Nonlinear Manifold Learning Part II
6.454 Summary

Erik Sudderth

October 14, 2002

1 Introduction

Manifold learning addresses the problem of finding low–dimensional structure within collections of high–dimensional data. Recent interest in this problem was motivated by the development of a pair of algorithms, *locally linear embedding* (LLE) [6] and *isometric feature mapping* (IsoMap) [8]. Both methods use local, linear relationships to derive global, nonlinear structure, although their specific assumptions and optimization criteria differ. For an introduction to these algorithms, as well as further motivation of the manifold learning problem, see [5].

In this survey, we discuss three manifold learning algorithms which adopt the basic structure of LLE, but attempt to address some of its shortcomings. The first, *Laplacian eigenmaps* [1], is primarily interesting because it provides a new theoretical framework for understanding LLE. This framework points the way to the *Hessian eigenmaps* [4] algorithm, which explicitly attempts to estimate, and minimize, the local curvature of the embedding function. Interestingly, this Hessian extension to LLE is asymptotically correct for a strictly larger class of embeddings than any previously known algorithm. Finally, the *charting* algorithm [2] casts manifold learning as a density estimation problem, thereby adding robustness to noisy or sparsely sampled data.

2 Locally Linear Embedding

We begin by reviewing the locally linear embedding (LLE) algorithm [6, 7]. We are given a set of n data points $x_1, x_2, \ldots, x_n$ in $\mathbb{R}^p$, and hypothesize that, at least approximately, they lie on some smooth manifold $\mathcal{X}$ of intrinsic dimensionality $q < p$. In other words, we assume $\mathcal{X}$ is the image of some coordinate space $\mathcal{Y} \subset \mathbb{R}^q$ under some smooth mapping $\rho : \mathcal{Y} \to \mathbb{R}^p$. We would like to find the coordinates $y_i \in \mathcal{Y}$ of each $x_i \in \mathcal{X}$, thereby recovering a lower–dimensional representation of the data set. Let $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{p \times n}$ and $Y = [y_1, y_2, \ldots, y_n] \in \mathbb{R}^{q \times n}$ denote the matrices of all input points and embedding coordinates, respectively.

Although there are many variants of LLE [7], all of them can be broken down into three stages. In their most basic form, these stages proceed as follows:

1. *Find nearest neighbors:* For each x_i, find the indices $\Gamma(i)$ of that point's nearest neighbors in $\mathbb{R}^q$. Typically, $\Gamma(i)$ is either the set of all points within some ball of fixed radius ϵ, or the K nearest neighbors. Note that the choice of ϵ or K implicitly encodes assumptions about the scale at which the manifold is "locally linear", and can strongly effect LLE's performance.

2. *Determine reconstruction weights:* For each x_i, determine the weights $W_{ij}, j \in \Gamma(i)$, which best reconstruct x_i in terms of its neighbors. LLE solves this problem by minimizing the cost function

$$\Psi_{\text{lle}}(W) = \sum_{i=1}^{n} ||x_i - \sum_{j \in \Gamma(i)} W_{ij} x_j||^2 \qquad \text{subject to} \qquad \sum_{j \in \Gamma(i)} W_{ij} = 1 \qquad \text{for all } i \quad (1)$$

   The form of $\Psi_{\text{lle}}(W)$, together with the normalization constraint, ensures that the optimal weights are invariant to translations, rotations, and scalings of the local neighborhood. Note that this cost function decomposes to provide an independent least–squares optimization for each local neighborhood, which can be solved in closed form.

3. *Determine low–dimensional embedding:* Fixing the weight matrix W from the previous stage, determine embedding coordinates Y which approximately respect the same local neighborhood relationships. LLE solves this problem by optimizing

$$\Phi_{\text{lle}}(Y) = \sum_{i=1}^{n} ||y_i - \sum_{j \in \Gamma(i)} W_{ij} y_j||^2 \qquad \text{subject to} \qquad \sum_{i=1}^{n} y_i = 0 \qquad \frac{1}{n} \sum_{i=1}^{n} y_i y_i^T = I \quad (2)$$

   The constraints effectively fix the translation, orientation, and scale of the embedding coordinates, to which $\Phi_{\text{lle}}(Y)$ is invariant. This cost function can alternatively be written as

$$\Phi_{\text{lle}}(Y) = \text{trace}(Y(I - W)^T(I - W)Y) \quad (3)$$

   where W is the sparse matrix of local reconstruction weights. The optimal solution to this problem chooses the rows of Y as the eigenvectors of $(I-W)^T(I-W)$ with smallest eigenvalues. The smallest eigenvalue is 0, with a constant eigenvector corresponding to the translational degree of freedom. The eigenvectors with the next q smallest eigenvalues then provide the best embedding in $\mathbb{R}^q$.

We shall see that the other manifold learning algorithms discussed in this survey share LLE's basic three–stage structure: find neighbors, estimate local properties of the manifold based on those neighbors, and determine a global embedding which preserves those properties.

3 Laplacian Eigenmaps

In this section, we present an alternative framework for manifold learning known as Laplacian eigenmaps [1]. We begin by presenting the algorithm, and then discuss its justification and relationship to LLE.

1. *Find nearest neighbors:* This stage is identical to LLE, except the neighborhoods are required to be symmetric ($i \in \Gamma(j)$ if and only if $j \in \Gamma(i)$).

2. *Construct weighted adjacency matrix:* Build a symmetric matrix W, where $W_{ij} \neq 0$ if and only if $i \in \Gamma(j)$:

$$W_{ij} = \exp\left\{-\frac{1}{2\sigma^2}||x_i - x_j||^2\right\} \qquad \text{if } j \in \Gamma(i) \tag{4}$$

   Here, σ sets the scale of the isotropic Gaussian kernel used to determine the graph weights. As $\sigma \to \infty$, W approaches the standard, unweighted adjacency matrix.

3. *Compute embedding from normalized Laplacian:* Let D be the diagonal matrix containing the sum of each row/column of the weight matrix W ($D_{ii} = \sum_{j \in \Gamma(i)} W_{ij}$). The symmetric matrix $\tilde{W} = D^{-1/2} W D^{-1/2}$ is then normalized so that $\sum_{j \in \Gamma(i)} \tilde{W}_{ij} = 1$ for all i. Laplacian eigenmaps chooses its embedded coordinates Y to minimize

$$\Phi_{\text{lap}}(Y) = \sum_{i=1}^{n} \sum_{j \neq i} \tilde{W}_{ij} ||y_i - y_j||^2 = \frac{1}{2} \operatorname{trace}(Y^T L Y) \qquad L = I - \tilde{W} \tag{5}$$

   subject to the same non–degeneracy constraints used by LLE (see equation (2)). The matrix L is the *normalized Laplacian* of the graph with weighted adjacency W. As with LLE, the optimum of equation (5) chooses Y to be the eigenvectors corresponding to the q smallest eigenvalues, excluding the constant eigenvector.

The objective function Φ_{lap} (equation (5)) attempts to maps pairs of points x_i, x_j which are nearby, and hence have large weight W_{ij}, to nearby locations y_i, y_j. Heuristically, the normalization $\tilde{W} = D^{-1/2} W D^{-1/2}$ allows variables with different neighborhood densities to be treated equally.

Note that this formulation is closely related to the normalized cuts framework for spectral clustering (see [1] and references therein).

3.1 Choice of Weights: The Laplace–Beltrami Operator

In this section, we discuss a potential justification for the weight function (equation (4)) used by Laplacian eigenmaps. Our presentation is based on [1], but modified and adapted based on additional insights derived from [4].

Suppose we would like to find a smooth, one–dimensional embedding $f : \mathcal{X} \to \mathbb{R}$ of a manifold $\mathcal{X} \subset \mathbb{R}^p$. We assume $\mathcal{X}$ is smooth, so that at every $x \in \mathcal{X}$, the *tangent space* $T_x(\mathcal{X})$ (spanned by vectors tangent to $\mathcal{X}$ at x) is well–defined. Within some neighborhood of x, every point $z \in \mathcal{X}$ has a unique closest point in $T_x(\mathcal{X})$. The tangent space thus inherits a (non–unique) orthonormal coordinate system from the corresponding local coordinates on $\mathcal{X}$.

Given the coordinate system in $T_x(\mathcal{X})$, we may compute the gradient vector $\nabla f(x)$. Although different coordinate systems give different gradients, the norm $||\nabla f(x)||$ is uniquely defined. Furthermore, for any point $z \in \mathcal{X}$, one can show that

$$|f(z) - f(x)| \leq ||\nabla f(x)||\ ||z - x|| + o(||z - x||) \tag{6}$$

Thus, to first order, $||\nabla f||$ measures how far apart f maps nearby points. If our goal is to find a map that best preserves locality on average, a reasonable objective is to minimize

$$\widetilde{\Phi}_{\text{lap}}(f) = \int_{\mathcal{X}} ||\nabla f||^2 \qquad \text{subject to} \qquad ||f|| = 1 \tag{7}$$

Let $\Delta(f) = \sum_i \frac{\partial^2 f}{\partial z_i^2}$, where z_i are the tangent space coordinates, denote the Laplacian operator (called the "Laplace–Beltrami operator" on a manifold). It can be shown that

$$\widetilde{\Phi}_{\text{lap}}(f) = \int_{\mathcal{X}} ||\nabla f||^2 = \int_{\mathcal{X}} \Delta(f) f \tag{8}$$

Thus, the function f minimizing $\widetilde{\Phi}_{\text{lap}}(f)$ must be an eigenfunction of the Laplace–Beltrami operator $\Delta(f)$, or equivalently a member of the null space of the following functional:

$$\mathcal{L}(f) = \int_{\mathcal{X}} (\Delta(f))^2 \tag{9}$$

We will return to this interpretation in the context of the Hessian eigenmaps algorithm.

The objective function (equation (5)) underlying Laplacian eigenmaps may be derived as a discrete approximation to the integral cost of equation (7). In particular, the normalized graph Laplacian L approximates the continuous Laplace–Beltrami operator Δ. The form of the weights W_{ij} may be motivated by viewing Δ as the limit of a heat distribution evolving on the manifold, and using a Gaussian approximation to the corresponding Green's function (see [1] for details). Note that this approximation is only justified when the manifold is very densely sampled.

3.2 Relationship to LLE

As discussed earlier, the first and third steps of the Laplacian eigenmaps algorithm are essentially equivalent to LLE. However, these methods differ in their choice of weight matrix W (step 2). In [1], these differences are analyzed. Let W be the weight matrix estimated by LLE. Using a Taylor approximation, one can show that at each point x_i,

$$(I - W)f(x)\Big|_{x=x_i} \approx -\frac{1}{2} \sum_{j \in \Gamma(i)} W_{ij}(x_i - x_j)^T H_i (x_i - x_j) \tag{10}$$

where H_i is the Hessian of f at x_i. Now suppose that the set of differences to neighboring points $\sqrt{W_{ij}}(x_i - x_j)$ form an orthonormal basis for the tangent space at x_i. It follows immediately that

$$\sum_{j \in \Gamma(i)} W_{ij}(x_i - x_j)^T H_i (x_i - x_j) = \text{trace}(H_i) = \Delta(f) \tag{11}$$

More generally, if the neighboring points are uniformly distributed on any sphere centered at x_i, the expectation of equation (10) is proportional $\text{trace}(H_i) = \Delta(f)$.

From these observations, we see that when the neighbors of each point x_i are placed uniformly in orientation, LLE and Laplacian eigenmaps should produce similar results. However, as data points are sampled more irregularly or asymmetrically, the assumptions underlying the Laplacian approximation become less valid. In these cases, one would expect the local least–squares fits underlying LLE to produce better results. For example, Laplacian eigenmaps uses positive weights, and thus always approximates points by elements of the convex hull of their neighbors, even at manifold boundaries where this is inappropriate.

4 Hessian Eigenmaps

The Hessian eigenmaps framework [4] draws on the Laplacian eigenmap theory discussed in Section 3.1. However, by replacing the Laplacian operator with the Hessian, one can correct a key deficiency of the Laplacian cost functional (equation (9)), producing an algorithm which is guaranteed to asymptotically recover the true manifold under fairly broad assumptions.

In this section, we begin by stating the theoretical results which can be proven for Hessian eigenmaps, relating them to exiting results for other algorithms. We then briefly survey the theory from which these results are derived. We conclude by presenting a discrete implementation of this theory inspired by LLE.

4.1 Asymptotic Convergence Guarantees

The first theoretical convergence guarantees for manifold learning algorithms were shown for the IsoMap algorithm [8]. Recall that a manifold $\mathcal{X} = \rho(\mathcal{Y})$ is the image of some coordinate space $\mathcal{Y}$ under a smooth mapping ρ. Let $d_{\mathcal{X}}(x, x')$ denote the distance of the shortest path between $x, x' \in \mathcal{X}$ which lies entirely within the manifold. Then, assuming an infinite number of data points are drawn from a positive distribution over the manifold, IsoMap will recover the true coordinates (up to a rigid transformation) under the following assumptions:

ISO1: ***Isometry*** For all pairs of points on the manifold, the manifold distance is equal to the Euclidean distance between their corresponding coordinates:

$$d_{\mathcal{X}}(x, x') = ||y - y'|| \qquad \text{for all} \qquad x = \rho(y),\ x' = \rho(y') \tag{12}$$

ISO2: ***Convexity*** The coordinate space $\mathcal{Y}$ is a convex subset of $\mathbb{R}^q$.

Both of these assumptions are directly tied to IsoMap's reliance on multidimensional scaling to embed estimates of geodesic distance in Euclidean space (see [5]).

Donoho and Grimes [3] have investigated the validity of these assumptions in the context of families of images. They argue that while isometry is often a reasonable assumption, the convexity requirement is frequently violated. Their Hessian eigenmaps framework leads to an algorithm which provides the same convergence guarantees under the following weaker assumptions:

LocISO1: ***Local Isometry*** For points $x' \in \mathcal{X}$ in a sufficiently small neighborhood around each point $x \in \mathcal{X}$, geodesic distances $d_{\mathcal{X}}(x, x')$ are identical to Euclidean distances $||y - y'||$ between the corresponding coordinates.

LocISO2: ***Connectedness*** The coordinate space $\mathcal{Y}$ is an open connected subset of $\mathbb{R}^q$.

We outline the proof of this result in the following section.

Currently, no comparable result is known for Laplacian eigenmaps or LLE, i.e. there are no known families of manifolds for which these methods are guaranteed to recover a rigid transformation of the true coordinate space. Based on the theory underlying Hessian eigenmaps, it seems likely that these methods are never guaranteed to be asymptotically correct.

4.2 Theoretical Framework

Consider first the coordinate space $\mathcal{Y} \subset \mathbb{R}^q$. For any twice differentiable function $g : \mathcal{Y} \to \mathbb{R}$, let $H_g^{\text{euc}}(y)$ denote the Hessian matrix, at the point $y \in \mathcal{Y}$, in Euclidean coordinates:

$$\left(H_g^{\text{euc}}\right)_{i,j}(y) = \frac{\partial^2 g(y)}{\partial y_i \partial y_j} \tag{13}$$

We then consider the following functional defined over smooth functions g:

$$\mathcal{H}^{\text{euc}}(g) = \int_{\mathcal{Y}} ||H_g^{\text{euc}}(y)||_F^2 \, dy \tag{14}$$

Here, $||H||_F^2$ denotes the squared Frobenius norm (sum of entries squared) of the matrix H. The nullspace of $\mathcal{H}^{\text{euc}}$ is the set of functions g with everywhere vanishing Hessian. It is straightforward to show that this space equals the span of the constant function and the q coordinate functions $g_i(y) = (y)_i$. Thus, any basis for the nullspace of $\mathcal{H}^{euc}$ will also provide a basis for the underlying coordinate space.

We now consider functions $f : \mathcal{X} \to \mathbb{R}$ defined on the manifold $\mathcal{X} = \rho(\mathcal{Y})$. Analogous to the definitions of the gradient and Laplacian operators in Section 3.1, we may use the tangent space coordinates $T_x(\mathcal{X})$ to compute the Hessian $H_f^{\tan}(x)$ at any $x \in \mathcal{X}$. Although the tangent Hessian will be different for different local coordinate systems, all of these Hessians share the same Frobenius norm $||H_f^{\tan}(x)||_F^2$, so that the following functional is well defined:

$$\mathcal{H}(f) = \int_{\mathcal{X}} ||H_f^{\tan}(x)||_F^2 \, dx \tag{15}$$

The key result of the Hessian eigenmaps framework is that the functionals $\mathcal{H}$ and $\mathcal{H}^{\text{euc}}$ share the same q+1 dimensional nullspace under the correspondence induced by the local isometry ρ (see [4] for details). In other words, a function $f : \mathcal{X} \to \mathbb{R}$ is in the nullspace of $\mathcal{H}$ if and only if $f \circ \rho : \mathcal{Y} \to \mathbb{R}$ is in the nullspace of $\mathcal{H}^{\text{euc}}$.

From the previous discussion, we see that the estimation of a basis for $\mathcal{X}$ is equivalent to the estimation of a basis for the nullspace of the Hessian functional $\mathcal{H}(f)$. It is natural to wonder whether a similar correspondence holds for the Laplacian functional $\mathcal{L}(f)$ (see equation (9)) underlying the Laplacian eigenmap algorithm, and also (approximately) LLE. However, while it is true that all affine functions have zero Laplacian, there exist nonlinear functions that also have zero Laplacian. Thus, while a basis for the nullspace of $\mathcal{L}(f)$ will contain the desired coordinate functions, it will also contain other functions which may distort the resulting embedding.

4.3 Hessian LLE

We now describe a Hessian LLE algorithm [4] which adapts the basic structure of LLE to estimate the nullspace of the functional $\mathcal{H}(f)$ defined in the previous section (equation (15)). As before, the algorithm has three stages:

1. *Find nearest neighbors:* This stage is identical to LLE (nonsymmetric neighborhoods are allowed).

2. *Estimate tangent Hessians:* For each data point x_i, perform a principle components analysis (PCA) of the neighboring points to find the best fitting q–dimensional linear subspace. Project the local neighborhood to this subspace, and use the result to construct a least squares estimate of the local Hessian matrix H_i (see [4]).

3. *Compute embedding from estimated $\mathcal{H}$ functional:* Using the discretization implied by the data points, and the local Hessian estimates H_i, construct a sparse $n \times n$ matrix $\bar{\mathcal{H}}$ which approximates the continuous operator $\mathcal{H}$ (see [4]). As with LLE, we then choose Y to be the eigenvectors corresponding to the q smallest eigenvalues, excluding the constant eigenvector.

Note that $\bar{\mathcal{H}}$ has the same sparsity structure as the matrices used by LLE and Laplacian eigenmaps. However, it differs in that second derivative information is used to estimate the nonzero entries of $\bar{\mathcal{H}}$.

5 Charting a Manifold

We have now explored a theoretical framework for understanding LLE, culminating in a Hessian–based extension of LLE with very attractive asymptotic guarantees. However, the framework we have discussed is purely deterministic, in that it assumes we observe data points which lie exactly on the manifold of interest. For most real data sets, however, we can at best hope that the date is "close" (in a probabilistic sense) to some manifold. Furthermore, although these algorithms rely on empirical estimates of differential operators, they do not consider the noise introduced into these estimates by sparse, inhomogeneous sampling of the manifold.

The charting algorithm [2] addresses these problems by casting manifold learning as a density estimation problem. In particular, charting first fits a mixture of Gaussian densities to the data, and then coordinates the local coordinates implied by each Gaussian's covariance into a single, global system. The density model underlying charting naturally provides a function mapping all coordinates to the high–dimensional manifold, rather than just an embedding of the given data.

Interestingly, charting retains LLE's basic three–step structure, including the attractive property that the optimal solution to each stage may be computed in closed form:

1. *Soft nearest neighbor assignment:* For each x_i, assign a weight W_{ij} to each x_j, $j \neq i$, according to a Gaussian kernel centered at x_i as in the Laplacian eigenmap framework (equation (4)). The bandwidth of the kernel should be chosen as $\sigma \approx r/2$, where r

is the radius over which the manifold is expected to be locally linear. See [2] for an interesting heuristic for automatically estimating r.

2. *Fit Gaussian mixture model:* Let $\mathcal{N}(x; \mu, \Lambda)$ denote a Gaussian density with mean μ and covariance Λ, evaluated at the point x. In charting, we model the high–dimensional data space by an n–component Gaussian mixture, where the component means are set to the observed data points x_i:

$$p\left(x \mid \Lambda\right) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(x; x_i, \Lambda_i) \tag{16}$$

Here, Λ denotes the covariances Λ_i placed around the n data points. The maximum likelihood covariance estimate would shrink all of the variances Λ_i to zero. To avoid this degenerate solution, charting places the following prior on the local covariances:

$$p\left(\Lambda\right) = \alpha \exp\left\{ - \sum_{i \neq j} W_{ij} D(\mathcal{N}(x; x_i, \Lambda_i) \,||\mathcal{N}(x; x_j, \Lambda_j)) \right\} \tag{17}$$

Here, $D(p||q)$ denotes the Kullback-Leibler divergence. This prior encourages neighboring Gaussian densities, as determined by the weights W_{ij}, to span similar subspaces.

The MAP covariance estimate may be determined in closed form by solving a coupled set of linear equations (see [2]). This estimate brings nonlocal information into the estimation of the local coordinate frames defined by the Λ_i matrices, ensuring that neighboring coordinates (or "charts") span similar subspaces.

3. *Connect local charts:* Suppose we would like to find an embedding in $\mathbb{R}^q$. For the k^{th} mixture component $\mathcal{N}(x; x_k, \Lambda_k)$, let $U_k = [u_{k1}, \ldots, u_{kn}]$ denote the projection of the n data points onto the q–dimensional subspace spanned by the q dominant eigenvectors of Λ_k. For each chart, we would like to determine a low–dimensional affine projection $G_k \in \mathbb{R}^{q \times q+1}$ which maps these points into the global coordinate frame. We couple these projections by requiring them to agree on data points for which they share responsibility, as encoded by the following objective:

$$\Phi_{\text{chart}}(G) = \sum_{k \neq j} \sum_{i=1}^{n} p_k(x_i) p_j(x_i) \left|\left| G_k \begin{bmatrix} u_{ki} \\ 1 \end{bmatrix} - G_j \begin{bmatrix} u_{ji} \\ 1 \end{bmatrix} \right|\right|_F^2 \tag{18}$$

Here, $p_k(x_i)$ is the posterior probability (as defined by the mixture model selected in step 2) that x_i was sampled from $\mathcal{N}(x; x_k, \Lambda_k)$.

The objective function of equation (18) may be rewritten as

$$\Phi_{\text{chart}}(G) = \text{trace}(GQQ^TG^T) \tag{19}$$

for an appropriate matrix Q (see [2]). Thus, as with LLE, the optimal embedding may be found by finding the bottom eigenvectors of an $n \times n$ symmetric matrix. This matrix may be made sparse by approximating very small posterior probabilities $p_k(x_i)$ to be zero.

As shown by examples in [2], charting may perform well on sparsely sampled manifolds which cause problems for other methods like LLE. However, while the different cost criteria underlying charting seem reasonable, there is currently no framework for determining the situations under which it will recover the true manifold geometry. Furthermore, although both charting and Laplacian eigenmaps use a Gaussian kernel function to define local neighborhoods, the relationship between their uses of this neighborhood has yet to be explored.

References

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003.

[2] M. Brand. Charting a manifold. In *Neural Information Processing Systems 15*. MIT Press, 2003.

[3] D. L. Donoho and C. Grimes. When does Isomap recover the natural parameterization of families of articulated images? Technical Report 2002-27, Stanford Statistics Department, 2002.

[4] D. L. Donoho and C. Grimes. Hessian eigenmaps: New locally linear embedding techniques for high–dimensional data. Technical Report 2003-08, Stanford Statistics Department, 2003.

[5] A. Ihler. Nonlinear manifold learning. MIT 6.454 Summary, 2003.

[6] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[7] L. K. Saul and S. T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

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