Han Z, Meng D, Xu Z *et al.* Incremental alignment manifold learning. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 26(1): 153–165 Jan. 2011. DOI 10.1007/s11390-011-1118-7

Incremental Alignment Manifold Learning

Zhi Han (韩　志), De-Yu Meng* (孟德宇), Zong-Ben Xu (徐宗本), and Nan-Nan Gu (古楠楠)

Institute for Information and System Sciences, Xi'an Jiaotong University, Xi'an 710049, China

E-mail: {dymeng, zbxu}@mail.xjtu.edu.cn; {han.zhi,nngu}@stu.xjtu.edu.cn

Received March 3, 2010; revised November 2, 2010.

Abstract A new manifold learning method, called incremental alignment method (IAM), is proposed for nonlinear dimensionality reduction of high dimensional data with intrinsic low dimensionality. The main idea is to incrementally align low-dimensional coordinates of input data patch-by-patch to iteratively generate the representation of the entire dataset. The method consists of two major steps, the incremental step and the alignment step. The incremental step incrementally searches neighborhood patch to be aligned in the next step, and the alignment step iteratively aligns the low-dimensional coordinates of the neighborhood patch searched to generate the embeddings of the entire dataset. Compared with the existing manifold learning methods, the proposed method dominates in several aspects: high efficiency, easy out-of-sample extension, well metric-preserving, and averting of the local minima issue. All these properties are supported by a series of experiments performed on the synthetic and real-life datasets. In addition, the computational complexity of the proposed method is analyzed, and its efficiency is theoretically argued and experimentally demonstrated.

Keywords alignment, incremental learning, manifold learning, nonlinear dimensionality reduction, out-of-sample issue

1 Introduction

Data coming from practical applications, such as biological sciences and multimedia information processing, are often of very high dimensionality, which generally causes unexpected difficulties to discover knowledge from them[1]. The encountered high-dimensional data are, however, very often with intrinsic low-dimensional structures. Consider, for instance, the gray-scale images of an object taken under fixed lighting conditions with a moving camera. Such an image would typically be represented by a brightness value and two camera orientation measures (e.g., up-down and left-right angles). Convenience and better results can be led in further data analysis related to clustering, visualization and searching if one can find and make full use of the intrinsic low-dimensional structure of the data. Therefore, to find the intrinsic low-dimensional structure, or equivalently, to look for the intrinsic low-dimensional representation of a dataset, has been the focus of data analysis. This is known frequently as the nonlinear dimensionality reduction (NLDR) techniques.

In recent years, an NLDR technique, called "manifold learning", has been specifically highlighted. It aims to generate the low-dimensional representation of the high-dimensional data distributed on a smooth manifold with intrinsic low dimension. The case embodies many important applications in machine learning and pattern recognition[2], image processing[3], remote sensing[4], biological data mining[5] and facial expression recognition[6]. So far a variety of manifold learning methods have been developed, such as locally linear embedding (LLE)[2], isometric feature mapping (ISOMAP)[3], Laplacian eigenmap[7], and others[8-13].

However, there still remain some problems on the current manifold learning research. There have several commonly encountered difficulties in application of the existing manifold learning methods, which include: (i) computational efficiency issue[14]: a manifold learning method always involves the solution of an eigenvalue problem scalable to the number of input data points. Hence when the data sizes are large, the computational load of manifold learning tends to be extremely heavy; (ii) applicability issue[15]: the effectiveness of the existing manifold learning methods cannot be guaranteed when the input dataset lies on some complex manifolds, such as the ones with intrinsic loopy structures or multiple clusters; (iii) out-of-sample issue[16]: the algorithms of the traditional manifold learning deal with the entire dataset simultaneously. Such a batch mode conducts the difficulty of their out-of-sample extension

Regular Paper

This work was supported by the National Basic Research 973 Program of China under Grant No. 2007CB311002 and the National Natural Science Foundation of China under Grant No. 60905003.

*Corresponding Author

when new data points become available. Besides, some other issues, like metric preserving and local minima averting problems[16], should also be under consideration. To a large extent, these difficulties have baffled the development and more substantial applications of current manifold learning.

Two approaches have been mainly adopted to alleviate the aforementioned difficulties in the latest research. The first one is the alignment approach[8,17-22]. Its idea is to perform global alignment on a mixture of local NLDR models constructed on the overlapping areas lying on the entire data manifold. The latest strategy of this approach is the locally multidimensional scaling (LMDS)[14]. LMDS attains an effective alignment learning through aligning a collection of overlapping local neighborhood patches by solving an eigenvalue problem scalable with the number of overlapping patches after getting the low dimensional embeddings of each patch with the classical MDS[23] method. This kind of approach has the following advantages: (i) the approach scales with the number of overlapping patches rather than the number of individual data points, and hence the computational cost can be evidently saved with respect to the original algorithm; (ii) it generally does not suffer from the problem of local optima contributed by its batch computation; (iii) it needs only local measures, such as inter-point distances within each neighborhood, as input, and hence avoid the negative influence of using the global measure estimation. However, just because the computations of the alignment approaches are almost in whole batch, they have the following disadvantages: (i) the out-of-sample extension of the approach is not direct; and (ii) the applicability domain of the algorithms cannot be substantially extended, such as for solving loopy situations.

The other one is the incremental approach[24-29]. Its idea is to utilize a point-by-point mode to incrementally compute the embedding coordinate of each data point in the input set. The latest useful incremental technique, called Riemannian manifold learning (RML), is suggested in [16]. The method incrementally learns the embedding coordinate of the new coming point by preserving the distances and angles between the point and its neighboring points. Actually, the disadvantages of the alignment approach are partially alleviated by the incremental approach. This is because, on one hand, the incremental approach is to iteratively compute the embedding coordinate for each new coming data point, and this naturally brings the convenience of the out-of-sample extension; on the other hand, the incremental approach caters for the dataset with more complicated structures, like the loopy manifold data[16]. Nevertheless, the incremental approach generally has low computational efficiency because it generally needs to take considerable computation in the large number of iterations, generally scalable to the number of individual data points in the entire dataset. Besides, the incremental learning computes the embedding of only one point at each iteration, and hence the low-dimensional coordinate of the new coming point might be distorted due to the shortage of the assistance of enough cooperative information. This deficiency tends to further yield negative impact on the subsequent processing of NLDR process. Such a case tends to happen in applications with some special manifold data, like the 5-cluster dataset shown in Fig.7, to which the RML incremental method cannot be effectively applied.

The aim of this paper is to propose a novel technique, combining both characteristics of the alignment and the incremental approaches, so as to alleviate the disadvantages of both methods simultaneously. Therefore, we can set up a more general manifold learning approach.

In what follows, the general idea and the implementation details of the IAM method are presented in Section 2. The computational complexity of the method is also introduced and evaluated in this section. To verify the effectiveness of the passage method, results obtained from a series of empirical studies performed on synthetic and real-world datasets are analyzed and interpreted in Section 3. The paper is then concluded with a summary and outlook for future research in Section 4.

2 Incremental Alignment Method

Given the dataset $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^{l} \subset \mathbb{R}^n$ residing on the manifold with intrinsic dimensionality d, the proposed method aims at calculating its intrinsic low-dimensional representation set $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^{l} \subset \mathbb{R}^d$ $(d < n)$. The method is called incremental alignment method because it learns data patch by patch incrementally and aligns them together to approach the final global result. Before presenting the algorithm of the proposed method, it is necessary to firstly introduce the process of the alignment and incremental processes involved in the proposed IAM algorithm as follows.

2.1 To Implement Patch-by-Patch Alignment

One of the basic assumptions of manifold learning is that a point together with its neighboring samples consists of a neighborhood patch with approximate linear figure and residing on the underlying manifold. Under such an assumption, the low-dimensional representation of each local neighborhood patch can be effectively calculated by virtue of some classical linear dimensionality reduction skills, such as the multi-dimensional scaling (MDS)[23] and the principle component analysis

(PCA)[30-31]. It has been theoretically and empirically verified that these classical techniques are of excellent performance in such linear cases. According to this assumption, it is easy to realize the alignment of the low-dimensional representations of two overlapping neighborhood patches through the following three steps.

Denote the overlapping neighborhood patches as $\boldsymbol{X}_1 = \{\boldsymbol{x}_i^1\}_{i=1}^{k_1}$ and $\boldsymbol{X}_2 = \{\boldsymbol{x}_i^2\}_{i=1}^{k_2}$ respectively, where k_1 and k_2 are the sizes of $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ respectively, and denote their intersection set as $\widetilde{\boldsymbol{X}} = \boldsymbol{X}_1 \bigcap \boldsymbol{X}_2 = \{\widetilde{\boldsymbol{x}}_i\}_{i=1}^{\widetilde{k}}$, where $\widetilde{k}$ is the size of $\widetilde{\boldsymbol{X}}$. Firstly, the d-dimensional embedding set of $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, denoted as $\boldsymbol{Y}_1 = \{\boldsymbol{y}_i^1\}_{i=1}^{k_1}$ and $\boldsymbol{Y}_2 = \{\boldsymbol{y}_i^2\}_{i=1}^{k_2}$ correspondingly, are obtained by applying MDS or PCA. The corresponding embedding subsets of $\widetilde{\boldsymbol{X}}$ in $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ are denoted as $\widetilde{\boldsymbol{Y}}_1 = \{\widetilde{\boldsymbol{y}}_i^1\}_{i=1}^{\widetilde{k}}$ and $\widetilde{\boldsymbol{Y}}_2 = \{\widetilde{\boldsymbol{y}}_i^2\}_{i=1}^{\widetilde{k}}$ respectively. The aim of the second step is to linearly transform $\widetilde{\boldsymbol{Y}}_1$ and $\widetilde{\boldsymbol{Y}}_2$ so as to let the transformed sets matched as well as possible, i.e., we want to find the linear transformation parameters $\boldsymbol{W}^*$ and $\boldsymbol{b}^*$ so that

$$\varPhi(\boldsymbol{W}, \boldsymbol{b}) = \sum_{i=1}^{\widetilde{k}} \|\widetilde{\boldsymbol{y}}_i^1 - (\boldsymbol{W}\widetilde{\boldsymbol{y}}_i^2 + \boldsymbol{b})\|^2 \qquad (1)$$

can be minimized, where $\boldsymbol{W} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{b} \in \mathbb{R}^d$. This quadratic optimization problem can be easily solved. We adopt the quasi-Newton method here due to its stable performance, well generalization capability, and efficiency. Denote the optimal value of $\varPhi(\boldsymbol{W}, \boldsymbol{b})$ as

$$E^* = \varPhi(\boldsymbol{W}^*, \boldsymbol{b}^*). \qquad (2)$$

In the third step, utilizing $\boldsymbol{W}^*$ and $\boldsymbol{b}^*$ obtained, the alignment of $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ can be easily achieved by uniting the following three sets:

$$\begin{aligned}&\{\boldsymbol{y}_i^1 | \boldsymbol{y}_i^1 \in \boldsymbol{Y}_1,\ \boldsymbol{y}_i^1 \notin \widetilde{\boldsymbol{Y}}_1\} \cup \\ &\{\boldsymbol{W}^*\boldsymbol{y}_i^2 + \boldsymbol{b}^* | \boldsymbol{y}_i^2 \in \boldsymbol{Y}_2,\ \boldsymbol{y}_i^2 \notin \widetilde{\boldsymbol{Y}}_2\} \cup \\ &\left\{\frac{\widetilde{\boldsymbol{y}}_i^1 + (\boldsymbol{W}^*\widetilde{\boldsymbol{y}}_i^2 + \boldsymbol{b}^*)}{2} | 1 \leqslant i \leqslant \widetilde{k}\right\}.\end{aligned} \qquad (3)$$

Fig.1 vividly illustrates how the above steps is processed. Note that the first and second sets in (3) corresponds to the points of $\boldsymbol{Y}_1/\widetilde{\boldsymbol{Y}}_1$ and $\boldsymbol{Y}_2/\widetilde{\boldsymbol{Y}}_2$ after alignment (depicted as squares and circles), and the third set represents the transformed points of the overlapping sets between $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ in the alignment process, and each point is actually the mean of the transformed ones corresponding to $\widetilde{\boldsymbol{Y}}_1$ and $\widetilde{\boldsymbol{Y}}_2$ respectively (demonstrated as the "$*$" points).

2.2 To Incrementally Search Patches

The figure of the manifold underlying the input data can be approximated by the overlapping local neighborhood patches superimposed on the dataset. For the given dataset $\boldsymbol{X}$, we aim at incrementally generating a patch sequence $C_1, C_2, \ldots, C_m$ which satisfies $\cup_{i=1}^m C_i = \boldsymbol{X}$ and $C_i \cap (\cup_{j<i} C_j) \neq \varnothing$, $\forall\ 2 \leqslant i \leqslant m$.

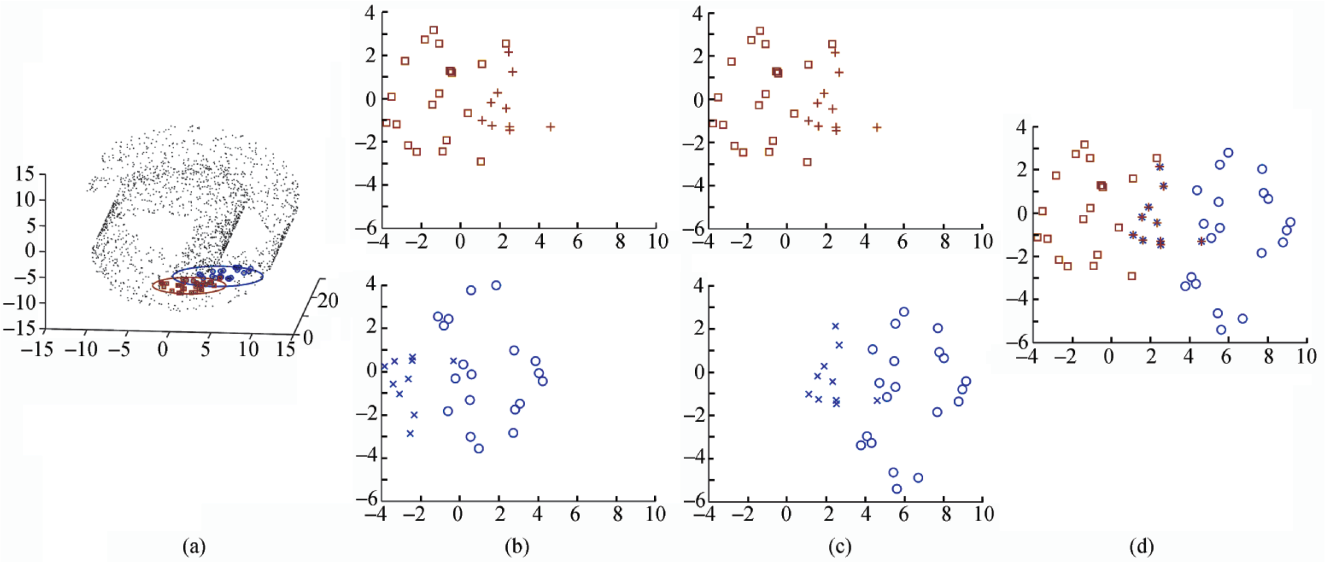


Fig.1. Graphical presentation of the alignment process of IAM. (a) Two neighborhood patches in Swiss roll dataset, denoted as squares and circles respectively, and points of each patch are enclosed together. (b) The first step of the alignment process. The squares and "+" points as well as the circles and "×" points denote the embeddings of two patches respectively. Specifically, "+" and "×" points denote the overlapping points of two patches respectively. (c) The second step of the alignment process. The transformed embedding points of two patches are denoted similar with denotations depicted in (b). (d) The third step of the alignment process. The aligned embeddings of both patches are depicted together.

Here we present an efficient greedy algorithm to realize this purpose, to make the proposed patch-by-patch alignment method be effectively implemented. The core idea of the algorithm is to incrementally yield a sequence of points from the original dataset, and then to construct the neighborhood patch sequence around the points in the sequence under appropriate neighborhood sizes which are adaptively selected to avoid well-known short-circuit issue. Evidently, three problems are involved in such an algorithm: 1) how to initiate the start point of the sequence; 2) how to adaptively specify the neighborhood size around a point in the sequence; and 3) how to incrementally generate the next point of the sequence.

About issue 1), the start point of the sequence is preferred to be randomly chosen from the original dataset. The experiments have illustrated that the selection of the start point will have unsubstantial effect on the final alignment result. Generally speaking, if the point is selected in the center of the dataset, as utilized in [32-33], the proposed method can be of a bit better performance. Yet this needs more computational expense. Hence simple random selection is still suggested.

The above issue 2) actually is the well-known neighborhood-size specification problem in manifold learning area. Specifically, if the size is chosen to be too large, the so-called "short-circuit" problem will occur, i.e., some neighborhood patches will deviate from the underlying manifold and hence will not have linear figures at all[32-33]. On the contrary, if the size is chosen too small, the discontinuity of the neighborhood graph arises. Therefore, we give the following algorithm in solving this issue.

Algorithm 1. Neighborhood Size Selection

```
Step I (K-NN Generation):
    Search $K$ nearest neighbors of $\boldsymbol{x}$:
        $K\text{-}NN(\boldsymbol{x}) = \{\boldsymbol{x}_i\}_{i=1}^{K}$;
    Calculate the diameter of $K\text{-}NN(\boldsymbol{x})$:
        $d_{\max} = \max_{\boldsymbol{x}_p, \boldsymbol{x}_q \in K\text{-}NN(\boldsymbol{x})} D(\boldsymbol{x}_p, \boldsymbol{x}_q)$;
Step II (Jumping Distance Detection):
    Initiate set $CurrentSet = \{\boldsymbol{x}\}$,
        $RemainSet = K\text{-}NN(\boldsymbol{x})/\{\boldsymbol{x}\}$;
    $j = 1$;
    while $RemainSet \neq \varnothing$ do
        $\boldsymbol{x}_j = \arg\min_{\boldsymbol{x}_i \in RemainSet} D(\boldsymbol{x}, \boldsymbol{x}_i)$;
        $k = j$;
        if $d(\boldsymbol{x}_j, CurrentSet) < \eta d_{\max}$ then
            $CurrentSet = CurrentSet \cup \{\boldsymbol{x}_j\}$;
            $RemainSet = RemainSet/\{\boldsymbol{x}_j\}$;
            $j = j + 1$;
        else then
            $k = k - 1$;
            terminate while;
        end if
    end while
```

The neighborhood size k of $\boldsymbol{x}$ can then be output and utilized.

In the algorithm, $D(\boldsymbol{x}_p, \boldsymbol{x}_q)$ is the distance between the points $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$ and $d(\boldsymbol{x}, CurrentSet)$ means the minimal distance between $\boldsymbol{x}$ and the points in set *CurrentSet*. K is the suitably large neighborhood size for helping avoiding the discontinuity problem. And the jumping distance here is finding a j-th nearest neighbor $\boldsymbol{x}_j$ which is not near to any of $\boldsymbol{x}$'s $(j-1)$ nearest neighbors compared with the diameter of the neighborhood patch. It causes a "jump" on the distance sequence $d(\boldsymbol{x}, CurrentSet)$. Hence, by collecting $\boldsymbol{x}$'s $(j-1)$-*NN*, the short-circuit can be easily avoided.

Both of the parameters K and η should be determined by the distribution of the dataset. The number of iteration in IAM we will introduce later is equal to the number of patches and the larger the K is, the less the patches are. So the choice of K should be large enough to make IAM faster but without worrying about the shortcut problem because of the jumping distance detection step. However, K cannot be chosen too large in order to keep the linearity of the patch because the alignment algorithm is under the assumption of local linearity. For η, experimentally, $\frac{1}{4}$ to $\frac{1}{3}$ will be suitable for most situations.

For issue 3), the following algorithm is specifically designed. The interpretation of the procedure of the algorithm will be given after presenting the algorithm as follows.

Algorithm 2. Incrementally Searching Patches

```
Step I:
    Initiate start point $\boldsymbol{x} \in \boldsymbol{X}$, $C_{\text{detected}} = \{\boldsymbol{x},$
        $k\text{-}NN(\boldsymbol{x})\}$, $C_{\text{undetected}} = \boldsymbol{X}/\{\boldsymbol{x}, k\text{-}NN(\boldsymbol{x})\}$,
        $S_1 = C_{\text{detected}}/\{\boldsymbol{x}\}$, $S_2 = \varnothing$;
Step II:
    while $C_{\text{undetected}} \neq \varnothing$ do
    Step II.1:
        Denote $\boldsymbol{z}$ as the first element in sequence $S_1$,
            $S_1 = S_1/\{\boldsymbol{z}\}$;
        Search visible neighbors[1] of $\boldsymbol{z}$: $VN(\boldsymbol{z})$;
        if $VN(\boldsymbol{z}) \subset C_{\text{detected}}$ then
            Goto next iteration;
        else then
            Specify neighborhood size $k$ of $\boldsymbol{z}$ by utilizing
            Algorithm 1 and goto Step II.2;
        end if
```

```
Step II.2:
        $C_{kNN} = k\text{-}NN(\boldsymbol{z}) \cap C_{\text{undetected}}$;
        $S_1 = S_1 \cup C_{kNN}$, $S_2 = S_2 \cup \{(\boldsymbol{z}, k)\}$;
        $C_{\text{detected}} = C_{\text{detected}} \cup C_{k\text{-}NN}$,
        $C_{\text{undetected}} = C_{\text{undetected}} / C_{k\text{-}NN}$;
    end while
Step III:
    Output $S_2 = \{\boldsymbol{x}_i, k_i\}_{i=1}^{s}$ as detected sequence.
```

Note that the denotation C_1/C_2 means deleting any element of C_2 from C_1 if it is contained in C_1.

In the above procedure, S_1 is the candidate set which records all the data might be at the "border" of detected data, and S_2 is the current detected point sequence, along with the correspondingly selected neighborhood sizes. C_{detected} records the dataset constructed by the overlapping of the neighborhood patches around the current point sequence S_2, and $C_{\text{undetected}}$ records the undetected points in the current shape, i.e., the complement of C_{detected} in X.

In Step II.1, from the current candidate set S_1, we incline to select the points near to the border of current detected set C_{detected} so as to accelerate the speed of the incremental alignment iteration. To this end, it only needs to detect whether the visible neighbors of a point in S_1 is completely belong to C_{detected}. If yes, it can be reasonably considered that the point is in the inner part of the detected patch set; otherwise, the point can be adopted as the next point in the sequence. And then in Step II.2, update all the sets including S_1, S_2, C_{detected} and $C_{\text{undetected}}$ for further searching.

It should be emphasized that the proposed algorithm always puts the nearest neighbors of the newly-detected point at the end of the candidate sequence S_1. This means that only after the procedure detects all of the neighboring points around a previously detected point, those of the subsequent candidates can then be considered. This property guarantees the collection of the neighborhood patches around the detected points in sequence S_2 incrementally expands in different directions, while averting the abnormality that it only stretches to some specific orientation. This can be directly observed from Fig.2, which shows the process of the proposed algorithm when being applied to the cylinder dataset as shown in Fig.6.

2.3 To Deal with Loopy Cases

However, combining the alignment and incremental idea mentioned above cannot solve some special situations, such as the cylinder like manifold data (as depicted in Fig.6). Here, we discuss how to make the algorithm more adaptive.

The NLDR of data lying on the manifold with intrinsic loopy structures is generally dealt with by cutting the data manifold along one generatrix line, and then unrolling the manifold data to construct the corresponding low-dimensional data embeddings[16,34]. In our case, the encountered problem by loopy cases is depicted in Fig.3. For a newly aligned neighborhood patch KNN_i in Step II of Algorithm 3, if the overlapping part between KNN_i and the set $CurrentX$ is just located on the generatrix line, it is evident that we cannot implement effective alignment between KNN_i and $CurrentX$ by applying the method presented in Subsection 2.1. Therefore, such a kind of neighborhood patches should be tackled in a specific way.

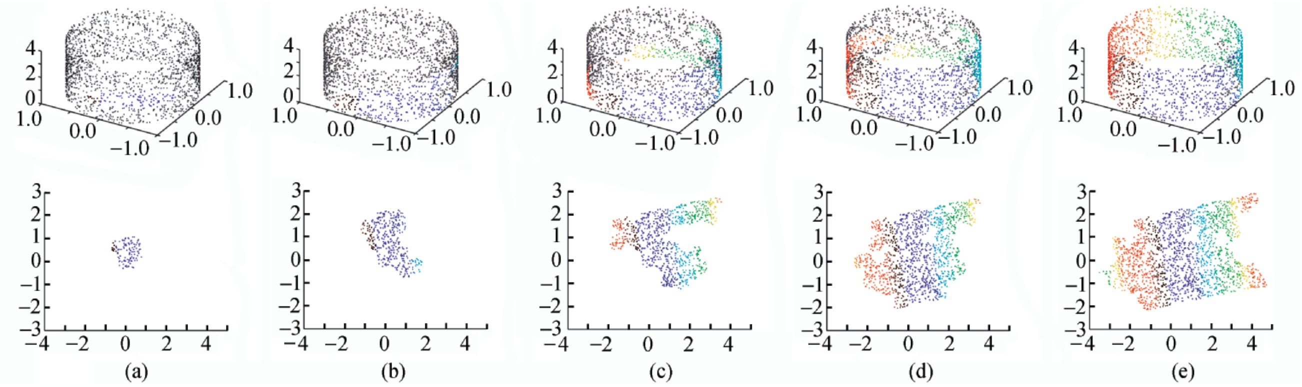


Fig.2. The incrementally selected neighborhood patches by applying the proposed incremental technique at (a) 6th, (b) 20th, (c) 50th, (d) 80th and (e) 120th iterations respectively. The lower sub-figures correspondingly show the low-dimensional embeddings of the neighborhood patches calculated by Algorithm 3.

[1] A point $\boldsymbol{a}$ is said to be a visible neighbor of $\boldsymbol{x}$, if there is no other point $\boldsymbol{z}$ that stays in the area between $\boldsymbol{a}$ and $\boldsymbol{x}$ to separate them. Equivalently, it requires that the angle formed by $\boldsymbol{zx}$ and $\boldsymbol{za}$ should be acute or right, but not obtuse, i.e., $\{\boldsymbol{a} \in k\text{-}NN(\boldsymbol{x}) | \langle \boldsymbol{x} - \boldsymbol{z}, \boldsymbol{a} - \boldsymbol{z} \rangle \geqslant 0, \forall \boldsymbol{z} \in k\text{-}NN(\boldsymbol{x})\}$. This property guarantees detection for least neighborhoods which can provide enough boundary information in incremental patch searching. The details can be referred to in [16].

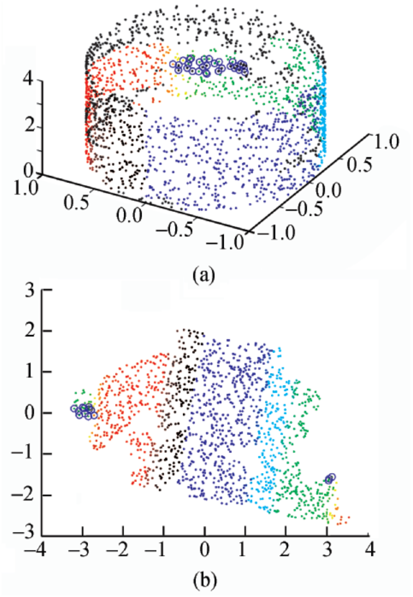


Fig.3. Graphical presentation of the loopy issue encountered by Algorithm 3. (a) The circles are figured as a local patch lying on the generatrix line of the loopy manifold. (b) Low-dimensional embeddings of the data points depicted in (a).

Here we introduce an extra alignment criteria for detecting and solving loopy cases. Firstly initiate a threshold $\varepsilon > 0$, and then for each involved neighborhood patch (i.e., KNN_i around $\boldsymbol{x}_i$), detect whether the alignment error E^* between KNN_i and $CurrentX$ (corresponding to the value calculated in (2)) is less than the preset threshold ε. If yes, the iteration is continued; otherwise, it means it has detected a loopy structure, because the large alignment error is caused by the overlapping of the new coming patch with two sides of the generatrix line while the calculated low-dimensional coordinates of these two sides are not connective but located in the two ends of the unfolded manifold. Then we need to make the new coming patch just overlapping with one side of the generatrix line for alignment as following: from the patch sequence KNN_j, $1 \leqslant j < i$, select intersectant one with KNN_i and where the smallest alignment error between KNN_i and KNN_j can be attained. Then by taking the selected KNN_j as the alignment base, the embedding set of KNN_i can then be reasonably calculated, and the loopy abnormity can be simultaneously alleviated. To reduce the number of parameters in the proposed algorithm, we choose ε automatically as follows: since the loopy issue tends not to occur in the beginning of the alignment iterations, we can estimate a suitable ε by setting it as the maximal alignment error in the first several iterations (say, 10) without regarding loopy structure.

After adding this part to the algorithm, the efficiency will be the most important issue we care about. However, for non-loopy cases, this step can be directly skipped, and the computation of the algorithm will not be affected at all. Furthermore, even for dataset with intrinsic loopy structures, the increase of the computational complexity for this part will be unsubstantial, which will be further evaluated in the next subsection.

To sum up, we give the whole IAM as follows:

Algorithm 3. Incremental Alignment Method

```
Step I (Incremental Step):
  Incrementally generate sequence
    $S_2 = \{(\boldsymbol{x}_i, k_i)\}_{i=1}^{s}$ by utilizing Algorithm 2;
  $CurrentX = \varnothing$, $CurrentY = \varnothing$;
Step II (Alignment Step):
Step II.1:
  Generate neighborhood set $kNN_1$ by $(\boldsymbol{x}_1, k_1)$;
  Calculate low-dimensional embeddings
    $\boldsymbol{Y}_1 = \{\boldsymbol{y}_j\}_{j=1}^{k_1}$ of $kNN_1$;
  $CurrentX = kNN_1$, $CurrentY = \boldsymbol{Y}_1$;
  for $i = 2, 3, \ldots, s$
Step II.2:
    Generate $kNN_i$ by $(\boldsymbol{x}_i, k_i)$;
    Calculate $\boldsymbol{Y}_i$ aligned to $CurrentY$ by utilizing the
      alignment method presented in Subsection 2.1;
    Calculate alignment error $E^*$ by (2);
    if $E^* < \varepsilon$ then
      Update $CurrentY$ with $\boldsymbol{Y}_i$ by (3), $CurrentX =$
        $CurrentX \cup kNN_i$;
      Goto next iteration;
    else then
      Goto Step II.3;
    end if
Step II.3 (Loopy Structure Adaptation):
    for $j$ from 1 to $i-1$, calculate alignment embeddings $\boldsymbol{Y}_i^{(j)}$ and alignment error $E(\boldsymbol{Y}_i^{(j)})$ aligned
      with low-dimensional embeddings in $CurrentY$
      of $kNN_j$ respectively (skip unoverlapping patches
      with $kNN_i$);
    $\boldsymbol{Y}_i^* = \underset{\boldsymbol{Y}_i^{(j)}, 1 \leqslant j < i}{\arg\min} E(\boldsymbol{Y}_i^{(j)})$;
    Update $CurrentY$ with $\boldsymbol{Y}_i^*$ by (3), $CurrentX =$
      $CurrentX \cup kNN_i$;
  end for
Step III:
  Output $\boldsymbol{Y} = CurrentY$ as the final embedding result
    of $\boldsymbol{X}$.
```

2.4 Computational Complexity of IAM

Undoubtedly, the computational speed of IAM is one of the most crucial issues in its applications. To clarify this point, the computational efficiency of IAM is examined in this subsection.

Let us consider the algorithm without loopy cases first. For Step I of Algorithm 3 (i.e., Algorithm 2), the nearest neighbor searching (Step II.1 of Algorithm 2) for each iteration substantially determines the computational complexity of this process. Since the number of iterations is equal to the size l of the input dataset $\boldsymbol{X}$, this step takes no more than $O(nl\log l)$ computational cost by utilizing the K-D trees or ball trees method[35-36]. For each iteration of Step II.2 in Algorithm 3, searching k_i-NN of $\boldsymbol{x}_i$ takes $O(nl\log l)$ time, and solving quasi-Newton optimization introduced in Subsection 2.1 costs no more than $O(Kd^2)$ time by utilizing the well-known BFGS method[37]. Therefore, altogether Step II.2 without considering loopy cases of Algorithm 3 costs no more than $O(s(nl\log l + Kd^2))$ computational expense. As $s = l/K$, the entire computational cost of IAM without loopy structure adaptation is at most $O(nl^2\log l/K + ld^2 + nl\log l)$. Compared with the computational complexity of most of the current manifold learning method, such as $O(Knl^2\log l + l^3)$ of ISOMAP[38], $O(nl^2 + nlK^3)$ of LLE[35], $O(nl\log l + lK^2 + l^2\log l)$ of RML[16], and $O(nl\log l + Kl + l^3/K^3)$ of LMDS[14]②, despite the computational complexity of IAM has no more orders of l than others, it has the advantage that it is inverse proportional to K. And as l increases, for most of instances, it means the dataset becomes more dense, then K can be even larger. Therefore, the real computational complexity of IAM is between $O(nl\log l)$ and $O(nl^2\log l)$. That is, the computational speed of IAM excels most of the current manifold learning methods.

For the amended IAM algorithm of considering the loopy cases, the supplemental Step II.3 generally has an immaterial effect on the computational speed of the whole algorithm. Actually, the patches lying on the generatrix line only consist of a minor part of the whole patches involved in the alignment process. That is to say, the time m implementing Step II.3 of Algorithm 3 is generally far less than the entire iteration time s of the alignment step. Since for each neighborhood patch kNN_j involved in Step II.3, the quasi-Newton optimization is implemented at most s times on the alignment of kNN_i and kNN_j, $1 \leqslant j < i$, the increased computational complexity of the amended IAM algorithm is thus no more than $O(msKd^2)$, whose order is generally lower than $O(lKd^2)$, and further lower than $O(lKnd)$. Therefore, the amended algorithm does not substantially impact the computational complexity of the whole algorithm. This will be further experimentally evaluated by a series of synthetic and real-world datasets in the next section.

3 Experimental Results

To evaluate the performance of the proposed method, multiple synthetic and real benchmark datasets were utilized as the test basis. Four synthetic datasets, including the Swiss roll, Swiss hole, cylinder, 5-cluster data (as depicted in the upper left figures of Figs. 4∼7

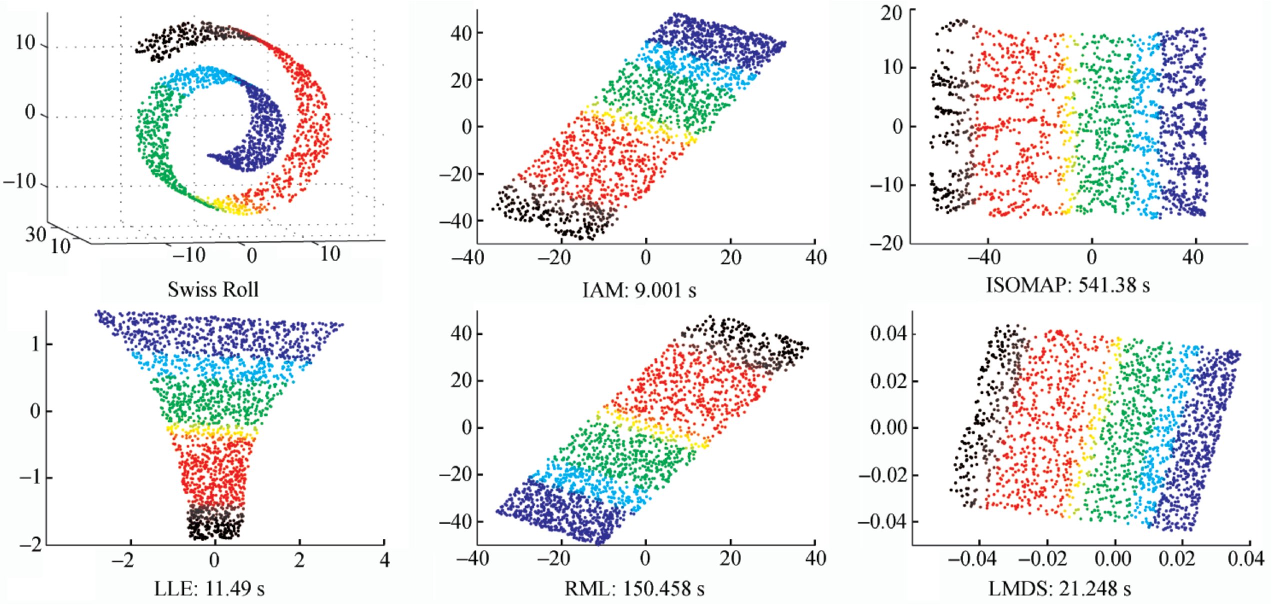


Fig.4. 2-D data embeddings of the Swiss roll dataset, calculated by IAM, ISOMAP, LLE, RML, and LMDS respectively.

② Actually, the computational complexity of LMDS is around $O(nl\log(l) + Kl + m^3)$, where m is the number of the overlapping patches and no more than l/K. We substitute the latter for the former in the above evaluation to make it easier to be compared with other complexities.

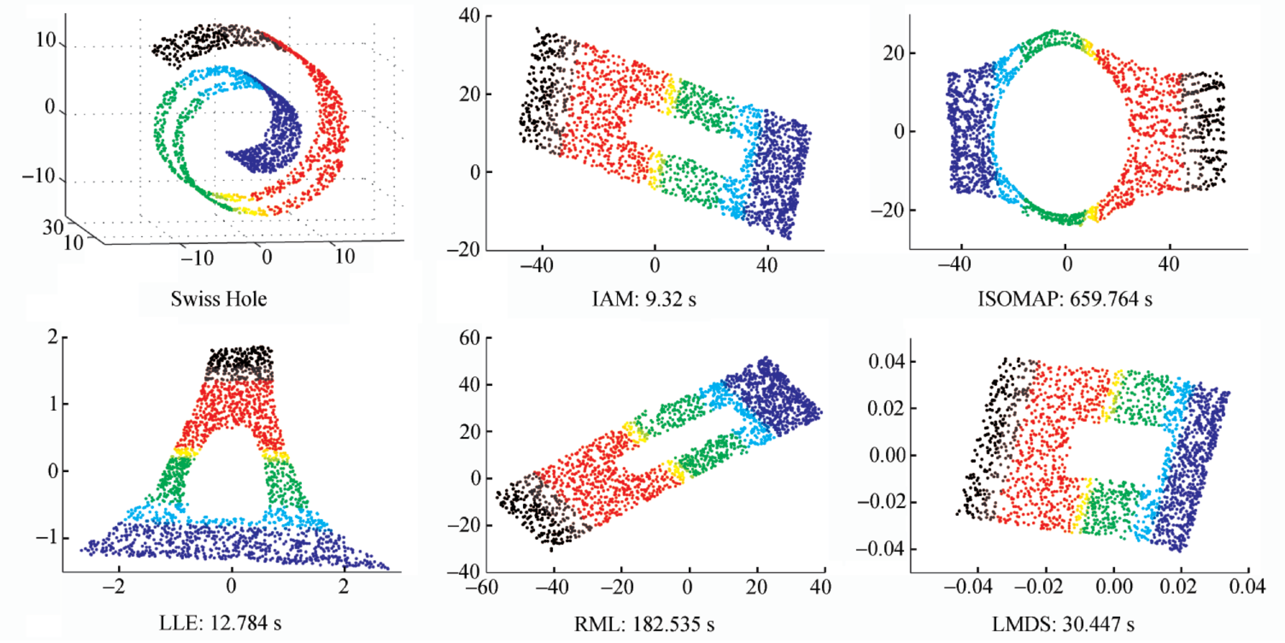


Fig.5. 2-D data embeddings of the Swiss hole dataset, calculated by IAM, ISOMAP, LLE, RML, and LMDS respectively.

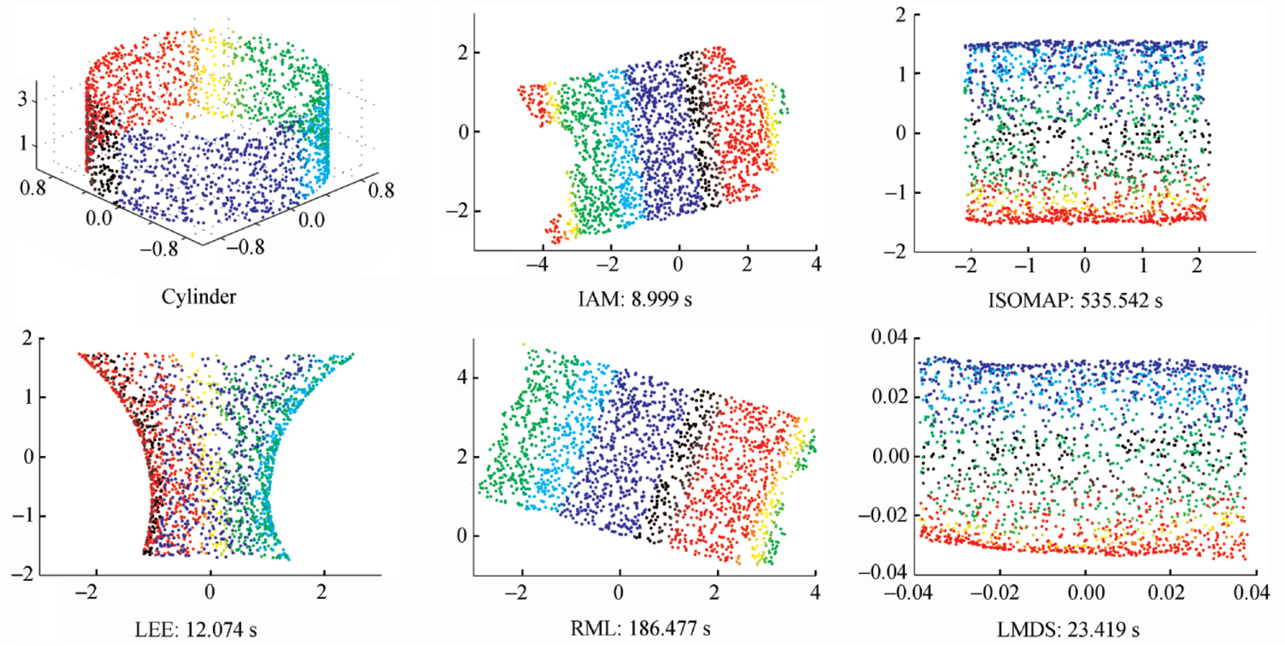


Fig.6. 2-D data embeddings of the cylinder dataset, calculated by IAM, ISOMAP, LLE, RML, and LMDS respectively.

respectively), are firstly employed for evaluation. And 2 real datasets, including ISOMAP face data and LLE face data, are analyzed for further substantiation. Furthermore, the experiment on the influence of data size on the efficiency and out-of-sample extension are shown later. All programs were implemented on the Matlab 7.0 platform. The implementation environment was the personal computer with AMD 2.6 GHz CPU, 4 GB memory and Windows XP operating system.

3.1 Experiments on Synthetic Data Sets

The proposed method is firstly tested on four synthetic datasets. For comparison of learning effects and time consumptions, four of the existing methods, including ISOMAP, LLE, RML, and LMDS methods, have

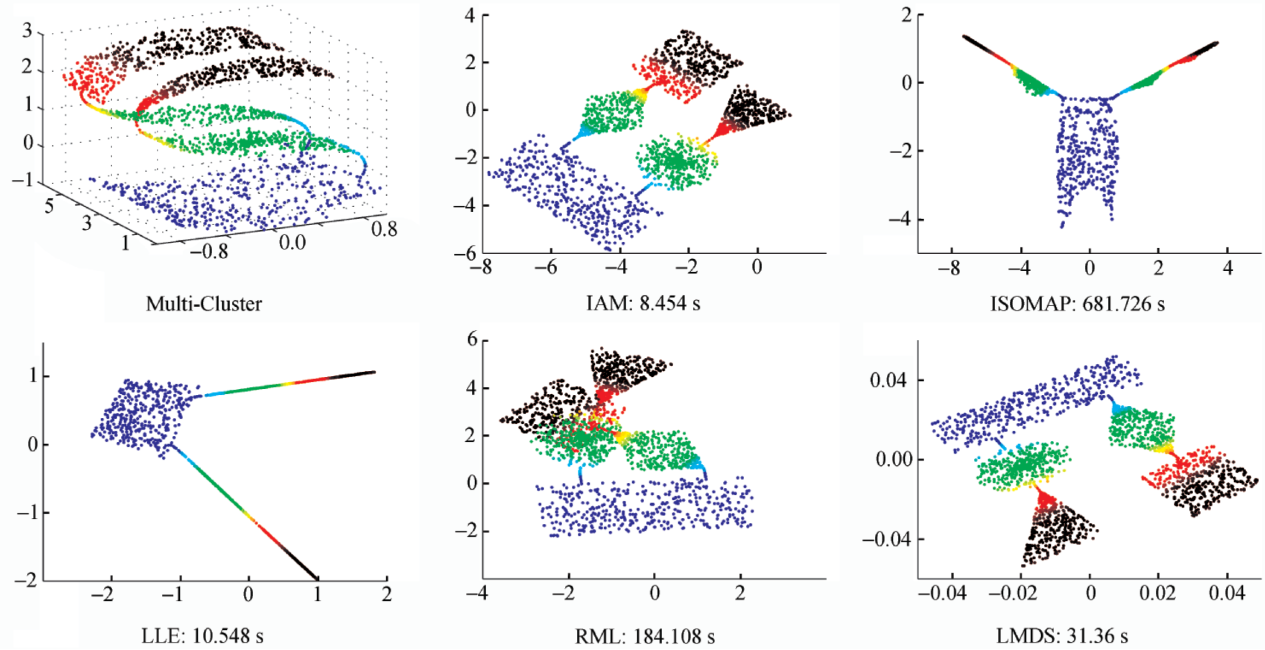


Fig.7. 2-D data embeddings of the 5-cluster dataset, calculated by IAM, ISOMAP, LLE, RML, and LMDS respectively.

also been utilized. The corresponding 2-D embedding results are depicted and corresponding time consumptions are shown in Figs. 4∼7 respectively. In the experiments, we will notice the disadvantage of incremental approach (represented by RML), such as time consuming and local minima problem, and disadvantage of alignment approach (represented by LMDS), such as weak extensibility for loopy datasets. The analysis for these results are listed detailedly as follows.

Swiss Roll Data. The dataset consists of 2000 points, randomly generated from the classical Swiss roll manifold, as depicted in Fig.4. From this figure, it is easy to observe that all of the five methods obtain satisfactory low-dimensional embeddings of the original data, except that the embedding figure obtained by LLE is a little distorted in the global scale.

Swiss Hole Data. The dataset resides on a Swiss-roll-like manifold distribution. Yet different from the classical Swiss roll manifold, a piece of sub-manifold is absent at the inner part of the manifold, wrapped from a 2-D rectangle area, as shown in Fig.5. 2150 samples were generated from such a Swiss hole manifold, and their embeddings calculated by the proposed method and four current methods are demonstrated in Fig.5. Evidently, IAM, RML, and LMDS perform perfectly by preserving the geometry around the hole. Yet ISOMAP and LLE yield distorted shapes around the hole.

Cylinder Data. The 2000 points of this dataset are sampled from a manifold distribution with cylinder shape, which is of intrinsic loopy structures obviously. By observing the experimental results depicted in Fig.6, it is evident that IAM as well as RML cuts the cylinder dataset along one generatrix line and unroll it to form a long stripe. However, other three methods (ISOMAP, LLE, and LMDS) produce incorrect mixed point clouds.

Multi-Cluster Data. The dataset with size 2080 is uniformly generated from the S-curve manifold composed by five clusters of globally connected sub-manifolds, wrapped by a 2-D rectangle, a diamond, a sphere, a square and a triangle respectively, as depicted in Fig.7. The figure also shows the 2-D embedding results obtained by the proposed method and other four current methods. Obviously IAM and LMDS can yield satisfactory results, in which the shapes of the five clusters are preserved and the global connection is maintained. ISOMAP and LLE can maintain the global connection, but the shapes of the clusters are degenerated. Note that RML also perfectly recovers the shapes of the original five clusters. However, the connections between embedding clusters are highly distorted such that the inter-cluster structures are seriously destroyed and the cluster information cannot be correctly reflected, which is actually caused by local minima problem.

To further support the computational complexity evaluation of IAM presented in Subsection 2.4, the computational times of all experiments were recorded and are depicted below the corresponding sub-figures of Figs.4∼7 respectively. Based on these records, it can be concluded that averagely the time consumption of IAM is about $\frac{1}{60}$, $\frac{3}{4}$, $\frac{1}{20}$, $\frac{1}{3}$ of that of ISOMAP, LLE, RML, and LMDS respectively. Together with the theoretical result obtained in Section 2, the efficiency of the proposed IAM is apparent.

3.2 Experiments on Real Datasets

To evaluate IAM on real-world dataset, ISOMAP face data and LLE face data were employed for testifying the performance of the proposed method. The objective is to embed each original high-dimensional data into a 3-D or 2-D space, in which the underlying data distributions can be easily visualized. The results are listed in detail as follows.

ISOMAP Face Data. The dataset contains 698 images, which are all 64×64-pixel (4096-dimensional) gray scale pictures. Fig.8 shows its 3-D representation calculated by the proposed IAM algorithm. It can be observed that in the embedding space, data points are uniformly distributed and well organized. Particularly, by observing the representative images shown next to the circled points in different parts of the space, it is easy to see that each coordinate axis of the embedding highly accords with one degree of representational freedom underlying the original data: x and y axes represent the degrees of left-right poses and up-down poses of the objective images respectively, and z axis reflects the angles of the light illuminating on the images.

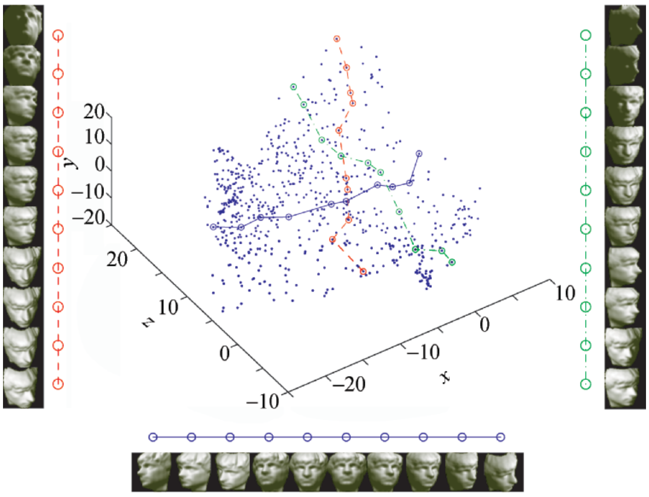


Fig.8. 3-D data embeddings of ISOMAP face data calculated by IAM.

LLE Face Data. The dataset consists of 1965 photographs, picturing the face of the similar person with various expressions. Each element of the set is scaled as a 20×28-pixel (560-dimensional) gray-scale image. The 2-D embedding points of the data obtained by applying the proposed IAM algorithm are depicted in Fig.9, along with some representative face images superimposed on the corresponding data points. From the figure, it can be observed that the similarity of the faces among the local regions is well preserved, and furthermore, the global cluster structure of similar facial expressions (such as smile, wink) is also nicely revealed.

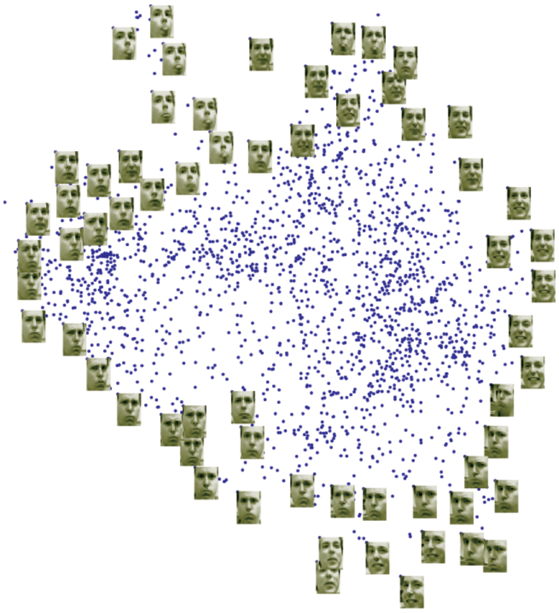


Fig.9. 2-D data embeddings of LLE face data calculated by IAM.

3.3 Experiments on Data Size Influence

As analysis in Subsection 2.4, the computational complexity of IAM is much less than most of existing manifold learning methods, which has been proved by previous experiments. Furthermore, the computational complexity of IAM should be influenced much less by the increase of data size. Here we show the relationship between time consumption and data size with 29 different data sizes for Swiss roll and cylinder datasets. In Fig.10 and Fig.11, we can tell the

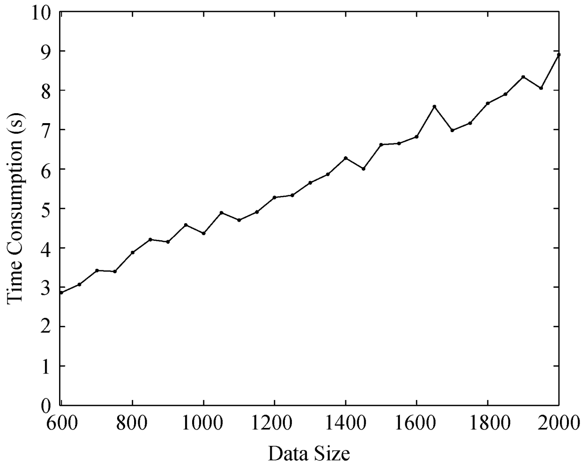


Fig.10. Relationship between time consumption and data size with 29 different data sizes for Swiss roll dataset.

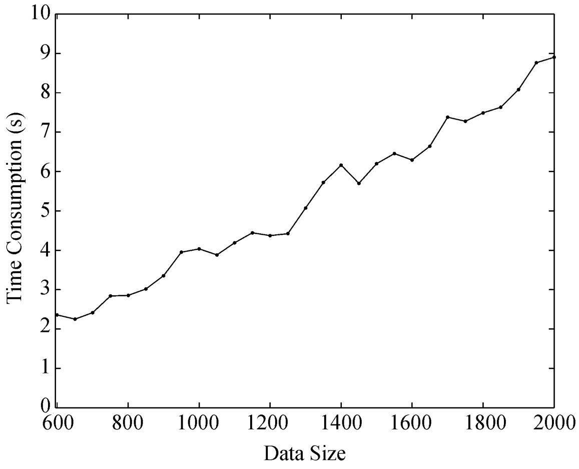


Fig.11. Relationship between time consumption and data size with 29 different data sizes for cylinder dataset.

computational complexity of IAM influenced by data size is very close to $O(l \log l)$, which is almost linear when data size l in such order of magnitude or higher. Additionally, although cylinder is a dataset with loopy structure, we see no significant extra time consumption in Fig.11 compared with the Swiss roll dataset in Fig.10. So it is convinced that the loopy structure adaptation step has very little influence on the efficiency of the algorithm.

3.4 Out-of-Sample Extension

One of the most valuable benefits of incremental method is out-of-sample extension. Particularly, for a newly input high-dimensional sample, firstly to search its k-NN under neighborhood size k specified by Algorithm 2, and to implement Step II of the IAM algorithm once on the neighborhood patch searched. Then the low-dimensional coordinate of the new sample in the representational space can be naturally generated. Fig.12 shows the experiment on out-of-sample extension. After we have the 2-dimensional embeddings of a Swiss roll dataset with data size of 1000, we can embed new coming samples into the original result by applying the method we mentioned above. Two groups of fifty extra new data are perfectly embedded into the original low-dimensional embeddings successively.

From this experiment, we can also see how incremental method works in IAM. It can update the global low-dimensional embeddings incrementally when new data involved. But different from incremental methods like [25, 29], which will update low-dimensional embeddings of whole dataset when new data involves, IAM is more like RML[16]. It only updates the embeddings locally near the new coming samples. Honestly, methods like [25, 29] will get more accurate global results as the data size gets larger and larger. However, IAM only

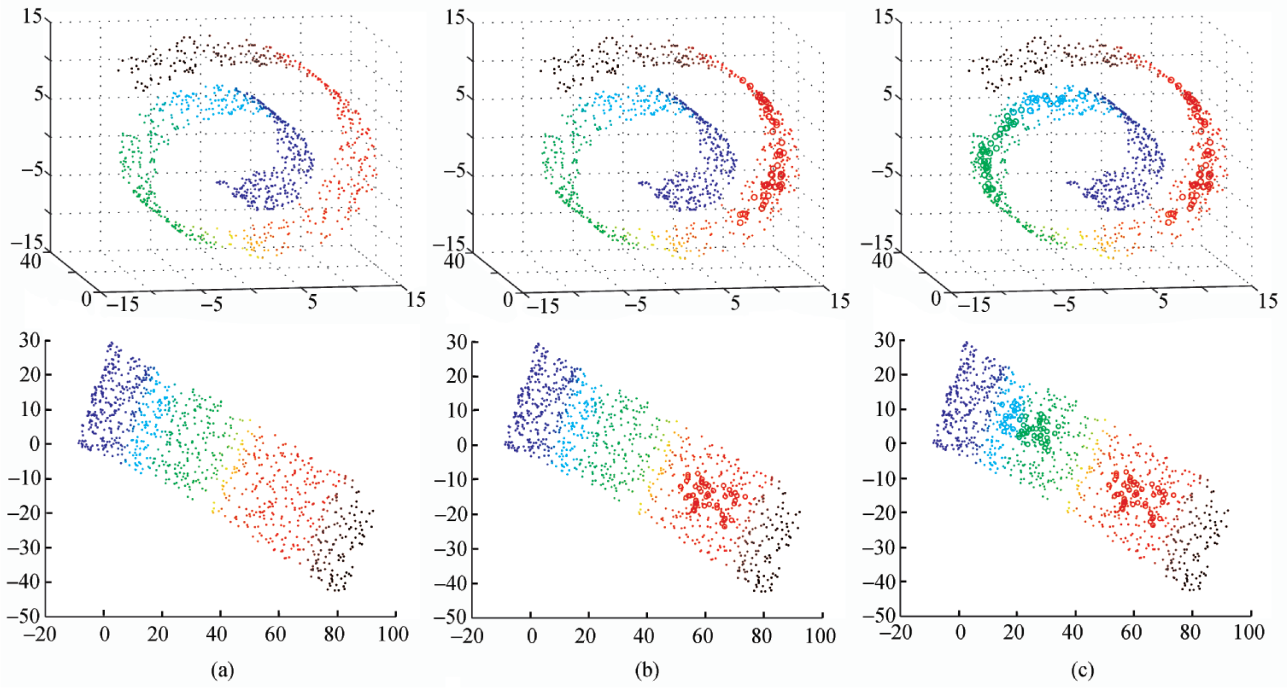


Fig.12. Out-of-sample extension of a Swiss roll dataset with original data size of 1000. The circles are figured as new coming samples. (a) 2-dimensional embeddings of Swiss roll dataset. (b) Fifty new coming samples are embedded into the low-dimensional result. (c) Further fifty new samples are embedded into the result.

refines limited results in the whole dataset but saves much more time in operation and the results can be also satisfactory because of its well local shape preserving property.

In summary, the proposed IAM algorithm outperforms the existing methods in the visualization task and computational speed on four synthetic datasets, and also achieves satisfactory embedding results on two real datasets. Furthermore, it has been proved that the size of dataset influences the efficiency of the algorithm very little because of its alignment property and as an incremental method, the out-of-sample extension can be easily approached.

4 Conclusion

In this paper, we have proposed a new manifold learning method, called the incremental alignment method (IAM). The most distinguished characteristic is the utilization of the patch-by-patch incremental implementation techniques. In specific, two steps have been involved in the proposed method: the incremental step and the alignment step. The former step incrementally searches and generates the neighborhood patch to be aligned in the latter step, and the latter one calculates the low-dimensional coordinates of the patch so generated and aligns them into the representational space. The proposed formulation has the following prominent advantages:

1) It has been theoretically and experimentally evaluated that the computational complexity of the proposed method is influenced much less by the increase of data size. This property makes IAM excels most of the current manifold learning methods in efficiency and can lead to large scale data applications in the future.

2) It has been empirically testified that the proposed method expands the applicability of the current manifold learning methods, such as the data lying on loopy manifold or multi-cluster manifold.

3) The incremental-learning property of the proposed method has implied its easy out-of-sample extension. The low-dimensional coordinate of newly input high-dimensional sample in the representational space can be easily generated just by operating one more iteration of the algorithm. The high simpleness and efficiency make IAM much more extensible.

4) The classical linear dimensionality technique (MDS or PCA) utilized in the alignment step of the proposed method has guaranteed the isometric property for each neighborhood patch. And consequently the global metric information of the entire data has also been well preserved on the low-dimensional representations calculated by the IAM method. The local minima issue has been well alleviated. This property promises their comparatively faithful performance on further pattern recognition applications.

The effectiveness of the proposed method will be further evaluated by more practical large-scale applications in our future investigation. Besides, the effectiveness of the IAM method needs to be further investigated when noise or outliers exist in the input data. Effort should be devoted to the analysis of these aspects in future research to further enhance the performance of the proposed method.

References

[1] Donoho D L. High-dimensional data analysis: The curses and blessings of dimensionality. American Math. Society Lecture, Match Challenges of the 21st Century, 2000.

[2] Roweis S T, Saul L K. Nonlinear dimensionality reduction by locally linear embedding. *Science*, Dec. 2000, 290(5500): 2323-2326.

[3] Tenenbaum J B, de Silva V, Langford J C. A global geometric framework for nonlinear dimensionality reduction. *Science*, Dec. 2000, 290(5500): 2319-2323.

[4] Bachmann C M, Ainsworth T L, Fusina R A. Exploiting manifold geometry in hyperspectral imagery. *IEEE Trans. Geoscience and Remote Sensing*, Mar. 2005, 43(3): 441-454.

[5] Lee J G, Zhang C S. Classification of gene-expression data: The manifold-based metric learning way. *Pattern Recognition*, Dec. 2006, 39(12): 2450-2463.

[6] Shin Y. Facial expression recognition of various internal states via manifold learning. *Journal of Computer Science and Technology*, Jul. 2009, 24(4): 745-752.

[7] Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 2003, 15(6): 1373-1396.

[8] Zhang Z, Zha H. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM J. Scientific Computing*, 2005, 26(1): 313-338.

[9] Donoho D L, Grimes C. Hessian eigenmaps: New locally linear embedding techniques for high-dimensional data. *Proc. the National Academy of Sciences*, 2003, 100(10): 5591-5596.

[10] Weinberger K, Saul L. Unsupervised learning of image manifolds by semidefinite programming. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, Washington DC, USA, Jun. 27-Jul. 2, 2004, pp.988-995.

[11] Lee J A, Lendasse A, Verleysen M. Nonlinear projection with curvilinear distances: ISOMAP versus curvilinear distance analysis. *Neurocomputing*, Mar. 2004, 57: 49-76.

[12] Hinton G, Roweis S. Stochastic neighbor embedding. In *Proc. NIPS 2002*, Vancouver, Canada, Dec. 9-14, 2002, pp.833-840.

[13] Agrafiotis D K, Xu H. A self-organizing principle for learning nonlinear manifolds. *Proceedings of the National Academy of Sciences*, 2002, 99(25): 15869-15872.

[14] Yang L. Alignment of overlapping locally scaled patches for multidimensional scaling and dimensionality reduction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Mar. 2008, 30(3): 438-450.

[15] de Silva V, Tenenbaum J B. Global versus local methods in nonlinear dimensionality reduction. In *Proc. NIPS 2003*, Vancouver and Whistler, Canada, Dec. 8-13, 2003, pp.705-712.

[16] Lin T, Zha H. Riemannian manifold learning. *IEEE Trans. Pattern Analysis and Machine Intelligence*, May, 2008, 30(5): 796-809.

[17] Roweis S T, Saul L K, Hinton G E. Global coordination of local linear models. In *Proc. NIPS 2001*, Vancouver, Canada,

Dec. 3-8, 2001, pp.889-896.

[18] Verbeek J. Learning nonlinear image manifolds by global alignment of local linear models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Aug. 2006, 28(8): 1236-1250.

[19] Bachmann C M, Alinsworth T L, Fusina R A. Exploiting manifold geometry in hyperspectral imagery. *IEEE Trans. Geoscience and Remote Sensing*, Mar. 2005, 43(3): 441-454.

[20] Teh Y W, Roweis S T. Automatic alignment of hidden representations. In *Proc. NIPS 2002*, Vancouver, Canada, Dec. 9-14, 2002, pp.841-848.

[21] Verveek J, Roweis S, Vlassis N. Non-linear CCA and PCA by alignment of local models. In *Proc. NIPS 2003*, Vancouver and Whistler, Canada, Dec. 8-13, 2003, pp.297-304.

[22] Zhang T, Yang J, Zhao D, Ge X. Linear local tangent space alignment and application to face recognition. *Neuralcomputing*, 2007, 70(7-9): 1547-1553.

[23] Cox T, Cox M. Multidimensional Scaling. Chapman and Hall, 1994.

[24] Law M H C, Zhang N, Jain A K. Nonlinear manifold learning for data stream. In *Proc. SIAM Data Mining*, Orlando, USA, Apr. 22-24, 2004, pp.33-44.

[25] Law M H C, Jain A K. Incremental nonlinear dimensionality reduction by manifold learning. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Mar. 2006, 28(3): 377-391.

[26] Kouropteva O, Okun O, Pietikäinen M. Incremental locally linear embedding. *Pattern Recognition*, 2005, 38(10): 1764-1767.

[27] Kouropteva O, Okun O, Pietikäinen M. Incremental locally linear embedding algorithm. In *Proc. Fourteenth Scandinavian Conference on Image Analysis*, Joensuu, Finland, Jun. 19-22, 2005, pp.521-530.

[28] Bengio Y, Paiement J F, Vincent P, Delalleau O, Le Roux N, Ouimet M. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. In *Proc. NIPS 2003*, Vancouver and Whistler, Canada, Dec. 8-13, 2003, pp.177-184.

[29] Zhao D, Yang L. Incremental isometric embedding of high dimensional data using connected neighborhood graphs. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2009, 31(1): 86-98.

[30] Jolliffe I T. Principal Component Analysis. Springer-Verlag, 1986.

[31] Yang J, Zhang D, Frangi A, Yang J. Two-dimentional PCA: A new approach to appearance-based face representation and recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, Jan. 2004, 26(1): 131-137.

[32] Meng D, Leung Y, Fung T, Xu Z. Nonlinear dimensionality reduction of data lying on the multi-cluster manifold. *IEEE Trans. Systems, Man and Cybernetics, Part B*, Aug. 2008, 38(4): 1111-1122.

[33] Meng D, Leung Y, Xu Z, Fung T, Zhang Q. Improving geodesic distance estimation based on locally linear assumption. *Pattern Recognition Letters*, May 2008, 29(7): 862-870.

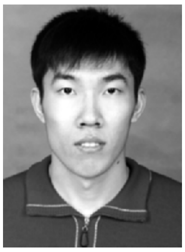
[34] Lee J A, Verleysen M. Nonlinear dimensionality reduction of data manifolds with essential loops. *Neurocomputing*, 2005, 67: 29-53.

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

[36] Friedman J H, Bentley J L, Finkel R A. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 1977, 3(3): 209-226.

[37] Nocedal J, Wright S J. Numerical Optimization, 2nd Ed. Berlin, New York: Springer-Verlag, 2006, p.24.

[38] de Silva V, Tenenbaum J B. Global versus local methods in nonlinear dimensionality reduction. In *Proc. NIPS 2002*, Vancouver, Canada, Dec. 9-14, 2002, pp.705-712.



Zhi Han received the B.Sc. and M.S. degrees in applied mathematics from Xi'an Jiaotong University (XJTU), China in 2005 and 2007, respectively. He is currently a Ph.D. candidate of applied mathematics in XJTU, and a joint Ph.D. candidate of statistics in University of California, Los Angeles, USA. His current research interests include nonlinear dimensionality reduction, manifold learning, computer vision and video representation.

De-Yu Meng received the B.Sc., M.Sc., and Ph.D. degrees in 2001, 2004, and 2008, respectively, all from XJTU, China. He is currently a lecturer with the Institute for Information and System Sciences, Faculty of Science, XJTU. His current research interests include priciple component analysis, nonlinear dimensionality reduction, feature extraction and selection, compressed sensing, and sparse machine learning methods.

Zong-Ben Xu received the M.Sc. degree in mathematics and the Ph.D. degree in applied mathematics from XJTU, China, in 1981 and 1987, respectively. In 1988, he was a postdoctoral researcher with the Department of Mathematics, The University of Strathclyde, Glasgow, U.K. He was a research fellow with the Information Engineering Department, the Center for Environmental Studies, and the Mechanical Engineering and Automation Department, The Chinese University of Hong Kong, China, and the Department of Computing, Hong Kong Polytechnic University, China. He is currently a professor with the Institute for Information and System Sciences, Faculty of Science, XJTU. His current research interests include manifold learning, neural networks, evolutionary computation, and multiple-objective decision-making theory.

Nan-Nan Gu received the B.Sc. degree in information and computing science from XJTU, China in 2006, and the M.Sc. degree in applied mathematics from XJTU, in 2009. Currently, she is working toward the Ph.D. degree in pattern recognition in the Institute of Automation, Chinese Academy of Sciences, Beijing, China. Her research interests include theory and application of manifold learning and nonlinear dimensionality reduction.