Dimensionality Reduction using Noisy Distance Data

Pratik Biswas 05157667 *

December 15, 2006

1 Motivation

The basic idea behind most dimensionality reduction algorithms is to find a low dimensional embedding of high dimensional data while roughly preserving the local relationships between the data points. The inputs to these algorithms are usually the point coordinates in the higher dimensional space or some form of mutual distance data between the points in the higher dimensional space.

However, very often, we are unable to capture the distance relationships between the points exactly, or for that matter between all the points. The examples I consider in this project are inspired from distance geometry, that is, to find a map of the relative positions of points in a low dimensional space given some incomplete and inaccurate information between the points. This model can be applied to sensor network localization or to molecule structure determination.

For example, in a wireless sensor network, the sensors are able to communicate with only a few sensors in a close neighborhood of its own. Using Received Signal Strength or Time of Arrival measurements, each sensor can arrive at a rough distance estimate of its neighbors. The challenge is to find the positions of all the sensors in space using these measurements [1],[2]. The situation is similar in a molecule structure prediction problem. Using NMR or X-ray crystallography measurements, we can estimate distances between some pairs of atoms. The objective is to determine the complete structure of the molecule in 3-D space [3],[4].

The general distance geometry problem can be stated as such: Given an incomplete and inaccurate distance matrix between a set of n points, where $x_i, i = 1, 2 \ldots, n$ are the positions of the n points in space (we consider the case of 2-D in this report), can we recover their relative positions in space?

While this problem may at first glance seem somewhat unrelated to dimensionality reduction, variants of dimensionality reduction techniques have been shown to have applicability in this space [5],[6]. This is because the problem involves finding a low-dimensional representation of points(since the distance information is generated from a low dimensional space in the first place) that respect the given distance constraints.

It should also be borne in mind that the distance matrix D, where $D_{ij} = \|x_i - x_j\|^2$, can be expressed in terms of the Gram matrix $G = X^T X$.

$D_{ij} = G_{ii} + G_{jj} - 2G_{ij}$.

Therefore, when there is complete and exact distance data, the exact positions of the points can be recovered by performing a matrix decomposition of the Gram or distance matrix. This idea is similar to what the PCA and MDS algorithms perform in practice.

The problem when we have noisy distance data is that the distance information which was corresponding to a lower dimensional space is distorted. Quite possibly, the distance information available to us no longer corresponds to a low dimensional embedding anymore. The dimensionality reduction techniques that do depend on distance information attempt to find low dimensional embeddings that preserve the given distance information as much as possible.

*Electrical Engineering, Stanford University, Stanford, CA 94305. E-mail: pbiswas@stanford.edu

These algorithms need to be evaluated in terms of their sensitivity to inaccurate distance data. In this project, I chose 3 techniques that use only incomplete distance information to find low dimensional embeddings : ISOMAP[7], Laplacian Eigenmaps[8] and Maximum Variance Unfolding[9].

I also considered using other methods PCA, MDS[10] and Locally Linear Embeddings[11],[12] etc, but some of these methods use either complete distance information, or knowledge of point positions in the higher dimensional space. For this project, I will stick to the 3 algorithms mentioned above.

2 Algorithm Descriptions

The ISOMAP algorithm first creates a k nearest neighbor graph and assigns each edge a length that equals the Euclidean distance between the two nodes connected. In our case, if there are less than k neighbors, all the neighbors are used. The second step is to compute the pairwise distance δ_{ij} , for all pairs of nodes i and j, as the length of the shortest paths connecting them on the graph (using Djikstra's algorithm). In the third step, it uses the pairwise distances δ_{ij} as inputs to MDS. More specifically, it computes the Gram matrix G from the distance matrix δ, estimates the dimension r by the number of significant eigenvalues of G, and constructs the low dimensional representations.

The Laplacian Eigenmaps method also begins by creating a k nearest neighbor matrix. However, the edge weights are set to create a weighted Laplacian (V, ϵ).

$W_{ij} = \exp(-d_{ij}^2)/\sigma^2 \quad \forall (i,j) \in \epsilon$.

The eigenvectors of the Laplacian can be used to represent the variation in the geometry of the graph. An optimization problem is solved to find a lower dimensional representation of the points while maintaining the structure of the Laplacian. In particular, the bottom eigenvectors encode most of this information and it turns out that we can extract a lower dimensional embedding from them.

The MVU algorithm attempts to 'unfold' the manifold by pulling the data points apart as far as possible, while faithfully preserving the local distances between nearby input data. It does so by maximizing the distances between all the points while ensuring that the given neighborhood distance relations are satisfied. This problem is formulated as a semidefinite program and the resulting distance matrix is used in the same way as in ISOMAP to obtain a relative map of the points.

It should be noted that in all cases, since only a relative map is obtained, I perform a least squares fitting method at the end to find the best affine mapping that maps the points as closely as possible to the original points. These 3 algorithms offer a reasonably complete picture of dimensionality reduction. ISOMAP is a variant of the MDS based methods that use the top eigenvectors of the Gram Matrix, the Laplacian method looks instead at the connectivity graph and infers structure using the lowest eigenvectors, and MVU is like a bridge between the 2 methods as explored in [13].

3 Results

The input is some mutual distance information between a set of points $x_1, x_2, \ldots, x_n$, generated randomly from a uniform distribution in 2-D space. Only distance data upto a certain radius R is available, that is, we will have distance information only between points which are within the cutoff distance R from each other. The given distance information is further perturbed by a multiplicative Gaussian noise, that is, $d_{ij} = \hat{d}_{ij}(1+\epsilon)$ where d_{ij} is the corrupted distance between x_i and x_j, $\hat{d}_{ij}$ is the true distance, and ϵ is $N(0, \sigma)$. Here when we refer to 10% noise, it means that $\sigma = 0.1$.

The analysis examines how closely the results with the noisy distance data will match the actual point positions. An appropriate choice of the measurement metric might be the RMSD error in the point positions. The number of points n, noise σ and the radius R are varied and its effect on the RMSD error is observed. [1]

[1] Code available at www.stanford.edu/ pbiswas/css229project/

In terms of accuracy, the MVU always outperforms the other 2 methods, followed by ISOMAP, especially when the noise is low(less than 10%) and there is enough distance information($R > 0.3$ for 50 points). It seems that the MVU technique best captures the distance information. ISOMAP and Laplacian Eigenmaps introduce other ideas such as the structure of the neighborhood graph to infer the point positions. While this might be a good idea when the distance information is very unreliable, for more accurate distance data, the eigenmap technique, in particular, is unable to exploit all the information given to us.

The example shown in Figure 1 shows the performance of the algorithms on a random graph of 50 points, in a square region of [-0.5,0.5], using $R = 0.4$ and 10% noise. The red stars are the results with erroneous distance data, the green circles are the actual positions of the points, and the blue lines show the discrepancy between the actual positions and estimated positions. As can be seen, the MVU approach has the best performance. It is, however, the slowest approach as well, since a large SDP is required to be solved.

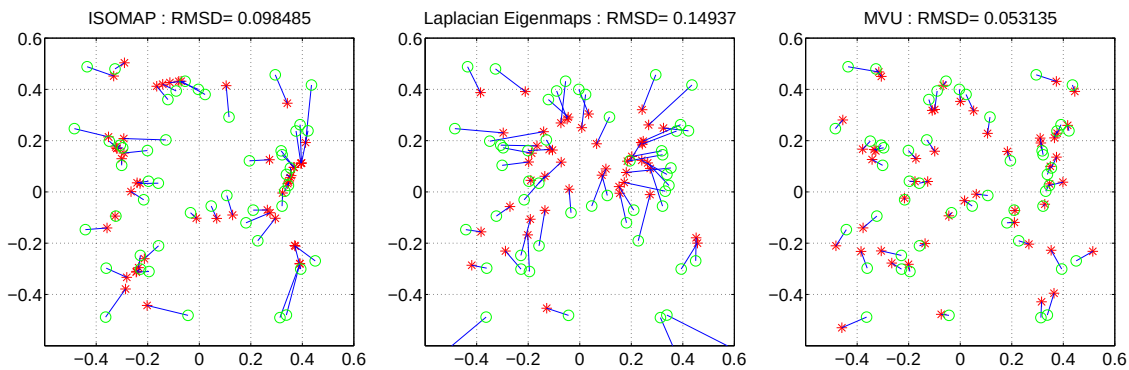


Figure 1: 2 dimensional embeddings for a set of 50 points obtained from distance corrupted by 10% Gaussian noise, Radius=0.4

When the noise is very high(50% or greater), the story is very different. The neighborhood graph is the most reliable for graph structure and the eigenmaps technique works very well. THE ISOMAP technique has the worst performance. This could be because possibly the error propagation is very high in the step where distances between unconnected points is found by using shortest paths, so much so that the distance matrix obtained is simply not valid.

The eigenmaps method, however, is far less susceptible to noise. By taking the smallest eigenvectors of the Laplacian, this method captures the more regular variations in the structure of the graph, just like the lower frequency components in a signal. The higher eigenvectors correspond to the more irregular variations. The assumption is that if the set of points is in a low dimensional space, most of the information will be encoded in the smallest eigenvectors. In some sense, it is the connectivity information that is more criticial than the exact distance information. Infact, for more irregular graph structures, it was observed that the eigenmaps method performed far worse than the other 2 methods.

The example in Figure 2 shows the performance on 250 points with $R = 0.1$, corrupted by 60% noise.

It is interesting to observe how the points which are close to each other tend to cluster together in all these algorithms. As future work, it might be interesting to investigate how to cluster points beforehand, and do the dimension reduction on the a reduced point set corresponding to just the clusters. This could help in reducing the computational times, especially for the MVU.

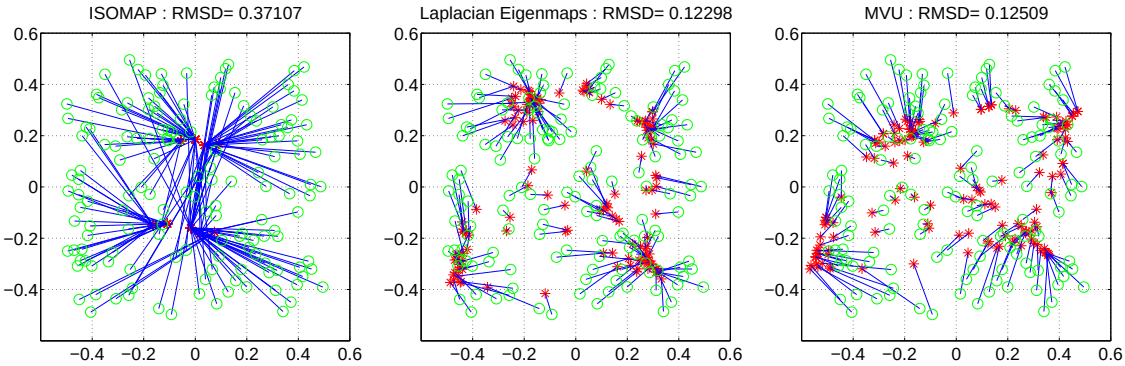
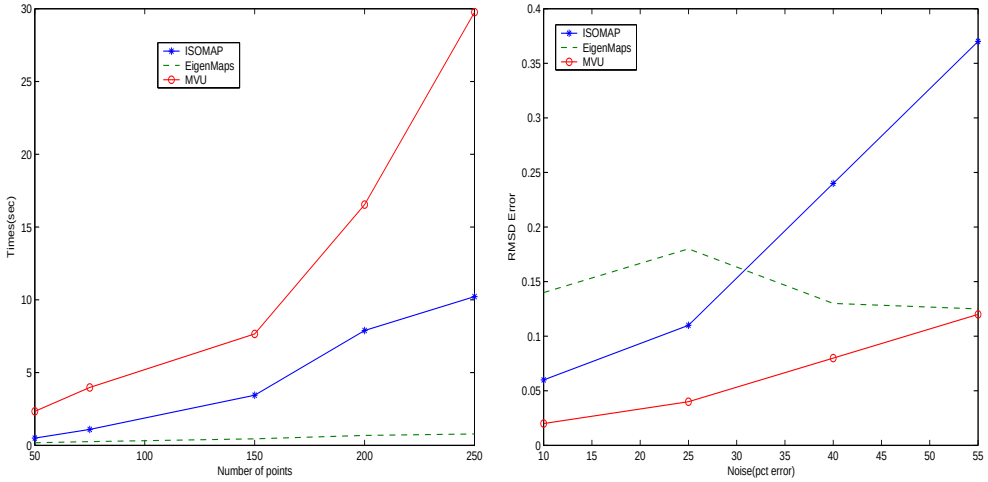


Figure 2: 2 dimensional embeddings for a set of 250 points obtained from distance corrupted by 60% Gaussian noise, Radius=0.1



(a) Computational Times

(b) Noise Sensitivities

While the MVU technique provides very accurate results, the computational effort spent was significantly larger and for very large sets, the computational time was clearly too large to offer any advantage over the other methods. Figure 3(a) shows how the methods scale with the size of the point set. Note that R is not the same for all sets of points, since a high R for a large set corresponds to a far higher connectivity. R is scaled such that connectivity stays with 7-8 on average.

The graph (Figure 3(b))shows how the RMSD error varies for the different methods for a set of 150 points with $R = 0.15$ and varying noise. It captures the sensitivities of the different approaches to noisy distances.

4 Conclusion

Based on the results, I would recommend using MVU when the distance information provided is not too noisy, and the number of points is small. If the number of points is high and computational time and effort are an issue, but the noise is still low, ISOMAP is a better alternative. But if the noise is high and the point set is large, eigenmaps is the best option.

References

[1] Andreas Savvides, Mani Srivastava, Lewis Girod, and Deborah Estrin. Localization in sensor networks. *Wireless sensor networks*, pages 327–349, 2004.

[2] Pratik Biswas, Tzu-Chen Liang, Kim-Chuan Toh, Ta-Chung Wang, and Yinyu Ye. Semidefinite programming approaches to sensor network localization with noisy distance measurements. *to appear in IEEE Transactions on Automation Science and Engineering, Special Issue on Distributed Sensing*.

[3] Gordon Crippen and Timothy Havel. *Distance geometry and molecular conformation*. Wiley, 1988.

[4] Pratik Biswas, Tzu-Chen Liang, Kim-Chuan Toh, and Yinyu Ye. An SDP based approach for anchor-free 3d graph realization. Technical report, Dept of Management Science and Engineering, Stanford University, *submitted to SIAM Journal on Scientific Computing*, March 2005.

[5] Kilian Weinberger, Fei Sha, Qihui Zhu, and Lawrence Saul. Graph regularization for maximum variance unfolding, with an application to sensor localization. In B. Schlkopf, J. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

[6] M. W. Trosset. Applications of multidimensional scaling to molecular conformation. *Computing Science and Statistics*, 29:148–152, 1998.

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

[8] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation, 2002.

[9] W. Weinberger, B. Packer, and L. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization, 2005.

[10] Trevor Cox and Michael A. A. Cox. *Multidimensional Scaling.* Chapman Hall/CRC, London., 2001.

[11] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding, 2000.

[12] D. Donoho and C. Grimes. Hessian eigenmaps: locally linear embedding techniques for high dimensional data. *Proc. of National Academy of Sciences*, 100(10):5591–5596, 2003.

[13] Lin Xiao, Jun Sun, and Stephen Boyd. A duality view of spectral methods for dimensionality reduction. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 1041–1048, New York, NY, USA, 2006. ACM Press.