A GREEDY, ADAPTIVE APPROACH TO LEARNING GEOMETRY OF NONLINEAR MANIFOLDS

Talal Ahmed and Waheed U. Bajwa

Dept. of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ 08854

ABSTRACT

In this paper, we address the problem of learning the geometry of a non-linear manifold in the ambient Euclidean space into which the manifold is embedded. We propose a bottom-up approach to manifold approximation using tangent planes where the number of planes is adaptive to manifold curvature. Also, we exploit the local linearity of the manifold to subsample the manifold data before using it to learn the manifold geometry with negligible loss of approximation accuracy. In our experiments, our proposed Geometry Preserving Union-of-Affine Subspaces algorithm shows more than a 100-times decrease in the learning time when compared to state-of-the-art manifold learning algorithm, while achieving similar approximation accuracy.

Index Terms— Manifold geometry, manifold learning, subsampling, tangent spaces, union-of-affine subspaces

1. INTRODUCTION

Many models in signal processing are built on the notion of conciseness of a class of signals in a given geometric structure. For many decades, the most popular geometric models in signal processing have been the subspace and the union-of-subspaces signal models. However, even if a class of signals does not have a subspace or a union-of-subspaces representation, it may be manifested in a low-dimensional non-linear manifold embedded in a high-dimensional ambient space. Prominent examples of signal classes particularly well-represented by a manifold signal model are face and handwritten images [1, 2]. If we can learn the manifold structure for a class of signals, it can lead us to interesting solutions to applications like data denoising, novel data acquisition paradigms, compression, etc., for that class of signals.

A lot of work has been done on manifold learning and dimensionality reduction that is aimed towards finding a low-dimensional embedding of data sampled from a high-dimensional non-linear manifold [3]. However, we are interested in a slightly different problem: *we want to learn the manifold geometry in the ambient space that the manifold is originally embedded in*. One solution to our problem is to learn a global parametric representation for the manifold, but this approach won't always work because real world data often lie on highly non-linear manifolds and it is unreasonable to assume that we will always be able to find a parametric representation that encompasses all the non-linearities of the manifold. Instead, we approximate the non-linear manifold structure locally only using low-dimensional tangent spaces.

Representing data using subspaces is a well-addressed problem. Many approaches in subspace clustering and hybrid linear modelling are geared towards clustering the dataset into groups such that data in each group can be well-approximated using a collection of subspaces [4, 5, 6]. However, the primary focus of these methods is on minimization of approximation error of training data so the union of subspaces cannot be expected to represent the underlying manifold geometry.[1]

2. PROBLEM FORMULATION AND OUR CONTRIBUTIONS

We are interested in learning the geometry of a smooth d-dimensional Riemannian manifold $\mathcal{M}$ embedded in $\mathcal{R}^D$, where it is assumed that $d \ll D$ and the embedding is such that the nonlinear structure is not self-intersecting. For the sake of this exposition, we assume that the manifold dimension d is known a priori; see [7] for possible means of estimating d from the data. In order to learn $\mathcal{M}$ in $\mathcal{R}^D$, we are given a collection of N data points, $\mathcal{X} = \{x_1, x_2, \ldots, x_N\} \subset \mathcal{R}^D$, that are sampled from $\mathcal{M}$.

In the literature, both bottom-up [8] and top-down [9] approaches are adopted to approximate manifold geometry in $\mathcal{R}^D$ using tangent planes. Considering a hypothetical scenario in which the dataset $\mathcal{X}$ is dense in $\mathcal{M}$, any bottom-up approach to manifold approximation that considers every sample of the dataset would be computationally impractical. However, by exploiting the local linearity of the manifold, the dataset $\mathcal{X}$ can be subsampled with minimal loss of information to get a reduced version of the dataset, $\mathcal{X}_{red}$, which can be used for manifold learning.

In both [8] and [9], a shortcoming of the proposed learning approaches is that the final number of tangent planes required to represent the manifold structure needs to be somehow estimated and preset. We propose a manifold-adaptive *bottom-up* approach to manifold learning inspired by the work in [8]: we start by assigning a tangent plane T_j to each sample $x_j \in \mathcal{X}_{red}$ as a representation of the local manifold geometry about x_j, and thus we start with a collection of tangent planes $\{T_j\}_{j \in A}$. Then, pairs of neighboring tangent planes indexed by A are merged under the constraint that the approximation error of each tangent plane T_j, $j \in A$, remains within a preset threshold ϵ where the approximation error of T_j is defined as

$$e_j = \frac{1}{|C_j|} \sum_{x \in C_j} \frac{\|x - P_j x\|}{\|x - c_j\|}. \tag{1}$$

Here, C_j is the set of training data samples associated with T_j, P_j is a projection operator for the tangent plane T_j, c_j is the empirical mean of all the samples in C_j and $\|.\|$ is the Euclidean norm. The error constraint ensures that while we merge neighboring planes and try to minimize the number of planes approximating the manifold geometry, each tangent plane in the reduced set of planes is an accurate approximation of the local manifold geometry where accuracy of the approximation is controlled by the value of ϵ: the lower the value of ϵ, the more accurate the approximation. Thus, the way we

[1] See Fig. 1(b) for an example.

set up the manifold learning problem makes our proposed algorithm – termed as Geometry Preserving Union-of-Affine Subspaces (GP UoAS) – adaptive to the manifold curvature: the final number of tangent planes representing the manifold geometry is a function of the preset threshold ϵ and the manifold curvature.

In Section 3, we lay down the framework of our approach and then propose GP UoAS. In Section 4, we show the performance of our algorithm using synthetic datasets and the MNIST dataset [10]. The paper is finally concluded in Section 5.

3. APPROXIMATING MANIFOLD GEOMETRY USING TANGENT PLANES

We propose a manifold learning algorithm that first subsamples the dataset $\mathcal{X}$ by exploiting the local linearity of the manifold to get a subsampled version of the dataset, $\mathcal{X}_{red}$, as explained in Section 3.1. Then, the reduced dataset $\mathcal{X}_{red}$ is used to learn a collection of tangent planes that approximate the manifold geometry, as explained in Section 3.2.

3.1. Preprocessing: Subsampling the Dataset

The process of subsampling the dataset $\mathcal{X}$ starts by randomly selecting a data point $x \in \mathcal{X}$. A neighborhood set N_x that is a set of K_{in} nearest neighbors of x in $\mathcal{X}$ with respect to the Euclidean metric and a plane of best-fit to all the points in this set N_x are associated with the randomly selected point x. The associated plane, i.e., the tangent plane, is characterized by an orthonormal basis matrix $\phi_x \in \mathcal{R}^{d\times D}$, obtained via Singular Value Decomposition (SVD) of all the points in N_x, and a vector $c_x \in \mathcal{R}^{D\times 1}$, which is the mean of all the samples in N_x. Next, the approximation error of the tangent plane to the points in N_x is calculated. If this error is smaller than ϵ_0, the neighborhood size is incremented by K_Δ samples. We keep on expanding the neighborhood size in increments of K_Δ samples as long as the approximation error of the samples in N_x by the associated tangent plane remains within ϵ_0. Finally, assuming ϵ_0 is a very small number, all samples in N_x are well-approximated by the associated tangent plane. At this point, the samples in the final neighborhood set N_x are marked for deletion and the original sample x is added to $\mathcal{X}_{red}$. This process of randomly selecting a sample x from $\mathcal{X}$, associating a tangent plane with x, expanding the associated neighborhood till the approximation error is within an acceptable bound, and marking the samples in the final neighborhood set for deletion goes on till the training dataset is exhausted (refer to Algorithm 1 for further details).

3.2. Main Task: Approximation using Geometry-Preserving Union of Affine Subspaces

The subsampling stage of our algorithm gives a downsampled version $\mathcal{X}_{red}$ of the original dataset along with a collection of tangent planes: with each $x_j \in \mathcal{X}_{red}$, sampled in stage 1 of the algorithm, is associated a tangent plane characterized by a basis matrix $\phi_j \in \mathcal{R}^{D\times d}$ and a subspace offset $c_j \in \mathcal{R}^D$. A cluster of training data $C_j = \{x_j\}$ is also initialized for the tangent plane associated with $x_j \in \mathcal{X}_{red}$.

Let A be a set of indices such that the initial set of tangent planes, $\mathcal{T}$, learnt in stage 1 of the algorithm, can formally be written as:

$$\mathcal{T} = \{\mathcal{T}_j\}_{j\in A} \text{ such that } \mathcal{T}_j = \{\phi_j, c_j, C_j\}.$$

To minimize the number of tangent planes representing the manifold geometry, pairs of tangent planes in $\mathcal{T}$ are fused till further fusion of any pair of tangent planes from $\mathcal{T}$ will give an error (1) larger than ϵ in the merged tangent plane. However, to ensure the fused planes adhere to the local manifold geometry, not every pair of planes in $\mathcal{T}$ is eligible for fusion. Our definition of fusibility of tangent planes is inspired by the definition of fusibility of clusters in [11]. Let $N_K(x)$ contain the K-nearest neighbors of $x \in \mathcal{X}_{red}$ in $\mathcal{X}_{red}$ with respect to the Euclidean distance metric. Then, Ω is the set of all pairs of fusible planes in $\mathcal{T}$ such that

$$\Omega = \{(i,j) : y_i \in N_K(y_j) \text{ or } y_j \in N_K(y_i) \\ \text{where } y_i \in C_i, y_j \in C_j \text{ s.t. } i,j \in A, i \neq j\}. \quad (2)$$

Among all the fusible pairs of planes, we define the best fusible pair as the one which gives the least approximation error for the associated training dataset after fusion:

$$(i^*, j^*) = \arg\min_{(i,j)\in\Omega} e_{proj}(\mathcal{T}_i, \mathcal{T}_j), \quad (3)$$

where

$$e_{proj}(\mathcal{T}_i, \mathcal{T}_j) = \frac{1}{|C_i \cup C_j|} \sum_{x\in C_i\cup C_j} \frac{\|x - P_k\, x\|}{\|x - c_k\,\|}, \quad (4)$$

where $\mathcal{T}_k$ is the tangent plane obtained from merging of $\mathcal{T}_i$ and $\mathcal{T}_j$, $C_i \cup C_j$ is the set of training data associated with $\mathcal{T}_k$, P_k is the projection operator for projection onto $\mathcal{T}_k$ and c_k is the empirical mean of all the samples in $C_i \cup C_j$. Note that evaluation of (4) and thus (3) is an expensive operation because (4) involves computing SVD for the samples in C_i and C_j. Thus, (3) would require a SVD computation step for each pair in Ω. To get rid of the SVD computation step for each fusible pair of planes, we derive an upper bound on (4) that relies on the following lemma:

Lemma 1. *If $\mathcal{T}_i \in \mathcal{T}$, $\mathcal{T}_j \in \mathcal{T}$, then*

$$\sum_{x\in C_i} \frac{\|x - P_k\, x\|}{\|x - c_k\,\|} \leq \sum_{x\in C_i} \frac{\|x - P_j x\|}{\|x - c_k\,\|}, \quad \textit{and} \quad (5)$$

$$\sum_{x\in C_j} \frac{\|x - P_k\, x\|}{\|x - c_k\,\|} \leq \sum_{x\in C_j} \frac{\|x - P_i x\|}{\|x - c_k\,\|}. \quad (6)$$

Proof. Each of (5) and (6) can be proved by contradiction. Suppose (5) is not true, then $\sum_{x\in C_i} \frac{\|x-P_k\, x\|}{\|x-c_k\,\|} > \sum_{x\in C_i} \frac{\|x-P_j x\|}{\|x-c_k\,\|}$, which implies $\sum_{x\in C_i\cup C_j} \frac{\|x-P_k\, x\|}{\|x-c_k\,\|} > \sum_{x\in C_i} \frac{\|x-P_j x\|}{\|x-c_k\,\|} + \sum_{x\in C_j} \frac{\|x-P_k\, x\|}{\|x-c_k\,\|}$. From our construction of the tangent planes, $\sum_{x\in C_j} \frac{\|x-P_k\, x\|}{\|x-c_k\,\|} \geq \sum_{x\in C_j} \frac{\|x-P_j x\|}{\|x-c_k\,\|}$, which leads to $\sum_{x\in C_i\cup C_j} \frac{\|x-P_k\, x\|}{\|x-c_k\,\|} > \sum_{x\in C_i\cup C_j} \frac{\|x-P_j x\|}{\|x-c_k\,\|}$. This is a contradiction due to our construction of $\mathcal{T}_k$ and the Eckart-Young theorem [12], thus (5) must be true. Inequality (6) can also be proved similarly. □

Lemma 1 is next used in the derivation of the following theorem.

Theorem 1. *If $\mathcal{T}_i \in \mathcal{T}$, $\mathcal{T}_j \in \mathcal{T}$, then*

$$e_{proj}(\mathcal{T}_i, \mathcal{T}_j) \leq \frac{1}{|C_i \cup C_j|}\Big(\sum_{x\in C_i} \frac{\|x - P_i x\|}{\|x - c_k\,\|} + \sum_{x\in C_j} \frac{\|x - P_j x\|}{\|x - c_k\,\|} + \sum_{x\in C_i\cup C_j} \frac{\|P_i x - P_j x\|}{\|x - c_k\,\|}\Big) =: \bar{e}_{proj}(\mathcal{T}_i, \mathcal{T}_j). \quad (7)$$

Proof. Rewriting (4),

$$e_{proj}(\mathcal{T}_i,\mathcal{T}_j) = \frac{1}{|C_i\cup C_j|}\Big(\sum_{x\in C_i}\frac{\|x-P_k\,x\|}{\|x-c_k\,\|}+\sum_{x\in C_j}\frac{\|x-P_k\,x\|}{\|x-c_k\,\|}\Big)$$

$$\overset{(a)}{\leq} \frac{1}{|C_i\cup C_j|}\Big(\sum_{x\in C_i}\frac{\|x-P_jx\|}{\|x-c_k\,\|}+\sum_{x\in C_j}\frac{\|x-P_ix\|}{\|x-c_k\,\|}\Big)$$

$$= \frac{1}{|C_i\cup C_j|}\Big(\sum_{x\in C_i}\frac{\|x-P_jx+P_ix-P_ix\|}{\|x-c_k\,\|}+\sum_{x\in C_j}\frac{\|x-P_ix+P_jx-P_jx\|}{\|x-c_k\,\|}\Big)$$

$$\overset{(b)}{\leq} \frac{1}{|C_i\cup C_j|}\Big(\sum_{x\in C_i}\frac{\|x-P_ix\|}{\|x-c_k\,\|}+\sum_{x\in C_j}\frac{\|x-P_jx\|}{\|x-c_k\,\|}+\sum_{x\in C_i\cup C_j}\frac{\|P_ix-P_jx\|}{\|x-c_k\,\|}\Big)$$

where (a) follows from Lemma 1 and (b) follows from the triangular inequality. □

Note that in contrast to evaluation of $e_{proj}(\mathcal{T}_i,\mathcal{T}_j)$ in (4), evaluation of $\bar{e}_{proj}(\mathcal{T}_i,\mathcal{T}_j)$ in (7) does not involve computing SVD of the data samples in $C_i\cup C_j$. Thus, instead of solving (3), we minimize an upper bound to the objective in (3):

$$(i^*,j^*) = \underset{(i,j)\in\Omega}{\arg\min}\,\bar{e}_{proj}(\mathcal{T}_i,\mathcal{T}_j). \tag{8}$$

If $\bar{e}_{proj}(\mathcal{T}_{i^*},\mathcal{T}_{j^*}) \leq \epsilon$, the best pair of planes $(\mathcal{T}_{i^*},\mathcal{T}_{j^*})$ is merged to obtain the tangent plane $\mathcal{T}_{k^*}$, resulting in a new collection

$$\mathcal{T} \leftarrow (\mathcal{T}\setminus\{\mathcal{T}_{i^*},\mathcal{T}_{j^*}\})\cup\{\mathcal{T}_{k^*}\}. \tag{9}$$

Summarizing our algorithm, once the set $\mathcal{T}$ is initialized by calculating the tangent plane at each $x\in\mathcal{X}_{red}$, the set of pairs of fusible tangent planes Ω is calculated as in (2), the best pair $(\mathcal{T}_{i^*},\mathcal{T}_{j^*})$ from the set of fusible pairs of planes is selected using (8), and the best pair of planes is merged as in (9) if the best pair of planes satisfies the approximation error constraint $\bar{e}_{proj}(\mathcal{T}_{i^*},\mathcal{T}_{j^*})\leq\epsilon$. This process of evaluating (2), (8) and (9) is repeated till $\bar{e}_{proj}(\mathcal{T}_{i^*},\mathcal{T}_{j^*})$ gives a value greater than ϵ after the evaluation of (8). In other words, we keep on finding and merging the best pair of fusible tangent planes till the best pair of fusible planes does not satisfy the approximation error constraint. The accuracy of manifold geometry approximation by the final set of tangent planes depends on the value of ϵ: the smaller the value of ϵ, the more accurate the estimate and vice versa. Our algorithm is outlined in Algorithm 1.

4. EXPERIMENTAL RESULTS

In all our experiments, we compare our algorithm with the state-of-the-art algorithm for learning manifold geometry in the ambient space using tangent spaces [11]. Because the method in [11] uses difference of tangents to merge neighboring tangent planes, we dub it as the 'Merging based on Difference of Tangents' (MDOT) method. To compare the performance of the two algorithms, we sample 1800 data points from different number of half-turns of a swiss roll. We fix the final number of tangents planes for MDOT algorithm at 10, whereas our algorithm adapts the number of planes required to approximate manifold geometry according to the manifold curvature. The following error is used as an approximation accuracy metric:

$$\textbf{error} = \sum_{j\in A}\frac{1}{|C_j|}\sum_{x\in C_j}\frac{\|x-\phi_j\phi_j^\top x\|}{\|x-c_j\|}.$$

The results reported in Table 1 show that the approximation error for MDOT algorithm increases with the increasing number of turns of the swiss roll, whereas our algorithm adapts to the increasing manifold curvature by increasing the number of tangent planes used to learn the swiss roll geometry. An example of union of tangent planes approximation of 3 half turns of a swiss roll using our algorithm is shown in Fig. 1(a).

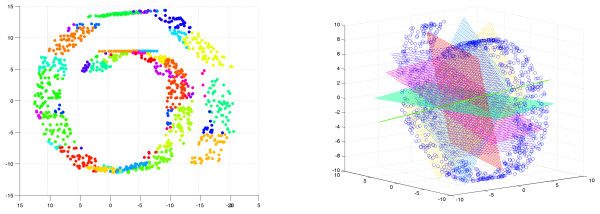
Algorithm 1: Learning Geometry-Preserving Union of Affine Subspaces

Input: Dataset: $\mathcal{X}$; maximum error in Stage 1: ϵ_0; starting neighborhood size in Stage 1: K_{in}; neighborhood increment size: K_Δ; neighborhood size in Stage 2: K; maximum error in Stage 2: ϵ; dimension of tangent planes: d
Output: Final set of tangent planes representing the manifold: $\mathcal{T}_j = \{\phi_j, c_j\}_{j\in A}$
Stage 1 (Subsampling the dataset):
Initialize: $\mathcal{X}_{red}\leftarrow\mathcal{X}$, $\mathcal{X}_{red1}\leftarrow\mathcal{X}$, $\phi\leftarrow\{\}$, $c\leftarrow\{\}$
while $\mathcal{X}_{red1}\neq\{\}$ **do**
  Uniformly at random select $x\in\mathcal{X}_{red1}$
  $K_0\leftarrow K_{in}$; $N_x\leftarrow K_0$ nearest neighbors of x in $\mathcal{X}$
  $c_x\leftarrow\frac{1}{|N_x|}\sum_{y\in N_x} y$; $[N_x^0]\leftarrow\{y-c_x : y\in N_x\}$
  $U_x\leftarrow$ left singular vectors of $[N_x^0]$ corresponding to its d-largest singular values
  $error = \frac{1}{|N_x|}\sum_{y\in N_x}\frac{\|y-U_xU_x^Ty\|}{\|y-c_x\|}$; $N_x^*\leftarrow N_x$
  while $error<\epsilon_0$ **do**
    $N_x\leftarrow N_x^*$; $K_0\leftarrow K_0+K_\Delta$
    $N_x^*\leftarrow K_0$ nearest neighbors of x in $\mathcal{X}$
    $error = \frac{1}{|N_x^*|}\sum_{y\in N_x^*}\frac{\|y-U_xU_x^Ty\|}{\|y-c_x\|}$
  end while
  $\phi\leftarrow\{\phi,U_x\}$; $c\leftarrow\{c,c_x\}$
  $\mathcal{X}_{red}\leftarrow\mathcal{X}_{red}\setminus N_x$; $\mathcal{X}_{red1}\leftarrow\mathcal{X}_{red1}\setminus\{N_x,x\}$
end while

Stage 2 (Merging the tangent planes):
$N_K(x)\leftarrow\{K$ nearest neighbors of x in $\mathcal{X}_{red}\}, x\in\mathcal{X}_{red}$
$C\leftarrow\{\{x\} : x\in\mathcal{X}_{red}\}$; Let A be a set of indices such that the set of tangent planes from stage 1 can be written as
$\mathcal{T}\leftarrow\{\mathcal{T}_j\}_{j\in A}$ such that $\mathcal{T}_j=\{\phi_j\in\phi, c_j\in c, C_j\in C\}$
loop
  $\Omega\leftarrow\{(i,j) : y_i\in N_K(y_j)$ or $y_j\in N_K(y_i)$, where $y_i\in C_i, y_j\in C_j$ such that $(i,j)\in A\}$
  $(i^*,j^*,\bar{e}^*_{proj}) = \underset{(i,j)\in\Omega}{\arg\min}\,\bar{e}_{proj}(C_i,C_j)$
  if $\bar{e}^*_{proj}<\epsilon$ **then**
    $\mathcal{T}\leftarrow(\mathcal{T}\setminus\{\mathcal{T}_{i^*},\mathcal{T}_{j^*}\})\cup\{\mathcal{T}_{k^*}\}$, where $\mathcal{T}_{k^*}$ is the plane obtained from merging planes $\mathcal{T}_{i^*}$ and $\mathcal{T}_{j^*}$
  else
    break the **loop**
  end if
end loop

To compare the computational complexity of the two algorithms, 3 half turns of a swiss roll are sampled with varying sampling density. For the MDOT algorithm, the number of tangent planes are set to 14. The results for this experiment are given in Table 2. Results show more than 2 orders of magnitude difference in the computational time of the two algorithms for similar approximation accuracy, and the computational advantage of our algorithm becomes more significant as sampling density on the manifold increases.

We also test our algorithm on a high-dimensional dataset – the MNIST database [10]. Setting $d=5$, we run both algorithms on 1000 images of digit zero randomly selected from the MNIST database. For our algorithm, we set $K_{in}=5$, $K_\Delta=1$, $K=6$ and we vary the value of ϵ to approximate the manifold of digits with



(a) (b)

Fig. 1. (a) Data sampled from a swiss roll projected onto the union of 17 tangent planes learnt using our proposed algorithm. (b) Flat approximation of swiss roll using median K-flats algorithm [13].

Half-turns of Roll	Tangent Planes (MDOT)	Error (MDOT)	Tangent Planes (GP UoAS)	Error (GP UoAS)
1	10	0.027	5.8	0.078
2	10	0.066	10.9	0.084
3	10	0.137	16.4	0.084
4	10	0.187	21.3	0.089
5	10	0.247	26.7	0.091

Table 1. Approximation of the underlying structure of 1800 data points randomly sampled from different turns of a swiss roll using the 'MDOT' algorithm with 10 planes and our manifold adaptive 'GP UoAS' algorithm.

Points sampled from the swiss roll	Time in seconds (MDOT)	Error (MDOT)	Time in seconds (GP UoAS)	Error (GP UoAS)
1800	2.3×10^3	0.090	13.8	0.080
3000	1×10^4	0.091	30.6	0.104
4200	3×10^4	0.097	65.1	0.079
5400	5.5×10^4	0.092	86.9	0.078
6600	9.6×10^4	0.091	185.9	0.085

Table 2. Time taken to learn the manifold structure with respect to the sampling density on the manifold, which is 3 half turns of a swiss roll in this experiment.

different number of tangent planes. The results in Fig. 2 show similar approximation performance for both the algorithms for different number of planes.

5. CONCLUSIONS

We have proposed a bottom-up approach to learning manifold structure using union of tangent planes. Our algorithm shows particularly impressive computational complexity performance when learning manifold geometry using dense training datasets. More importantly, our algorithm adapts the number of tangent planes required for manifold geometry approximation with curvature of the manifold to ensure the approximation error remains within acceptable threshold.

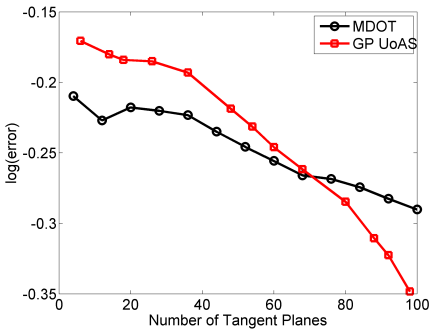


Fig. 2. Approximation of the underlying manifold structure of 1000 images of digit '0' – extracted from the MNIST database – using different number of tangent planes.

6. REFERENCES

[1] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1994, pp. 84–91.

[2] G. E. Hinton, P. Dayan, and M. Revow, "Modeling the manifolds of images of handwritten digits," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 65–74, 1997.

[3] L. J. van der Maaten, E. O. Postma, and H. J. van den Herik, "Dimensionality reduction: A comparative review," *J. Machine Learning Research*, vol. 10, no. 1-41, pp. 66–71, 2009.

[4] U. Von Luxburg, "A tutorial on spectral clustering," *Stat. and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[5] T. Zhang, A. Szlam, Y. Wang, and G. Lerman, "Hybrid linear modeling via local best-fit flats," *Int. J. Computer Vision*, vol. 100, no. 3, pp. 217–240, 2012.

[6] Z. Ghahramani and G.E. Hinton, "The EM algorithm for mixtures of factor analyzers," *Technical Report CRG-TR-96-1, University of Toronto*, 1996.

[7] B. Kgl, "Intrinsic dimension estimation using packing numbers," in *Adv. Neural Inform. Process. Syst.*, 2002, pp. 681–688.

[8] S. Karygianni and P. Frossard, "Linear manifold approximation based on differences of tangents," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2011, pp. 973–976.

[9] W. K. Allard, G. Chen, and M. Maggioni, "Multi-scale geometric methods for data sets II: Geometric multi-resolution analysis," *Appl. and Computational Harmonic Anal.*, vol. 32, no. 3, pp. 435–462, 2012.

[10] Y. LeCun and C. Cortes, "The MNIST database of handwritten digits," `http://yann.lecun.com/exdb/mnist/`, 1998.

[11] S. Karygianni and P. Frossard, "Tangent-based manifold approximation with locally linear models," *arXiv preprint arXiv:1211.1893*, 2012.

[12] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[13] T. Zhang, A. Szlam, and G. Lerman, "Median k-flats for hybrid linear modeling with many outliers," in *Proc. IEEE 12th Int. Conf. Comput. Vision Workshops*, 2009, pp. 234–241.