Nonlinear Dimensionality Reduction for Hyperspectral Image Classification

Proposal

Tim Doster (tdoster@umd.edu)
Advisors:
Dr. John Benedetto (jjb@math.umd.edu)
Dr. Wojciech Czaja (wojtek@math.umd.edu)

October 20, 2010

Abstract

Today with sensors becoming more complex and cost no longer a deterrent to storing large amounts of data; analysts need methods to reduce the volume of stored data and reveal its important facets. Dimensionality reduction, particularly non-linear dimensionality reduction, is a solution to this problem. In this paper, we will look at two nonlinear dimensionality reduction algorithms, Local Linear Embedding and Isomap. These algorithms both have been shown to work well with artificial and real world data sets, but are computationally expensive to execute. We solve this problem for both algorithms by applying landmarks or out of sample extensions. Finally, we will apply these algorithms first to artificial data sets for validation and then to hyperspectral images for the application of classification.

1 Background

Dimensionality reduction is a field of mathematics that deals with the complexities of very large data sets and attempts to reduce the dimensionality of the data while preserving the important characteristics of the data. These algorithms are becoming more important today because the complexity of sensors have increased as well as the ability to store massive amounts of data. For example, hyperspectral sensors, which we will discuss below, record roughly a hundred times the amount of information as a typical optical sensor. With this high number of dimensions being recorded it becomes no longer feasible for analysts to examine the data without the assistance of computer algorithms to reduce the number of dimensions but still keep the intrinsic structure of the data intact.

There are two main branches of dimensionality reduction: linear and non-linear. In this project we will focus on non-linear algorithms as they have been shown to perform at least as well as linear algorithms but in many cases much better. We have chosen to study two of the leading non-linear algorithms, of about fifteen, in the field of dimensionality reduction: Local Linear Embedding and ISOMAP. The details of these algorithms will be presented in following sections.

A hyperspectral image (HSI), in general, has hundreds of spectral bands in contrast to a normal digital image which has three spectral bands (blue, red, and green) and thus offer a more complete part of the light spectrum for viewing and analysis [5]. This high dimensionality makes HSI good candidates for the methods of dimensionality reduction. A regular digital image can be viewed as a collection of three-dimensional spectral vectors, each representing the information for one pixel. Similarly a hyperspectral image can be viewed as a collection of D-dimensional spectral vectors, each representing the information for one pixel. Hyperspectral images typically include spectral bands representing the ultraviolet (200-400 nanometers), visible (400-700 nanometers), near infrared (700-1000 nanometers), and short-wave infrared (1000-4000 nanometers). In Figure 1, a representation of the light spectrum is shown with the approximate coverage of a hyperspectral image.

Thus, HSI are favored over regular images for some applications such as forestry and crop analysis, mineralogy, and surveillance. The spectrum of vegetation, for example, is quite different from that of man-made objects (around 1100-1600 nanometers) even if painted to camouflage in with local vegetation. In this case, a simple photograph would not be able to pick out the man made objects as well as a hyperspectral image. A hyperspectral image can produce a traditional red-blue-green image by resampling

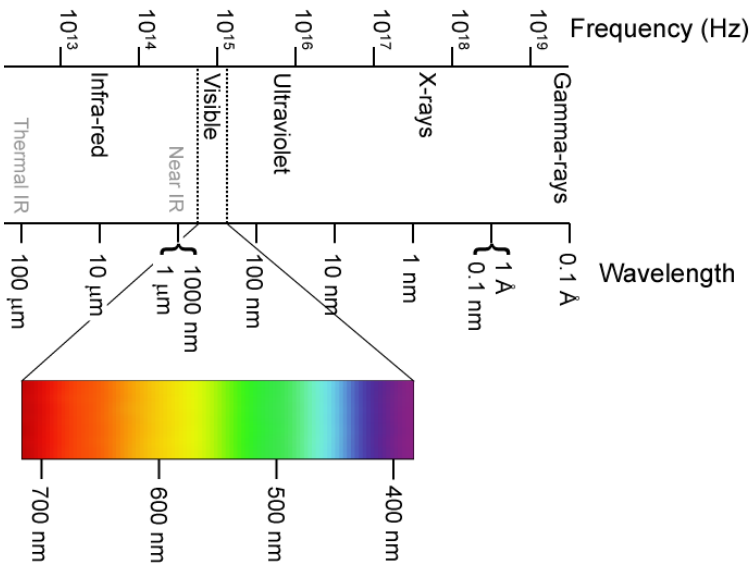

Figure 1: Electromagnetic Spectrum showing the ultra violet, visible, near-infrared, and shortwave infrared

the image using the human visual response or any three spectral bands desired.

HSI are collected with special detectors that can be placed on high structures, flown in planes, or contained in satellites. As the plane traveled, it records the amount solar radiation reflected back from the ground at specific wavelengths line by line (like a push broom) and these are later assembled, with necessary smoothing done to remove effects from the uneven travel of the plane, into a complete hyperspectral image. The sensor onboard the plane works by collecting the emitted solar radiation that is reflected off the ground or object on the ground. As the solar radiation enters the atmosphere, it is altered by the presence of water molecules and other particulate matter in the atmosphere as shown in Figure 2. The same effect happens once the solar radiation is reflected off the ground or object. The data that are recorded by the sensor are known as the radiance spectrum. The reflectance spectrum for a particular band is the ratio of the reflected radiation at that band to the incident radiation at that band, and can be recovered from the collected radiation spectrum by using atmospheric correction equations. In this paper we will use the Quick Atmospheric Correction (QUAC) algorithm found in ENVI to correct any raw images.

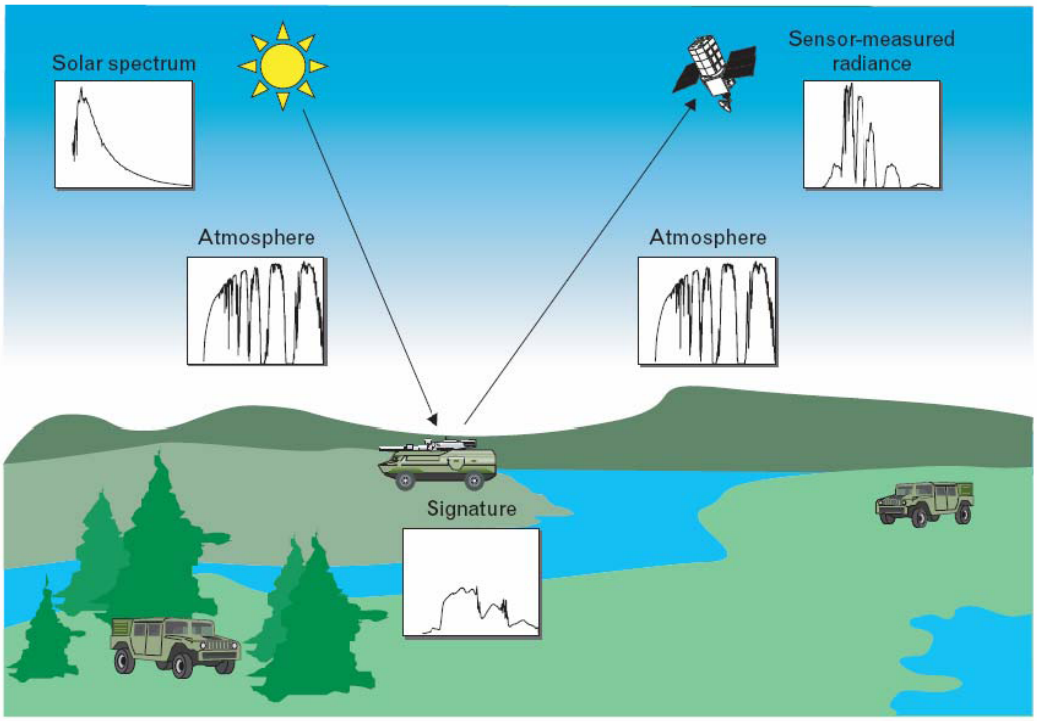

Figure 2: The path of solar radiation from the sun to the hyperspectral sensor (in this case on a satellite) [5]

One of the areas of research into HSI is image classification. The major goal of image classification is to classify the pixels of an image into some number of classes with the use of training data. This allows for example the creation of vegetation maps near wetlands. Bachmann [1] proposes using non-linear dimensionality reduction algorithms to first process the data into a lower dimension before using classification algorithms on the data set. This allows for the similarities and dissimilarities of the data members to become more evident as well as reducing the computation time (though this is greatly offset by the dimensionality reduction algorithm complexities).

2 Approach

2.1 Local Linear Embedding

Local Linear Embedding (LLE) [7, 6] developed by Saul and Roweis is a nonlinear manifold-based approach to dimensionality reduction. LLE seeks to preserve the local properties for each data point when the data is projected to a lower dimension.

The LLE algorithm contains three major steps (note that steps 1 and 2 are often done in unison for efficiency) and proceeds as follows:

Step 0: Let $X = \{X_1, X_2, \ldots, X_N\}$ be a set of vectors (in our case the spectrum of each pixel) with $X_i \in \mathbb{R}^D$. To better utilize memory we will forgo the three dimensional hyperspectral data cube and instead think of the HSI as a two dimensional matrix where the columns represent the pixels and the rows represent the spectral channels.

Step 1: Create a directed adjacency graph, G_K, for the data set X, where X_i is connected to X_j if it is one of the K-nearest-neighbors (KNN) of X_i. Any metric can be used for the KNN calculation but Euclidean (which we will use) and spectral angle are the most common. We will denote the set $U = \{U_i\}_{i=1}^N$ where U_i is the set of pixels that are the KNN of X_i.

Step 2: Calculate the reconstruction weights, W_i, for each X_i by using the cost function:

$$E(W) = \sum_{i=1}^{N} |X_i - \sum_{j \neq i} W_{i,j} X_j|^2.$$

To find $W(i, j)$, the cost function is minimized subject to $W(i, l) = 0$ if $X_l \notin U_i$ and $\sum_{j=1}^{N} W(i, j) = 1$. By forcing the weights to sum to 1 we are removing the effects of translations of points. The use of the cost function ensures that points are not dependent upon rotations and rescaling. Now the set of weights will represent the underlying geometric properties of the data set.

Step 3: Now by use of a similar cost function we will map each X_i to a lower dimensional Y_i. The cost function we are minimizing is:

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

and we minimize it by fixing $W(i,j)$ and optimizing Y_j. Saul and Roweis were able to show that minimizing the cost function is equivalent to finding the $d+1$ smallest eigenvalues, $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{L+1}$, and their corresponding eigenvectors, $V_1, V_2, \ldots, V_{d+1}$ of the matrix $(I-W)^T(I-W)$. We reject the smallest eigenvector as it is the unit vector with eigenvalue 0. Now we use the remaining eigenvectors to project X from N dimensions to d dimensions: $X_i \mapsto (V_2(i), V_3(i), \ldots, V_{d+1}(i))$.

2.2 ISOMAP

ISOMAP [8] developed by Tenenbaum, Silva, and Langford, is a nonlinear manifold based approach to dimensionality reduction like LLE. ISOMAP tries to maintain the geodesic distances between points in the data set when the data is projected down. By focusing on geodesic distance and not the distance in the higher dimensional space ISOMAP is less prone to short circuiting.

ISOMAP contains three major steps:

Step 0: Let $X = \{X_1, X_2, \ldots, X_N\}$ be a set of vectors (in our case the spectrum of each pixel) with $X_i \in \mathbb{R}^D$. To better utilize memory we will forgo the three dimensional hyperspectral data cube and instead think of the HSI as a two dimensional matrix where the columns represent the pixels and the rows represent the spectral channels.

Step 1: Create a directed adjacency graph, G_K, for the data set X, where X_i is connected to X_j if it is one of the K-nearest-neighbors (KNN) of X_i. Any metric can be used for the KNN calculation but Euclidean (which we will use) and spectral angle are the most common. We will denote the set $U = \{U_i\}_{i=1}^N$ where U_i is the set of pixels that are the KNN of X_i.

Step 2: Let G be a graph constructed from the information in G_K. The edge (i,j), the distance from X_i to X_j, will be defined as the pairwise Euclidean distance if $X_j \in U_i$ and if $X_j \notin U_i$ then the distance will be ∞. Now find the shortest pairwise path distances for the graph and update the edge information. To find the pairwise shortest path distance we will use Dijkstra's algorithm.

Step 3: Let S be the matrix corresponding to the graph G and by use of the cost function:

$$\Phi(Y) = \sum_{i,j}^{N} |S_{i,j}^2 - ||Y_i - Y_j||^2,$$

an optimal embedding can be achieved. Tenenebaum showed that minimizing the cost function is equivalent to finding the $d+1$ smallest eigenvalues, $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{L+1}$, and their corresponding eigenvectors, $V_1, V_2, \ldots, V_{d+1}$ of $\frac{1}{2}H^T SH$ where H is the centering matrix. We reject the smallest eigenvector as it is the unit vector with eigenvalue 0. Now we use the remaining eigenvectors to project X from N dimensions to d dimensions: $X_i \mapsto (\sqrt{\lambda_2}V_2(i), \sqrt{\lambda_3}V_3(i), \ldots, \sqrt{\lambda_{d+1}}V_{d+1}(i))$.

2.3 Numerical Difficulties

There are two main numerical challenges in LLE and ISOMAP: finding the KNN for all data points and solving the eigensystem. To find the KNN for each data point a brute search would have complexity $O(N^2)$ but this can be reduced to $O(NLogN)$ using several different more intelligent algorithms. Solving the eigensystem requires complexity of $O(N^2)$. Since these steps cannot be avoided or have their complexity reduced more than noted, we must look to landmarks or out of sample extensions [3] for our dimensionality reduction algorithms. ISOMAP has additional complexity over LLE in that the minimal pairwise distance from for each point must be found which has complexity, using brute force of $O(N^3)$ and with Dijkstra's algorithm of $O(N^2LogN)$.

Another numerical challenge is dealing with very large data sets (as HSI are) efficiently in memory. Most of this can be overcome by handling the HSI as a two dimensional matrix instead of the more natural three dimensional matrix. We can also overcome memory issues by respecting the way that our programming language of choice registers memory and store pixel vectors as such.

2.4 Landmarks

Landmarks work by doing the computationally complex work on a small subset of the data points, which are either chosen at random or intelligently, and then apply their mapping to the other non-landmark points in such a way as to minimize the error between the normal embedding and the landmark embedding. By using landmarks we reduce the complexity of finding KNN to $O(\delta Log\delta)$ and solving the eigensystem in $O(\delta^2)$. Applying the mappings to the other data points will have much less complexity then finding the KNN and solving the eigensystem for all the points. The tradeoff is of course accuracy in the final embedding.

2.5 Software

The algorithms proposed will be encoded in C++ to provide maximum flexibility and inter computer operability. We will make use of the Basic Linear Algebra Subroutines (BLAS) and the Linear Algebra PACK (LAPACK) for C++. These libraries will be used for matrix decomposition and solving eigensystems and are the standard for mathematical and engineering work in C++. We will also use the Approximation Nearest Neighbor [2] package for C++ for finding the nearest neighbors for each pixel in the algorithms. For any prototyping and for running the dimensionality reduction toolbox we will use Matlab (and possibly ENVI/IDL). To read in the hyperspectral images, display hyperspectral images, perform atmospheric calibration (if necessary), and run image classification codes we will use IDL/ENVI..

2.6 Hardware

At this point we are only planning to run these algorithms on a modern desktop PC or laptop so no special hardware is required. If we extend the project to make use of parallel computing there are computers on the math network that have eight and sixteen cores that could be used for testing. More specific information on the hardware used will be included in the mid-year and final report.

3 Validation and Testing

Before we can use the coded versions of our dimensionality reduction algorithms we will need to verify that they are working as the authors of the algorithms intended. We will use three methods to ensure proper implementation.

First, as we code the algorithms we will compare them with the results and structure of those found in the Matlab Dimensionality Reduction Toolbox. This will allow us to diagnose any serious errors in the coding while we are still in the development process.

Once the code is complete and ready for testing we will use two additional validation tests to ensure the proper working order for the algorithms. For these final validation steps we will make use of several topological structures, seen in Figure 3, the swiss roll, broken swiss roll, twinpeaks, and helix, who are defined in three dimensions but are known to lie on a two dimensional manifold (these shapes are defined in the Matlab Dimensionality Reduction Toolbox). Due to the topological nature of these structures, they are perfect for dimensionality reduction testing since we can be sure that the data actually lies in the plane.

The intrinsic dimensionality of a data structure is the approximate dimension of the manifold that the data lies on. The Maximum Likelihood Estimator (MLE) is one such method from the literature that can accurately measure the intrinsic dimensionality of a data set. By using the MLE method we can measure the intrinsic dimensionality of our four artificial data sets before and after dimensionality reduction. By showing that:

$$\lim_{n\to\infty} \frac{I(X_n)}{I(Y_n)} \to 1,$$

where $I(X_n)$ and $I(Y_n)$ are the intrinsic dimensionality of the original data set and the mapping, respectively, we can show that the algorithms performed correctly.

Lastly we will use the trustworthiness ($T(k)$) and continuity ($C(k)$) measures [4]:

$$T(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in U_i^{(k)}} (r(i,j) - k)$$

$$C(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in V_i^{(k)}} (\hat{r}(i,j) - k)$$

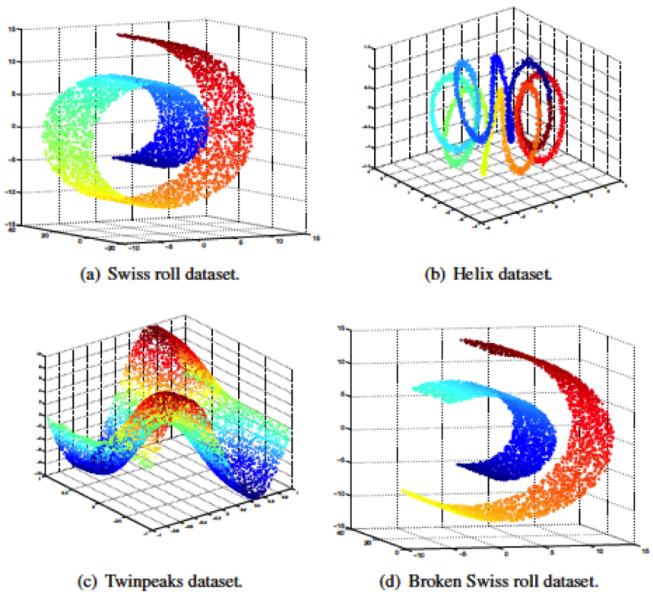

Figure 3: Clockwise from top right: Helix, Broken Swiss Roll, Twin Peaks, Swiss Roll[4].

where $r(i,j)$ is rank of data point $j \in Y_n$ according to pairwise distances between Y_n data points, $U_i^{(k)}$ is the set of points that are among the KNN but not in X_n. Similarly, $\hat{r}(i,j)$ and $V_i^{(k)}$ are defined as the dual of $r(i,j)$ and $U_i^{(k)}$.

We can compare the results of our implementation to the established results found in [4]. The trustworthiness measure ranks the mapping produced on how well it avoids putting points close together in the low dimensional space that are not close in the high dimensional space. The continuity measure, in much the same way as the trustworthiness measure, ranks the mapping on how well it preserves points that are close in the high dimensional space by checking if they are close in the low dimensional space.

As an application of dimensionality reduction we will use the algorithms discussed above for HSI classification. We will compare the classification results of the original data set with those of various dimensionality reduced data sets for each method and with and without landmarks. The results will be compared to ground truth to determine if or how well the dimension reduction algorithms improved the classification results. One would expect from evidence presented in the literature for the dimen-

sion reduced data sets to outperform the original data sets in the image classification application.

4 Data

Artificial data sets for the swiss roll, twin peaks, helix, and broken swiss roll will be used for validation. These data sets can be created to any size that is required for testing.

HSI and ground truth images from the AVIRIS sensor will be used for the application part of the project. The images available are on the order of hundreds of pixels by hundreds of pixels with 150 spectral bands. At this size the images might have to be cropped to allow the dimensionality reduction algorithms to process them before landmarks are introduced. We also have requests in for more extensive data from two past colleagues (this data will not be needed until April). An example of data sets available in presented in Figure 4.

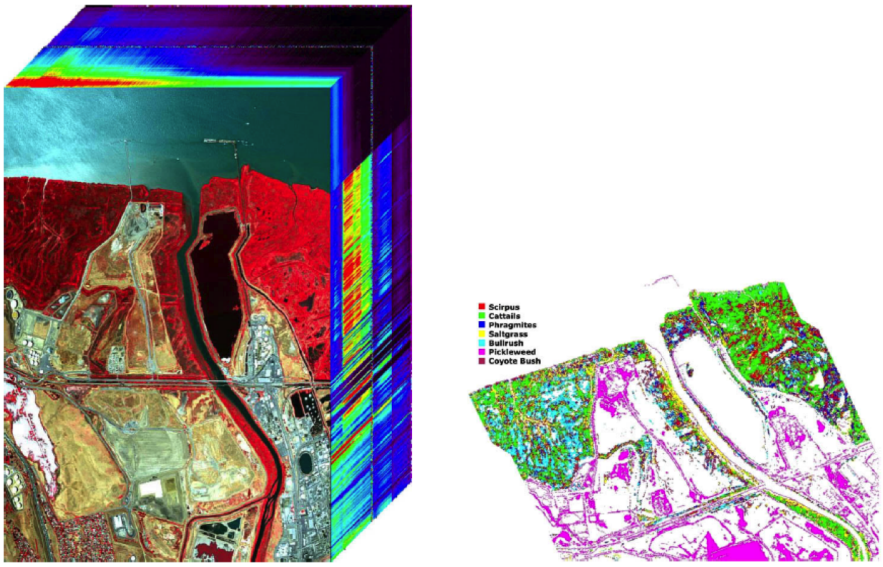

Figure 4: Left is an HSI from the PROBE1 sensor and the right is the corresponding ground truth image showing the position of various types of vegetation.

5 Timeline

•Septemeber and October - Read literature, prototype algorithms and prepare proposal documents.
•October and November - Implement LLE and ISOMAP in C++ and validate the algorithms. This will require learning BLAS, LAPACK and ANN packages. Additionally code for validation must be written.
•December - Prepare end of semester report and presentation.
•January - Write code to link C++ algorithms with IDL/ENVI.
•February and March - Implement landmarks and validate algorithms again with landmarks.
•April - Use algorithms with and without landmarks on hyperspectral classification images.
•May - Prepare final presentation.

5.1 Possible Extensions

Time permitting and after completion of the tasks defined in this proposal document we would like to look at perhaps parallelizing these algorithms by tiling the images among several processors using OpenMP. We may also consider using other nonlinear dimensionality reduction techniques or implementing algorithms to figure out the optimal number of nearest neighbors to select.

6 Milestones

• December 1 - LLE and ISOMAP code has passed the validation tests.
• February 1 - IDL/ENVI can call C++ code and C++ code can return to IDL/ENVI.
• April 1 - Landmark code has passed validation tests.
• May 1 - Results from HSI classification has been obtained.

7 Deliverables

At the end of this year-long course, the goal is to have the ISOMAP and LLE algorithms with landmark points coded up in C++ as well as the necessary code to link the input and output of the algorithms to IDL/ENVI. We will strive to deliver code that is optimized, fully documented, and easily extendable to new applications. At the end of the fall and spring semester we will write a report detailing the work, code, tests, and validation steps performed up to that point and any problems that were encountered. We will also provide detailed steps to reproduce any of the results presented in the reports with the test data that was used.

References

[1] Bachmann, Ainsworth, and Fusina, *Exploting Manifold Geometry in Hyperspectral Imagery*, IEEE Transactions of Geoscience and Remote Sensing **43** (2005), 441–454.

[2] Bengio and Paiement, *ANN: A Library for Approximate Nearest Neighbor Searching*, (2010).

[3] Bengio, Paiement, and Vincent, *Out-of-sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering*, (2003).

[4] Maatan, Postma, and Herik, *Dimensionality Reduction: A Comparative Review*, (2008).

[5] D. Manolakis, D. Marden, and G. Shaw, *Hyperspectral Image Processing for Automatic Target Detection Applications*, Lincoln Laboratory Journal **14** (2003), 79–116.

[6] S. Roweis and L. Saul, *An Introduction of Local Linear Embedding*, Unpublished manuscript, 2001.

[7] L. Saul and S. Roweis, *Reduction by Locally Linear Embedding*, Science **209** (2000), 2323–2327.

[8] Tenenbaum, Silva, and Langford, *A Global Geometric Framework for Nonlinear Dimensionality Reduction*, Science **209** (2000), 2319–2323.