Iterative Hyperplane Merging: A Framework for Manifold Learning

Harry Strange
hgs08@aber.ac.uk
Reyer Zwiggelaar
rrz@aber.ac.uk

Department of Computer Science, Aberystwyth University, Aberystwyth, SY23 3DB, Wales, UK

Abstract

We present a framework for the reduction of dimensionality of a data set via manifold learning. Using the building blocks of local hyperplanes we show how a global manifold can be reconstructed by iteratively merging these hyperplanes. A Minimum Spanning Tree provides the skeleton needed to traverse the manifold so that the local hyperplanes can be used to build a global, locally stable, embedding. We show state of the art results when compared against existing manifold learning approaches using benchmark synthetic data. We also show how our technique can be used on real world image data.

1 Manifold Learning

The area of dimensionality reduction has received much attention over the last few years, thanks in part to the growth in the number of non-linear, or manifold learning, techniques. At its core, any dimensionality reduction algorithm takes a set of high dimensional samples and returns a representation of lower dimensionality that retains certain features found in the high dimensional space. In the simple case a dimensionality reduction algorithm will project the data onto the basis of a global feature, such as the hyperplane of maximum variance (i.e. Principal Components Analysis [9]). More complex algorithms will aim to recover a low-dimensional non-linear manifold embedded in the high dimensional space (e.g. ISOMAP [16]).

It is to this family of manifold learning algorithms that much focus has been given in the computer vision and pattern recognition communities over the past decade. Many problems in these fields require a low dimensional representation to be found as working in higher dimensions can often be problematic [2]. Dimensionality reduction and manifold learning techniques have also been used to reveal patterns in image data [21]. As an example, consider a data set consisting of a sequence of images showing a rotating 3-dimensional object. Across the dataset the object rotates around one of its axes. If each of these images were to be thought of as a point in high-dimensional space (the dimensionality being equal to the number of pixels in the image) then they would lie on a simple circular manifold that is parameterized by the degree of rotation of the object. This means that each image can be discriminated using only 1-dimension - the degree of rotation - as opposed to, in the case of a 128×128px image, $16,384$ dimensions. This reduction of dimensionality overcomes many computational and mathematical problems associated with high-dimensional learning [2, 6].

BMVC 2010 doi:10.5244/C.24.18

The simplest form of dimensionality reduction is one in which the non-linearity of the data is ignored and the entire data is projected onto a single linear basis. This approach, known as Principal Components Analysis (PCA), was first developed by Hotelling in 1933 [9] and has been widely used since (e.g [10, 13]). It finds the global hyperplane of maximum variance across the data and projects all points onto the low-dimensional basis vectors of this hyperplane. A covariance matrix is constructed across all the samples and the eigenvectors of this matrix provide the linear basis from which the low-dimensional projection matrix can be formed. If the data set is inherently linear then PCA will work well, but few real world data sets are purely linear and as such PCA will fail to find an optimal embedding of the data. This linear limitation of PCA has lead to much research into non-linear, or manifold learning, techniques (e.g. [14, 16, 22]). All of these techniques work on the assumption that the data lie on a learnable low-dimensional manifold embedded within the high-dimensional input space. For example, ISOMAP [16] constructs a geodesic distance graph across the data to approximate distances across the manifold. The eigenvectors of this geodesic distance matrix form the low-dimensional basis upon which the high-dimensional data can be mapped. Since ISOMAP considers the distance information across the manifold it is seen as a global method for manifold learning (as it maintains a global property - the geodesic interpoint distances). Conversely there exist methods which seek to maintain local properties of the data, such as Locally Linear Embedding [14] which aims to maintain local interpoint relations between the high and low-dimensional spaces. Recently a class of manifold learning algorithm has emerged that seeks to combine the strengths of global and local techniques. This family of global-local techniques aim to preserve both local and global properties of the data. One such example is Local Tangent Space Alignment (LTSA) [22]. LTSA constructs local models around each sample based on its local tangent space. These tangent spaces are then globally aligned via the solution of a minimization problem to produce the low-dimensional embedding of the high-dimensional data. A more recent suite of techniques proposed by Goldberg *et al*. in [7] use PCA and Procrustes analysis to build and align local models. The main algorithm presented, Greedy Procrustes (GP), finds the embeddings of neighborhoods iteratively using PCA and then aligning them to neighboring neighborhoods using Procrustes analysis.

Our technique, Iterative Hyperplane Merging, continues this idea of globally aligning local models using PCA and Procrustes. A clustering algorithm is used to partition the data into local models. PCA is run on these local models to produce local low-dimensional hyperplanes. These local hyperplanes are then iteratively merged to produce a global alignment of the local models. As such Iterative Hyperplane Merging is able to preserve the local properties of the data across a global scale.

2 Iterative Hyperplane Merging

Iterative Hyperplane Merging (IHM) can be intuitively thought of as a form of local PCA where PCA is applied at a local scale to produce low-dimensional local hyperplanes. These local hyperplanes are then globally aligned to produce the final low-dimensional embedding. A clustering algorithm is employed to create the partitions needed to form the local hyperplanes and a Minimum Spanning Tree (MST) is used as the basis for forming the global alignment of the hyperplanes.

The clustering step is used to capture local information and form the local hyperplanes. The lack of overlap between local models is novel when compared against many existing

local-global techniques (e.g. [5, 7, 22]) and aims to reduce any within model distortion. This reduction in local distortions leads to a more locally faithful embedding.

To obtain a faithful global embedding we perform a pre-order traversal on the Minimum Spanning Tree (MST) of the inter-hyperplane distance graph. When walking from one node in the MST to another we merge the hyperplanes gradually building a global embedding of the data. The Minimum Spanning Tree has many properties that are well suited to our algorithm. Firstly, as outlined by Robins in [13], the MST provides all the information needed to describe the connectedness of the data. Since topology can be, at least simplistically, thought of as the connectedness of a data the MST provides a good approximation for the topology of a manifold. However, there is insufficient metric information contained within the MST to be used as a geodesic graph for isometric techniques (e.g ISOMAP[16]). Secondly, the non-cyclic nature of the MST ensures that when traversing the tree we won't get caught in any local cycles and every vertex in the tree will be visited in the correct 'topological order'.

2.1 Algorithm

We take as input a high-dimensional set of samples $\mathbf{X} = \{x_i\}_{i=1}^n \in \mathbb{R}^q$ sampled from a low-dimensional manifold $\mathscr{M} \in \mathbb{R}^p$ embedded within $\mathbb{R}^q$ (where $p \ll q$). The goal of any manifold learning algorithm is to recover a set of samples $\mathbf{Y} = \{y_i\}_{i=1}^n \in \mathbb{R}^p$ from $\mathbf{X}$ that best approximate the p-dimensional manifold $\mathscr{M}$. We assume that at a local scale the manifold $\mathscr{M}$ is homeomorphic to Euclidean space and is a C^∞-manifold (i.e. it is smooth differentiable).

The q-dimensional set of samples $\mathbf{X}$ can be partitioned into k-local hyperplanes by initially using any clustering algorithm to partition the data. Throughout the rest of this paper we will be using either a Gaussian Mixture Model clustering scheme or a constrained k-means clustering algorithm so we will briefly outline the basic methodology of each here[1]. For Gaussian Mixture Modelling we are interested in a particular Gaussian Mixture Model (GMM) where the number of components is equal to n and the parameters are defined as $\Theta = \{\Theta_{i|c}, i = 1,\ldots,n\}$ where $\Theta_{i|c}$ is the mean and covariance matrix of the ith Gaussian density function and $c = 1,\ldots,k$. The output of the likelihood GMM function related to a partition ω_c is a weighted sum of n component densities:

$$p(x \mid \omega_c) = \sum_{i=1}^{n} P(\Theta_{i|c} \mid \omega_c) p(x \mid \omega_c, \Theta_{i|c}) \tag{1}$$

where $P(\Theta_{i|c} \mid \omega_c) = \mu_{ic}$ is the prior probability of the ith component parameter. These mixture parameters are chosen such that $\sum_i \mu_{ic} = 1$. By fixing the means and covariance matrices of each partition we can now assign a sample to partition ω_l if:

$$\omega_l = \arg\max_{y_c} p(x \mid \omega_c) \tag{2}$$

The weights $\mu = \{\mu_{ic}\}$ of the GMM functions $p(x \mid \omega_l), l = 1,\ldots,k$ can be found by solving a constrained minimization function [15]. For constrained k-means we wish to find a set of partitions, $\omega_1, \omega_2, \ldots, \omega_k$, such that the distance between each point, x_i, and its nearest

[1] For a full description of both of these algorithms we refer the reader to [2, 15] for Gaussian Mixture Modelling and [6, 21] for constrained k-means clustering.

partition center, $\bar{\omega}_c$, is minimized:

$$\min_{\omega_1,\ldots,\omega_k} \sum_{i=1}^{n} \min_{h=1,\ldots,k} \left(\frac{1}{2} \parallel x_i - \omega_h \parallel_2^2 \right) \tag{3}$$

with the specific constraint that no cluster, ω_c, is smaller than the minimum cluster size, h, $\mid \omega_c \mid \geq h$.

The clustering step described above produces a partitioning, Ω, such that $\bigcup \Omega = \mathbf{X}$ and for any two distinct partitions, $\omega_i \in \Omega$ and $\omega_j \in \Omega$, $\omega_i \cap \omega_j = \varnothing$. The partitioning should be chosen such that as far as possible $\mid \omega_1 \mid \approxeq \mid \omega_2 \mid \approxeq \ldots \approxeq \mid \omega_k \mid$ although the ability to achieve this is heavily dependant on the choice of clustering algorithm. Since we assume that the manifold $\mathscr{M}$ is locally linear these partitions can be used to find the k-local hyperplanes of the data set. We define a hyperplane as

$$\Pi_i = \{ \forall x \mathbf{U}\mathbf{U}^T ; \bar{\omega}_i \mid x \in \omega_i, \lambda \mathbf{U} = \mathbf{C}\mathbf{U} \} \tag{4}$$

where $\bar{\omega}_i$ is the mean of the samples in ω_i, $\mathbf{C}$ is the covariance matrix of the samples in ω_i and $\mathbf{U}$ is a matrix containing as columns the top p-dimensional eigenvectors sorted according to their associated eigenvalues, λ. The k-hyperplanes are therefore now intrinsically p-dimensional but are still embedded within the q-dimensional space. To globally align these local hyperplanes we first need to find their local connectivity. To do this we construct a Minimum Spanning Tree (MST) [8] across the data using the means of the hyperplanes as vertices for the tree. This ensures that all vertices are connected with minimum cost. The MST is found by firstly connecting all vertices to form a dense graph $G =< V, E >$ where E is the edgelist connecting all vertices, and the vertex list $V = \{\bar{\Pi}_i\}_{i=1}^{k}$ (where $\bar{\Pi}_i$ is the mean of the hyperplane Π_i). The MST, $T =< V, E' >$, is then a subgraph of G with the same vertex set V but a reduced edge set $E' \subset E$ (where E' is the edge set of minimal cost) [2]. T now provides us with an approximation of the topology of the hyperplanes and thus a coarse approximation of the topology of $\mathbf{X}$ (since there is a direct mapping between the connectivity of the hyperplanes and the connectivity of $\mathbf{X}$).

We can now use T to find the global connectivity of the hyperplanes by walking along T merging hyperplanes across each step. To walk along the MST we use a simple pre-order traversal [19] which ensures that parents are visited before children, and siblings are visited in left-to-right order. To describe the process of pre-order traversal we denote the first child of a node v as first$[v]$. next$[v]$ denotes the next sibling of node v, last$[v]$ denotes the last child of node v and size$[v]$ denotes the number of nodes in the subtree of T rooted at v for all $v \in V$. order$[v]$ gives the order in which v is to be visted. So we visit the first node with order$[v] = 1$, then order$[w] = 2$, until we reach order$[z] = k$. Given a random node, r, set at the root node for traversal, a bijection order $\Psi : V \rightarrow \{1, \ldots, k\}$ is a pre-order traversal of T if order$[r]$ = 1 and

- order[first$[v]$] = order$[v] + 1$ *(if v is not a leaf)*
- order[next$[v]$] = order$[v]$+size$[v]$ *(if v is not a last child)*

for all $v \in V$. Given this bijection order Ψ and the direct mapping between hyperplanes and our MST we can say that Π_{Ψ_1} is the first hyperplane to be visited in the pre-order traversal and Π_{Ψ_k} is the kth hyperplane to be visited. We denote $\Pi_{\Psi_a} \rightarrow \Pi_{\Psi_b}$ as the traversal from the

[2] We omit the full algorithmic outline for forming a MST. For a more detailed description we refer to [8]

ath hyperplane to the bth hyperplane. Since we wish to merge hyperplanes when traversing from one to another we define a function

$$f(\Pi\Psi_a \rightarrow \Pi\Psi_b) = \mathbf{A}\mathbf{U}_b\mathbf{U}_b^T + (\bar{\Pi}_b - (\bar{\Pi}_b\mathbf{U}_b\mathbf{U}_b^T)) \tag{5}$$

which maps the hyperplane $\Pi\Psi_a$ onto the hyperplane $\Pi\Psi_b$, where $\mathbf{A} = \forall x \in \Pi_a$ and $\lambda_b\mathbf{U}_b = \mathbf{C}_b\mathbf{U}_b$. Since the mapping is cumulative, with a new local hyperplane being added to the global embedding at each iteration, we can define the globally aligned embedding as a matrix augmentation:

$$\mathbf{Y} \leftarrow f(\Pi\Psi_{1\ldots(k-1)} \rightarrow \Pi\Psi_{(k)}) \leftarrow f(\Pi\Psi_{1\ldots(k-2)} \rightarrow \Pi\Psi_{(k-1)}) \leftarrow \ldots \leftarrow f(\Pi\Psi_1 \rightarrow \Pi\Psi_2) \tag{6}$$

It is worth noting at this point an optional extra step that can be used to increase the robustness of an embedding. During the walk along T if last$[v]$ is reached then we need to backtrack until we reach v. Backtracking involves visiting and projecting onto hyperplanes we have already visited. Normally each backtracking step is treated as the same as a forwarding step and the above function $f(\Pi\Psi_{a'} \rightarrow \Pi\Psi_a)$ is used (where $\Pi\Psi_{a'}$ is the image of $\Pi\Psi_a$ having gone through iterative projections). However, $f(\Pi\Psi_{a'} \rightarrow \Pi\Psi_a) \neq f(\Pi\Psi_a \rightarrow \Pi\Psi_{a'})$ since the projection matrix that moves us from the linear subspace of $\Pi\Psi_{a'}$ onto $\Pi\Psi_a$ is orthogonal to $\Pi\Psi_{a'}$. This means that an extra alignment step is needed so that $f(\Pi\Psi_{a'} \rightarrow \Pi\Psi_a) \approx f(\Pi\Psi_a \rightarrow \Pi\Psi_{a'})$. The alignment step applies simple Procrustes analysis[11] to translate, scale and rotate $\Pi\Psi_{a'}$ to align to $\Pi\Psi_a$. Once the translation vector, scale value and rotation matrices have been found we can align the embeddings by adding an extra constraint to the function:

$$f(\Pi\Psi_{1,\ldots,a'} \rightarrow \Pi\Psi_a) = \mathbf{b}(\mathbf{A}\mathbf{U}_a\mathbf{U}_a^T + (\bar{\Pi}_a - (\bar{\Pi}_a\mathbf{U}_a\mathbf{U}_a^T)))\mathbf{T} + \mathbf{v} \tag{7}$$

where $\mathbf{b}$ is the isomorphic scale value, $\mathbf{T}$ is the rotation matrix and $\mathbf{v}$ is the translation vector.

Once the traversal algorithm outlined above has finished, with or without backtracking, we are able to obtain the final low-dimensional embedding by performing PCA on the matrix $\mathbf{Y}$

$$\Lambda\mathbf{V} = \mathbf{C}_\mathbf{Y}\mathbf{V} \tag{8}$$

where $\mathbf{C}$ is the covariance matrix of $\mathbf{Y}$ and $\mathbf{V}$ is a matrix containing as columns the top p-dimensional eigenvectors sorted according to their associated eigenvalues, Λ.

3 Results

We use synthetic data to analyse the performance of our algorithm on a dataset with a known manifold. A real world image database is then used to show how our algorithm can handle more difficult manifolds. All our experiments are performed using the MATLAB programming environment. For Gaussian Mixture Modelling we use a MATLAB version of Bouman's C implementation [4] which has the added benefit of being unsupervised so will calculate the optimal number of clusters for a given dataset (with the only parameter being an initial guess of the number of clusters needed).

3.1 Synthetic Data

For synthetic data we use the benchmark Swiss Roll data set [16]. It consists of a highly curved 2-dimensional plane wrapped to a Swiss Roll in 3-dimensional space (See Figure 2(a)). Gaussian Mixture Modelling is used to form the partitions needed to create the local hyperplanes and the backtracking step described in Eq. (7) is used. We compare our technique against three widely used algorithms: Principal Components Analysis (PCA), ISOMAP and Local Tangent Space Alignment (LTSA). A visual comparison of how these algorithms perform acroos a range of noise levels is shown in Figure 2. PCA (Fig. 2(e) - Fig. 2(h)) fails to find the underlying manifold at all noise levels since it is simply applying a linear projection to the data. ISOMAP (Fig. 2(i) - Fig. 2(l)) performs better and at all noise levels is able to unwrap the Swiss Roll, although it does introduce unwanted holes in the manifold so the topology, at least at a local scale, is distorted. LTSA performs well when no noise is present (Fig. 2(m)) but fails to find the correct embedding when significant amounts of noise are added (Fig. 2(o) - Fig. 2(p)). IHM (Fig. 2(q) - Fig. 2(t)) consistently performs well managing to find the global shape of the manifold as well as preserving local relations (although the neighborhoods are somewhat squeezed and expanded at the highest noise level).

Three error measures are used to analyse the performance of a manifold learning technique at maintaining both local and global properties of the data. Mean relative rank errors, trustworthiness and continuity [17, 20], are used to measure the local stability as they compare the differences between local neighborhoods in both the high and low dimensional spaces. Intuitively trustworthiness measures the number of samples that exist in a neighborhood in the high-dimensional space but not in the same low-dimensional neighborhood. Continuity measures the number of samples that have entered the low-dimensional neighborhood and do not appear in the same high-dimensional neighborhood. As such they are a good measure of the local connectivity of the data as they measure the amount of change at

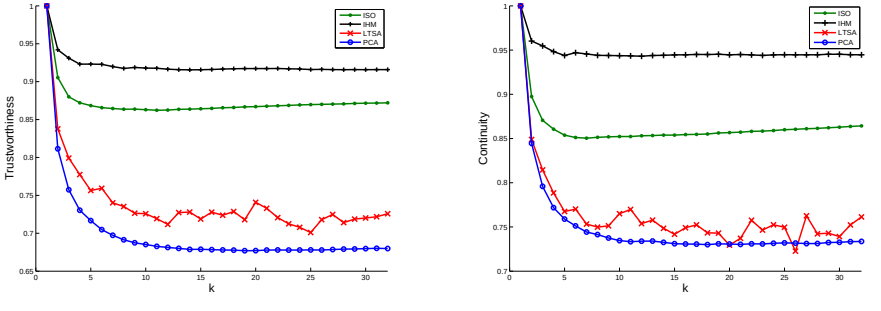


Figure 1: Graph of results for the trustworthiness (left) and continuity (right) of different algorithms when trying to unroll the Swiss Roll dataset with 2000 samples. The neighborhoods for Isomap and LTSA were averaged over the range $k = [2, 32]$. For IHM we use GMM to find optimal cluster sizes with an initial cluster size estimate of 32.

Algorithm	Trustworthiness	Continuity	Mean Square Error
PCA	0.70 ($\pm$ 0.06)	0.75 ($\pm$ 0.05)	0.62 ($\pm$ 0)
ISOMAP	0.87 ($\pm$ 0.02)	0.86 ($\pm$ 0.03)	0.24 ($\pm$ 0.24)
LTSA	0.74 ($\pm$ 0.05)	0.76 ($\pm$ 0.05)	0.23 ($\pm$ 0.36)
IHM	**0.92** ($\pm$ 0.02)	**0.94** ($\pm$ 0.01)	**0.02** ($\pm$ 0.02)

Table 1: Results on Swiss Roll with 2000 samples. The neighborhoods for Isomap and LTSA were averaged over the range $k = [2,32]$. For IHM we use GMM to find optimal cluster sizes with an initial cluster size estimate of 32.

a local scale. The trustworthiness of an embedding is measured by

$$\mathscr{T} = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in U_k(i)} (r(i,j) - k) \tag{9}$$

Similarly continuity is measured by

$$\mathscr{C} = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in V_k(i)} (\hat{r}(i,j) - k) \tag{10}$$

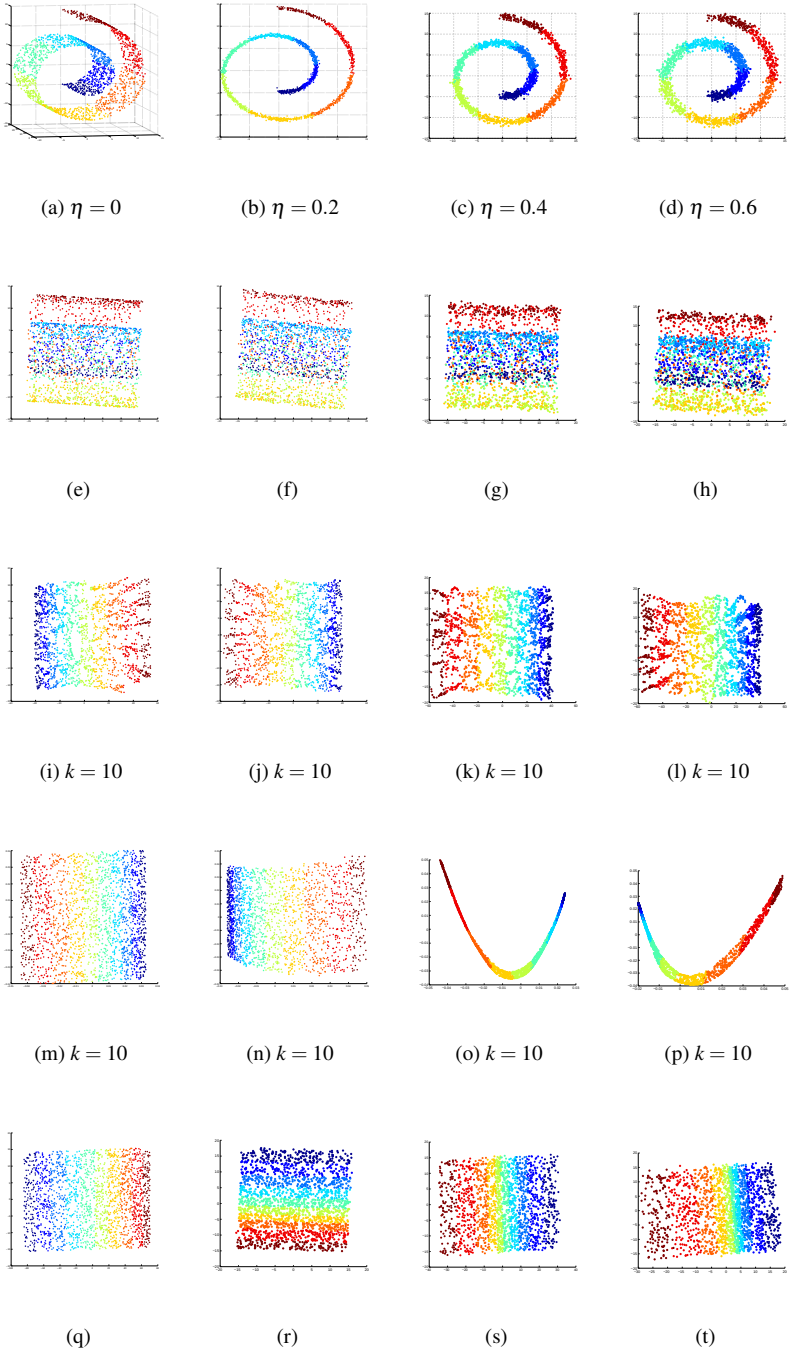
where n is the total number of samples and k is the size of the neighborhoods we wish to measure the trustworthiness and continuity of. $r(i,j)$ is the rank of the data sample $\mathbf{X}_j$ sorted according to the Euclidean distance from sample $\mathbf{X}_i$ in the high-dimensional space, similarly $\hat{r}(i,j)$ is the rank of the data sample $\mathbf{Y}_j$ sorted according to the Euclidean distance from $\mathbf{Y}_i$ in the low-dimensional space. U_k is the set of all samples that in the k-neighborhood of i in the low-dimensional space but not in the high-dimensional space. V_k is the set of samples that are in the k-neighborhood of i in the high-dimensional space but not in the low-dimensional space. Figure 1 shows that over a range of neighborhood sizes IHM is locally stable with the values averaging at $\mathscr{T} = 0.92$ and $\mathscr{C} = 0.94$ (see Table 1). When compared against other techniques IHM, like ISOMAP, is able to consistently maintain local relations with a low variation between embeddings.

Procrustes analysis [11] is used to measure the global stability of an embedding. The embedding, as well as the original unwrapped data, is centered and scaled into a 1×1 square. Procrustes analysis then applies scaling, translation, reflection and rotation in order to minimize the squared distances between an embedding and the original unwrapped data. The lower the sum of squared distances the closer the embedding is to the original data. As shown in Table 1 IHM provides a significant improvement over existing techniques for maintaining global stability. IHM is consistently able to find a globally faithful embedding with low variation between embeddings, as opposed to ISOMAP and LTSA where there is a high variation between embeddings.

3.2 Image Data

To test our algorithm on real world image data we use the Frey faces dataset[3]. The data contains 1965 images of size 20×28 pixels taken from sequential frames of a small video. We use constrained k-means clustering to partition the data as the computational complexity and running time is much reduced when using k-means in high-dimensional space. We also

[3]Frey faces dataset available from http://cs.nyu.edu/~roweis/data.html



(a) $\eta = 0$ (b) $\eta = 0.2$ (c) $\eta = 0.4$ (d) $\eta = 0.6$

(e) (f) (g) (h)

(i) $k = 10$ (j) $k = 10$ (k) $k = 10$ (l) $k = 10$

(m) $k = 10$ (n) $k = 10$ (o) $k = 10$ (p) $k = 10$

(q) (r) (s) (t)

Figure 2: (a) 3D Swiss Roll data set with zero added noise, (b-d) the same data with varying uniform noise shown in 2D projection. (e-h) PCA embeddings, (i-l) ISOMAP embeddings with $k = 10$, (m-p) LTSA embeddings with $k = 10$, (q-t) IHM embeddings using GMMs with automatic parameter estimation and an initial cluster size estimate of 32.

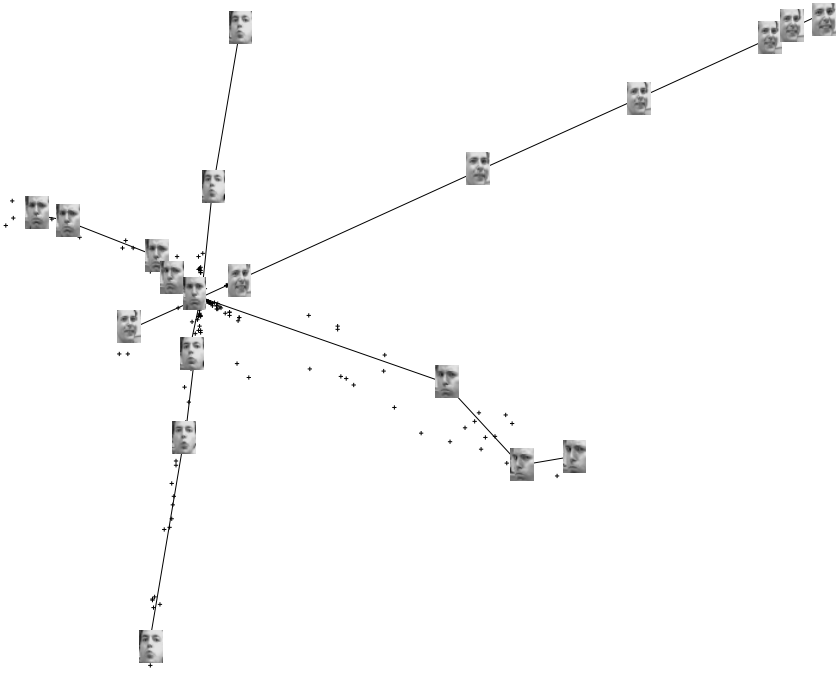


Figure 3: 2-dimensional embedding of the Frey faces dataset found using IHM. The results were obtained using k-means clustering for partitioning, with $k = 64$. The manifold is parameterized by expression with the extremes of the different expressions appearing on the outer edge of the embedding and the central points representing more neutral expression.

omit the backtracking step in Eq. (7) and instead use the forward tracking only method based on Eq. (5). The k-means algorithm was run with $k = 64$ and the minimum cluster size was set to $h = 6$.

The video from which the data is taken shows a face moving through a variety of expressions as well as slight changes in left-right pose. The 2-dimensional embedding found using IHM is shown in Figure 3. The main variation in the embedding is facial expression with the 'extremes' of the expressions (e.g. smiling, frowning) appearing on the edges of the manifold. The points lying in the centre of the manifold are the more neutral, less discriminable, expressions. When compared with other results using this dataset our embedding shows more meaningful structure. The results found in Figure 4 in [7] show that nearby images in the input space match nearby images in the output space, but the output embedding is homogeneous without any apparent of discernable structure. The embeddings found in Figure 3 in [14] reveal slightly more structure in the data but the distribution of expressions is hard to track. Our results show that there is structure in the data with expressions distributed across the manifold and at a local scale images in the local input neighborhood match those in the local output neighborhood.

4 Discussion & Future Work

We have introduced a new unsupervised manifold learning algorithm, Iterative Hyperplane Merging, that is based on the iterative merging of local hyperplanes. By using a clustering algorithm (e.g. Gaussian Mixture Modelling or k-means) to partition the data into local hyperplanes, which are then aligned according to a simple walk on the minimum spanning tree of the hyperplanes, we can achieve leading results on benchmark synthetic data. When compared against PCA, ISOMAP and LTSA our algorithm is capable of discovering the shape of the manifold even in the presence of substantial noise. We have shown using various error measures that our algorithm is able to maintain both global and local properties of the data. We have also shown how our algorithm is capable of learning a non-linear face based image manifold parameterized by facial expression.

One limitation of our algorithm is the reliance on a clustering algorithm to partition the samples into local hyperplanes. In high-dimensional spaces this can be both computationally expensive and slow. Also, if the size of the clusters are incorrectly chosen then this can lead to a short-circuited or disconnected manifold [1]. One possible approach to overcome this problem would be to replace the clusters with local tangent spaces. So rather than merging local hyperplanes the local tangent spaces of each sample would be merged. This could provide both an improvement in run-time as well as in embedding accuracy.

References

[1] M. Balasubramanian and E. L. Schwartz. The isomap algorithm and topological stability. *Science*, 295:7a (Techincal Comments), 2002.

[2] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[3] K. P. Bennett, P. S. Bradley, and A. Demiriz. Constrained k-means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, May 2000.

[4] C. A. Bouman. Cluster: An unsupervised algorithm for modeling Gaussian mixtures. Available from http://www.ece.purdue.edu/~bouman, April 1997.

[5] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 15*, pages 961–968. MIT Press, 2003.

[6] D. L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. Technical report, Stanford University Department of Statistics, 2000.

[7] Y. Goldberg and Y. Ritov. Local procrustes for manifold embedding: a measure of embedding quality and embedding algorithms. *Machine Learning*, 77(1):1–25, 2009.

[8] R. L. Graham and P. Hell. On the history of the Minimum Spanning Tree Problem. *IEEE Annals of the History of Computing*, 7(1):43–57, 1985.

[9] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[10] R. Huber, H. Ramoser, K. Mayer, H. Penz, and M. Rubik. Classification of coins using an eigenspace approach. *Pattern Recognition Letters*, 26(1):61–75, 2005.

[11] D. G. Kendall. A survery of the statistical theory of shape. *Statistical Science*, 4(2): 87–99, 1989.

[12] J. A. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Springer, 2007.

[13] V. Robins. *Computational Topology at Multiple Resolutions*. PhD thesis, Department of Applied Mathematics at the University of Colorado, Boulder, 2000.

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

[15] H. Sahbi. A particular Gaussian Mixture Model for clustering and its application to image retrival. *Soft Computing*, 12(7):667–676, 2008.

[16] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2322, 2000.

[17] S. Theodorodis and K. Koutroumbas. *Pattern Recognition*. Academic Press, 3 edition, 2006.

[18] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 586–591, 1991.

[19] G. Valiente. *Algorithms on Trees and Graphs*. Springer-Verlag, Berlin Heidelberg, 2002.

[20] J. Venna and S. Kaski. Local multidimensional scaling. *Neural Networks*, 19:889–899, 2006.

[21] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70:77–90, 2006.

[22] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM journal on scientific computing*, 26(1):313–338, 2004.