SIAM Inter Conf on Data Mining (SDM), Minneapolis, Minnesota, 2007, pp. 180–191

Nonlinear Dimensionality Reduction using Approximate Nearest Neighbors*

Erion Plaku Lydia E. Kavraki

Abstract

Nonlinear dimensionality reduction methods often rely on the nearest-neighbors graph to extract low-dimensional embeddings that reliably capture the underlying structure of high-dimensional data. Research however has shown that computing nearest neighbors of a point from a high-dimensional data set generally requires time proportional to the size of the data set itself, rendering the computation of the nearest-neighbors graph prohibitively expensive.

This work significantly reduces the major computational bottleneck of many nonlinear dimensionality reduction methods by efficiently and accurately approximating the nearest-neighbors graph. The approximation relies on a distance-based projection of high-dimensional data onto low-dimensional Euclidean spaces. As indicated by experimental results, the advantage of the proposed approximation is that while it reliably maintains the accuracy of nonlinear dimensionality reduction methods, it significantly reduces the computational time.

1 Introduction

Dimensionality reduction methods [6, 12, 15, 17, 18, 21, 24, 27, 39, 41, 42, 44, 46, 48] are widely used in many scientific applications. Dimensionality reduction facilitates the analysis of large amounts of high-dimensional data by extracting low-dimensional embeddings that effectively characterize the input data. It is often the case that data generated by scientific applications exhibit highly nonlinear features which can only be captured by nonlinear dimensionality reduction methods [15, 17, 18, 21, 39, 41, 42, 44, 46, 48].

Progress in scientific applications requires the efficient analysis of increasingly complex, large, and high-dimensional data. The extraction of low-dimensional embeddings that effectively characterize such data becomes progressively challenging.

Motivated by the success of Isomap [42], many nonlinear dimensionality reduction methods developed in recent years rely on the nearest-neighbors graph to capture the connectivity of the input data [15, 17, 18, 21, 39, 41, 44, 48]. The nearest-neighbors graph is typically computed by connecting each data point to its k nearest neighbors as determined by a distance metric. The nearest-neighbors graph enables dimensionality reduction methods to effectively characterize even global and nonlinear features present in a data set which cannot be adequately captured by linear dimensionality reduction methods [12,24,27]. Nonlinear dimensionality reduction methods that rely on the nearest-neighbors graph typically follow a common framework as summarized below.

1. Compute the nearest-neighbors graph of the input data by connecting each data point to its k nearest neighbors as defined by a distance metric. The weight of an edge in the graph is equal to the distance between its two endpoints.

2. Use the nearest-neighbors graph to construct a distance matrix M. A low-dimensional embedding is obtained by normalizing M and using MDS (multidimensional scaling) [12] to extract the top eigenvectors from M to form the basis of the embedding.

The nearest-neighbors graph which is used to effectively capture the data connectivity constitutes however the major computational bottleneck of nonlinear dimensionality reduction methods [5, 11, 15, 17, 18, 21, 39, 41, 42, 44, 48]. Although researchers have developed many nearest-neighbors algorithms [3, 8, 10, 13, 16, 19, 20, 25, 26, 33, 37, 47], theoretical and quantitative analysis has shown that computing nearest neighbors of a point from a high-dimensional data set requires in general time proportional to the size of the data set itself [4, 7, 23, 25, 29, 38, 43]. Consequently, nonlinear dimensionality reduction methods that rely on the nearest-neighbors graph are rendered computationally inefficient for the characterization of considerably large and high-dimensional data sets.

In this work we address the major computational bottleneck of nonlinear dimensionality reduction methods that rely on the nearest-neighbors graph. The proposed method, termed hcDPES (Hill-Climbing Distance-based Projection onto Euclidean Space), significantly reduces this bottleneck by efficiently and accurately approximating the nearest-neighbors graph. Mo-

*Work on this paper has been supported in part by NSF CNS 0615328, NSF 0308237, NIH GM078988 and a Sloan Fellowship to LK. Experiments reported in this paper have been obtained on equipment supported by NSF CNS 0454333, and NSF CNS 0421109 in partnership with Rice University, AMD, and Cray. The authors are with Rice University, Department of Computer Science, Houston, TX, 77005, USA. Email: {plakue, kavraki}@cs.rice.edu.

tivated by the work in [38], hcDPES projects a given high-dimensional data set onto a low-dimensional Euclidean space and computes neighbors in the projected space. The projection is based on a careful selection of pivots utilizing a hill-climbing approach that attempts to maximize the correlation of distances between data points and the corresponding projections.

We validate the accuracy and demonstrate the efficiency of hcDPES using high-dimensional data derived from different applications, such as image processing and robot motion planning. Image processing provides classical examples to test the reliability of hcDPES. The robot motion planning data provides challenging examples for nonlinear dimensionality reduction, since the data is highly nonlinear and distances between points are defined by complex and computationally expensive metrics [9]. Although the experiments in this paper use Isomap [17, 42], the approximate nearest-neighbors graph computed by hcDPES could be used with other nonlinear dimensionality reduction methods that rely on the nearest-neighbors graph. We chose Isomap [17, 42] as a test case due to recent work [15], which effectively characterizes high-dimensional scientific data with similar characteristics to the robot motion planning data.

The experimental results indicate that the approximate nearest-neighbors graph computed by hcDPES is remarkably similar to the exact nearest-neighbors graph. As a result, the low-dimensional embeddings extracted by Isomap using the exact nearest-neighbors graph or using the approximate nearest-neighbors graph computed by hcDPES are practically indistinguishable. Furthermore, hcDPES computes the approximate nearest-neighbors graph significantly faster than other approximate nearest-neighbors algorithms that maintain the same level of accuracy as hcDPES. Therefore, the advantage of hcDPES is that while it maintains the accuracy of nonlinear dimensionality reduction methods, it significantly reduces the computational time required to extract low-dimensional embeddings that effectively characterize high-dimensional data.

The rest of the paper is as follows. In §2, we discuss related work. In §3, we describe hcDPES and discuss how to quantitatively measure the accuracy of hcDPES. In §4 we describe the test cases and present the results of the experiments. We conclude in §5 with a discussion.

2 Related Work

2.1 Isomap

Let $S = \{s_1, \ldots, s_n\}$ denote a given data set and let $\rho : S \times S \rightarrow \mathbb{R}^{\geq 0}$ denote a distance metric. In the case of Isomap [17, 42], the distance matrix M approximates geodesic distances between points in S. As recent work [15] has shown, the approximation of geodesic distances is quite effective for characterizing nonlinear high-dimensional data derived from scientific applications. The geodesic distance between two points $s_i, s_j \in S$ is defined as the length of the shortest path from s_i to s_j when the path is constrained to lie on the surface of the manifold underlined by S and ρ. The geodesic distance between s_i and s_j is approximated as the length of the shortest path in the nearest-neighbors graph. Isomap selects L points as landmarks and uses the nearest-neighbors graph to construct an $L \times L$ distance matrix M, where the entries of M are computed as shortest-path distances between all landmark pairs. Details can be found in [17, 42].

2.2 Approximate Nearest Neighbors

Approximate nearest-neighbors algorithms trade off accuracy for computational efficiency [2,14,31,34] by computing neighbors that are not necessarily the same as the exact nearest neighbors. In many cases, the k neighbors of a point s computed by approximate algorithms are within an ϵ-ball from the exact nearest neighbors of s. The constant $\epsilon \geq 0$ expresses a trade-off between accuracy and computational efficiency, where accuracy indicates how close the approximate nearest neighbors are to the exact nearest neighbors. In the case of an n-point d-dimensional Euclidean data, approximate nearest neighbors of a point can be computed in $(d \log n/\epsilon)^{O(1)}$ time and $n^{1/\epsilon^{O(1)}}$ space, as summarized in [25]. The problem however remains challenging for general metrics, such as those commonly used in many scientific applications [1, 9, 15, 22, 30, 35, 36]. The efficiency or accuracy of approximate nearest-neighbors algorithms is typically reduced when general metrics are used instead of Euclidean distances [25].

A distinctive feature of hcDPES in contrast to existing work [2, 14, 31, 34] is that hcDPES does not use ϵ-balls to express the trade-off between accuracy and efficiency of computing approximate nearest neighbors. The challenge with ϵ-balls is that small values of ϵ are needed to compute approximate nearest neighbors that are close to the exact nearest neighbors. However, when ϵ is small computational advantages over exact nearest-neighbors algorithms are minimal, since a considerable amount of time is spent guaranteeing that all k approximate nearest neighbors are within an ϵ-ball from the exact nearest neighbors. In contrast, hcDPES quickly computes many approximate nearest neighbors without ensuring that all of them are within an ϵ-ball from the exact nearest neighbors. If a neighbor computed by hcDPES is far away, the nonlinear dimensionality reduction methods can easily reject it without impacting the results, since several neighbors are computed for each data point. In practice, however, as indicated

by the experimental results, the approximate nearest neighbors computed by hcDPES are within a small ϵ-ball from the exact nearest neighbors. The approximate nearest-neighbors graph computed by hcDPES is thus remarkably similar to the exact nearest-neighbors graph. Furthermore, hcDPES consistently offers significant computational speedups over approximate nearest-neighbors algorithms that maintain the same level of accuracy as hcDPES.

3 Methods

The approximation of the nearest-neighbors graph is based on projecting the data set S onto a low-dimensional Euclidean space according to distances between points in S and a set of pivots. The idea comes from our earlier work [38], which quantitatively analyzed the computational efficiency of proximity algorithms as a function of the data dimensionality in the context of robot motion planning [9]. The approximation of the nearest-neighbors graph in [38] is rather coarse and is limited in practicality to data up to 50–60 dimensions. As such, it is not suitable for dimensionality reduction since it fails to reliably capture the connectivity of high-dimensional data. In contrast, as indicated by the experimental results, the approximate nearest-neighbors graph computed by hcDPES captures the connectivity of high-dimensional data practically as well as the nearest-neighbors graph. This is achieved by a careful selection of pivots utilizing a hill-climbing approach that attempts to maximize the correlation of distances between data points and the corresponding projections. Furthermore, distances from data points to pivots are transformed using classical MDS [12] to minimize the distortion of distances resulting from the projection of high-dimensional data onto a low-dimensional Euclidean space. Consequently, the distance matrix M constructed in step 2 of §2.1 using the exact nearest-neighbors graph is remarkably similar to the distance matrix that is obtained when using the approximate nearest-neighbors graph computed by hcDPES. Since the embedding depends only on M, hcDPES can be used to effectively characterize nonlinear high-dimensional data.

In the rest of this section, we describe how the projection of S onto a low-dimensional Euclidean space is obtained and then describe how the projection is used to compute the approximate nearest-neighbors graph.

3.1 Projection of High-Dimensional Data

Let $E(S) = \{e(s_1), \ldots, e(s_n)\}$ denote the projection of S onto $\mathbb{R}^m$ for some fixed $m > 0$, where $e(s) \in E(S)$ denotes the projection of $s \in S$ onto $\mathbb{R}^m$. The objective is to find a projection $E(S)$ that can be efficiently computed while preserving the relative distances between points in S, i.e., the projections of points in S that are close according to ρ should be close according to the Euclidean distance in $\mathbb{R}^m$. Pseudocode is provided in Algorithm 3.1.

ALGORITHM 3.1. Distance-based projection of high-dimensional data onto a Euclidean space.

Input:
$S = \{s_1, \ldots, s_n\}$, a set of data points
$\rho : S \times S \to \mathbb{R}^{\geq 0}$, distance metric
m, dimension of Euclidean space
h, number of pivots
Output:
$E(S) = \{e(s_1), \ldots, e(s_n)\} \subset R^m$, projection of S

1: $P \subset S \leftarrow$ select h pivots (Algorithm 3.2)
2: $D(S, P, \rho) \leftarrow$ a $n \times h$ matrix
3: **for** each $s_i \in S$ **do**
4: **for** each $p_j \in P$ **do**
5: $v(s_i)[j] \leftarrow \rho(s_i, p_j)$
6: **end for**
7: $D(S, P, \rho)[i] \leftarrow$ set i-th row to $v(s_i)$
8: **end for**
9: $E(S) \leftarrow$ perform landmark MDS [17] to $D(S, P, \rho)$
10: **return** $E(S)$

The projection illustrated in Algorithm 3.1 is based on distances according to ρ between points in S and a set of $h > m$ pivots $P = \{p_1, \ldots, p_h\} \subset S$ (line 1). The pivot selection method is described in §3.2. For each $s \in S$, let $v(s)$ denote the vector of distances from s to each pivot, i.e., the j-th entry of $v(s)$ is equal to $\rho(s, p_j)$ (lines 4–6). Let $D(S, P, \rho)$ be the $n \times h$ distance matrix where the i-th row is equal to $v(s_i)$ (line 7). The projection $E(S)$ is then efficiently obtained by performing landmark MDS [17] on $D(S, P, \rho)$ (line 9).

3.2 Selection of Pivots

The selection of pivots greatly affects how well $E(S)$ preserves the relative distances of points in S. The selection scheme developed in this work attempts to maximize the correlation between distances according to ρ for any two points $s', s'' \in S$ and distances according to the Euclidean metric in R^h for $v(s')$ and $v(s'')$. If such distances correlate well, then the distortion resulting from the projection of S onto $E(S)$ is small. Pseudocode is given in Algorithm 3.2

The selection of pivots P is based on a hill-climbing algorithm. Let $C \subset S$ denote a set of candidate pivots selected from S (line 1). Initially each pivot in P is selected uniformly at random from the set C (line 2). At the beginning of each iteration, an index i is selected

uniformly at random from $\{1, \ldots, h\}$ (line 7). The i-th pivot of P is replaced by the point $c \in C$ that maximizes the correlation between distances according to ρ of points $s', s'' \in S$ and Euclidean distances between the corresponding $v(s')$, $v(s'')$ vectors (lines 8–18). More precisely, let $T \subset S \times S$ consist of pairs of points selected from S (line 3). Let $D(T, \rho) = \{\rho(s', s'') : (s', s'') \in T\}$ denote the distances according to ρ of each pair $(s', s'') \in T$ (line 4). Similarly, let $D(T, ||\cdot||) = \{||v(s'), v(s'')|| : (s', s'') \in T\}$ denote the distances according to the Euclidean metric in $\mathbb{R}^h$ between the corresponding $v(s'), v(s'')$ vectors (line 5). The i-th pivot is selected as the point $c \in C$ that maximizes the Pearson correlation between $D(T, \rho)$ and $D(T, ||\cdot||)$. This process is repeated until no significant improvement in the correlation between $D(T, \rho)$ and $D(T, ||\cdot||)$ occurs or when a maximum number of iterations is exceeded.

ALGORITHM 3.2. Pivot selection for projecting high-dimensional data onto a Euclidean space.

Input:
$S = \{s_1, \ldots, s_n\}$, a set of data points
$\rho : S \times S \to \mathbb{R}^{\geq 0}$, distance metric
h, number of pivots
Output:
$P = \{p_1, \ldots, p_h\} \subset S$, a set of pivots

```
1: C ⊂ S ← select candidate pivots at random
2: P ⊂ C ← select h initial pivots from C
3: T ⊂ S × S ← select pairs of points at random
4: D(T, ρ) ← {ρ(s', s'') : (s', s'') ∈ T}
5: D(T, ||·||) ← {||v(s'), v(s'')|| : (s', s'') ∈ T}
6: repeat
7:    i ← at random from {1, ..., h}
8:    p ← NIL; max_corr ← 0.0
9:    for each c ∈ C do
10:      temporarily set the i-th pivot of P to c
11:      update D(T, ||·||)
12:      corr ← correlation(D(T, ||·||), D(T, ρ))
13:      if corr > max_corr then
14:         max_corr ← corr
15:         p ← c
16:      end if
17:   end for
18:   set the i-th pivot of P to p
19: until termination criteria
20: return P
```

The replacement of the i-th pivot of P (line 10) changes $D(T, ||\cdot||)$. However, it is not necessary to recompute $D(T, ||\cdot||)$ at each iteration from scratch (line 11). In fact the replacement of pivot p_i by p causes only the i-th entry of vector $v(s)$ for each unique element s among the pairs in T to be changed to $\rho(p, s)$. Hence, for each $(s', s'') \in T$, $||v(s'), v(s'')||^2$ can be updated by subtracting $(\rho(p_i, s') - \rho(p_i, s''))^2$ from it and adding $(\rho(p, s') - \rho(p, s''))^2$ to it. Therefore, $D(T, ||\cdot||)$ can be easily updated by precomputing distances according to ρ between each $c \in C$ and each unique element among the pairs in T and then using a table lookup to update $||v(s'), v(s'')||$ for each $s', s'' \in T$.

The selection of pivots requires the evaluation of ρ for each $(s', s'') \in T$. It also requires the evaluation of ρ between each unique point in T and points in C. Therefore, the total number of evaluations of ρ is at most $|T| + 2|T||C|$. Since Algorithm 3.2 is essentially a greedy approach, random restarts and other techniques can be used to reduce the likelihood of getting stuck in local minima [40]. The experimental results presented in §4 indicate that projection of high-dimensional metric data using the pivot selection scheme described in this section can be used to efficiently and accurately approximate the nearest-neighbors graph.

3.3 Approximation of Nearest Neighbors

Initially, the projection $E(S)$ of S onto R^m is computed as described in §3.1. Then, k approximate nearest neighbors of each point $s \in S$ according to the distance metric ρ are computed in two steps:

1. compute $K > k$ candidate neighbors as the K exact nearest neighbors of $e(s)$ from $E(S)$ according to the Euclidean distance metric in $\mathbb{R}^m$; and
2. select the k closest points to s according to ρ from the K candidate neighbors.

When the projection $E(S)$ preserves the relative distances of points in S, it also preserves the relative ordering of points in S. Consequently, if a point s' is one of the k nearest neighbors of a query point s, then most likely $e(s')$ is also one of the k nearest neighbors of $e(s)$. The computation of $K > k$ candidate neighbors further improves the accuracy of hcDPES, since the likelihood that $e(s')$ is not one of the K nearest neighbors of $e(s)$ is even smaller. We remark that any exact nearest-neighbors algorithm can be used to compute the candidate neighbors in step 1. In our implementation, we used GNAT [8] and CoverTree [5], since these methods have been shown to perform well in practice.

The approximation of nearest-neighbors graph computed by hcDPES has certain computational advantages. The projection of S onto a low-dimensional Euclidean space $\mathbb{R}^m$ significantly reduces the number of distance evaluations required to resolve proximity queries. Additional computational gain results from the evaluation of the Euclidean distance in $\mathbb{R}^m$ which

can be typically computed more efficiently than complex distance metrics often used in scientific applications [8, 15, 28, 45].

3.4 Measuring the Approximation Accuracy

Let E be the embedding obtained from the application of a nonlinear dimensionality reduction method using the nearest-neighbors graph. Let $D(E)$ denote the vector of distances of all the pairs in $E \times E$. Similarly, let E_{hcDPES} and $D(E_{\text{hcDPES}})$ be the embedding and the vector of distances resulting when the nearest-neighbors graph is replaced by the approximate nearest-neighbors graph computed by hcDPES. The similarity between E and E_{hcDPES} is given by the Pearson correlation between $D(E)$ and $D(E_{\text{hcDPES}})$. A high correlation indicates that E_{hcDPES} preserves well distances between points in E and thus is similar to E.

The quality of approximate nearest neighbors depends on distances between approximate and exact nearest neighbors for each query point. One such common measure is the ratio of false dismissals (RFD) error [14], defined as follows:

$$\text{RFD}_\epsilon = \frac{1}{k} \sum_{s \in \text{ANN}_S(s_i,k)} \begin{cases} 1, & \rho(s, s_i) > (1+\epsilon) d_k, \\ 0, & \text{otherwise}, \end{cases}$$

where $d_k = \max_{s' \in \text{NN}_S(s_i,k)} \rho(s_i, s')$, $\text{NN}_S(s_i, k)$ denotes the k exact nearest neighbors of s_i, and $\text{ANN}_S(s_i, k)$ denotes the k approximate nearest neighbors of s_i computed by hcDPES. The RFD_ϵ error, $\epsilon \geq 0$, indicates the fraction of points in $\text{ANN}_S(s_i, k)$ that are $(1+\epsilon)$-times farther away from the k-th nearest neighbor of s_i. Since the RFD error does not measure how far each neighbor is from s, we also use the ratio of distance error (RDE) [14]. The RDE error is $1 - \alpha/\beta$, where α and β are the sum of distances from s to each exact and approximate nearest neighbor, respectively, i.e.,

$$\text{RDE} = 1 - \frac{\sum_{s \in \text{NN}_S(s_i,k)} \rho(s, s_i)}{\sum_{s \in \text{ANN}_S(s_i,k)} \rho(s, s_i)}.$$

The RDE error indicates how close approximate nearest neighbors are to the exact nearest neighbors.

4 Experiments and Results

The experiments in this paper provide quantitative evidence that (i) low-dimensional embeddings extracted from high-dimensional data by nonlinear dimensionality reduction methods, i.e., Isomap [17, 42], using the exact nearest-neighbors graph or using the approximate nearest-neighbors graph computed by hcDPES are remarkably similar, and (ii) hcDPES significantly reduces the computational time. We use several test cases selected from different application areas, such as image processing and robot motion planning.

4.1 Comparisons and Presentation of Results

To facilitate discussion, we write I-Exact and I-Approx to denote Isomap using the exact or an approximate nearest-neighbors graph, respectively. We also write I-A to specify the particular method A used by Isomap for the computation of the proximity graph.

Experiments in this paper use GNAT [8] and CoverTree [5] for comparisons with other exact and approximate nearest-neighbors algorithms, since GNAT and CoverTree have been shown to work well in practice. Both GNAT and CoverTree can be used to compute approximate nearest neighbors by specifying an ϵ, i.e., the approximate nearest neighbors computed by GNAT_ϵ and $\text{CoverTree}_\epsilon$ are guaranteed to have $\text{RFD}_\epsilon = 0.0$. When $\epsilon = 0.0$, GNAT and CoverTree compute exact nearest neighbors. For each experiment, we report the following results:

a. Accuracy of extracted embeddings. This is measured as the Pearson correlation between embeddings extracted by the application of I-Exact and I-hcDPES, as described in §3.4.

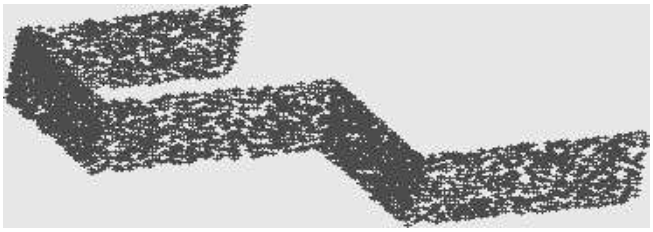
b. Accuracy of approximating the nearest-neighbors graph. This is measured using the RFD and RDE errors, as described in §3.4.

c. Computational speedup. We report the speedup obtained by I-hcDPES when compared to Isomap using other efficient proximity methods: (i) exact: $\text{GNAT}_{\epsilon=0.0}$, $\text{CoverTree}_{\epsilon=0.0}$ and (ii) approximate: $\text{GNAT}_{\epsilon=0.15}$, $\text{CoverTree}_{\epsilon=0.15}$.

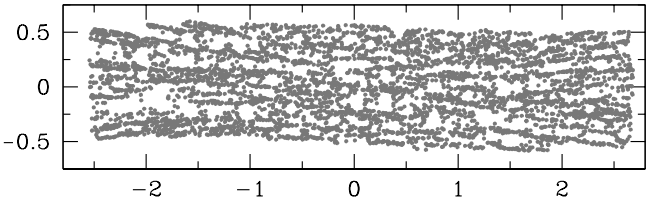
The parameter values used in the experiments of this work are as follows: number of candidate pivots $|C| = 0.1n$; $|T| = \min(5000, n)$; number of candidate neighbors $K = k + 10$; number of pivots $h = 2m$. These values were determined empirically based on the data sets used for the experiments. We are in the process of automating the selection of good parameter values.

4.2 Test Case: A Low-Dimensional Surface in a High-Dimensional Euclidean Space

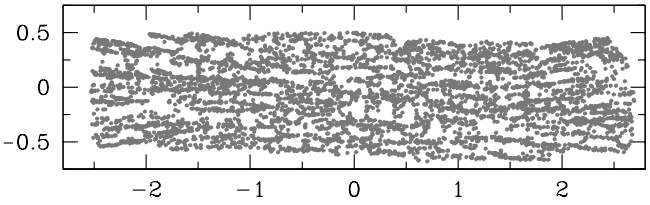
The first test case consists of a low-dimensional surface embedded in a high-dimensional Euclidean space. The objective of the nonlinear dimensionality reduction method is to accurately extract the low-dimensional surface from the high-dimensional Euclidean space. Figure 1(a) shows a data set consisting of points sampled uniformly at random from a two-dimensional surface in $\mathbb{R}^3$. When a nonlinear dimensionality reduction method is applied to such a data set, the result-



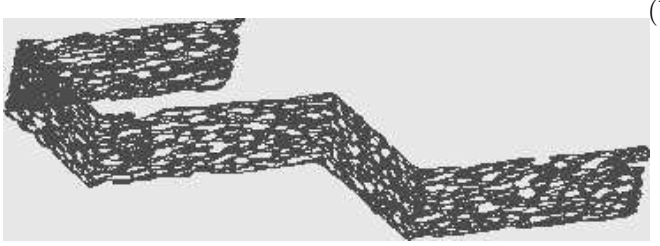
(a) A two-dimensional surface in $\mathbb{R}^3$



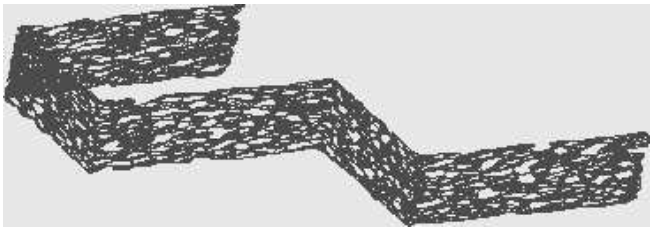
(b) Extracted surface computed by I-Exact



(c) Extracted surface computed by I-hcDPES

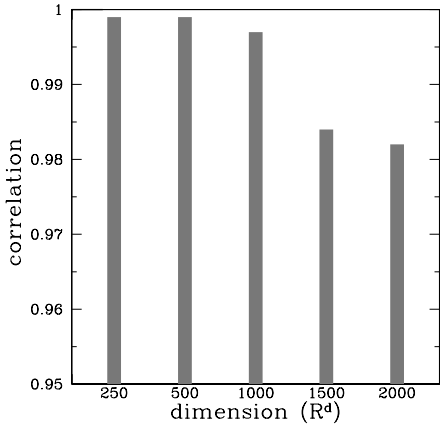


(d) Exact nearest-neighbors graph



(e) Approximate nearest-neighbors graph by hcDPES

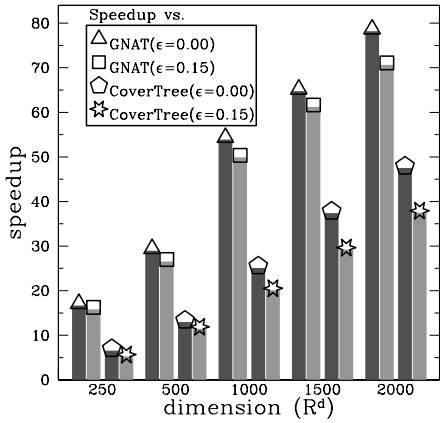
Figure 1: The extraction of a two-dimensional surface in $\mathbb{R}^3$ by I-Exact and I-hcDPES shown in (a). A comparison of (b) and (c) indicates that the embeddings computed by I-Exact and I-hcDPES are almost identical. The reason is that the approximate nearest-neighbors graph computed by hcDPES (shown in (e)) is remarkably similar to the exact nearest-neighbors graph (shown in (d)).



(a) Accuracy of extracted embeddings

	dimension ($\mathbb{R}^d$)				
error	250	500	1000	1500	2000
$RFD_{0.00}$	0.056	0.067	0.095	0.055	0.056
$RFD_{0.02}$	0.015	0.021	0.034	0.016	0.016
$RFD_{0.05}$	0.003	0.004	0.008	0.003	0.004
$RFD_{0.07}$	0.001	0.001	0.003	0.001	0.002
$RFD_{0.10}$	0.000	0.000	0.001	0.000	0.000
$RFD_{0.11}$	0.000	0.000	0.000	0.000	0.000
RDE	0.002	0.002	0.003	0.002	0.002

(b) Accuracy of approximating the nearest-neighbors graph



(c) Computational efficiency

Figure 2: Accuracy and efficiency of hcDPES on extracting low-dimensional surfaces from high-dimensional Euclidean spaces. (a) The high correlation values indicate that such embeddings are practically the same. (b) The accuracy of hcDPES is due to the approximation of the nearest-neighbors graph, which is remarkably similar to the exact nearest-neighbors graph, as indicated by the small RFD and RDE errors. (c) hcDPES offers significant computational speedups over other methods.

ing embedding in $\mathbb{R}^2$ should resemble a rectangle, since the underlying structure of the surface is rectangular. Figure 1(b) indicates that I-Exact accurately extracts the two-dimensional rectangular surface. Figure 1(c) shows the embedding in $\mathbb{R}^2$ as computed by I-hcDPES. A qualitative comparison of Figure 1(b) and (c) reveals that the embeddings computed by I-Exact and I-hcDPES are practically indistinguishable. As it will be quantified later in the section, the reason for the accuracy of I-hcDPES is that the approximate nearest-neighbors graph is almost identical to the exact nearest-neighbors graph, shown in Figure 1(d) and (e).

We compare the results obtained by I-Exact and I-hcDPES when applied to data sets consisting of points sampled uniformly at random from m-dimensional surfaces in $\mathbb{R}^d$. We progressively increase the dimension d and measure the impact on the accuracy and efficiency of I-hcDPES. Each m-dimensional surface in our test case can be conceptually thought of as an m-dimensional hypercube bent at several parts, as illustrated in Figure 1(a). Each m-dimensional surface consists of $\ell = 5$ parts and is constructed by selecting uniformly at random in $\mathbb{R}^d$ an origin o and $m+\ell$ unit vectors $U = \{u_1, \ldots, u_{m+\ell}\}$ such that $U_i = \{u_1, \ldots, u_{m-1}, u_i\}$ is linearly independent for each $m \leq i \leq \ell$. The j-th part of the surface, $1 \leq j \leq \ell$, contains all the points in

$$\{o + \sum_{\ell=0}^{j-1} u_{m+\ell} + \alpha_m^j u_{m+j} + \sum_{i=1}^{m-1} \alpha_i^j u_i : 0 \leq \alpha_i^j, \alpha_m^j \leq 1\}.$$

The corresponding data set consists of 10000 points obtained by sampling uniformly at random 2000 points on each part of the surface. The distance between two data points is defined as the Euclidean distance in $\mathbb{R}^d$.

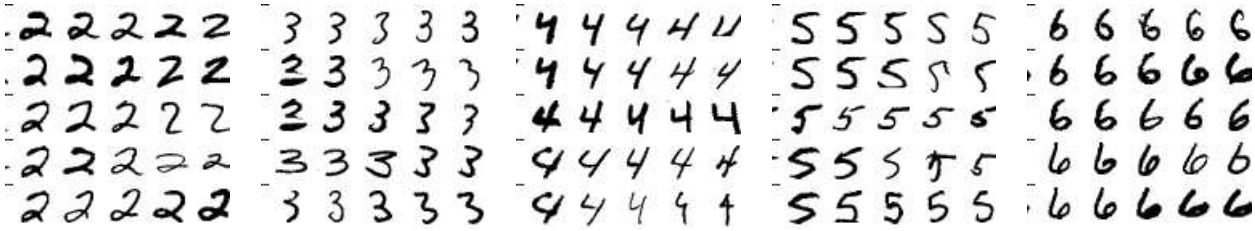
We performed tests on data sets corresponding to 20-dimensional surfaces in $\mathbb{R}^{250}$, $\mathbb{R}^{500}$, $\mathbb{R}^{1000}$, $\mathbb{R}^{1500}$, and $\mathbb{R}^{2000}$. Figure 2(a) indicates the correlation between the 20-dimensional embeddings obtained by I-Exact and I-hcDPES. The high correlation values indicate that the surfaces extracted by I-Exact and I-hcDPES are almost identical. The correlation remains above 0.99 not only when $d = 250$ but even as d is progressively increased to $d = 2000$. The accuracy of I-hcDPES is due to the high-quality approximation of the nearest-neighbors graph computed by hcDPES. As shown in Figure 2(b), the RFD and RDE errors are small. In fact, $\text{RFD}_{0.0}$ is less than 0.1 in all cases. This indicates that the approximate nearest neighbors include above 90% of the exact nearest neighbors. Furthermore, the RFD_ϵ error quickly drops to zero when slightly larger values of ϵ are considered. As shown in Figure 2(b), in all cases $\text{RFD}_\epsilon = 0.0$ for $\epsilon \geq 0.11$. The small values of RFD that even those few approximate nearest neighbors computed by hcDPES that are not the same as the exact nearest neighbors are in fact close to the exact nearest neighbors, i.e., within a 0.11-ball. This is also indicated by the small RFD error, which shows that differences in distances between approximate and exact nearest neighbors are negligible. As a result, the proximity graph computed by hcDPES accurately approximates the nearest-neighbors graph and thus enables I-hcDPES to maintain the reliability of I-Exact.

Figure 2(c) indicates the speedup of I-hcDPES when compared to Isomap using other efficient exact or approximate nearest-neighbors methods to compute the proximity graph. As illustrated in Figure 2(b), using hcDPES to compute the approximate nearest-neighbors graph makes Isomap considerably more efficient. The computational speedup obtained by I-hcDPES increases rapidly as the dimension d increases. We observe that for $d = 2000$, I-hcDPES yields computational speedups of over 75 times when compared to Isomap using GNAT to compute the exact nearest-neighbors graph, i.e., $\text{I-GNAT}_{\epsilon=0.0}$. Even when GNAT is used to compute an approximate nearest-neighbors graph, the computational speedup obtained by I-hcDPES is still above 70 times. A comparison with I-CoverTree, which generally performed better than I-GNAT, indicates that I-hcDPES still offers significant computational improvements. The computational speedups obtained by I-hcDPES are above 45 and above 35 times when CoverTree is used to compute the exact nearest-neighbors graph or an approximate nearest-neighbors graph, respectively. We also note that when GNAT and CoverTree are used to approximate the nearest-neighbors graph, the value of ϵ is set to 0.15. Recall that the approximate nearest neighbors computed by hcDPES were all within a 0.11-ball from the exact nearest neighbors. That is, approximate nearest-neighbors computed by hcDPES are not only of better or comparable quality to approximate nearest-neighbors computed by methods that rely on specifications of ϵ-balls, but are also computed at a fraction of the computational cost.

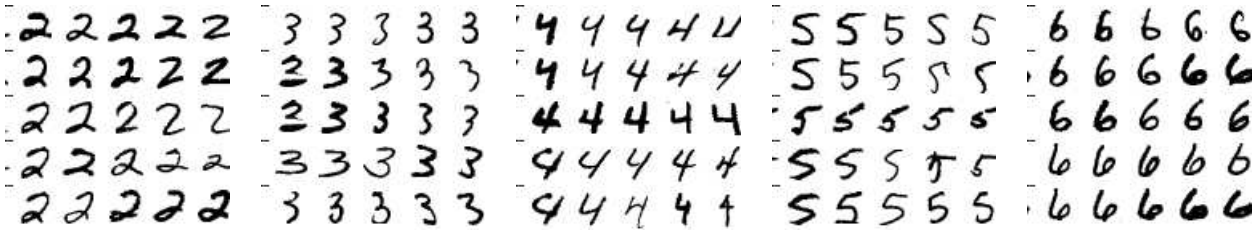
Figure 2 indicates that by using hcDPES to approximate the nearest-neighbors graph, Isomap accurately extracts low-dimensional surfaces from high-dimensional Euclidean spaces at a fraction of the computational cost required if other exact or approximate methods were used to compute the proximity graph.

4.3 Test Case: Image Processing

We also tested the accuracy and efficiency of hcDPES on images of handwritten digits. The handwritten digits exhibit highly nonlinear features making them suitable for testing nonlinear reduction methods. We use the MNIST database [32], which consists of approximately 6000 grayscale images for each digit. Each image is

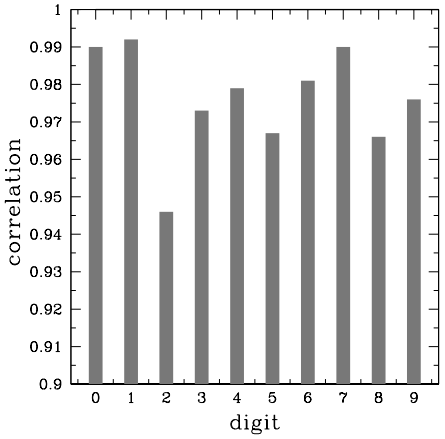


(a) Interpolated images obtained by I-Exact

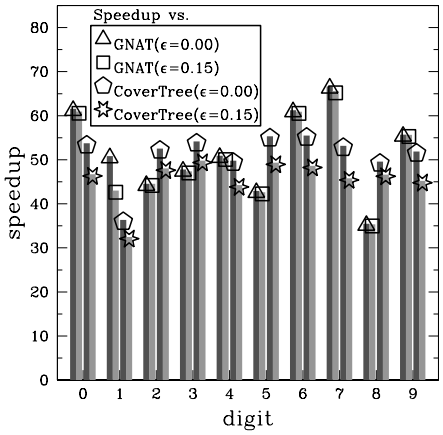


(b) Interpolated images obtained by I-hcDPES

Figure 3: Nonlinear features captured by I-Exact and I-hcDPES on handwritten images of digits.



(a) Accuracy of extracted embeddings



(c) Computational efficiency

(b) Accuracy of approximating the nearest-neighbors graph

	digit									
error	0	1	2	3	4	5	6	7	8	9
$RFD_{0.00}$	0.068	0.080	0.094	0.084	0.066	0.079	0.074	0.099	0.088	0.086
$RFD_{0.02}$	0.017	0.034	0.027	0.018	0.014	0.017	0.021	0.036	0.019	0.027
$RFD_{0.05}$	0.002	0.009	0.003	0.001	0.001	0.001	0.003	0.008	0.001	0.004
$RFD_{0.07}$	0.001	0.004	0.001	0.000	0.000	0.000	0.001	0.003	0.000	0.001
$RFD_{0.10}$	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.000
$RFD_{0.11}$	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
RDE	0.003	0.005	0.003	0.003	0.002	0.003	0.003	0.005	0.003	0.003

Figure 4: Accuracy and efficiency of hcDPES on handwritten images of digits. (a) The high correlation values indicate that I-hcDPES is practically as accurate as I-Exact. (b) The approximate nearest-neighbors graph computed by hcDPES is remarkably similar to the exact nearest-neighbors graph, as evidenced by the small RFD and RDE errors. (c) Comparisons with other exact and approximate methods reveal that I-hcDPES is significantly more efficient.

28×28 pixels. We rasterize each image and represent it as a point in $\mathbb{R}^{784}$. The distance between two images is then defined as the Euclidean distance in $\mathbb{R}^{784}$ between the corresponding points. In each experiment, we obtain 8-dimensional embeddings of the images of each digit. Figures 3 and 4 contain a summary of the results.

When a dimensionality reduction method accurately captures the nonlinear features, then linearly interpolating between any two embeddings reveals how the corresponding images transition from one to the other. Let emb(A) and emb(B) denote the embeddings of two images A and B from the data set. An intermediate image from A to B at time $t \in [0, 1]$ is computed as the image C from the data set whose embedding emb(C), according to the Euclidean distance, is the closest to $t * \text{emb}(A) + (1 - t) * \text{emb}(B)$. Figure 3(a) and (b) illustrate the interpolation results for several digits as computed by I-Exact and I-hcDPES, respectively. We observe smooth transitions from one image to the other. As an illustration, in rows 1-3 of digit 2, we see how the knot of digit 2 transitions into a straight segment, a highly nonlinear feature. We also see how a large image scales into a small image while also changing the curvature, as in digit 2 row 4, digit 5 rows 3-4. We observe that although there are some differences in the interpolated images computed by I-Exact and I-hcDPES, e.g., digit 2 rows 1,5, digit 3 row 4, digit 5 row 5, digit 6 row 2, the transitions are smooth in both cases and accurately capture the nonlinear features.

Figure 4 shows results that quantify the qualitative comparisons of Figure 3. In each case, the correlation between embeddings obtained by I-Exact and I-hcDPES is remarkably high, as shown in Figure 4(a). Therefore, quantitatively differences in the embeddings obtained by I-Exact and I-hcDPES are negligible. As it was the case in our first example in §4.2, this is due to the accurate approximation of the nearest-neighbors graph computed by hcDPES, as indicated by the small RFD and RDE errors in Figure 4(b). In fact, above 90% of the approximate nearest neighbors computed by hcDPES are the same as the exact nearest neighbors and all of the approximate nearest neighbors are within a 0.10-ball from the exact nearest neighbors. Furthermore, the RFD error is at most 0.005, further indicating that the approximate nearest-neighbors computed by hcDPES are remarkably close to the exact nearest neighbors. The advantage of hcDPES is that it offers considerable speedups over exact nearest-neighbors methods or approximate nearest-neighbors methods that maintain the same level of accuracy as hcDPES, as shown in Figure 4(c). We observe that I-hcDPES is between 35–65 times more efficient than I-GNAT$_{\epsilon=0.0}$ and 35–55 times more efficient than I-CoverTree$_{\epsilon=0.0}$. Even when I-hcDPES is compared to Isomap using GNAT and CoverTree with $\epsilon = 0.15$ for approximate nearest neighbors, the speedup obtained by I-hcDPES is still high, 35–65 and 32–50 times, respectively.

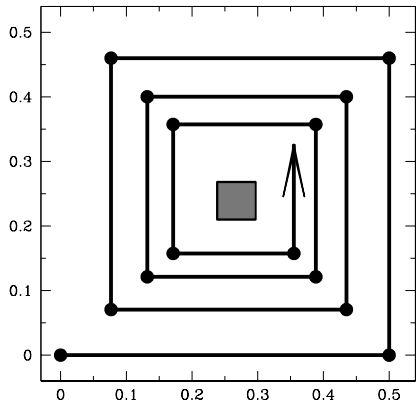


Figure 5: Example of a manipulator. The manipulator consists of many links connected by revolute joints. The objective of the motion planner is to find a sequence of motion that would avoid collisions with the obstacle (gray square) and take the manipulator from its current coiled configuration to a straight-line configuration.

Therefore, as in the first case, by accurately approximating the nearest-neighbors graph, hcDPES significantly reduces the computational bottleneck imposed by the nearest-neighbors graph on nonlinear dimensionality reduction.

4.4 Test Case: Robot Motion Planning

The purpose of this case is to test the efficiency and accuracy of hcDPES when distances between points are non-Euclidean. We select an example from robot motion planning, since it provides highly nonlinear data. The aim of motion planning is to enhance the autonomy of robots by automatically planning motions that are necessary for a robot to carry out a specific task. Motivated by the need for safe urban search and rescuing, recently, considerable attention has been devoted to the development of algorithms for efficiently planning motions of highly articulated and modular robots [45]. Planning motions for such complex robots is a highly nontrivial task due to the large number of joints and modules. The most successful motion planning methods rely on an extensive sampling of the configuration space and the computation of the nearest-neighbors for the generated samples [9]. The resulting nearest-neighbors graph captures the connectivity of the configuration space and is used to plan motions that take the robot from one configuration to another. To illustrate the idea, Figure 5(a) shows a serpentine manipulator in a

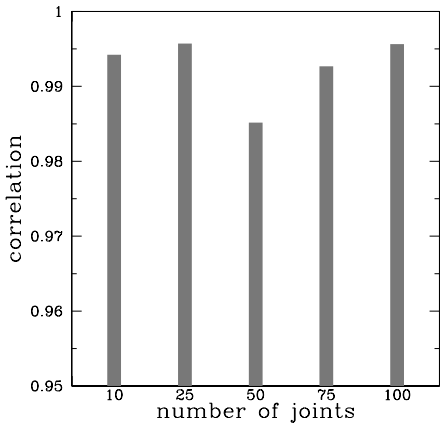
coiled configuration. The manipulator has several revolute joints. The manipulator assumes different configurations depending on the rotations at each joint. The objective of the motion planner is to find a path from the coiled configuration that the manipulator is currently in to a straight-line configuration where the manipulator is completely stretched out. The path should avoid self-collisions and collisions with the obstacle, which is represented in Figure 5 by a square.

We use data sets representing 10000 collision-free configurations of the manipulator. We obtain different data sets by increasing the number of joints of the manipulator and using the motion planner PRM (Probabilistic RoadMap) [28] to generate collision-free configurations. Each configuration indicates the rotation at each joint. Let $v(c) = \{p_1(c), \ldots, p_d(c)\}$ denote the position of each of the d joints after each joint is rotated by the value specified by configuration c. The distance between two configurations c_1 and c_2 is defined as the Euclidean distance between $v(c_1)$ and $v(c_2)$. Note that due to the rotations this distance is non-Euclidean. Although the dimensionality of the configuration space of the manipulator is equal to the number of joints, the underlying structure of each data set is of much lower dimensionality. This is due to the fact that the manipulator is hyper redundant since it is constrained to operate in a 2-dimensional plane.

Figure 6 contains a summary of the results where each data set is embedded in $\mathbb{R}^2$. We observe in Figure 6(a) that the correlation between the embeddings obtained by I-Exact and I-hcDPES is considerably high. This indicates that even in the case of non-Euclidean distance metrics, I-hcDPES is as accurate as I-Exact. The small RFD and RDE errors, shown in Figure 6(b), indicate that differences between approximate nearest neighbors computed by hcDPES and exact nearest neighbors are negligible. As a result, the embeddings emerging from the application of I-hcDPES and I-Exact are almost identical. Figure 6(c) indicates the speedup obtained by I-hcDPES. Figure 6(c) reveals that I-hcDPES is significantly more efficient than when Isomap uses other methods to compute or approximate the nearest-neighbors graph. Therefore, even when non-Euclidean metrics define the distance between any two points the approximate nearest-neighbors graph computed by hcDPES can be used for nonlinear dimensionality reduction without any considerable loss of accuracy. Furthermore, the computational time for nonlinear dimensionality reduction is significantly reduced.

5 Discussion

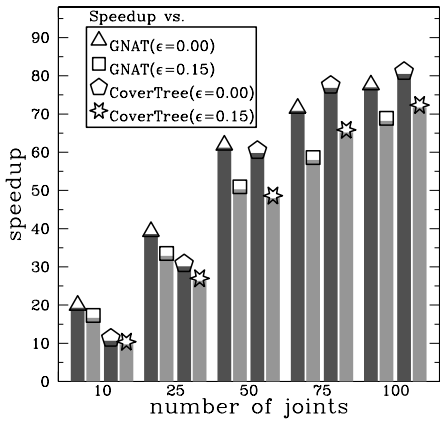
We have developed an efficient approximate nearest-neighbors method, hcDPES, that can be used to sig-



(a) Accuracy of extracted embeddings

	number of joints				
error	10	25	50	75	100
$RFD_{0.00}$	0.075	0.080	0.095	0.098	0.095
$RFD_{0.02}$	0.038	0.034	0.041	0.040	0.040
$RFD_{0.05}$	0.013	0.008	0.011	0.010	0.009
$RFD_{0.07}$	0.006	0.004	0.005	0.003	0.004
$RFD_{0.10}$	0.001	0.001	0.002	0.001	0.001
$RFD_{0.11}$	0.000	0.000	0.000	0.000	0.000
RDE	0.006	0.005	0.006	0.006	0.006

(b) Accuracy of the approximate nearest-neighbors graph



(c) Computational efficiency

Figure 6: Accuracy and efficiency of hcDPES on a robot motion planning test case. The robot motion planning data is highly nonlinear and distances are defined by complex non-Euclidean metrics. (a) Embeddings extracted by I-Exact and I-hcDPES are practically the same, as evidenced by the high correlation. (b) The small RFD and RDE errors indicate that hcDPES accurately approximates the nearest-neighbors graph. (c) hcDPES considerably improves the computational efficiency of nonlinear dimensionality reduction by approximating the nearest-neighbors graph at a fraction of the computational cost required by other methods.

nificantly reduce the computational bottleneck of nonlinear dimensionality reduction methods that rely on the nearest-neighbors graph. The approximation of the nearest-neighbors graph is based on the idea of projecting the data set onto a low-dimensional Euclidean space. The projection uses distances from each point in the data set to a set of carefully selected pivots. The objective of the pivot selection strategy is to maximize the correlation of distances between points in the data set and their corresponding projections. The overall effect of the pivot selection strategy and the projection is the efficient computation of accurate approximate nearest neighbors, as indicated in §4.

The accuracy and efficiency of hcDPES have been demonstrated using test cases from different application areas. We have also quantitatively analyzed the performance of hcDPES when non-Euclidean metrics define distances between points in the data sets. The results indicate that the embeddings emerging from the application of nonlinear dimensionality reduction using the exact nearest-neighbors or the approximate nearest-neighbors computed by hcDPES are practically indistinguishable, as evidenced by the remarkably high correlation values. The advantage of using hcDPES is that it significantly reduced the computational time required for nonlinear dimensionality reduction.

By significantly reducing the bottleneck imposed by the computation of the nearest-neighbors graph, hcDPES can be used by nonlinear dimensionality reduction methods to extract low-dimensional embeddings of considerably large and high-dimensional data sets where distances between points are defined by complex metrics. Potential applications which we are currently addressing include the analysis of molecular motions of proteins and the behavior of motion planning algorithms on high-dimensional problems.

References

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, *Basic local alignment search tool*, Journal of Molecular Biology, 215 (1990), pp. 403–410.

[2] S. Arya, D. M. Mount, and S. Nathan, *An optimal algorithm for approximate nearest neighbor searching in fixed dimensions*, Journal of the ACM, 45 (1998), pp. 891–923.

[3] S. Berchtold, D. Keim, and H.-P. Kriegel, *The X-tree: An index structure for high-dimensional data*, in International Conference on VLDB, Mumbai, India, 1996, pp. 28–39.

[4] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, *When is "nearest neighbor" meaningful?*, Lecture Notes in Computer Science, 1540 (1999), pp. 217–235.

[5] A. Beygelzimer, S. Kakade, and J. Langford, *Cover trees for nearest neighbor*, in Interantional Conference on Machince Learning, Pittsburgh, Pennsylvania, 2006, pp. 97–104.

[6] J. Bi, K. P. Bennett, M. Embrechts, C. M. Breneman, M. Song, and I. G. Elisseeff, *Dimensionality reduction via sparse support vector machines*, Journal of Machine Learning Research, 3 (2003), pp. 1229–1243.

[7] A. Borodin, R. Ostrovsky, and Y. Rabani, *Lower bounds for high dimensional nearest neighbor search and related problems*, in ACM Symposium on Theory of Computing, Atlanta, Georgia, 1999, pp. 312–321.

[8] S. Brin, *Near neighbor search in large metric spaces*, in International Conference on VLDB, San Francisco, California, 1995, pp. 574–584.

[9] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, MIT Press, Cambridge, Massachusetts, 2005.

[10] P. Ciaccia, M. Patella, and P. Zezula, *M-tree: An efficient access method for similarity search in metric spaces*, in International Conference on VLDB, Athens, Greece, 1997, pp. 426–435.

[11] K. L. Clarkson, *Nearest neighbor queries in metric spaces*, Discrete and Computational Geometry, 22 (1999), pp. 63–93.

[12] T. Cox and M. Cox, *Multidimensional Scaling*, Chapman & Hall, 2nd ed., 2000.

[13] B. Cui, B. C. Ooi, J. Su, and K.-L. Tan, *Contorting high-dimensional data for efficient main memory knn processing*, in ACM SIGMOD International Conference on Management of Data, San Diego, California, 2003, pp. 479–490.

[14] B. Cui, H. T. Shen, J. Shen, and K.-L. Tan, *Exploring bit-difference for approximate knn search in high-dimensional databases*, in Australasian Database Conference, Newcastle, Australia, 2005, pp. 165–174.

[15] P. Das, M. Moll, H. Stamati, L. E. Kavraki, and C. Clementi, *Low-dimensional free energy landscapes of protein folding reactions by nonlinear dimensionality reduction*, Proceedings of the National Academy of Science USA, 103 (2006), pp. 9885–9890.

[16] B. V. Dasarathy, *Nearest-neighbor approaches*, in Handbook of Data Mining and Knowledge Discovery, W. Klosgen, J. M. Zytkow, and J. Zyt, eds., Oxford University Press, New York, New York, 2002, pp. 88–298.

[17] V. de Silva and J. Tenenbaum, *Global versus local methods in nonlinear dimensionality reduction*, in Advances in NIPS, S. Becker, S. Thrun, and K. Obermayer, eds., MIT Press, Cambridge, MA, 2002, pp. 705–712.

[18] D. Donoho and C. Grimes, *Hessian eigenmaps: new locally linear embedding techniques for high-dimensional data*, Proceedings of the National Academy of Science, 100 (2003), pp. 5591–5596.

[19] J. H. Freidman, J. L. Bentley, and R. A. Finkel, *An algorithm for finding best matches in logarithmic expected time*, ACM Transactions on Mathematical Software, 3 (1977), pp. 209–226.

[20] V. Gaede and O. Günther, *Multidimensional access methods*, ACM Computing Surveys, 30 (1998), pp. 170–231.

[21] X. Geng, D.-C. Zhan, and Z.-H. Zhou, *Supervised nonlinear dimensionality reduction for visualization and classification*, IEEE Transactions on Systems, Man and Cybernetics, 35 (2005), pp. 1098–1107.

[22] H. He, W. Graco, and X. Yao, *Application of genetic algorithm and k-nearest neighbour method in medical fraud detection*, in Asia-Pacic Conference on Simulated Evolution and Learning, 1998, pp. 74–81.

[23] A. Hinneburg, C. C. Aggarwal, and D. A. Keim, *What is the nearest neighbor in high dimensional spaces?*, in International Conference on VLDB, Cairo, Egypt, 2000, pp. 506–515.

[24] A. Hyvärinent, J. Karhunen, and E. Oja, *Independent Component Analysis*, John Wiley & Sons, 2001.

[25] P. Indyk, *Nearest neighbors in high-dimensional spaces*, in Handbook of Discrete and Comp. Geometry, J. E. Goodman and J. O'Rourke, eds., CRC Press, 2004, pp. 177–196.

[26] H. V. Jagadish, B. C. Ooi, K.-L. Tan, C. Yu, and R. Zhang, *iDistance: An adaptive B+-tree based indexing method for nearest neighbor search*, ACM Transactions on Database Systems, 30 (2005), pp. 364–397.

[27] I. Jolliffe, *Principal Components Analysis*, Springer-Verlag, New York, 1986.

[28] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, *Probabilistic roadmaps for path planning in high-dimensional configuration spaces*, IEEE Transactions on Robotics and Automation, 12 (1996), pp. 566–580.

[29] F. Korn, B.-U. Pagel, and C. Faloutsos, *On the 'dimensionality curse' and the 'self-similarity blessing'*, IEEE Transactions on Knowledge and Data Engineering, 13 (2001), pp. 96–111.

[30] W.-S. Ku, R. Zimmermann, H. Wang, and C.-N. Wan, *Adaptive nearest neighbor queries in travel time networks*, in ACM International Workshop on Geographic Information Systems, Bremen, Germany, 2005, pp. 210–219.

[31] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, *Efficient search for approximate nearest neighbor in high dimensional spaces*, SIAM Journal of Computing, 30 (2000), pp. 457–474.

[32] Y. LeCun, *MNIST handwritten digit database*. http://www.research.att.com/yann/ocr/mnist/.

[33] K.-I. Lin, H. Jagadish, and C. Faloutsos, *The TV-tree: An index structure for high-dimensional data*, International Journal on VLDB, 3 (1994), pp. 517–549.

[34] T. Liu, A. W. Moore, A. Gray, and K. Yang, *An investigation of practical approximate nearest neighbor algorithms*, in Advances in NIPS, L. K. Saul, Y. Weiss, and L. Bottou, eds., MIT Press, Cambridge, MA, 2005, pp. 825–832.

[35] I. Lotan and F. Schwarzer, *Approximation of protein structure for fast similarity measures*, Journal of Computational Biology, 11 (2004), pp. 299–317.

[36] S. McGinnis and T. L. Madden, *BLAST: at the core of a powerful and diverse set of sequence analysis tools*, Nucleic Acids Research, 32 (2004), pp. W20–W25.

[37] A. N. Papadopoulos and Y. Manolopoulos, *Nearest Neighbor Search: A Database Perspective*, Series in Computer Science, Springer Verlag, Berlin, Germany, 2005.

[38] E. Plaku and L. E. Kavraki, *Quantitative analysis of nearest-neighbors search in high-dimensional sampling-based motion planning*, in International Workshop on Algorithmic Foundations of Robotics, New York, NY, 2006. In press. Available at http://www.wafr.org/papers/p31.pdf.

[39] S. Roweis and L. Saul, *Nonlinear dimensionality reduction by Locally Linear Embedding*, Science, 290 (2000), pp. 2323–2326.

[40] S. J. Russell and P. Norvig, *Artificial intelligence: A modern approach*, (2002).

[41] L. K. Saul and S. T. Roweis, *Think globally, fit locally: unsupervised learning of low dimensional manifolds*, Journal of Machine Learning Research, 4 (2003), pp. 119–155.

[42] J. Tenenbaum, V. de Silva, and J. Langford, *A global geometric framework for nonlinear dimensionality reduction*, Science, 290 (2000), pp. 2319–2323.

[43] R. Weber, H.-J. Schek, and S. Blott, *A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces*, in International Conference on VLDB, New York, New York, 1998, pp. 194–205.

[44] K. Q. Weinberger, F. Sha, and L. K. Saul, *Learning a kernel matrix for nonlinear dimensionality reduction*, in International Conference on Machine Learning, 2004, pp. 106–113.

[45] A. Wolf, H. B. B. Jr., R. Casciola, A. Costa, M. Schwerin, E. Shammas, and H. Choset, *A mobile hyper redundant mechanism for search and rescue tasks*, in IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, Nevada, 2003, pp. 2889–2895.

[46] L. Yang, *Distance-preserving projection of high-dimensional data for nonlinear dimensionality reduction*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26 (2004), pp. 1243–1246.

[47] R. Zhang, P. Kalnis, B. C. Ooi, and K.-L. Tan, *Generalized multidimensional data mapping and query processing*, ACM Transactions on Database Systems, 30 (2005), pp. 661–697.

[48] Z. Zhang and H. Zha, *Principal manifolds and nonlinear dimension reduction via local tangent space alignment*, SIAM Journal of Scientific Computing, 26 (2004), pp. 313–338.