GEOMETRIC MANIFOLD APPROXIMATION USING LOCALLY LINEAR APPROXIMATIONS

BY TALAL AHMED

A thesis submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Prof. Waheed U. Bajwa

and approved by

New Brunswick, New Jersey

May, 2016

ABSTRACT OF THE THESIS

Geometric Manifold Approximation using Locally Linear Approximations

by Talal Ahmed

Thesis Director: Prof. Waheed U. Bajwa

The design and analysis of methods in signal processing is greatly impacted by the model being selected to represent the signals of interest. For many decades, the most popular geometric models in signal processing have been the subspace and the union-of-subspaces signal models. However, there are classes of signals that are not well-represented by either the subspace or the union-of-subspaces model, but are manifested in a low-dimensional non-linear manifold embedded in a high-dimensional ambient space. Though a lot of work has been done on low-dimensional embedding of manifold sampled data, few works address the problem of approximating the manifold geometry in the ambient space. There is value in capturing the geometric variations of a non-linear manifold in the ambient space, as shown in this work.

In this work, the local linearity of a manifold is exploited to address the problem of approximating the geometric structure of a non-linear non-intersecting manifold using a union of tangent planes (affine subspaces). The number of approximating tangent planes adapts with the varying manifold curvature such that the manifold geometry is always well approximated within a given accuracy. Also, the local linearity of the manifold is exploited to subsample the manifold data before using it to learn the manifold geometry, with negligible loss of approximation accuracy. Owing to this subsampling feature, the proposed approach shows more than a 100-times decrease in the learning time when compared to state-of-the-art manifold learning algorithms, while achieving similar approximation accuracy.

Because the approximating tangent planes extend indefinitely in space, the data encoding problem becomes complicated in the aforementioned learning approach. Thus, in the second half of the thesis, the manifold approximation problem is reformulated such that the geometry is approximated using a union of tangent patches, instead of tangent planes. Then, the data encoding problem is formulated as a series of convex optimization problems, and an efficient solution is proposed to solve each of the convex problems. Last, the value of capturing manifold geometry is demonstrated by showing the denoising performance of our proposed framework on both synthetic and real data.

Acknowledgements

I am grateful to my research advisor Waheed Bajwa whose insights, confidence and time was crucial in the completion of this research work. I am also grateful to my labmates and colleagues for providing just the right dose of encouragement and support at the right moments. I would also like to thank all my friends and family members without whom I won't have made it till here.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Data Models in Signal Processing

Data models play an increasingly important role in information processing. The data model helps in distinguishing the class of interesting signals from the class of uninteresting ones. Furthermore, the chosen model has a major impact on the design and analysis of signal processing tools and methods for the class of signals we are interested in. One such prominent model is the bandlimited signal model where it is assumed that the signal of interest can be represented as a linear combination of sinusoids. The prominent Nyquist sampling theorem is derived under this bandlimited signal model assumption.

1.2 Non-linear Manifold Signal Model

Owing to computational advances of recent years, classical linear models are slowly but surely being replaced by their nonlinear generalizations. The union-of-subspaces (UoS) model and the (nonlinear) manifold model in particular stand out among popular nonlinear data models [2]. In order to incorporate any nonlinear model into most information processing tasks, one requires scalable algorithms for (*i*) data-adaptive learning of geometric structure of the underlying model (structure learning) and (*ii*) projecting arbitrary data onto the learned geometric structure (data encoding). Significant progress has been made in this regard for the UoS model under the rubrics of *dictionary learning* and *sparse coding* [3–5]. But relatively less progress has been made toward these two goals for the manifold model. Our focus in this work is on the (nonlinear) manifold model and we present novel algorithms for structure learning and data encoding under the manifold model assumption.

Though little work has been done on structure learning and data encoding algorithms for manifold sampled data, there are classes of signals that are manifested in a low-dimensional non-linear manifold embedded in a high-dimensional ambient space. The manifestation is such that the intrinsic dimension of the manifold is equal to the number of degrees of freedom for that class of signals which is much smaller than the ambient dimension. Prominent examples of signal classes particularly well-represented by a manifold signal model are face and handwritten images [6, 7]. In this thesis, we address the structure learning and data encoding problem for similar classes of signals.

1.3 Relevant Prior Work

While manifold data models have been considered in the literature for more than a decade now, most of the work in manifold learning is aimed towards finding a low-dimensional embedding of data sampled from a high-dimensional non-linear manifold. The popular nonlinear dimensionality reduction techniques can be divided into the following three main types.

- There are techniques that attempt to preserve global properties of the original data in the low-dimensional representation e.g. multidimensional scaling (MDS) [8], ISOMAP [9], Maximum Variance Unfolding (MVU) [10].
- Then there are methods that are based on solely preserving properties of small neighborhoods around the data samples. The main idea behind such set of techniques is that the global layout of the data can be captured by preserving local properties of the manifold data. Some of the popular local nonlinear techniques for dimensionality reduction are: Local Linear Embedding (LLE) [11], Laplacian Eigenmaps [12], Hessian LLE [13], and LTSA (Linear Tangent Space Alignment) [14].
- Finally, there are techniques that combine the two aforementioned dimensionality reduction approaches: they compute a number of *locally* linear models and perform a *global* alignment of these linear models. Locally linear coordination [15]

and manifold charting [16] are examples of such methods.

Similarly, there are works that learn global parameterization of the manifold by learning locally linear approximations of the manifold data. For example, in [17], the manifold data is approximated using locally linear parameterizations of the manifold. In [17], the manifold parametrization is learnt in two stages: first, the manifold data is clustered into K groups and then each group is approximated using an affine subspace. There are also other works that combine local probabilistic models to find a global parameterization of the manifold [16, 18]. However, in this work, our objective is not to learn a global parameterization of the manifold but to learn the geometry of the underlying manifold structure.

There is also a lot of work on data representation using a union of affine subspaces under the UoS model assumption. Such approaches often fall under the category of subspace clustering or hybrid linear modeling. The popular iterative subspace clustering approach is to proceed by clustering the training data into groups and learning a subspace representation for each group. Often, the clustering and subspace estimation steps are performed iteratively such that the sum of approximation errors is minimized [1, 19]. Then, there are also the algebraic methods for clustering data under the assumption that the data is sampled from a union of subspaces [20–22]. In particular, the subspace clustering problem is formulated as a problem of representing subspace data using homogeneous polynomials in [20].

Note that, in principle, one can use the methods and algorithms developed under the UoS model assumption to learn an approximation of the data sampled from a nonlinear manifold (Remember that the Union of Subspace (UoS) model assumes that the training data is actually sampled from a union of subspaces). However, while such an approach will lead to small approximation error for the training data, it will fail to capture the geometry of the underlying manifold. As shown in Fig. 1.3, the median K-flats algorithm [1] does a good job at approximating the training data, but the K-flats approximation fails to capture the manifold geometry. One of the goals of this thesis in this regard is demonstrating there is value in preserving the manifold geometry while learning an approximation of the training data.

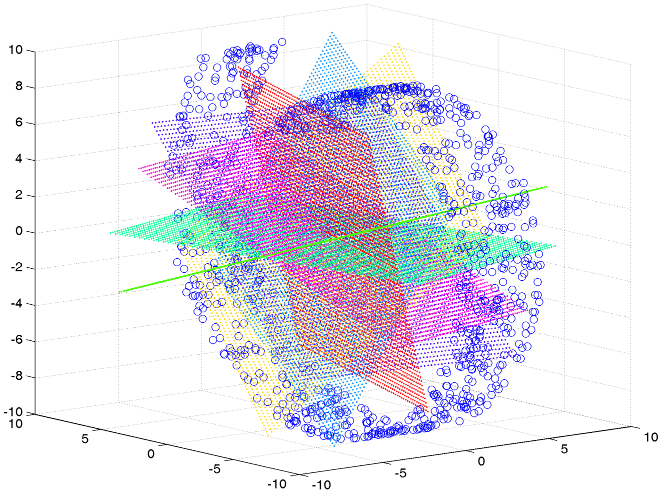


Figure 1.1: Learning a union of affine subspace approximation of swiss roll using median K-flats algorithm [1].

In this thesis, we are interested in the following problem: we want to *learn* the manifold *structure* in the *ambient space* that the manifold is originally embedded in. We are interested in capturing the manifold geometry as we learn an approximation of the manifold in the ambient space. Many other works that focus on learning manifold structure in the ambient space assume parametric manifolds [23]. However, the parametric approach won't always work because real world data often lie on highly non-linear manifold 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.

Note that a manifold is locally euclidean, so the manifold geometry can be locally approximated using linear approximations. The problem of approximating manifold geometry using local linear approximations (affine subspaces) has been addressed in recent papers using both bottom-up and top-down approaches to manifold approximation [24, 25]. The work in [24] proposes a bottom-up approach to manifold approximation in the embedding space: a tangent space is estimated at each sample in the dataset, and pairs of tangent planes are merged based on the difference of tangents till the number of approximating planes is reduced to a preset number. Considering that a tangent space is estimated for each sample in the dataset, such an algorithm becomes computationally expensive as the sampling density on the manifold increases. Another limitation of [24] is that it does not adapt to underlying manifold geometry; instead, it requires as input the correct number of affine subspaces to approximate the manifold structure. Furthermore, the work in [24] lacks algorithm for data encoding. Because the tangent planes (affine subspaces) extend indefinitely in space, it's not easy to address the data encoding problem in the problem setting put forth by Karygianni and Frossard [26].

1.4 Our Approach and Contributions

In our work, we approximate the non-linear manifold structure locally only using linear structures. We use linear approximations because the manifold is locally linear and projecting samples onto linear structures is computationally cheaper. In Chapter 2, we formulate the problem of manifold learning as the problem of finding a union of d-dimensional tangent planes (affine subspaces) approximation of the training dataset such that the set of planes captures the underlying manifold geometry with the least number of planes possible for a given approximation accuracy. Our contributions in Chapter 2 are as follows:

- The way we set up the manifold approximation problem makes the final number of approximating planes adaptive to manifold curvature. This addresses the problem of knowing the correct number of planes required for approximating the manifold structure with a given accuracy [24].
- We propose a greedy subsampling procedure that makes the manifold approximation algorithm computationally feasible even when dealing with training sets densely sampled from the manifold.
- We avoid the inherent expensive SVD computations when merging tangent planes planes.

However, as discussed earlier, affine subspaces extend indefinitely in space which complicates the data encoding process. In Chapter 3, we address the geometric manifold approximation problem by approximating the manifold using a *union of tangent patches* (UoTP) instead of tangent planes. A tangent patch can be seen as that subset of an approximating tangent plane that actually represents the local manifold geometry. This problem reformulation enables us to propose a data encoding algorithm for the manifold sampled data. Our contributions in Chapter 3 are as follows:

- We propose a data-adaptive procedure for approximating the manifold structure using a *union of tangent patches* (UoTP) such that the union of tangent patches represent the manifold geometry.
- The corresponding problem of encoding new data samples onto the learnt manifold structure is formulated as a solution to a series of convex programs and we present efficient means of solving these convex programs.
- We demonstrate the value in capturing manifold geometry by demonstrating the denoising performance of the presented structure learning and data encoding procedure on synthetic and real data.

Chapter 2

Manifold Approximation using Union of Tangent Planes

2.1 Introduction

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 [27] 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 [24] and top-down [25] approaches are adopted to approximate manifold geometry in $\mathcal{R}^D$ using tangent planes. In [24], the neighborhood graph of each sample is constructed by connecting each sample to its K nearest neighbors. The tangent space of each sample is formed by the d eigenvectors that correspond to the d largest eigenvalues of the data samples in the neighborhood of the selected sample. Once the tangent spaces are computed, they are merged in a greedy fashion. Because the method in [24] uses difference of tangents to merge neighboring tangent planes, we dub it as the 'Merging based on Difference of Tangents' (MDOT) method.

However, 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 approximation. Another shortcoming of the proposed manifold approximation approaches in

[24] and [25] is that the final number of tangent planes required for manifold approximation needs to be somehow estimated and preset. We address these shortcomings in the following bottom-up approach to manifold geometry approximation using a union of tangent planes.

2.2 Overview of Our Approach

We propose a manifold-adaptive *bottom-up* approach to manifold learning inspired by the work in [24]: 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

$$e_j = \frac{1}{|C_j|} \sum_{x \in C_j} \frac{\|x - P_j x\|}{\|x - c_j\|}. \tag{2.1}$$

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 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 2.3, we lay down the framework of our approach and then propose GP UoAS. In Section 2.4, we show the performance of our algorithm using synthetic datasets and the MNIST dataset [28].

2.3 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 2.3.1. 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 2.3.2.

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 $K_\triangle$ samples. We keep on expanding the neighborhood size in increments of $K_\triangle$ 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).

2.3.2 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 (2.1) 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 [24]. 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

$$\begin{aligned} \Omega = \{(i,j) : y_i \in N_K(y_j) \text{ or } y_j \in N_K(y_i) \\ \text{where } y_i \in C_i, y_j \in C_j \text{ s.t. } i,j \in A, i \neq j\}. \end{aligned} \tag{2.2}$$

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:

$$(i^*, j^*) = \underset{(i,j) \in \Omega}{\arg\min}\, e_{proj}(\mathcal{T}_i, \mathcal{T}_j), \tag{2.3}$$

where

$$e_{proj}(\mathcal{T}_i, \mathcal{T}_j) = \frac{1}{|C_i \cup C_j|} \sum_{x \in C_i \cup C_j} \frac{\|x - P_k x\|}{\|x - c_k\|}, \tag{2.4}$$

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 (2.4) and thus (2.3) is an expensive operation because (2.4) involves computing SVD for the samples in C_i and C_j. Thus, (2.3) 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 (2.4) that relies on the following lemma:

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

$$\sum_{x \in C_i} \frac{\|x - P_k x\|}{\|x - c_k\|} \leq \sum_{x \in C_i} \frac{\|x - P_j x\|}{\|x - c_k\|}, \quad \text{and} \tag{2.5}$$

$$\sum_{x \in C_j} \frac{\|x - P_k x\|}{\|x - c_k\|} \leq \sum_{x \in C_j} \frac{\|x - P_i x\|}{\|x - c_k\|}. \tag{2.6}$$

Proof. Each of (2.5) and (2.6) can be proved by contradiction. Suppose (2.5) 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 [29], thus (2.5) must be true. Inequality (2.6) 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*

$$\begin{aligned} e_{proj}(\mathcal{T}_i, \mathcal{T}_j) &\leq \frac{1}{|C_i \cup C_j|}\Big(\sum_{x \in C_i} \frac{\|x - P_i x\|}{\|x - c_k\|} + \sum_{x \in C_j} \frac{\|x - P_j x\|}{\|x - c_k\|} + \sum_{x \in C_i \cup C_j} \frac{\|P_i x - P_j x\|}{\|x - c_k\|}\Big) \\ &=: \bar{e}_{proj}(\mathcal{T}_i, \mathcal{T}_j). \end{aligned} \tag{2.7}$$

Proof. Rewriting (2.4),

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

$\overset{(a)}{\leq} \frac{1}{|C_i \cup C_j|}\big(\sum_{x \in C_i} \frac{\|x - P_j x\|}{\|x - c_k\|} + \sum_{x \in C_j} \frac{\|x - P_i x\|}{\|x - c_k\|}\big)$

$= \frac{1}{|C_i \cup C_j|}\big(\sum_{x \in C_i} \frac{\|x - P_j x + P_i x - P_i x\|}{\|x - c_k\|} + \sum_{x \in C_j} \frac{\|x - P_i x + P_j x - P_j x\|}{\|x - c_k\|}\big)$

$\overset{(b)}{\leq} \frac{1}{|C_i \cup C_j|}\big(\sum_{x \in C_i} \frac{\|x - P_i x\|}{\|x - c_k\|} + \sum_{x \in C_j} \frac{\|x - P_j x\|}{\|x - c_k\|} + \sum_{x \in C_i \cup C_j} \frac{\|P_i x - P_j x\|}{\|x - c_k\|}\big)$

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 (2.4), evaluation of $\bar{e}_{proj}(\mathcal{T}_i, \mathcal{T}_j)$ in (2.7) does not involve computing SVD of the data samples in $C_i \cup C_j$. Thus, instead of solving (2.3), we minimize an upper bound to the objective in (2.3):

$$(i^*, j^*) = \underset{(i,j)\in\Omega}{\arg\min}\, \bar{e}_{proj}(\mathcal{T}_i, \mathcal{T}_j). \tag{2.8}$$

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

$$\mathcal{T} \leftarrow (\mathcal{T} \setminus \{\mathcal{T}_{i^*}, \mathcal{T}_{j^*}\}) \cup \{\mathcal{T}_{k^*}\}. \tag{2.9}$$

Let us analyze the computational savings in skipping the SVD computation for each fusible pair of planes. Let λ be the iteration number such that $\lambda = 1, 2, \ldots$, let $\mathbf{C}_\lambda$ be the set of clusters associated with the tangent planes at iteration λ, let Ω_λ be the set of fusible tangent plane pairs at iteration λ, let Ω_{total} be the summation of Ω_λ over all iterations, and let $\mathcal{L}$ be the final number of tangent planes approximating the manifold structure. Let n be the number of samples in the subsampled dataset $\mathcal{X}_{red}$. Note that at the initialization stage, a tangent plane is estimated at every sample in the subsampled dataset $\mathcal{X}_{red}$. Thus, n is also the size of the set of tangent planes at initialization. Then,

$$|\Omega_\lambda| \leq \sum_{C_i \in \mathbf{C}_\lambda} K|C_i| = K \sum_{C_i \in \mathbf{C}_\lambda} |C_i| = K(n - \lambda),$$

$$\Omega_{\text{total}} = \sum_{\lambda=0}^{\lambda=n-\mathcal{L}} |\Omega_\lambda| \leq \sum_{\lambda=0}^{\lambda=n-\mathcal{L}} K(n-\lambda) = \mathcal{O}(n^2).$$

Thus, the savings in the number of SVD computations are $\mathcal{O}(n^2)$ where n is the number of samples in the subsampled dataset $\mathcal{X}_{red}$.

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 (2.2), the best pair $(\mathcal{T}_{i^*}, \mathcal{T}_{j^*})$ from the set of fusible pairs of planes is selected using (2.8), and the best pair of planes is merged as in (2.9) 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 (2.2), (2.8) and (2.9) is repeated till $\bar{e}_{proj}(\mathcal{T}_{i^*}, \mathcal{T}_{j^*})$ gives a value greater than ϵ after the evaluation of (2.8). 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.

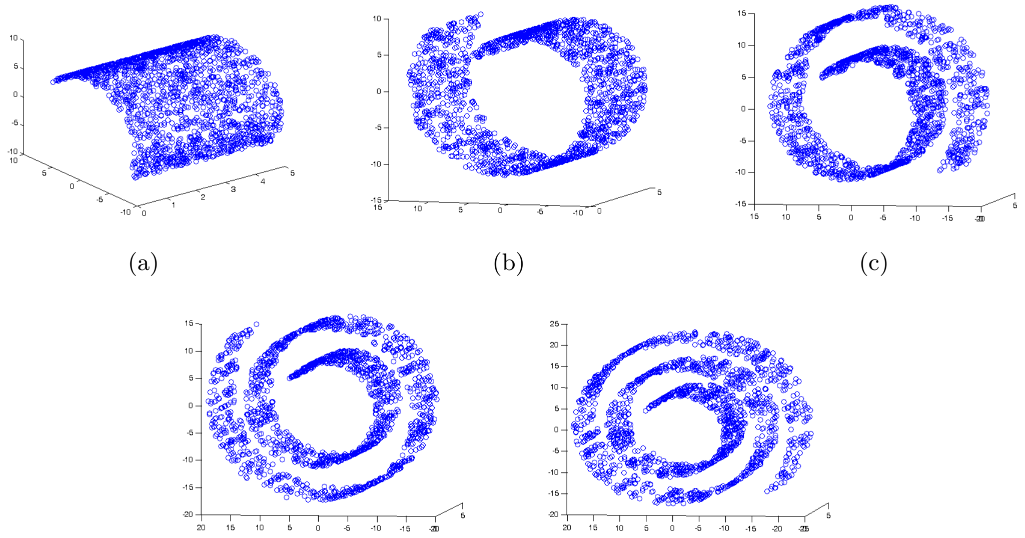
2.4 Experimental Results

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 [24]. The algorithm was dubbed as the MDOT method at the start of this chapter. To compare the performance of the two algorithms, we sample 1800 data points from different number of half-turns of a Swiss roll. An example of the different types of Swiss rolls used in this experiment is shown in Fig. 2.1. With the increasing number of half turns, the curvature of the manifold (Swiss roll) increases. With the increasing curvature of the manifold, increasing number of tangent planes are required for Swiss roll approximation. Because the MDOT algorithm requires the correct number of planes required for approximating the training data, we fix the final number of tangents planes for MDOT algorithm at 10. In comparison, our algorithm adapts the number of planes required to approximate manifold geometry with 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 structure learning results for different half-turns of a Swiss roll are shown in Table 2.1. Table 1 shows 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. 2.2.

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 2.2. 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.



(a)

(b)

(c)

(d)

(e)

Figure 2.1: Examples of Swiss rolls used in the structure learning experiment: (a) with 1 half-turn, (b) with 2 half-turns, (c) with 3 half-turns, (d) with 4 half-turns, and (e) with 5 half-turns.

We also test our algorithm on a high-dimensional dataset – the MNIST database [28]. 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 different number of tangent planes. The results in Fig. 2.3 show similar approximation performance for both the algorithms for different number of planes.

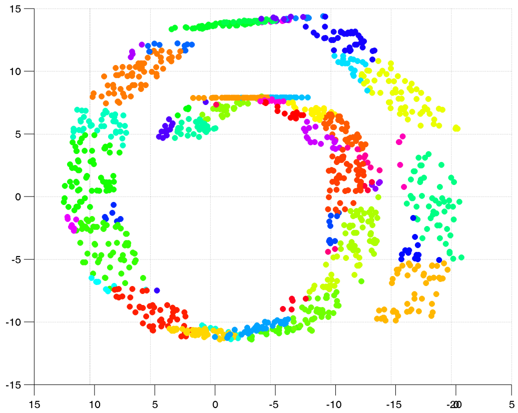


Figure 2.2: (a) Data sampled from a swiss roll projected onto the union of 17 tangent planes learnt using our proposed algorithm. (b) Flat approximation of swiss roll using median K-flats algorithm [1].

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 2.1: 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 2.2: 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.

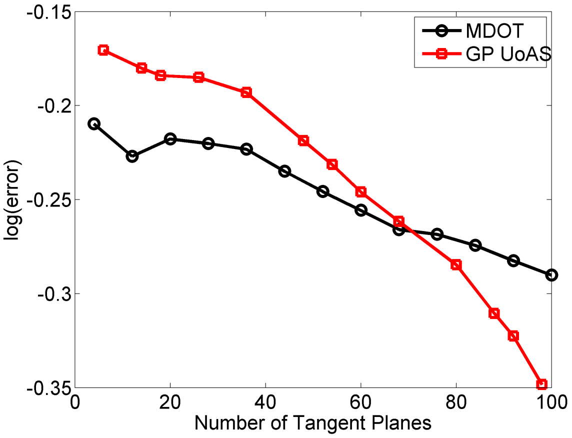


Figure 2.3: Approximation of the underlying manifold structure of 1000 images of digit ‘0’ – extracted from the MNIST database – using different number of tangent planes.

Algorithm 1: Learning Geometry-Preserving Union of Affine Subspaces

```
Input: Dataset: $\mathcal{X}$; maximum error in Stage 1: $\epsilon_0$; starting neighborhood size in Stage 1: $K_{in}$; neighborhood increment size: $K_\Delta$; neighborhood size in Stage 2: $K$; maximum error in Stage 2: $\epsilon$; 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
    $error = \frac{1}{|N_x|} \sum_{y \in N_x} \frac{\|y - U_x U_x^T y\|}{\|y - c_x\|}$; $N_x^* \leftarrow N_x$
    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 $\mathcal{X}$
        $error = \frac{1}{|N_x^*|} \sum_{y \in N_x^*} \frac{\|y - U_x U_x^T y\|}{\|y - c_x\|}$
    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 $(i, j) \in A\}$
    $(i^*, j^*, \bar{e}_{proj}^*) = \arg\min_{(i,j) \in \Omega} \bar{e}_{proj}(C_i, C_j)$
    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
```

Chapter 3

Manifold Approximation using Union of Tangent Patches

3.1 Introduction

The manifold approximation problem was addressed in the previous chapter by approximating the manifold structure using a union of tangent planes (affine subspaces). The proposed Geometry Preserving Union-of-Affine Subspaces (GP UoAS) approach not only adapts with the varying manifold curvature but also uses a greedy subsampling procedure to decrease the size of the training dataset with negligible loss of information. Though the approximation approach addressed many of the problems with similar manifold approximation procedures in [24] and [25], the approximating tangent planes extend indefinitely in the ambient space, which makes the data encoding process complicated. In this chapter, we reformulate the manifold approximation problem such that the manifold geometry is captured using a union of tangent patches (see Section 3.2). As mentioned before, a tangent patch is that subset of a tangent plane (or affine subspace) that gives a local linear approximation of the manifold. The problem reformulation enables us to convert the data encoding problem into a series of convex programs, each of which can be solved using an efficient algorithm (see Section 3.3.2. Finally, we use the data encoding algorithm along with the proposed structure learning approach in this chapter to show the value of capturing manifold geometry in denoising synthetic as well as real data (see Section 3.4).

3.2 Problem Formulation

In this chapter, again, we address the problem of approximating the geometry of a smooth d-dimensional Riemannian manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$ but using a collection of tangent patches, where it is assumed that $d \ll D$ and the embedding is such that the nonlinear structure is not self-intersecting. Also, we address the problem of projecting a new data sample $x \in \mathbb{R}^D$ onto the learnt collection of tangent patches. For the sake of this exposition, we assume that the manifold dimension d is known a priori; see [27] for possible means of estimating d from the data. In order to learn $\mathcal{M}$ in $\mathbb{R}^D$, we are given a collection of N data points, $\mathcal{X} \subset \mathbb{R}^D$, that are sampled from $\mathcal{M}$.

The objective is to use the manifold-sampled dataset $\mathcal{X}$ to find a set of tangent patches, $\mathcal{T}$, under constraint on the approximation error (defined later) of each tangent patch. Let $\mathcal{A}$ be a set that indexes the set of tangent patches, $\mathcal{T}$. Then the set of patches can formally be written as:

$$\mathcal{T} = \{\mathcal{T}_k\}_{k \in \mathcal{A}} \text{ such that } \mathcal{T}_k = \{\phi_k, c_k, \mathcal{C}_k, r_k, r_k^{'}\}.$$

Here, $\phi_k \in \mathbb{R}^{D \times d}$ is an orthonormal basis matrix that spans a tangent plane, $c_k \in \mathbb{R}^D$ is the offset of the tangent plane, $\mathcal{C}_k \subset \mathcal{X}$ is the training data associated with the patch $k \in \mathcal{A}$, $r_k, r_k^{'} \in \mathbb{R}^D$ define the subset of the tangent plane that forms the tangent patch $k \in \mathcal{A}$.

Mathematically, the set of points in tangent patch $\mathcal{T}_k$, $k \in \mathcal{A}$, can be expressed as:

$$\hat{\mathcal{M}}_k = \{y | y = \phi_k w + c_k, r_k \preccurlyeq y \preccurlyeq r_k^{'}, w \in \mathbb{R}^d\}, k \in \mathcal{A}, \tag{3.1}$$

where $\preccurlyeq$ represents component-wise inequality. The union of tangent patches (UoTP) approximation of $\mathcal{M}$ can be expressed as:

$$\hat{\mathcal{M}} = \bigcup_{k \in \mathcal{A}} \hat{\mathcal{M}}_k. \tag{3.2}$$

The approximation error associated with each tangent patch $\mathcal{T}_k$, $k \in \mathcal{A}$ can be defined as follows:

$$e_k = \frac{1}{|C_j|} \sum_{x \in C_j} \frac{\|x - \phi_k \phi_k^{\top}(x - c_k) - c_k\|}{\|x - c_k\|}, \tag{3.3}$$

where $\|.\|$ is the Euclidean norm.

Now, we have all the tools to define the manifold approximation problem more concretely. The objective is to find a UoTP based GMA $\hat{\mathcal{M}}$ of the manifold $\mathcal{M}$ such that the approximation error associated with each tangent patch $\mathcal{T}_k$, $k \in \mathcal{A}$ is within ϵ.

To reduce the storage cost of the learnt manifold structure $\hat{\mathcal{M}}$, another objective is to minimize the number of tangent patches approximating $\mathcal{M}$. The error constraint ϵ on each tangent patch ensures that while the number of tangent patches approximating the manifold $\mathcal{M}$ is minimized, each tangent patch $k \in \mathcal{A}$ still remains an accurate estimate of the local manifold geometry. Thus, we essentially minimize the number of tangent patches capturing the manifold geometry under constraints on the accuracy of manifold approximation. Note that the error constraint on each tangent patch makes the final number of patches required for manifold approximation adaptive to the manifold curvature, which addresses the problem of finding the number of linear structures required for manifold approximation in [24–26]. Since $\mathcal{M}$ is approximated using a collection of tangent patches where each tangent patch captures the local manifold geometry, we name our proposed approach as Geometry Preserving Union-of-Tangent-Patches (GP UoTP) algorithm.

3.3 Approximating Manifold Geometry using Tangent Patches

The proposed GP UoTP algorithm for learning a UoTP approximation of $\mathcal{M}$ is explained in Sect. 3.3.1. We formulate the data encoding problem and propose an efficient algorithm for encoding new data samples onto the learnt manifold structure $\hat{\mathcal{M}}$ in 3.3.2.

3.3.1 Learning Geometry-Preserving Union of Tangent Patches Approximation

We begin by associating a neighborhood set $\mathcal{N}_K(x)$ with each data sample $x \in \mathcal{X}$. The neighborhood set $\mathcal{N}_K(x)$ is a set of K nearest neighbors of x in $\mathcal{X}$ with respect to the Euclidean metric. Next, we initialize a collection of tangent patches $\mathcal{T}$ such that a tangent patch $\mathcal{T}_x$ is associated with each $x \in \mathcal{X}$. The associated tangent patch is characterized by an orthonormal basis matrix $\phi_x \in \mathbb{R}^{D \times d}$, obtained via Singular Value Decomposition (SVD) of all the points in $\mathcal{N}_K(x)$, a vector $c_x \in \mathbb{R}^{D \times 1}$, which is the mean of all the samples in $\mathcal{N}_K(x)$, and a training subset $\mathcal{C}_x$, which is initialized as $\mathcal{C}_x = \{x\}$.

Also, we define the subset of the tangent plane $\{\phi_k w + c_k\}, w \in \mathbb{R}^d, k \in \mathcal{A}$, that forms the tangent patch $\mathcal{T}_k$ by the radii r_k and $r_k^{'}$. The radii are defined by $r_{k,i} = \min\limits_{x \in \mathcal{C}_k} x_i$, and $r_{k,i}^{'} = \max\limits_{x \in \mathcal{C}_k} x_i$, $i \in \{1, ..., D\}$, $k \in \mathcal{A}$. Here, x_i denotes the i-th element of x. Defining $r_k = \{r_{k,i}\}$ and $r_k^{'} = \{r_{k,i}^{'}\}$, and using $\mathcal{A}$ to be a set of indeces, the initial set of tangent patches, $\mathcal{T}$, can formally be written as:

$$\mathcal{T} = \{\mathcal{T}_j\}_{j \in \mathcal{A}} \text{ such that } \mathcal{T}_j = \{\phi_j, c_j, \mathcal{C}_j, r_j, r_j^{'}\}.$$

To minimize the number of tangent patches representing the manifold geometry, pairs of tangent patches in $\mathcal{T}$ are fused till further fusion of any pair of tangent patches from $\mathcal{T}$ will give an error (3.3) larger than ϵ in the merged tangent patch. However, to ensure only neighboring tangent patches are fused together, not all pairs of patches in $\mathcal{T}$ are eligible for merging. We use the definition of fusibility of clusters in [26] to define the set of all pairs of fusible patches in $\mathcal{T}$ as

$$\begin{aligned} &\Omega \leftarrow \{(i,j) \in \mathcal{A} : y_i \in C_i, y_j \in C_j \text{ such that} \\ &y_i \in \mathcal{N}_K(y_j) \text{ or } y_j \in \mathcal{N}_K(y_i)\}. \end{aligned} \tag{3.4}$$

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

$$(i^*, j^*) = \underset{(i,j) \in \Omega}{\arg\min}\; e_{proj}(\mathcal{T}_i, \mathcal{T}_j), \tag{3.5}$$

$$\text{such that } \; e_{proj}(\mathcal{T}_i, \mathcal{T}_j) \leq \epsilon,$$

$$e_{proj}(\mathcal{T}_i, \mathcal{T}_j) = \frac{1}{|\mathcal{C}_k|} \sum_{x \in \mathcal{C}_k} \frac{\|x - \phi_k \phi_k^\top (x - c_k) - c_k\|_2}{\|x - c_k\|_2}. \tag{3.6}$$

Here, if we use $\mathcal{T}_k$ to denote the tangent patch obtained from merging of $\mathcal{T}_i$ and $\mathcal{T}_j$ then $\mathcal{C}_k = \mathcal{C}_i \cup \mathcal{C}_j$ is the set of training data associated with $\mathcal{T}_k$, c_k is the empirical mean of all the samples in $\mathcal{C}_k$, and ϕ_k is computed by finding the d largest eigenvectors of the matrix $U_k = \frac{1}{2}(\phi_i \phi_i^\top + \phi_j \phi_j^\top)$. Note that $\mathcal{T}_k$ is essentially a mean tangent patch to the patches $\mathcal{T}_i$ and $\mathcal{T}_j$ because U_k can be seen as a solution to the following:

$$U_k = \underset{U \in \mathbb{R}^{D \times D}}{\arg\min} \frac{1}{2}(\|\phi_i \phi_i^\top - U\|_F^2 + \|\phi_j \phi_j^\top - U\|_F^2).$$

Finally, we merge the best pair of patches $(\mathcal{T}_{i^*}, \mathcal{T}_{j^*})$ to obtain the tangent patch $\mathcal{T}_{k^*}$, resulting in a reduced set of patches

$$\begin{aligned} \mathcal{T} &\leftarrow (\mathcal{T} \setminus \{\mathcal{T}_{i^*}, \mathcal{T}_{j^*}\}) \cup \{\mathcal{T}_{k^*}\}, \\ \mathcal{A} &\leftarrow (\mathcal{A} \setminus \{i^*, j^*\}) \cup \{k^*\}. \end{aligned} \tag{3.7}$$

Summarizing our algorithm, once the set $\mathcal{T}$ is initialized by calculating the tangent patch at each $x \in \mathcal{X}$, the set of pairs of fusible tangent patches Ω is calculated as in (3.4), the best pair $(\mathcal{T}_{i^*}, \mathcal{T}_{j^*})$ from the set of fusible pairs of patches is selected using (3.5), and the best pair of patches is merged as in (3.7). This process of evaluating (3.4), (3.5) and (3.7) is repeated till we keep on getting a solution for (3.5). In other words, we keep on finding and merging the best pair of fusible tangent patches till the best pair of fusible patches does not satisfy the approximation error constraint. The final union-of-tangent-patches approximation of $\mathcal{M}$ is then obtained as:

$$\hat{\mathcal{M}} = \bigcup_{k \in \mathcal{A}} \{y : y = \phi_k w + c_k, r_k \preccurlyeq y \preccurlyeq r_k^{'}, w \in \mathbb{R}^d\}. \tag{3.8}$$

The accuracy of manifold geometry approximation of $\mathcal{M}$ by $\hat{\mathcal{M}}$ depends on the value of ϵ: the smaller the value of ϵ, the more accurate the approximation and vice versa. The complete algorithm is outlined in Algorithm 2.

The radii are defined by $r_{k,i} = \min\limits_{x \in \mathcal{C}_k} x_i$, and $r_{k,i}^{'} = \max\limits_{x \in \mathcal{C}_k} x_i$, $i \in \{1, ..., D\}$, $k \in \mathcal{A}$. Here, x_i denotes the i-th element of x. Defining $r_k = \{r_{k,i}\}$ and $r_k^{'} = \{r_{k,i}^{'}\}$

3.3.2 Encoding New Data Samples

In this section, we discuss how a point $x \in \mathbb{R}^D$ can be projected onto the learnt manifold structure $\hat{\mathcal{M}}$ to get $\hat{x}$ such that $\hat{x} = \underset{z \in \hat{\mathcal{M}}}{\arg\min} \|x - z\|_2$. In principle, we can project the data sample $x \in \mathbb{R}^D$ onto each of the tangent patches $\hat{\mathcal{M}}_k$, $k \in \mathcal{A}$, and then the patch $\hat{\mathcal{M}}_{k^*}$, $k^* \in \mathcal{A}$ that minimizes the mean square error of projection onto $\hat{\mathcal{M}}$ can be used for encoding x onto $\hat{\mathcal{M}}$. Mathematically, this problem can be formulated as the

Algorithm 2: Learning Geometry-Preserving Union of Tangent Patches (GP UoTP)

```
Input: Dataset: $\mathcal{X}$; neighborhood size: $K$; maximum error: $\epsilon$; dimension of tangent patches: $d$
Output: Final set of patches: $\mathcal{T}_j = \{\phi_j, c_j, \mathcal{C}_j, r_j, r_j^{'}\}_{j \in \mathcal{A}}$
Initialization: Initializing a tangent patch with each $x \in \mathcal{X}$:
$\mathcal{N}_K(x) \leftarrow \{K \text{ nearest neighbors of } x \text{ in } \mathcal{X}\}, x \in \mathcal{X}$
$c_x \leftarrow \frac{1}{|\mathcal{N}_x|} \sum_{y \in \mathcal{N}_x} y$
$[\mathcal{N}_x^0] \leftarrow \{y - c_x : y \in \mathcal{N}_K(x)\}$
$\phi_x \leftarrow$ left singular vectors of $[\mathcal{N}_x^0]$ corresponding to its $d$-largest singular values
$\mathcal{C}_x \leftarrow \{\{x\} : x \in \mathcal{X}\}$
Let the set of all tangent patches be indexed by a set $\mathcal{A}$ such that $\mathcal{T} \leftarrow \{\mathcal{T}_j\}_{j \in \mathcal{A}}$ with $\mathcal{T}_j = \{\phi_j, c_j, \mathcal{C}_j\}$
Merging the tangent patches:
loop
  $\Omega \leftarrow \{(i,j) \in \mathcal{A} : y_i \in C_i, y_j \in C_j$ such that $y_i \in \mathcal{N}_K(y_j)$ or $y_j \in \mathcal{N}_K(y_i)\}$
  $(i^*, j^*) = \arg\min_{(i,j) \in \Omega} e_{proj}(\mathcal{C}_i, \mathcal{C}_j)$
  if $e_{proj}(\mathcal{C}_{i^*}, \mathcal{C}_{j^*}) < \epsilon$ then
    $\mathcal{T} \leftarrow (\mathcal{T} \setminus \{\mathcal{T}_{i^*}, \mathcal{T}_{j^*}\}) \cup \{\mathcal{T}_{k^*}\}$, where $\mathcal{T}_{k^*}$ is the patch obtained from merging patches $\mathcal{T}_{i^*}$ and $\mathcal{T}_{j^*}$
  else
    break the loop
  end if
end loop
$r_j = \{\min_{x \in \mathcal{C}_j} x_i\}_{i=1}^{i=D}$, $r_j^{'} = \{\max_{x \in \mathcal{C}_j} x_i\}_{i=1}^{i=D}$, $j \in \mathcal{A}$
```

following sequence:

$$\begin{aligned} & w_k = \arg\min_{w \in \mathbb{R}^d} \|\phi_k w + c_k - x\|_2^2 \\ & \text{subject to} \quad r_k \preccurlyeq \phi_k w + c_k \preccurlyeq r_k^{'},\ k \in \mathcal{A}, \text{ and} \qquad (3.9) \\ & k^* = \arg\min_{k \in \mathcal{A}} \|\phi_k w_k + c_k - x\|_2^2. \qquad (3.10) \end{aligned}$$

Once the projection coefficients of x onto each patch in $k \in \mathcal{A}$ are calculated in (3.9), the encoding patch can be selected using (3.10). The final solution of the encoding procedure is then (k^*, w_{k^*}) where k^* is the tangent patch selected for projecting x onto $\mathcal{M}$ and w_{k^*} represents the encoding coefficients. Thus, we have formulated the problem of encoding x onto $\hat{\mathcal{M}}$ as a series of convex optimization problems, one for each $k \in \mathcal{A}$ in (3.9). However, the problem of solving each of the convex optimization problems in

(3.9) still needs to be addressed.

We note that each tangent patch $\hat{\mathcal{M}}_k, k \in \mathcal{A}$ can be seen as the intersection of an affine subspace B_k and a polyhedron D_k defined as: $B_k = \{y : y = \phi_k w + c_k, w \in \mathbb{R}^D\}$, and $D_k = \{y : r_k \preccurlyeq y \preccurlyeq r_k^{'}\}$. Thus, the problem of projection onto patch $\mathcal{T}_k$ can be seen as a problem of projection onto intersection of convex sets B_k and D_k. This hints towards the possible use of projection onto convex sets (POCS) method [30]. POCS algorithm can be used for finding a point in the intersection of given closed convex sets. However, to project x onto $\mathcal{T}_k$, which is equivalent to solving (3.9) for a given $k \in \mathcal{A}$, we need the projection of x onto $B_k \cup D_k$.

To rewrite the encoding problem in a convenient form, we define the following indicator functions for each tangent patch. For each $k \in \mathcal{A}$, let f_k be the indicator function of set B_k and g_k be the indicator function of set D_k such that:

$$f_k(x) = \begin{cases} 0, & x \in B_k; \\ +\infty, & \text{otherwise,} \end{cases}$$

$$g_k(x) = \begin{cases} 0, & x \in D_k; \\ +\infty, & \text{otherwise.} \end{cases}$$

Then (3.9) and (3.10) can be rewritten as:

$$y_k = \operatorname*{arg\,min}_{y \in \mathbb{R}^D} f_k(y) + g_k(y) + \|y - x\|_2^2,\ k \in \mathcal{A}, \tag{3.11}$$

$$k^* = \operatorname*{arg\,min}_{k \in \mathcal{A}} \|y_k - x\|_2^2. \tag{3.12}$$

Considering that, for each patch $k \in \mathcal{A}$, the summation of indicator functions $(f_k + g_k)(y) \neq \infty$ only when $y \in B_k \cap D_k$, the solution to (3.11) for each $k \in \mathcal{A}$ is the projection of x onto the set $\hat{\mathcal{M}}_k = B_k \cap D_k$. A recent paper [31] shows that (3.11) has a unique solution: $y_k = prox_{f_k + g_k}(x)$, where $prox_{f_k + g_k}(.)$ refers to the proximity operator of the sum of the functions f_k and g_k. Proximity operator associated with $f_k + g_k$ can be looked upon as a generalization of the notion of projection onto the underlying convex set $B_k \cap D_k$. To the best of our knowledge, there is no operator for projection onto a subset of an affine subspace, which is $\mathcal{T}_k = B_k \cap D_k$ in this case. Thus, $y_k = prox_{f_k+g_k}(x)$ cannot be calculated directly. However, we can use one of the proximal methods outlined in [31], mentioned as Dykstra's projection algorithm [32], to split the problem of projecting onto $B_k \cap D_k$ into projecting onto sets B_k and D_k separately. In other words, (3.11) can be solved even if $prox_{f_k+g_k}(.)$ is unknown, but the proximity operators for B_k and D_k are available. The projection operator for set B_k is defined as:

$$P_{B_k}(z) = \phi_k \phi_k^T (z - c_k) + c_k, \quad z \in \mathbb{R}^D. \tag{3.13}$$

The projection operator for set D_k is defined as:

$$P_{D_k}(z) = \{v_i\}_{i=1}^{i=D}, \quad z \in \mathbb{R}^D, \tag{3.14}$$

where

$$v_i = \begin{cases} z_i, & r_{k,i} \le z_i \le r'_{k,i}; \\ \underset{q \in \{r_{k,i}, r'_{k,i}\}}{\arg\min} \; |z_i - q|, & \text{otherwise.} \end{cases}$$

Given the projection operators for the sets B_k and D_k, Dykstra's projection algorithm for projecting $x \in \mathbb{R}^D$ onto $\hat{\mathcal{M}}_k = B_k \cap D_k$ is outlined in Algorithm 3.

Algorithm 3: Dykstra's Projection Algorithm

```
Initialize $x_0 = x$, $p_0 = 0$, $q_0 = 0$
for $n = 0, 1, 2, ...$ do
    $y_n = P_{B_k}(x_n + p_n)$
    $p_{n+1} = x_n + p_n - y_n$
    $x_{n+1} = P_{D_k}(y_n + q_n)$
    $q_{n+1} = y_n + q_n - x_{n+1}$
end for
```

If we use Dykstra's projection algorithm (Algorithm 3) to find the projection of x onto $\hat{\mathcal{M}}_k$, the sequence of output $\{x_i\}, i = 1, 2, 3, ...$ produced by Dykstra's algorithm is guaranteed to converge to the projection of x onto $\hat{\mathcal{M}}_k$ [33]. The final output of the encoding procedure outlined by (3.11) and (3.12) is the encoding patch k^* and the projection of $x \in \mathbb{R}^D$ onto the encoding patch $y_{k^*} \in \mathbb{R}^D$. The encoding coefficients, $w_{k^*} \in \mathbb{R}^d$, can be calculated using $w_{k^*} = \phi_{k^*}^\top (y_k^* - c_k^*)$.

3.4 Experimental Results

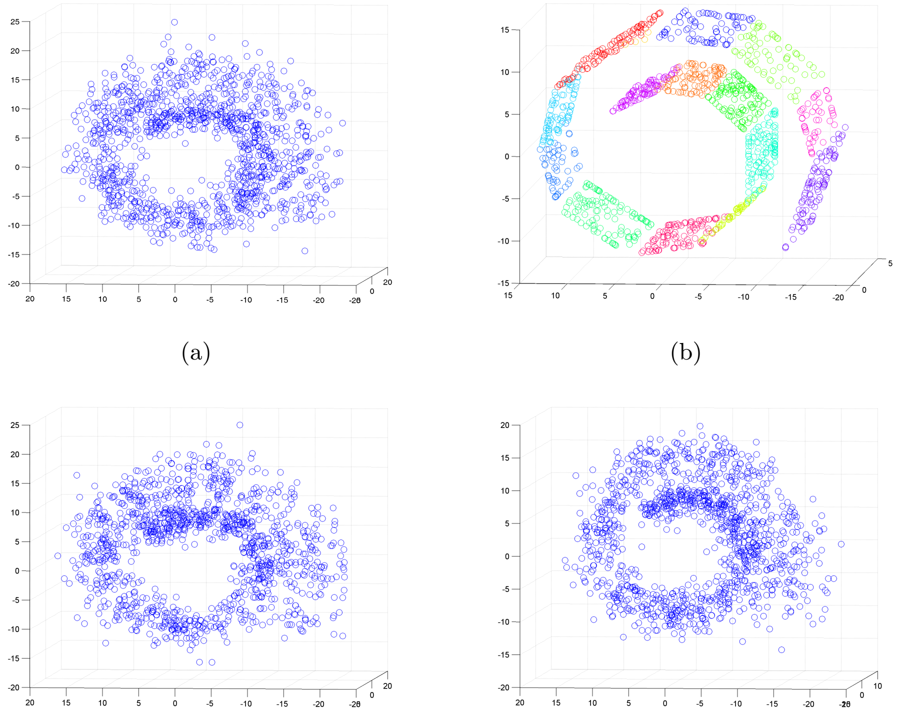
To demonstrate the value in preserving geometry of the underlying manifold structure when approximating manifold-sampled data, we perform denoising experiments for manifold-sampled noisy data. In the experiments performed on synthetic data, the training data is generated by uniformly at random sampling 1200 *training* data points from 3 half-turns of a swiss roll in $\mathbb{R}^3$. The proposed Algorithm 1 is used to learn a union of tangent patches approximation of the training data. The K-SVD algorithm [5] is used to approximate the training data using an overcomplete dictionary such that the number of dictionary atoms and sparsity level are set to 20 and 2, respectively.

We randomly sample 1200 *test* data points from 3 half-turns of a swiss roll where the test data samples are arranged column-wise in a matrix Y. Additive white gaussian noise (AWGN) is added to Y to get a noisy version of the test dataset: Y_n. To denoise the noisy test dataset using the GP UoTP framework, Y_n is projected on the learnt GP UoTP structure using Algorithm 3 to get the denoised dataset $\hat{Y}$. Similarly, when denoising using K-SVD, Y_n is sparse coded on the learnt dictionary using the Orthogonal Matching Pursuit (OMP) algorithm [34]. The dataset Y_n is also denoised using the Haar wavelet basis at a scale of 2 and an optimized soft threshold of 1.05 to get the denoised dataset $\hat{Y}$. The mean square error (MSE) of denoising the dataset Y_n is defined as:

$$MSE = \frac{1}{N} \|Y - \hat{Y}\|_F^2 \tag{3.15}$$

where N is the number of data samples. In the experiments involving the swiss roll, $N = 1200$ and the SNR is set to 10dB, where SNR $= 10 \log \frac{\|Y\|_F^2}{\mathbb{E}[\|Y - Y_n\|_F^2]}$. The noisy test dataset Y_n is shown in Fig. 3.1(a) and the denoised dataset from the GP UoTP, K-SVD and wavelet based approaches can be seen in Fig. 3.1(b), 3.1(c) and 3.1(d), respectively. Note the gain in denoising performance is possible because of following a geometric approach to denoising, which hints towards the potential of working with GMA-based information processing algorithms.

Denoising experiments are also performed on a real-world dataset: the MNIST database [28]. The dataset Y is compiled by randomly selecting 1000 images of the digit 0 from the MNIST database and arranging the images column-wise in the matrix



(a) (b)

(c) (d)

Figure 3.1: (a) Noise added to data sampled from 3-half turns of a swiss roll (SNR = 10dB). Data is denoised using (b) GP UoTP (MSE = 9.69 dB), (c) K-SVD (MSE = 11.88 dB) and (d) Haar wavelet thresholding (MSE = 10.98 dB).

Y. The GP UoTP, Merging based on Difference of Tangents (MDoT) [24] and the Median K-Flats (MKF) [1] algorithms are used to learn a union of tangent patches, a union of tangent planes and a union of flats approximation of the dataset Y, respectively. The dimension of the patch, plane or flat used for data approximation is set to 5 for these experiments. Additive white gaussian noise is added to the dataset Y in the same manner as before to get a noisy version of the dataset Y_n. The dataset Y_n is denoised by projecting it on the learnt structure (union of patches, planes or flats) for each of the aforementioned algorithms to get the denoised dataset $\hat{Y}$. The experiments described above for denoising based on GP UoTP, MDoT and MKF algorithms are repeated for varying number of patches, planes and flats respectively. The denoising performance for each algorithm can be seen in Fig. 3.2 where the mean square error (MSE) is as defined in (3.15).

Similar experiment for denoising Y_n from the MNIST dataset is also repeated for

K-SVD [5] algorithm. For K-SVD, the number of dictionary atoms are varied while the sparsity level is set to 5 (the number of dictionary atoms have to be less than $N = 1000$). The comparison between Fig. 3.2 and 3.3 shows that if we use the GP UoTP denoising procedure with large enough number of patches, the GP UoTP algorithm gives better denoising performance than the K-SVD algorithm.

We also display a few examples of noisy versions of images from the MNIST dataset, and their corresponding denoised versions obtained using the aforementioned denoising algorithms. In particular, AWGN is added to three different images of digit 0 from the MNIST dataset. To denoise each of the three noisy images using the GP UoTP framework, each image is projected on the learnt GP UoTP structure using Algorithm 3. For the K-SVD algorithm, the number of dictionary atoms are set at 200 and the dictionary is learnt using the K-SVD algorithm on the dataset Y. Each noisy image is sparse coded on the learnt dictionary using the Orthogonal Matching Pursuit (OMP) algorithm [34]. Finally, we also denoise the three example noisy images using the Haar wavelet basis at a scale of 2 and a threshold of $\sqrt{2 * \log(D) * Pn}$ where Pn is the noise power and $D = 28 \times 28$ in this case. The noisy images and their corresponding denoised versions are displayed in Fig. 3.4.

From the figure, it can be seen the denoising results using the wavelet based method are worst than the denoising results obtained via the GP UoTP encoding method. The denoising performance of the K-SVD algorithm is comparable with the denoising performance of the GP UoTP algorithm. However, note that the number of tangent patches in the GP UoTP algorithm are set at 5; whereas, the number of dictionary atoms for the K-SVD algorithm have to be set at much greater than 5 to get comparable results.

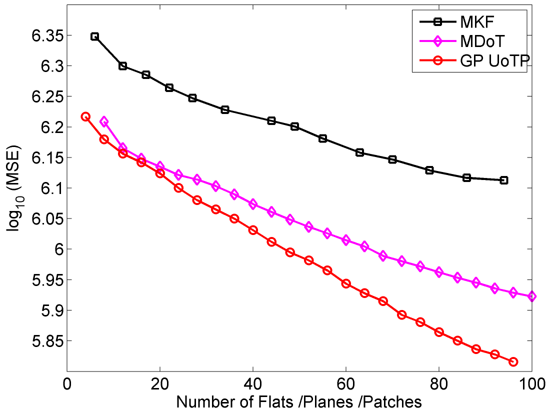


Figure 3.2: Noise is added to 1000 images of digit 0 extracted from the MNIST database (SNR = 10dB). Noisy data is projected onto the learnt flats/planes/patches for denoising.

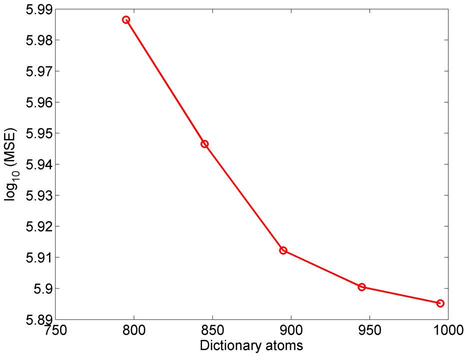


Figure 3.3: Noise is added to 1000 images of digit 0 extracted from the MNIST database (SNR = 10dB). For denoising, the noisy data is sparse coded on the dictionary learnt using K-SVD.

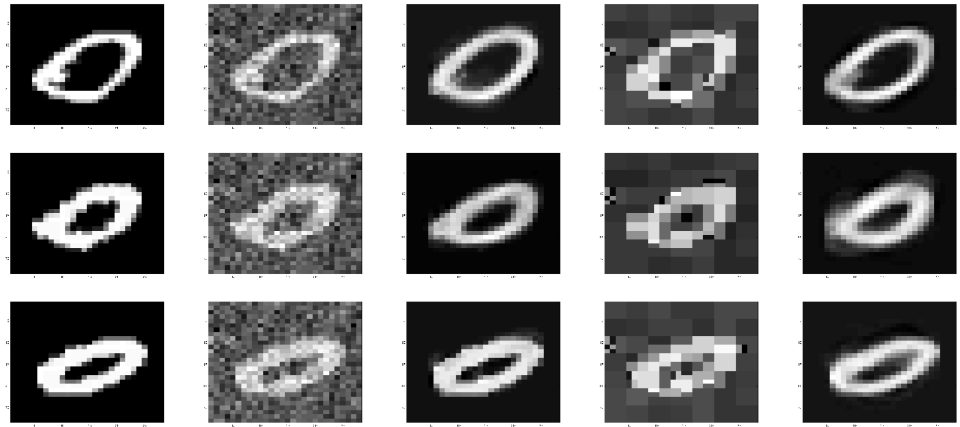


Figure 3.4: In column 1, three examples of digit 0 from the MNIST dataset are displayed. Noise is added to each image in column 1 such that the SNR is 5dB, and the noisy images are shown in column 2. The noisy images in column 2 are encoded onto the GP UoTP and the learnt dictionary atoms in column 3 and 5, respectively. The denoising results based on wavelet based thresholding method are given in column 4. The average logarithm reconstruction error in column 3 is 5.521, in column 4 is 6.150 and in column 5 is 5.789. The logarithm reconstruction error is defined as log of the energy in the difference between the images in column 1 and images in columns $3, 4$ and 5, respectively.

Chapter 4

Conclusions

Many real-world signals may not have a subspace or a union-of-subspace representation, but they may be manifested in a low-dimensional non-linear manifold embedded in a high-dimensional ambient space. In this thesis, we proposed structure learning as well as data encoding algorithm for classes of signals well represented by the manifold model. We showed the value in capturing manifold geometry while approximating the manifold structure using a union of linear structures. Our experimental results show that geometry preserving manifold approximations can pave way for information processing algorithms for manifold sampled data that perform better than the state-of-the-art methods for data processing.

In particular, in Chapter 2, we proposed a bottom-up approach to approximating manifold structure using union of tangent planes. The manifold approximation stage is cascaded with a subsampling stage that exploits the local linearity of the manifold for gready subsampling of the training dataset. Owing to this preprocessing step, our algorithm shows particularly impressive computational complexity performance when learning manifold geometry using dense training dataset. 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 even as the manifold curvature increases.

In Chapter 3, we proposed an algorithm for learning a union of tangent patches approximation of a non-linear non-intersecting manifold. Moreover, we proposed an efficient routine for encoding new data points onto the union of tangent patches approximating the manifold. We also demonstrated the value of preserving the manifold geometry in manifold approximation by demonstrating the denoising performance of the proposed structure learning and data encoding algorithm.

4.1 Future Directions

The value of preserving manifold geometry in various information processing tasks needs to be further explored. The value of capturing manifold geometry in denoising manifold sampled data has been demonstrated in this work, but preserving geometry may lead to better classification (or perhaps more efficient sampling paradigms) for classes of signals living on a manifold.

An important direction is to characterize the relationship between various metrics of interest in our approximation approach: manifold curvature, maximum local approximation error ϵ, number of approximating planes, ambient space dimension, intrinsic dimension of manifold, sampling density of the training dataset on the manifold. In particular, the following are some interesting questions that need to be addressed:

- For a given value of ϵ, how do the number of approximating tangent planes scale with the manifold curvature?
- For a given value of ϵ and manifold curvature, can we find an upper bound on the approximation error of a sample from the true manifold?
- How do we need to scale the value of ϵ with the manifold curvature and the sampling density of the training dataset on the manifold such that the approximation error of the true manifold is within a given range?

Bibliography

[1] 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*, pages 234–241, 2009.

[2] R. G. Baraniuk, V. Cevher, and M. B. Wakin. Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective. *Proc. of the IEEE*, 98(6):959–971, 2010.

[3] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. Lee, and T. J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Computation*, 15(2):349–396, 2003.

[4] J. Mairal et al. Online dictionary learning for sparse coding. In *Proc. of the 26th Annu. Int. Conf. on Mach. Learning*, pages 689–696. ACM, 2009.

[5] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.*, 54(11):4311–4322, 2006.

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

[7] G. E. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Trans. Neural Netw.*, 8(1):65–74, 1997.

[8] Trevor F Cox and Michael AA Cox. *Multidimensional scaling.* CRC press, 2000.

[9] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric

framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[10] Kilian Q Weinberger and Lawrence K Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *AAAI*, volume 6, pages 1683–1686, 2006.

[11] Lawrence K Saul and Sam T Roweis. An introduction to locally linear embedding. *unpublished. Available at: http://www. cs. toronto. edu/˜ roweis/lle/publications. html*, 2000.

[12] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[13] David L Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003.

[14] Zhen-yue Zhang and Hong-yuan Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *Journal of Shanghai University (English Edition)*, 8(4):406–424, 2004.

[15] Yee W Teh and Sam T Roweis. Automatic alignment of local representations. In *Advances in neural information processing systems*, pages 841–848, 2002.

[16] Matthew Brand. Charting a manifold. In *Advances in neural information processing systems*, pages 961–968, 2002.

[17] Jie Ni, Pavan Turaga, Vishal M Patel, and Rama Chellappa. Example-driven manifold priors for image deconvolution. *Image Processing, IEEE Transactions on*, 20(11):3086–3096, 2011.

[18] Sam T Roweis, Lawrence K Saul, and Geoffrey E Hinton. Global coordination of local linear models. *Advances in neural information processing systems*, 2:889–896, 2002.

[19] Raffaele Cappelli and Davide Maltoni. Multispace kl for pattern representation and classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(9):977–996, 2001.

[20] Rene Vidal, Yi Ma, and Shankar Sastry. Generalized principal component analysis (gpca). *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(12): 1945–1959, 2005.

[21] Yi Ma, Allen Y Yang, Harm Derksen, and Robert Fossum. Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM review*, 50(3):413–458, 2008.

[22] Ying Wu, Zhengyou Zhang, Thomas S Huang, and John Y Lin. Multibody grouping via orthogonal subspace decomposition. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–252. IEEE, 2001.

[23] Z. Y. Zhang and H. Y. Zha. Principal manifolds and nonlinear dimension reduction via tangent space alignment. *J. Shanghai Univ.*, 8(4):406–424, 2004.

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

[25] 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.*, 32(3):435–462, 2012.

[26] S. Karygianni and P. Frossard. Tangent-based manifold approximation with locally linear models. *Signal Processing*, 104:232 – 247, 2014.

[27] B. Kgl. Intrinsic dimension estimation using packing numbers. In *Adv. Neural Inform. Process. Syst.*, pages 681–688, 2002.

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

[29] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[30] H. H. Bauschke and J. M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996.

[31] P. L. Combettes and J. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Eng.*, pages 185–212. Springer, 2011.

[32] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in Order Restricted Statistical Inference*, pages 28–47. Springer, 1986.

[33] H. H. Bauschke and P. L. Combettes. A Dykstra-like algorithm for two monotone operators. *Pacific J. of Optimization*, 4(3):383–391, 2008.

[34] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Conf. Rec. of The Twenty-Seventh Asilomar Conf. on Signals, Syst. and Comput.*, pages 40–44. IEEE, 1993.