

# S-Isomap++: Multi Manifold Learning from Streaming Data

Suchismit Mahapatra  
Computer Science and Engineering  
State University of New York at Buffalo  
suchismi@buffalo.edu

Varun Chandola  
Computer Science and Engineering  
State University of New York at Buffalo  
chandola@buffalo.edu

**Abstract**—Manifold learning based methods have been widely used for non-linear dimensionality reduction (NLDR). However, in many practical settings, the need to process streaming data is a challenge for such methods, owing to the high computational complexity involved. Moreover, most methods operate under the assumption that the input data is sampled from a single manifold, embedded in a high dimensional space. We propose a method for streaming NLDR when the observed data is either sampled from multiple manifolds or irregularly sampled from a single manifold. We show that existing NLDR methods, such as Isomap, fail in such situations, primarily because they rely on smoothness and continuity of the underlying manifold, which is violated in the scenarios explored in this paper. However, the proposed algorithm is able to learn effectively in presence of multiple, and potentially intersecting, manifolds, while allowing for the input data to arrive as a massive stream.

**Keywords**-Manifold Learning; Streaming Data; Isomap; Clustering;

## I. INTRODUCTION

Ability to analyze massive streams of data is a valuable aspect of any modern data science pipeline. This is important in many contexts, such as high-performance high-fidelity numerical simulations [28], high-resolution scientific instrumentation (microscopes, DNA sequencers, etc.) [29], and even *Internet of Things* [31], where a huge number of devices are currently connected to the Internet and feeding a variety of data streams. Such data sources typically monitor or measure complex system behaviors, using a large number of parameters. Dimensionality reduction methods [30] are typically used to map the resulting high-dimensional data into a smaller, manageable space. If the data is assumed to lie on a hyperplane, linear dimensionality reduction methods such as Principal Component Analysis (PCA) [23], etc., maybe applied. However, in many settings, especially when dealing with complex scientific and natural phenomenon, the data might lie on a non-linear manifold, in which case, non-linear dimensionality reduction methods are more appropriate.

Non-linear dimensionality reduction(NLDR) comes at a cost; most existing NLDR methods have a computational complexity of  $O(n^3)$ ,  $n$  being the size of the data. The issue is further exacerbated when the data is streaming, where obtaining exact solution at every step of the stream is

computationally infeasible. While adaptations of existing NLDR methods, such as Isomap [1] and Local Linear Embedding (LLE) [4], have been proposed for handling data streams [10], [25], such methods, which typically rely on incremental updates of the underlying solution, do not scale well to massive streams. In a recent work [7], a two phase strategy has been proposed to adapt Isomap to streaming data. The algorithm, called *S-Isomap*, operates on the core principle that a small batch of data is necessary to *learn* the underlying small-dimensional manifold using an exact and computationally expensive, but data-bounded, learning method. The remainder of the stream may be *mapped* onto the learnt manifold using a relatively inexpensive mapping procedure.

However, the above solution, and other related efforts to adapt NLDR methods to streaming data [10], rely on the assumption that the data samples lie on a *single* low-dimensional manifold. There have been limited attempts that allow for multiple manifolds [13], [14], however, they assume that the manifolds do not intersect in any ambient space. This is illustrated in Figures 1 and 2. In Figure 1, the synthetic data set in the top panel consists of four “patches” in 2D space which are embedded onto different regions of a 3D *Swiss-Roll*. Thus the 3D patches data set maybe considered as the high-dimensional data set consisting of samples from multiple manifolds. Direct application of Isomap, which assumes that data comes from a single manifold, results in poor recreation of the ground truth (Figure 1c). An existing method, *M-Isomap* [13], that explicitly handles multiple manifolds, gives somewhat better results (Figure 1d). In Figure 2, the synthetic data set consists of data from two 2D manifolds embedded in a 3D space, as an isometric swiss-roll and a plane, intersecting with each other. In this case, both Isomap and M-Isomap fail (Figure 2c), primarily because M-Isomap assumes that the multiple manifolds do not intersect.

The core contribution of this paper is a streaming non-linear dimensionality reduction algorithm, called **S-Isomap++**. The algorithm assumes that the high dimensional input data consists of samples that truly lie on one or more, potentially intersecting, low-dimensional manifolds and are embedded into the high dimensional space via non-linear transformations. The proposed algorithm extends the widely

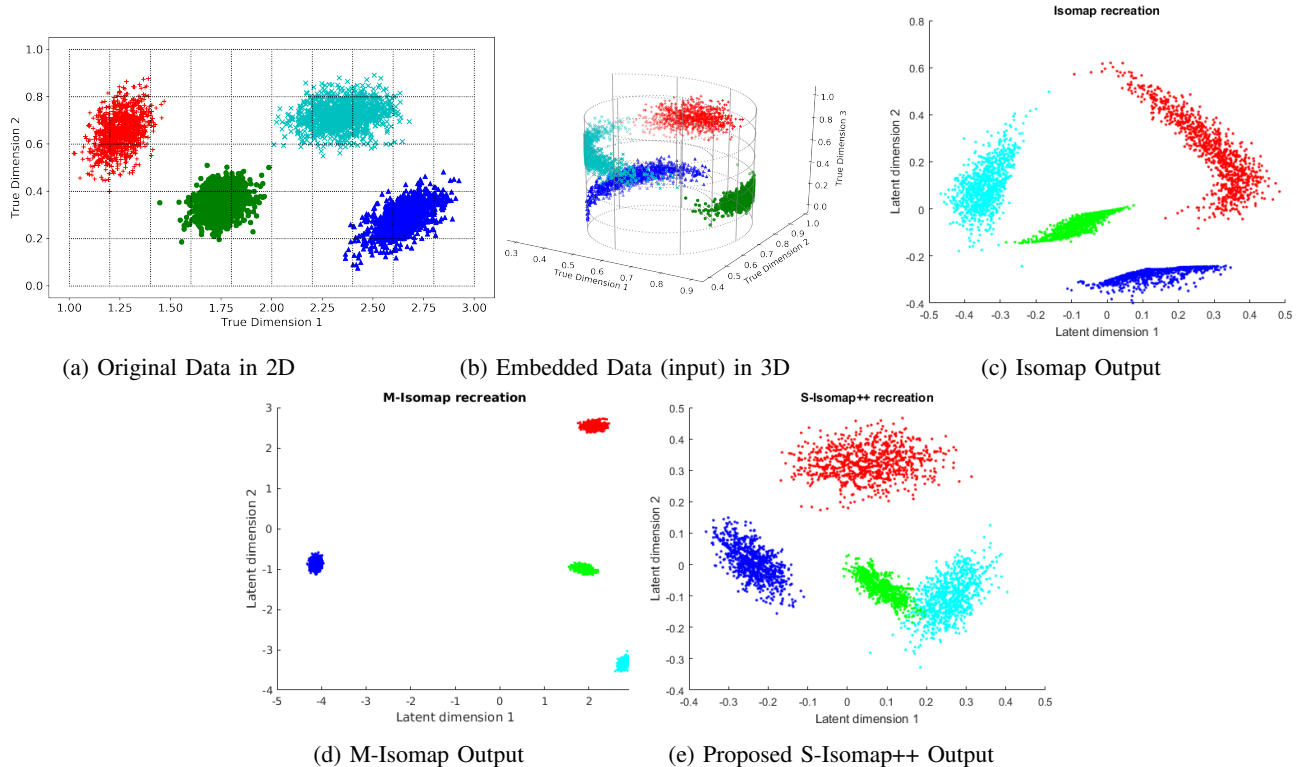


Figure 1: Multi-manifold *patches* data set. The 2D samples in (a) are embedded into 3D in (b) via the Euler Isometric mapping technique [7]. The reduction to 2D is obtained using: (c). Isomap, (d). M-Isomap [13], and (e). the proposed S-Isomap++ algorithm.

used Isomap algorithm to handle multiple intersecting manifolds in a streaming setting. Thus, the proposed algorithm operates under one of the least restrictive set of assumptions, explored so far in the context of NLDR methods (See Figures 1e and 2d). Moreover, the ability to handle large streams of data makes it highly applicable in a broad variety of domains.

Another contribution of the paper is a novel *tangent based clustering* strategy to separate samples from the input batch, in the original high-dimensional space, into different clusters. Each cluster is processed independently to obtain the manifold and the corresponding low-dimensional reduction of the corresponding data samples, using Isomap. The reduced data samples are then mapped into a common *ambient* space by exploiting the relationship between the samples across the clusters in the original space. The streaming samples are then mapped, in parallel, on each manifold. An evaluation strategy is employed to choose the best manifold for each streaming sample.

The rest of the paper is organized as follows: we provide necessary background about manifold learning in Section II. Related works are discussed in Section III. The proposed algorithm, S-Isomap++, is presented in Section IV. Experimental results on synthetic and benchmark datasets, are

summarized in Section V.

## II. BACKGROUND AND MOTIVATION

Our motivation for this work stems from one of the foundation principles of *Manifold Learning*, which assumes that the distribution of the data in the high-dimensional observed space is not uniform and in reality, the data lies near a non-linear low-dimensional manifold embedded in the high-dimensional space. In many real-world problems such as those resulting from multi-modal or unevenly sampled distributions, the data lies on multiple manifolds of possibly different “dimensionalities” and is typically separated by regions of low density as depicted in Figure 3. Thus, to find a representative low-dimensional embedding of the data, one needs to first cluster the data appropriately and subsequently find a low-dimensional representation for the data in each cluster. Even then, manifolds can be very close to each other and can have arbitrary intrinsic dimensions, curvature and sampling which makes it a hard problem to solve.

### A. Defining a Manifold

Mathematically, a manifold  $\mathcal{M}$  is defined as a metric space with the following property: if  $x \in \mathcal{M}$ , then there exists some neighborhood  $\mathcal{U}$  of  $x$  and  $\exists n$  such that  $\mathcal{U}$  is homeomorphic to  $\mathbb{R}^n$  [16].

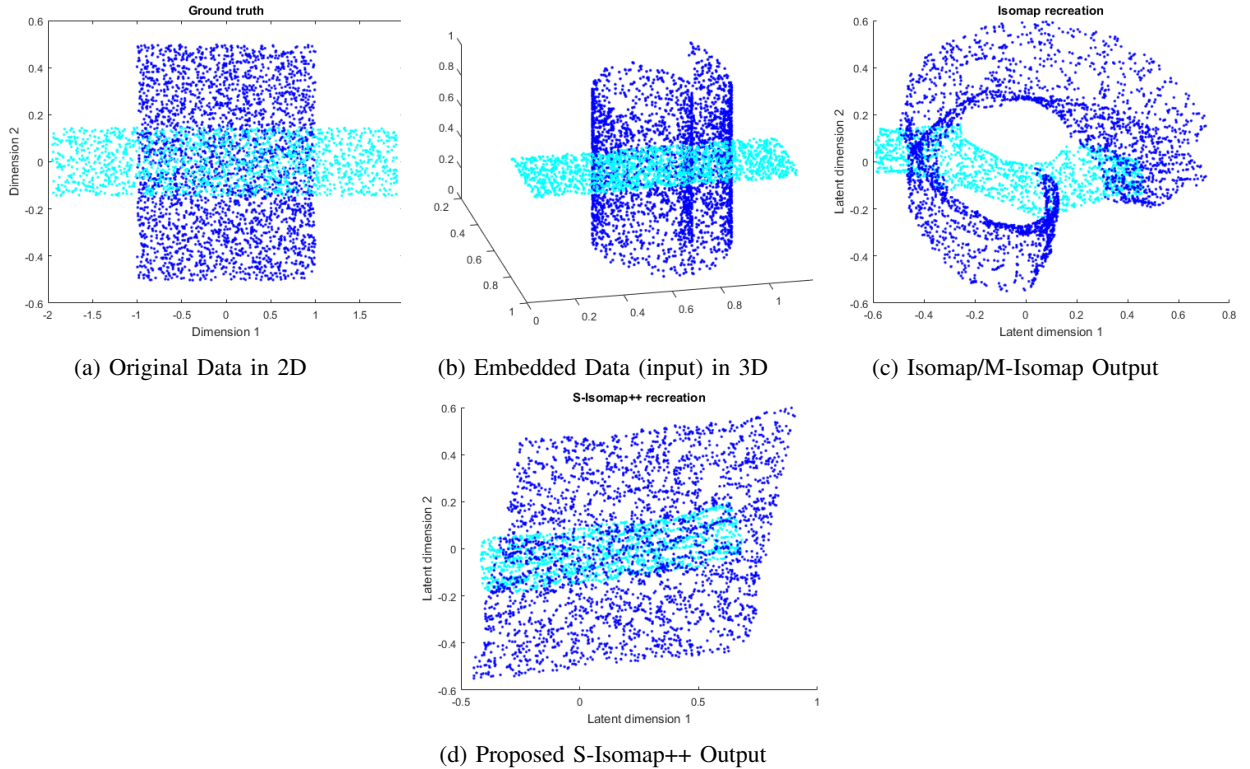


Figure 2: Multi-manifold *intersecting* data set. One set of 2D samples (blue) in (a) are embedded into 3D in (b) via the Euler Isometric mapping technique [7]. Second set (cyan) are embedded using a linear mapping. The reduction to 2D is obtained using: (c). Isomap/M-Isomap, and (d). the proposed S-Isomap++ algorithm. Both Isomap and M-Isomap give the same output because M-Isomap cannot handle intersecting manifolds and, thus, reverts to a single manifold scenario.

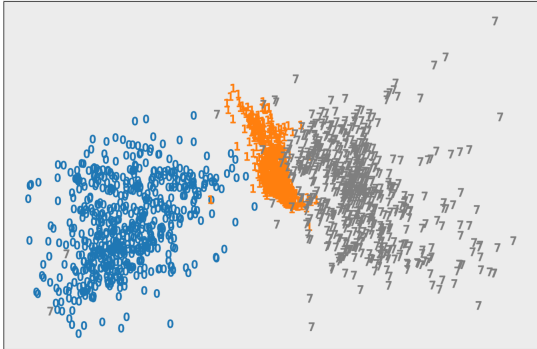


Figure 3: 2-D reduction of a sample of images from the MNIST digits dataset. Real-world data generally lies near multiple manifolds and is usually separated by regions of low density.

The global structure of the high-dimensional ambient space can be more complicated. Usually manifolds are embedded in high-dimensional spaces, but the intrinsic dimensionality is typically low due to fewer degrees of freedom in the underlying data generating process.

## B. Nonlinear Dimensionality Reduction

Typically, nonlinear dimensionality reduction (NLDR) techniques are used as learning methods for discovering the underlying low-dimensional structure from samples from high-dimensional data. Existing techniques typically exploit either the global (Isomap, Minimum Volume Embedding [5]) or local (LLE, Laplacian Eigenmaps [6]) properties of the manifold to map each high-dimensional point  $\mathbf{x}_i \in \mathbb{R}^D$  to its corresponding low-dimensional embedding,  $\mathbf{y}_i \in \mathbb{R}^d$ . They are used as a generic non-linear, non-parametric technique to approximate probability distributions in high-dimensional spaces.

The Isomap algorithm, being a global NLDR technique should ideally provide a more faithful representation and preserve geometry irrespective of scale i.e. map data samples which are close in the manifold to points which are close in the low-dimensional embedding and similarly for distant samples. However, it struggles when dealing with multi-modal and non-uniform distributions.

Most existing NLDR techniques, perform a similar series of data transformations as shown in Figure 4. First, a neighborhood graph is constructed, where each node of the graph is connected to its  $k$  nearest neighbors. This involves

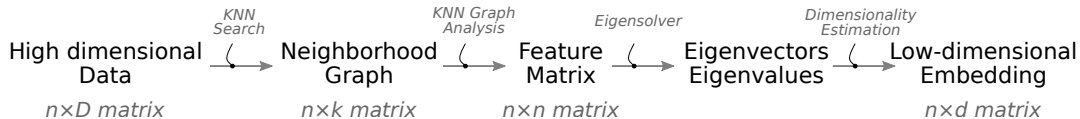


Figure 4: General non-linear spectral dimensionality reduction workflow.

computing  $\mathcal{O}(n^2)$  pairwise distance values. Next, a feature matrix is computed from this neighborhood graph, which encodes properties of the data that should be preserved during dimensionality reduction. For example, in the Isomap formulation, the feature matrix stores shortest paths between each pair of points in the neighborhood graph, which is an approximation of the actual geodesic distance between the points. The cost to compute the feature matrix generally varies in the range  $\mathcal{O}(n)$  and  $\mathcal{O}(n^3)$ . To obtain the low representation of the input data, the feature matrix is factorized and the first  $d$  eigen vectors/values form the output  $Y$ . This step has a  $\mathcal{O}(n^3)$  cost.

When used on data streams, NLDR methods typically have to recompute the entire manifold for every new streaming data point, which is computationally expensive. In such scenarios, there is the need for incremental techniques (Out-of-Sample technique [10], S-Isomap [7]), which can process the new streaming points “cheaply”, compared to the traditional batch techniques without affecting the quality of the embedding significantly.

### C. Handling Multiple Manifolds

In the ideal scenario, when manifolds are densely sampled and sufficiently separated, existing NLDR methods can be extended to perform clustering before applying the dimensionality reduction step [13], [18], by choosing an appropriate local neighborhood size so as not to include points from other manifolds and still be able to capture the local geometry of the manifold. However, if the manifolds are close or intersecting (See Figures. 2, 6), such methods typically fail.

## III. RELATED WORK

Most existing NLDR techniques can only deal with a single manifold which leads to them discovering error-prone low dimensional embeddings given inter-manifold distances are usually much larger than the intra-manifold distances.

Wu *et al.* [20] was among the earliest attempts to work with multiple manifolds via NLDR techniques. Since then, other sophisticated approaches [11], [13], [15], [17], [24] have emerged, apart from techniques in the area of manifold alignment [9], [22] and manifold clustering [2], [13]. Some assumed a supervised setting [11], [17], and learn multiple sub-manifolds corresponding to different given classes in a dataset. The MMDA method [15] is based on Locality Preserving Projections. Similarly, the SMCE algorithm [24] makes assumptions about sparsity and linearity of the embedding.

There have been earlier attempts to cluster sub-manifolds [2], [13], which are primarily based on the idea of forming a graph with edges only between a node and its nearest neighbors. However, these methods cannot deal with intersecting manifolds when it is possible for the local neighborhood of a point to have nearest neighbors from different sub-manifolds. Manifold alignment approaches [9], [22] typically align manifolds using a set of correspondences between data points. Whereas [9] uses Procrustes Analysis, [22] tries to solve a constrained embedding problem, where the embeddings of the corresponding points from different sets are constrained to be identical.

In a batch setting, the M-Isomap [13] algorithm comes close to our proposed work. The algorithm attempts to work with multiple manifolds embedded in a high-dimensional space. First, it performs clustering to identify the individual sub-manifolds via a nearest neighbor approach and subsequently runs Isomap on each of these sub-manifolds. Finally, it stitches the sub-manifolds together via a set of support points, by finding an optimal transformation between the embeddings uncovered by Multidimensional Scaling(MDS) [21] and Isomap, respectively. However, the nearest neighborhood clustering strategy employed can misrepresent individual sub-manifolds if they are intersecting and/or very close to each other by grouping them together (See Figure 2).

## IV. METHODOLOGY

There are two key challenges that a streaming manifold learning algorithm has to address: 1) handle streaming data in a scalable manner, and, 2) learn in presence of multiple, possibly intersecting, manifolds.

The proposed S-Isomap++ algorithm follows the two-phase strategy proposed in our earlier work [7], where we first learn exact manifolds from an initial batch, and then employ a computationally inexpensive mapping method to process the remainder of the stream. An error metric is used to decide on when to *switch* from expensive and exact learning to inexpensive and approximate mapping [7]. To address the second challenge, we first cluster the batch data using a *tangent-based manifold clustering* approach and then apply exact Isomap on each cluster. The resulting low-dimensional data for the clusters is then *stitched* together to obtain the data reduced to a low (and closer to true) dimensionality.

The overall S-Isomap++ algorithm is outlined in Algorithm 1. The algorithm takes a batch data set,  $\mathcal{B}$  and the

streaming data,  $\mathcal{S}$  as inputs such that,  $\mathcal{B}, \mathcal{S} \in \mathbb{R}^D$ . Note that in practical applications, one might not have data split into batch and streaming parts. In that scenario, one may track the quality of the output of the batch phase using suitable error metrics [7], and switch when a reliable solution for the batch is obtained. For simplicity, we will assume that the optimal batch size has been pre-determined. The processing is split into two phases: a batch learning phase (Lines 1–12) and a streaming phase (Lines 13–20). The batch learning phase consists of three steps:

- Step 1: Cluster samples in  $\mathcal{B}$  into  $p$  clusters (Line 1).
- Step 2: Learn  $p$  individual manifolds corresponding to each cluster, and map samples within each cluster to a low-dimensional representation<sup>1</sup> (Lines 6–7).
- Step 3: Map reduced samples from individual manifolds into a global reduced space (Lines 8–12).

In the streaming phase, each sample in the stream set  $\mathcal{S}$  is mapped onto each of the  $p$  manifolds by using an inexpensive mapping procedure (Lines 14–17). The nearest manifold is identified by comparing each reduced representation of the sample to the “center” of each manifold (Line 18), and choosing the corresponding reduced representation for the stream sample (Line 19).

The individual components of the proposed S-Isomap++ algorithm are discussed in the subsequent subsections.

#### A. Clustering Multiple Intersecting Manifolds

The objective of the first step in Algorithm 1 is to separate the batch samples into clusters, such that each cluster corresponds to one of the multiple manifolds present in the data. Note that, in this paper, we do not assume that the number of manifolds ( $p$ ) is specified; it is automatically inferred by the clustering algorithm. In cases of uneven/low density sampling, the clustering strategy discussed might possibly generate many small clusters. In such cases, one can try to merge clusters, based on their affinity/closeness to allow the number of clusters to remain within required limits. Given that the batch samples lie on low-dimensional and potentially intersecting manifolds, it is evident that the standard clustering methods, such as K-Means [27], that operate on the observed data in  $\mathbb{R}^D$ , will fail in correctly identifying the clusters.

To handle this challenge, we propose a novel clustering algorithm that is based on the notion of smoothness of manifold surfaces. Consider a single batch data sample,  $\mathbf{x}_i \in \mathbb{R}^D$ . Let  $\mathcal{N}(\mathbf{x}_i)$  be the set of  $k$  nearest neighbor samples of  $\mathbf{x}_i$  in the batch  $\mathcal{B}$ . Let  $\mathcal{T}_i$  denote a  $d'$  dimensional *tangent plane* represented using  $d'$  basis vectors,  $\mathbf{t}_{i1}, \mathbf{t}_{i2}, \dots, \mathbf{t}_{id'}$ , i.e.,  $\mathcal{T}_i = \text{span}(\mathbf{t}_{i1}, \mathbf{t}_{i2}, \dots, \mathbf{t}_{id'})$ . Here,  $d'$  denotes the intrinsic

<sup>1</sup>The true dimensionality of the manifolds corresponding to the clusters can vary. We assume that the true dimensionality for each cluster has been determined using techniques such as studying the spectral properties of the geodesic distance matrix computed as part of Isomap learning (See Figure 5).

---

#### Algorithm 1 S-Isomap++

---

**Input:** Batch dataset:  $\mathcal{B}$ , Streaming dataset:  $\mathcal{S}$ ; Parameters:  $\epsilon, k, l, \lambda$

**Output:**  $\mathcal{Y}_{\mathcal{S}}$ : low-dimensional representation for  $\mathcal{S}$

1:  $\mathcal{C}_{i=1,2,\dots,p} \leftarrow \text{FIND\_CLUSTERS}(\mathcal{B}, \epsilon)$   
 2:  $\xi_s \leftarrow \emptyset$

3: **for**  $1 \leq i \leq p$  **do**  
 4:    $\mathcal{LDE}_i \leftarrow \text{ISOMAP}(\mathcal{C}_i)$   
 5: **end for**

6:  $\xi_s \leftarrow \bigcup_{i=1}^p \bigcup_{j=i+1}^p \text{NN}(\mathcal{C}_i, \mathcal{C}_j, k) \cup \text{FN}(\mathcal{C}_i, \mathcal{C}_j, l)$

7:  $\mathcal{GE}_s \leftarrow \text{MDS}(\xi_s)$

8: **for**  $1 \leq j \leq p$  **do**

9:    $\mathcal{I} \leftarrow \xi_s \cap \mathcal{C}_j$

10:    $\mathcal{A} \leftarrow \begin{bmatrix} \mathcal{LDE}_j^{\mathcal{I}} \\ e^{\mathcal{I}} \end{bmatrix}$

11:    $\mathcal{R}_i, t_i \leftarrow \mathcal{GE}_{\mathcal{I},s} \times \mathcal{A}^T (\mathcal{A}\mathcal{A}^T + \lambda \mathbf{I})^{-1}$

12: **end for**

13: **for**  $s \in \mathcal{S}$  **do**

14:   **for**  $1 \leq i \leq p$  **do**

15:      $y_s^i \leftarrow \text{S-ISOMAP}(s, \mathcal{C}_i)$

16:      $\mathcal{GE}_s^i \leftarrow \mathcal{R}_i y_s^i + t_i$

17:   **end for**

18:    $\text{index} \leftarrow \text{argmin}_i |y_s^i - \mu(\mathcal{C}_i, \mathcal{R}_i, t_i)|$

19:    $\mathcal{Y}_{\mathcal{S}} \leftarrow \mathcal{Y}_{\mathcal{S}} \cup y_s^{\text{index}}$

20: **end for**

21: **return**  $\mathcal{Y}_{\mathcal{S}}$

---

dimensionality of the tangent plane. We assume that each  $\mathbf{x}_i$  belongs to a single manifold  $\mathcal{M}_j, \exists j \in \{1, 2, \dots, p\}$ .

The proposed clustering algorithm (Algorithm 2) is based on the following intuition: For a given sample,  $\mathbf{x}_i$ , and its neighbor  $\mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i)$ :

$$\text{If } \mathcal{M}_i = \mathcal{M}_j \Rightarrow \phi(\mathcal{T}_i, \mathcal{T}_j) \geq \epsilon \quad (1)$$

$\phi(\mathcal{T}_i, \mathcal{T}_j) = \cos \theta$ , where  $\theta$  is the angle between the two tangent planes<sup>2</sup>,  $\mathcal{T}_i$  and  $\mathcal{T}_j$ . Similarly,

$$\text{If } \mathcal{M}_i \neq \mathcal{M}_j \Rightarrow \phi(\mathcal{T}_i, \mathcal{T}_j) < \epsilon \quad (2)$$

In other words, within a tight neighborhood, a given data sample and its neighbors are expected to lie on tangent planes that are approximately similar in orientation, and, thus, the cosine of the angle between the two planes will be closer to 1 ( $\cos \theta \approx 1$ ). However, if a sample’s neighborhood

<sup>2</sup> $\mathcal{T}_i$  and  $\mathcal{T}_j$  are the tangent planes for the samples  $\mathbf{x}_i$  and  $\mathbf{x}_j$ .

---

**Algorithm 2** Tangent Manifold Clustering

---

```
1: function FIND_CLUSTERS( $\mathcal{B}$ ,  $\epsilon$ )
2:    $\mathcal{S}_{i=1,2\dots n} \leftarrow \text{MSVD}(\mathcal{B})$ 
3:    $labels \leftarrow \mathbf{0}_{n \times 1}$ 
4:    $idx \leftarrow \mathbf{1}$ 
5:   while  $labels_{i=1,2\dots n} \neq \mathbf{0}$  do
6:      $\mathcal{C}_{idx}, labels \leftarrow \text{CLUSTER}(\mathcal{B}, \mathcal{S}, labels, idx, \epsilon)$ 
7:      $idx \leftarrow idx + \mathbf{1}$ 
8:   end while
9:   return  $\mathcal{C}_{i=1,2\dots p}$ 
10: end function
```

---

---

**Algorithm 3** Incremental Partitioning Strategy

---

```
1: function CLUSTER( $\mathcal{B}, \mathcal{S}, labels, index, \epsilon$ )
2:    $\mathcal{C}_{index} \leftarrow \emptyset, \mathcal{C}_{old} \leftarrow \emptyset$ 
3:    $\mathcal{I} \leftarrow \{i | labels_i = \mathbf{0}\}, idx \sim \text{RANDOM}(\mathcal{I})$ 
4:    $\mathcal{C}_{index} \leftarrow \mathcal{C}_{index} \cup \mathcal{B}_{idx}$ 
5:    $\mathcal{C}_{old} \leftarrow \mathcal{C}_{old} \cup \mathcal{B}_{idx}, labels_{idx} \leftarrow index$ 
6:    $count_{new} \leftarrow \mathbf{1}, mode = \text{'L1'}$ 
7:   while  $count_{new} > \mathbf{0}$  do
8:      $count_{new} \leftarrow \mathbf{0}, \mathcal{C}_{new} \leftarrow \emptyset$ 
9:     for  $\forall i \in \mathcal{C}_{old}$  do
10:       $\mathcal{I}_{knn} \leftarrow \text{KNN}(\mathcal{B}, i)$ 
11:      for  $\forall j \in \mathcal{I}_{knn}$  do
12:        if  $labels_j = \mathbf{0}$  then
13:           $sim_{i,j} \leftarrow \text{SIM}(\mathcal{S}_i, \mathcal{S}_j, mode)$ 
14:          if  $sim_{i,j} \geq \epsilon$  then
15:             $\mathcal{C}_{new} \leftarrow \mathcal{C}_{new} \cup \mathcal{B}_j$ 
16:             $labels_j \leftarrow index$ 
17:             $count_{new} \leftarrow count_{new} + \mathbf{1}$ 
18:          end if
19:        end if
20:      end for
21:    end for
22:     $\mathcal{C}_{index} \leftarrow \mathcal{C}_{index} \cup \mathcal{C}_{new}, \mathcal{C}_{old} \leftarrow \mathcal{C}_{new}$ 
23:  end while
24:  return  $\mathcal{C}_{index}, labels$ 
25: end function
```

---

contains samples that lie on other intersecting manifolds, their tangent planes should be significantly different, and  $\cos \theta \ll 1$ .

1) *Learning a Tangent Plane for a Given Sample:* We use *Multiscale Singular Value Decomposition* (or MSVD [12]) on the local neighborhood of  $\mathbf{x}_i$ , to determine basis vectors,  $\mathbf{t}_{i1}, \mathbf{t}_{i2}, \dots, \mathbf{t}_{id'}$ , which define the tangent plane,  $\mathcal{T}_i$ . Use

of SVD allows us to follow the intuitions expressed in (1) and (2), since it explores directions in which the spread of points is maximal. In the presence of multiple intersecting manifolds, these directions get mangled up, whereas non-intersecting regions have better agreement with regards to principal directions.

MSVD allows us to deal with the problem of estimating the intrinsic dimension of noisy, high-dimensional point clouds. For the linear case, SVD analysis can estimate the intrinsic dimensionality of  $\mathcal{M}$  correctly, with high probability. However, when  $\mathcal{M}$  is a nonlinear manifold, curvature forces the dimensionality of the best-approximating hyperplane to be much higher, which hinders attempts to uncover the true intrinsic dimensionality of  $\mathcal{M}$ .

MSVD estimates the intrinsic dimensionality of  $\mathcal{M}$  by computing the singular values,  $\sigma_i^{z,r}$  for all  $\forall z \in \mathcal{M}$  at different scales  $r > 0$  and  $i \in \{1, 2, \dots, D\}$ . Small values of  $r$  lead to not enough samples in  $\mathcal{B}(z, r)$ , while large values of  $r$  lead to curvature making the SVD computation over estimate the intrinsic dimensionality. At the right scale (value of  $r$ ), the true  $\sigma_i^{z,r}$ 's separate from the noise  $\sigma_i^{z,r}$ 's due to their different rates of growth and the true dimensionality of  $\mathcal{M}$  is revealed. Figure 5 demonstrates how  $\sigma_i^{z,r}$  behave over different scales when MSVD is done a noisy  $\mathbb{R}^5$  sphere embedded in  $\mathbb{R}^{100}$  ambient space. Notice how the noise dimensions decay out, leaving only the primary components at the appropriate scale.

2) *Computing Angle Between Two Tangent Planes:* We explore several strategies of computing the similarity between a pair of tangent planes,  $\mathcal{T}_i$  and  $\mathcal{T}_j$ . As mentioned earlier, this is equivalent to computing the cosine of the angle between the two planes. We consider one approach, as proposed by Gunawan *et al.* [3]. Let  $\mathcal{T}_i$  and  $\mathcal{T}_j$  be orthonormal subspaces<sup>3</sup>. If  $\theta$  is the angle between  $\mathcal{T}_i$  and  $\mathcal{T}_j$ , then:

$$\phi(\mathcal{T}_i, \mathcal{T}_j) = \cos \theta = \sqrt{\det(\mathcal{N}\mathcal{N}^\top)} \quad (3)$$

where  $\mathcal{N}$  is a matrix, such that  $\mathcal{N}[u][v] = \langle \mathbf{t}_{iu}, \mathbf{t}_{jv} \rangle$ , where  $\mathbf{t}_{iu}$  is the  $u^{th}$  basis vector for  $\mathcal{T}_i$  and  $\mathbf{t}_{jv}$  is the  $v^{th}$  basis vector for  $\mathcal{T}_j$ . Additionally, when the dimensionality of  $\mathcal{T}_i$  and  $\mathcal{T}_j$  is same, the expression simplifies to:

$$\phi(\mathcal{T}_i, \mathcal{T}_j) = \cos \theta = |\det(\mathcal{N})| \quad (4)$$

Alternately, one can use the following procedure. Without loss of generality, let us assume that  $\mathbf{t}_{i1}, \mathbf{t}_{i2}, \dots, \mathbf{t}_{ik}$  are the singular vectors for the plane  $\mathcal{T}_i$  corresponding to the top  $k$  singular values. Similarly, let  $\mathbf{t}_{j1}, \mathbf{t}_{j2}, \dots, \mathbf{t}_{jk}$  be the top- $k$  singular vectors for the plane  $\mathcal{T}_j$ . Then we can compute

<sup>3</sup>One can use QR factorization to orthonormalize any subspace, which is not already orthonormal.

$\phi(\mathcal{T}_i, \mathcal{T}_j)$  as:

$$\phi(\mathcal{T}_i, \mathcal{T}_j) = \frac{1}{k} \sum_{l=1}^k |\mathbf{t}_{il}^\top \mathbf{t}_{jl}| \quad (5)$$

We refer to the above as the  $L1$  metric. In the same way, one can define the  $L2$  metric as:

$$\phi(\mathcal{T}_i, \mathcal{T}_j) = \sqrt{\frac{1}{k} \sum_{l=1}^k (\mathbf{t}_{il}^\top \mathbf{t}_{jl})^2} \quad (6)$$

3) *Tangent Manifold Clustering Algorithm*: The proposed tangent manifold clustering strategy is outlined in Algorithm 2. Algorithm 3 is the support method to the above. The inputs to the Algorithm 2 are the batch dataset  $\mathcal{B}$  and a threshold value  $\epsilon$ .

Algorithm 2 initially calls  $\text{MSVD}(\cdot)$  (See Section IV-A1) on the input batch set,  $\mathcal{B}$ , to decide on an appropriate scale  $r$  to use and subsequently to extract the top- $k$  singular vectors  $\mathcal{S}_{i=1,2,\dots,n}$  for all  $\mathbf{x}_i \in \mathcal{B}$ , at the scale  $r$ . Initially all points are unlabeled i.e. *labels* is all zeros initially. Algorithm 2 calls  $\text{CLUSTER}(\cdot)$  repeatedly till all  $\mathbf{x}_i \in \mathcal{B}$  have labels assigned to them, which represents the different clusters,  $\mathcal{C}_i$  for  $i = 1, 2 \dots m$  where  $\bigcup_{i=1}^m \mathcal{C}_i = \mathcal{B}$ .

Algorithm 3, which contains the function  $\text{CLUSTER}(\cdot)$ <sup>4</sup>, works as follows: it picks a currently unlabeled  $\mathbf{x}_i$  at random, and assigns it to a new cluster  $\mathcal{C}_{index}$ . Subsequently, it looks at the unassigned nearest neighbors of  $\mathbf{x}_i$  i.e.  $\mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i)$  and checks to see how close their tangent planes are. If they are similar enough i.e. the similarity score  $\phi(\mathcal{T}_i, \mathcal{T}_j) \geq \epsilon$ , then the unassigned nearest neighbor is assigned to  $\mathcal{C}_{index}$ . The algorithm proceeds similarly in a breadth-first manner till no new points remain to be tested.

It internally calls Algorithm  $\text{SIM}(\cdot)$  to measure similarity, using one of the three strategies, discussed in Section IV-A2 (See (4), (5), and (6)).

### B. Processing multiple manifolds

The S-Isomap++ algorithm independently learns the manifolds for each cluster (Lines 3–5). However, since these manifolds are not necessarily aligned with respect to each other, an additional step is needed to represent the reduced samples from each cluster into a common space. We refer to this process as *stitching*, and is essential to recreate the final reduced representation. This step, similar to the approach in M-Isomap, maintains the information of the global location of different manifolds using a set of support points which form the skeleton on which it can later place the different manifolds. This support set is formed using the  $k$  nearest neighbor pairs as well as the  $l$  farthest neighbor pairs between every pair of manifolds present i.e.  $\forall \{\mathcal{C}_i, \mathcal{C}_j\}_{j \neq i}$ , let

<sup>4</sup>We use ‘L1’ as the mode by default (Line 6) since it provides the best accuracy.

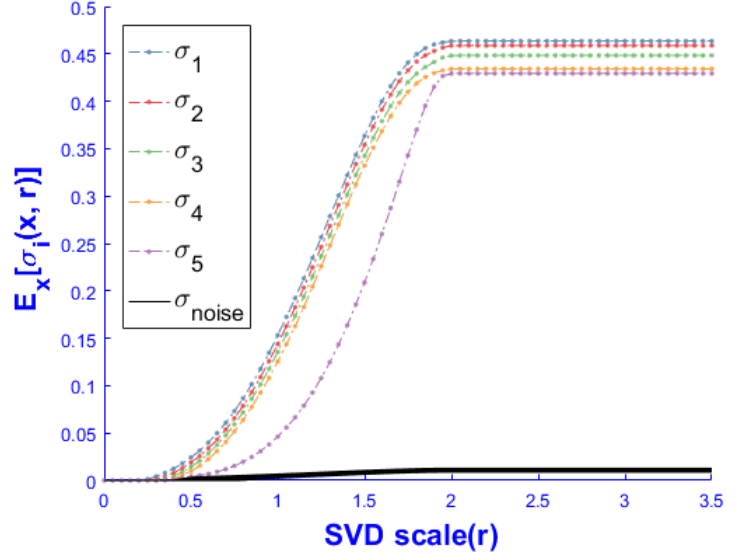


Figure 5: Multiscale SVD on a noisy  $\mathbb{R}^5$  sphere embedded in  $\mathbb{R}^{100}$  ambient space.

---

### Algorithm 4 Similarity between tangent planes between points

---

```

1: function SIM( $\mathcal{S}_i, \mathcal{S}_j, mode$ )
2:    $\eta_{i=1,2,\dots,k} \leftarrow \text{extract}(\mathcal{S}_i)$ 
3:    $\kappa_{i=1,2,\dots,k} \leftarrow \text{extract}(\mathcal{S}_j)$ 
4:
5:   if  $mode = 'L1'$  then
6:      $score \leftarrow \frac{1}{k} \sum_{i=1}^k |\eta_i^T \kappa_i|$ 
7:
8:   else if  $mode = 'L2'$  then
9:
10:     $score \leftarrow \sqrt{\sum_{i=1}^k \frac{1}{k} (\eta_i^T \kappa_i)^2}$ 
11:
12:   else if  $mode = 'HG'$  then
13:      $\mathcal{M}_\eta \leftarrow \text{matrix}(\eta_{i=1,2,\dots,k})$ 
14:      $\mathcal{M}_\kappa \leftarrow \text{matrix}(\kappa_{i=1,2,\dots,k})$ 
15:
16:      $\mathcal{M} \leftarrow \mathcal{M}_\eta^T \mathcal{M}_\kappa$ 
17:      $score \leftarrow |\det(\mathcal{M})|$ 
18:   end if
19:
20:   return  $score$ 
21: end function

```

---

$\mathcal{X}_{i,j} \in \mathbb{R}^{|\mathcal{C}_i| \times |\mathcal{C}_j|}$  denote the  $\mathbb{R}^D$  Euclidean distance matrix between all points in clusters  $\mathcal{C}_i$  and  $\mathcal{C}_j$ , then support set  $\xi_s$  contains the co-ordinates (index sets  $\mathcal{I}_i$  and  $\mathcal{I}_j$  from  $\mathcal{C}_i$  and  $\mathcal{C}_j$  respectively) of both the smallest  $k$  values as well as the largest  $l$  values in  $\mathcal{X}_{i,j}$ . The former are calculated by method  $\text{NN}(\cdot)$  and the latter  $\text{FN}(\cdot)$  (Line 6). Subsequently, a global reduced space embedding  $\mathcal{G}\mathcal{E}_s$  for this support set is calculated using MDS (Line 7). After this, for each manifold  $\mathcal{M}_j, \exists j \in \{1, 2 \dots p\}$ , a least-squares problem is solved to generate the transformation components  $\mathcal{R}_i, t_i$  which can project reduced samples from each cluster into the global space (Lines 8–12).

### C. Mapping Streaming Samples

In the streaming part, each sample in the stream set  $\mathcal{S}$  is mapped onto each of the  $p$  manifolds in parallel, using the inexpensive S-ISOMAP( $\cdot$ ) algorithm proposed in our earlier work [7] (Line 15) and subsequently mapped to the global space using  $\{\mathcal{R}_i, t_i\} \exists i \in \{1, 2 \dots p\}$  (Line 16). The nearest manifold is identified by comparing each reduced representation of the sample to the mean  $\mu(\cdot)$  of each manifold (Line 18), and choosing the corresponding reduced representation for the stream sample (Line 19).

## V. RESULTS AND ANALYSIS

### A. Experimental Setup

We present several experiments here on a variety of data sets to illustrate the behavior of different approaches proposed in the Section IV.

We use four different datasets in our experiments. Given swiss roll datasets are typically used for evaluating manifold learning algorithms, we use the Euler Isometric Swiss Roll dataset, proposed by Schoeneman *et al.* [7], wherein a  $\mathbb{R}^2$  data set having  $n = 3000$  points, chosen at random, are embedded into  $\mathbb{R}^3$  using a non-linear function  $\psi(\cdot)$ . We use this in conjunction with a  $\mathbb{R}^3$ -dimensional hyperplane passing through it as shown in Figure. 2b having  $n = 1500$  points, chosen at random. We know the ground truth for both parts (See Figure. 2a). We use this to evaluate the S-Isomap++ algorithm as shown in Figure. 2. We also use an extension of this, wherein two  $\mathbb{R}^3$ -dimensional hyperplanes pass through the Isometric Swiss Roll, wherein the points are chosen in random and each hyperplane has  $n = 3000$  points, as shown in Figure. 6.

Apart from this, we use different artificial datasets consisting of intersecting manifolds i.e. two intersecting  $\mathbb{R}^3$ -dimensional unit hyperspheres, having  $n = 1000$  points each and a  $\mathbb{R}^3$ -dimensional plane intersecting a  $\mathbb{R}^3$ -dimensional hypersphere, again having  $n = 1000$  points each, as shown in Figure. 7. We use these datasets to test our tangent manifold approach more rigorously. We also use patches on the Euler Isometric Swiss Roll dataset (Figure. 1) which are Gaussian in nature, to study the effect of the different

parameters, apart from evaluating our algorithm, as well as the MNIST digits dataset.

Our evaluation metrics for the experiments primarily focus on 1) ability on our tangent manifold clustering strategy to be able to cluster points from multiple intersecting/non-intersecting manifolds correctly, 2) test the quality of the embedding uncovered by our algorithm, for the streaming dataset  $\mathcal{S}$ , with regards to agreeability with ground truth via an appropriate distance metric, as well as, tightness of clustering and last but not the least, 3) scalability of our algorithm over different sizes of both batch and streaming datasets  $\mathcal{B}$  and  $\mathcal{S}$  respectively.

### B. Results on Artificial Datasets

1) *Gaussian patches on Isometric Swiss Roll*: Figures. 1c, 1d, 1e demonstrate the results with this dataset for Isomap, M-Isomap and our approach respectively. Both the M-Isomap and S-Isomap++ algorithms can deal with individual manifolds better than Isomap, which severely deforms the individual clusters. It should also be noted that whereas both the M-Isomap and S-Isomap++ algorithms required small values of  $k$  i.e.  $k = 8$  to operate, Isomap needed values of  $k \geq 500$  to even work. As a consequence, idiosyncrasies i.e. short-circuiting become a factor to distort the uncovered embedding. M-Isomap has scaling issues and can only seem to attempt to position the individual manifolds in the global ambient space correctly, without being able to recreate the spread, which defined the individual manifolds. We think that M-Isomap internally normalizes individual manifolds which results in this behavior. Our approach, S-Isomap++ is the most robust in its recreation of the ground truth.

2) *Intersecting Swiss-roll with  $\mathbb{R}^3$ -dimensional plane*: Figure. 2 demonstrates our experiments with this dataset. We evaluate different algorithms to see how well they recreate the ground truth (Figure. 2a). Both Isomap and M-Isomap produce the same output, given M-Isomap employs a nearest-neighbor based clustering strategy to disambiguate between manifolds, and hence is unable to handle intersecting manifolds, which results in highly distorted recreations of the ground truth. As before, S-Isomap++ produces the most robust recreation of the ground truth. Figure 6, demonstrates how well S-Isomap++ recreates the original manifolds, in case the batch  $\mathcal{B}$  is clustered correctly. M-Isomap/Isomap are unable to recreate the ground truth and severely contort the ground truth.

3) *Tangent Manifold Clustering*: Here we present clustering results for intersecting manifolds. (See Figures. 2, 7 for the different datasets). Table I below demonstrates accuracy values<sup>5</sup> with which the L-1, L-2 metric schemes proposed in this work, along with the technique proposed by Gunawan *et*

<sup>5</sup>Gunawan’s approach was unable to distinguish between the intersecting manifolds scenarios and always clustered them as one and hence its accuracy was 0.5 in all cases.



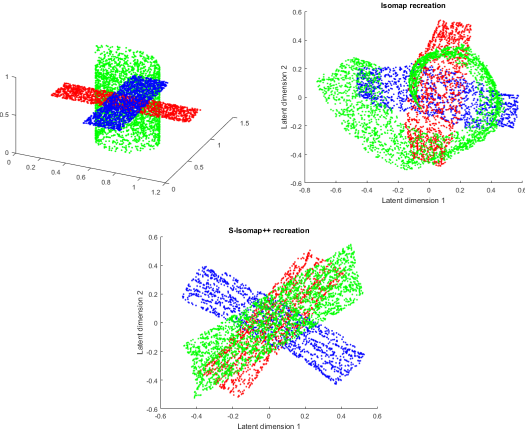


Figure 6: Top Left: Actual manifolds in  $\mathbb{R}^3$  space, clustered to demonstrate individual manifolds, Top Right: Recreation by Isomap/M-Isomap, Bottom Row: Recreation by our approach, S-Isomap++.

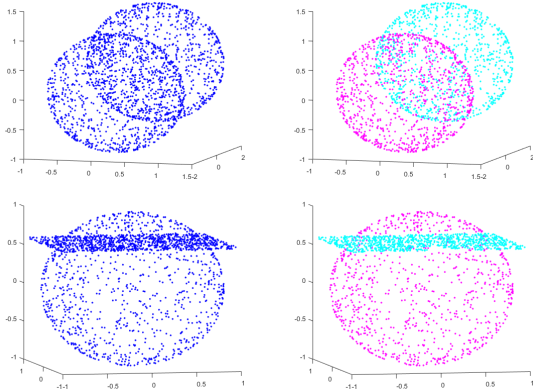


Figure 7: Left: Original datasets unclustered, Right: Clustered using the proposed tangent clustering method.

*al.* [3] clustered the different intersecting manifolds. The L-2 metric performed much better than Gunawan’s approach, however the L-1 metric performed the best. The accuracy values are also indicative of the level of difficulty associated with clustering the different scenarios correctly.

Method	L-1	L-2	Gunawan
Sphere-Sphere	<b>0.825</b>	0.619	0.5
Sphere-Plane	<b>0.759</b>	0.602	0.5
Swiss Roll-Plane	<b>0.838</b>	0.621	0.5

Table I: Accuracy scores for the different tangent manifold clustering approaches.

4) *Effect of different parameters:* Here we present results of the effect of changing the different parameters of the S-Isomap++ algorithm, while keeping all other parameters

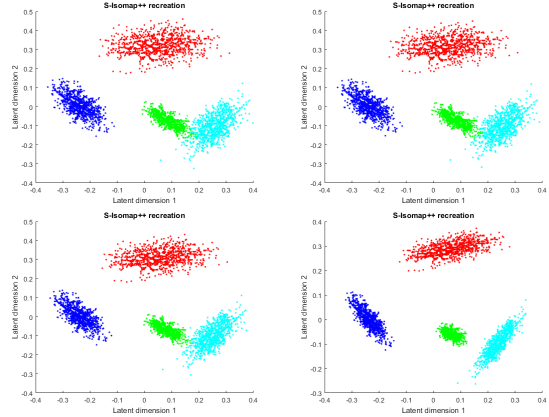


Figure 8: Effect of changing  $\lambda$ . Top Left:  $\lambda = 0.01$ , Top Right:  $\lambda = 0.02$ , Bottom Left:  $\lambda = 0.04$ , Bottom Right:  $\lambda = 0.16$

*fixed.* Figures 8, 9, 10, demonstrates the effect of parameter  $\lambda$ ,  $k$  and  $l$  on the embeddings uncovered by the S-Isomap++ algorithm. Larger values of  $k$  seems to make the manifolds more uniform or rounded. Larger values of parameter  $l$  seem to stretch the manifolds. Parameter  $\lambda$  seem to separate the manifolds apart when it has larger values. This is really interesting since it means we can use it to visualize manifolds better on account of separability.

Figure 11 demonstrates the scalability of our algorithm with regards to streaming data  $\mathcal{S}$ . Batch  $\mathcal{B}$  having size  $n = 2000$  was used for this experiment. The timing results are in log scale and clearly demonstrate the efficiency gained. M-Isomap has the same result as Isomap since it cannot distinguish between intersecting manifolds and treats them as one. While the run-time for Isomap/M-Isomap increases rapidly with increasing stream size, the run time for S-Isomap++ does not grow much at all, making it highly conducive to large stream processing.

### C. Results on MNIST Dataset

Table II below shows results for different digits of the MNIST dataset. Using a batch dataset  $\mathcal{B}$  of size  $n = 2000$ , a streaming dataset  $\mathcal{S}$  of size  $m = 4000$  was recreated in 3D by the S-Isomap++ algorithm, for each of the digits. Subsequently the 3D recreation was compared to the 3D ground truth obtained by running Isomap on all digits, using the Procrustes Error metric to measure the quality of the recreation.

The Procrustes Error metric determines an optimal alignment between two matrices  $\mathcal{X}$  and  $\mathcal{Y}$  and returns a goodness-of-fit criterion, based on sum of squared errors. As the results below demonstrate, the recreation error is pretty low, even after embedding in the common global space. This shows the efficacy of the S-Isomap++ algorithm.

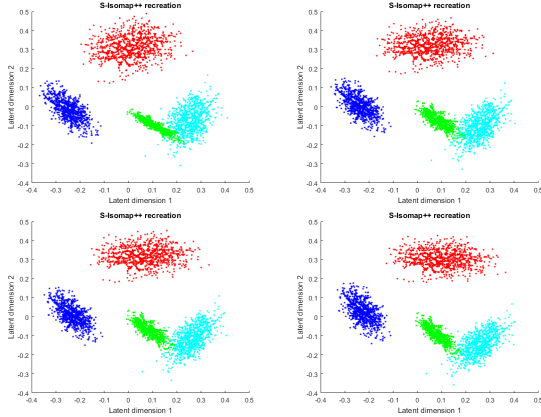


Figure 9: Effect of changing  $k$ . Top Left:  $k = 8$ , Top Right:  $k = 16$ , Bottom Left:  $k = 24$ , Bottom Right:  $k = 32$

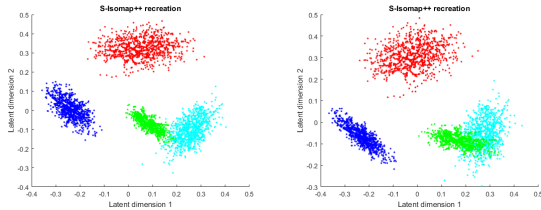


Figure 10: Effect of changing  $l$ . Left:  $l = 1$ , Right:  $l = 4$

digit '0'	<b>0.0296</b>	digit '3'	<b>0.0364</b>	digit '6'	<b>0.0476</b>
digit '1'	<b>0.0806</b>	digit '4'	<b>0.0586</b>	digit '8'	<b>0.0712</b>
digit '2'	<b>0.0499</b>	digit '5'	<b>0.0449</b>	digit '9'	<b>0.0498</b>

Table II: Procrustes error values for different digits of the MNIST dataset, computed by comparing the original with 3D recreation via S-Isomap++.

## VI. CONCLUSION

The proposed S-Isomap++ algorithm allows for scalable non-linear dimensionality reduction of streaming high-dimensional data. By allowing for the samples to belong to multiple manifolds, or sampled non-uniformly from a single manifold, we have developed an algorithm that can be applied to a wide variety of practical settings. Moreover, the two-phase strategy for streaming Isomap, first proposed in [7], and adapted here for multiple manifold learning, allows us to scale a computationally intensive algorithm (Isomap) to arbitrarily large streams.

The ability to cluster data lying on multiple intersecting manifolds is a key innovation, proposed as the Tangent Manifold Clustering algorithm, allows us to automatically identify the number of underlying manifolds. One limitation of the method, however, is that it assumes that all manifolds are represented in the batch data set, which means that a novel manifold behavior that might appear subsequently in the stream, will not be learned. This issue will be studied in

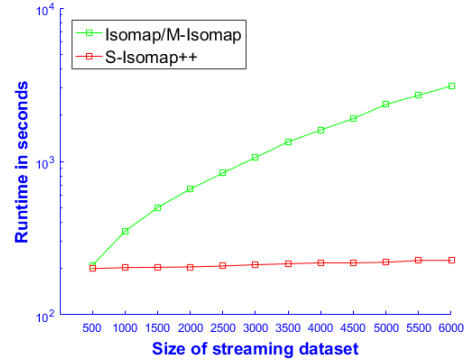


Figure 11: The results are in log scale and demonstrate the scalability of our proposed latent algorithm.

future research.

## VII. ACKNOWLEDGMENT

This material is based in part upon work supported by the National Science Foundation under award numbers CNS - 1409551 and IIS - 1651475. Computing support was provided by the Center for Computational Research.

## REFERENCES

- [1] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [2] Richard Souvenir and Robert Pless. Manifold clustering. In *Tenth IEEE ICCV, 2005*, volume 1, pages 648–653. IEEE, 2005.
- [3] Hendra Gunawan, Oki Neswan, and Wono Setya-Budhi. A formula for angles between subspaces of inner product spaces. *Contributions to Algebra and Geometry*, 46(2):311–320, 2005.
- [4] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
- [5] Kilian Q Weinberger, Benjamin Packer, and Lawrence K Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *AISTATS*, 2005.
- [6] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in NIPS, 2002*, pages 585–591, 2002.
- [7] Frank Schoeneman, Suchismit Mahapatra, Varun Chandola, Nils Napp, and Jaroslav Zola. Error metrics for learning reliable manifolds from streaming data. In *Proceedings of 2017 SDM*, pages 750–758. SIAM, 2017.
- [8] Salah Rifai, Yann N Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. In *Advances in NIPS*, pages 2294–2302, 2011.
- [9] Chang Wang and Sridhar Mahadevan. Manifold alignment using procrustes analysis. In *Proceedings of the 25th ICML*, pages 1120–1127. ACM, 2008.
- [10] Martin HC Law and Anil K Jain. Incremental nonlinear dimensionality reduction by manifold learning. *IEEE transactions on PAMI*, 28(3):377–391, 2006.

- [11] R Hettiarachchi and James F Peters. Multi-manifold lle learning in pattern recognition. *Pattern Recognition*, 48(9):2947–2960, 2015.
- [12] Anna V Little, Jason Lee, Yoon-Mo Jung, and Mauro Maggioni. Estimation of intrinsic dimensionality of samples from noisy low-dimensional manifolds in high dimensions with multiscale svd. In *IEEE/SP 15th Workshop on SSP'09*, pages 85–88. IEEE, 2009.
- [13] Mingyu Fan, Hong Qiao, Bo Zhang, and Xiaoqin Zhang. Isometric multi-manifold learning for feature extraction. In *IEEE 12th ICDM, 2012*, pages 241–250. IEEE, 2012.
- [14] Mingyu Fan, Xiaoqin Zhang, Hong Qiao, and Bo Zhang. Efficient isometric multi-manifold learning based on the self-organizing method. *Inf. Sci.*, 345(C):325–339, 2016.
- [15] Wankou Yang, Changyin Sun, and Lei Zhang. A multi-manifold discriminant analysis method for image feature extraction. *Pattern Recognition*, 44(8):1649–1657, 2011.
- [16] Michael Spivak. A comprehensive introduction to differential geometry, vol. 1, 3rd edition, publish or perish, berkeley, 1979. *Google Scholar*, 1979.
- [17] Marwan Torki, Ahmed Elgammal, and Chan Su Lee. Learning a joint manifold representation from multiple data sets. In *20th ICPR, 2010*, pages 1068–1071. IEEE, 2010.
- [18] Marzia Polito and Pietro Perona. Grouping and dimensionality reduction by locally linear embedding. In *Advances in NIPS*, pages 1255–1262, 2002.
- [19] Alvina Goh and René Vidal. Clustering and dimensionality reduction on riemannian manifolds. In *IEEE Conference on CVPR, 2008*, pages 1–7. IEEE, 2008.
- [20] Yiming Wu and Kap Luk Chan. An extended isomap algorithm for learning multi-class manifold. In *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*, volume 6, pages 3429–3433. IEEE, 2004.
- [21] Joseph B Kruskal and Myron Wish. *Multidimensional scaling*, volume 11. Sage, 1978.
- [22] Jihun Ham, Daniel D Lee, and Lawrence K Saul. Semisupervised alignment of manifolds. In *AISTATS*, pages 120–127, 2005.
- [23] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.
- [24] Ehsan Elhamifar and René Vidal. Sparse manifold clustering and embedding. In *Advances in NIPS*, pages 55–63, 2011.
- [25] Olga Kouropteva, Oleg Okun, and Matti Pietikäinen. *Incremental Locally Linear Embedding Algorithm*, pages 521–530. Springer Berlin Heidelberg, 2005.
- [26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [27] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
- [28] Mojie Duan, Jue Fan, Minghai Li, Li Han, and Shuanghong Huo. Evaluation of dimensionality-reduction methods from peptide folding–unfolding simulations. *Journal of chemical theory and computation*, 9(5):2490–2497, 2013.
- [29] Y. Ruan, G.L. House, S. Ekanayake, U. Schutte, J.D. Bever, H. Tang, and G. Fox. Integration of clustering and multidimensional scaling to determine phylogenetic trees as spherical phylograms visualized in 3 dimensions. In *IEEE/ACM Int. Symp. Clust. Cloud Grid Comput.*, pages 720–729, 2014.
- [30] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative review. *Journal of Machine Learning Research*, 10:66–71, 2009.
- [31] Feng Chen, Pan Deng, Jiafu Wan, Daqiang Zhang, Athanasios V. Vasilakos, and Xiaohui Rong. Data mining for the internet of things: Literature review and challenges. *International Journal of DSN*, 11(8), 2015.
- [32] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on PAMI*, 24(5):603–619, 2002.