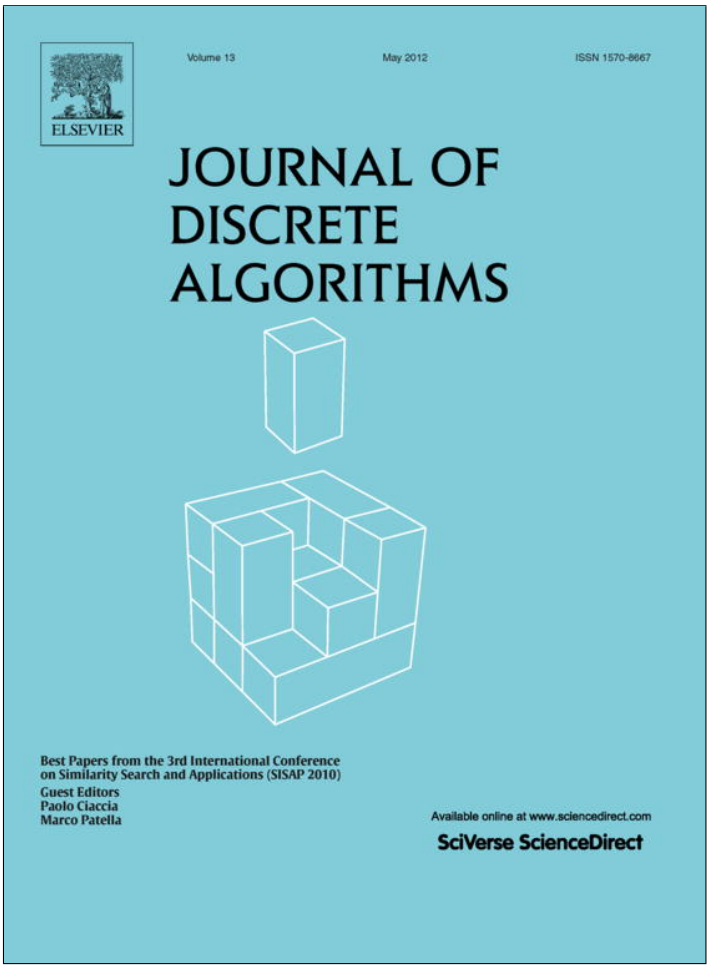

Journal of Discrete Algorithms 13 (2012) 32–46

Contents lists available at SciVerse ScienceDirect

Journal of Discrete Algorithms

www.elsevier.com/locate/jda

Pivot selection: Dimension reduction for distance-based indexing

Rui Mao [a,*], Willard L. Miranker [b,1], Daniel P. Miranker [c,2]

[a] Shenzhen University, 3688 Nanhai Rd., Office Tower #342, Shenzhen, Guangdong, 518060, China
[b] Yale University, 227 Church Street, PH2E, New Haven, CT 06510, USA
[c] University of Texas at Austin, 1 University station, C0500, Austin, TX 78712, USA

ARTICLE INFO

Article history:
Available online 29 October 2011

Keywords:
Similarity query
Metric space
Dimension reduction
Intrinsic dimension
Pivot selection
Pivot space model

ABSTRACT

Distance-based indexing exploits only the triangle inequality to answer similarity queries in metric spaces. Lacking coordinate structure, mathematical tools in R^n can only be applied indirectly, making it difficult to theoretically study metric-space indexing. Toward solving this problem, a common algorithmic step is to select a small number of special points, called *pivots*, and map the data objects to a low-dimensional space, one dimension for each pivot, where each dimension represents the distances of a pivot to the data objects. We formalize a "pivot space model" where all the data objects are used as pivots such that data is mapped from metric space to R^n, preserving all the pairwise distances under L^∞. With this model, it can be shown that the indexing problem in metric space can be equivalently studied in R^n. Further, we show the necessity of dimension reduction for R^n and that the only effective form of dimension reduction is to select existing dimensions, i.e. pivot selection. The coordinate structure of R^n makes the application of many mathematical tools possible. In particular, Principle Component Analysis (PCA) is incorporated into a heuristic method for pivot selection and shown to be effective over a large range of workloads. We also show that PCA can be used to reliably measure the intrinsic dimension of a metric space.

1. Introduction

Given a database S and a distance function d, a similarity query comprises finding all the data points in S that are close in distance to the query object. Both range queries and nearest neighbor queries are of general interest [7,12,30,34]. It has been shown that k-nearest neighbor queries are systematic applications of range queries [7]. This presentation focuses on range queries.

Range query (q, r). Given a query object q and a similarity measurement d, seek all data objects x within distance r to q, i.e., $d(q, x) \leqslant r$. r is called the radius [7,15,26].

Intuitively, a range query (q, r) seeks all points contained in a ball centered at q with radius r in the metric space. This ball is called the *query ball*.

Although multi-dimensional indexing methods supporting similarity queries have been intensively studied, these approaches are limited to data that can be represented in R^n and to the Euclidean distance function [14]. Distance-based

* Corresponding author. Fax: +86 755 2655 8644 3.
E-mail addresses: mao@szu.edu.cn (R. Mao), miranker@cs.yale.edu (W.L. Miranker), miranker@cs.utexas.edu (D.P. Miranker).
[1] Fax: +1 203 432 7226.
[2] Fax: +1 512 471 9541.

1570-8667/$ – see front matter
doi:10.1016/j.jda.2011.10.004

indexing [7,15,26], also known as metric-space indexing, does not make use of data domain information. The available information about the data is derived solely from an oracle that computes the distance between pairs of objects. The only requirement is that the distance function be a metric. The data set can be clustered and a data structure compiled off-line. During an on-line search, the triangle inequality enables the elimination of clusters of data as possible solutions. As a consequence, on-line similarity queries may be quickly computed.

This, so-called, "black-box" model is advantageous for any application where the data cannot be effectively mapped to feature vectors. Even though one is not inclined to use distance-based indexing for multi-dimensional problems, distance-based indexing subsumes multi-dimensional indexing, enabling a uniform programming model for problems that can be recast into metric spaces.

The generality of the approach is also its challenge. What makes distance-based indexing difficult is the lack of coordinate structure. The dimension of the data is not explicit. As a result, mathematical tools developed for R^n are not directly applicable to distance-based problems.

A common method is to first map the metric space to R^k, where k is less than the number of points, and then to answer the similarity query with geometric methods that have been developed for R^k. To summarize and formalize this method, we propose the *pivot space model*. With this model, there are three steps to answer a similarity query in a general metric space. In step 1, the data in the metric space is mapped into a subset of R^k, called the *pivot space*. Then, a region in R^k, named the *query cube* containing all the query results is determined. In step 2, multi-dimensional methods are applied to retrieve all the points in the pivot space that fall into the query cube. In step 3, all the points retrieved in step 2 are compared with the query object to remove the false positives (Section 3).

It has been shown that any finite metric space of n points can be mapped isometrically into R^n, one dimension for each point, using the L^∞ norm [20] (let $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$, then $L^\infty(A, B) = \max\{|a_i - b_i|,\ 1 \leqslant i \leqslant n\}$). In our model we call the image of a dataset so constructed the *complete pivot space*. It will follow that the evaluation of a query in the complete pivot space may be accomplished using multi-dimensional indexing methods but will require a calculation for each of the n dimensions. Since n is the size of the database, dimension reduction is necessary.

The necessity of dimension reduction is also supported from a practical perspective: the memory issue. If the complete pivot space is directly considered, $O(n^2)$ space is required to store all the distances, which is impossible for large real datasets. The memory issue will not be further discussed since it is not the focus of this paper.

Next, we prove that any dimension reduction technique that creates new dimensions will still require a calculation for each of the n dimensions. Therefore, the only effective form of dimension reduction for the complete pivot space is one that selects a subset of the existing dimensions. This form is called pivot selection.

To demonstrate how a general dimension reduction technique can be applied to distance-based problems, we design a pivot selection algorithm based on Principal Component Analysis (PCA), a popular dimension reduction technique for multi-dimensional spaces (Section 5). The basic idea is to select existing dimensions that would best approximate the eigenvectors computed by PCA. Empirical results show that our pivot selection heuristic outperforms a deterministic corner-selection and is comparable to a well-accepted non-deterministic incremental selection heuristic [5].

Several analytic studies of high-dimensional query problems have concluded that the indexability of data is sensitive to its dimension and that the performance of tree-based indexing on fixed size vector data sets degrades to a sequential scan as the dimension of the data increases [2,11,27,31]. Based on the pivot space model, we propose to estimate the intrinsic dimension of a metric space dataset based on the eigenvalues associated with the principal components of the data's image in R^n. By using controlled dataset, empirical results show that our method gives more accurate estimates of intrinsic dimension (Section 6).

2. Related work

The development of distance-based indexing algorithms is commonly decomposed into to two sub-problems, pivot selection and partitioning methods. Papers that focus on partitioning methods dominate the literature. Partitioning methods fall into three categories, hyper-plane partitioning, vantage-points and bounding spheres. These are well discussed in a pair of surveys and a text [7,15,26].

Only a few methods have been investigated for pivot selection [3–5,18,19,33]. Bustos et al. have published the most recent work. They exploit sampling and seek to choose a set of pivots should maximize the mean and minimize the variance of the pairwise L^∞ distances of the data in the pivot space. Their heuristic selects pivots that maximize the mean [5]. Further, they argue that good pivots are usually outliers, but that the reverse is not true. The Metric Tree (M-tree) [9] bulkload algorithm selects pivots randomly but does not use any sampling [8]. However, results show that different pivots can have dramatically different impacts (Section 4.3). SA-tree starts from a random point and selects the centers of neighboring cells of a Voronoi diagram as pivots [22].

Distance-based indexing algorithms also often use the farthest-first-traversal (FFT) k-center clustering algorithm to choose pivots. It is a fast and convenient way to identify corners, or outliers. FFT minimizes the maximum cluster diameter and gives a result at most twice the optimal diameter [16]. Its time and space complexities are both $O(n)$, where the number of clusters is considered as a constant. The use of FFT is based on Yianilos' observations made of uniformly distributed points in the unit square [33]. He proposes to select the corners of the data set as pivots for his Vantage Point Tree (VPT). Multi-Vantage Point Tree (MVPT) [3], an extension of VPT, selects multiple corners as pivots.

Brin, in GNAT trees, suggests index trees be constructed with a variable number of pivots such that a larger number of pivots are used for higher populated clusters to maintain tree balance [4].

What determines the indexability of data is not the dimension of the domain where the data is represented, but the intrinsic dimension, a property of the distribution of the data. The intrinsic dimension is invariant no matter how the data is represented as long as there is no distortion of the pairwise distances. Many authors have tried to define and quantify intrinsic dimension [7,10,18]. We consider the intrinsic dimension to be the dimension of a Euclidean space into which the data can be embedded with small distortion.

Prior estimations of intrinsic dimension relate the distribution of distances among pairs of elements in an arbitrary metric space to the distribution of data in the vector space. The following are two existing methods.

Method 1. Chavez et al. [7] define the intrinsic dimension of a metric space as $\rho = \mu^2/2\sigma^2$, where μ and σ^2 are the mean and variance of the pairwise distances.

Method 2. Another approach [18,29] measures how the volume of a hyper-ball, that is, the number of points contained in it, changes with respect to the radius. Let r be the range query radius, and n be the average number of range query results. Then, linear regression can be performed on the logarithm of n and r, and the resulting slope coefficient is an estimate of the intrinsic dimension.

Method 1 is computationally simple, while its intuitive meaning is not clear. Ramakrishnan et al. use similar forms to that of Method 1 to determine the stability of workloads [2,27]. Method 2 is a variation of the Minkowski fractal dimension [10,13]. It is limited by values of r and the assumption of uniform distribution.

3. Pivot space model

Due to the lack of coordinate structure in general metric spaces, a widely used approach is to map the general metric space into R^n, and then to respond to the query with multi-dimensional indexing. In this section, we summarize this approach and propose the pivot space model. A theorem is given showing that the same pivot space can be produced from a dataset in a metric space, or a dataset in R^n.

3.1. General steps

In this section, let R^n denote a general real coordinate space of dimension n. There are three steps.

Step 1.

(1) Map the data into R^n.
(2) Map the query object into R^n.
(3) Determine a region in R^n that completely covers the range query ball.

Almost all existing distance-based indexing methods proceed via pivot selection to map the data into R^n, i.e. to select some particular point in the database as pivots and to represent each data point by its distances to the pivots. We name the image so obtained the *pivot space*. Given a similarity query, the distances between the query object and pivots are determined. This essentially maps the query object into the pivot space.

The shape of the image of the query ball of a range query (q, r) in a general metric space is not clear in a pivot space. However, it can be proved, from the triangle inequality, that the image of the query ball is completely covered by a hypercube of edge length $2r$ in the pivot space [7], which is actually a ball of radius r in the new metric space specified by the pivot space and the L^∞ distance. We call the hypercube the *query cube*. Fig. 1 shows an example where 2 pivots are selected. All the points in the query ball are mapped into the query square in the 2-d pivot space plane. Points outside of the square can be discarded using the triangle inequality while points within the square cannot be discarded. To answer the query, one first retrieves all the points in the square. Then their distances to q are computed directly to determine whether their pre-images are in the query ball.

Step 2. Exploit multi-dimensional techniques to retrieve all the points in the region determined in Step 1.

The basic idea of Step 2 is divide-and-conquer based on the data coordinates. Many partition methods in multi-dimensional indexing can be applied here. For example, MVPT's partition is similar to a k-d tree [1].

Step 3. For each point retrieved in Step 2, compute its distance to the query object to remove false positives.

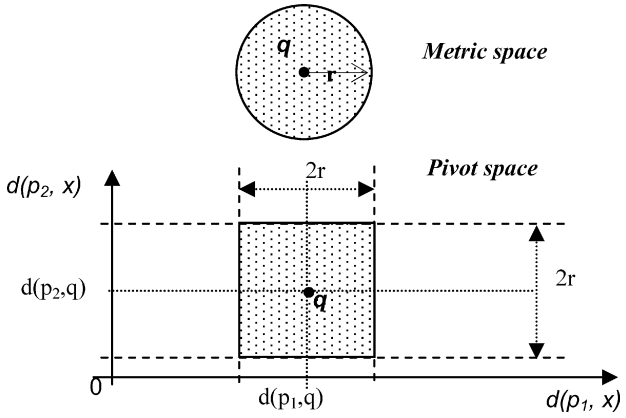

Fig. 1. The query ball of a range query in a general metric space is covered by a square in the pivot space.

	p_1^p	p_2^p	...	p_k^p
x_1^p	$d(x_1, p_1)$	$d(x_1, p_2)$	...	$d(x_1, p_k)$
x_2^p	$d(x_2, p_1)$	$d(x_2, p_2)$	...	$d(x_2, p_k)$
...	...	...	...	...
...	...	...	...	...
x_n^p	$d(x_n, p_1)$	$d(x_n, p_2)$	...	$d(x_n, p_k)$

Fig. 2. Point-pivot pairwise $n \times k$ distance matrix $D_{P,d}(S)$.

In Step 1, the query cube is a superset of the image of the query ball. Therefore, after all the points in the query cube are retrieved, their distances to the query object have to be computed to remove the points whose pre-images are not covered by the query ball.

Step 1 is the focus in this paper.

3.2. Pivot space model

Let (M, d) be a metric space, where M is the space containing the data, and d is a metric distance function. Let $S = \{x_i \mid x_i \in M,\ i = 1, 2, \ldots, n\}$ be the database, $n \geqslant 1$. S is a finite indexed subset of M. Duplicates are not allowed.

Let $P = \{p_j \mid j = 1, 2, \ldots, k\}$ be a set of pivots. $P \subseteq S$. Duplicates are not allowed.

Definition 1 *(Pivot space, $F_{P,d}(S)$).* Given the set of pivots $p_j \in P$, each point in S can be mapped to a non-negative number, which is its distance to p_j. That is, the following mappings can be defined:

$$f_j : M \to R^+ \cup \{0\}, \qquad f_j(x) = d(x, p_j), \quad p_j \in P,\ j = 1, \ldots, k.$$

The ordered collection of these k mappings defines a vector-valued mapping $F_{P,d}$ on M, which maps a point in M to a point in the non-negative orthant of R^k. The jth coordinate of the image represents the distance to p_j:

$$F_{P,d} : M \to R^k: \quad x_p \equiv F_{P,d}(x) = \big(f_1(x), \ldots, f_k(x)\big) = \big(d(x, p_1), \ldots, d(x, p_k)\big) \in F_{P,d}(M).$$

We say x_p, a k-dimensional vector in R^k, is the image of x. The *pivot space* of S is defined as the image set of S under $F_{P,d}$:

$$F_{P,d}(S) = \big\{x_p \,\big|\, x_p = F_{P,d}(x) = \big(d(x, p_1), \ldots, d(x, p_k)\big),\ x \in S\big\}.$$

$F_{P,d}(x)$ can be written as $F(x, P, d)$ and $F_{P,d}(S)$ as $F(S, P, d)$ to show that these mappings are specified in term of three parameters. Pivot space shall also refer to the vector space R^k in which $F_{P,d}(S)$ resides, since confusion will not result.

Here and in what follows, the superscript p denotes objects in the pivot space.

Definition 2 *(Point-pivot pairwise distance matrix $D_{P,d}(S)$).* $D_{P,d}(S)$ is an $n \times k$ matrix (Fig. 2), whose (i, j)th element is the distance from the ith data point to the jth pivot. Each row vector (x_i^p) can be regarded as a point in the pivot space specified by all the distances from a database point to each of the pivots, and each column vector (p_j^p) can be regarded as a basis vector in the pivot space specified by the distances from all points to the jth pivot:

$$D_{P,d}(S) = \big(x_1^p, x_2^p, \ldots, x_n^p\big)^T = \big(p_1^p, p_2^p, \ldots, p_k^p\big),$$

where the x_i^p's are the row vectors:

$$x_i^p \equiv F_{P,d}(x_i) = (d_{i1}, d_{i2}, \ldots, d_{ik}),$$

and p_j^p is the column vector:

$$p_j^p \equiv (d_{1j}, d_{2j}, \ldots, d_{nj})^T.$$

Moreover:

$$d_{ij} = d(x_i, p_j) \geqslant 0, \quad i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, k.$$

$D_{P,d}(S)$ can also be written as $D(S, P, d)$.

Properties of the pivot space and the distance matrix include:

(1) Each point in $F_{P,d}(S)$ is a row in $D_{P,d}(S)$. $F_{P,d}(S)$ and $D_{P,d}(S)$ are representations of one another.
(2) Because different points in S can have the same distance to a pivot, there might be duplicates among the rows of $D_{P,d}(S)$. Therefore, the correspondence between the database points and points in the pivot space $F_{P,d}(S)$, or the rows of $D_{P,d}(S)$, is many-to-one. Similarly, there is a many-to-one correspondence between the pivots and the points in $F_{P,d}(S)$, or the columns of $D_{P,d}(S)$.

The following characterizes the case when all the database points are used as pivots.

Definition 3 *(Complete pivot space $F_d^c(S)$ and complete distance matrix $D_d^c(S)$).* $F_d^c(S) = F_{S,d}(S)$ and $D_d^c(S) = D_{S,d}(S)$. That is, when all points in S are selected as pivots, the complete pivot space of S is the pivot space of S, and the complete distance matrix of S is the distance matrix of S.

The complete pivot space and the complete distance matrix have the following properties.

(1) The dimension of the complete pivot space $F_d^c(S)$ is n, the size of S.
(2) Because duplicates are not allowed in S, there are no repetitions in the ordered set of distances from one point to all others. Therefore, there are no duplicates in $F_d^c(S)$.
  (2.1) Each point in the complete pivot space has exactly one coordinate with value zero, while all other coordinates are positive. That is, all such points reside in the non-negative orthant of R^n.
  (2.2) $D_d^c(S)$ is a symmetric $n \times n$ matrix, with zeros on the main diagonal and positive entries elsewhere.
  (2.3) The correspondence between points in S and points in $F_d^c(S)$ (row vectors of $D_d^c(S)$) is one-to-one. The correspondence between points in S and the basis vectors in $F_d^c(S)$ (column vectors of $D_d^c(S)$) is one-to-one.
(3) $D_d^c(S)$ contains all and only the information provided by the distance oracle. Different data sets from various metric spaces can have the same complete pairwise distance matrix.

The following shows the identicalness between pivot spaces.

Theorem 1. *Given metric space (M, d), database S, and pivot set P, then $F(S, P, d) = F(F(S, P, d), F(P, P, d), L^\infty)$, and $D(S, P, d) = D(F(S, P, d), F(P, P, d), L^\infty)$.*

Proof. It suffices to show that

$$F(S, P, d) = F\big(F(S, P, d), F(P, P, d), L^\infty\big).$$

The following 5 relations follow by definition:

$$F(S, P, d) = \big\{x^p \mid x^p = F(x, P, d) = \big(d(x, p_1), \ldots, d(x, p_k)\big),\ x \in S\big\}$$
$$F(P, P, d) = \big\{p^p \mid p^p = F(p, P, d) = \big(d(p, p_1), \ldots, d(p, p_k)\big),\ p \in P\big\}$$
$$F\big(F(S, P, d), F(P, P, d), L^\infty\big) = \big\{F\big(x^p, F(P, P, d), L^\infty\big)\big\}$$
$$F\big(x^p, F(P, P, d), L^\infty\big) = \big(L^\infty\big(x^p, p_1^p\big), \ldots, L^\infty\big(x^p, p_k^p\big)\big),\ x^p \in F(S, P, d)$$
$$L^\infty\big(x^p, p^p\big) = L^\infty\big[\big(d(x, p_1), \ldots, d(x, p_k)\big), \big(d(p, p_1), \ldots, d(p, p_k)\big)\big] = \max\big\{\big|d(x, p_j) - d(p, p_j)\big|,\ j = 1, 2, \ldots, k\big\}.$$

Equality holds in one of the triangle inequalities:

$$\big|d(x, p_j) - d(p, p_j)\big| \leqslant d(x, p), \quad j = 1, 2, \ldots, k,$$

since for $p \in P \subseteq S$, there exists a t such that $p = p_t$, $t \in \{1, \ldots, k\}$.

Therefore, we deduce that $L^\infty(x^p, p^p) = d(x, p)$, and $F(x^p, F(P, P, d), L^\infty) = (d(x, p_1), \ldots, d(x, p_k)) = x^p$.

Thus $F(S, P, d) = F(F(S, P, d), F(P, P, d), L^\infty)$, as required. □

In others word, Theorem 1 says that for any pivot space $F(S, P, d)$ generated from a metric space (S, d) and a set P of k pivots, an identical pivot space can be generated from a subset of R^k, i.e. $F(S, P, d)$, with pivots $F(P, P, d)$ and the L^∞ distance. Therefore, when dealing with the pivot space, there is no difference whether the pivot space is created from a metric space dataset S or a real coordinate dataset $F(S, P, d)$. Thus, we can assume that the original data is $F(S, P, d)$, in R^k.

In the complete pivot space, Theorem 1 becomes:

$$F^c_{L^\infty}\big(F^c_d(S)\big) = F^c_d(S), \quad \text{and thus} \quad D^c_{L^\infty}\big(F^c_d(S)\big) = D^c_d(S).$$

Note that it states the known fact that any finite metric space (size n) is isometric to a metric space formed by a subset of R^n (R^{n-1}, to be more precise) with the L^∞ distance [20].

In the following, Corollary 1 states that the mapping from the metric space to the complete pivot space can be applied recursively and the result remains invariant. Corollaries 2 and 3 describe the impact of Theorem 1 on distance-based indexing. The proofs are omitted as they are straightforward.

Corollary 1. *Let* $F^c_{L^\infty}{}^{(1)}(F^c_d(S)) = F^c_{L^\infty}(F^c_d(S))$ *and* $F^c_{L^\infty}{}^{(n+1)}(F^c_d(S)) = F^c_{L^\infty}(F^c_{L^\infty}{}^{(n)}(F^c_d(S)))$, $n \geqslant 1$. *Then*:

$$F^c_{L^\infty}{}^{(n)}\big(F^c_d(S)\big) = F^c_d(S), \quad n \geqslant 1.$$

Corollary 2. *Since the data set S from any general metric space can be mapped isometrically into the complete pivot space with the* L^∞ *metric without loss of any distance information, instead of indexing S in a black-box metric space, it is equivalent to index* $F^c_d(S)$ *in the vector space* R^n.

Corollary 3. *The intrinsic dimension of S and* $F^c_d(S)$ *are equal.*

This is because that the intrinsic dimensions are specified by the pairwise distances and the pairwise distance matrices of S and $F^c_d(S)$ are the same.

4. Dimension reduction for distance-based indexing

This section is focused on Step 1 of tackling distance-based indexing via R^n. The underlying question is how to map a metric space to R^n? We study the question from the perspective of dimension reduction. The discussion starts from the complete pivot space, the most straightforward case and the case with all the information. Issues under discussion include "can we evaluate similarity queries in the complete pivot space directly?", "how to perform dimension reduction for the complete pivot space?", "why is pivot selection important?", and "how to select pivots?".

4.1. Evaluating similarity queries in the complete pivot space directly

Theorem 2 answers the question "can we evaluate similarity queries in the complete pivot space directly?". It is straightforward and helps to understand Section 4.2.

Theorem 2. *Evaluation of similarity queries in the complete pivot space degrades the query performance to linear scan.*

Proof. Given a similarity query object, the computation of its coordinates in all dimensions of the complete pivot space is already a linear scan of the database. □

Therefore, to answer the similarity query posed in the metric space in R^n, dimension reduction in the complete pivot space is inevitable.

4.2. Pivot selection: the only form of effective dimension reduction

Theorem 2 establishes the necessity of dimension reduction in the complete pivot space. The underlying reason is the lack of a coordinate structure in a general metric space. In the following, we discuss the type of dimension reduction that can be applied to the complete pivot space. A dimension reduction technique is termed "effective" if evaluation of similarity queries in the space generated by this technique does not degrade to a linear scan. Not all dimension reduction methods for multi-dimensional indexing are effective for distance-based indexing.

Let $S = \{x_i \mid i = 1, 2, \ldots, n\}$, be the database, and let d be the distance oracle. Let y be an arbitrary point in the metric space, and let its coordinates in the complete pivot space $F^c_d(S)$ be $y^{(1)}, y^{(2)}, \ldots, y^{(n)}$. Then, $y^{(i)} = d(y, x_i)$, $i = 1, 2, \ldots, n$. Let a new coordinate $y^{(n+1)} = G(y^{(1)}, y^{(2)}, \ldots, y^{(n)})$, be specified by some unspecified function G. For example, G might be a linear combination of its variables. Given a query object q, to compute coordinate $q^{(n+1)}$, coordinates $q^{(1)}, q^{(2)}, \ldots, q^{(n)}$ in the complete pivot space $F^c_d(S)$ have to be computed first. This already represents a linear scan of the database. Therefore, if

Table 1
An example of new dimension based on all existing dimensions where $n = 2$.

	Original dimensions		New dimension: 45° line
	$y^{(1)} = d(y, x_1)$	$y^{(2)} = d(y, x_2)$	$y^{(3)} = (y^{(1)} + y^{(2)})/\sqrt{2}$
x_1	$d(x_1, x_1)$	$d(x_1, x_2)$	$(d(x_1, x_1) + d(x_1, x_2))/\sqrt{2}$
x_2	$d(x_2, x_1)$	$d(x_2, x_2)$	$(d(x_2, x_1) + d(x_2, x_2))/\sqrt{2}$

	p_1	p_2	...	x'_n
x_1	$d(x_1, p_1)$	$d(x_1, p_2)$	...	$d(x_1, x'_n)$
x_2	$d(x_2, p_1)$	$d(x_2, p_2)$	...	$d(x_2, x'_n)$
...	...	...	...	...
...	...	...	...	...
x_n	$d(x_n, p_1)$	$d(x_n, p_2)$	...	$d(x_n, x'_n)$

Fig. 3. Pivot selection keeps some of the columns of the complete distance matrix and removes the others.

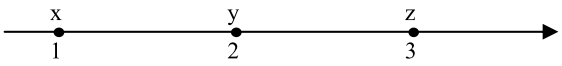

Fig. 4. Example of pivot selection: 3 points on a line.

a dimension reduction technique creates new dimensions based on all existing dimensions, evaluation of similarity queries in the space generated by this technique degrades to a linear scan. Namely, answering the query without using the index data structure.

Table 1 shows an example where $n = 2$, i.e. there are only two points in the database. Therefore, the complete pivot space has dimension 2 and the data is mapped on to a 2-d plane. Assume a new dimension is created along the 45° line, then the coordinate of y under the new dimension can be computed as: $y^{(3)} = (y^{(1)} + y^{(2)})/\sqrt{2}$. To compute $y^{(3)}$, one has to know $y^{(1)}$ and $y^{(2)}$, which is a linear scan of the database already.

Although the power of dimension reduction methods in multi-dimensional indexing is clear, the cost to set up the coordinate structure based on new dimensions is excessive. Therefore, dimension reduction for the complete pivot space should only select existing dimensions.

With k pivots, the dimension of the pivot space is k, decreased from n in the complete pivot space. For the pairwise distance matrix, pivot selection specifies a sub-matrix with the same number of rows just by removing some columns (usually most). This corresponds to dimension reduction in the complete pivot space (Fig. 3).

Therefore, pivot selection is the only effective form of dimension reduction for the complete pivot space. It is the only effective form of mapping from a metric space to R^k.

4.3. The importance of pivot selection

To date, the major attention in this area has been on the partitioning problem of Step 2. The importance of pivot selection is not fully recognized, and most methods just try to mimic multi-dimensional indexing in metric spaces.

Pivot selection can be viewed as a process of information loss. The information available to Step 2 is limited by pivot selection. Pivot selection impacts the distribution of data in the pivot space, which is critical to the search performance of Step 2. Moreover, in a pivot space produced from a "better" set of pivots, data are typically more distinguishable from each other. Thus, the number of false positives is reduced and Step 3 is faster.

For example, take three points $\{x, y, z\}$ located on an axis at 1, 2 and 3, respectively (see Fig. 4). If x (or z) is selected as the pivot, then the three points will have three unique and distinguishable distances from the pivot, namely 0, 1 and 2, respectively. However, if y is the pivot, both x and z will have distance 1 from it, making them indistinguishable.

A second example is to select two pivots for points randomly sampled from the unit square with the Euclidean norm. The common method is to use the FFT algorithm to select the two farthest opposite corners of the data [16]. For uniform vector data, these two corners are close to the two ends of the data's first principal component (with the largest eigenvalue). The distances from a point to the two farthest opposite points largely correlate with each other. If a point is close to the first pivot, usually it is far from the second pivot. As a result, if two points are not distinguishable by the first pivot (their distances to the first pivot are similar), they are not likely to be particularly distinguishable by the second pivot. Therefore, selecting these two pivots does not provide much more information than selecting just one pivot. An alternative is to select two corners on the 1st and 2nd principal components. Since the principal components are orthogonal to each other, the two pivots in this case define two less correlated dimensions. If two points are not distinguishable by one pivot, they can still be distinguished by the other pivot.

Fig. 5 shows the pivot spaces of points randomly sampled from the unit square with different pivots. The two pivots are opposite corners $(0, 0)$ and $(1, 1)$ in (a) and are neighboring corners $(0, 0)$ and $(1, 0)$ for (b). We can see that the pivot space in (b) spreads out more widely (less densely) than that in (a). Therefore, data in pivot space in (b) are more distinguishable.

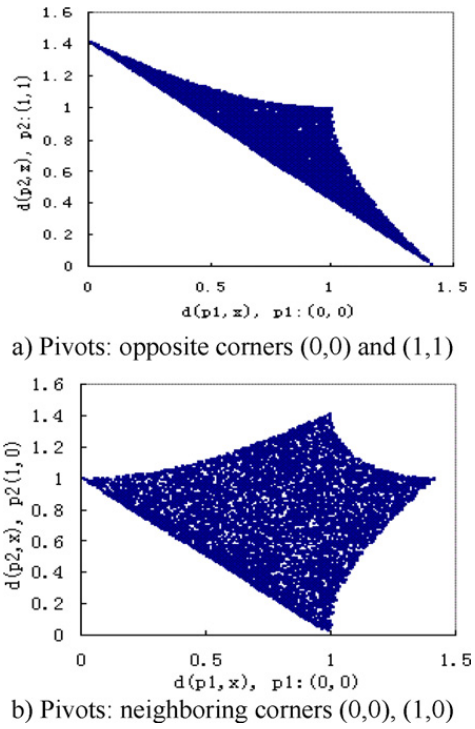

Fig. 5. Pivot spaces (2 pivots) of points randomly sampled from the unit square, with different selection of pivots.

Although pivots are always corners, different corners can still have much different impact on query performance. This supports Bustos et al.'s observation that outliers might not be good pivots [5].

4.4. Adapt dimension reduction to pivot selection

We propose a heuristic to adapt general dimension reduction methods that create new dimensions to distance-based problems. The basic idea is to select existing coordinates that best approximate the new dimensions created by dimension reduction.

One way is to select the existing coordinate with the maximum correlation, or the minimum angle, with the new dimensions created by dimension reduction. Bustos et al. [5] suggest that a good choice of pivots should maximize the mean of the pairwise distances in the pivot space. To increase the mean of the pairwise distance also, we consider the covariance instead of the correlation. That is, for each new dimension, select the point whose image in the pivot space has the largest projection on that new dimension.

With this heuristic, a general dimension reduction technique can be adapted to pivot selection in two steps. First, run the dimension reduction on the complete pivot space to create the new dimensions. Second, select the pivots with the largest projections on the new dimensions.

We show how to adapt PCA to pivot selection with this heuristic in the next section.

5. PCA in distance-based indexing

In this section, we apply PCA to pivot selection and to the estimation of intrinsic dimension.

5.1. PCA for pivot selection

In Step 3 (see in Section 3.1), points of the query cube in the pivot space need to be checked by computing the distance with the query object directly. Therefore, it is of key importance to reduce the number of points in the query cube. We aim to maximize the variance of the data along the dimensions of the pivot space, which means the data points are more distinguishable among each other. Because PCA considers the distribution of the whole data set and maximizes the variance of the data along each principal component, we choose it for pivot selection.

A difficulty with PCA is the computational cost. Even a fast approximation usually takes $O(n^2)$ time, which is too expensive for large databases even if it is computed off-line. As Bustos et al. [5] point out, although good pivots are usually outliers, outliers are not always good pivots. A few bad outliers can easily ruin the results of selection. However, the set of outliers forms a good candidate set for good pivots. PCA can be conducted on the candidate set. In other words, PCA is not performed on the complete pivot space, but on a pivot space with outliers as pivots. Since the size of the set of outliers is much smaller than the database, the computational cost of PCA is decreased.

```
PivotSelection ((S, d): data set, k: number of pivot, c: constant)
{   //run FFT to create a candidate set of size k * c
1.  candidate = FFT(S, k * c);
    //generate the pivot space with candidate as the pivot set
2.  PS = F(S, candidate, d);
    //run PCA on PS
3.  PCSET = EMPCA(PS, k);
    //for each PC, find the point with the largest project on it
    Pivots = {};
4.  for each PC ∈ PCSET
        Pivots = Pivots U argmax_x(Proj(PC, x));
    return Pivots;    }
```

Fig. 6. Algorithm for pivot selection.

Our pivot selection algorithm is shown in Fig. 6. In step 1, a number of outliers of the data are found by FFT and taken to form a candidate set of pivots. The constant c is named the *FFT-scale*, because of the fact that if k pivots are to be selected, ck corners will be selected by FFT. Empirical results show that a good choice of c is approximately 30. In step 2, the distances between the corners and the database points are computed to form the distance matrix of the candidate outlier pivot space, on which PCA is performed in step 3. The PCA algorithm applied is EMPCA [25], which can compute only the given number of principal components with the largest eigenvalues. Finally in step 4, for each principal component, the data point whose image in the pivot space has the largest projection on that principal component is selected as a pivot. Since k and ck is much smaller than the database size, the time complexity is $O(n)$. The algorithm is compared with FFT and Bustos et al.'s incremental sampling selection method [5].

5.2. Estimating the intrinsic dimension

According to previous work [20] and the pivot space model, the complete pivot space together with the L^∞ distance is isometric to the original metric space. Therefore, methods to estimate the intrinsic dimension of R^n [6,17] are now applicable to a metric space. We introduce a third method to estimate the intrinsic dimension based on the relative change of eigenvalues of PCA in the complete pivot space, equivalently, of the complete distance matrix $D_d^c(S)$.

Method 3. Let $Q = \{q_1, q_2, \ldots, q_n\}$ be the principal components (PCs) of the data in the complete pivot space. Let $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ be the variances (eigenvalues) corresponding to each PC, $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_n \geqslant 0$. Note that the λ_i are normalized so that they sum to 1. The intrinsic dimension is estimated as

(1) $\hat{d} = \text{argmax}_i(\lambda_i/\lambda_{i+1})$, $i = 1, \ldots, n-1$, and
(2) $\sum_{j=1}^{\hat{d}} \lambda_j > 0.6$, and
(3) $0.015 \leqslant \lambda_{\hat{d}+1} \leqslant 0.035$.

Condition (1) indicates that the eigenvalues decrease relatively the most from λ_i to λ_{i+1}. Condition (2) guarantees that at least 60% of the variance of the data is maintained if they are reduced to dimension $\hat{d}$. Condition (3) ensures that principal components with tiny ($< 1.5\%$) variance (eigenvalue) will be excluded and principal components with larger ($> 3.5\%$) variance will be included.

The three methods are evaluated using a suite of workloads. The results in Section 6 show that all three methods are asymptotically correct, while Method 3 gives quantitatively more accurate estimates for data whose intrinsic dimension are known.

6. Empirical results

The empirical study involves two test suites, the MoBIoS test suite [21] and the SISAP test suite [28] used in [5]. Two index structures are used, i.e. the MVP tree [3] and the pivot table [23]. The effect of the constant FFT-scale is first study. Then the query performance of our PCA method and the incremental selection method is compared. At last, the PCA-based method to estimate intrinsic dimension is applied to the MoBIoS test suite.

6.1. Test suite

The MoBIoS test suite consists of synthetic vector data, biological data, real vector data and an image dataset [21].

The synthetic vector data consists of multi-dimensional vector data of uniform, exponential and normal distributions. Different dimensions have independent identical distributions. The real vector data consists of the US cartographic boundary data of Texas and Hawaii.

Table 2
Summary of MoBIoS test suite.

Workload	Total size	Distance oracle	Domain dimension
Vector (uniform)	1M		1–20
Vector (exponential)	100k		1–10
Vector(normal)	100k	L^1, L^2, L^∞ norm	1–10
Texas	190k		2
Hawaii	9k		2
Mass-spectra	$<$ 90k	Fuzzy cosine distance	40 000
Protein	100k	Weighted edit distance	6–18
DNA	$<$ 256k	Hamming distance	9–18
Image	10 221	L-norms (linear comb.)	66

Three types of biological data are considered: (1) the amino-acid sequence fragments of the yeast proteome with weighted-edit distance based on the metric PAM substitution matrix, mPAM [32], (2) the DNA sequence fragments of the Arabidopsis genomes with Hamming distance, and (3) analytically determined peptide fragmentation spectra of human and E. coli proteins with a pseudo-semi-metric cosine distance.

The protein sequence dataset contains FASTA formatted amino-acid translations extracted from GenBank/EMBL/DDBJ records that are annotated with one or more CDS features. We split the sequences into overlapping fixed length fragments or q-grams. The fragment length is 5. There is one q-gram for each sequence element. The distance between two fragments is their global alignment score. The substitution matrix used is mPAM [32], which makes the distance function metric. The distance values are integers between 0 and 30.

In our mass spectra data, each spectrum is represented as a very high-dimensional binary vector. A fuzzy measure of the shared peaks count (SPC) can be interpreted as a modified form of cosine distance. Two peaks are marked as being equal if the absolute difference of their m/z values lies within a certain tolerance. Based on the cosine distance, the distance between two data points is the angle between their vector representations. Thus, the distance values are continuous in $[0, \pi/2]$. For mass-spectra, most of the data point pairs have the maximum distance, indicating that points are far away from each other, and there is no intrinsic clustering.

The image dataset consists of 10 221 images. Each image is represented by 3 sets of features reflecting the image properties in structure, texture, and color. Each set of features can be considered as a vector. Consequently, an image can be represented by three vectors, with length 3 for structure, 15 for color and 48 for texture. For each feature set, a distance function is defined. For texture and structure features, the distance functions are both L2 norm. For the color feature, the distance function is actually L1 norm. The final distance is a linear combination of the distances of each feature set, with coefficients of value 1/3 respectively. Since both L2 and L1 norms have the metric properties, the final distance also has the metric properties.

The MoBIoS test suite is summarized in Table 2. Please note that database size refers to the number of data records in a database.

Two datasets from the SISAP test suite [28] are involved, i.e. the English dictionary and the NASA image archives. The English dictionary consists of 69 069 words and uses the edit distance. The NASA archives consist of 40 700 images represented as 20-dimensional vectors with the Euclidean distance.

Please note that uniformly distributed vector dataset is included in both the MoBIoS test suite and the SISAP test suite. In this paper, uniform vector dataset is included when we study the SISAP test suite.

6.2. Index structures and methodology

For MVPT [3], the partition algorithm is clustering partition [18]. The number of pivots is 2 for Texas and mass-spectra data, 4 for DNA and protein data, and 3 for all others. Based on each pivot, 3 partitions are generated. The maximum number of data points in each index leaf node is 100.

For pivot table [23], number of pivots used are 5, 10, 15, ..., 100.

The sizes of the databases are all 100k, except for those small workloads for which only limited amounts of data is available. Since distance evaluation in a metric space is usually costly, we use the average number of distance calculations, which is implementation independent, to answer 5000 (for MVTP, or 10% of the database size) range queries as the performance measure of each index. The queries are chosen sequentially from the beginning of the dataset files of each workload. The radii of range queries are chosen so that approximately 0.01% of the databases are returned as query results.

6.3. FFT-scale

To reveal the effect of FFT-scale, 100 MVP indices are built for each workload using PCA-based pivot selection with FFT-scale varying from 1 to 100. The query performances of 4 workloads are plotted in Fig. 7. We can see that the query performance gets better (number of distance computations decreases) as the FFT-scale increases. The query performance is stable for FFT-scale greater than 20–40. The results of other data show a similar trend. Therefore, we use FFT-scale = 30 in further experiments.

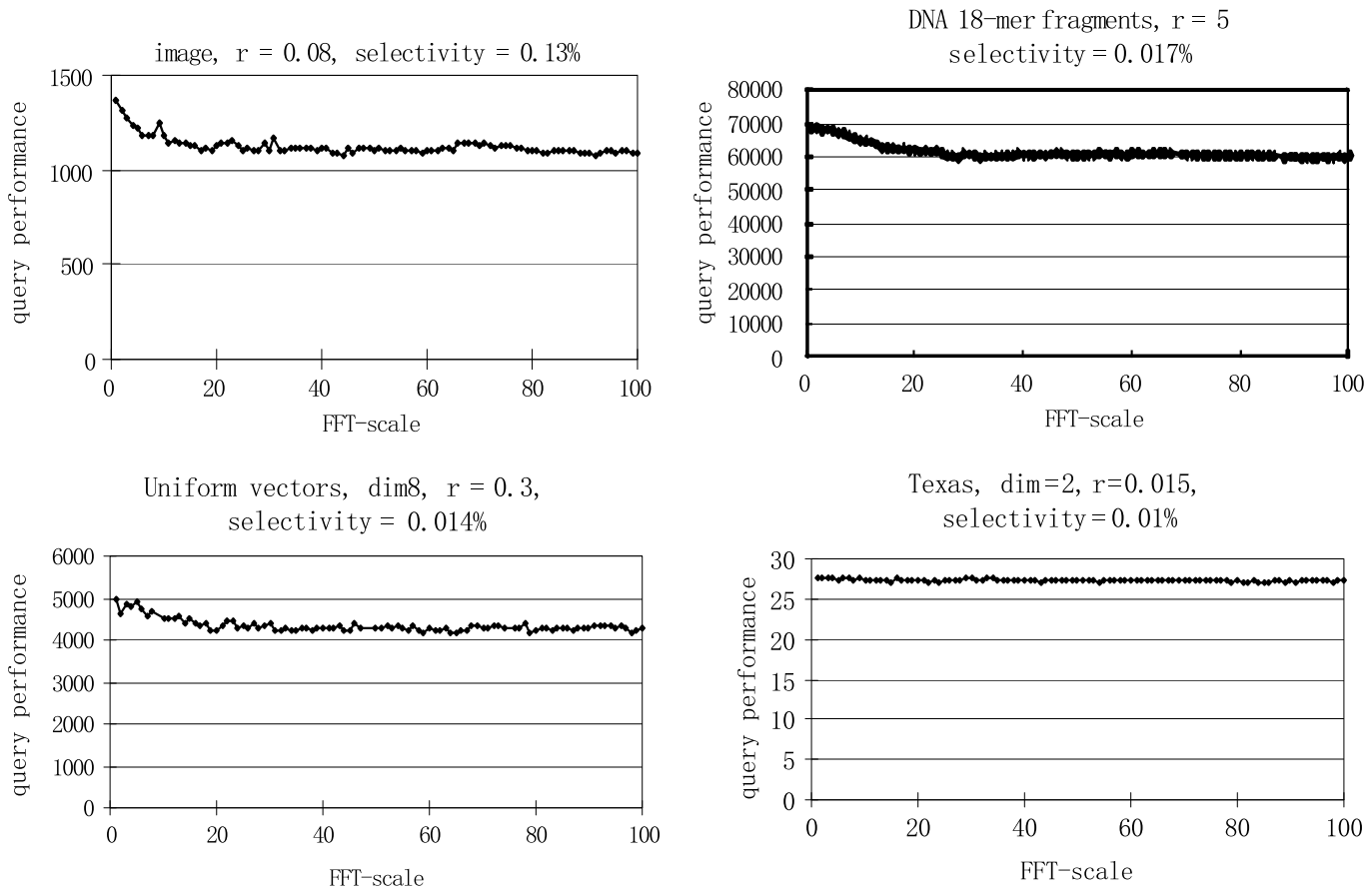

Fig. 7. Effect of FFT-scale on query performance.

Table 3
Comparison of pivot selection heuristics, measured by query cost per the number of distance calculations.

Dataset	Radius	Selectivity	FFT	Incremental				PCA	
			Query	Build	Query	*A*	*N*	Build	Query
DNA 18-mer fragments	5	0.017%	68357.2	60.9M	62417.7	9k	10	57.5M	60968.1
Texas	0.015	0.011%	27.7	66.6M	26.7	30k	10	58.8M	27.5
Uniform vector dimension 8	0.3	0.015%	4952.5	66.3M	4822.6	30k	10	58.5M	4394.4
Image	0.08	0.13%	1371.4	5.4M	1209.9	5k	15	4.6M	1106.1

6.4. Comparison of pivot selection heuristics

6.4.1. On MVPT and the MoBIoS test suite

We compare the three pivot selection heuristics, i.e. FFT, Bustos et al.'s incremental selection method [5], and the PCA-based method. MVP indices are built with different heuristics and their query performances are compared.

The construction cost of indices is measured by number of distance computations to build them. To be fair, we make sure the PCA-based method and the incremental method have similar construction cost. There are two parameters, *A* and *N* for the incremental method. Per Bustos et al. [5], we try to use a large value for *A* and a small value for *N*. The results are shown in Table 3. Moreover, since it is hard to make the two methods have exactly the same construction cost, we always allow more construction cost for incremental method (see Table 3).

The query performances of the three methods are also shown in Fig. 8 (normalized by the value of the PCA method). Both Fig. 8 and Table 3 show that for all workloads, FFT always yields the worst query performance. For most of the workloads, the PCA based method yields the best query performance. The only case where the incremental method yields the best performance is for Texas data, where all the three methods do not differ much. We believe this is because Texas data are of low intrinsic dimension and so are easy to index.

6.4.2. On MVPT and the SISAP test suite

In this section, we compare the query performance the three pivot selection heuristics on the SISAP test suite using MVP index. The number of pivot is always 3. For each dataset, three indexes are built for each pivot selection heuristic,

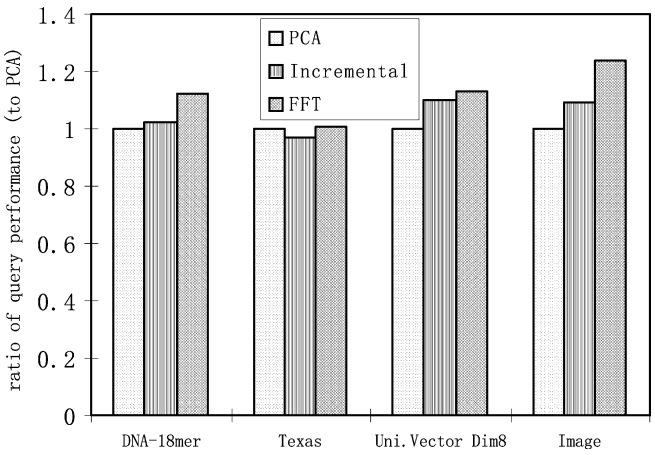

Fig. 8. Query cost of three pivot selection methods.

Table 4
Range query radii and selectivity of the SISAP test suite.

Dataset	Uniform vector dim 8					NASA image archives			English dictionary	
Size	100 000					40 150			69 000	
Radius	0.2	0.25	0.3	0.35	0.4	0.35	0.4	0.45	1	2
Selectivity	0.002%	0.005%	0.015%	0.043%	0.11%	0.05%	0.08%	0.15%	0.005%	0.04%

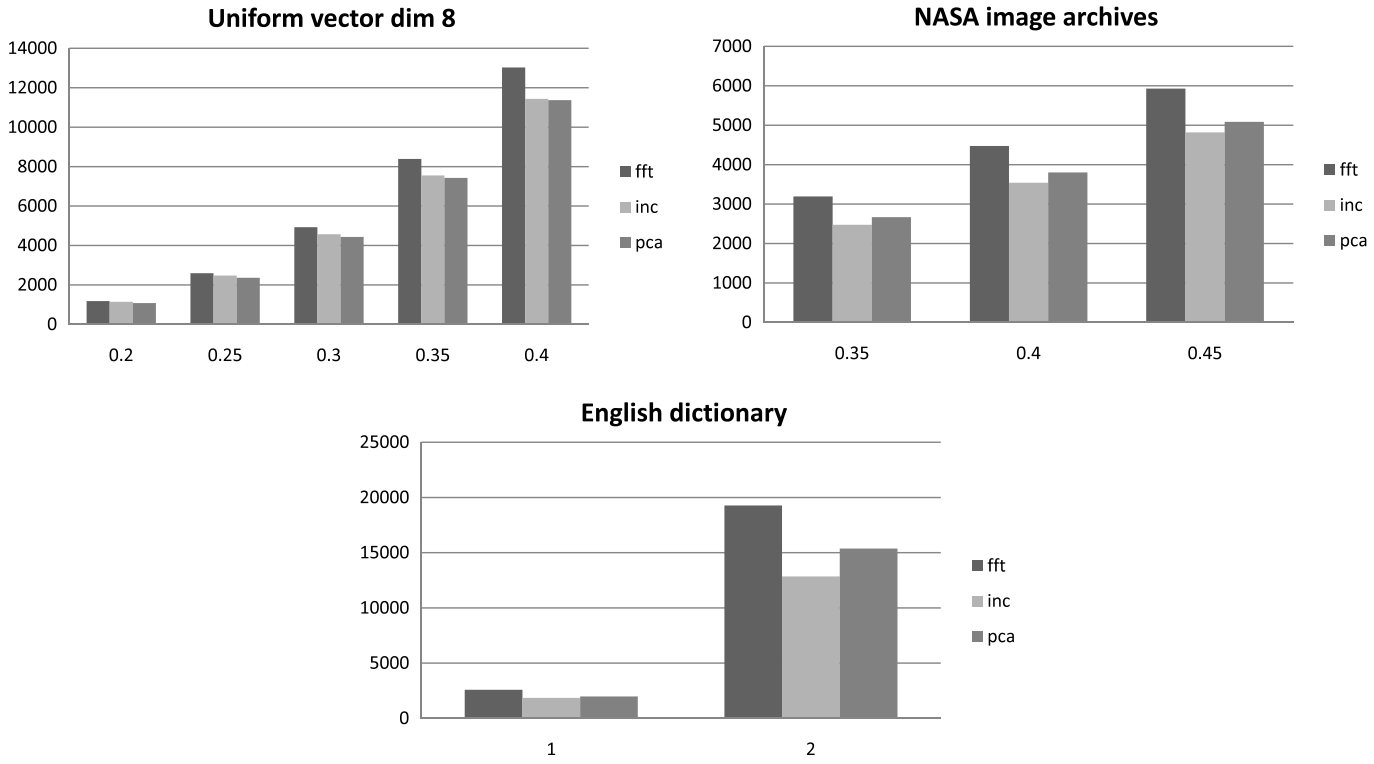

Fig. 9. Query performance of three pivot selection methods on the SISAP test suite using MVPT. (The vertical axis shows number of distance calculations while the horizontal shows range query radius.)

respectively. 10% of the database taken from the beginning of the data file is used as range query objects with a number of radii. The radii and the corresponding selectivity are listed in Table 4.

Fig. 9 shows the query performance of the three pivot selection heuristics on the SISAP test suite using MVPT. FFT turned out to be the worst for all the cases. The PCA method outperforms the incremental method for uniform vector data with dimension 8, but is outperformed by the incremental method for NASA image and English dictionary. However, the differences between the PCA and the incremental methods are marginal for all the cases except radius 2 of English dictionary.

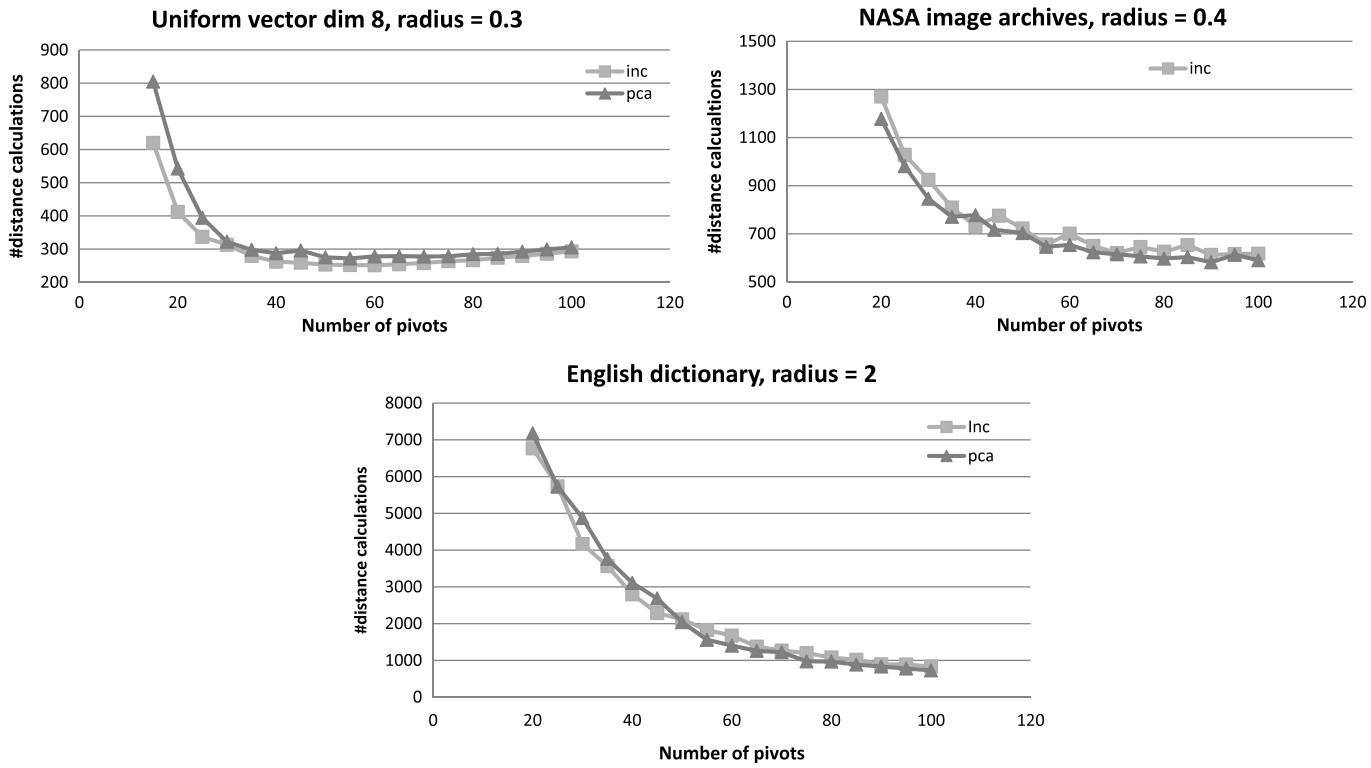

Fig. 10. Query performance of three pivot selection methods on the SISAP test suite using pivot table.

6.4.3. On pivot table and the SISAP test suite

To further compare the PCA method and the incremental method, we run them on the SISAP test suite using pivot table. The range query radii are fixed while the number of pivot varies. The results are shown in Fig. 10. We can see that the incremental method slightly outperforms the PCA method for uniform vector while slightly outperformed by the PCA method for NASA images. For English dictionary, the incremental method is slightly better while the number of pivots is less than 50, and is slightly worse while the number of pivots is greater than 50. Again, the differences are marginal for almost all the cases.

6.5. Estimate of intrinsic dimension

Finally, we show results of estimated intrinsic dimension on all the workloads with the three methods. Wherever possible, we vary the domain dimension to see how the estimates change. If a linear relationship is observed, we compute the slope and intercept of the relationship by linear regression. For vector data, we use L^1 and L^∞ norms in addition to the L^2 norm. The estimates are shown in Table 5. First, we can see that as domain dimensions increase, estimates given by all the three methods increase linearly (one exception is the regression method and protein data). Therefore, we think all the three methods are asymptotically accurate. Methods 1 and 2 have different values for slope and intercept for different data and metrics. No generally applicable relationship can be drawn from Methods 1 and 2. Method 3, except in a few cases, always gives $d+1$ ($q+1$) as the estimate. Thus, Method 3 is more consistent and stable.

Moreover, Method 3 almost always yields $d+1$ for the vector data of dimension d, no matter what is the probability distribution of the data and the metric. This is consistent with the observation that if the distances from an unknown vector to $d+1$ vectors are known, then the coordinates of the unknown vector can be computed accurately. The three methods are consistent with each other for image data.

In our opinion, intrinsic dimension is an indicator of the difficulty of indexing. However, it is not a solo indicator. In addition to intrinsic dimension, the alphabet size and the type of distance function are also indicators of difficulty of indexing. Intrinsic dimension represents the level of the difficulty of indexing, or indexing complexity. It is analogous to the exponent in the time complexity of algorithms. For example, intrinsic dimensions 1 and 2 in describing the difficulty of indexing are analogous to $O(n)$ and $O(n^2)$ in describing time complexity of algorithms.

7. Conclusions and future work

The “power” of metric-space indexing is the generality of the abstraction and encapsulation: just provide a distance function. But, we (the community) have been powerless to do anything other than apply what we can make work with respect to R^n. To date, this has been done opportunistically. The theorems in this paper reveal that within certain boundaries

Table 5
Estimates of intrinsic dimension of three methods.

Workload	Domain dimension	Distance oracle	Intrinsic dimension		
			$\mu^2/2\sigma^2$	Regression	$\text{argmax}_i(\lambda_i/\lambda_{i+1})$
Vector (uniform)	$D = 1$–20	L^∞	$1.72d - 1.81$	$0.73d + 0.88$	$d+1$ ($d \neq 3, 4$), 4, 7 ($d = 3, 4$)
		L^1	D	$0.75d + 0.84$	$d+1$
		L^2	$1.41d - 0.71$	$0.78d - 0.72$	$d+1$
Vector (exponential)	$D = 1$–10	L^∞	$0.244d + 0.446$	$0.676d + 0.62$	$d+1$
		L^1	$0.499d - 0.0006$	$0.737d + 0.482$	$d+1$
		L^2	$0.427d + 0.113$	$0.72d + 0.534$	$d+1$
Vector (normal)	$D = 1$–10	L^∞	$0.644d + 0.559$	$0.858d + 0.325$	$d+1$
		L^1	$0.875d + 0.002$	$0.863d + 0.32$	$d+1$
		L^2	$0.989d - 0.145$	$0.872d + 0.305$	$d+1$
Texas	2	$L^\infty/L^1/L^2$	1.29/1.42/0.87	1.54/1.54/1.51	3
Hawaii	2	$L^\infty/L^1/L^2$	0.31/0.26/0.36	1.47/1.45/1.44	2
Protein q-gram	$q = 6$–18	Weighted edit distance	$2.46q + 2.32$	$-0.08q + 4.16$	$q+1$ ($q < 18$), 17 ($q = 18$)
DNA q-gram	$q = 9$–18	Hamming distance	$1.27q + 0.37$	$0.14q + 2.52$	$q+1$ ($q < 18$), 21 ($q = 18$)
Mass-spectra	40 000	Fuzzy cosine distance	0.62	1.23	2
Image	66	Linear combination of L-norms	5.26	4.85	5

we can in fact be methodical about this, and, for example, we now even understand how to exploit PCA in this context. No surprise, the PCA inspired methods, while necessarily still heuristic, is comparable to heuristics inspired by ad-hoc observations.

We believe the objective function of incremental sampling is worthy but the associated algorithm is computationally expensive. Further, incremental sampling might be made more efficient by sampling from a set of corners instead of the whole dataset. There are other dimension reduction methods for a vector space with annealing characteristics. We anticipate more research along this direction.

By virtue of considering the complete pivot space, our work has drawn a direct parallel between distance-based indexing and high-dimensional indexing. Both have to do dimension reduction and remove false positives. One may project a finite metric space to a reduced dimension coordinate system, but no proper subset of the pivots can faithfully recreate the geometry of the space. Thus, pivot selection results in the loss of information, typical of dimension reduction techniques. Hence, dimension reduction is integral to indexing both finite metric spaces and high-dimensional data.

We show that the intrinsic dimension of data in a metric space can be estimated by the intrinsic dimension of the complete pivot space. Thus, methods for vector spaces are now applicable to metric spaces. We have demonstrated how a PCA-based method can be applied to estimate the intrinsic dimension of the metric space. More work on this topic is expected.

Another interesting piece of future work is to determine the optimal number of pivots, and its relationship with the intrinsic dimension.

Acknowledgements

We sincerely thank Gonzalo Navarro, Glen Nuckolls, and Piotr Indyk for their comments and suggestions.

This research was supported by the following grants: a grant from the US National Science Foundation, DBI-0640923; a J. Tinsley Oden Faculty Research Fellowship at the Institute for Computational Engineering and Science, the University of Texas at Austin, USA; NSF-China: 61033009, 61003272, 61170076; China NSF-GD grant: 10351806001000000; a grant from the Computer Architecture Key Lab of Chinese Academy of Sciences: "Transportation and optimization of Hadoop and GeDBIT on Loongson based platforms"; Shenzhen Foundational Research Project: JC201005280408A, JC200903120046A; a grant from the Shenzhen–Hong Kong Innovation Circle Project: ZYB200907060012A.

Willard L. Miranker, an author of this paper, passed away before the final publication. His co-authors dedicate this paper to him.

References

[1] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (9) (1975) 509–517.
[2] K.S. Beyer, J. Goldstein, R. Ramakrishnan, U. Shaft, When is "nearest neighbor" meaningful?, in: The 7th International Conference on Database Theory, Springer-Verlag, 1999.
[3] T. Bozkaya, M. Ozsoyoglu, Indexing large metric spaces for similarity search queries, ACM Trans. Database Syst. 24 (3) (1999) 361–404.
[4] S. Brin, Near neighbor search in large metric spaces, in: The 21st International Conference on Very Large Data Bases (VLDB'95), Morgan Kaufmann Publishers Inc., 1995.
[5] B. Bustos, G. Navarro, E. Chavez, Pivot selection techniques for proximity searching in metric spaces, Pattern Recogn. Lett. 24 (14) (2003) 2357–2366.
[6] F. Camastra, Data dimensionality estimation methods: a survey, Pattern Recogn. 36 (12) (2003) 2945–2954.
[7] E. Chavez, G. Navarro, R. Baeza-Yates, J. Marroqu, Searching in metric spaces, ACM Comput. Surv. 33 (3) (2001) 273–321.
[8] P. Ciaccia, M. Patella, Bulk loading the M-tree, in: 9th Australasian Database Conference (ADO'98), 1998.
[9] P. Ciaccia, M. Patella, P. Zezula, M-tree: an efficient access method for similarity search in metric spaces, presented at the 23rd International Conference on Very Large Data Bases (VLDB'97), Athens, Greece, 1997.

[10] K.L. Clarkson, Nearest-neighbor searching and metric space dimensions, in: Nearest-Neighbor Methods for Learning and Vision: Theory and Practice, MIT Press, 2006, pp. 15–59.
[11] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, A.E. Abbadi, High dimensional nearest neighbor searching, Inf. Syst. 31 (6) (2006) 512–540.
[12] E. Frentzos, K. Gratsias, N. Pelekis, Y. Theodoridis, Algorithms for nearest neighbor search on moving object trajectories, GeoInformatica 11 (2) (2007) 159–193.
[13] P. Grassberger, I. Procaccia, Measuring the strangeness of strange attractors, Phys. D 9 (1–2) (1983) 189–208.
[14] A. Guttman, R-trees: a dynamic index structure for spatial searching, in: Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, 1984.
[15] G.R. Hjaltason, H. Samet, Index-driven similarity search in metric spaces, ACM Trans. Database Syst. (TODS) 28 (4) (2003) 517–580.
[16] D.S. Hochbaum, D.B. Shmoys, A best possible heuristic for the k-center problem, Math. Oper. Res. 10 (2) (1985) 180–184.
[17] B. Kegl, Intrinsic dimension estimation using packing numbers, Adv. Neural Inf. Process. Syst. 15 (2003) 681–688.
[18] R. Mao, W. Xu, S. Ramakrishnan, G. Nuckolls, D.P. Miranker, On optimizing distance-based similarity search for biological databases, in: The 2005 IEEE Computational Systems Bioinformatics Conference (CSB 2005), 2005.
[19] R. Mao, W. Xu, N. Singh, D.P. Miranker, An assessment of a metric space database index to support sequence homology, Int. J. Artif. Intell. Tools (IJAIT) (2005) 867–885.
[20] J. Matousek, Lectures on Discrete Geometry, Springer-Verlag, New York, 2002, 497 pp.
[21] MoBIoS test suite, http://aug.csres.utexas.edu/mobios-workload/.
[22] G. Navarro, Searching in metric spaces by spatial approximation, in: Proceedings of the String Processing and Information Retrieval Symposium & International Workshop on Groupware, IEEE Computer Society, 1999.
[23] G. Navarro, Analyzing metric space indexes: What for?, in: The Proceedings of the Second International Conference on Similarity Search and Applications (SISAP2009), 2009, pp. 3–10.
[24] S.R. Ramakrishnan, R. Mao, A.A. Nakorchevskiy, J.T. Prince, W.S. Willard, W. Xu, E.M. Marcotte, D.P. Miranker, A fast coarse filtering method for peptide identification by mass spectrometry, Bioinformatics 22 (2006) 1524–1531.
[25] S. Roweis, EM algorithms for PCA and SPCA, Neural Inf. Process. Syst. 10 (1997) 626–632.
[26] H. Samet, Foundations of Multidimensional and Metric Data Structures, Morgan Kaufmann, 2006.
[27] U. Shaft, R. Ramakrishnan, When is nearest neighbors indexable?, in: Tenth International Conference on Database Theory (ICDT 2005), Springer, 2005.
[28] SISAP test suite, http://www.sisap.org.
[29] C. Traina Jr., A. Traina, C. Faloutsos, Distance exponent: A new concept for selectivity estimation in metric trees, Technical Report CMU-CS-99-110, Computer Science Department, Carnegie Mellon University, 1999.
[30] J.K. Uhlmann, Satisfying general proximity/similarity queries with metric trees, Inform. Process. Lett. 40 (4) (1991) 175–179.
[31] R. Weber, H.J. Schek, S. Blott, A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces, in: International Conference on Very Large Data Bases, 1998.
[32] W. Xu, D.P. Miranker, A metric model of amino acid substitution, Bioinformatics 20 (8) (2004) 1214–1221.
[33] P.N. Yianilos, Data structures and algorithms for nearest neighbor search in general metric spaces, in: The Fourth Annual ACM–SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 1993.
[34] C. Yu, B. Cui, S. Wang, J. Su, Efficient index-based KNN join processing for high-dimensional data, Inf. Softw. Technol. 49 (4) (2007) 332–344.