1} = \mathbf{0} \tag{8.3.4}$$

$$(Z^TZ)W = Z^TX - \lambda\mathbf{1} \tag{8.3.5}$$

Let us define $C = Z^TZ$ called the neighborhood correlation matrix.

$$W = C^{-1}(Z^TX - \lambda\mathbf{1}) \tag{8.3.6}$$

Substituting for W from this equation into the constraint $\mathbf{1}^TW = 1$ we get:

$$\lambda = \frac{\mathbf{1}^TC^{-1}Z^TX - 1}{\mathbf{1}^TC^{-1}\mathbf{1}} \tag{8.3.7}$$

8.4. **Lower Dimensional embedding.** The final step is to construct a low dimensional embedding based on the reconstruction weights. Let

$$Y = [Y_1|Y_2|\dots|Y_N] \tag{8.4.1}$$

a $d \times N$ matrix, be the corresponding lower dimensional embedding of X ($d < D$). Find Y to minimize

$$\Phi(Y) = \sum_{i=1}^{N} \|Y_i - \sum_{j=1}^{N} W_{ij} Y_j\|^2 \tag{8.4.2}$$

In matrix form this can be written as.

$$\begin{aligned} \Phi(Y) &= Tr[(Y - YW)^T (Y - YW)] \\ &= Tr[(Y - YW)(Y - YW)^T] \end{aligned} \tag{8.4.3}$$

$$\Phi(Y) = Tr[Y(I - W)(I - W)^T Y^T] \tag{8.4.4}$$

Define $M = (I - W)^T (I - W)$. Then

$$\Phi(Y) = Tr[Y M^T Y^T] = Tr[Y M Y^T] \tag{8.4.5}$$

since M is a symmetric matrix. In order to make the problem well posed we introduce additional constraints.

- The cost function $\Phi(Y)$ is invariant to translation of the coordinates. we remove this translational degree of freedom by requiring that the outputs be centered at the origin i.e. $Y\mathbf{1} = \mathbf{0}$.
- In order to remove the rotational degree of freedom and fix the scale we constrain Y to have unit covariance i.e. $YY^T = I$.

We first minimize the cost function subject to the second constraint.

$$Y = arg \min_{Y} Tr[Y M Y^T] \tag{8.4.6}$$

subject to the constraint that $W^T W = I$. Later from this solution we show how the translational invariance constraint ca be imposed. Again we use the method of lagrange multipliers it can be shown that the solution is a generalized eigen value problem of the form

$$MY^T = Y^T \Sigma \tag{8.4.7}$$

Σ is the diagonal matrix of the eigen values. Also

$$\Phi(Y)_{minimum} = Tr[\Sigma] \tag{8.4.8}$$

So we can choose the eigen vectors corresponding to the bottom most $d + 1$ eigen values. we ignore the last eigen vector with eigen value 0. Consider

$$M\mathbf{1} = (I - W)(I - W)^T \mathbf{1} = \mathbf{0} \tag{8.4.9}$$

using the fact that $W^T 1 = 1$ i.e. the weights sum to 1. Therefore the bottom most eigen value will be 0 with the corresponding eigen vector $\mathbf{1}$. Now by the orthogonality constraint all other eigen vectors should be orthogonal to $\mathbf{1}$ i.e.

$$(Y^T)^T = Y\mathbf{1} = \mathbf{0} \tag{8.4.10}$$

hence discarding the last eigen vector enforces the translation invariance constraint. This corresponds to the free translation mode of eigen value 0. Since M is a sparse matrix efficient methods exist to compute the eigen values.

References

1. *http://www.cse.msu.edu/ lawhiu/manifold/index.html*.
2. M. Balasubramanian, E. L. Schwartz, J. B. Tenenbaum, V. de Silva, and J. C. Langford, *The isomap algorithm and topological stability*, Science **295** (2002).
3. M. Belkin and P. Niyogi, *Laplacian eigenmaps and spectral techniques for embedding and clustering*, Advances in Neural Information Processing Systems (2001).
4. M. Bernstein, V. de Silva, J. Langford, and J Tenenbaum, *Graph approximations to geodesics on embedded manifolds*, Tech. report, Department of Psychology, Stanford University, 2000.
5. V. de Silva and J. B. Tenenbaum, *Nonlinear estimation and classification*, ch. Unsupervised learning of curved manifolds, Springer-Verlag, New York, 2002.
6. ———, *Local versus global methods for nonlinear dimensionality reduction*, Advances in Neural Information Processing Systems **15** (2003).
7. J. Ham, D. D. Lee, S. Mika, and B. Schlkopf, *A kernel view of the dimensionality reduction of manifolds*.
8. S. Roweis and L. Saul, *Nonlinear dimensionality reduction by locally linear embedding*, Science **290** (2000), 2323–2326.
9. L. Saul and S. Roweis, *Think globally, fit locally: Unsupervised learning of low dimensional manifolds*, Journal of Machine Learning Research **4** (2003), 119–155.
10. H. S. Seung and D. D. Lee, *The manifold ways of perception*, Science **290** (2000), 2268.
11. J. B. Tenenbaum, *Mapping a manifold of perceptual observations*, Advances in Neural Information Processing Systems **10** (1998).
12. J. B. Tenenbaum, V. de Silva, and J. C. Langford, *A global geometric framework for nonlinear dimensionality reduction*, Science **290** (2000), 2319–2323.

Department of Electrical Engineering, University of Maryland, College Park
E-mail address: vikas@umiacs.umd.edu