Winter 1997

A metric-based scientific data model for knowledge discovery

David T. Kao
University of New Hampshire, Durham

A Metric-Based Scientific Data Model for Knowledge Discovery

by

David T. Kao

B.S., National Central University, 1986
M.S., Michigan State University, 1991

DISSERTATION

Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Doctor of Philosophy

in

Computer Science

December, 1997

The dissertation has been examined and approved.

Dissertation Co-Director, R. Daniel Bergeron
Professor of Computer Science

Dissertation Co-Director, Ted M. Sparr
Professor of Computer Science

Raymond Greenlaw
Associate Professor of Computer Science

Georges G. Grinstein
Professor of Computer Science
University of Massachusetts, Lowell

Samuel D. Shore
Professor of Mathematics

December 16, 1987
Date

Dedication

To my father, Yu-San Kao, and my mother, Fong-Mei Lee, for their understanding, patience, and love.

Acknowledgements

Many people have contributed, in big and small ways, to the completion of this dissertation and as importantly, to the quality of my life during this journey. I would like to take advantage of this opportunity to acknowledge their effort and express my appreciation.

Professor R. Daniel Bergeron has been a constant source of inspiration. His vision played a pivoting role in identifying my research topic and defining my dissertation scope. I can always rely on him for sound advice and honest opinion.

Professor Ted M. Sparr introduced me to the field of scientific database systems. His extensive knowledge in database and statistics helped me to put my research into perspective and prevented me from reinventing wheels.

Thanks to Professor Raymond Greenlaw this dissertation has many fewer inconsistencies and editorial pitfalls. His thoroughness has set a new standard in writing for me. All the remaining inconsistencies and mistakes are solely mine.

Professor Georges G. Grinstein has provided many valuable comments and fresh ideas for me to ponder. It was a dinner conversation with him and Dan in Atlanta, Georgia that instilled the confidence in me to pursue my dissertation research.

Professor Samuel D. Shore introduced me to the fields of *logic*, *topology*, and *metric theory*, which serve as the theoretical foundation of this dissertation. He is not only my mentor in mathematics, but also a very dear friend.

During my Ph.D. study, I have had the opportunity to work with Professor Pilar de la Torre on *trie* complexity analysis. While this collaboration did not directly contribute to my dissertation research, it has been a very rewarding experience.

I enjoyed the opportunity to coauthor with Mike Cullinane on two occasions. The many hours of fun I spent with him and Sam in our weekly topology seminars will be missed a lot.

I would also like to thank the staff of computer science department for their administrative support. Linda Andrews is always pleasant and attentive to any mundane problem I brought to her. Gina Ross and Andy Evans have provided excellent hardware and software service.

Many fellow graduate students have enriched my life on a daily basis. They will be remembered and I believe that our separate paths will cross in the future.

I am grateful that my fiancée, Evelyn Pu, has been extremely patient and sup-

portive. Although we are geographically separated by thousands of miles, she never fails to make her presence felt through our many phone calls.

Last but not the least, I have to thank my parents. They have faith in me. It is through their encouragement and support that I was able to come to the States to pursue my graduate study. They always give me the best: to them, I am forever in debt.

Table of Contents

List of Figures

List of Tables

Abstract

A METRIC-BASED SCIENTIFIC DATA MODEL
FOR KNOWLEDGE DISCOVERY

by

David T. Kao
University of New Hampshire, December, 1997

Scientific data, which include multimedia (e.g., images, audio, and video) and non-standard data (e.g., finger prints and DNA sequences), is characterized by rich and complex *inter-instance relationships* in addition to the *inter-entity relationships* found in traditional data. *Conventional data models* are insufficient for modeling such inter-instance relationships. This thesis proposes a *metric-based scientific data model* from the notions of *data-as-functions* and *pseudo-quasimetrics*, which are used to model inter-entity and inter-instance relationships respectively. Compared to other scientific data models, the metric-based conceptual model can be applied to many data sets where geometric views might not otherwise be available.

A detailed approach is outlined for exploring and deriving pseudo-quasimetrics to represent inter-instance relationships in a wide variety of data. In particular, we introduce the notion of *observable properties* and show how it can be applied with ideas from *point set topology* to systematically derive metrics from nonmetric data components such as categorical data. We also demonstrate the use of *continuity* as a mathematically precise tool to validate metrics derived through the proposed approach.

In order to support the metric-based model at the physical level, we developed two simple mechanisms, the *multipolar mapping*, for transforming a pseudo-metric space into a multidimensional space, and the *median transformation*, for deriving a pseudo-metric from a pseudo-quasimetric. After application of multipolar mapping and (possibly) median transformation, it is easy to use existing *point spatial data structures* such as *quadtree* or *octree* for metric data storage and access. The results of our performance analysis demonstrate that the multipolar approach is robust and stable over a wide range of data parameters for data sets with intrinsic dimensionality of 10 or less. While it is still unclear whether the multipolar approach offers significant performance advantage on proximity

queries for data sets of very high dimensionality. preliminary results for 100 dimensional data still show excellent performance on nearest neighbor queries.

Chapter 1

Introduction

1.1 Scientific Data

Interrelationships in *scientific data* are fundamentally more implicit, difficult to capture, and harder to model than those in other database applications. For most conventional applications, the only interrelationships of interest are the ones among different semantic entities, such as the ones which can be described by an ER diagram [16, 12]. Instances of a particular semantic entity – a tuple of a relation or an object of a class – are only related by sharing a set of common properties of that semantic entity. In most situations, such *inter-entity* relationships are known in advance and expressed directly as metadata.

For scientific data, there is another aspect of complexity in the interrelationships – the interrelationships among *instances* of the same semantic entity. While the precise meaning of *scientific data* is still open to discussion, we define scientific data as data which possess rich *inter-instance relationships*. For example, both *multimedia* (e.g., images, audio, and video) and *non-standard data* (e.g., finger prints and DNA sequences) are considered as scientific data by our definition. A *scientific database system* must manage data before such inter-instance relationships are well understood. In fact, one major objective of *data mining* or *knowledge discovery in databases (KDD)* [19, 18, 6] is to discover, quantify and further validate these relationships.[1]

[1]The process of exploring meaningful patterns in data is known by different names in various research disciplines. In particular, the term *exploratory data analysis (EDA)* [63] has been used extensively in the past for such activity by the statistics community. For the rest of this thesis, we use the term *knowledge discovery* to represent all kinds of activities involving exploring, extracting, and discovering information or

In a broad sense, *scientific database systems* are database systems with integrated support for the management and analysis of scientific data. Similar to conventional database systems, scientific database systems also model and manage data according to its *semantic structure*, i.e., the interrelationships in data. Since part of the semantic structure is determined or evolves as the data analysis process proceeds, the coupling between data management and data analysis is a tight one. A *scientific data model* is a coherent framework for modeling both inter-entity and inter-instance relationships to support data management and analysis activities.

This thesis summarizes our research in the development of a formal scientific data model. As part of the thesis, a data model and a class of *physical data models* are proposed. The data model, which we call the *metric-based scientific data model*, is developed to extend the capabilities of existing *conceptual* and *implementation* data models such that inter-instance relationships can be properly represented. A class of physical data models are developed for efficient implementation of such a conceptual extension.

At the conceptual level, our scientific data model is based on a mathematical framework developed from two basic notions - *data-as-functions* and *pseudo-quasimetrics* - which are used for modeling inter-entity and inter-instance relationships respectively. By representing data as functions, the mathematical function formulations of data can be used for modeling inter-entity relationships. A detailed approach is outlined for exploring and deriving pseudo-quasimetrics to represent inter-instance relationships in a wide variety of data. In particular, we introduce the notion of *observable properties* and show how it can be applied with ideas from *point set topology*[2] [5] to systematically derive metrics from non-metric data components. Finally, we demonstrate the use of *continuity* as a mathematically precise tool to validate metrics derived through the proposed approach.

At the physical level, various *hierarchical metric data structures* can be utilized as physical data models to implement our metric-based conceptual model. In order to have efficient implementations, we propose an innovative approach to derive a new class of hierarchical metric data structures from *point spatial data structures*.[3] Instead of performing direct decomposition on *metric data* as is done for existing hierarchical metric data struc-

knowledge from data.

[2] *Point set topology* is also known as *general topology*.

[3] In point spatial data structures, the spatial objects to be stored are *discrete* points in multidimensional spaces, compared to other spatial data structures such as *region*, *rectangle*, or *line* spatial data structures which are designed for *continuous* spatial objects [54, 53].

tures such as *metric trees* [65, 64] and *vp-trees* [67], we define a class of simple *proximity-preserving* mappings from *pseudo-metric spaces* to *multidimensional spaces*, which we call *multipolar mappings*. By applying multipolar mappings to metric data, hierarchical decompositions can be done in multidimensional space, and various existing well-understood point spatial data structures [51, 52], such as *quadtree*, *octree*, or *k-d tree*, can be utilized as hierarchical metric data structures.

1.2 Taxonomy of Conventional Data Models

A *data model* is a collection of conceptual tools for describing data, relationships, semantics, operations, and constraints [43]. We can categorize data models by how they describe the data abstraction. Traditional database architecture partitions data models into three different levels of abstraction: *conceptual*, *implementation*, and *physical* [16]. Ideally, data abstraction at each level is independent of the ones in the other two.

Conceptual data models are based on the users' perceptions of data. They provide a high-level abstraction of the real world entities and the interrelationships among them. The *Entity-Relationship (ER) model* proposed by Chen [7] along with its various extensions [57, 61, 24] is the most widely used conceptual data model.

Physical data models specify the low-level organization of how data is stored in the computer. Physical data models are usually invisible to database users. It is the database system designer's job to define the physical data models and decide how they are implemented. Common physical data models include *B-tree*, *B^+-tree*, *B^*-tree*, *R-tree*, and so on.

Implementation data models serve as a bridge between conceptual and physical data models. They provide concepts at an intermediate level of abstraction still understandable by users but with enough detail to define the way data is logically organized within the computer. The interactions between users and database systems also take place at this abstraction level. The interface is implemented using a *DDL (data definition language)* and a *DML (data manipulation language)*. A DDL is used to specify the logical structures of data while a DML is used to access data stored in the logical structures specified by the DDL. Common implementation data models include *hierarchical*, *network*, *relational*, and *object-oriented*.

The description of a database in a specific data model is called a *schema*. While

ideally the choices of data models at the three abstraction levels should be independent from each other, in reality they are correlated. Generally, not every data model at one abstraction level supports all the models at the higher level(s) equally well. Even two different schemas of the same data model can have different requirements which are better served by different data models at the lower level(s). Thus, when a new data model is introduced or an old data model is extended, it is important to make sure that there exist data models at lower level(s) to provide the necessary support efficiently.

1.3 Inter-Instance Relationships

The focus of conventional data models is inter-entity relationships which are of primary interest in traditional data. However, scientific data also encapsulate complex inter-instance relationships. Let us illustrate such inter-instance relationships through a simple example.

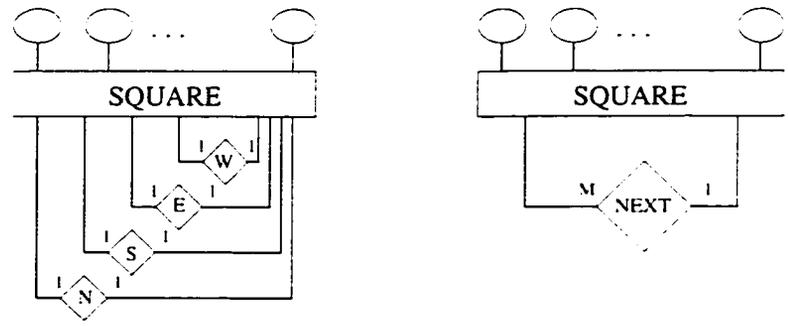
A multi-band land satellite image can be represented as a two-dimensional array of records with each of the records representing multiple measurements of a specific small land area. Each such record shares the same membership information of the semantic entity in question, in this case, the image or the 2-D array representation of the image. The membership information includes the format of the record and other relevant information such as the domain and possible constraints on each measurement. These are the structures and relationships which can be modeled by conventional conceptual data models. However, in addition to the membership sharing, records of the 2-D array are also interrelated spatially. While a conventional conceptual data model such as the ER model can associate each record with its spatial coordinates, the spatial relationship among records can not be adequately described.

Let us use the entity name SQUARE to represent the records of the 2-D satellite image. Using the ER model, the conceptual schema of the image is illustrated in Figure 1.1(a). The ER diagram shows that there are four possible relationships among instances of entity SQUARE - NORTH, SOUTH, EAST, and WEST - based on the spatial relationship among instances. However it fails to represent:

1. The way instances of SQUARE are related through NORTH, SOUTH, EAST, or WEST: For example, we can see from the ER diagram that relation NORTH is a one-to-one

relation and thus, no instance can be immediately to the NORTH of two different instances and no instance can have more than one instance as its immediate NORTH. Beyond that, we know nothing about the way instances are related through the NORTH relation. For example, the constraint that the NORTH relation introduces no cycles (i.e., there is no instance s such that s is NORTH of ... itself) can not be seen from the ER diagram.[4]

2. The way the relationships NORTH, SOUTH, EAST, and WEST are interrelated: For example, given two instances s_1 and s_2, we can not conclude from the ER diagram that s_1 is the NORTH of s_2 if and only if s_2 is the SOUTH of s_1. Neither can we say that the EAST of the SOUTH of an instance, if it exists, is the same as the SOUTH of the EAST of that instance.



(a) (b)

Figure 1.1: ER Schema Diagrams of the Satellite Image

The second issue is a classical example of one well-known limitation of the ER model - it is not possible to express relationships among relationships [43]. For conventional database applications, this limitation can be overcome by the introduction of the *aggregation* construct as an extension to the ER model. However, in the case of the 2-D satellite image, aggregation results in a single relation, say NEXT, by collapsing NORTH, SOUTH, EAST, and WEST (Figure 1.1(b)). This simply reduces the second issue into the first issue above. Various other extensions to the ER model have been proposed to alleviate this inherent

[4] In fact, even implementation schema derived from this ER diagram can not enforce this constraint efficiently. A violation of this kind can not be detected by checking the content of a specific instance; it is necessary to traverse all instances in the transitive closure.

limitation [61, 62, 24]. None of them succeeds in providing an effective mechanism to describe complex inter-instance relationships in general.

In summary, ER diagrams can not represent the regularities, constraints, or patterns of the relationships among instances of an entity. For example, from the ER schema, it is not possible to tell the differences among the three distinct patterns in Figure 1.2. Since making relationships explicit is one of the most basic objectives of data models – at the conceptual level in particular – conventional conceptual data models are inadequate as models for scientific data. Clearly, we need a conceptual scientific data model that supports the representation of inter-instance relationships. This is the principal contribution of the *metric-based scientific data model* described in this thesis.

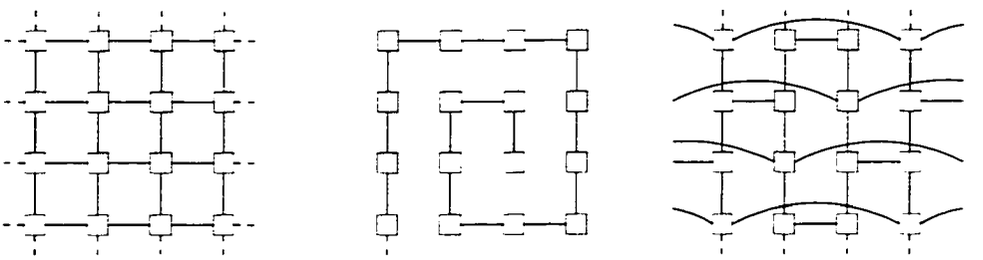


Figure 1.2: Different Patterns of Inter-Instance Relationships

As mentioned in Section 1.2, the development of a new conceptual data model may necessitate the development of new data models at the two lower abstraction levels. The development of supporting data models for our metric-based scientific data model at the implementation and physical levels is discussed later. In particular, the development and analysis of supporting physical data models, i.e., *hierarchical metric data structures*, form an important component of this thesis.

In the rest of this thesis, the general term "scientific data model" is used in place of "conceptual scientific data model" or "scientific data model at the conceptual level."

1.4 Models of Inter-Instance Relationships

While there exist differences in terminology and implementation, existing scientific data models are all based on the same conceptual model of inter-instance relationships – the *geometric model*. Depending on the complexity of the inter-instance relationships, three primitives, *geometry*, *topology*, and *indexable topology*, can be used to support the geometric

model at the implementation level (Chapter 2). Each of the existing scientific data models incorporates some or all of the three *implementation primitives*. There also exists a group of data structures to support each of these notions in a physical data model. Figure 1.3 illustrates a hierarchy of data models for inter-instance relationships, where each data model is represented by a rectangle. Instead of representing each of the existing scientific data models, only the modeling primitives employed, represented by ellipses, are illustrated in the implementation layer. In general, the more expressive and flexible the data model and the modeling primitive, the wider the range of data that can be modeled, but the less efficient the supporting data structures.

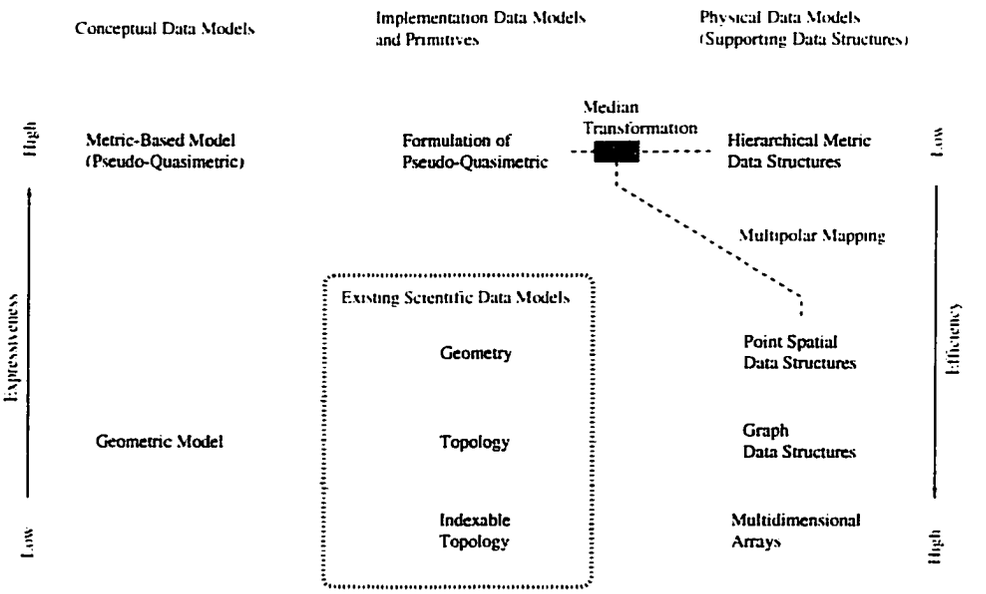


Figure 1.3: Data Models and Primitives for Representing Inter-Instance Relationships

In this thesis, we propose the use of a special data function, the *pseudo-quasimetric*, as a basic conceptual modeling primitive for representing inter-instance relationships (Chapters 3 and 4). In terms of expressiveness, a pseudo-quasimetric is more powerful than the geometric model and can be applied to a much wider range of scientific data. While there are no data structures readily available to support pseudo-quasimetrics as a conceptual model, there exists a group of *hierarchical metric data structures*, such as *metric trees* [65, 64] and *vp-trees* [67], which can be utilized to support *pseudo-metrics*, a more restrictive notion than pseudo-quasimetric. In Chapter 5, we propose an innovative approach, the *multipolar mapping*, to use point spatial data structures for supporting pseudo-metrics. In addition,

we develop a simple and practical technique, *median transformation*, for utilizing existing hierarchical metric data structures as well as multipolar mappings to support pseudo-quasimetrics.

It should be noted that the term *topology* has dual interpretations in this thesis. The first interpretation is based on the conventions used and understood by researchers and practitioners in various fields of computer science. Based on this interpretation, a topology is a description of connection patterns (Section 2.2). Formally, it can be represented as a *directed graph*. For instance, we can talk about the topology of a computer network, or the topology of a computational grid. The second interpretation of topology in this thesis is based on its formal definition in general topology (Section 3.4). In this sense, we can talk about things such as *the usual topology of the real numbers*, or *neighborhood structure of a topological space*. Nonetheless, there exists a connection between the two interpretations namely, both interpretations provide a way to describe the idea of "closeness," "nearby," or "neighborhood." In this thesis, the intended interpretation of the term topology is obvious from its context.

1.5 Thesis Outline

Chapter 2 presents a survey of existing scientific data models and the primitives employed for representing inter-instance relationships, namely, *geometry*, *topology* and *indexable topology*. Four models are selected based on the criterion that each one can represent inter-instance relationships explicitly based on one or more of these primitives. While none of these models adequately addresses the problem of modeling inter-instance relationships, the survey provides a framework for the proposed metric-based scientific data model.

In Chapter 3, we give a brief introduction to the mathematical foundation of the metric-based model. Some basic terminology and notions from *general topology* and *metric theory* are covered there. The main theme is the study of semantics for *continuity* in different mathematical settings. In the subsequent chapters, the notion of continuity plays an important role in both the metric-based model and the supporting data structures.

Chapter 4 describes our proposed metric-based data model. Several examples are given to illustrate the basics. A systematic approach is developed to derive metrics from non-metric attributes such as the ones in categorical data. Last but not the least, we present a formal method for validating the derived model.

Chapter 5 is devoted to the study of physical data models, i.e., supporting data structures, for the proposed metric-based data model. Two innovative approaches, *multipolar mapping* and *median transformation*, are developed for this purpose. They enable the applications of existing *hierarchical metric data structures* and *point spatial data structures* as physical data models for the metric-based model.

In Chapter 6, we present a performance study on major parameters of the multipolar mappings. In particular, we are interested in learning how these parameters affect efficiency of *proximity queries* in data sets of various characteristics. The results give us a simple guideline to formulate an effective multipolar mapping for a wide range of data.

Chapter 7 provides concluding remarks and directions for future research.

Chapter 2

Scientific Data Models

2.1 Introduction

The need for modeling scientific data with complex inter-instance relationships has long been recognized by the *computational fluid dynamics* and *scientific visualization* communities. Notable work includes the *computational grid model* [40, 60] and its various extensions, the AVS model of Gelberg et al. [23], the *fiber bundle model* of Haber, Lucas and Collins [28], and the *lattice model* of Bergeron and Grinstein [2, 35]. From the perspective of modeling inter-instance relationships, they all share the same *geometric model* at the conceptual level. While there exist differences in terminology and methodology, the objective for all the existing scientific data models at the implementation level is to derive a compact representation of "structured data" or "data with regular patterns" based on some *adjacency relations* in the *logical space* on which a mapping to the *physical space* is defined. For instance, in a 2-D logical space, *four-neighbor adjacency* and *eight-neighbor adjacency* are two common adjacency relations. These models are not capable of efficient representation of highly complex inter-instance relationships where closed form adjacency relation specifications are not available.

Advances in *spatial data models* [54] present a different approach for modeling inter-instance relationships. In a spatial data model, the inter-instance relationships are represented as a set of coordinates in a *vector space*. Spatial data models are extremely useful as the foundation for *spatial data structures* such as the *quadtree* or *R-tree* [52, 53], which have numerous applications [51]. However, for non-spatial data, there might not exist a natural mapping from data point records to some vector space such that the inter-instance

relationships are represented by coordinates in that vector space.

It should be noted that there also exists a group of scientific data exchange standards which do not fall into the three layer data model taxonomy. Nonetheless, in a subtle way, their formats do represent implicit conceptual views of the data. The most popular two such data exchange standards are CDF (Common Data Format) developed at NASA [49, 26, 45] and HDF (Hierarchical Data Format) developed at NCSA [46]. The conceptual models implicitly defined by both CDF and HDF are encompassed by the general computational grid model (Section 2.4). Data exchange standards such as these are specifications of file formats which can be considered as scientific data models for file systems.

This chapter summarizes the four scientific data models which represent inter-instance relationships explicitly: computational grid model, AVS model, fiber bundle model, and lattice model. Before we get into details of the four respective data models (Sections 2.4, 2.5, 2.6, and 2.7), it is useful to study the general concepts of data organization (Section 2.2) and common operations performed on scientific data (Section 2.3).

2.2 Implementation Primitives and Physical Data Models

A *scientific data set* is a finite set of *data points*. Each data point corresponds to observations made at a point or small area in the *physical space* at a specific time or period. A physical space can be a volume of atmosphere, a magnetic energy field, a specific biological population, or a computer simulation of such physical entities. Each data point is represented by a record of attributes reflecting observations made at that point. Since the physical space is where the data sampling takes place, it is also called the *sampling space*.[1] Since we are only interested in scientific data, the term *data set* is used instead of *scientific data set* in the thesis.

At the conceptual level, existing scientific data models are all based on the geometric view of the physical space - the *geometric model*. For a given data set, while there is exactly one physical space, there can be many different geometric views of the same data set. The possibility of multiple geometric views enables us to explore implicit interrelationships in data, both inter-entity and inter-instance, without the limitation imposed or suggested by the process used for sampling data in the physical space. Among the existing scientific data models, the *lattice model*, in particular, is specially designed to facilitate

[1] The physical space is also known in the literature as the *physical domain* or the *physical object* [40].

multiple geometric views (Section 2.7).

While the existing scientific data models differ in terminology and objectives, they are all based on one or more of the three *implementation primitives* identified in this section for representing the inter-instance relationships at the implementation level. These primitives are *geometry*, *topology*, and *indexable topology*. In terms of the complexity of the inter-instance relationships they are capable of expressing, the three notions form a hierarchy with geometry being the most powerful and indexable topology being the least. This kind of expressive power comes at the price of efficiency, however. The notions of geometry, topology, and indexable topology provide a framework for studying the four scientific data models presented in this chapter and our proposed metric-based model. One important objective is to derive suitable physical data models, i.e., data structures, for each of these primitives. While there is a group of data structures for each of these primitives, a scientific data model might need to be supported by more than one kind of data structure at the physical level. Figure 1.3 illustrates the relationships among all the data models, primitives, and supporting data structures discussed in this section.

2.2.1 Geometries

The term *geometry* has many different meanings. Given a data set, we define *geometry* as an implementation primitive for its inter-instance relationships, i.e., the interrelationships among its data points, based on a direct interpretation of the conceptual geometric model. Specifically, a geometry consists of two components:

1. a mapping from data records to coordinates in the geometric space,
2. a distance function in the geometric space.

Essentially, geometry describes the interrelationships among data points based on their locations in the geometric space. Geometric information is encoded in the form of attributes of data points, *metadata* of the data set, and the geometric view supplied by the user. Attributes selected for representing geometric coordinates are called *geometric attributes*. A data set can have many different geometries. Different sets of attributes might be used as geometric attributes, resulting in different geometric spaces, and thus, different geometries. For the same geometric space, different choices of distance functions also induce different geometries.

Note that for many data sets, there might not be a meaningful geometric representation for its inter-instance relationships. While there is always a natural geometric view for a spatial data set, the physical space for a non-spatial data set might not be a coordinate space - for instance, it might be a *metric space* or a *binary relation*. A geometric view and a geometric representation, i.e., a geometry, might not be readily available in such cases.

The most obvious way to store a data set based on its geometry is to utilize some *point spatial data structure* [54, 53] encoding that geometry. Point spatial data structures, however, are generally not as efficient as multidimensional arrays where each data record can be accessed by its numerical index. For many data sets with a "simple" or "regular" geometry, it is often possible to derive a better data structure for its access and storage. The reduction in complexity results in the notions of *topology* and *indexable topology*.

2.2.2 Topologies

A *topology*[2] is a directed graph defined on a set of data points based on a chosen geometry. Theoretically, a topology is equivalent to a distance function which takes only two values, 0 and 1. There exists an arc, i.e., directed edge, from p to q in a topology if and only if the distance from p to q is 0. The mapping from data records to geometric coordinates in the geometric space, i.e., the first component of a geometry, is not part of a topology. A topology is usually derived to achieve one of the following two objectives:

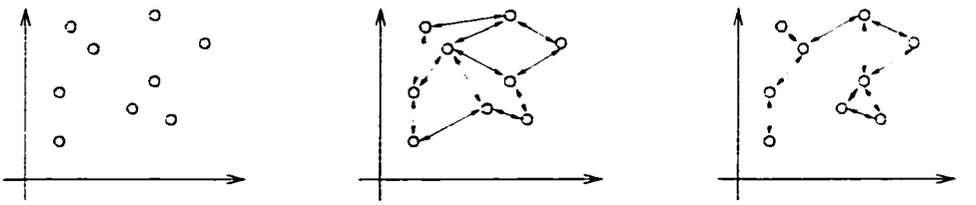
1. the geometry can be traversed or explored more efficiently through the topology [60].

2. specific computational procedures can be applied to the data more efficiently [40].

Depending on the requirements of an application, more than one topology can be derived from the geometry. Nevertheless, the existence of an arc from a point p to a point q usually implies that q is relatively close to p (it does not necessarily imply that p is close to q). Under most circumstances, the distance function specified by a geometry is *symmetric* (Section 3.2) and so is the simplified two-value distance for the topology derived from it. Thus, undirected graphs suffice to describe most topologies. For the rest of the thesis, unless specified otherwise, we assume undirected graphs for topologies.

In general, a useful topology for a data set is a trade-off between the following two factors:

[2]Refer to Chapter 3 for the second interpretation of the term *topology*.



(a) (b) (c)

Figure 2.1: Two Different Topologies Derived from the Same Geometry

1. The topology closely reflects the geometry.

2. The topology consists of mostly simple and repetitive patterns such that data access and/or computation are facilitated.

Figure 2.1(a) illustrates the geometry of a data set, assuming Euclidean distance. For the convenience of computation and access, we might derive a topology as depicted in Figure 2.1(b). As can be seen, the edges roughly indicate closeness between points in the geometry, with a few exceptions. Figure 2.1(c) depicts another topology introduced by setting a distance threshold such that two points are connected by an edge if and only if their distance is lower than the threshold. Although this topology represents the geometry in a more precise way - the existence of an edge represents a certain degree of closeness, as specified by the threshold - it does not consist of regular patterns to facilitate computation or storage access. Thus, it might not be as useful as the first topology.

The derivation of a topology which closely reflects the geometry is usually achieved through a series of refinements. The initial topology is often constructed from some variant of the basic notion of *nearest neighbor*. For instance, we can construct an initial topology by connecting each point to its nearest neighbor, or the nearest k neighbors. The approach for deriving Figure 2.1(c) is also based on a variant of the nearest neighbor notion.

For a data set where the inter-instance relationships can be adequately modeled by a topology, it is often better to derive a data structure using such a topology than to utilize a point spatial data structure based on its geometry. Each edge in the topology can be implemented as a bi-directional pointer from one data point to another. We can also label each point with a unique index and store the *adjacency matrix*. Either way, access to data points still often requires some sequential traversal of part of the topology, and random

access might be very expensive. If the topology of a data set is highly regular, it is possible to model its geometry with a special class of topologies, *indexable topologies*, that enable especially efficient access.

2.2.3 Indexable Topologies

Data are stored and retrieved by computers based on indexes. Mathematically, the process of indexing corresponds to mapping each *unit* of data to elements in an *index set*. An index set does not have to be equipped with any structure on its elements. However, in practice, we do want some mathematical structure on an index set, so data can be organized in a more meaningful way and be efficiently manipulated.

In many scientific and engineering applications, it is a common practice to map data points in a physical space, to a simpler *logical space* to simplify data analysis and computation. For instance, a manifold in a three-dimensional physical space can be mapped to a rectangular area of a two-dimensional logical space. In this section, we study mappings from the geometric space - recall that multiple geometric spaces can be associated with one physical space - to discrete logical spaces in which each coordinate can only take integer values. Such logical spaces are known in mathematics as *vector spaces over the integer field*. For convenience, we call such logical spaces *index spaces*.

The objective is to find a mapping such that the data points are uniformly distributed and properly aligned with each axis in an index space. Thus, the coordinates of the index space can serve as indexes for points in the data set such that a multidimensional array can be used for its storage. Specifically, we are looking for a mapping possessing the following three qualities:

1. The mapping should be an *injection*, i.e., *one-to-one*. Thus, every point in the geometric space is mapped to a unique point in the index space.[3]

2. A mapping should be *continuous*, so that data points close in the geometric space stay close in the index space.[4]

[3] In reality, this criterion can be relaxed to a certain extent. A limited number of data points in the geometric space can be mapped into a single indexing point in the index space. Examples are *hashing* or the *bucket* varieties of many data structures.

[4] However, the reverse might not be true. Points close in the index space are not necessarily close in the geometric space.

3. The logical coordinates of all the data points should be uniformly distributed in a "rectangular" area (not necessarily two-dimensional) in the index space. However, there might exist a relatively small number of points in this area which are not mapped into, corresponding to unassigned indexes. This quality enables us to specify a range of valid index access for each dimension of the index space.

We call mappings having all three qualities *index mappings*. Note that in the second criterion, continuity is defined in the context of some meaningful *neighborhood structures* on both the geometric and index spaces (see Chapter 3 for the definitions and semantics of continuity and neighborhood structure). Under most circumstances, these neighborhood structures are induced by some distance measure.

Figure 2.2 illustrates an index mapping. As can be seen, the coordinates on the index space correspond to the indexes of a two dimensional array.

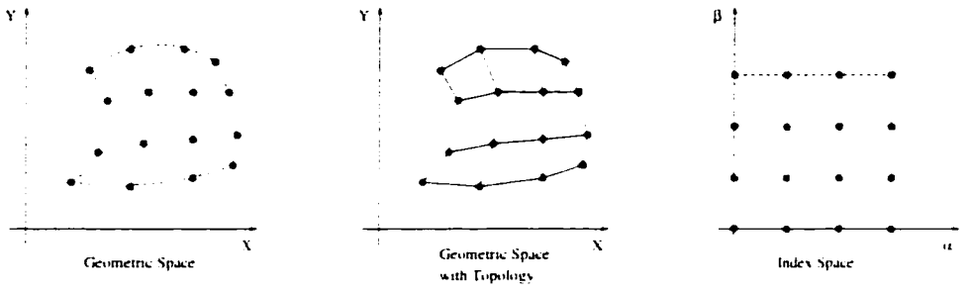


Figure 2.2: Geometric Space and Index Space

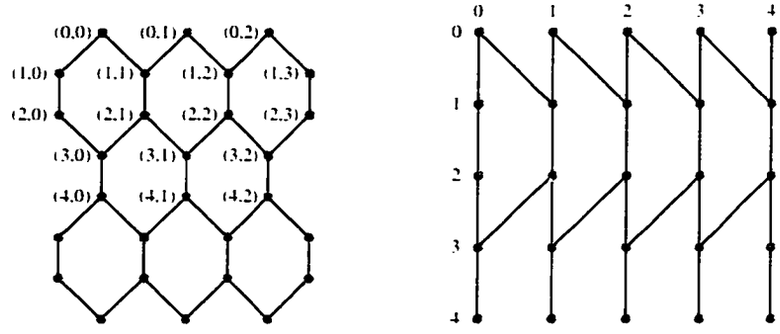
For a data set X with a topology consisting of regular patterns, points in the data set can be indexed by a set of coordinates in the index space such that a closed form *connection rule*, $C : X \rightarrow \mathcal{P}(X)$, can be defined to describe the adjacency relations, i.e., arcs or edges, in the topology.[5] Unless otherwise specified, the connection rule among coordinates is based on the *four-neighbor* adjacency relation of 2-D grids and its analogue in higher dimensions. For example, a 3-D grid point (i, j, k) is connected to six neighboring points: $(i-1, j, k)$, $(i+1, j, k)$, $(i, j-1, k)$, $(i, j+1, k)$, $(i, j, k-1)$, and $(i, j, k+1)$. Not all of the neighboring points have to exist. This is the implicit connection rule for the topology of Figure 2.2. A topology having this kind of connection rule is known as the *structured grid* in the computational grid model (Section 2.4). Connection rules other than the default one, such as the *eight-neighbor* adjacency relation, can also be specified.

[5] $\mathcal{P}(X)$ represents the *power set* of X.

Given a data set, the union of an index mapping and a connection rule uniquely specifies a topology - an *indexable topology*. Not every topology can have such a compact representation as an indexable topology, however. The directed graph implied by the index mapping and connection rule of an indexable topology need not be the same as the one described by the topology from which the indexable topology is derived. Nevertheless, from the graph perspective, an indexable topology is usually a *subgraph* of the topology used for its derivation.

Topologies other than the orthogonal grid illustrated in Figure 2.2 can also be indexed. In Figure 2.3(a), we give an example of a topology which is not an orthogonal grid. The connections are also illustrated on the 2-D index space in Figure 2.3(b). For this example, the connection rule is defined as follows.

$$C((i,j)) = \begin{cases} \{(i-1,j),(i+1,j),(i+1,j+1)\} & \text{if } i \bmod 4 = 0, \\ \{(i-1,j-1),(i-1,j),(i+1,j)\} & \text{if } i \bmod 4 = 1, \\ \{(i-1,j),(i+1,j-1),(i+1,j)\} & \text{if } i \bmod 4 = 2, \\ \{(i-1,j),(i-1,j+1),(i+1,j)\} & \text{if } i \bmod 4 = 3. \end{cases}$$



(a) (b)

Figure 2.3: Non-Default Connections

2.2.4 Structured Data

There are data sets whose data points exhibit a repetitive pattern only along a proper subset of all the dimensions. An index mapping on this kind of data set might introduce dimensions in index space in which there are no connections between coordinates

along that direction. Indexable topologies can still be useful in such cases where repetitive patterns in topologies are "sparse."

Figure 2.4(a) illustrates a data set consisting of 21 data points. Although the data points appear to be distributed randomly on the X-Y plane, they do exhibit regular patterns along the Z axis. Since there is no meaningful index mapping which can map points on the same X-Y plane to a two-dimensional index plane, we have to map those points to a sequence of one-dimensional indexes. For instance, we can start with an arbitrary point, assigning it an index 1, and proceed to its nearest neighbor, assigning it an index 2, and so on (Figure 2.4(b)). Through this indexing, the data set can be mapped to a two dimensional index space with axes α and β corresponding to indexes on the X-Y plane and along the Z axis respectively (Figure 2.5(a)). The indexes on α may not reflect the closeness relationship among points. For instance, the point with $\alpha = 7$ is actually closer to the point with $\alpha = 1$ than the one with $\alpha = 5$ of the same X-Y plane in the geometric space. It should be noted that all the points are still uniformly distributed and properly aligned with each axis in the index space.

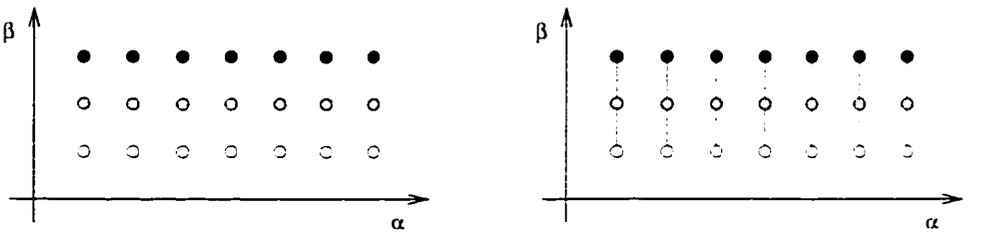
(a) (b)

Figure 2.4: Data Set with Pattern along One Dimension Only

Figure 2.5(b) illustrates an indexable topology of the data set based on the following connection rule:

$$C((i,j)) = \{(i-1,j),(i+1,j)\}.$$

While random access to individual data points on the X-Y plane (i.e., along the α axis of the index space) is not possible based on this indexable topology, random access along the Z axis (i.e., the β axis of the index space) can be easily achieved. Retrieval of all points on the same slice is also possible.



(a) (b)

Figure 2.5: Index Space for the Data Set in Figure 2.4

A data set is *structured* if a subset of its topology can be represented by a closed form connection rule. Otherwise, it is *unstructured*. In other words, the data set is considered structured if an indexable topology can be defined on it. Note, however, that if a good portion of important edges in the topology are not represented in the indexable topology, the indexable topology may not be useful. If no useful indexable topologies can ever be derived from any meaningful topology on the geometry, we can consider that data set to be unstructured with the respect to the given geometry.

Intuitively, a data set is considered structured, if its geometry can be described by a topology consisting of regular patterns. This translates to the existence of a "natural" indexing scheme for its points based on the topology such that a multidimensional array can be efficiently used for its storage. Continuity of the index mapping assures that such an indexing scheme is natural; that is, points close in the geometric space stay close in the index space. For structured data, there might be more than one useful index mapping and thus, more than one natural indexing scheme.

The actual data structures to be employed for data storage and access are determined by the topology of the data set. For a structured data set, where there is a useful indexable topology, an array structure can be incorporated as part of the data structure for the data set. Suppose the indexable topology described in Figure 2.5(b) is only part of the topology, where there are connections along the α axis, unspecified in the figure. Let G denote an unspecified data structure which can be used to store data points on the same X-Y plane according to the topology.[6] The data set can be stored as a one dimensional array of three elements where each element is an instance of G. Thus, random access of a

[6]Many approaches have been proposed for efficient graph representations.

slice of data along the β axis is made possible. A useful indexable topology enables us to have at least some degree of random access in a structured data set.

In practice, given a structured data set, its *dimension* usually refers to the dimension of its index space or the subset of the index space that allows random access. The dimension of an unstructured data set is considered to be zero. Since there might be more than one useful index mapping for a particular data set, the common notion for dimensionality of a data set is not well-defined. This problem is addressed in the lattice model [2, 35] (Section 2.7).

The real advantage of the indexable topology lies in the ability to utilize multidimensional arrays for data storage and access. While there are many efficient data structures around, the multidimensional array remains the most popular since it closely resembles the way data is actually stored by a computer and its cells can be efficiently accessed based on simple calculations on their indexes.

2.3 Basic Operations

Since the ultimate objective of a scientific data model is to facilitate scientific data analysis in a scientific database environment, it is important to identify the basic operations on data involved in the process of scientific data analysis. Ideally, a scientific data model should enable effective and efficient implementation of these operations. In this section, we focus on several generic types of operations on scientific data.

2.3.1 Subset Operations

Subset operations correspond to SELECT and PROJECT operations in relational algebra. It can be argued that subset is the most important operation that can be performed on scientific data. There are many different types of subset operations, depending on how the selection criteria are formulated. Nonetheless, they can be classified into three major categories: *topological subset*, *geometric subset*, and *value subset*. The topological subset operations perform the selection based on the topology of the data set. Geometric subset operations perform the selection based on the geometric coordinates of data points. Value subset operations depend upon the data values at each point.

Similar to the relational PROJECT, a topological subset operation has the potential to reduce the dimensionality of a data set. For structured data, it is usually relatively

straightforward to implement and often involves simple manipulation on the array indexes. For data with irregular or complex topology, topological subset operations correspond to partial traversals of a graph. The result of a topological subset operation is usually a connected subgraph of the topology of the data set. The general procedure for performing topological subset operations on complex topologies consists of the following two steps:

1. Locate the seed: A seed is the first identified point belonging to the target subset. It is usually discovered by calculation on the indexes of the topology. If such index calculations are not possible, as with unstructured data sets, a seed has to be located by an index search starting from an arbitrary point of the topology. Since only one seed is required, *depth-first search (DFS)* is preferred over *breadth-first search (BFS)*.

2. Determine the subset: The target subset can be found by either a DFS or BFS starting from the seed.

The geometric subset operation is usually the computationally most intensive unless the geometry is closely mirrored in the topology. It involves the processing of geometric attributes at each data point. The general procedure for geometric subset operations is similar to the one for topological subset. The searches performed in each of the two steps are now guided by values of geometric attributes. One popular geometric subset operation is known as *geometric slicing*, which is used for visualizing high dimensional data sets.

An important geometric subset operation is *proximity query* which originates from the spatial data models. Due to recent advances in utilizing database systems for access and storage of multimedia (e.g., images, audio, and video) and non-standard data (e.g., finger prints and DNA sequences), the applications of *proximity query* are no longer limited to spatial data. Note that while the proximity query is considered as a geometric subset operation, it does not rely on mapping of records to the geometric space, but is based exclusively on the distance function defined in the geometric space. Compared to normal geometric operations, proximity query can be applied to a much wider range of data sets. The complex inter-instance relationships in multimedia and non-standard data, or scientific data in general, suggest:

1. There might not be a meaningful geometric representation of the inter-instance relationships and ordinary geometric operations can not be applied.

2. Although there might exist a geometry, one or more axes of the geometric space is unordered or nonmetric. As a result, most geometric notions, such as slicing, might not be well defined in the geometric space.

3. While there might be a geometry and most geometric operations are applicable, there is usually no obvious reduction to a topology which can properly model the geometry. Thus, topological subset operations might not be available or useful to implement geometric subset operations.

For the reasons just described, the applications of proximity query are proliferating [48, 39, 3]. In fact, the proximity query is an integral part of the proposed metric-based scientific data model.

The value subset operation is equivalent to the relational SELECT operation. Although it is conceptually easy, special considerations must be made for efficient implementation. Building indexes on attributes frequently involved in selection criteria is one common technique. Applications of value subset operations include statistical analysis, outlier identification, etc.

The boundaries between the three kinds of subset operations are not always well-defined. The categorization depends on the nature of the data set as well as the data structure chosen for its storage and access. For instance, as is stated in Section 2.2.1, different sets of attributes might be used as geometric attributes, resulting in different geometries. Thus, a value subset operation in one implementation can be a geometric subset operation in another. It should also be noted that there exist procedures, such as iso-surface location and boundary detection, which may involve all three kinds of subset operations.

2.3.2 Analysis and Exploration

Analysis and exploration is one important activity to be supported by scientific database systems. Techniques such as *resampling*, *scaling*, *translation*, and *rotation* are often applied to the raw data set such that subsequent data analysis routines can be successfully performed.

Given a data set with its geometry specified, resampling is a procedure to derive new data sets from the original one with different geometries. The derived sets may contain more or fewer data points than the original one. Although they have different geometries, the

new data sets produced by the resampling process share the same *geometric interpretation* of the original data set, because the resampling is performed on the original geometry. The topologies of the new data sets may be different from that of the original data set. Resampling is usually applied to derive a new data set with the kind of geometry or topology that may be required by specific data analysis procedures. Resampling can also be applied to each one of a collection of data sets for normalization. Thus, operations such as merging can be performed between data sets.

Various techniques can be used for resampling. Among them, *interpolation* is the most common. Interpolation is based upon the assumption that the domain to be interpolated is continuous in nature. Otherwise, there is no foundation to assume "good" behavior between data points, and the result of interpolation is meaningless. There are various kinds of interpolation methods. Mathematically, interpolation is a function defined for data points in the *neighborhood* of the point to be interpolated (see Chapter 3 for the definition of neighborhood). In addition, interpolation is a common technique used to fill missing data values.

Operations like scaling, translation and rotation are also based on a specific geometry of the original data. The results can be viewed as new data sets with geometries having the same *geometric interpretation* of the original one or just as new geometries of the same data set.

Among all data analysis approaches, statistical analysis is the most commonly used. Statistical operators are often applied to subsets of the data. Although they typically do not rely upon the geometry of the data set, they are essential for exploratory data analysis, in which the interrelationships among data values are to be found. In addition, they can also give us information to measure the quality of the data set. Statistical operations are often applied after a resampling or subset operation.

Other data analysis techniques include *FFT* [66], *wavelets* [13, 17], *syntactic pattern recognition* [22] (useful in *scene analysis*), and various classification and clustering techniques.

2.3.3 Miscellaneous Database Operations

All database retrievals are just subset operations. Other database operations include *updates*, *indexing*, and *joins*. Updates in scientific data models are similar to the ones

in conventional data models. In addition to the normal indexing applied to data values, indexes can be built on the topology to make data retrievals in scientific data models more efficient. For structured data which can be stored in a multidimensional array, the array indexes can serve as the indexes of the topology.

Join is a classic operation of the relational data model. It is used to correlate one relation with another. Due to the differences between conventional data and scientific data, the notion of a relational join operation has to be extended to merge two scientific data sets. If two data sets share the same geometry (which directly implies that they have the same geometric space) they can be easily joined, i.e., merged. For data sets with different geometries, meaningful joins can still be done by applying one of the three following techniques:

1. Resampling: Resampling can be applied to either one or both of the two data sets in order to convert them to the same geometry.

2. Revised Join Condition: Instead of joining two data points at exactly the same geometric location, we can redefine the join condition such that two or more data points can be joined if they are sufficiently close in the geometric space.

3. Outer Joins: Join operations only match data points satisfying the join condition. Hence, data points of a data set without "related" points in the other data set participating in the join operation are eliminated from the result. Instead of eliminating all these "unrelated" points in both data sets, *outer joins* keep all such points in one or both of the data sets. There are three kinds of outer joins: *left*, *right*, and *full* outer joins [16]. Left and right outer joins keep all "unrelated" points of one specific data set, while full outer joins keep all such points in both data sets. Full outer joins can be performed on two data sets with very different geometries without losing data. Outer joins are also useful when one of the two data sets to be joined is significantly smaller than the other.

2.4 Computational Grid

Computational grid generation arose from the need to compute solutions to partial differential equations of *computational fluid dynamics (CFD)* on physical spaces with

complex geometry [40]. It is also useful for CFD visualization and general scientific visualization as well. A computational grid describes a mapping from the logical space to the physical space. It depicts the interrelationship among data and is useful for interpolation, slicing, and other data manipulation or visualization purposes.

Computational grids are user-imposed topologies on data sets to simplify computation and storage. This is exemplified by the original application for which computational grids were developed, computational fluid dynamics. Various kinds of grids have since been utilized by people in the CFD and visualization communities. The terminology is often conflicting in the literature. We choose to base our discussion on the terminology and work of Speray and Kennon [60].

2.4.1 Taxonomy of Computational Grids

Seven types of computational grids are identified by Speray and Kennon [60]. They are presented in order of increasing generality and complexity. In the list below, the names of most of the grid types are followed by the mapping from the indexes of grid points to corresponding coordinates in the physical space. *Block structured* and *hybrid* grids can not be formulated as such simple mappings, however. For convenience, only the mapping of 3-D grids is presented. Mapping for grids of other dimensionality are similar. Let (i, j, k) represent the index for a grid point of a 3-D grid.

1. CARTESIAN (i, j, k)

   A Cartesian grid is a typical 3-D matrix with no explicit physical coordinates, so indexes map identically to physical space.

2. REGULAR $(i\,dx, j\,dy, k\,dz)$

   Along each axis Λ, grid points are equally spaced at intervals of size $d\lambda$. The intervals on different axes do not have to be the same. Cells are identical rectangular prisms (bricks) of size $dx \times dy \times dz$ aligned with the axes.

3. RECTILINEAR $(x(i), y(j), z(k))$

   Along each axis Λ, grid points might not be equally spaced and the physical coordinates of axis Λ are determined by a mapping $\lambda(\Lambda)$. The function λ is *strictly increasing*, i.e., $i_1 < i_2 \Leftrightarrow \lambda(i_1) < \lambda(i_2)$. Cells are still rectangular prisms aligned with the axes, but of different sizes.

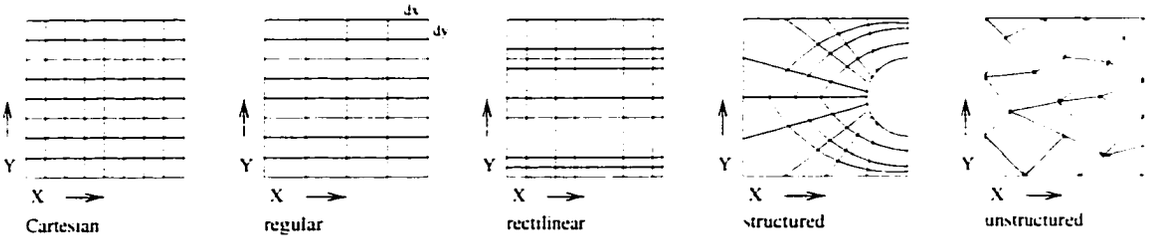


Figure 2.6: Various 2-D Grids

4. STRUCTURED ($x(i,j,k), y(i,j,k), z(i,j,k)$)

   A structured grid is logically a Cartesian grid which is subjected to non-linear transformations so as to fill a volume or wrap around an object. This type, also known as *curvilinear*, allows a non-boxy volume to be gridded.

5. BLOCK STRUCTURED

   A block structured grid is a composition of several different structured grids. Each component, i.e., block, of a block structured grid is a structured grid by itself and has its own mapping function from the logical space to the physical space. Extra information has to be provided as metadata to specify the connections among blocks - in particular, the connections from the boundary points of one block to the boundary points of the adjacent blocks.

6. UNSTRUCTURED ($x(i), y(i), z(i)$)

   An unstructured grid is represented as a list of points. There is no implicit connectivity and no implied topology. Connectivity must be supplied in some other form. Speray and Kennon also assume that the connections of points in an unstructured grid form a homogeneous collection of non-overlapping cells such as tetrahedra, hexahedra, prisms, pyramids, etc. [60]. However, there need not be a uniform pattern for the connections among cells.

7. HYBRID

   A hybrid grid is a composition of structured and unstructured grids. Similar to the block structured grid, metadata should be supplied to specify the connections among components.

Figure 2.6 shows 2-D grids for some of the above varieties.

2.4.2 Computational Grid Summary

From the database perspective, the operations on structured grids are straightforward, since they are based on a common data structure - a multidimensional array. Value subset operations can be easily achieved by a simple database SELECT operation. Topological subset operations are also easy and efficient, only involving the manipulations of array indexes.

Geometric subset operations can be achieved by a single value subset operation based computation done on the values of geometric parameters. This requires visiting every single grid point and deciding whether or not it is a member of the target subset. For efficiency, geometric subsets can be computed by a sequence of topological and value subset operations as follows:

1. Locate an arbitrary point or points of the target subset either through some indexes or database SELECT operations.
2. Perform a topological subset operation to retrieve points topologically close to the points located in the previous step.
3. Perform a value subset operation on the topological subset from the previous step.
4. Repeat Steps 2 and 3, as necessary.

Topological and geometric subset operations are significantly more expensive on unstructured grids which usually do not allow efficient indexing on topology. Theoretically, such operations might require the traversal of all data points.

Computational grids can be considered as the reverse of index mappings. There exist topologies which can not be modeled by any of the seven grids. Figure 2.7(a) illustrates such a topology. In order to model the same data set as a grid, a different topology has to be induced from the geometry. Figure 2.7(b) shows such a topology, which consists of a homogeneous collection of non-overlapping triangles.

All structured and block structured grids are considered structured data based on our definition (Section 2.2.3). However, unstructured grids exhibiting some regular patterns can also be mapped to an index space with at least one dimension of usual distance by an index mapping. One such topology is illustrated in Figure 2.7(c). In fact, even topologies without grid representation can be structured (Figure 2.7(d)).

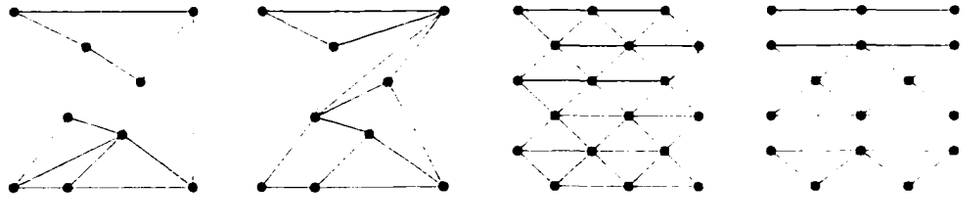


Figure 2.7: Topologies Which Can Not Be Modeled by Computational Grids

The taxonomy of computational grids presented here is quite standard in the computer graphics and visualization community with much of its popularity due to the simple underlying data structure, the multidimensional array, normally used to implement structured grids. The majority of current conceptual scientific data models incorporate some notions of computational grids, either implicitly or explicitly, as part of their framework. The lattice model (Section 2.7), in particular, is a notable extension of computational grids.

2.5 Application Visualization System (AVS)

Gelberg et al. [23] present a set of data structures and corresponding algorithms for visualizing scientific data in AVS (Application Visualization System), a general purpose visualization environment [15]. We loosely call the set of conceptual data structures defined in [23] as the *AVS data model*. AVS supports two major classes of data - structured and unstructured. In order to distinguish the two classes of data from the structured and unstructured data defined in Section 2.2.3, we call the ones described in [23] *AVS structured data* and *AVS unstructured data*, respectively.

2.5.1 AVS Structured Data

AVS structured data has two components - a *logical* organization of data elements into multidimensional arrays, and a *physical* mapping of each data element into the geometry under study. There are three types of AVS structured data: *uniform*, *rectilinear*, and *irregular*. While all three data types have an underlying logical structure based on a regular array, they vary in terms of the representation of the physical mapping.

Uniform data is orthographic, with constant spacing between the nodes. It has an

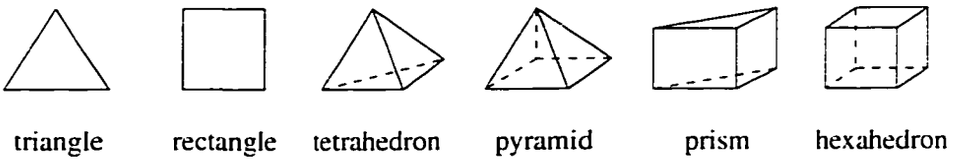


Figure 2.8: Primitive Cell Types

implied physical mapping identical to its logical structure. This kind of data is equivalent to the Cartesian grid as described in [60]. Rectilinear data is also orthographic, with non-constant spacing between nodes. This kind of data is equivalent to the rectilinear grid as described in [60]. Irregular data is non-orthographic, with variable spacing between nodes. Each data element has an explicit coordinate specifying its location in the physical space. Irregular data is equivalent to the structured grid in [60].

2.5.2 AVS Unstructured Data

While AVS structured data is essentially the same as the structured grid in the computational grid model, AVS unstructured data is a true extension of the computational grid model because it introduces the notion of *cells*. AVS unstructured data is a set of connected points in 2D or 3D space, consisting of vertices, aggregates of vertices into edges, aggregates of edges into faces, and aggregates of faces into polyhedra, or cells. Unlike AVS structured data, AVS unstructured data is not based on arrays of data, but instead upon groups of cells.

An AVS unstructured data set is a list of cells where cells can be one of the following primitive types: triangles, rectangles, tetrahedra, pyramids, prisms, or hexahedra (see Figure 2.8). Unlike the unstructured grid of the computational grid model, AVS allows the representation of data sets consisting of a heterogeneous collection of cells. This gives AVS the power to represent many kinds of nongrid topologies as AVS unstructured data.

Although AVS simplifies the representation of a topology by modeling localized patterns in the topology as cells, it does not provide a mechanism to facilitate the modeling of connections among cells. All such connections have to be specified explicitly. Although there might exist regular patterns in the connections among cells in an AVS unstructured data set, AVS can not take advantage of them.

An AVS unstructured data set consisting of cells of the same type is equivalent

to the unstructured grid in the computational grid model. AVS unstructured data sets exhibiting regular patterns are considered structured in our definition, since there exist logical transformations to index spaces with at least one dimension of usual distance.

2.5.3 AVS Summary

For structured data in the AVS model, the subset operations are the same as the ones for structured grids in the computational grid model. Since AVS does not model the interrelationship among cells, topological subsets are irrelevant on AVS unstructured data. Since data points in a cell are close to each other in the physical space, a basic assumption in the AVS model, geometric subset operations can be made more efficient by a two stage process - selection of cells followed by selection of points.

Gelberg et al. [23] provide a practical set of data structures for data visualization. The AVS structured data types correspond to the structured grids described in [60]. It is the idea and implementation of the AVS unstructured data types that are of most interest. Through the introduction of cells, the AVS model can support some data sets more efficiently than the unstructured grids of the computational grid model by capturing local regularities in cells.

2.6 Fiber Bundle

The fiber bundle model of Haber, Lucas, and Collins [28] is based on the mathematics of *fiber bundles*. The emphasis is on the effective representation of *field* data as might arise in a variety of scientific applications. One distinct feature of the fiber bundle model is that a systematic method can be easily formulated to estimate the data value of an arbitrary point in the field through interpolation, which makes it especially useful in visualization.

2.6.1 Fiber Bundle Model of Field Data

A *field* is an object comprised of a *base* and *dependent data*. A base is a *manifold*[7] whose coordinates are the independent variables for the field. The dependent data prescribes a value of the dependent variable for every position on the base.

[7] A manifold is simply an abstract surface of arbitrary dimension. See Appendix A for a precise definition.

Mathematically, a base is a manifold $\Omega \subset B$: where B is a *base-coordinated space*. Every position on Ω is associated with a *base position vector*, $\Xi \in B$. A *manifold-coordinate space*, M, and a *manifold domain* (or *manifold*), $\omega \subset M$, are introduced to describe Ω. A position on the manifold is designated by a *manifold position vector*, $\xi \in M$. The relation between Ω and ω is established by an invertible, differentiable mapping $\Phi : \omega \rightarrow B$ such that:

$$\Xi = \Phi(\xi): \quad \Omega = \Phi(\omega).$$

Figure 2.9 illustrates the mapping Φ. Note that the dimensions of the manifold-coordinate space and the base-coordinate space need not be 2 and 3 as illustrated, although the manifold-coordinate space does usually have a lower dimension than the base-coordinate space.

The dependent data for a given field is designated by the variable $y \in Y$, where Y is the *dependent variable space*. Mathematically, dependent data can be viewed as a continuous mapping $\Psi : \omega \rightarrow Y$, such that

$$y = \Psi(\xi), \ \forall \xi \in \omega.$$

Note that while Φ has to be invertible and differentiable, Ψ only has to be continuous. The dependent variable space can be almost anything as long as there is a *neighborhood structure* on it such that continuity of Ψ can be interpreted (see Section 3.3 for the definition of neighborhood structure and its relation to continuity). In Figure 2.9, the mathematical structure of Y is left unspecified intentionally.

A *field space*, S, is defined as $\Omega \times Y$. A *field*, F, is obtained by assigning a specific value of the dependent variable $y \in Y$, to every position on the manifold. Thus, a field F is a pair of mappings (Φ, Ψ) that share the same manifold domain ω. F can be defined as:

$$F = (\omega, \Phi, \Psi).$$

A field space is simply the union of all fields that share the same base Ω and the same dependent variable Y. In the standard mathematical terminology, a field space is called a *fiber bundle* and a field is a *fiber bundle section*.[8]

[8] In fact, to be precise, they are called a *bundle* and a *bundle section* respectively. See Appendix A for their definitions.

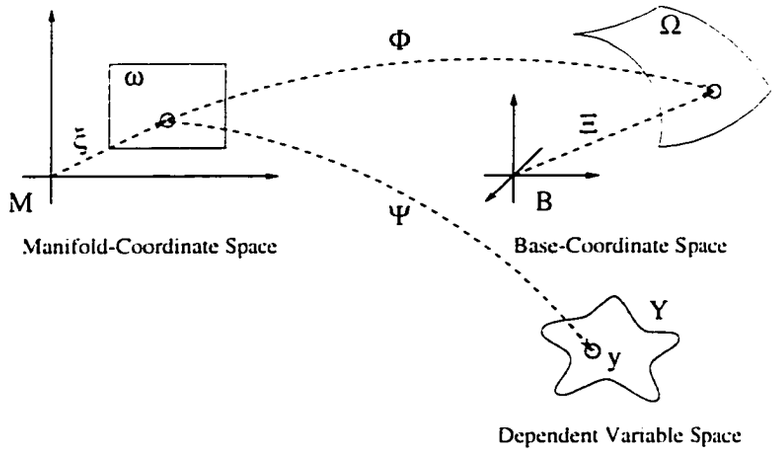


Figure 2.9: Components of a Field

Sometimes, a data set is represented or modeled by more than one field. For instance, a particular data set S can consist of two different observations made at each of a finite set of points on a manifold in the base space, i.e., the physical space. S can be modeled by two fields sharing the same base-coordinate space and manifold-coordinate space. $F_1 = (\omega, \Phi, \Psi_1)$ and $F_2 = (\omega, \Phi, \Psi_2)$.

The notion of field enables interpolation on a manifold such that the value of an arbitrary point on the manifold can be estimated from a finite binary relation. $(\Phi(\xi), \Psi(\xi)), \xi \in \omega$. With *piecewise field representations* and *compact field representations*, such interpolation can be extended to complex objects in the physical space.

2.6.2 Piecewise Field Representations

Sometimes, the entire base domain Ω might be too complicated to be described by a single mapping Φ and a simple range of the manifold coordinates ξ, or the variation of the dependent variable y might be too complicated to be conveniently described as a single *analytic function*[9] Ψ over ω. In such cases, it is often useful to subdivide the base into a number of segments so that simple, local descriptions of the manifold ω and the mapping Φ and Ψ suffice within each segment. Each such segment and its corresponding Φ and Ψ define a *field element*. Figure 2.10 illustrates a base consisting of two segments, each of which is a part of two separate field elements. $F_1 = (\omega_1, \Phi_1, \Psi_1)$ and $F_2 = (\omega_2, \Phi_2, \Psi_2)$. It

[9]Analytic functions are functions which can be represented by power series, i.e., functions of the form $f(x) = \sum_{n=0}^{\infty} c_n (x-a)^n$. See [50] for their mathematical properties.

is possible that $Y_1 = Y_2$, however.

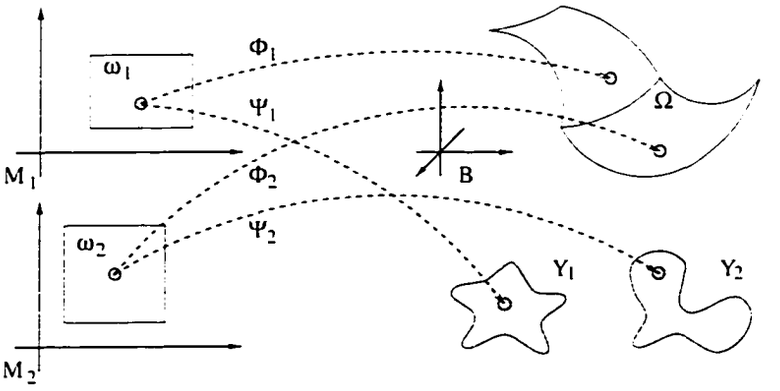


Figure 2.10: A Base Consists of Two Segments

A field element consists of a base representation, a dependent variable representation and a global topology description. These are specified in a standard format that is unique to each *element type*. Each element type is defined by a unique combination of a manifold space, a range of the manifold coordinates to define the manifold domain ω, and the parametric models for specifying the mapping Φ and Ψ. A field element description includes a specification of the field element type and the data needed to define the global topology and the complete parametric definitions of Φ and Ψ.

2.6.3 Compact Field Representations

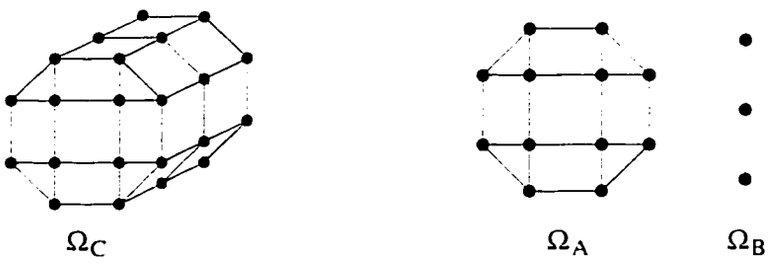
A complex data object might need to be decomposed into a large set of field elements. Fortunately, by decomposing the data object carefully, it is often possible to decompose the object into collections of field elements of the same element type. Thus, a compact representation can be derived to describe each such collection consisting of coherent field elements.

For instance, the Cartesian product operation is a powerful tool for generating compact representations of higher-dimensional fields using lower-dimensional fields. A *product field* is the Cartesian product of two lower-dimensional fields. Suppose we have defined a pair of field spaces that share the same dependent-variable space, $S_A = \Omega_A \times Y$ and $S_B = \Omega_B \times Y$. The *product field space* $S = S_A \times S_B \equiv \Omega \times Y$, where $\Omega = \Omega_A \times \Omega_B$.

Let n, n_A, and n_B be the number of field elements of Ω, Ω_A, and Ω_B respectively.

Clearly, $n = n_A \times n_B$. The parameters associated with each of the n field elements in Ω can be derived from the parameters of the $n_A + n_B$ field elements in Ω_A and Ω_B systematically.

Figure 2.11(a) illustrates a field F_C with a complex base which can be represented by the cross product of two simpler fields F_A and F_B described in Figure 2.11(b). While Ω_A is still rather complicated, Ω_B is as simple as a one-dimensional base can get. Ω_A can be further decomposed as the cross product of yet another two bases.



(a) (b)

Figure 2.11: Cross Product of Fields

2.6.4 Fiber Bundle Summary

The fiber bundle field data model provides a way to decompose a complex data object into components which can be described by simple mappings. Given a collection of data points and the topology among them, the fiber bundle model can be used to store the data set along with the topology, and interpolate data values at arbitrary points on the geometry.

The ability to generate complex fields, i.e., topologies, through the cross product of simpler fields of lower dimensions enables us to take advantage of regular patterns occurring along specific dimensions of the topology. While the AVS model extends the computational grid to model repetitive local patterns through cells, the fiber bundle model extends the computational grid to model repetitive dimensional patterns through cross products.

2.7 Lattice

The *lattice data model* was first proposed by Bergeron and Grinstein [2] and was extended by Kao, Bergeron, and Sparr [35]. The lattice model is based on the idea that

data are mathematical functions defined on physical spaces. Similar to the computational grid and the fiber bundle data model, the major goal of the lattice model is to capture the adjacency interrelationships among sampled data points.

It should be noted that the term, lattice data model, is also used by Hibbard et al. to describe a very different approach [30, 31] based on the idea that scientific data objects are usually approximations to mathematical objects, and that data objects can be ordered according to the precision of that approximation. In the lattice data model of Bergeron and Grinstein [2], the order relation exists between points in the function domain of a single data object. In the lattice model of Hibbard et al. [30, 31], the order relation exists between different data objects.

2.7.1 Multiple Views of a Data Set

A data set is a collection of data points which represent observations made at various locations in a physical space (Section 2.2). Based on a physical space, the set of attributes can be partitioned into two disjoint sets, *geometric attributes* and *nongeometric attributes*. The set of geometric attributes spans the physical space, and the set of nongeometric attributes spans a *value space*. The data can be viewed as a function from a physical space (independent variables) to a value space (dependent variables) [34].

Given a data set, the physical space where the data sampling actually takes place is called the *sampling space*. There exist views for a data set other than the one based on the sampling space, since we might want to study a data set based on geometric spaces other than the sampling space. Theoretically, any subset of the attributes can be used as geometric attributes to specify a geometric space. The lattice model provides a mechanism to incorporate multiple views, called *lattices*, with a given data set. It also describes the interrelationships among different views.

2.7.2 Definition of Lattice

A lattice is a function from an index space to a value space. Figure 2.12 illustrates the relationships among index space, physical space, and value space.

In its simplest (and most common) form, the points of a lattice are related to its "neighbors" in a regular rectangular pattern. Such a lattice maps readily to a multidimensional rectangular array, and is called a *rectangular lattice*. Let L_k^n represent an

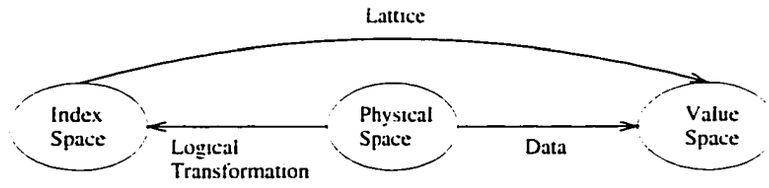


Figure 2.12: Lattice as a Function

n-dimensional rectangular lattice of points with k data attributes. The *lattice dimension* of L_k^n is defined as n, the *data dimension* of L_k^n is defined as k, and the *norm* of L_k^n is defined as $n + k$. The lattice dimension is actually the dimension of lattice topology. An *n-dimensional lattice* means a lattice with a lattice dimension of n. For instance, L_k^0 is just a set, and L_k^1 is a linear list. Through different geometry mappings, we can have different interpretations of the data being stored in one lattice. The dimension of the geometry need not be the same as the dimension of lattice topology. For example, a 2D lattice can be mapped to a surface in 3D.

There are some pre-assumed structures associated with every non-zero dimensional rectangular lattice: i.e., adjacent elements in the associated array of the lattice are assumed to be related in some way. This pre-assumed structure is described by the indexes of the associated array. By extending the *adjacency* among data tuples in the associated array, we can often capture part of the interrelationships among data points in the sampling space without storing or specifying them explicitly. In other words, not all of the location parameters have to be stored in order to store the geometry. The extended *adjacency* is called the *connectivity* of a lattice and it can be specified by a set of *connectivity methods* defined on the indexes of the associated array. The geometry of the lattice can be reconstructed from array indexes, connectivity methods, and location parameters.

L_k^0 has a 0-dimensional associated array, which has no indexes at all. No connectivity methods can be defined on it. The geometry of the lattice has to be inferred from the location parameters exclusively. It is easy to see that one data set might have more than one lattice representation. The transformation from one lattice representation of a data set to another is said to be *lossless* if the transformation is invertible. A non-lossless transformation is a *lossy* transformation.

Every data set has a trivial 0-dimensional lattice representation. Starting from the 0-dimensional lattice, we can usually transform it to some higher order lattice of the

same norm without loss of information by specifying a set of connectivity methods. Some repository data sets with inherent structures have trivial mappings to non-zero dimensional lattices.

2.7.3 Lattice Summary

The lattice model framework is flexible enough to accommodate a large collection of scientific data sets and a wide range of data representations. In particular, rectangular lattices map directly to multidimensional arrays. However, more work is needed to extend its utility to non-rectangular lattices. The precise notions of geometry, topology, and indexable topology presented in Section 2.2 are inspired by previous work on the lattice model [2, 35] and can be easily incorporated.

2.8 Summary

As we can see from the models described in the previous sections, there are consistent and well constructed models for highly structured scientific data but there is no clear way to have a compact representation of the interrelationship among data elements in a complex data set. Ideally, such a representation (or model) should have the following properties:

1. *Flexible:* It should be able to accommodate various kinds of structured and unstructured data sets.
2. *Comprehensive:* It should also model metadata, such as the representation of error.
3. *Efficient:* The model should be efficient in terms of data storage and retrieval.
4. *Effective:* There should be a systematic and effective method to map the conceptual data model into a database implementation level schema.

Above all, it should be noted that a data model is just a vehicle to support exploratory data analysis and visualization. It is important to understand the needs of data analysis/visualization procedures, and thus a data model can be developed to facilitate them.

Chapter 3

Mathematical Preliminaries

3.1 Introduction

In order to formally describe the proposed *metric-based scientific data model* (Chapter 4) and the theoretical aspect of *multipolar mapping* (Chapter 5), we need to introduce several basic mathematical notions from the fields of *metric theory* and *general topology*.

Sections 3.2, 3.3, and 3.4 present a natural progression from *distance* to *neighborhood* to *topology*, allowing for the introduction of various *metric spaces*, *neighborhood spaces*, and *topological spaces*; and the study of *continuity* in these settings. The proofs of many claims made in this chapter are omitted, but most are easily derived or found in [5, 58].

3.2 Distance Function and Metric Axioms

Definition 3.2.1 (Distance Function) *A* distance function *(or* distance*) d on a nonempty set X is a function such that $d : X \times X \longrightarrow \mathbb{R}^+ \cup \{0\}$. The notation $d(p, q)$ is read as* the distance from p to q.

Given a nonempty set X, any function $d : X \times X \to [0, \infty)$ imposes a notion of "abstract" distance on the points of X. We use the word "abstract" because it may not be immediately clear that an arbitrary nonnegative real-valued function on $X \times X$ satisfying no particular axioms necessarily describes what can be regarded on an intuitive level as a "distance." For example, it may not appear reasonable to allow $d(p, p) > 0$ for some point $p \in X$. In fact, there are several *metric axioms* which many mathematicians and scientists

would expect a function $d : X \times X \to [0, \infty)$ to satisfy if d is to describe a "reasonable" notion of distance among points of X. In Table 3.1, the four most common metric axioms are listed.

Let d be a distance function on X and $\forall p, q, r \in X$:	
M1. $d(p,p) = 0$	
M2. $d(p,q) \leq d(p,r) + d(r,q)$	M2′. $d(p,q) \leq \max\{d(p,r), d(r,q)\}$
M3. $d(p,q) = d(q,p)$	
M4. $d(p,q) = d(q,p) = 0 \Longrightarrow p = q$	M4′. $d(p,q) = 0 \Longrightarrow p = q$

Table 3.1: Basic Metric Axioms

Axioms M2′ and M4′ are the "strong" versions of axioms M2 and M4 respectively. Note that axiom M1 requires that the distance from a point to itself be 0, a quality known as *reflexivity*. Axiom M2 effectively tells us that when we wish to "move" from one point p to another point q, there is no advantage (from the point of view of minimizing distance) in "visiting" some other point of X along the way. This is commonly known as the *triangle inequality*. From a different perspective, it also guarantees that $d(p,q)$ represents the minimum effort to "move" from p to q. Axiom M3 is known as *symmetry*. If there exists a subset of X in which the distance between each pair of points is 0, axiom M4 identifies the subset with a single representative point, a quality that is sometimes called *identity of indiscernibles* [58].

Definition 3.2.2 (Pseudo-Quasimetric) *Let d be a distance function on X. d is a* pseudo-quasimetric *and (X,d) is a* pseudo-quasimetric space *if axioms* M1 *and* M2 *are satisfied.*

Definition 3.2.3 (Pseudo-Metric) *Let d be a distance function on X. d is a* pseudo-metric *and (X,d) is a* pseudo-metric space *if axioms* M1, M2, *and* M3 *are satisfied.*

Definition 3.2.4 (Metric) *Let d be a distance function on X. d is a* metric *and (X,d) is a* metric space *if axioms* M1, M2, M3, *and* M4 *are satisfied.*

Example 3.2.5 *Let $\mathbb{R}$ denote the set of real numbers and $\mathbb{Z}^+$ the set $\{1, 2, 3, \ldots\}$ of positive*

integers. For each $n \in \mathbb{Z}^+$, *define* $d_n : \mathbb{R}^n \times \mathbb{R}^n \to [0, \infty)$ *by*

$$d_n((x_1, \ldots, x_n), (y_1, \ldots, y_n)) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \ .$$

Note that d_1 *describes the usual distance between real numbers on the number line,* d_2 *the usual distance between points in the* Cartesian plane, *and* d_3 *the usual distance between points in three-dimensional* Euclidean space. *The distance function* d_n *is a metric, generally known as the* Euclidean distance *for* $\mathbb{R}^n$. ∎

Although many distances employed in mathematical or scientific settings are metrics, the following examples point out that none of the axioms M1 - M4 need be considered essential if $d : X \times X \longrightarrow [0, \infty)$ is to describe some notion of distance.

Example 3.2.6 *Suppose* A *and* B *are cities connected by one-way rail lines. The rail line from* A *to* B *is five miles long, but the rail line from* B *to* A *is only four miles long. Let* $X = \{A, B\}$ *and define* $d : X \times X \to [0, \infty)$ *by*

$$d(A, A) = d(B, B) = 0, \quad d(A, B) = 5, \quad d(B, A) = 4.$$

Observe that although d *describes the rail distances between cities, it does not satisfy axiom* M3. ∎

Example 3.2.7 *Let* $\varepsilon > 0$ *and suppose that a real number* x *may be used as an approximation for a real number* y *provided that the Euclidean distance from* x *to* y *is less than* ε. *We can view* ε *as the smallest unit measurable by a particular measuring device. Then* $d_\varepsilon : \mathbb{R} \times \mathbb{R} \to [0, \infty)$ *defined by*

$$d_\varepsilon(x, y) = \begin{cases} 0, & \text{if } |x - y| < \varepsilon; \\ 1, & \text{otherwise.} \end{cases}$$

models this situation in the sense that it identifies those real numbers "close enough" to a given real number x *to be considered approximations for* x. *Note that* d_ε *violates* M2 *and* M4. ∎

Example 3.2.8 *Let* Q *be the set of rational numbers in* $[0, 1]$ *and* $P = [0, 1] - Q$. *Given* $x \in [0, 1]$, *recall that* x *has a unique decimal representation which does not terminate.*

For each $x \in [0,1]$ and each positive integer k, let $x(k)$ be the k-th decimal digit in this representation of x. Now define $d : [0,1] \times [0,1] \to [0,\infty)$ by

$$d(x,y) = \begin{cases} 1. & \text{if } x,y \in P: \\ 0. & \text{if } x,y \in Q \text{ and } x = y: \\ \frac{1}{2^n}. & \text{otherwise, where } n \text{ is the least of} \\ & \text{all } k \in \mathbb{Z}^+ \text{ such that } x(k) \neq y(k). \end{cases}$$

The motivation behind the definition of d stems from the attempt to approximate an irrational in $[0,1]$ using rationals in $[0,1]$. Since it would be impractical to use irrationals as approximations, we have constructed d so that irrationals become "far" from one another (even from themselves). Thus, d does not satisfy M1. *It should also be noted that d does not satisfy* M2 *($d(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}) = 1$, since $\frac{\sqrt{2}}{2}$ is irrational, and, as $\frac{\sqrt{2}}{2} = 0.707106\ldots$, $d(\frac{\sqrt{2}}{2}, 0.7\bar{1}) = d(0.7\bar{1}, \frac{\sqrt{2}}{2}) = \frac{1}{4}$, so that $d(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}) > d(\frac{\sqrt{2}}{2}, 0.7\bar{1}) + d(0.7\bar{1}, \frac{\sqrt{2}}{2})$). Of course, in a very concrete and useful way, d does describe a notion of "distance" on the points of $[0,1]$ (points of $[0,1]$ which are "close" to an irrational $p \in [0,1]$, as measured by d, are exactly those rationals in $[0,1]$ which "agree" with p in the first several decimal places).* ∎

Quite often, the mathematical structure imposed (naturally or otherwise) on a set can be represented in the form of a binary relation on that set.

Example 3.2.9 *For each $n \in \mathbb{Z}^+$, the set $\{1, \ldots, n\}$ is called an* initial segment *of $\mathbb{Z}^+$. Define*

$$\begin{aligned} \mathcal{B}_f &= \{\emptyset\} \cup \{x | x : A \to \{0,1\} \text{ for some initial segment } A \text{ of } \mathbb{Z}^+\}. \\ \mathcal{B}_i &= \{x | x : \mathbb{Z}^+ \to \{0,1\}\}. \end{aligned}$$

$\mathcal{B}_f$ and $\mathcal{B}_i$ are isomorphic to the sets of finite and infinite bit strings, respectively. Thus, $\mathcal{B} = \mathcal{B}_f \cup \mathcal{B}_i$ is the set of all bit strings. Given $x, y \in \mathcal{B}$, we say that x is a prefix *of y, denoted $x \sqsubseteq y$, provided that the domain of x is a subset of the domain of y and for each k in the domain of x, $x(k) = y(k)$. Now define $d_\sqsubseteq : \mathcal{B} \times \mathcal{B} \to [0,\infty)$ by*

$$d_\sqsubseteq(x,y) = \begin{cases} 0. & \text{if } x \sqsubseteq y: \\ 1. & \text{otherwise.} \end{cases}$$

Then, given $x, y \in \mathcal{B}$, x is a prefix of y if and only if $d_\sqsubseteq(x,y) = 0$. Thus, $d_\sqsubseteq$ is a representation of the prefix order $\sqsubseteq$ imposed on $\mathcal{B}$. ∎

Observe that $\sqsubseteq$ introduced in Example 3.2.9 is a binary relation on the set of all bit strings. The situation described in that example can be generalized as follows. Given a binary relation R on a nonempty set X, define $d_R : X \times X \to [0, \infty)$ so that

$$d_R(x, y) = \begin{cases} 0. & \text{if } (x, y) \in R; \\ 1. & \text{otherwise.} \end{cases}$$

Then d_R encapsulates the information included in the binary relation R in the sense that $(x, y) \in R$ if and only if $d_R(x, y) = 0$.

From the examples considered, we have shown that while basic metric axioms are satisfied by many common distance functions, there exist useful mathematical structures which are better represented by distance functions satisfying none or only some of those metric axioms. In the literature, the distance function defined in Definition 3.2.1 is sometimes referred as *weak distance function (wdf)* to emphasize the fact that it might not satisfy any metric axioms [59].

Based on Definition 3.2.1, a distance function d on a set X has to be defined for all $(p, q) \in X \times X$. However, for many domains, such a "comprehensive" distance might not be necessary or might not even exist.

Definition 3.2.10 (Partial Distance) *Given a set X, a* partial distance on X *is a distance function d defined for all $(p, q) \in X_A \times X_B$, where X_A and X_B are nonempty subsets of X.*

There are also times when we are only interested in the distances originating from one particular fixed point.

Definition 3.2.11 (Point Distance) *Given a set X and a point $p \in X$, a* point distance on p *is a distance function defined on $\{p\} \times X$.*

Clearly, a point distance is a partial distance by definition. There are times when we want to examine a distance or a partial distance d from a single point perspective, say point p. The point distance induced from d on p is represented by $d|_p$.

3.3 Continuity and Neighborhoods

A function can possess many mathematical properties of which *continuity* is probably the most important one. The basic idea of continuity can be described in a single

statement.

> A function $f : X \to Y$ is *continuous* at $p \in X$ if and only if the images of points in X that are "close" to p are "close" to $f(p)$ in Y.

The precise definition of *closeness* in the above statement depends on the mathematical structure in context. Since X and Y might not have the same mathematical structure, the meaning of closeness in X and Y can be very different. The notion of distance function provides an intuitive definition of closeness. Using distance functions, continuity can be defined as follows.

Definition 3.3.1 (Continuity by Distances) *Let X, Y be nonempty sets and d_X, d_Y be distance functions for X and Y, respectively. A function $f : X \to Y$ is* continuous *at a point $p \in X$ provided that for each $\varepsilon > 0$ there exists $\delta > 0$ such that $\forall x \in X$, $d_X(p, x) < \delta \Longrightarrow d_Y(f(p), f(x)) < \varepsilon$.*

Since we are always able to represent a binary relation using a distance function, Definition 3.3.1 provides us with a natural notion of continuity for a function whose domain and codomain have binary relations imposed on them. Continuity in such a setting can be characterized in terms of the binary relations.

Proposition 3.3.2 (Continuity and Binary Relations) *Let R_X, R_Y be binary relations on the nonempty sets X and Y, respectively. For $f : X \to Y$ and $p \in X$, the following are equivalent:*

1. *f is continuous at p.*
2. *$\forall x \in X, (p, x) \in R_X \Longrightarrow (f(p), f(x)) \in R_Y$.*

Based on distance functions, *sphere* is a mathematical notion for representing all the points within a certain degree of closeness or *proximity* to a fixed point (i.e., the *center* of the sphere).

Definition 3.3.3 (Sphere) *Let d be a distance function on X, $p \in X$, and $r > 0$. The* sphere centered at p of radius r *is defined as*

$$S_d(p, r) = \{q \in X \mid d(p, q) < r\}.$$

The following notation is used to identify a sphere restricted *to X', a subset of X.*

$$S_d|_{X'}(p, r) = S_d(p, r) \cap X'.$$

The set of all spheres centered at p is then denoted $\mathcal{S}_d(p)$.

Example 3.3.4 *If d_1 is the Euclidean distance for $\mathbb{R}$, then, given $p \in \mathbb{R}$ and $\varepsilon > 0$, $S_{d_1}(p, \varepsilon)$ is the open interval $(p - \varepsilon, p + \varepsilon)$ consisting of all real numbers strictly between $p - \varepsilon$ and $p + \varepsilon$.*

If d_2 is the Euclidean distance for $\mathbb{R}^2$, then, given $p \in \mathbb{R}^2$ and $\varepsilon > 0$, $S_{d_2}(p, \varepsilon)$ consists of all points in the Cartesian plane lying within the interior of the circle centered at p of radius ε.

If d_3 is the Euclidean distance for $\mathbb{R}^3$, then, given $p \in \mathbb{R}^3$ and $\varepsilon > 0$, $S_{d_3}(p, \varepsilon)$ consists of all points in three-dimensional space lying within the interior of the sphere (here we are using the word sphere *in the context of three-dimensional Euclidean geometry) centered at p of radius ε.* ∎

Example 3.3.5 *Let R be a binary relation on a nonempty set X and $p \in X$. Then*

$$S_{d_R}(p, \varepsilon) = \begin{cases} \{x \in X | (p, x) \in R\}. & \text{if } \varepsilon \leq 1; \\ X. & \text{otherwise.} \end{cases}$$

In particular, if $\mathcal{B}$ and $\sqsubseteq$ are defined as in Example 3.2.9, then given $p \in \mathcal{B}$,

$$S_{d_\sqsubseteq}(p, \varepsilon) = \begin{cases} \{x \in \mathcal{B} | p \sqsubseteq x\}. & \text{if } \varepsilon \leq 1; \\ \mathcal{B}. & \text{otherwise.} \end{cases}$$

Thus, the "smallest" $d_\sqsubseteq$-sphere centered at a bit string p is the set of all bit strings having p as a prefix. ∎

Example 3.3.6 A sphere need not contain its center. *Let d be the distance function for $[0, 1]$ introduced in Example 3.2.8 and note that $\frac{\sqrt{2}}{2} \notin S_d(\frac{\sqrt{2}}{2}, 1)$.* ∎

Example 3.3.7 A sphere may be empty. *Let X be a nonempty set and $d : X \times X \to [0, \infty)$ be defined by $d(x, y) = 1, \forall x, y \in X$. Then $\forall x \in X, S_d(x, 1) = \emptyset$.* ∎

Intuitively, a sphere centered at p represents a set of points which are considered "near" p, where "near" is defined by a particular distance function. Thus, any set which contains a sphere centered at p includes points which are "near" p.

Definition 3.3.8 (Neighborhood) *Let d be a distance function for a nonempty set X. A* neighborhood *of $p \in X$ is a superset of a sphere centered at p. The set of neighborhoods of p is denoted $\mathcal{N}_d(p)$.*

Clearly, any sphere centered at p is a neighborhood of p. We can characterize continuity using either spheres or neighborhoods.

Proposition 3.3.9 (Continuity, Spheres and Neighborhoods) *Let X, Y be nonempty sets and d_X, d_Y be distance functions for X and Y, respectively. For $f : X \to Y$ and $p \in X$, the following are equivalent:*

1. *f is continuous at p.*
2. *For each $S \in \mathcal{S}_{d_Y}(f(p))$ there exists $S' \in \mathcal{S}_{d_X}(p)$ such that $f[S'] \subseteq S$.*
3. *For each $N \in \mathcal{N}_{d_Y}(f(p))$ there exists $N' \in \mathcal{N}_{d_X}(p)$ such that $f[N'] \subseteq N$.*

Note that the characterization of continuity given in Proposition 3.3.9 (3) is based on the neighborhoods induced by distance functions but makes no explicit reference to distances. It thus appears intuitively plausible to develop a notion of continuity for a function from a set X to a set Y provided that appropriate notions of "neighborhood of a point" exist or can be imposed on X and Y. Within the context of a set X for which a distance function d has been prescribed, given $p \in X$, we observe that:

1. If $N \in \mathcal{N}_d(p)$, then there exists $\varepsilon > 0$ such that $S_d(p, \varepsilon) \subseteq N$. So, if $N \subseteq M \subseteq X$, we can conclude that $M \in \mathcal{N}_d(p)$. Hence, *supersets of neighborhoods of a point are neighborhoods of that point.*
2. If $N_1, N_2 \in \mathcal{N}_d(p)$, then there exist $\varepsilon_1, \varepsilon_2 > 0$ such that $S_d(p, \varepsilon_1) \subseteq N_1$ and $S_d(p, \varepsilon_2) \subseteq N_2$. So, if $\varepsilon = \min\{\varepsilon_1, \varepsilon_2\}$, then $S_d(p, \varepsilon) \subseteq N_1 \cap N_2$ so that $N_1 \cap N_2 \in \mathcal{N}_d(p)$. Hence, *the intersection of two neighborhoods of a point is a neighborhood of that point.*

These observations motivate the following definition (due to Smyth [59]).

Definition 3.3.10 (Neighborhood Structure) *Let X be a nonempty set. A* neighborhood structure *on X is a map $\mathcal{N} : X \to \mathcal{P}(\mathcal{P}(X))$ (for any set Y, $\mathcal{P}(Y)$ denotes the power set of Y) which satisfies:*

1. $\forall x \in X$. *if* $N \in \mathcal{N}(x)$ *and* $N \subseteq M \subseteq X$. *then* $M \in \mathcal{N}(x)$:

2. $\forall x \in X$. *if* $N_1, N_2 \in \mathcal{N}(x)$ *then* $N_1 \cap N_2 \in \mathcal{N}(x)$.

For each $x \in X$. $\mathcal{N}(x)$ *is the set of* neighborhoods *of* x. *The pair* $(X, \mathcal{N})$ *is called a* neighborhood space.

Example 3.3.11 *If* d *is a distance function on a nonempty set* X. *then* $\mathcal{N}_d$ *is a neighborhood structure on* X. ∎

Example 3.3.12 Given a neighborhood structure $\mathcal{N}$ on X. it is possible that $\mathcal{N} \neq \mathcal{N}_d$ for any distance function d for X. *Let* $X = \{f | f : [0,1] \to \mathbb{R}\}$. *For each* $f \in X$. *define* $\mathcal{A}(f) = \{A_{F,\varepsilon}(f) | F \subseteq [0,1]$ *is finite and* $\varepsilon > 0\}$. *where. for each finite* $F \subseteq [0,1]$ *and each* $\varepsilon > 0$. $A_{F,\varepsilon}(f) = \{g \in X | \ |f(x) - g(x)| < \varepsilon. \forall x \in F\}$. *Now define* $\mathcal{N} : X \to \mathcal{P}(\mathcal{P}(X))$ *by* $\mathcal{N}(f) = \{N \subseteq X | A \subseteq N$ *for some* $A \in \mathcal{A}(f)\}$ *and note that* $\mathcal{N}$ *is a neighborhood structure on* X.

> Claim 1: For any $\mathcal{U} \subseteq \mathcal{N}(f)$ for which $N \in \mathcal{N}(f) \implies \exists U \in \mathcal{U}$ with $U \subseteq N$. there exist $U_1, U_2 \in \mathcal{U}$ such that $U_1 \not\subseteq U_2$ and $U_2 \not\subseteq U_1$.

Proof: *See [42].*

Now consider any distance function d *for* X.

> Claim 2: $\mathcal{N} \neq \mathcal{N}_d$.

Proof: *Suppose to the contrary that* $\mathcal{N} = \mathcal{N}_d$. *Then.* $\forall f \in X. \mathcal{S}_d(f) \subseteq \mathcal{N}(f)$ *and if* $N \in \mathcal{N}(f)$. *there exists* $S \in \mathcal{S}_d(f)$ *such that* $S \subseteq N$. *Thus. by Claim 1.* $\exists S_1, S_2 \in \mathcal{S}_d(f)$ *such that* $S_1 \not\subseteq S_2$ *and* $S_2 \not\subseteq S_1$. *This contradicts the trivial observation that given two spheres centered at the same point. one is a subset of the other.* ∎

Example 3.3.13 *Let* $X = \{0,1,2\}$ *and define* $\mathcal{N}(0) = \{\{0,1\}, X\}$. $\mathcal{N}(1) = \{\{1,2\}, X\}$. *and* $\mathcal{N}(2) = \{\{0,2\}, X\}$. *Then* $\mathcal{N}$ *is a neighborhood structure on* X. ∎

Continuity is defined within the setting of neighborhood spaces as expected.

Definition 3.3.14 (Continuity by Neighborhood Structures) *Let* $(X, \mathcal{N}_X)$. $(Y, \mathcal{N}_Y)$ *be neighborhood spaces.* $f : X \to Y$. *and* $p \in X$. f *is* continuous *at* p *provided that for each* $N \in \mathcal{N}_Y(f(p))$ *there exists* $N' \in \mathcal{N}_X(p)$ *such that* $f[N'] \subseteq N$.

3.4 Topologies and Distances

We have shown that the notions of *neighborhood* and *continuity* can be extended to the general setting of neighborhood spaces. But it should be pointed out that neighborhood spaces are not the usual mathematical setting for the study of such ideas. The primary (and much more commonly employed) alternative is that of *topological spaces.*

Definition 3.4.1 (Topological Neighborhood) *Let d be a distance function for a nonempty set X. $p \in X$. and $N \subseteq X$. N is a* topological neighborhood *of p provided that there exists $A \subseteq N$ such that*

1. $p \in A$;
2. $\forall x \in A. \exists \varepsilon_x > 0$ *with* $S_d(x, \varepsilon_x) \subseteq A$.

We denote by $\mathcal{N}_{\mathcal{T}_d}(p)$ the set of topological neighborhoods of p.

Observe that a topological neighborhood of p is always a neighborhood of p. In other words, given a distance function d for a nonempty set X. $\mathcal{N}_{\mathcal{T}_d}(p) \subseteq \mathcal{N}_d(p). \forall p \in X$. The converse may fail, however.

Example 3.4.2 *Even if d is a distance function for X which satisfies* M1. *it is possible that $\mathcal{N}_d(p) \not\subseteq \mathcal{N}_{\mathcal{T}_d}(p)$ for some $p \in X$. Let X and $\mathcal{N}$ be as in Example 3.3.13. Then $\mathcal{N} = \mathcal{N}_d$. where d is the distance function for X defined by*

$$d(x, y) = \begin{cases} 1. & \text{if } (x, y) \in \{(0, 2), (1, 0), (2, 1)\}; \\ 0. & \text{otherwise.} \end{cases}$$

Note that d satisfies M1 *and $\{0, 1\} \in \mathcal{N}_d(0)$. But $\{0, 1\} \notin \mathcal{N}_{\mathcal{T}_d}(0)$.* ∎

Definition 3.4.3 (Open) *Let d be a distance function for a nonempty set X. $A \subseteq X$ is* open *if it contains a topological neighborhood of each of its points.*

Example 3.4.4 *If d is a distance function for X satisfying* M2. *then any sphere is open. It then follows that $\mathcal{N}_d(x) = \mathcal{N}_{\mathcal{T}_d}(x). \forall x \in X$.* ∎

Let $\mathcal{T}$ denote the collection of all open subsets of a set X for which a distance function has been defined. $\mathcal{T}$ satisfies:

T1. $\emptyset, X \in \mathcal{T}$.

T2. If $A_i \in \mathcal{T}. \forall i \in I$. then $\bigcup_{i \in I} A_i \in \mathcal{T}$ (i.e., $\mathcal{T}$ is closed under arbitrary unions).

T3. If $A_1, \ldots, A_n \in \mathcal{T}$ for some $n \in \mathbb{Z}^+$. then $\bigcap_{i=1}^{n} A_i \in \mathcal{T}$ (i.e., $\mathcal{T}$ is closed under finite nonempty intersections).

Proposition 3.4.5 *Let d be a wdf for a nonempty set X and $A \subseteq X$. The following are equivalent:*

1. *A is open.*

2. *$\forall x \in A. \exists \varepsilon_x > 0$ such that $S_d(x, \varepsilon_x) \subseteq A$.*

We denote by $\mathcal{T}_d$ the collection of open subsets of a set X for which a distance function d has been specified. Using Proposition 3.4.5, $\mathcal{T}_d$ can be represented as

$$\mathcal{T}_d = \{A \subseteq X | \forall p \in A. \exists \varepsilon > 0 \text{ such that } S_d(p, \varepsilon) \subseteq A\}. \tag{3.1}$$

Definition 3.4.6 (Topology and Topological Space) *Let X be a nonempty set and $\mathcal{T} \subseteq \mathcal{P}(X)$. $\mathcal{T}$ is a* topology *on X if it satisfies* T1 T3. *The pair $(X, \mathcal{T})$ is called a* topological space *and the members of $\mathcal{T}$ are called* open sets.

For every set X, there are two trivial topologies that can be introduced upon it: $\{\emptyset, X\}$, the *indiscrete topology* (or the *trivial topology*), and $\mathcal{P}(X)$, the *discrete topology*, where $\mathcal{P}(X)$ represents the *power set* of X (i.e., the collection of all subsets of X). The discrete topology is the finest topology of X, while the indiscrete topology is the coarsest.

Example 3.4.7 *If d is a distance function for a set X, then $\mathcal{T}_d$ is a topology on X.* ∎

$\mathcal{T}_d$ is called *the topology induced by d.* Based on Proposition 3.4.5 and Equation 3.1, there is a unique topological space corresponding to an arbitrary distance. However, for an arbitrary topological space $\mathcal{T}$, there exist infinitely many distances d such that $\mathcal{T} = \mathcal{T}_d$.[1] Topological spaces are, in fact, much more general mathematical structures than distance spaces. Let us illustrate the idea with a simple example.

[1] Kopperman proves that all topologies on any given set can be obtained through the use of "suitable generalized" metrics [42].

Example 3.4.8 *Let (X, d_1) and (X, d_2) be two distance spaces. $X = \{a, b, c\}$ and the values of d_1 and d_2 are depicted in Figure 3.1 (assuming that both d_1 and d_2 are symmetric). Although $d_1 \neq d_2$, we have $\mathcal{T}_{d_1} = \mathcal{T}_{d_2} = \mathcal{P}(X)$, the discrete topology of X. In fact, for an arbitrary finite distance space (X, d), $\mathcal{T}_d = \mathcal{P}(X)$ if and only if d satisfies axiom* M4. ∎

Figure 3.1: Two Distances on the Same Set

Every topology on a set induces a neighborhood structure on that set in the following way. Given a topology $\mathcal{T}$ on X, define $\mathcal{N}_{\mathcal{T}} : X \to \mathcal{P}(\mathcal{P}(X))$ by

$$\mathcal{N}_{\mathcal{T}}(x) = \{N \subseteq X | x \in A \subseteq N \text{ for some } A \in \mathcal{T}\}.$$

Thus, an open set is a neighborhood of each of its points. Note that if d is a distance function for X, then $\{N \subseteq X | x \in A \subseteq N \text{ for some } A \in \mathcal{T}_d\}$ is indeed the collection of topological neighborhoods of $x \in X$ (so the notation $\mathcal{N}_{\mathcal{T}_d}(x)$ is not ambiguous). We will say that a neighborhood structure $\mathcal{N}$ on X is *topological* if there exists a topology $\mathcal{T}$ on X such that $\mathcal{N} = \mathcal{N}_{\mathcal{T}}$.

Example 3.4.9 Not all neighborhood structures are topological. *Let X and $\mathcal{N}$ be as in Example 3.3.13. If $\mathcal{N} = \mathcal{N}_{\mathcal{T}}$ for some topology $\mathcal{T}$ on X, then $\{0, 1\} \in \mathcal{T}$ and $\{1, 2\} \in \mathcal{T}$ so that $\{1\} \in \mathcal{T}$, a contradiction.* ∎

A notion of continuity can be introduced within a topological setting.

Definition 3.4.10 (Continuity by Topology) *Let $(X, \mathcal{T}_X)$, $(Y, \mathcal{T}_Y)$ be topological spaces, $f : X \to Y$, and $p \in X$. f is* topologically continuous *at p if for each $N \in \mathcal{N}_{\mathcal{T}_Y}(f(p))$ there exists $N' \in \mathcal{N}_{\mathcal{T}_X}(p)$ such that $f[N'] \subseteq N$.*

The reader interested in learning more about topology may consult [44, 5].

3.5 Summary

The notions of neighborhood and continuity developed in the settings of neighborhood spaces and topological spaces are natural extensions of those ideas as they are commonly understood in the (weak) distance setting. They provide us with powerful tools for representing and studying the inter-instance relationships of data. Interested readers are referred to [10] for more information on this topic.

Chapter 4

A Metric-Based Scientific Data Model

4.1 Introduction

This chapter presents the framework of a *metric-based scientific data model* based on *data-as-functions* and *pseudo-quasimetrics*. The data function formulation and the domain pseudo-quasimetric encapsulate the inter-entity and inter-instance relationships respectively. In addition to the theoretical foundation, a detailed approach is outlined for exploring and deriving metrics in a wide variety of data. In particular, we introduce the notion of *observable properties* and show how it can be applied with ideas from *point set topology* to systematically derive metrics from *nonmetric* data components. The approach presented can also serve as a useful paradigm for *knowledge discovery* from the metric perspective. Finally, we demonstrate the use of *continuity* as a mathematically precise tool to validate metrics derived through the proposed approach.

4.2 Data as Functions

This section presents the idea of *data-as-function* which is central to our metric-based scientific data model. Formal definitions for *data function*, *data set*, and other essential terminology are given here. The relationship between a given data set and the collection of possible data functions which can be defined on it is also illustrated. A brief discussion is made to compare our *data-as-function* formalism with the *functional data model*. The ob-

jective is to establish the background for using data function formulation as a mathematical model for inter-entity relationships in scientific data.

4.2.1 Data Sets and Functions

Data sampling is the process of recording observations made on an *object* which might or might not actually exist in the physical world. An object can come from direct measurement or computer simulation in a wide range of domains, such as a volume of the atmosphere, a magnetic energy field, a piece of land, a mass of soil, a specific biological population, or a progression of urban development for a city. Data sampling results in a set of observations, called a *data set*. Although most of the objects under study are inherently infinite, we must deal with finite data sets in order to complete the analysis in a finite amount of time. Depending on the nature of the physical phenomena to be observed and the sampling techniques employed, a data set is a *finite approximation* of the *object behavior* at a specific time or over a particular time period.

Mathematically, *data* can be considered as a function from a *domain space* to a *value space*. The domain space is the mathematical representation of the object of interest. It is the set of points where the phenomena or behavior to be observed takes place. The mathematical structure of a domain space (i.e., the relationship among points in the domain space) is known as *domain geometry*. Common domain geometries can have many different forms such as *orders* and *vector spaces*.

Given a data set, there might be more than one meaningful function which can be defined on it [36]. In fact, formulating such functions is one of the most important goals for knowledge discovery. In this context, a scientist needs to experiment with different data function formulations on a data set using various data analysis tools. This process includes a "trial-and-error" component driven both by intuition and by the feedback the scientist receives from the analysis. In the past, most knowledge discovery has been done by statistical analysis coupled with visualization, which proves to be only partially effective. Statistical analysis focuses on the analysis of values of the value space for simple and well understood domain spaces. It is often of very limited use for exploring domain geometries of nonmetric or unordered domain spaces.

4.2.2 Formulations of Data Functions

Mathematically, data is formulated as a function $f : D \longrightarrow V$, where D and V are the domain space and value space, respectively. Excluding *metadata*, a *data set* is just a mathematical *set*. The elements of a data set are the recorded observations, called *observation points*, *data points*, or simply *points*. A point can be represented as a tuple $(a_1, a_2, \ldots, a_n)$, where a_i's are the *attribute values*, as in the *relational data model*.

Many data functions can be defined on a data set S: some are useful, others are not. In general, a data function maps some subset of the attributes to another subset plus zero or more new attributes, called *derived attributes*. Let $A_1, A_2, \ldots, A_n$ be the domains of $a_1, a_2, \ldots, a_n$ respectively. Similarly, let $z_1, z_2, \ldots, z_m$ and $Z_1, Z_2, \ldots, Z_m$ be derived attributes and their domains. A data function can be defined as

$$f : \prod_{i \in I_D} A_i \longrightarrow \prod_{i \in I_V} A_i \times \prod_{i \in I_Z} Z_i. \tag{4.1}$$

where $I_D, I_V \subseteq \{1, 2, \ldots, n\}$, and $I_Z = \{1, 2, \ldots, m\}$. Given an arbitrary data set S, there is a natural *membership function* f_S defined as

$$f_S : A_1 \times A_2 \times \cdots \times A_n \longrightarrow B,$$

where $B = \{T, F\}$, such that $f_S(p) = T$ if $p \in S$, $f_S(p) = F$ otherwise. Many other data functions can be defined on a given data set. Let us illustrate this point with a simple example from environmental study.

Example 4.2.1 *Let S be a data set consisting of tuples of the form (x, y, z, t, p), where x, y, z are the three-dimensional atmospheric coordinates over a particular metropolitan area located in a valley, t is the time index, and p is the pollutant density. The goal is to study how nearby mountains affect the pollutant circulation. The usual data function on S to be analyzed is $f_0(x, y, z, t) = p$. However, from the same data set S, we might want to study the behavior of several other functions such as*

1. $f_1(x, y, z) = \bar{p}$

   $\bar{p}$ is the average pollutant density over time.

2. $f_2(x, y, t) = \hat{p}$

   $\hat{p}$ is the total amount of pollutants in the air column over a small area represented

by a surface point (x, y) at time t. Let Δ denote a small time period. If rain starts falling at t, $\hat{p}$ would provide some useful information about the water quality during the interval $(t, t + \Delta)$.

3. $f_3(t, p) = r$

   r is the percentage of air volume over the area which has pollutant density p or above

3. $f_3(t, p) = r$

   r is the percentage of air volume over the area which has pollutant density p or above

We have shown that a given data set can be associated with many different functions and thus different domain spaces. Given a data set, finding or defining a meaningful data function is one of the most important goals for the knowledge discovery process. In Section 4.7, the importance of this generalization becomes apparent.

4.2.3 Functional Data Model

Our concept of "data as functions" is a generalization of the idea of *functional data models* (FDMs) [56, 4] in databases. The main modeling primitives of a functional data model are "entities" and "functions." A function takes an entity as the argument and returns another entity. The key operation of functional data models is *function composition* which is used to define *derived functions* and formulate queries in *functional query languages*. There are several proposals for functional data models and functional query languages, among which the DAPLEX model and language [55] is the best-known [27, 16].

Although our "data as functions" notion is similar to FDMs in concept, there are subtle yet significant differences between them.

1. The functions in FDMs are not pure mathematical functions. FDMs functions not only describe the semantic structure of data but also dictate the way in which data is stored and retrieved.

2. In an FDM database, the semantic structure of data has to be known (or assumed) in advance. Even though multiple views (as entities/functions) can be defined later, they have to be derived from the original set of entities/functions which are used for data modeling and storage. The distinction between functions and data sets is not clear in FDMs.

3. In FDMs, there is no notion of attributes and all entities/functions derived from the original set of entities/functions are necessarily composite. This makes the data analysis process more cumbersome.

In summary, FDMs use functions primarily to describe the semantic structure of abstract entities. Only one group of basic (i.e., non-composite) functions can be defined for a set of entities. The information encapsulated by entities is accessed through those basic functions or their composites. In this perspective, the FDM approach shares some characteristics with the object-oriented approach. In fact, some database researchers view the functional and object-oriented approaches as the same thing under different names [11].

Our approach puts emphasis on the mathematics of functions. Functions are not viewed as data access mechanisms but as mathematical notions to be studied. Instead of having a set of entities/functions on which all derived entities/functions are based, we treat each function as a peer to all the others. Thus, the relationships among functions (i.e., *orders of data*: Section 4.2.4) can be studied based on mathematics rather than the way they might have been derived in FDMs.

4.2.4 Orders of Data

Let $\mathcal{F}_S$ denote all the functions which can be defined on a data set S – either basic or not. $\mathcal{F}_S$ is a *partial order*, based on its information content. The intuition behind this is that if two different functions have exactly the same information content, we would like to identify one with the other – even though the two functions might have very different formats. Informally, we would like to say that a function f is more *informative* than a function g, denoted by $g \leq f$, if all the *objects* and *theories* observable or derivable from g can also be observed or derived from f.

In this context, both *objects* and *theories* are viewed as products of scientific discovery. Objects are entities existing in the real world whose existence might be observed. Epistemologically, theories are not real world entities; instead, theories are created by us, although not entirely by our free imagination. The process of creation is constrained by the data and conceptual resources available to us [32]. It should be noted that the discovery of both objects and theories depends not only on the state of affairs of the real world, but also on our cognitive apparatus, conceptual system, and background knowledge. Thus, given the same data, a theory derivable by one person might not be derivable by another. This

is a minor point in our context and thus, is not addressed in our informal definition of *informativeness*.

The development of a mathematically precise informativeness relation will enable us to study a data set at the next higher abstract level, the *function domain*, which describes the interrelationships among functions defined on the same data set. With such an *order of functions* in place, systematic data function formulations will be possible within the process of exploratory data analysis. Although we have not yet been able to define the informativeness relation on $\mathcal{F}_S$ mathematically, we can provide the definition of *richness* which approximates the informativeness relation in a limited sense.

Distinct attributes might have the same domain. The attributes indicate different *roles*, or interpretation, for the domain [16]. For instance, two different attributes a_i and a_j might mathematically share the same domain $\mathbb{R}$, the set of all real numbers. However, for arbitrary $r \in \mathbb{R}$, r may have different semantics for a_i and a_j. In order to formulate the richness definition, we would like to distinguish two attribute domains A_i and A_j, $i \neq j$, semantically, even if $A_i = A_j$ mathematically. This is achieved through the introduction of *semantic tag*.

A *semantic tag* or identifier (name) of an attribute a_i is represented as $t(a_i) \in T$, where T is an arbitrary set. The function t is a one-to-one function such that no two attributes, including derived ones, can have the same tag. Through the use of semantic tags, two mathematically identical attribute domains A_i and A_j, $i \neq j$, can be distinguished with their semantic tags $(t(a_i), A_i) \neq (t(a_j), A_j)$ since $t(a_i) \neq t(a_j)$.

For convenience, given a function f, we define the set of *tagged domain attributes*, $\mathcal{A}_f = \bigcup_{i \in I_D} \{(t(a_i), A_i)\}$, and the set of *tagged value attributes*,

$$\mathcal{A}'_f = \bigcup_{i \in I_V} \{(t(a_i), A_i)\} \cup \bigcup_{i \in I_Z} \{(t(z_i), Z_i)\},$$

(see Equation 4.1).

Definition 4.2.2 (Richness) *A function f is* richer *than a function g, denoted by $g \leq f$, if $\mathcal{A}_g \subseteq \mathcal{A}_f$.*

Note that the richness relation is a *preorder* instead of a partial order. The reason is that the richness relation does not precisely reflect the true order of information content, since the definition of the richness relation is based on the domain spaces only. The intuition

of the richness relation is that the more attributes we keep in the domain of a data function, the more information remains intact. For instance, in Example 4.2.1, if there are two measurements of pollutant density at the same time and same location, both of them are represented by $f_S(x, y, z, t, p)$, the membership function of the pollutant density data set S. However, this distinction is lost in f_0. Based on this relation, the order of the functions in Example 4.2.1 is illustrated in Figure 4.1. Note that the membership function is a *maximal point* of this relation.

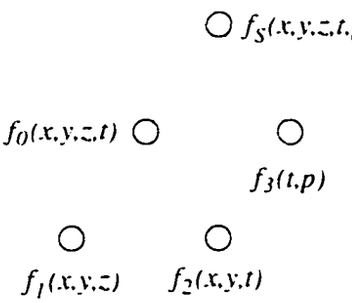


Figure 4.1: Example of a Richness Relation

4.2.5 Inter-Entity Relationships and Function Formulation

Let S be a data set consisting of 7 attributes $a_1, \ldots, a_7$ with attribute domains $A_1, \ldots, A_7$ respectively. We define a data function $f : D \longrightarrow V$, where $D = A_1 \times A_3 \times A_6$ and $V = A_2 \times A_7$. The inter-entity relationships specified by the function formulation of f can be illustrated by an ER diagram (see Figure 4.2). Since f is a mathematical function, the corresponding relationship in the ER model can not be *many-to-many*. This is not a limitation, however. To describe a many-to-many relationship between two sets D and V in a function formulation, a function $f' : D \longrightarrow \mathcal{P}(V)$, where $\mathcal{P}(V)$ is the *power set* of V, can be defined.

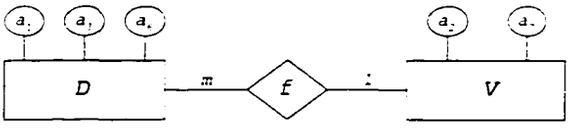


Figure 4.2: Inter-Entity Relationships Specified by Function Formulation

Once a data function formulation is determined, there are two kinds of inter-

instance relationships among data points. They can be specified by the mathematical structures on D and V respectively, i.e., the domain geometry and the value geometry. In the process of knowledge discovery, the attributes of the value space are often selected such that there exists a simple natural value geometry. Determining the domain geometry, however, is often one of the objectives of knowledge discovery.

Since the value geometries are usually simple and predetermined, we focus on the exploration and modeling of domain geometries only. Unless otherwise specified, the term "inter-instance relationship" is used interchangeably with "domain geometry." In subsequent sections, we present the mathematical basis and a systematic approach to use *pseudo-quasimetrics* as a foundation for exploring and modeling domain geometries.

4.3 Functional Dependency among Attributes

Given a data set S and a function f to be analyzed, the domain space of f is determined. However, there might be more than one meaningful domain geometry associated with the domain space. Finding such a meaningful domain geometry is one of the major goals of knowledge discovery.

Syntactically, each element of a domain space is a composite of values of domain attributes - similar to a *tuple* or *record* in a relational database. Given a function and, hence, its domain space, the set of domain attributes is determined. From the perspective of domain geometry, the set of domain attributes can be partitioned into a set of *variates*[1] based on some meaningful semantic properties.

Similar to an attribute, a variate can be an unordered set, *a preorder, a partial order, a total order, a lattice, a complete lattice, an algebraic field*, and so on [14]. The set of variates of a domain space depends upon its domain geometry. A variate in the domain geometry is typically constructed from one or more domain attributes. An attribute is the smallest *syntactic* unit while a variate is the smallest *semantic* unit. The difference between variates and attributes is illustrated in the following example.

Let the domain space D be a square piece of land. If the land is flat and void of obstacles, the "normal" distance from a point $p_1 = (x_1, y_1)$ to point $p_2 = (x_2, y_2)$ can be

[1] In *multivariate data analysis* [29], a *variate* is a linear combination of variables, i.e., attributes, where the coefficients are determined by applying multivariate techniques to the data set. Thus, each variate is a unique description for the group of interrelated variables. We extend the variate definition to represent any kind of combination, aggregation, or integration of a set of interrelated variables.

defined as the *Euclidean distance*

$$d_E(p_1, p_2) \equiv \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

and the corresponding domain geometry is called the *Euclidean geometry* (Figure 4.3(a)). To be specific, d_E is the Euclidean distance of $\mathbb{R}^2$. Euclidean distances of $\mathbb{R}^n$, $n \in N$, also known as the *usual distances*, can be similarly defined. However, given the same domain space D, a different domain geometry might impose restrictions such that only movements along the x-axis and y-axis are allowed. In this case, the distance between p_1 and p_2 becomes the *Manhattan distance*

$$d_M(p_1, p_2) \equiv |x_1 - x_2| + |y_1 - y_2|.$$

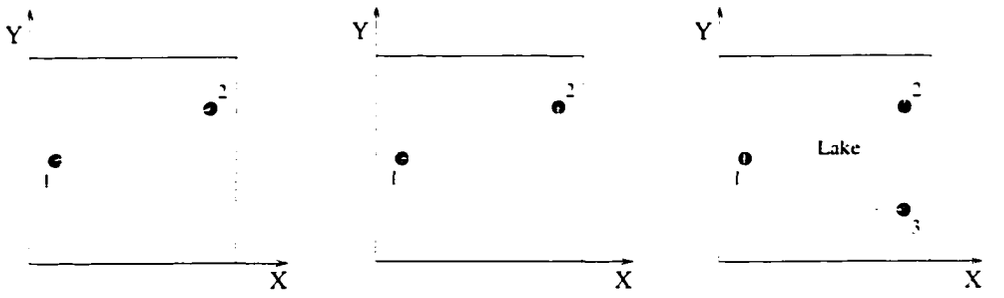
and the corresponding domain geometry is called the *Manhattan geometry* (Figure 4.3(b)). Comparing the formulations of d_E and d_M, we notice that both of them can be represented in terms of

$$d_x(p_1, p_2) \equiv |x_1 - x_2|. \qquad d_y(p_1, p_2) \equiv |y_1 - y_2|.$$

the Euclidean distances on the x-axis and y-axis respectively. This is not a coincidence. In both Euclidean and Manhattan geometries, the x dimension and the y dimension are *functionally independent*. For such domain geometries, we can treat the elements of the domain space as composites of values from functionally independent attributes. Instead of finding the distance of the domain directly, we can start by finding the distance for each of the attributes first. Once we have the distances for each attribute, their functional independence assures us that the domain distance can be represented as a function of individual attribute distances. In both the Euclidean and Manhattan geometries, each attribute is independent of the other, and thus a variate by itself.

Suppose there is an impassable lake in the middle of the land of interest (Figure 4.3(c)). Let $L \subseteq D$ represent the lake region in the domain space D. The domain space stays the same – a set of two dimensional coordinates within the land boundaries, but the domain geometry is changed. Let us still keep the movement restrictions of Manhattan geometry and call this geometry the Winnipesaukee geometry. Let us denote the distance function of the Winnipesaukee geometry by d_W, even though its precise mathematical formulation is too complicated to be presented here.[2] To enforce the impassability of L, we

[2]Here are some general ideas for formulating $d_W(p, q)$. For $p, q \in D - L$, $d_W(p, q) = d_M(p, q)$, if q is "visible" from p (i.e., q can be seen from p without the lake area in front of q). Otherwise, $d_W(p, q) =$



(a) Euclidean (b) Manhattan (c) Winnipesaukee

Figure 4.3: Different Domain Geometries on the Same Domain Space

only need to ensure that passing through L is prohibitively expensive in the geometry, so the distance between two points in $D - L$ is not measured through L. This can be done by defining $d_W(p,q) = c \cdot d_M(p,q)$ for $p,q \in L$, where $c > 0$ is a very large constant. Now, consider the distances $d_W(p_1,p_2)$ and $d_W(p_1,p_3)$. Clearly, $d_W(p_1,p_2) > d_W(p_1,p_3)$, even though $d_x(p_1,p_2) = d_x(p_1,p_3)$ and $d_y(p_1,p_2) = d_y(p_1,p_3)$.

This shows that the Winnipesaukee distance d_W can not be a function of d_x and d_y. Otherwise $d_W(p_1,p_2)$ and $d_W(p_1,p_3)$ would have the same value. The difference between $d_W(p_1,p_2)$ and $d_W(p_1,p_3)$ is caused by the *functional dependency* between attributes x and y in the geometry.[3] In other words, the distance between two elements in the domain space can not be determined by their distance in the x dimension and the distance in the y dimension alone. Thus, we can not treat x and y as two separate variates. Instead, we must treat the pair (x,y) – which may be considered as two variates in both Euclidean or Manhattan geometries – as a single variate in the Winnipesaukee geometry.

4.4 Pseudo-Quasimetrics

From both the data modeling and knowledge discovery perspectives, the general notion of distance functions, i.e., the weak distance functions, provides the most flexibility

$\min(\cup_{r \in V_p}\{d_M(p,r) + d_M(r,q)\})$, where V_p is the set of the corners of the lake which are "visible" to p. $d_W(p,q)$, in which exactly one of p and q is in L, can be formulated in a similar but much more complicated way.

[4]It should be noted that, in addition to the functional dependency just described, there exist other kinds of dependencies in a set of attributes. For instance, a set of attributes might exhibit *multicollinearity* [29].

and expressiveness. However, in general, the more metric axioms a distance function satisfies, the more likely it is that we can develop efficient data structures based on the distance function. Section 4.4.1 studies the impact of various metric axioms on data structures. Together with other considerations, the results of such study convince us that pseudo-quasimetric represents a good balance of conceptual expressiveness and implementation efficiency, and thus, a proper modeling primitive for representing inter-instance relationships. Section 4.4.2 examines the relationship between pseudo-quasimetrics and partial orders.

4.4.1 Motivation

An obvious way to develop a data structure based on distances is to decompose a data set into a collection of spheres. All existing *hierarchical metric data structures* are based on this principle. Axioms M1, M2, and M3 (as satisfied by a pseudo-metric) are particularly useful qualities for efficient access to spheres.

Assume that d is a distance function on X, and $S_d(p, s)$ and $S_d(q, t)$ are two spheres. M1 guarantees that the center of a sphere is always in the sphere, e.g., $p \in S_d(p, s)$ and $q \in S_d(q, t)$. M2 enables us to determine the set-inclusion relationship between $S_d(p, s)$ and $S_d(q, t)$ by $d(p, q)$ using the following rule:

$$d(p, q) \leq s - t \Longrightarrow S_d(q, t) \subseteq S_d(p, s),\ \forall p, q \in X.$$

Thus, a collection of spheres can be organized into a hierarchy based on set-inclusion relationships. With Axioms M1, M2, and M3, another rule can be established:

$$d(p, q) \geq s + t \Longrightarrow S_d(p, s) \cap S_d(q, t) = \emptyset,\ \forall p, q \in X.$$

Together, the two rules enable efficient *proximity retrievals* (e.g., find all points within distance s from point p) from the sphere hierarchy based on pseudo-metric.

Axiom M4 is not as essential, compared to M1, M2, and M3. M4 assures that every point can be differentiated from the rest based on the distance perspective. For a pseudo-metric, it is possible that there are clusters consisting of indiscernible points. This is generally not a problem, however, as long as the sizes of those clusters are not extremely large. This situation is analogous to *hashing* where a group of data points might share the same hash index. In other words, that group of points are indiscernible based on hash indexes. Other similar examples include the *bucket* varieties of many data structures.

Although M3 seems to be as essential as M1 and M2, we have discovered a simple method to transform a pseudo-quasimetric to pseudo-metric by inducing M3 artificially such that there is a precise correlation between spheres in the original pseudo-quasimetric space and the ones in the transformed pseudo-metric space (Chapter 5). This approach enables efficient access to spheres decomposition of pseudo-quasimetric space.

Thus, pseudo-quasimetric seems to be a reasonable compromise between distance function (for model expressiveness) and pseudo-metric (for implementation efficiency). In addition, there exists theoretical reason for using pseudo-quasimetrics as the modeling primitive of inter-instance relationships. Pseudo-quasimetrics guarantee that the *neighborhood space* [59] defined by the pseudo-quasimetric is the same as the one defined by the topology induced by it. Thus, continuity is consistent among pseudo-quasimetric spaces and topological spaces from the perspective of distances. Interested readers should refer to [10] for more details on this matter.

The rest of this chapter can be equally applied to general distance functions which are particularly useful for knowledge discovery purpose where the data structure issues are of no concern. For convenience, we use the term *pq-metric* for pseudo-quasimetric in subsequent context.

4.4.2 From Pseudo-Quasimetrics to Partial Orders

A pq-metric space can be ordered in many different ways. For a space of size n, there are 2^{n^2} possible orderings.[4] *Metric-based orders* are derived from a pq-metric (or a set of pq-metrics) on the space, while other orders are not. We are only interested in metric-based orders.

There are many ways to derive metric-based orders. We describe one such method. First, we pick a fixed point p in the pq-metric space, called the *geometric center*. Second, all the points are ordered based on their distances from the geometric center. Clearly, the resulting order is a preorder, since there might be more than one point at the same distance from the geometric center (i.e., there might be no antisymmetry). Note that only a point pq-metric on p is required for this ordering.

Let the domain space D be a square piece of land as defined in Section 4.3. Suppose there is a lake region L on D (Figure 4.4(a)). We now define a geometry similar to the

[4]This is the number of different binary relations which can be defined on a set of size n. Thus, even non-preorder binary relations are accounted for.

Winnipesaukee geometry, except we allow movements in the x and y directions simultaneously. Because of the existence of the impassable lake region, Euclidean distance does not always reflect the practical distance between two points in this geometry. Since we do not know how to formulate the pq-metric on the geometry mathematically, ordering D based on a pq-metric seems to be a rather difficult task.

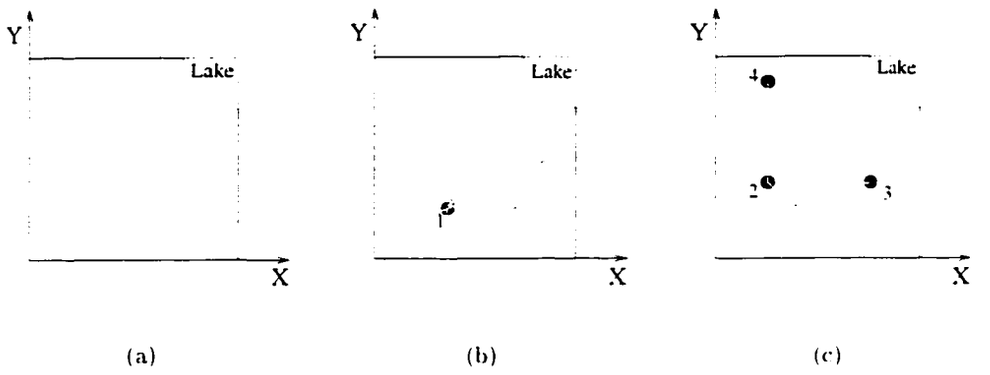


Figure 4.4: Choosing a Geometric Center

However, if we choose point p_1 as the geometric center, the Euclidean distance d_E would indeed reflect the practical distances between p_1 and all the points in D (Figure 4.4(b)). In other words, d_E can be taken as a valid point pq-metric on p_1. Thus, the domain space can be ordered based on $d_E|_{p_1}$. If point p_2 is chosen to be the geometric center instead, we have $d_E(p_2, p_3) = d_E(p_2, p_4)$ which does not indicate the practical distance from p_2 to p_3 (Figure 4.4(c)). Formulating a valid point pq-metric on p_2 might require formulating a partial pq-metric on $D - L$.

This example demonstrates an important observation that a comprehensive pq-metric is not always required to derive a metric-based order of the domain space. A valid point pq-metric can be induced by a comprehensive pq-metric on a much simpler geometry, and a metric-based order can be derived from that point pq-metric.

4.5 Observable Properties

In the course of the metric-driven knowledge discovery process, the derivation of domain pq-metrics is based upon the individual variate pq-metrics. A variate might have

an explicit and natural distance function or order from which a variate pq-metric can be derived relatively easily. Nonetheless, from time to time, there exist variates

1. which have no explicit order or distance function defined on them, or
2. whose explicit order or distance function are meaningless with respect to their semantic structure, or
3. for which an alternative order or distance function can be useful for identifying their semantic structure.

Nominal categorical data [1] consist of variates which satisfy the first two cases. Scientific data subject to classification satisfies the third. In this section, we present a systematic approach to derive pq-metrics for those variates.

4.5.1 Order-Theoretic versus Set-Theoretic

Given an arbitrary variate (or set) X void of any mathematical structure, there are two basic ways to "organize" X (i.e., define a mathematical structure on X): the *order-theoretic* approach and the *set-theoretic* approach.

The *order-theoretic* approach is based on *binary relations*. Through the introduction of a binary relation, say r, (X, r) can be a preorder, partial order, total order, lattice, etc. However, defining a binary relation on X itself is often a very difficult task, because a binary relation involves direct comparison of pairs of points in X. In many aspects, defining a binary relation is very similar to defining a distance function. For instance, given a binary relation r on X and $p, q \in X$, a distance function d can be defined as,

$$\begin{aligned} d(p,q) &= 0, && \text{if } (p,q) \in r; \\ d(p,q) &= 1, && \text{otherwise.} \end{aligned}$$

Due to the amount of domain knowledge involved and its inherent complexity, the order-theoretic approach is of limited use for organizing an arbitrary set.

The *set-theoretic* approach is based on *differentiation*. The goal of differentiation is to distinguish a subset of X from the rest of X. From the logic perspective, differentiation is achieved by applying a *predicate* on elements of X. Those satisfying the predicate are differentiated from the ones that fail. For $p \in X$, the satisfaction of a certain predicate φ

can be viewed as p possesses an *observable property*[5] φ. From a slightly different view point, the subset of X differentiated by φ can be "identified" with φ. Thus, an observable property or a predicate on X is identified with a subset of X. After multiple differentiations, the structure of X can be defined as $\mathcal{O} \subseteq \mathcal{P}(X)$, which is called a *set of observable properties* of X.

Extra care has to be taken in the process of "identifying" an observable property with a subset of X. For instance, the set of elements in X having property P_1 may be exactly the same as the set having property P_2. In this case, the set is to be identified with a new observable property $P_1 \wedge P_2$. From the set-theoretic perspective, property P_1 and property P_2 are not distinguishable in X.

Although there might be no apparent order on X, $\mathcal{P}(X)$ is a partial order based on the set-inclusion relation and $\mathcal{O} \subseteq \mathcal{P}(X)$ can have additional mathematical properties as a subset of a partial order. For instance, $\mathcal{O}$ can be an *upper set*, a *lower set*, an *ideal*, or a *filter* on $\mathcal{P}(X)$ [33]. Apart from the set-inclusion relation, $\mathcal{O}$ can also have a set-theoretic structure such as topology. The idea of observable properties can be regarded as a generalization of nominal categorical data in which each distinct data value is considered as an observable property itself.

4.5.2 Property-Based Metrics

Given a set X and a set of observable properties $\mathcal{O} \subseteq \mathcal{P}(X)$, there are many ways to derive a pq-metric on X based on $\mathcal{O}$. A pq-metric derived in this way is called a *property-based pq-metric* or a *set-theoretic pq-metric*. Based on their formulations, property-based pq-metrics can be classified into two categories: *population-independent* and *population-dependent*. A population-independent property-based pq-metric on X is a pq-metric that assigns fixed metric values even if the membership of X changes, as long as the sets of observable properties possessed by the original elements in X stay the same. In other words, it is independent from the population in the variate domain X.

Definition 4.5.1 (Population-Independence) *For a set X and $\mathcal{O} \subseteq \mathcal{P}(X)$, a pq-metric $d_{(X,\mathcal{O})}$ is* population-independent *if and only if for arbitrary set X' and $\mathcal{O}' \subseteq \mathcal{P}(X')$ such that $\mathcal{O}|_{X \cap X'} = \mathcal{O}'|_{X \cap X'}$, $d_{(X,\mathcal{O})}(p,q) = d_{(X',\mathcal{O}')}(p,q), \forall p,q \in X \cap X'$, where $\mathcal{O}|_S = \{P \cap$*

[5] A property is *observable*, if its presence or absence is apparent, beyond all doubt. The time required to determine that is not an issue. In other contexts [58], observable property means that the presence (not absence) of such a property can be determined in a finite amount of time.

$S|P \in \mathcal{O}\}.\forall S \subseteq X$.[6] *A pq-metric* $d_{(X,\mathcal{O})}$ *is* population-dependent *if and only if it is not population-independent.*

Population-independence is not always a desirable characteristic for a pq-metric, especially when X is not fixed. From time to time, we need a pq-metric which can distinguish objects with rare properties from the common ones - those constituting a significant part of the variate domain. For instance, we might want to identify those data values with rare qualities in some categorical data. *Property topologization* is one way to derive a *population-dependent* pq-metric.

4.5.3 Property Topologization

Property topologization is a process for augmenting a set of observable properties by making each observable property an open set and constructing the least topology having those opens. The resulting topology is made to be the new augmented set of observable properties. For a set of observable properties $\mathcal{O}$, we use the notation $\mathcal{O}^*$ to represent the augmented set of observable properties. Mathematically, $\mathcal{O}^*$ is derived by the following process:

1. Find all possible intersections among elements in $\mathcal{O}$. This results in a *base*[7] for $\mathcal{O}^*$.

2. Each element of $\mathcal{O}^*$ is the union of some subset of the base.

It can be easily verified that $\mathcal{O}^*$ derived this way is indeed the least topology containing $\mathcal{O}$. Although this property topologization process seems computationally expensive, more efficient algorithms can be applied when the initial set of observable properties is *well-behaved* - satisfying certain mathematical criteria.

It should be noted that even though domain knowledge is required for identifying the original observable properties, the topologization process itself (deriving $\mathcal{O}^*$ from $\mathcal{O}$) can be done automatically without domain knowledge and human intervention. Properly implemented, the process can be made transparent to users.

[6] In fact, the definition of population-independence can be made less restrictive by replacing the condition $d_{(X,\mathcal{O})}(p,q) = d_{(X',\mathcal{O}')}(p,q)$ with $d_{(X,\mathcal{O})}(p,q) = c \cdot d_{(X',\mathcal{O}')}(p,q)$ where $c \in \mathbb{R}$. After all, two pq-metrics can be considered as "equivalent" on a set X if they differ only by a constant factor on all the points in X. This equivalence relation can be further generalized to *ordered equivalence*. Two pq-metrics d_1 and d_2 are ordered equivalent on X, if and only if $d_1(p,q) \leq d_1(p,r) \iff d_2(p,q) \leq d_2(p,r).\forall p,q,r \in X$.

[7] A *base* of a topology $\mathcal{T}$ is a subset $\mathcal{B}$ of $\mathcal{T}$ such that every open set is a union of elements of $\mathcal{B}$.

4.5.4 Metric Evaluation

There are many ways to formulate pq-metrics based upon the set of observable properties. The choice of such formulations depends on (1) the set-theoretic structure of the set of observable properties, and (2) the semantic structure of the variate domain. Thus, variate pq-metric formulation might or might not involve domain knowledge.

If d is an appropriate variate pq-metric based on $\mathcal{O}$, ideally most properties in $\mathcal{O}$ should be opens in $\mathcal{T}_d$, the topology induced by d. Ideally, if d' is an appropriate variate pq-metric based on $\mathcal{O}^*$, the topologization of $\mathcal{O}$, $\mathcal{T}_{d'}$ should be *similar*[8] to $\mathcal{O}^*$. Nonetheless, as illustrated in Example 3.4.8, the same topology can be induced by very different pq-metrics. Thus, evaluation of this kind is not a precise one. True evaluation and validation of pq-metrics have to be done at a more abstract level (Section 4.8).

4.6 Variate Metric Derivation

In Chapter 3, and Section 4.5, we establish the foundation for deriving property-based variate pq-metrics through a set-theoretic approach. The complete process is illustrated in Figure 4.5. Note that domain knowledge is only required for the very first step of the process - identifying the initial set of observable properties. In this section, the derivation process is demonstrated in detail through a simple example based upon the six steps described in Figure 4.5.

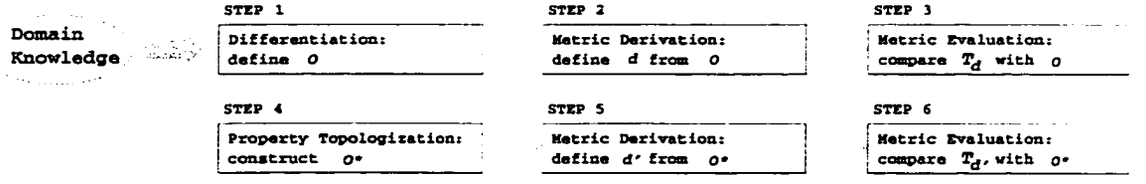


Figure 4.5: The Process of Variate Metric Derivation

Consider a variate $X = \{9, 11, 12, 16, 20, 25, 30, 49\}$. Although there is a natural order and metric on integers, let us ignore that and treat X as unordered and nonmetric. In fact, we derive a pq-metric on X which is completely different from the natural one. Each positive integer other than 1 can be decomposed into a product of *prime numbers*. In

[8] Unfortunately, there is no precise definition for one set to be "similar" to another. Here, we loosely say two sets are similar if their intersection is relatively large compared to their sizes.

this example, we take the view that the existence of a specific prime number in an integer's product expansion is an observable property. The prime factorization of each element in X is presented below:

9	$= 3^2$	**11**	$= 11^1$	**12**	$= 2^2\,3^1$	**16**	$= 2^4$
20	$= 2^2\,5^1$	**25**	$= 5^2$	**30**	$= 2^1\,3^1\,5^1$	**49**	$= 7^2$

STEP 1 Although there are five prime numbers - 2, 3, 5, 7, and 11 - that exist in the expansions of the integers in X, let us assume that only the existence of 2, 3, and 5 are observable or of interest. Let P_2, P_3, and P_5 denote the subsets of integers in X containing prime numbers 2, 3, and 5 respectively. Neither **11** or **49** has 2, 3, or 5 as prime factors. The fact itself can be defined as an additional observable property, $P_\emptyset$. We have

$$P_2 = \{\mathbf{12}, \mathbf{16}, \mathbf{20}, \mathbf{30}\}.\ P_3 = \{\mathbf{9}, \mathbf{12}, \mathbf{30}\}.\ P_5 = \{\mathbf{20}, \mathbf{25}, \mathbf{30}\}.\ P_\emptyset = \{\mathbf{11}, \mathbf{49}\}.$$

Let $\mathcal{O} = \{P_2, P_3, P_5, P_\emptyset\}$ represent a set of observable properties. Note that although integers are used in X, they are actually treated like nominal categorical data. Once we have $\mathcal{O}$ defined, the elements in X are only observable through $\mathcal{O}$. For instance, with $\mathcal{O} = \{P_2, P_3, P_5, P_\emptyset\}$, the fact that **16** has four 2's in it is no longer observable. From $\mathcal{O}$, all we know is that **16** has at least one 2 factor.

The introduction of $P_\emptyset$ is to make sure that $\mathcal{O}$ is a *cover* of X (i.e., $X \subseteq \bigcup_{P \in \mathcal{O}} P$) a convenient feature which allows easier metric formulation. If $P_\emptyset = \emptyset$, it can be safely removed from $\mathcal{O}$. Similarly, if any of the properties of interest results in an empty set, it can also be removed from $\mathcal{O}$.

STEP 2 A trivial metric d_1 can be defined as

$$\begin{aligned} d_1(p, q) &= 0. && \text{if } p = q; \\ d_1(p, q) &= |\mathcal{O}| - Z_{\mathcal{O}}(p, q). && \text{otherwise.} \end{aligned}$$

where $Z_{\mathcal{O}}(p, q)$ is the number of observable properties shared by p, q, i.e., $Z_{\mathcal{O}}(p, q) = |\{P \in \mathcal{O} | \{p, q\} \subseteq P\}|, \forall p, q \in X$. It is easy to verify that d_1 is indeed a metric according to Definition 3.2.4. The intuition of d_1 is that the more properties $p, q \in X$ share, the closer they are. For instance, $d_1(\mathbf{12}, \mathbf{30}) = 2 < d_1(\mathbf{12}, \mathbf{16}) = 3 < d_1(\mathbf{12}, \mathbf{25}) = 4$. In Table 4.1, all values of d_1 over X are listed.

STEP 3 Since metric d_1 satisfies M4, $\mathcal{T}_{d_1} = \mathcal{P}(X)$ (see Example 3.4.8). We have $\mathcal{O} \subseteq \mathcal{T}_{d_1}$ and every observable property is an open in $\mathcal{T}_{d_1}$. In fact, the formulation of d_1

d_1	9	11	12	16	20	25	30	49
9	0	4	3	4	4	4	3	4
11	4	0	4	4	4	4	4	3
12	3	4	0	3	3	4	2	4
16	4	4	3	0	3	4	3	4
20	4	4	3	3	0	3	2	4
25	4	4	4	4	3	0	3	4
30	3	4	2	3	2	3	0	4
49	4	3	4	4	4	4	4	0

Table 4.1: Distances Specified by d_1 over X

is general enough that for an arbitrary set X and $\mathcal{O} \subseteq \mathcal{P}(X)$ such that $P_\emptyset \in \mathcal{O}$, we have $\mathcal{T}_{d_1} = \mathcal{P}(X)$ and $\mathcal{T}_{d_1}$ has every property as an open. It can be seen from the formulation that d_1 is population-independent. The objective of subsequent steps is to derive a population-dependent metric.

STEP 4 Since $(P_2 \cup P_3 \cup P_4) \cap P_\emptyset = \emptyset$, the topologization of $\mathcal{O}$ can be made easier by dividing $\mathcal{O}$ into $\mathcal{O}_1 = \{P_\emptyset\}$ and $\mathcal{O}_2 = \{P_2, P_3, P_5\}$, and topologizing separately $\mathcal{O}_1$ and $\mathcal{O}_2$. After $\mathcal{O}_1^*$ and $\mathcal{O}_2^*$ are derived, from the definition of topology we have $O^* = \{A_1 \cup A_2 | A_1 \in \mathcal{O}_1^*, A_2 \in \mathcal{O}_2^*\}$. We have $\mathcal{O}_1^* = \{\emptyset, \{\mathbf{11.49}\}\}$ and $\mathcal{O}_2^*$ consisting of the following 14 open sets:

$\emptyset$
$\{\mathbf{30}\}$
$\{\mathbf{12.30}\}$ $\{\mathbf{20.30}\}$
$\{\mathbf{9.12.30}\}$ $\{\mathbf{12.20.30}\}$ $\{\mathbf{20.25.30}\}$
$\{\mathbf{9.12.20.30}\}$ $\{\mathbf{12.16.20.30}\}$ $\{\mathbf{12.20.25.30}\}$
$\{\mathbf{12.16.20.25.30}\}$ $\{\mathbf{9.12.20.25.30}\}$ $\{\mathbf{9.12.16.20.30}\}$
$\{\mathbf{9.12.16.20.25.30}\}$

Thus $\mathcal{O}^*$ has 28 open sets in total which include all the members in $\mathcal{O}_2^*$ and the union of $\{\mathbf{11.49}\}$ with each element in $\mathcal{O}_2^*$. Each open set corresponds to an observable property. For instance, $\{\mathbf{9.12.20.30}\} = P_3 \cup (P_2 \cap P_5)$ corresponds to the property $P_3 \vee (P_2 \wedge P_5)$.

STEP 5 There might or might not be a pq-metric (see Section 3.4) which can induce $\mathcal{O}^*$. For $p, q \in X$, let us define a pq-metric d_2 on X as

$$d_2(p, q) = Z_{\mathcal{O}^*}(p, p) - Z_{\mathcal{O}^*}(p, q).$$

where $Z_{\mathcal{O}^*}(p, q)$ is the number of opens in $\mathcal{O}^*$ containing both p and q (the same function Z used in the definition of d_1). Clearly, $d_2(p, p) = 0, \forall p \in X$. It should be obvious that d_2

is not symmetric although $Z_{\mathcal{O}\cdot}(p,q) = Z_{\mathcal{O}\cdot}(q,p), \forall p,q \in X$. In Tables 4.2 and 4.3, we list all $Z_{\mathcal{O}\cdot}$ and d_2 values over X. For $p,q,r \in X$, we have

$$d_2(p,r) + d_2(r,q) - d_2(p,q) = Z_{\mathcal{O}\cdot}(r,r) + Z_{\mathcal{O}\cdot}(p,q) - Z_{\mathcal{O}\cdot}(p,r) - Z_{\mathcal{O}\cdot}(r,q) \geq 0.$$

and thus d_2 satisfies the triangle inequality. It should be noted that $P_\emptyset \in \mathcal{O}$ assures $d_2(p,q) \neq 0$ for $p \in P_\emptyset$ and $q \in X - P_\emptyset$.

$Z_{\mathcal{O}\cdot}$	**9**	**11**	**12**	**16**	**20**	**25**	**30**	**49**
9	10	5	10	4	8	4	10	5
11	5	14	10	4	10	5	13	14
12	10	10	20	8	16	8	20	10
16	4	4	8	8	8	4	8	4
20	8	10	16	8	20	10	20	10
25	4	5	8	4	10	10	10	5
30	10	13	20	8	20	10	26	13
49	5	14	10	4	10	5	13	14

Table 4.2: Values of $Z_{\mathcal{O}\cdot}$

d_2	**9**	**11**	**12**	**16**	**20**	**25**	**30**	**49**
9	0	5	0	6	2	6	0	5
11	9	0	4	10	4	9	1	0
12	10	10	0	12	4	12	0	10
16	4	4	0	0	0	4	0	4
20	12	10	4	12	0	10	0	10
25	6	5	2	6	0	0	0	5
30	16	13	6	18	6	16	0	13
49	9	0	4	10	4	9	1	0

Table 4.3: Values of d_2

The pq-metric d_2 can be interpreted in several different ways, depending on how we choose to interpret the collection of observable properties $\mathcal{O}$. Some possible interpretations include:

1. Let us look at all the point pq-metrics $d_2|_{\mathbf{9}}, d_2|_{\mathbf{11}}, \ldots, d_2|_{\mathbf{49}}$, which correspond to rows in the table of d_2: clearly **30** is the most distant from all the others. This is because **30** has a rare quality - being the only number having all three properties P_2, P_3, and P_5. On the other end of the spectrum, **16** is the closest to all the others. This is because the only property **16** has is the most common one P_2.

2. It is easy to see that $d_2|_9 \sim d_2|_{25}$ (i.e., $d_2|_9$ and $d_2|_{25}$ give the exact same collection of distance assignments, only in different permutations). This tells us that the sets of properties possessed by **9** and **25** have similar occurrence frequencies in the variate population, although they might be very different. The same thing can be said for $d_2|_{12} \sim d_2|_{20}$. The observation, $d_2|_{11} = d_2|_{49}$, indicates that **11** and **49** are indistinguishable in $\mathcal{O}$. From another perspective, pq-metric d_2 is only symmetric on pairs **(9, 25)**, **(12, 20)**, and **(11, 49)**.

3. Taking $d_2|_{20}$ for example, we have $d_2(\mathbf{20}, \mathbf{30}) = 0$, since for all the observable properties in $\mathcal{O}$ (also $\mathcal{O}^*$), the existence of **20** always implies **30**. In other words, **30** has all the observable properties **20** possesses. $d_2(\mathbf{20}, \mathbf{12}) = 4$ tells us that **20** and **12** are likely to cluster together, since $d_2(\mathbf{20}, \mathbf{30}) = d_2(\mathbf{12}, \mathbf{30}) = 0$. Further, **25** is a bit closer to **20** than **9** and **16**, because $d_2(\mathbf{25}, \mathbf{20}) = 0$.

There exist many other interpretations as well. The important thing is that d_2 is population-dependent. For instance, adding elements to X or removing elements from X may affect the distance measure between two original elements.

STEP 6 Now, we want to ask: *How good is d_2 in modeling (or describing) $\mathcal{O}^*$?* To answer this question, we have to compare $\mathcal{T}_{d_2}$, the topology induced by d_2, with $\mathcal{O}^*$. Let $S_p, \forall p \in X$ represent the smallest of all possible spheres $S_{d_2}(p, \varepsilon)$ centered at p (i.e., $\{q | d_2(p, q) < \varepsilon\}$ as $\varepsilon \longrightarrow 0$).

$$
\begin{array}{llll}
S_{\mathbf{9}} = \{\mathbf{9}, \mathbf{12}, \mathbf{30}\} & & S_{\mathbf{11}} = \{\mathbf{11}, \mathbf{49}\} & \\
S_{\mathbf{12}} = \{\mathbf{12}, \mathbf{30}\} & & S_{\mathbf{16}} = \{\mathbf{12}, \mathbf{16}, \mathbf{20}, \mathbf{30}\} & \\
S_{\mathbf{20}} = \{\mathbf{20}, \mathbf{30}\} & & S_{\mathbf{25}} = \{\mathbf{20}, \mathbf{25}, \mathbf{30}\} & \\
S_{\mathbf{30}} = \{\mathbf{30}\} & & S_{\mathbf{49}} = \{\mathbf{11}, \mathbf{49}\} &
\end{array}
$$

Using the spheres, $\mathcal{T}_{d_2}$ can be easily derived. If d_2 is reasonably well defined, $\mathcal{T}_{d_2}$ would not be far different from $\mathcal{O}^*$. It is not hard to see that $\mathcal{T}_{d_2} = \mathcal{O}^*$ and, therefore, d_2 is an *optimal* pq-metric of $\mathcal{O}^*$.

In the above example, other observable properties can be defined to describe the variate values more precisely. For instance, if we put a new value **8** into X, there is no way to distinguish **8** from **16** using $\mathcal{O} = \{P_2, P_3, P_5, P_\emptyset\}$ or $\mathcal{O}^*$. With a more comprehensive $\mathcal{O}$, we can derive pq-metrics based on subtler characteristics, such as the greatest common divisor of two elements - the greater the *gcd*, the shorter the distance. In fact, in the metric-driven knowledge discovery process, the formulation of the set of observable properties might be

progressively refined based on domain knowledge and the validity of the metric derived (see Figure 4.5).

Property topologization appears to be a viable approach to derive pq-metrics from unordered and nonmetric variate domains, such as nominal categorical data. Even though domain knowledge is required for identifying the original observable properties, the topologization process itself (deriving $\mathcal{O}^*$ from $\mathcal{O}$) can be done automatically without domain knowledge and human intervention.

4.7 Domain Metric Derivation

In Sections 4.5 and 4.6, we demonstrated the process for deriving variate pq-metrics. Based on variate pq-metrics, we developed a systematic approach for domain pq-metric derivation of a wide range of domain spaces. Given a data set to be studied, the domain metric derivation process consists of the following steps:

1. Identify the data function.
2. Identify the variates.
3. Derive pq-metrics for each of the variates.
4. Aggregate the variate pq-metrics into the domain pq-metric (i.e., composite metric).
5. Validate the domain pq-metric.

Although all the steps involve domain-specific knowledge, once formulated, that knowledge is applied in an algorithmic way.

Many different data functions can be defined on a data set, although it might not always be clear what the "right" data function is. Sometimes, the domain-specific knowledge might also be very difficult to formulate. Let us illustrate the metric derivation process (excluding metric validation, which is the subject of Section 4.8) by an example.

Example 4.7.1 *CAM (Computer-Assisted Matchmaker) is a project supported by a major computer professional association to help its members find persons they would like to date. CAM consists of a large database of personal information for its participating members. For simplicity, let us assume each entry in the database is a record of the form:*

(GENDER, AGE, HEIGHT, WEIGHT, LEISURE_ACTIVITIES, EMAIL_ADDRESS).

We denote the six attributes by G, A, H, W, L, and E. For convenience, we also use the same set of symbols to represent respective attribute domains. A search of CAM consists of the following steps:

1. *The user specifies a record which describes his/her perfect date – the search target.*
2. *The user describes a pq-metric to be used for measuring the distances between the search target and all the records in CAM.*
3. *The user specifies the selection criterion (e.g., k-nearest neighbors, or all entries within a preset distance).*
4. *CAM does the search and outputs the results.*

In CAM, the data function is not obvious from the data set itself. Can the value space be one of the six attribute domains G, A, H, W, L, and E, or is it a cross product of more than one of these attribute domains? Does the value space contain domains of derived attributes?

Since we are interested in the pq-metrics among entries in the data set (i.e., the domain geometry of the function), we would like to take the cross product of all attribute domains of the data set as the domain space. The value space should be a set of domain pq-metric values. However, a point in the domain space does not give us a pq-metric value automatically, because pq-metric values take on pairs of points instead of points. In other words, the value space should be a metric-based order on which the selection is based. We can choose a fixed point o in the domain space as the geometric center, and make the value space be the distances between o and points in the domain space. Thus, the value space can be ordered based on the usual order of $\mathbb{R}$. In CAM, the geometric center should be the point specified by the user as the search target in Step 1. Now, the data function f of CAM can be defined as

$$f : G \times A \times H \times W \times L \times E \longrightarrow X,$$

where $X \subseteq \mathbb{R}^+ \cup \{0\}$ is the distance between the given domain point and the geometric center as specified in Step 2. Note that it is necessary to include E as part of the domain space, so there exists a bijection (one-one and onto) between a CAM entry and a point in

the domain space. From the perspective of the relational data model, E is the key *of the CAM database. However, E need not participate in the pq-metric.*

The first step of pq-metric derivation is to identify the variates since some attributes might be functionally dependent, *or we might want to group several attributes into a single variate. One such possibility is to integrate H and W into a single variate, so we can group people according to their physique, such as slim, average, full-figured, and so on. For simplicity, however, let us treat H and W as two separate variates. Assuming there are only two possible values for G - male and female - d_G can be defined as*

$$d_G(p,q) = \begin{cases} 0. & \text{if } p,q \text{ are both male or female:} \\ 1. & \text{otherwise.} \end{cases}$$

For A, H, and W, let us use the usual distance (Section 4.3) functions on them as their pq-metrics. E not only serves as an identifier for each entry in CAM, it might also contain affiliation information of the person. Conceptually, we can also define a nontrivial pq-metric on E. Nonetheless, for simplicity, let us ignore the pq-metric contribution of E by defining $d_E(p,q) = 0, \forall p,q$.

L is the variate which is not ordered. It would be difficult to derive a meaningful pq-metric on it directly. For simplicity, let us assume $L = \mathcal{P}(L_0)$, *where* L_0 = { BOWLING, COOKING, GOLF, MOVIES, PAINTING, READING, SKIING, TENNIS, VIDEO_GAMES }.

Let us examine one trivial pq-metric which can be derived directly from L: Map each member in L into a binary string of 9 *bits. The leftmost bit corresponds to* BOWLING, *the first element in* L_0, *the second bit from the left corresponds to* COOKING, *the second element in* L_0, *and so on.* 1 *indicates the presence of the hobby while* 0 *indicates the absence of the hobby. The distance between two elements in L is defined as the number of* 1*'s in the result of a bit-wise XOR operation of their corresponding bit strings. Let us denote this metric as* d_L^*. *Now consider* p_1 = { GOLF, TENNIS }, p_2 = { BOWLING, SKIING }, *and* p_3 = { READING, PAINTING }. *Even though* $d_L^*(p_1,p_2) = d_L^*(p_1,p_3) = 4$, *we can see that both* p_1 *and* p_2 *are sports oriented, implying that* p_1 *should probably be closer to* p_2 *than to* p_3.

The reason for the above problem is that we implicitly treat each element in L_0 *as equal. In other words, we implicitly assume a non-zero constant distance between all pairs of different elements in* L_0. *A better pq-metric can be derived by imposing some groupings on L.*

Notice that L is a relatively large set ($|L| = 2^{|L_0|} = 2^9 = 512$) and defining groups based on L can be a tedious and inefficient task. In cases such as this, it is often easier to define observable properties as subsets of L_0 instead of L. Suppose we are interested in the following four properties of L_0: PHYSICALITY, COMPETITIVENESS, INFORMATIVENESS, *and* CREATIVITY.[9] *Table 4.4 depicts the relationship between activities in L_0 and those properties.*

ACTIVITIES IN L_0	PHYSICALITY	COMPETITIVENESS	INFORMATIVENESS	CREATIVITY
BOWLING	x	x		
COOKING				x
GOLF	x	x		
MOVIES			x	
PAINTING				x
READING			x	
SKIING	x			
TENNIS	x	x		
VIDEO_GAMES		x	x	x

Table 4.4: Observable Properties for L_0

Now, we can impose a topology ΩL_0 on L_0 which is the least topology (ordered by set-inclusion) that contains the four property-based opens. Recall $L = \mathcal{P}(L_0)$ and $\Omega L_0 \subseteq \mathcal{P}(L_0)$. In other words, L is the discrete topology (the finest topology) of L_0, and $\Omega L_0 \subseteq L$ is coarser. For $p, q \in L$, the distance $d_L(p, q)$ can be defined as the number of opens of ΩL_0 containing p minus the number of opens of ΩL_0 containing both p and q.

Now we have a pq-metric for each of the variates. Since each variate is independent of the others, the domain pq-metric is a function of the variate pq-metrics. In the simplest form, the domain pq-metric can be a linear combination of the variate pq-metrics:

$$d \equiv c_G \cdot d_G + c_A \cdot d_A + c_H \cdot d_H + c_W \cdot d_W + c_L \cdot d_L,$$

where $c_G, c_A, c_H, c_W, c_L \in \mathbb{R}$. ∎

The topologization process is highly adaptive to variates with various domain structures – the structure among points in the variate domain – as can be seen in the prime numbers example in Section 4.6 and Example 4.7.1. It is also population-dependent and thus the population information can be incorporated into the derived pq-metric if it is desirable.

[9] We can utilize the idea of *specialization preorder* [37] of open sets to describe similar properties at different levels. For example, we could further classify a property such as PHYSICALITY into several different levels of intensities (e.g., low, medium, and high).

The topology-based pq-metric definitions presented in the examples are by no means the only ones which can be defined. Lots of possibilities exist and need to be explored further.

4.8 Continuity for Metric Validation

As shown in Section 4.7, the domain metric derivation process is guided by domain-specific knowledge. In this process, some decisions and compromises have to be made. Nonetheless, no matter how the domain metric is derived, either through the process just described or other approaches, we need to validate it.

The validation is based on the mathematical notion of continuity. The continuity of a data function depends on the neighborhood structures of both its domain space and value space (Definition 3.3.14). Since the neighborhood structure on the value space is usually known in advance, the goal of domain metric derivation is to derive a pq-metric such that the neighborhood structure on the domain space induced by the pq-metric would make the data function continuous. Thus, a pq-metric can be validated if it preserves the continuity of the data function. In [34], we define continuity in several different mathematical settings and describe its semantics in the context of data analysis. We demonstrate the relationship between data continuity and metric validity in the following example.

Example 4.8.1 *As in the prime numbers example of Section 4.6, let*

$$X = \{\mathbf{9}, \mathbf{11}, \mathbf{12}, \mathbf{16}, \mathbf{20}, \mathbf{25}, \mathbf{30}, \mathbf{49}\}$$

be the domain space of interest. We define a data function $f : X \longrightarrow C$, where $C = \{a, b, c\}$, as follows:

$$\begin{array}{llllllll} f(\mathbf{9}) & = a & f(\mathbf{11}) & = a & f(\mathbf{12}) & = b & f(\mathbf{16}) & = a \\ f(\mathbf{20}) & = b & f(\mathbf{25}) & = a & f(\mathbf{30}) & = c & f(\mathbf{49}) & = a \end{array}$$

The intuition of f is that $f(p)$ is an indication of the number of different observable properties p possesses, so $P_2 = \{\mathbf{12}, \mathbf{16}, \mathbf{20}, \mathbf{30}\}$, $P_3 = \{\mathbf{9}, \mathbf{12}, \mathbf{30}\}$, $P_5 = \{\mathbf{20}, \mathbf{25}, \mathbf{30}\}$, and $P_\emptyset = \{\mathbf{11}, \mathbf{49}\}$. Thus, a, b, and c correspond to one, two, and three observable properties respectively. In order to evaluate the continuity of f, we have to know the geometries of domain space X and value space C. Let us specify the value geometry by the topology $\Omega C = \{\emptyset, \{c\}, \{b, c\}, \{a, b, c\}\}$. The idea of ΩC is that the opens $\{a, b, c\}$, $\{b, c\}$, and $\{c\}$ correspond to "having at least one, two, and three observable properties" respectively. Let

N_p. $\forall p \in C$ represent the smallest neighborhood of p. We have $N_a = \{a, b, c\}$. $N_b = \{b, c\}$, and $N_c = \{c\}$. A pq-metric d_3 on domain X can be defined as $d_3(p, q) = |K_p - K_q|$, such that $\forall x \in X$. x is the product of K_x prime numbers (for instance, $K_9 = 2, K_{11} = 1, K_{12} = 3, \ldots$). Note that $\{c\}$ is a neighborhood of $f(\mathbf{30}) = c$. However, for every neighborhood M of $\mathbf{30}$ induced by d_3, $\{\mathbf{12}, \mathbf{20}, \mathbf{30}\} \subseteq M$ and $\{b, c\} \subseteq f[M]$. Thus, f is not continuous with respect to d_3 and d_3 is not valid for f.

On the other hand, if we use the pq-metric d_2 in Section 4.6 as the domain metric, it is easy to verify that f is continuous with respect to d_2 and d_2 is valid for f. ∎

In order to validate a metric derived for CAM in Example 4.7.1, all we have to do is to check whether the data function is continuous or not. Unfortunately, this is not possible without further domain-specific knowledge on the value space. Recall that the values in the value space of CAM depend on the domain metric and the choice of geometric center. We really do not have the "true" values (i.e., values independent of the domain metric) in the value space for us to evaluate the continuity. Thus, the validation can only be done if we have an empirical data set which provides the "true" values in the value space. Alternatively, we can form a panel of consultants and psychologists to give us their expert estimation of the "true" values in the value space. Based on their opinions, we can conclude whether a given domain metric is valid or not.

Metric validation is part of the "trial-and-error" process involved in knowledge discovery. Given a data function, there are times when a mathematically valid domain metric is extremely difficult (or computationally expensive) to derive. In such cases, we might want to accept a *partially valid* domain metric such that the data function is continuous at a large subset of the domain.

4.9 Summary

Scientific data is characterized by rich and complex interrelationships, especially inter-instance relationships. Based on data-as-functions and pseudo-quasimetrics, this chapter presents a formal mathematical model suitable for modeling complex interrelationships in scientific data. Compared to the spatial data models and existing scientific data models arising in computational fluid dynamics and scientific visualization, the proposed model offers more flexibility and generality.

In addition to the mathematical foundation, we present a detailed approach for metric derivation. The approach itself is also useful as a paradigm for knowledge discovery from the metric perspective. Since our model is based on formal mathematical semantics, the results can be formally validated. The notion of continuity is used as a precise mathematical tool for validating the results of the metric derivation process, either for data modeling or knowledge discovery purposes.

From the knowledge discovery perspective, we believe the metric-based data model has tremendous potential as the foundation for developing various data mining mechanisms. In particular, the process for metric derivation based on observable properties can be very valuable for data mining in categorical data which are pervasive in social sciences.

Chapter 5

Physical Data Models

5.1 Introduction

A new data model leads to the development of data models at the lower abstraction levels. Certainly, a conceptual or implementation model is not very useful if there is no efficient physical data model to support and materialize it. While the DDL and DML of the proposed metric-based data model are yet to be developed, it suffices to have an *ad hoc* pseudo-quasimetric formulation as the implementation model for most applications. The focus of this chapter is the study of supporting data structures, i.e., physical data models, for the metric-based conceptual and implementation models.

Metric data are data where the proximity relationships are either expressed *extensionally* by explicit distance values or defined *intensionally* by a derived distance function, such as the metric-modeled scientific data discussed in Chapter 4. The most obvious physical data model for supporting metric-based models is a group of data structures known collectively as the *hierarchical metric data structures*, including *metric trees*, *vp-trees*, and their variations. However, due to the reasons detailed in Section 4.4, all existing hierarchical metric data structures can only support pseudo-metrics.

This chapter presents an innovative, yet surprisingly simple, approach for deriving a new class of hierarchical metric data structures from tested-and-proven *point spatial data structures*. Instead of performing direct decomposition on metric data as is done for metric trees and vp-trees, we define a class of simple *proximity-preserving* mappings from *pseudo-metric spaces* to *multidimensional spaces*[1], which we call *multipolar mappings*. By

[1]The term *k-dimensional space*, or *multidimensional space* in general, is often used in the literature to

applying multipolar mappings to metric data. hierarchical decompositions can be done in multidimensional space. and various point spatial data structures. such as *quadtree*. *octree*. or *k-d tree*. can be utilized for storing the metric data.

The idea of mapping data points to multidimensional spaces based on distance functions is not new. Various *multivariate data analysis* [29] techniques exist for mapping a metric data set to a multidimensional space such that the implicit structure of the data. as modeled by a distance function. can be observed or visualized. These techniques include *factor analysis* [8]. *principal component analysis* [20]. and *multidimensional scaling* [9]. Although these techniques might have a limited value for organizing static metric data. the mappings they employ are *data-dependent* and are not as useful for applications to dynamic metric data.

However. similar to the requirement of existing hierarchical metric data structures. multipolar mapping only works on pseudo-metrics. To solve both problems. we develop a simple and practical approach. *median transformation*. which can be used to derive pseudo-quasimetrics from pseudo-metrics.

The rest of this chapter starts with a brief introduction to the existing hierarchical data structures and related terminology. Based on the mathematical foundations established in Chapter 3. we present the definition of a multipolar mapping. Several important characteristics of multipolar mappings are studied in the context of employing point spatial data structures for managing metric data. Finally. we describe the notion of *median transformation* and show how it can enable both the point spatial data structures and existing hierarchical data structures to store metric data with pseudo-quasimetrics. The proofs for major theoretical results of this chapter are presented separately in Appendix B.

5.2 Hierarchical Metric Data Structures

We consider a *data set* as a finite set of *records* or *data points*. Hierarchical data structures are based on *hierarchical decompositions* of the data sets they intend to store.

represent *k-dimensional real coordinate space*. denoted by $\mathbb{R}^k$. With *vector addition* and *scalar multiplication* defined. $\mathbb{R}^k$ can be made into a *vector space over the real field* or simply a *real vector space*. The vector space $\mathbb{R}^k$ on which *inner product* and *norm* are defined is known as *Euclidean k-space* [50]. The Euclidean distance of $\mathbb{R}^k$. denoted by E_k. follows directly from the definition of norm. Although in the computer science literature the terms k-dimensional space and multidimensional space are often used to represent Euclidean k-space with Euclidean distance. we use these terms to refer to the mathematical notion of k-dimensional real coordinate space which allows definitions of new operations. such as distance functions other than the Euclidean distances.

Binary search tree [41] is a classic example. In fact, all data structures based on trees are hierarchical data structures. Common non-hierarchical data structures include *linked list*, *inverted list*, *hashing*, and *multidimensional array*.

Nievergelt, Hinterberger, and Sevcik decompose hierarchical data structures into two groups: those that organize the specific set of data to be stored, and those that organize the embedding space from which the data is drawn [47].[2] Binary search tree, *point quadtree* [52, 51], and *metric tree* [65] belong to the first group. *Region quadtree* [52, 51] and *trie*[3] [21, 25] belong to the second.

Comparative search techniques are used for the first group - the boundaries between partitions of the search space are determined by the data values to be stored. Thus, the boundaries are dynamic during the life cycle of the data structure. Let us consider a binary search tree for example. A particular data point of the left subtree at the root level might be moved to the right subtree at a later time, as the data set evolves by adding new points or deleting old points. *Address computation techniques* are used for the second group - boundaries between partitions of the data space are fixed regardless of data values. For instance, let us consider a 26-ary trie based on the alphabet A-Z. The data point corresponding to the string XEROX always belongs to the 24th subtree at the root level.

Hierarchical metric data structures are simply hierarchical data structures which employ *hierarchical metric decomposition*. All existing hierarchical metric data structures belong to the first group. In fact, they are all based on the basic notions of the *metric trees* [65].[4] There are two kinds of hierarchical metric decompositions for metric trees: *sphere decomposition* and *generalized hyperplane decomposition* (Figure 5.1).

A sphere decomposition divides the current data partition X into two subpartitions:

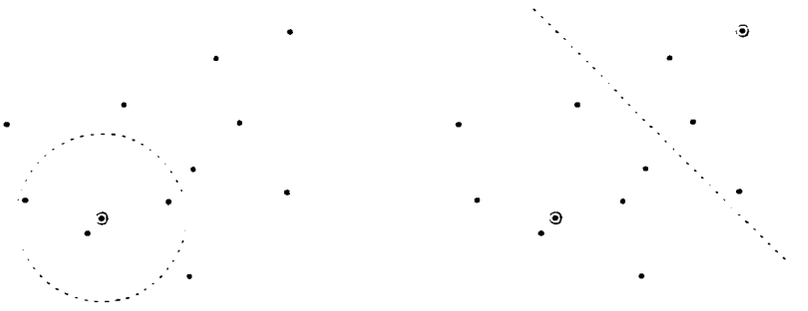
1. $S(c, r)$.
2. $X - S(c, r)$.

where $c \in X$ and $r \in \mathbb{R}^+$. Theoretically, any such c and r can be used to carry out the division. However, in order to have a more balanced metric tree, the choice of c and r at

[2] In fact, Nievergelt et al. suggest such a decomposition in the context of *search techniques* rather than hierarchical data structures. Nonetheless, the way a data set is searched is closely tied to its hierarchical organization. Thus, this decomposition can be equally applied.

[3] Trie is also known as *digital tree*.

[4] Metric trees are probably better known as *vp-trees* [67], particularly in field of *image retrieval*.



(a) Sphere (b) Generalized Hyperplane

Figure 5.1: Hierarchical Metric Decomposition

each level is crucial. There is much ongoing research in this direction [68, 3].

A generalized hyperplane decomposition divides the current data partition X into two subpartitions:

1. $\{x \in X \mid d(c_1, x) \geq d(c_2, x)\}$.
2. $\{x \in X \mid d(c_1, x) < d(c_2, x)\}$.

where $c_1, c_2 \in X$. Again, the choice of c_1 and c_2 is an important factor for balancing the tree. While it takes one distance computation in a sphere decomposition to classify a point, it takes two in a generalized hyperplane decomposition. For this reason, most current hierarchical metric data structures are based on metric trees with a sphere decomposition.

For a metric data set, there is a certain criterion its distance function has to satisfy in order for any meaningful hierarchical decomposition to take place. For instance, consider the following distance function d on X.

$$d(x, y) = \begin{cases} 0 & \text{if } x = y. \\ 1 & \text{otherwise.} \end{cases}$$

Although the distance function d is a metric, there is no meaningful hierarchical decomposition possible based on d. Yianilos proposes a criterion, called the *ZPS (Zero Probability Spheres)* property, which effectively excludes all such distance functions and many others [67]. Although, the ZPS property might be too strong for our purposes, we assume that all distance functions discussed in this paper satisfy the ZPS property.

5.3 Multipolar Mappings

In this section, we introduce the notion of *multipolar mapping* and one of the most important theoretical results of this chapter - multipolar mappings are continuous.

Definition 5.3.1 (Multipolar Mapping) *Let X be a nonempty set, $|X| = n$, and let d be a distance function on X. For $k \in \mathbb{N}$ and $k \leq n$, a* multipolar mapping with k poles *or simply a* k-polar mapping, *$M_k : (X, d) \longrightarrow \mathbb{R}^k$, is defined as*

$$M_k(x) = (d(p_1, x), \ldots, d(p_k, x)), \quad \forall x \in X,$$

where $P = \{p_1, \ldots, p_k\} \subseteq X$ is the set of poles. *To identify a specific k-polar mapping with a set of poles P, the alternative notation $M_{k,P}$ is used. Let $\mathcal{M}_k(X)$ represent all k-polar mappings on X and $\mathcal{M}(X)$ represent all multipolar mappings on X. That is,*

$$\begin{aligned} \mathcal{M}_k(X) &= \{M_{k,P} \mid \forall P \subseteq X, \ s.t. \ |P| = k\}. \\ \mathcal{M}(X) &= \bigcup_{k \leq n} \mathcal{M}_k(X). \end{aligned}$$

From the perspective of $\mathbb{R}^k$ coordinates, $M_{k,P}$ is a *lossy* transformation for $k < |X|$. The distance between two arbitrary points $x, y \in X - P$ can not be derived from $M_{k,P}(x)$ and $M_{k,P}(y)$. However, when $k = |X|$, all distance information is preserved. If $k = 1$, the result can be used to derive a *linear order* on X.

Multipolar mappings possess the important quality that they preserve the proximity relationships in pseudo-metric spaces. In the most primitive form, using the notion of distance only, the proximity-preserving quality can be informally stated as:

> If the distance from x to y is r, the distance from the image of x to the image of y is bounded by $h(r)$.

$h(r)$ is a *strictly increasing* function. The proximity-preserving quality of a mapping corresponds directly to the *continuity* of a function. The general notion of continuity is defined in the context of *neighborhood* which has many different formulations in various mathematical settings [10].

In Appendix B, we prove that given a point x in a pseudo-metric space, and an arbitrary k-polar mapping, the images of all points within distance r from x are within distance $\sqrt{k}\, r$ from the image of x. Mathematically, we have

$$M_{k,P}[S_d(x, r)] \subseteq S_{E_k}(M_{k,P}(x), \sqrt{k}\, r), \quad \forall M_{k,P} \in \mathcal{M}(X), \forall x \in X, \forall r \in \mathbb{R}^+. \tag{5.1}$$

where (X, d) is a pseudo-metric space.

Suppose we have a set of metric data X in which the proximity is defined by a pseudo-metric. After applying M_k to X, we can store the data in a k-dimensional point spatial data structure. To find all points within distance r from a particular point x, all we need to do is to perform a proximity search at $M_k(x)$ on the point spatial data structure with a search distance $\sqrt{k}\, r$.

Let X be a metric data set, $|X| = 100$ and d be its distance function. The relative distances between pairs of points in X are illustrated on a 2-D plane of Figure 5.2. In general, there is no guarantee that points of a metric data set can be plotted on a 2-D plane such that the relative distances among points are correctly illustrated. It so happens that this particular metric data set can be illustrated on a 2-D plane, because it is synthesized by computing the Euclidean distance on a set of randomly generated points in a 2-dimensional space. Let us assume that the coordinates used to compute the distances and plot the points (Figure 5.2) are not available.

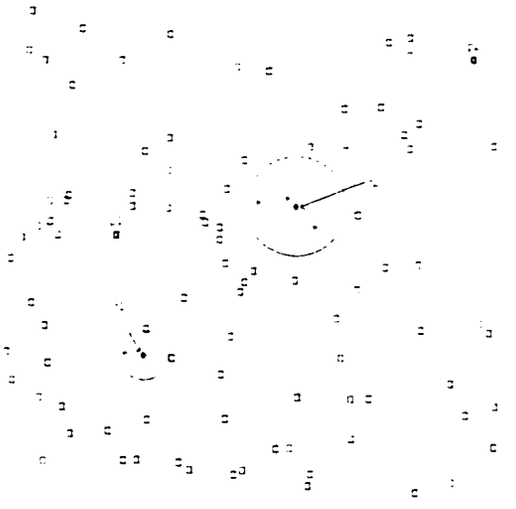


Figure 5.2: Distance Illustration of X

Two spheres, $S_1 = S_d(c_1, 5)$ and $S_2 = S_d(c_2, 10)$, are shown in Figure 5.2. Points inside the spheres are represented by $\cdot$, those outside by $\square$. S_1 and S_2 contains 2 and 4 points respectively. Recall that the only information we have about X is its distance matrix. The objective of a k-polar mapping is to map a metric data set to a k-dimensional space based exclusively on its distance information, which is either represented explicitly as

a distance matrix or implicitly as a distance function. For demonstration purposes, let us map X into a 2-dimensional space using a 2-polar mapping. Two points p_1 and p_2, both labeled in Figure 5.2, are randomly selected as poles for constructing $M_2[X]$. The issue of informed pole selection is discussed later.

Figure 5.3 illustrates the 2-dimensional space with $M_2[X]$ plotted, where $c_1' = M_2(c_1)$, $c_2' = M_2(c_2)$, $p_1' = M_2(p_1)$, and $p_2' = M_2(p_2)$. Note that the two axes A_1 and A_2 represent $d(x, p_1)$ and $d(x, p_2)$ respectively. The two spheres illustrated are $S_1' = S_{E_2}(c_1', 5\sqrt{2})$ and $S_2' = S_{E_2}(c_2', 10\sqrt{2})$. All the points in S_1 and S_2 are mapped into S_1' and S_2' respectively. We have $|S_1'|_{M_2[X]}| = 5$, and $|S_2'|_{M_2[X]}| = 10$.[5]

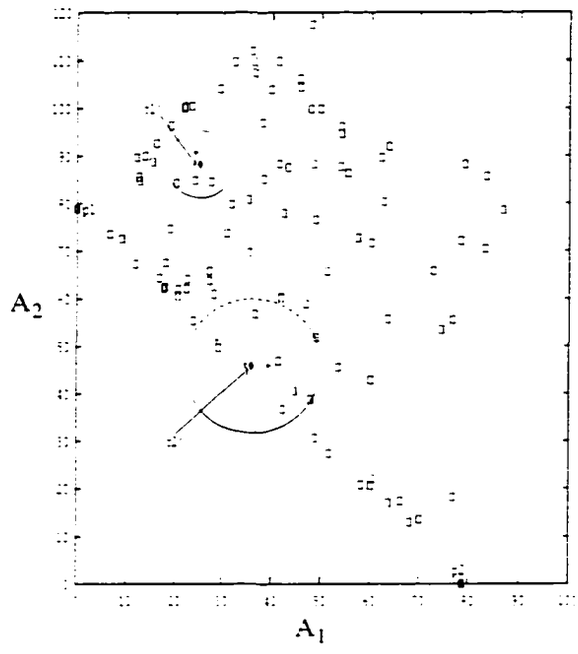


Figure 5.3: Point Plot of $M_2[X]$

Assuming that the evaluation of the distance between a pair of points in the metric data X is counted as a single basic operation, the computational complexity of M_k is $k\,n - \binom{k}{2}$ where $n = |X|$ is the size of the metric data set. Multipolar mappings are well adapted to dynamic data with frequent insertions and deletions of data records, since $M_k(p), \forall p \in X$ stays unchanged while some other points are removed and new points are added. In other words, with respect to a fixed set of poles, the formulation of a multipolar mapping is

[5]The sizes of both restricted spheres are computed directly from $M_2[X]$ instead of examining Figure 5.3. Only one of the two points near the border of $S_2'|_{M_2[X]}$ is actually inside the sphere.

data-independent, a quality not possessed by mappings employed by multivariate analysis techniques such as factor analysis, principal component analysis, and multidimensional scaling. The only exception involves the deletion of a pole. In such a case, we can always treat the record associated with the pole as a *pseudo record* and keep it for reference when new points are added in the future.

It is possible to have two different points x_1, x_2 in a pseudo-metric space (X, d) such that $d(x_1, x_2) = 0$. From the triangle inequality, $d(p, x_1) = d(p, x_2), \forall p \in X$. In other words, the images of x_1 and x_2 are the same no matter what multipolar mapping is applied. This is not a problem, however. All we need to do is to associate a set of data records where the distances among them are 0 to a single point in the point spatial data structure we choose to implement. Although a similar technique can be easily applied to metric trees and vp-trees for storing metric data with pseudo-metrics, both Uhlmann and Yianilos preclude such possibilities by restricting the application of metric trees and vp-trees to metric space only [65, 64, 67].

5.4 Maxico Distances

Although Euclidean distances are the natural distances of multidimensional spaces, they are not the most appropriate distances as search radii for multipolar mappings. This section presents the notion of *maxico distances* and shows that maxico distances, when used to define a search radius in a multidimensional space, represent a smaller sphere than is required for Euclidean distances.

Definition 5.4.1 (Maxico Distance) *The* maxico (abbreviation of "maximum coordinate difference") distance function of $\mathbb{R}^k$, $m_k : \mathbb{R}^k \times \mathbb{R}^k \longrightarrow \mathbb{R}^+ \cup \{0\}$, *is defined as*

$$m_k((a_1, \ldots, a_k), (b_1, \ldots, b_k)) = max\{|a_1 - b_1|, \ldots, |a_k - b_k|\}.$$

It can be verified that maxico distances are metrics. A sphere based on the maxico distance of $\mathbb{R}^k$, say $S_{m_k}(x, r)$, corresponds to a k-dimensional *hypercube* with edge length $2r$ and x as its center. Similar to the results of Euclidean distances (see Equation 5.1), multipolar mappings also preserve proximity relationships with regard to maxico distances in multidimensional spaces.

In Appendix B, we prove that given a point x in a pseudo-metric space, and an arbitrary k-polar mapping, the images of all points within distance r from x are within

maxico distance r from the image of x. Mathematically, we have

$$M_{k,P}[S_d(x,r)] \subseteq S_{m_k}(M_{k,P}(x),r). \quad \forall M_{k,P} \in \mathcal{M}(X). \forall x \in X. \forall r \in \mathbb{R}^-. \tag{5.2}$$

where (X,d) a pseudo-metric space.

Comparing Equation 5.1 with 5.2, we can see that the same sphere $S_d(x,r)$ is mapped into both $S_{E_k}(M_{k,P}(x), \sqrt{k}\ r)$ and $S_{m_k}(M_{k,P}(x),r)$ in $\mathbb{R}^k$. It is proved in Appendix B that

$$S_{m_k}(x,r) \subseteq S_{E_k}(x, \sqrt{k}\,r). \quad \forall k \in \mathbb{N}. x \in \mathbb{R}^k. r \in \mathbb{R}^+. \tag{5.3}$$

Figure 5.4 illustrates this relation in $\mathbb{R}^2$. $S_{m_2}(x,r)$ is represented by the light shaded circular area, and $S_{E_2}(x, \sqrt{k}\,r)$ is represented by the dark shaded rectangular area.

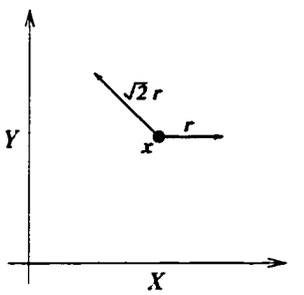


Figure 5.4: Comparison between Spheres Based on m_2 and E_2

From Equations 5.1, 5.2, and 5.3, it is clear that, compared to Euclidean distances, maxico distances reduce the space that must be searched in multidimensional spaces. Thus, maxico distances can potentially decrease the number of points to be examined for proximity searches. For instance, based on the metric data set of Section 3, we have

x	r	$\lvert S_d(x,r)\rvert$	$\lvert S_{E_2}\rvert_{M_2[X]}(x, \sqrt{2}\,r)\rvert$	$\lvert S_{m_2}\rvert_{M_2[X]}(x,r)\rvert$
c_1	5	2	5	5
c_2	10	4	10	8

Note that the maxico distance eliminates two points from $S_{E_2}|_{M_2[X]}(c_2, 10\sqrt{2})$ while it has no improvement over $S_{E_2}|_{M_2[X]}(c_1, 5\sqrt{2})$.

A final note on maxico distance is that if the search target happens to be one of the poles, no distance evaluation is required to carry out a proximity search, since all points

inside a restricted maxico sphere of radius r are within distance r to the search target in the metric data set. Formally, this is expressed as

$$M_{k,P}[S_d(x,r)] = S_{m_k}|_{M[X]}(M_{k,P}(x),r). \quad \forall M_{k,P} \in \mathcal{M}(X), \forall x \in P, \forall r \in \mathbb{R}^+. \tag{5.4}$$

5.5 Dimensionality Issue

We now examine how the efficiency of multipolar mappings are influenced by dimensionality. Intuitively, we consider a multipolar mapping M_b more *efficient* than M_a if and only if given an arbitrary proximity search, M_b has to examine only a subset of the points examined by M_a. Formally, *efficiency* is defined as follows:

Definition 5.5.1 (Efficiency) *Let (X,d) be a pseudo-metric space, and M_a, M_b are a,b-polar mappings respectively. M_b is more* efficient *than M_a, denoted by $M_a \leq M_b$, if and only if*

$$M_b(y) \in S_{m_b}(M_b(x),r) \Longrightarrow M_a(y) \in S_{m_a}(M_a(x),r). \quad \forall x,y \in X, \forall r \in \mathbb{R}^+.$$

The above definition has to be read in light of Equation 5.2. The efficiency relation defined in this way is a *partial order*.

Definition 5.5.1 is defined to compare two multipolar mappings. In the same spirit, we can say maxico distances are more "efficient" than Euclidean distances, since given a k-polar mapping and a proximity search, the set of points that must be examined for m_k is a subset of that required by E_k.

In addition to the efficiency advantage over Euclidean distances, maxico distances guarantee that efficiency increases with the dimensionality of multipolar mappings. Using the notion of maxico distances, we prove that given two sets of poles P_1 and P_2, if $P_1 \subseteq P_2$, then the multipolar mapping based on P_2 is more efficient than the one based on P_1. In other words, if the images of x and y under M_{k_2,P_2} are within r of each other, then the images of x and y under M_{k_1,P_1} are also within r of each other. Formally, this is expressed by

$$M_{k_2,P_2}(y) \in S_{m_{k_2}}(M_{k_2,P_2}(x),r) \Longrightarrow M_{k_1,P_1}(y) \in S_{m_{k_1}}(M_{k_1,P_1}(x),r). \quad \forall x,y \in X, \forall r \in \mathbb{R}^+. \tag{5.5}$$

where (X,d) is a pseudo-metric space, $P_1, P_2 \subseteq X$, $k_1 = |P_1|$, $k_2 = |P_2|$, and $P_1 \subseteq P_2$. See Appendix B for a proof.

Note that a similar conclusion like Equation 5.5 can not be drawn for Euclidean distances. In other words, for $P_1 \subseteq P_2$,

$$M_{k_2,P_2}(y) \in S_{E_{k_2}}(M_{k_2,P_2}(x), \sqrt{k_2}\, r) \not\Longrightarrow M_{k_1,P_1}(y) \in S_{E_{k_1}}(M_{k_1,P_1}(x), \sqrt{k_1}\, r).$$

See Appendix B for details.

From Equation 5.5, the efficiency of a particular multipolar mapping can be improved by adding new poles (while keeping the original ones) to increase its dimensionality. This is demonstrated by an experiment performed on the metric data set shown in Figure 5.2. The data set consists of 100 points with a *diameter*, the largest distance value on the set, of 127.83 and a mean distance value of 50.39. Its distance distribution histogram is shown in Figure 5.5.

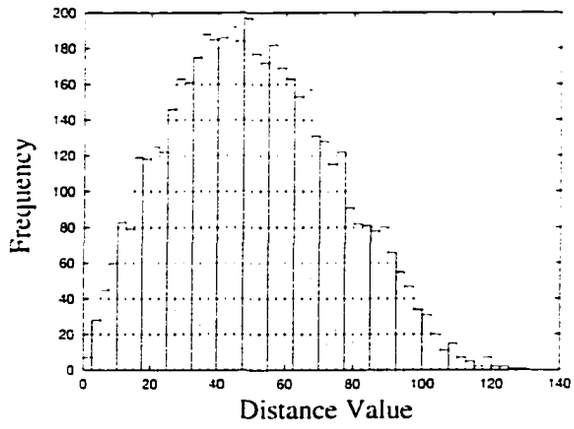


Figure 5.5: Distance Distribution Histogram

We generated 10 sequences of poles, $P_1, \ldots, P_{10}$, of 10 poles each such that every point in the data set was randomly placed into one pole sequence. For each P_i, we assign a random order to its elements and derive a sequence of 10 k-polar mappings, $k = 1, \ldots, 10$. A reasonable assumption is that most proximity searches have search radii much less than the mean distance. Based on this assumption, for each k-polar mapping in the 10 P_i's, we compute the average size of all the 100 restricted spheres of search radius 5 in the k-dimensional space. The results are listed in Table 5.1.

To put the numbers in Table 5.1 into perspective, we compute the average number of points inside a sphere of radius 5 in the original metric data as 1.70. This is the lower bound of the values in Table 5.1. Clearly, the contribution of additional poles is much

Pole Set	Number of Points Visited (Columnized by Dimensionality)									
	1	2	3	4	5	6	7	8	9	10
P_1	15.40	11.40	3.20	2.54	2.54	2.38	1.98	1.90	1.80	1.80
P_2	12.14	2.76	2.14	1.98	1.82	1.80	1.72	1.72	1.72	1.72
P_3	17.96	4.72	2.20	1.88	1.80	1.80	1.74	1.72	1.70	1.70
P_4	13.44	3.36	2.00	1.80	1.78	1.74	1.74	1.72	1.72	1.72
P_5	10.84	6.84	2.62	2.60	1.90	1.84	1.78	1.76	1.74	1.74
P_6	11.38	2.30	1.82	1.82	1.78	1.78	1.78	1.74	1.74	1.74
P_7	10.82	2.36	2.14	1.84	1.84	1.84	1.78	1.74	1.72	1.72
P_8	14.70	5.12	3.12	2.18	1.86	1.80	1.78	1.78	1.74	1.74
P_9	13.36	3.10	1.88	1.80	1.80	1.80	1.80	1.76	1.76	1.76
P_{10}	17.00	3.84	2.44	2.24	1.86	1.86	1.82	1.82	1.78	1.78

Table 5.1: Number of Points Visited versus Dimensionality

more significant at lower dimensionalities. The biggest improvement occurs when a 1-polar mapping is extended to a 2-polar mapping by adding one pole. For the majority of the 10 pole sequences, the average number of points needed to be searched quickly decreases to below 2.00 after only 4 poles are used. In particular, the pole sequence P_3 achieves perfect score 1.70 after 9 poles are used.[6] This perfect score indicates a situation where there is no misclassification for any proximity search at the fixed radius. All points mapped into the corresponding sphere in the k-dimensional space are exactly those inside the sphere of the same radius in the original metric data (see Equation 5.2). Formally, this is expressed as

$$M_k[S_d(x, r)] = S_{m_k}|_{M[X]}(M_k(x), r), \quad \forall x \in X.$$

In fact, for an arbitrary finite metric data set and an arbitrary proximity query, the condition of no misclassifications can always be satisfied after a certain dimensionality threshold [38].

5.6 Average Interpole Distance

While the efficiency measure of Definition 5.5.1 is useful for developing theoretical results such as Equation 5.5, it is too restrictive for practical purposes. For general performance evaluation, we adopt the following less precise definition of efficiency.

[6]Note that the value 1.70 is the average size of all 100 spheres of radius 5 in the data set. Since two decimals are used, there is no round-off error.

M_b is more *efficient* than M_a, if for an average proximity search, M_b examines fewer points than M_a does.

With this definition in place, we are ready to study one major factor influencing the efficiency of multipolar mappings of fixed dimensionalities - the *average interpole distance.*

Definition 5.6.1 (Average Interpole Distance) *Given a set of k poles ($k \geq 2$), $P = \{p_1, \ldots, p_k\}$, the* average interpole distance of P, *$ipd(P)$, is defined as*

$$ipd(P) = \binom{k}{2}^{-1} \sum_{p_i, p_j \in P,\, i<j} d(p_i, p_j).$$

Consider the extreme case of $M_{2,P}$, where $P = \{p_1, p_2\}$ and $d(p_1, p_2) = 0$. From the triangle inequality, $d(p_1, x) = d(p_2, x), \forall x \in X$. Clearly, $M_{2,P}$ is functionally identical to $M_{1,\{p_1\}}$ and $M_{1,\{p_2\}}$. The efficiency advantage of dimensionality cannot be realized. In general, once the dimensionality is fixed, the larger the average interpole distance, the better the efficiency. To illustrate this point, we present the results of a limited efficiency study based on all possible 2-polar mappings of the same metric data set shown in Figure 5.2.

Recall that the data set consists of 100 points with a diameter of 127.83 and a mean distance value of 50.39. Its distance distribution histogram is shown in Figure 5.5. Given a particular 2-polar mapping, the size of the restricted sphere of radius 5 centered at each point is measured, and the mean restricted sphere size is computed. For each 2-polar mapping, its average interpole distance and mean restricted sphere size are used to plot a point in Figure 5.6a. There are 4,950 possible 2-polar mappings, and thus, 4,950 dots.[7] Notationally, the points in Figure 5.6a are represented by,

$$\left(\text{ipd}(P),\ E\left[\left| S_{m_2|_{M_{2,P}}[X]}(x, 5) \right| \right] \right),\quad \forall M_{2,P} \in \mathcal{M}_2(X),$$

where given a random variable V defined on X, $E[V]$ represents its expected value. The h value indicated in the figure represents $E[|S_d(x, 5)|]$, the average number of points inside a sphere of radius 5 in X. Figures 5.6b and 5.6c are the results of similar analyses at search radii 10 and 15 respectively.

At the lower end of the average interpole distance scale, the efficiency of all three search radii (in terms of number of points expected to visit) increases significantly as the

[7]Theoretically, different 2-polar mappings with the same average interpole distance can have the same mean restricted sphere size. Thus, some dots may be superimposed on each other.

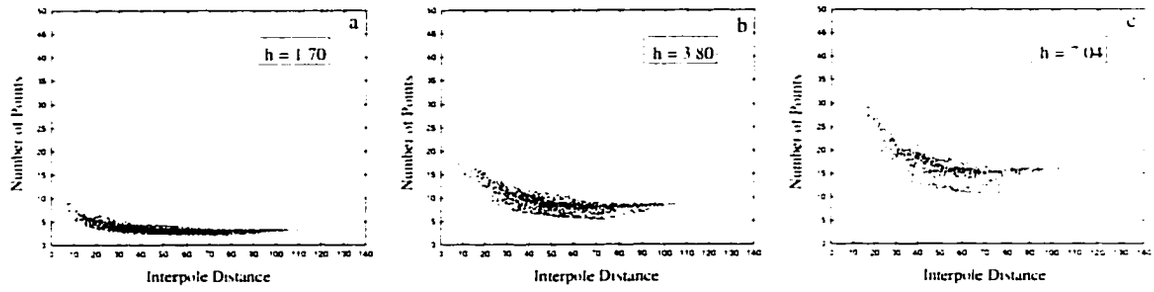


Figure 5.6: Number of Points Examined at Radius 5. 10. and 15

average interpole distance increases. The efficiency begins to level out around the mean distance. Note that the best efficiency is not achieved at the higher end of the average interpole distance scale. but rather in the middle section right after the mean distance. This phenomenon seems to be particularly pronounced for proximity searches with larger search radii (see Figure 5.6). It is also clear that the larger the search radius. the bigger the variation in efficiency for a given average interpole distance value. Details of performance analysis for pole selection based on average interpole distances are presented in Sections 6.8 and 6.10.

5.7 Coordinate Volume Reduction

It can be noticed in Figure 5.3 that there are no points mapped into the triangle area marked by the origin. p_1'. and p_2'. In fact. $M_2[X]$ is bounded by the following three lines in $\mathbb{R}^2$.

$$\begin{aligned} x + y &= d(p_1, p_2). \\ x - y &= d(p_1, p_2). \\ -x + y &= d(p_1, p_2). \end{aligned}$$

where x, y correspond to axes A_1, A_2 respectively (see Figure 5.7). This is not a happenstance. since for an arbitrary point $x \in X$.

$$\begin{aligned} d(p_1, x) + d(p_2, x) &\geq d(p_1, p_2). \\ d(p_1, x) - d(p_2, x) &\leq d(p_1, p_2). \\ -d(p_1, x) + d(p_2, x) &\leq d(p_1, p_2). \end{aligned}$$

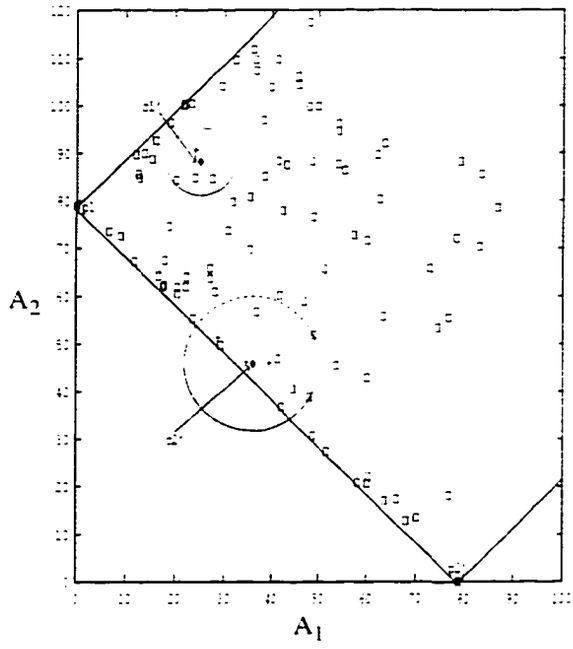


Figure 5.7: Areas Void of Points

Similar results can be derived for $M_k[X]$ where $k > 2$. For instance, $M_3[X]$ is bounded by 9 planes in the following table. For convenience, let $d_{ij} = d(p_i, p_j)$.

$$\begin{array}{lll} x+y=d_{12} & y+z=d_{23} & z+x=d_{13} \\ x-y=d_{12} & y-z=d_{23} & z-x=d_{13} \\ -x+y=d_{12} & -y+z=d_{23} & -z+x=d_{13} \end{array}$$

The planes of the first row are illustrated in Figure 5.8, where x, y, z correspond to axes A_1, A_2, A_3 respectively. The other planes of each column are perpendicular to the one in the first row. In general, $M_k[X], \forall k \geq 2$, is bounded by $3\binom{k}{2}$ hyperplanes.

The *coordinate volume* of $M_2[X]$ - the volume of the minimum rectangular area with all edges parallel to the axes required to contain $M_2[X]$ - can usually be reduced by rotating both the A_1 and A_2 axes by $\frac{\pi}{4}$, resulting in a more balanced representation when $M_2[X]$ is stored in a hierarchical data structure. A *compact 2-polar mapping* is a composition of a $\frac{\pi}{4}$ rotation and a 2-polar mapping.

Definition 5.7.1 (Compact 2-Polar Mapping) *A* compact 2-polar mapping *is defined as*

$$M^c_{2,p} = \begin{bmatrix} \cos\frac{\pi}{4} & \sin\frac{\pi}{4} \\ -\sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{bmatrix} M_{2,p}$$

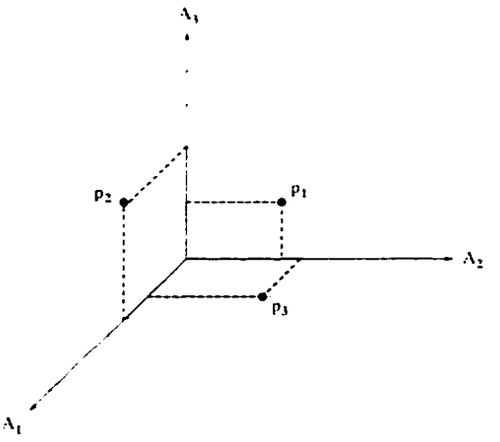


Figure 5.8: Bounding Planes for $M_3[X]$

where $M_{2,p}$ is an arbitrary 2-polar mapping.

Note that rotation is an *orthogonal transformation* which is *distance-preserving*, a stronger condition than proximity-preserving. Equation 5.1 is still true for compact 2-polar mappings. However, the reduction of the coordinate volume occupied by $M_k[X], k > 2$, through similar rotation or rotations is much harder to achieve. For multipolar mappings of higher dimensions, a non-trivial reduction in coordinate volume by orthogonal transformations is probably only possible through exhaustive trial-and-error.

Note that the distance-preserving quality of orthogonal transformations such as rotations or *translations* only applies to Euclidean distances in general. While translations also preserve maxico distances, rotations do not.

5.8 Provisions for Pseudo-Quasimetrics

While most commonly encountered distance functions are either metrics or pseudo-metrics, there exist metric data where symmetry, i.e., axiom M3, cannot be guaranteed. Multipolar mappings are not able to preserve proximity for pseudo-quasimetrics in those occasions. This section presents a method for introducing symmetry in pseudo-quasimetrics. This approach allows indirect application of multipolar mappings to many "well-behaved" pseudo-quasimetrics at the expense of reduced efficiency.

Definition 5.8.1 (Median Transformation) *Let d be a distance function on X. The* median *of d is defined as*

$$d^{+}(x,y) = \frac{1}{2}\left(d(x,y) + d(y,x)\right). \quad \forall x,y \in X.$$

Note that d^{+} is also a distance function on X.

It is easy to prove that if d is a pseudo-quasimetric, its median d^{+} is a pseudo-metric.

Let d be a distance function on X. For finite X, we define

$$\hat{\alpha} = \max\{\ |d(x,y) - d(y,x)|\ \mid\ \forall x,y \in X\}.$$

Note that $\hat{\alpha}$ might not be well-defined for an infinite space.[8] In Appendix B, given a finite pseudo-quasimetric space (X,d), we prove that if the distance between a pair of points is within r in (X,d), the distance between the same pair of points is within $r + \frac{\hat{\alpha}}{2}$ in (X,d^{+}). Formally, this is expressed by

$$S_d(x,r) \subseteq S_{d^{+}}(x, r + \frac{\hat{\alpha}}{2}). \quad \forall x \in X, \forall r \in \mathbb{R}^{+}. \tag{5.6}$$

The result enables us to transform a pseudo-quasimetric to a pseudo-metric with proximity preserved.

From Equations 5.1 and 5.6, the following result can be derived.

$$M_{k.P}[S_d(x,r)] \subseteq S_{E_k}(M_{k.P}(x), \sqrt{k}(r + \frac{\hat{\alpha}}{2})). \quad \forall M_{k.P} \in \mathcal{M}_{d^{+}}(X), \forall x \in X, \forall r \in \mathbb{R}^{+}. \tag{5.7}$$

where $\mathcal{M}_{d^{+}}(X)$ represents the set of all multipolar mappings on (X,d^{+}). From Equations 5.2 and 5.6, a result parallel to Equation 5.7 can be derived for maxico distances:

$$M_{k.P}[S_d(x,r)] \subseteq S_{m_k}(M_{k.P}(x), r + \frac{\hat{\alpha}}{2}). \quad \forall M_{k.P} \in \mathcal{M}_{d^{+}}(X), \forall x \in X, \forall r \in \mathbb{R}^{+}. \tag{5.8}$$

For most applications, the distances among data points are not given explicitly, but rather are values of an analytic distance function which computes distance between a pair of points based on the information, i.e., attribute values of data records associated with the two points. From the formulation of the distance function and the domains of attributes participating in distance computation, it is often possible to derive the value of $\hat{\alpha}$ without exhaustive computation of all distance values.

[8] If the value $|d(x,y) - d(y,x)|$, $x,y \in X$ is bounded in an infinite space X, the value of $\hat{\alpha}$ can be defined as $\sup\{\ |d(x,y) - d(y,x)|\ \mid\ \forall x,y \in X\}$.

In reality, $\hat{\alpha}$ has to be relatively small, compared to the mean value of d^{-}, for efficient proximity search based on Equation 5.8. For many metric data sets with pseudo-quasimetrics, efficiency can often be improved by computing "local" $\hat{\alpha}$ values. Let us define $\alpha : x \longrightarrow \mathbb{R}^{+}$ as

$$\alpha(x) = \max\{\ |d(x,y) - d(y,x)|\ \mid\ y \in X\}.\quad \forall x \in X.$$

Equations 5.7 and 5.8 are still true with $\hat{\alpha}$ replaced by $\alpha(x)$. Similar to the derivation of $\hat{\alpha}$, it is often possible to compute $\alpha(x)$ without evaluating distance values between x and all other points. In such cases, for a search target x, its $\alpha(x)$ value can be generated on the fly. The advantage of α over $\hat{\alpha}$ is demonstrated in the following simple example.

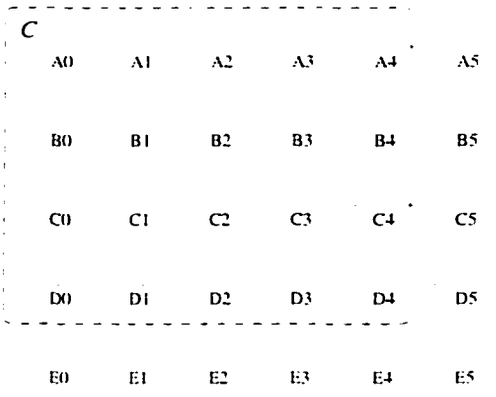


Figure 5.9: Example of Asymmetric Distance

Figure 5.9 illustrates an abstract street map where all streets are two-way except for the two streets specifically marked. Let d be a distance function defined on the set of street corners, $X = \{A0, \ldots, E5\}$. The distance from one street corner x to another one y is measured by the minimum numbers of street segments required to travel from x to y. For instance, $d(A2, D2) = 5$, $d(C5, E5) = 2$, and $d(E5, C5) = 4$. Clearly, d is a pseudo-quasimetric. Let C represent the set of street corners inside the area marked by the dashed rectangle in Figure 5.9. It is easy to verify that

$$\begin{aligned} |\, d(x,y) - d(y,x)\,| &= 0, \quad \text{if } x \in C \text{ or } y \in C. \\ |\, d(x,y) - d(y,x)\,| &\leq 2, \quad \text{otherwise.} \end{aligned}$$

Thus, we have $\dot{\alpha} = 2$, and

$$\alpha(x) = \begin{cases} 0 & \text{if } x \in C. \\ 2 & \text{otherwise.} \end{cases}$$

Compared to $\dot{\alpha}$, α has the capability to model local asymmetry, and the efficiency can be greatly improved for many search targets which are less influenced by asymmetric distance values (i.e., those points in C).

Note that the application of the proposed technique is not limited to data structures incorporating multipolar mappings. In fact, it can be applied to all hierarchical metric data structures based on pseudo-metrics, including metric trees and vp-trees.

5.9 Multipolar Mapping versus Existing Approaches

There are two kinds of information in a metric data set. First, there is information associated with each data point. Second, there is information describing the interrelationships among data points, as encapsulated by the distance function. For a large metric data set, the distance matrix is almost never pre-computed, since the size of the distance matrix can easily outgrow the size required to store the information for the data points.

The evaluation or computation of the distance between a pair of points is a good choice as a basic measure counted toward the complexity or performance analysis of hierarchical metric data structures. In fact, this is exactly the complexity measure used for the limited performance studies in [64, 67]. Since metric trees with sphere decomposition and the two non-bucket varieties of vp-trees share the same computational complexity when the number of distance evaluations is used as the complexity measurement, we use the term *sphere trees* to represent them as a group.

Given a metric data set of size n, to construct a balanced sphere tree takes $n \log n$ distance evaluations. The number of distance evaluations required to perform a proximity search in a sphere tree is the same as the number of internal nodes visited during the search. In a balanced sphere tree, it takes $\log n$ evaluations to locate just one point, i.e., a proximity search with an infinitely small search radius. Since both the metric trees and vp-trees were originally proposed for finding nearest neighbors instead of performing general proximity searches, the performance analyses in [64, 67] are limited to very small search radii. However, even for small search radii, the number of internal nodes visited for locating nearest neighbors is much larger than the theoretical lower bound $\log n$.

To compute a k-polar mapping requires $kn - \binom{k}{2}$ evaluations. There are no more evaluations needed to construct a point spatial data structure based on the mapping. Compared to the $n \log n$ evaluations required for constructing the sphere trees, our approach is certainly much more efficient. As for proximity search, the number of distance evaluations needed is the number of points inside the corresponding sphere in the k-dimensional space (see Equation 5.2). While a more comprehensive performance study of multipolar mappings is still under way, it is clear to us that compared to sphere trees, point spatial data structures based on multipolar mappings require significantly fewer distance evaluations for proximity searches of small search radii.

We realize that when the distance evaluation is not a costly operation, our approach might not outperform metric trees and vp-trees. Because of the fundamental differences in these two approaches, only an empirical comparative study can flush out the difference in performance under this circumstance.

Another potential advantage of the multipolar approach over the sphere trees is that the multipolar approach only requires the ability to compute distances from the points designated as poles to the other points in the data set. Spheres trees require the ability to compute distance between arbitrary points in the data set. This could be a significant issue when it is necessary to carry physical experiments or simulations in order to evaluate a distance function.

5.10 Summary

Multipolar mapping and median transformation enable the application of existing hierarchical metric data structures and point spatial data structures to support the metric-based model at the physical level. Through this approach, metric data sets modeled by pseudo-quasimetrics can be accessed effectively and efficiently through proximity queries.

While comprehensive performance results are not available at this moment, point spatial data structures based on multipolar mappings demonstrate significant theoretical improvements over existing hierarchical metric data structures such as metric trees and vp-trees. There are two additional advantages to our approach. First, existing point spatial data structures can be utilized with little or no modifications. Second, compared to existing hierarchical metric data structures, our approach may require less distance information from the metric data.

Chapter 6

Performance Analysis

6.1 Introduction

In order to evaluate the functionality and performance of multipolar mappings as the foundation for a new class of efficient hierarchical metric data structures proposed in Chapter 5, we present the results of a series of performance experiments.

The objective of these experiments is to explore the interrelationships among major parameters of data sets and multipolar mappings. The results show us how various multipolar mappings perform under different circumstances. The performance index is based on the efficiency of *proximity queries*. In order to have a consistent and complete collection of data sets with varying degrees of quantitative characteristics, synthesized metric data sets are utilized for these experiments.

6.2 Data Synthesis

A metric data set of size n is synthesized by generating n points in a multidimensional space where the Euclidean distance is used to calculate the distance between pairs of points of the data set. The same process is also used to generate the metric data set illustrated in Figure 5.2 of Section 5.3. A metric data set produced by this synthesis process is determined by three factors:

1. the dimensionality of the embedding space,
2. the number of points to be generated, and

3. the placement policy of those points.

To provide a standard framework for comparative study, all synthesized data sets were generated such that the distance values fall into the standard interval $[0, 100]$. This was achieved by restricting each coordinate of a data point in an m-dimensional embedding space to the interval $[0, 100m^{-\frac{1}{2}}]$. In other words, all points are selected from an m-dimensional hypercube of side $100m^{-\frac{1}{2}}$ where the distance between the furthest pairs of corners is exactly 100.

Our point placement policy is not concerned with the individual position of each data point, but rather the overall distance distribution induced by those positions as a group. There are many placement schemes which can affect the distance distribution. For analysis purposes, we focus on the *degree of clusterness* (or just *clusterness*) exhibited by the actual point placement. As is shown in subsequent sections, the degree of clusterness indeed affects overall distance distribution of a data set.

Based on clusterness, a synthesized metric data set is quantitatively characterized by the following 5 parameters:

m	intrinsic dimensionality
n	number of data points
p	percentage of points in clusters
c	number of clusters
r	cluster radius

The value of p is in the interval $[0, 1]$. First, $(1-p)n$ points are randomly and uniformly selected from the embedded hypercube in the m-dimensional space. Second, the remaining pn points are randomly and uniformly selected from the c embedded spheres of radius r. A *uniform data set* is a synthesized data set with $p = 0$. A data set with $p > 0$ is a *clustered data set*.

Note that the notion of intrinsic dimensionality is just an artifact of the data synthesis process. A similar notion might also be applicable to many real world metric data sets. For instance, a census data set consisting of records of 1,000 attributes is sometimes said to have 1,000 dimensions. Although the number of dimensions can have an impact on the behavior of distance functions defined on the data set, a metric data set has no dimensionality from the distance perspective. Another subtle point to keep in mind is that a metric data set synthesized in this way is a pseudo-metric. It is not a *metric* since the synthesis program might generate two or more points in the embedding multidimensional space

with identical coordinates. However, the chance is extremely small and of no meaningful consequence in our performance analysis.

6.3 Distance Distributions and Proximity Accumulations

In order to illustrate the performance analysis results, two kinds of graphs are used extensively in this chapter - *distance distribution* and *proximity accumulation*. Both graphs are *histograms*, although in this thesis they are usually drawn as continuous curves instead of sequences of boxes, which allows multiple histograms to be overlayed in a single figure for easier comparison.

Figure 6.1 is a distance distribution graph for a collection of 10 data sets of size 10,000 and varying intrinsic dimensionalities from 1 to 10. Each curve in Figure 6.1 represents a distance histogram of a specific data set, in which 100 *bins* are employed. The x-axis of the distribution represents distance values which ranges from 0 to 100, the fixed standard interval, and there are $(10^4)^2 = 10^8$ distance values among the 10^4 points of each data set.[1] By dividing the full interval of distance distribution [0, 100] into 100 bins, i.e., subintervals, of equal size and counting the percentage of the 10^8 distance values falling into each interval (represented as the y-axis), we have a histogram and a curve can be plotted to connect the tops of all adjacent boxes.[2] The height of the curve is dependent on the number of bins used for the underlying histogram. This is not a problem, however, since we are only interested in the differences among curves.

Many distance distribution graphs are presented in this chapter. For convenience, we use the shorthand $(m, n, p, c, r) = (1 \ldots 10, 10^4, 0, 0, 0)$ in the caption to indicate the characteristics of the data set or data sets in question (Figure 6.1). The meanings of individual symbols m, n, p, c, and r are described in Section 6.2. The keys listed on the upper right corner of the figure correspond to the data sets described by the (m, n, p, c, r) shorthand. The first key indicates the data set with $m = 1$, the second key for $m = 2$, and so on.

For example, based on Figure 6.1, we can see that for $m = 2$, about 2% of distance

[1] One might argues that there are only $\frac{1}{2}10^4 \times (10^4 - 1)$ "significant" distance values, since our synthesis process produces pseudo-metrics. This is a moot point, however. For our analysis purposes, the reflexivity and symmetry in the synthesized metric data are not taken into account.

[2] To reduce the computation time, only a subset of all 10^8 possible distance values are sampled to derive the distance distribution. In this case, we randomly select 100 points and use all 10^6 distance measures induced by the 100 point metrics as samples.

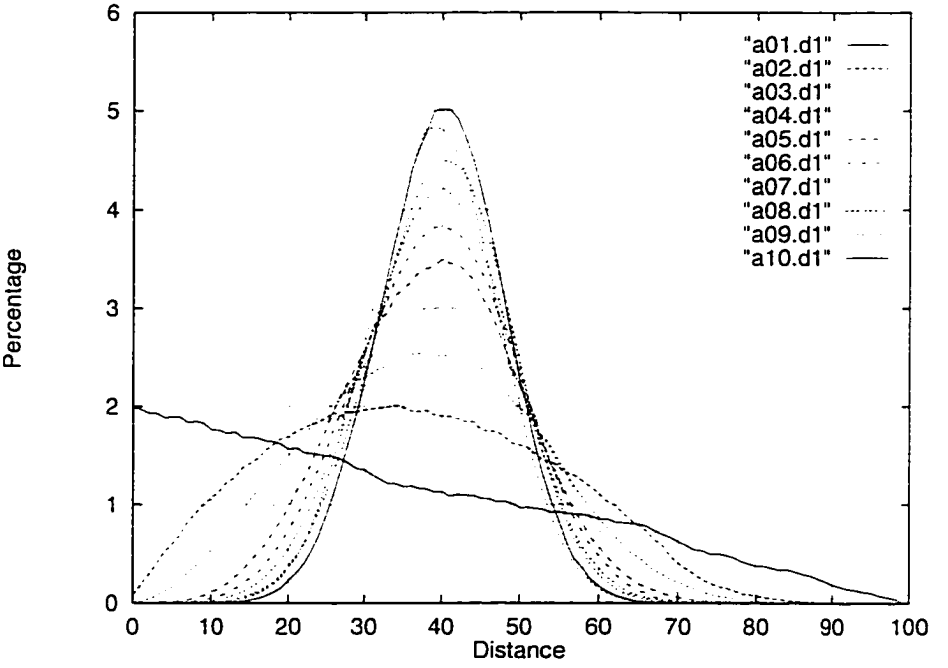


Figure 6.1: $(m, n, p, c, r) = (1 \ldots 10, 10^{4}, 0, 0, 0)$. Distance distribution.

values fall into [30, 31] (recall that this histogram has 100 bins). In other words, for an arbitrary point, we expect that about 2% of the points in the data set are at least 30 and at most 31 away from it. Similarly, we can expect that for $m = 1$, there are about 2% of the points are within 1 unit away from an arbitrary point. As m increases, this percentage drops drastically. For $m \geq 7$, for an arbitrary point, virtually all the other points are at least at a distance of 10 away.

This phenomenon is known as the "curse of dimensionality." That is, the spatial density of data in the embedding space decreases rapidly as the dimensionality of the embedding space increases. Figure 6.1 shows that while all distance values fall into the same standard interval, the higher the intrinsic dimensionality, the more centralized, i.e., less spread, the distance distribution becomes. In other words, as the intrinsic dimensionality grows, for an arbitrary point, the chance of having a close-by neighboring point decreases significantly. This phenomenon has a significant performance impact on all proximity search mechanisms.

Compared to a distance distribution graph, a proximity accumulation graph en-

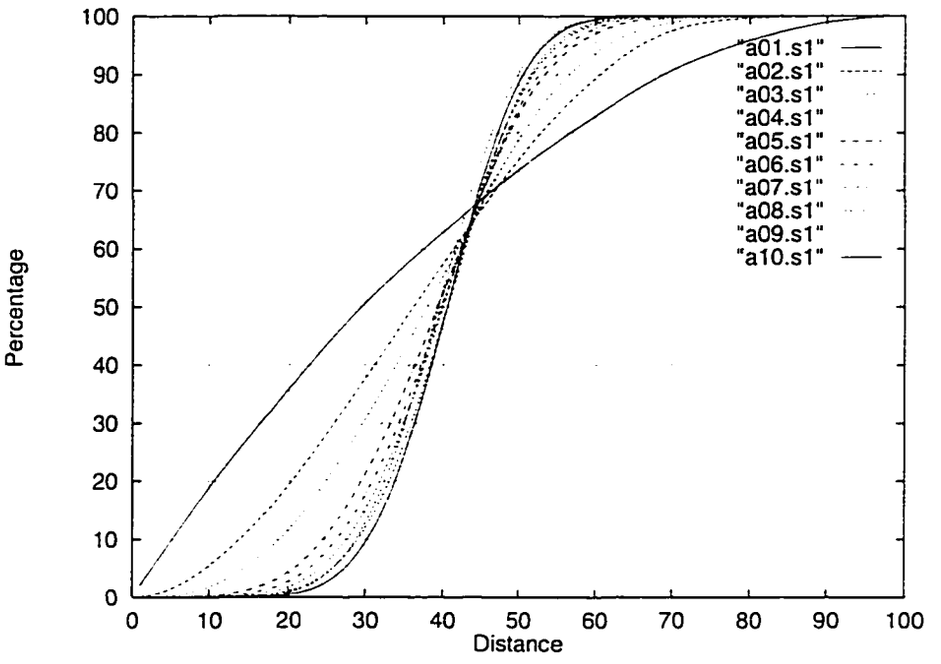


Figure 6.2: $(m.n.p.c.r) = (1\ldots10.10^4.0.0.0)$. Proximity accumulation.

ables us to examine the same data set from a proximity perspective. Figure 6.2 is a proximity accumulation graph showing the proximity structures of the same 10 data sets illustrated in Figure 6.1. Similar to a distance distribution graph, the x-axis of a proximity graph represents the standard distance interval $[0.100]$. The y-axis represents the expected percentage of points falling into a sphere with its radius specified on the x-axis. Unlike distance distribution graphs, proximity graphs are accumulative in nature. Thus, while the height of a distance distribution curve depends on the number of bins, the height of a proximity accumulation curve stays the same no matter how many bins are used. The "curse of dimensionality" can also be clearly observed in Figure 6.2.

The proximity accumulation graph is a useful tool for illustrating the efficiency of proximity searches after the employment of a multipolar mapping. Given a fixed distance as search radius, the proximity graph of the transformed data set gives the expected number of points needed to check inside the search radius. While multipolar mappings preserve proximity for pseudo-metrics, the results of a multipolar mapping has different distance distribution and proximity accumulation compared to the original one. By plotting both

the proximity curves of the original metric data and the transformed data. the percentage of points that map to false positives is clearly illustrated.

6.4 Intrinsic Dimensionalities

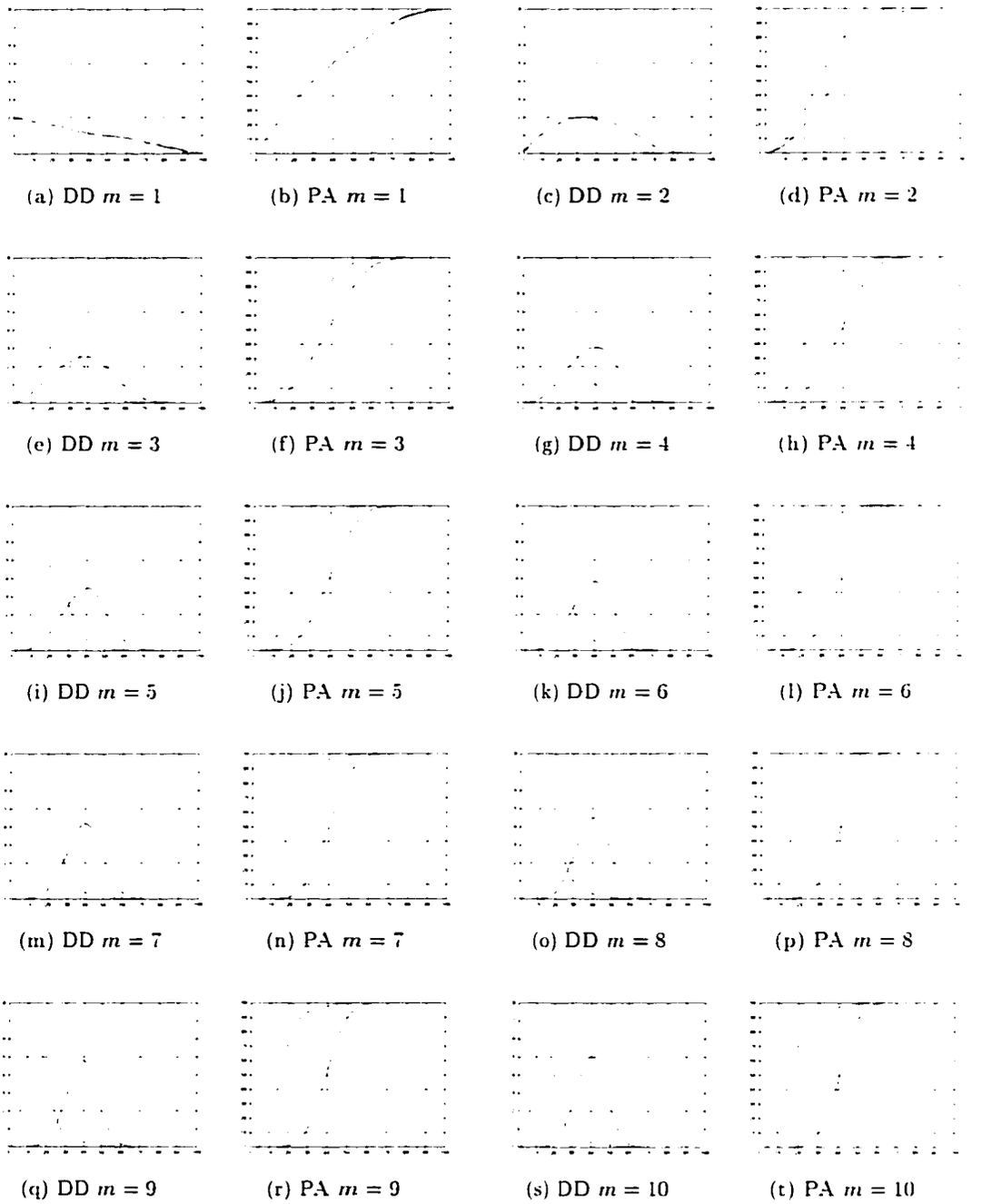
This section presents performance results of multipolar mappings for uniform data sets of the same size but different intrinsic dimensionalities. This provides us with the basic insights and benchmarks of the performance issues from which subsequent results can be put into perspective. All the results presented in this section are based on multipolar mappings with randomly selected poles. The performance impact of pole selection is presented in a later section.

Figures 6.1 and 6.2 show the distance distributions and proximity accumulations of data sets of 10.000 points with intrinsic dimensionalities varying from 1 to 10. The performance of a 5-polar mapping on these data sets is plotted in Figure 6.3. For each data set. a pair of distance distribution and proximity graphs are presented. There are two curves in each graph which correspond to the original data set and the multipolar-transformed data set respectively.

Since the distance distributions and proximity accumulations for the 1-dimensional data set and its transformation under the multipolar mapping are almost identical. there appears to be only one curve in both Figures 6.3(a) and (b). The impact of the "curse of dimensionality" shows clearly in Figure 6.3. Once the intrinsic dimensionality exceeds 3. the distance distribution of the transformed data set shifts away from the one of the original data set. As a result. for small search radii. the percentage of false hits increases dramatically which makes multipolar-based proximity search less efficient.

Theoretically. the "curse of dimensionality" would disappear if the size of the data set increases along with the intrinsic dimensionality. However. to avoid the "curse of dimensionality". the growth rate in size would have to be exponential to match a linear growth rate in dimensionality. In order to demonstrate this point. a similar series of experiments was done on a data set of 100,000 points.

Figure 6.4 shows the distance distributions and proximity accumulations of data sets of 100.000 points with intrinsic dimensionalities varying from 1 to 10. The performance of 5-polar mappings on these data sets is plotted in Figure 6.5. Two observations can be drawn from a simple comparison between Figures 6.3 and 6.5:



(a) DD $m = 1$ (b) PA $m = 1$ (c) DD $m = 2$ (d) PA $m = 2$

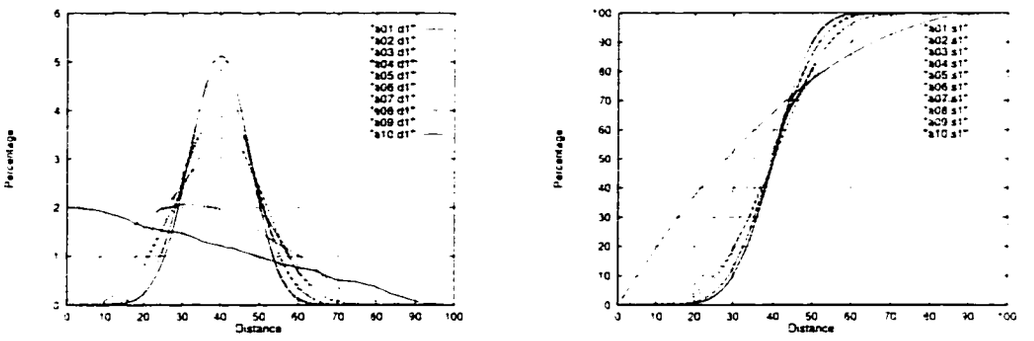
(e) DD $m = 3$ (f) PA $m = 3$ (g) DD $m = 4$ (h) PA $m = 4$

(i) DD $m = 5$ (j) PA $m = 5$ (k) DD $m = 6$ (l) PA $m = 6$

(m) DD $m = 7$ (n) PA $m = 7$ (o) DD $m = 8$ (p) PA $m = 8$

(q) DD $m = 9$ (r) PA $m = 9$ (s) DD $m = 10$ (t) PA $m = 10$

Figure 6.3: $(m.n.p.c.r) = (1 \ldots 10.10^4.0.0.0)$. Distance distributions and proximity accumulations.



(a) Distance Distribution

(b) Proximity Accumulation

Figure 6.4: $(m.n.p.c.r) = (1\ldots10.10^5.0.0.0)$. Distance distribution and proximity accumulation (overlayed).

1. The curse of dimensionality stays unchallenged in these larger data sets.

2. Multipolar mappings have an almost linear time complexity (constant. percentage wise) as the sizes of the data sets are increased 10 fold. This is one major advantage of multipolar mappings which is extensively explored in Section 6.6.

Table 6.1 shows in detail the performance of 5-polar mappings as the size of a 2-dimensional data set increases from 10.000 to 100.000. The numbers in the table indicate the expected number of points inside a sphere centered at a randomly selected point with the specified radii.

6.5 Clustered Data

Data points of a real data set are rarely distributed both randomly and uniformly in the sampling space or geometric space. A uniform distribution often includes explicit regular patterns for point placement such as those data sets which can be modeled by regular grids (Section 2.4). A nonuniform distribution might or might not have explicit point placement patterns. However, the notion of nonuniformity implies *clusterness*. which can be considered as implicit point placement patterns. Note that the term "point placement" does not necessarily mean point coordinates but rather the interrelationships among points. i.e.. the inter-instance relationships.

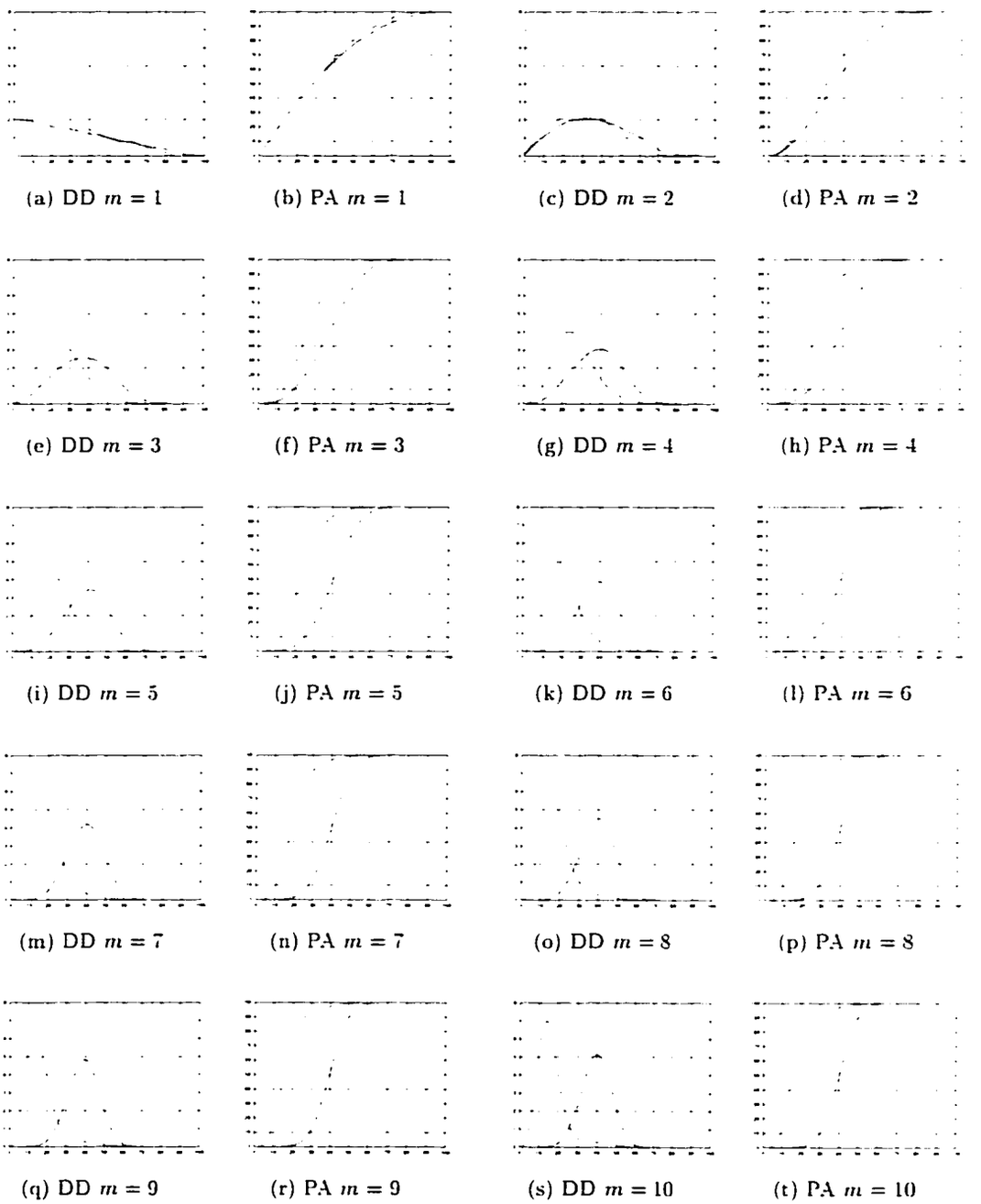


Figure 6.5: $(m.n.p.c.r) = (1 \ldots 10.10^5.0.0.0)$. Distance distributions and proximity accumulations.

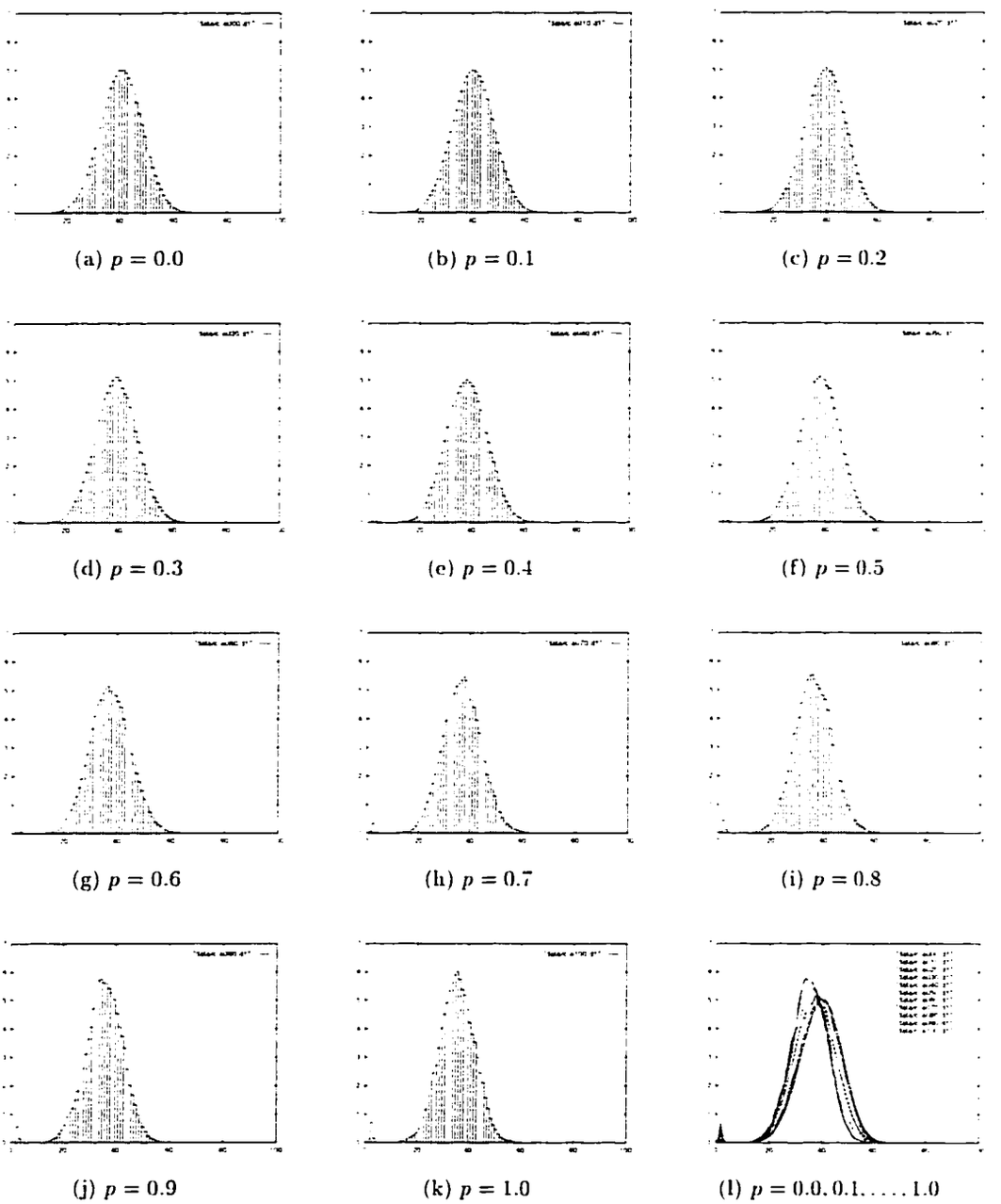
Distance	10^4 Points		10^5 Points	
	Original	5-polar	Original	5-polar
1	7.7000	8.9600	62.6400	86.5700
2	26.4100	30.9700	243.9800	335.8800
3	55.6600	64.9500	541.2000	741.2700
4	95.9700	113.8700	957.1500	1300.5400
5	148.9100	175.2300	1483.4500	2011.5800
6	210.0100	249.9500	2115.8800	2866.2300
7	280.4900	335.0100	2857.8200	3866.4500
8	363.4000	434.9400	3690.0600	4994.7000
9	455.0700	543.6100	4632.2300	6254.7000
10	555.1800	661.6100	5659.6300	7624.6100

Table 6.1: Performance of 5-polar mappings on 2-dimensional data sets of sizes 10^4 and 10^5.

In other words, real data without explicit inter-instance relationships are often clustered, since, from a distance perspective, complex and implicit inter-instance relationships are often manifested as a collection of clusters. Although there are various kinds of clusters, for performance analysis purposes we focus on their most generic form – sphere. As mentioned in Section 6.2, two parameters, c and r, are used to specify the number and radius of clusters, in the form of spheres, embedded in the synthesized metric data set. Parameter p is used to specify, indirectly, the number of points in clusters.

A series of experiments on data sets with $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$, where $p = 0.0, 1, \ldots, 1.0$, was performed. Figures 6.6(a)-(k) illustrate their respective distance distribution histogram with 100 bins. Recall that each distance distribution represents the average of 100 point metrics at randomly selected points. In Figure 6.6(l), the curves of all these histograms are overlayed for easy comparison. Note that as p increases, the size of the bump near the origin also grows. In order to better illustrate the impact of clusterness, Figure 6.7 shows the distance distribution histograms in the interval $[0, 20]$ for the same data sets with 10,000 bins.

Figure 6.8 shows the distance distribution histogram on the interval $[0, 20]$ for each of the data sets after a 5-polar mapping is applied. Similar to Figure 6.7, 10,000 bins are used. Note that all subfigures in Figure 6.7 and Figure 6.8 have the same scale. Based on Equation 5.2 in Section 5.4, the area of the dark region in any interval $[0, t], t \leq 20$, of a subfigure in Figure 6.7 would have to be less than the area of the dark region in the same



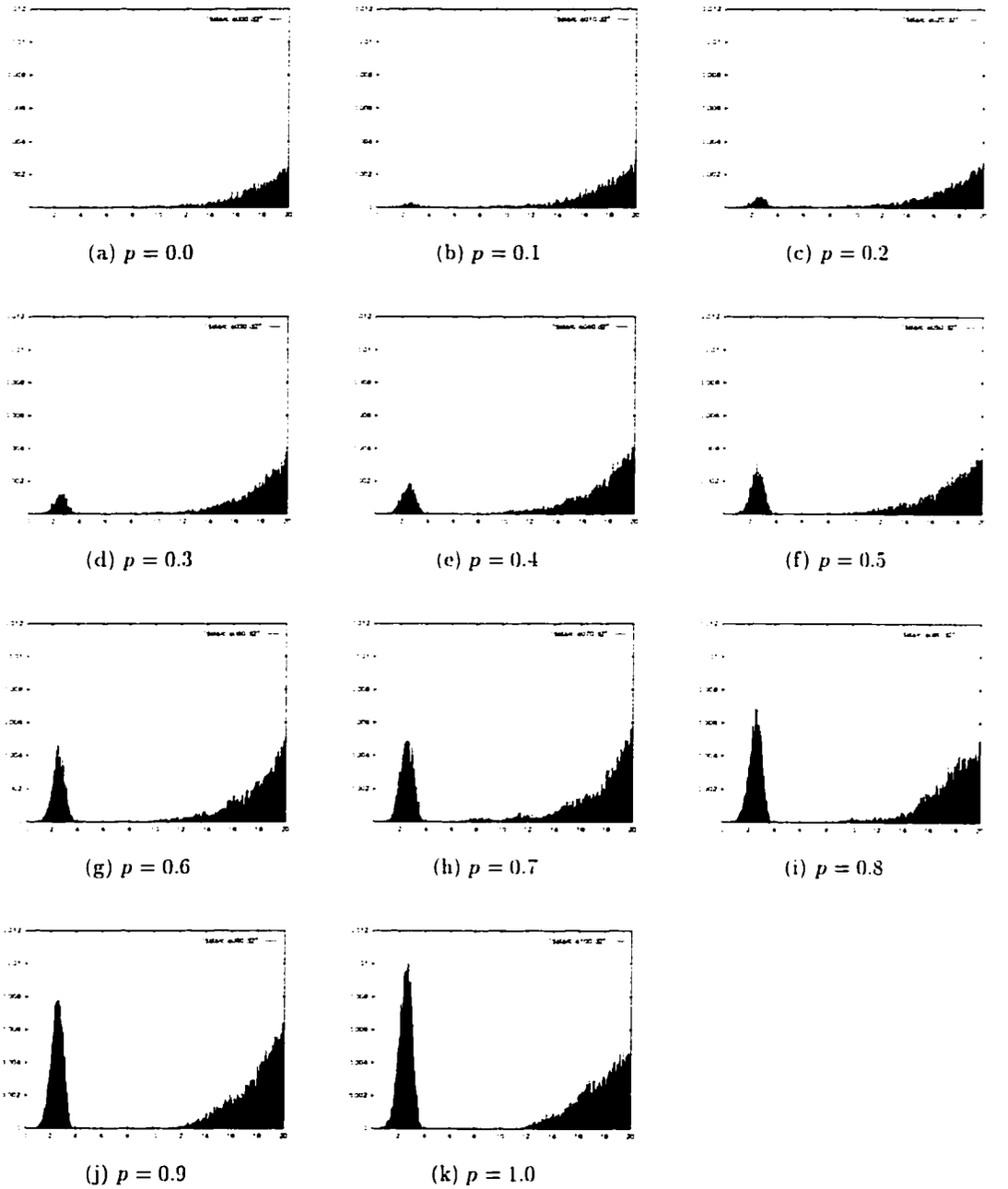
(a) $p = 0.0$ (b) $p = 0.1$ (c) $p = 0.2$

(d) $p = 0.3$ (e) $p = 0.4$ (f) $p = 0.5$

(g) $p = 0.6$ (h) $p = 0.7$ (i) $p = 0.8$

(j) $p = 0.9$ (k) $p = 1.0$ (l) $p = 0.0, 0.1, \ldots, 1.0$

Figure 6.6: $(m, n, p, c, r) = (10, 10^4, 0.0 \ldots 1.0, 100, 2)$. Distance distribution histograms.



(a) $p = 0.0$ (b) $p = 0.1$ (c) $p = 0.2$

(d) $p = 0.3$ (e) $p = 0.4$ (f) $p = 0.5$

(g) $p = 0.6$ (h) $p = 0.7$ (i) $p = 0.8$

(j) $p = 0.9$ (k) $p = 1.0$

Figure 6.7: $(m, n, p, c, r) = (10, 10^4, 0.0 \ldots 1.0, 100, 2)$. Partial distance distribution histograms.

interval of the corresponding subfigure in Figure 6.8.

For instance, let us compare Figure 6.7(k) with Figure 6.8(k). All points constituting the small peak in [0.4] of Figure 6.7(k) are mapped by the 5-polar mapping into a small peak and its adjacent area in [0.4] of Figure 6.8(k) (Equation 5.2, Section 5.4 and Theorem B.3, Appendix B).

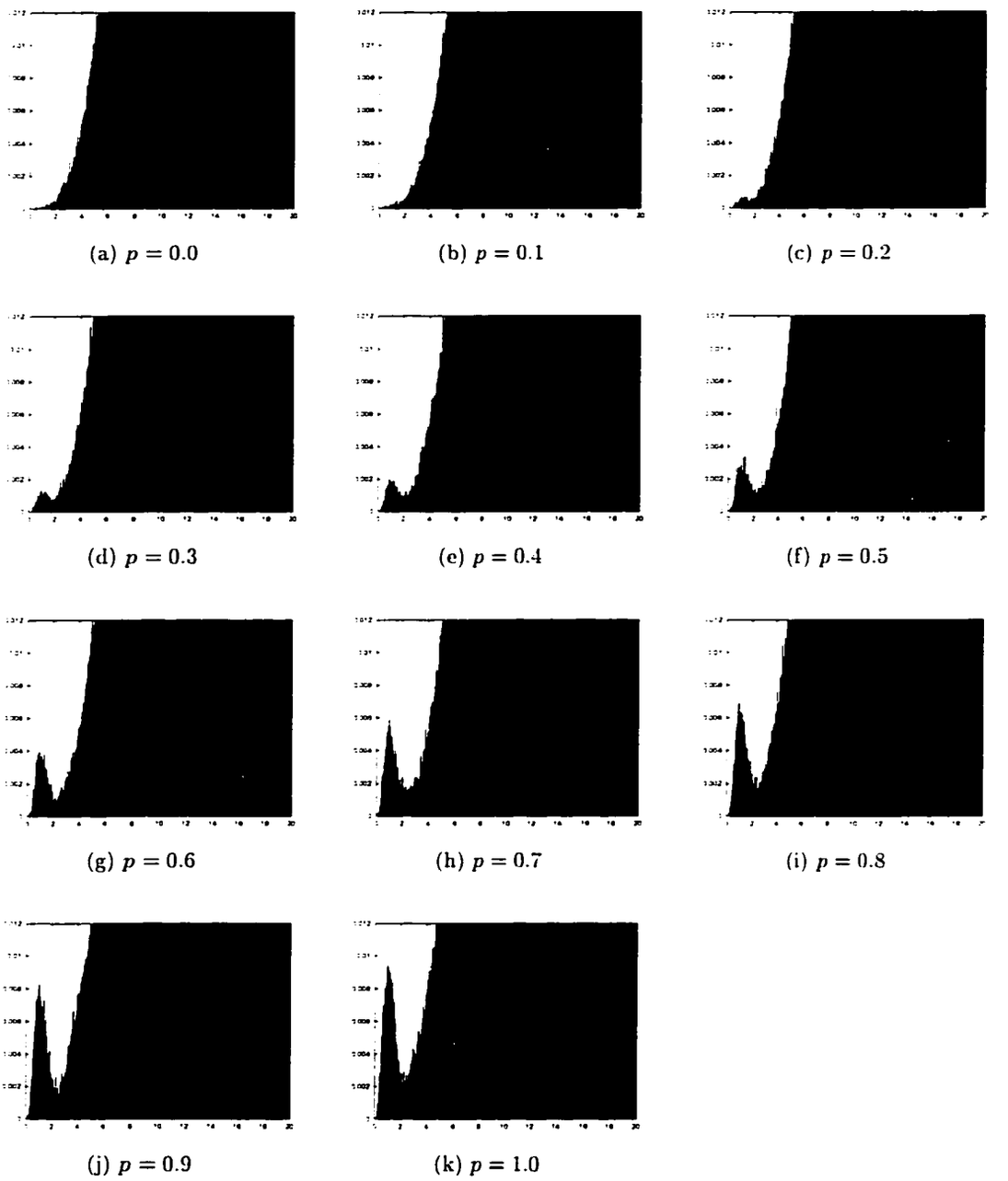
Now let us examine the case of a uniform data set (Figure 6.7(a)). For an arbitrary point, a proximity query of radius 8 or less is extremely unlikely to find its nearest neighbor. However, lots of points would be mapped into a sphere of radius 8 by a 5-polar mapping (Figure 6.8(a)). Locating the nearest neighbor for an arbitrary point is most likely to be a very inefficient process under such circumstances.

On the other hand, we expect that a proximity query of radius 2 would suffice to find the nearest neighbor for an arbitrary point in the data set of $p = 0.5$ (Figure 6.7(f)).[3] Only a very small percentage of points would have to be checked in this case (Figure 6.8(f)).

Figure 6.9 illustrates the performance of nearest neighbor queries. The x-axis represents p, the degree of clusterness, while the y-axis represents the percentage of points to be checked to carry out the queries. It is obvious from the figure that the performance improves significantly as p increases from 0 to 0.2. However, as p increases beyond 0.2, the density of each cluster reaches a level such that there exist many near but non-nearest neighbors to an in-cluster search target. A significant portion of these points are mapped to images which are even closer to the image of the search target than the image of the nearest neighbor. As a result, the performance degrades very slightly and very gradually as p increases from 0.2 to 1.

Figure 6.10 illustrates the performance of proximity queries at various search radii. The notation $n\%$ indicated in each subfigure represents a search radius at $n\%$ of the average distance of the metric data set. Similar to Figure 6.10, the x-axis represents p, the degree of clusterness. PC indicates a graph for percentage of points checked for proximity queries. In a PC graph, the lower the percentage, the better the performance. FP indicates a graph for percentage of false positives. An FP graph has to be interpreted in light of its corresponding PC graph, however. Specifically, given a data set and a multipolar mapping, its PC and

[3] In fact, the appropriate search radius for nearest neighbor queries on a particular data set can be determined by summing the individual bin percentages starting from the leftmost bin, i.e., the one closest to the origin. Since there are 10,000 points in the data set, once the accumulated percentage reaches 0.01%, the search radius corresponding to the last bin in the summation is the expected radius for locating the nearest neighbor.



(a) $p = 0.0$ (b) $p = 0.1$ (c) $p = 0.2$

(d) $p = 0.3$ (e) $p = 0.4$ (f) $p = 0.5$

(g) $p = 0.6$ (h) $p = 0.7$ (i) $p = 0.8$

(j) $p = 0.9$ (k) $p = 1.0$

Figure 6.8: $(m, n, p, c, r) = (10, 10^4, 0.0 \ldots 1.0, 100, 2)$. Partial distance distribution histograms after 5-polar mappings.

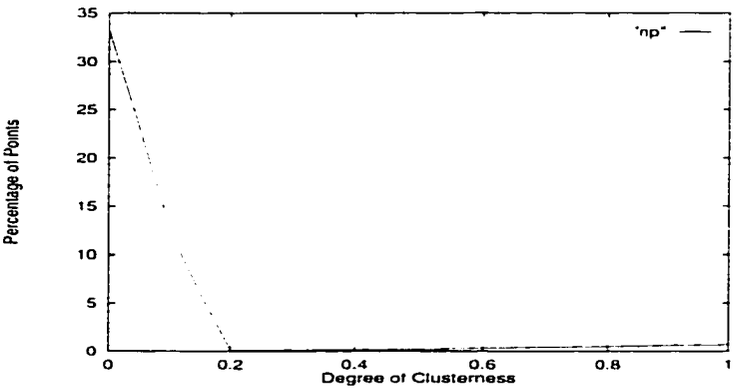


Figure 6.9: $(m.n.p.c.r) = (10.10^4.0.0 \ldots 1.0.100.2)$. Performance of nearest neighbor queries.

FP graphs are related by:

$$P_F(r) = \frac{P_M(r) - P_T(r)}{P_T(r)}.$$

where $P_T(r)$ represents the percentage of points expected in a sphere of radius r in the original data set - the percentage of true positives: $P_M(r)$ represents the percentage of points expected in a sphere of radius r in the multipolar-transformed data set - the y index of a PC graph. $P_F(r)$ is the expected percentage of false positives at search radius r the y index of an FP graph.

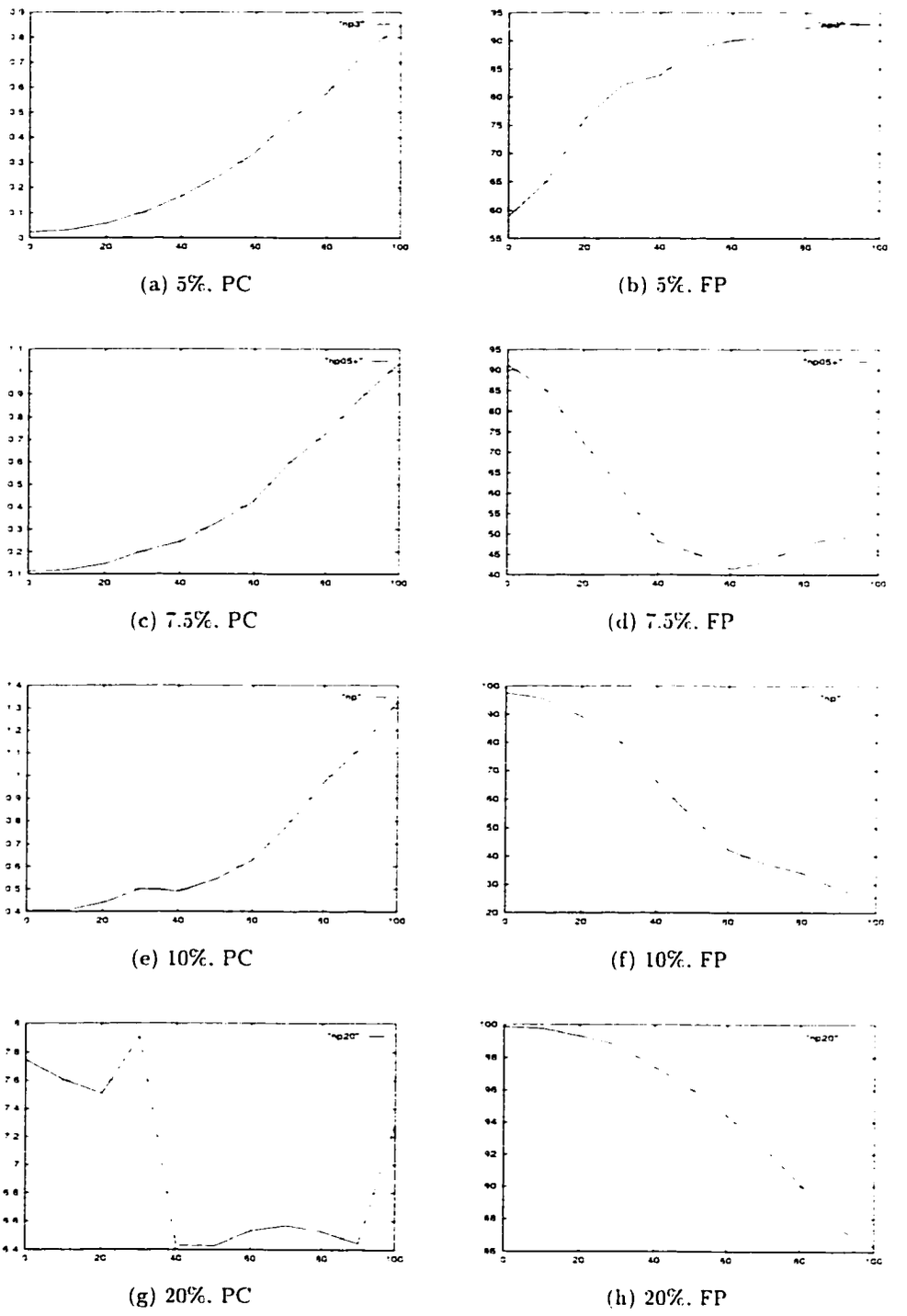
Some remarks about the PC graphs in Figure 6.10 are in order.

1. From Figure 6.10(a), (c) and (e), we can see that the PC performance decreases as p increases, since the larger the p value, the more points are expected to be mapped into a sphere of a modest radius. This effect can also be easily observed in Figure 6.8.

2. The performance of 10% average distance proximity query at $p = 0.3$ seems to be slightly out of order (Figure 6.10(e)) - a phenomenon caused by a poor set of randomly selected poles (Section 6.8).

3. For 20% average distance proximity queries, the PC performance is rather erratic (Figure 6.10(g)), since such a large search radius (compared to the size of clusters) effectively masks the contribution of different p values. Thus, the quality of poles dictates the overall performance in this situation.

4. Consistent with Figure 6.10(e), Figure 6.10(g) also suggests that the set of poles used for $p = 0.3$ are indeed poor selections.



(a) 5%. PC (b) 5%. FP

(c) 7.5%. PC (d) 7.5%. FP

(e) 10%. PC (f) 10%. FP

(g) 20%. PC (h) 20%. FP

Figure 6.10: $(m, n, p, c, r) = (10, 10^4, 0.0 \ldots 1.0, 100, 2)$. Performance of proximity queries at various search radii.

With respect to the FP graphs, the average distances for this series of data sets range from 37 to 39 approximately (Figure 6.6). Hence we can make the following observations.

1. 5% of the average distance is around 2 which is not large enough to include the entire range of the small peaks corresponding to the clusters (Figure 6.7). Thus $P_T(2)$ is very small compared to $P_M(2)$. Since the growth rate of $P_M(2)$ is faster than $P_T(2)$, the percentage of false positives increases as p increases (Figure 6.10(b)).

2. 7.5% of the average distance is around 3. A significant portion of the small peak of clusters is included in $P_T(3)$. In other words, there is significant growth in $P_T(3)$ as p increases. At the same time, the growth rate of $P_M(3)$ is rather modest and the difference between $P_M(3)$ and $P_T(3)$ gets smaller. As a result, $P_F(3)$ decreases as p increases for 7.5% (Figure 6.10(d)).

3. The same reasoning of the 7.5% FP graph can also be applied to the FP graphs for 10% and 20% (Figures 6.10(f) and (h)).

For a 10-dimensional data set, in order to counter the effect of the curse of dimensionality and get a relatively stable performance measure of both nearest neighbor and proximity queries, it seems that 20% clusterness, i.e., $p = 0.20$, is necessary (Figures 6.9 and 6.10). Also, it might be unrealistic to expect real world metric data to have 100% clusterness, i.e., $p = 1.00$. In the subsequent experiments, we focus on data sets with $p = 0.25, 0.50, 0.75$.

6.6 Data Set Sizes

From the preliminary results of Section 6.4, it seems that the multipolar approach has an almost linear time complexity for proximity queries. Although we have not yet done a formal complexity analysis, conceptually, this conjecture is very plausible. Recall that the time complexity of a proximity query depends on the number of points falling into a specific sphere in the transformed data. As the data set size increases, the point density increases linearly. If the linear increment in point density is somewhat preserved by the multipolar mapping, a linear increment in the number of points inside the sphere would be witnessed. In addition, an increase in point density would result in a smaller search radius required

for nearest neighbor queries, and we suspect that the time complexity of nearest neighbor queries would be even better than linear. Two series of experiments were conducted to explore these two conjectures. The first used data sets of sizes from 10^4 to 10^5; the second from 10^5 to 10^6.

Figure 6.11 shows the performance results of 5-polar mappings on metric data sets with $(m, n, p, c, r) = (3, n, 20, 100, 20)$, where $n = 10^4, 2 \times 10^4, \ldots, 10 \times 10^4$. All the results are estimated values based on 100 randomly selected points from the data sets.

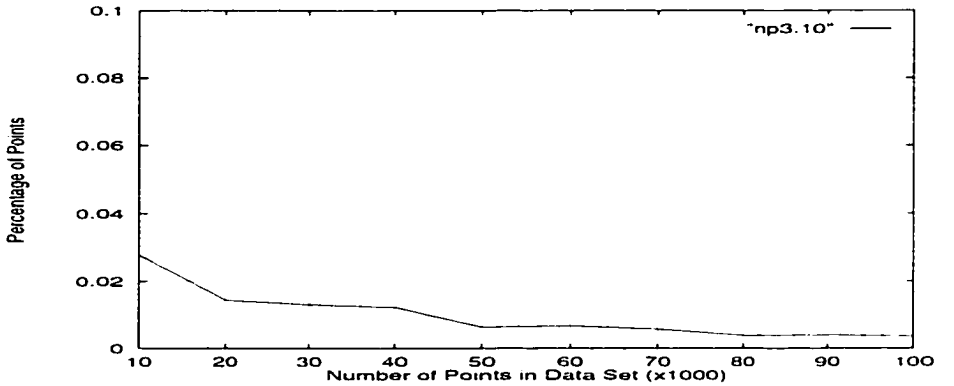
Figure 6.12 shows the performance results of 5-polar mappings on metric data sets with $(m, n, p, c, r) = (3, n, 20, 100, 20)$, where $n = 10^5, 2 \times 10^5, \ldots, 10 \times 10^5$. For easy comparison, the result of the data set with $n = 10^4$ is also included in the figure. All the results are estimated values based on 100 randomly selected points from the data sets.

From both Figures 6.11(a) and 6.12(a), it is clear that the time complexity of nearest neighbor queries is better than linear. Note that a linear time complexity would appear as a horizontal line. Upon closer examination, it seems that the percentage of points to be checked for nearest neighbor queries on $n = 10^5$ is about 10% of the percentage for $n = 10^4$. Similarly the percentage for $n = 10^5$ is about 10% of the one for $n = 10^6$. Thus, both curves suggest close to constant time complexity for nearest neighbor queries based on our performance model.

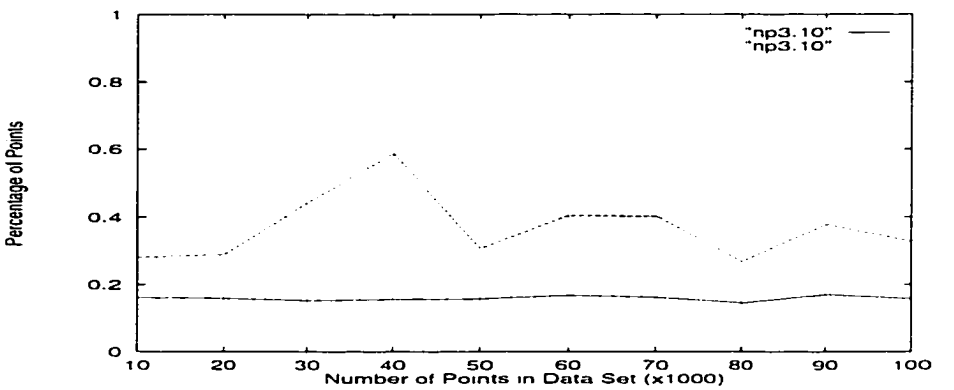
While the curves in Figures 6.11(b) and 6.12(b) are not strict horizontal lines, they are strong evidences suggesting that the overall time complexity for proximity queries is indeed linear, or close to linear. Although there are peculiar cases such as the ones for $n = 4 \times 10^4, 5 \times 10^5, 7 \times 10^5$, they are exceptions rather the rule. One has to remember that there exist performance differences among 5-polar mappings on the same data set based on choice of pole sets. We believe that we happened to pick an underperformer in each of the cases for $n = 4 \times 10^4, 5 \times 10^5, 7 \times 10^5$. The performance differences among multipolar mappings on the same data set with the same number of poles is studied in Section 6.8.

6.7 Number of Poles

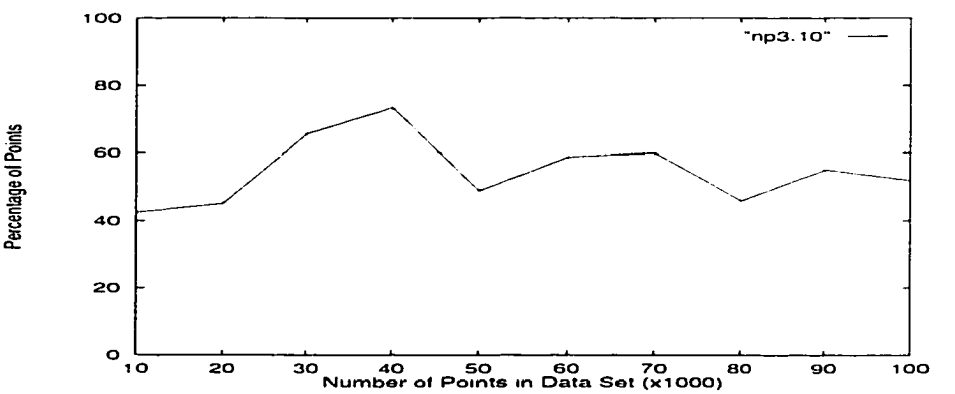
From Equation 5.5, Section 5.5 and Theorem B.6, Appendix B, it is clear that a pole set P_1 would outperform a pole set P_2 if $P_2 \subset P_1$. Based on the theoretical result, we conjecture that if $|P_2| < |P_1|$, P_1 is *likely* to outperform P_2. The objective of this section is to validate this conjecture experimentally.



(a) Nearest neighbor queries.

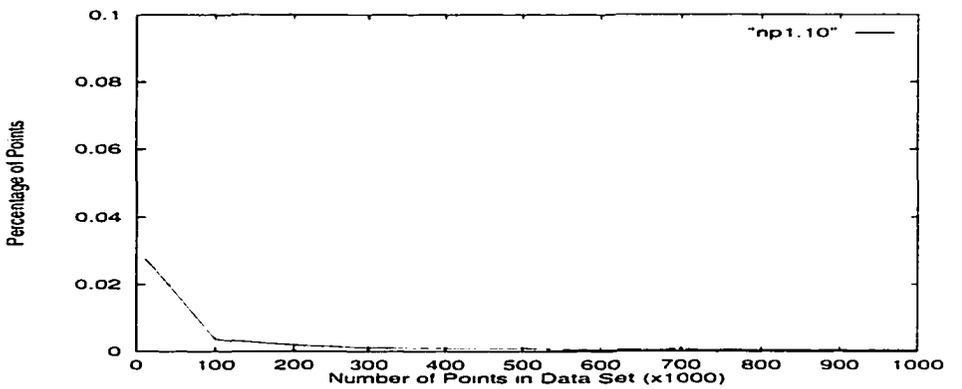


(b) 10% proximity queries.

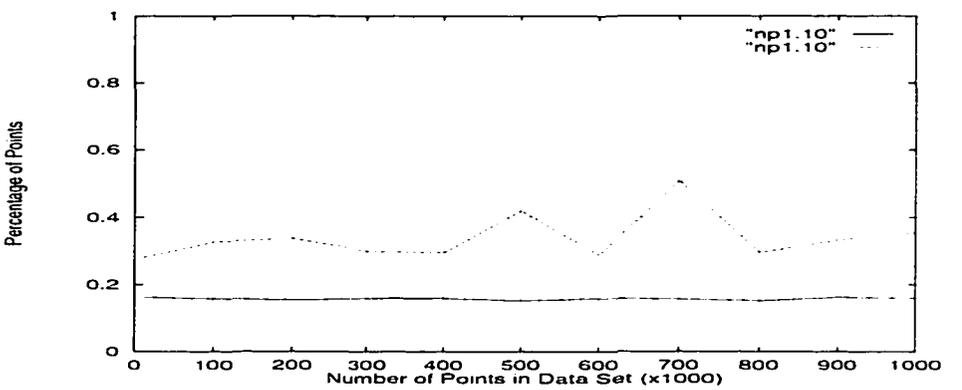


(c) False-positives for 10% proximity queries.

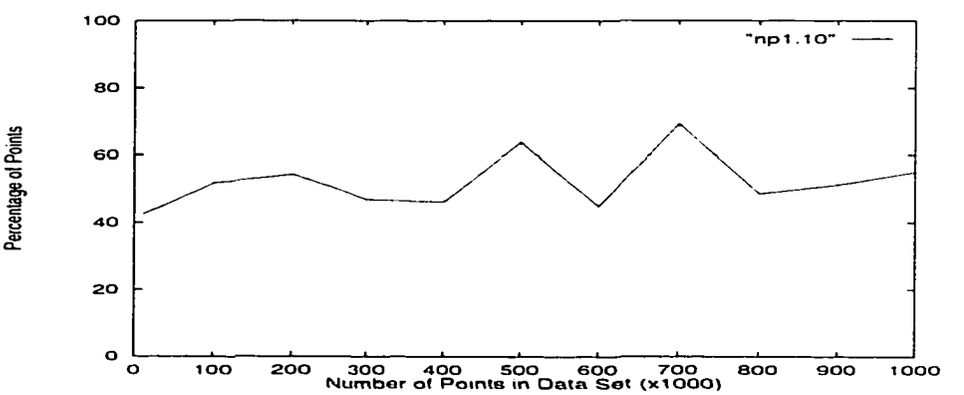
Figure 6.11: $(m.n.p.c.r) = (3.n.20.100.2)$. $n = 10^4.2 \times 10^4 \ldots . 10 \times 10^4$. Efficiency for data set of various sizes.



(a) Nearest neighbor queries.



(b) 10% proximity queries.



(c) False-positives for 10% proximity queries.

Figure 6.12: $(m, n, p, c, r) = (3, n, 20, 100, 2)$. $n = 10^5, 2 \times 10^5, \ldots, 10 \times 10^5$. Efficiency for data set of various sizes.

Three collections of data sets were generated with an identical set of parameters. $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$, where $p = 0.25, 0.50, 0.75$. The experiment results appear in Figures 6.13, 6.14, and 6.15, based on 100 randomly selected points from each of the transformed data sets.

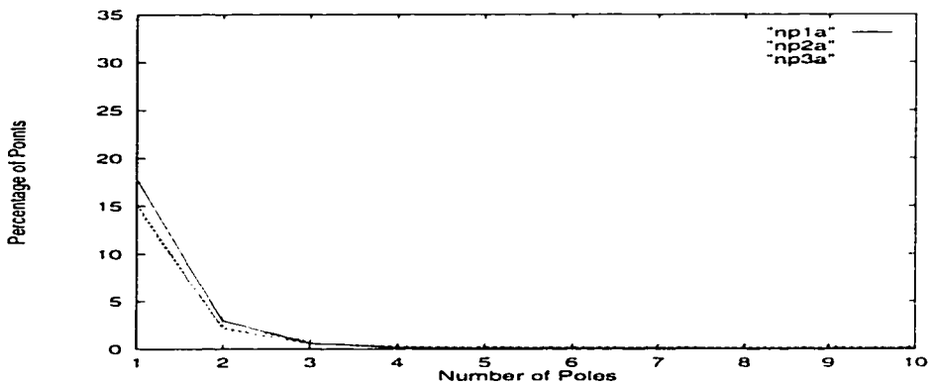
Based on results from the three independent series, clearly, as the number of poles increases, performance improves. From subfigures (a) and (b) of Figures 6.13, 6.14, and 6.15, it is clear that the performance does not vary significantly for this range of p values. Nonetheless, data sets with lower p values do have a slightly better overall performance, except when only a small number of poles are employed (e.g., three poles or less). Performance with regard to a small number of poles has more to do with the quality of pole sets than the degree of clusterness. As the number of poles increases, the performance differences among sets of poles diminish and the effect of clusterness prevails. However, subfigure (c) of Figures 6.13, 6.14, and 6.15 show that lower p values do tend to have more false positives than the ones with higher p values. Since a low p value implies a low P_f value, the percentage of false positives rises as p increases.

6.8 Pole Selection

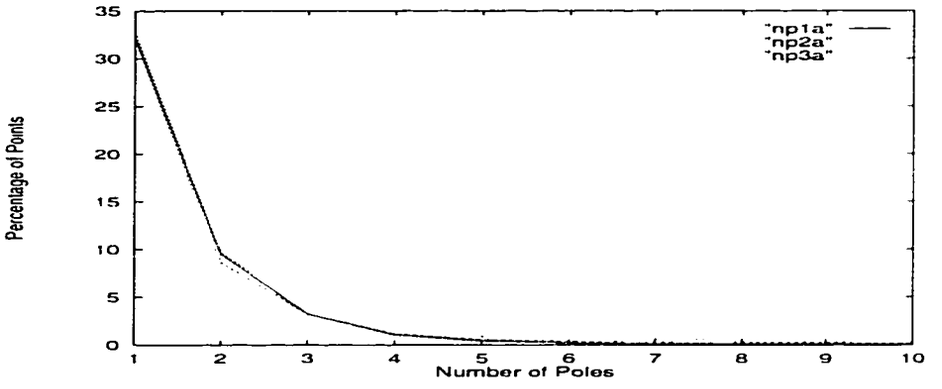
From the results presented in Sections 6.5, 6.6, and 6.7, it is clear that the selection of poles does have an impact on the performance of multipolar mappings. Compared to data parameters such as size, dimensionality and clusterness, or the number of poles utilized, the impact of pole selection is usually less significant. However, we have little control over the data parameters, and a higher number of poles results in higher space complexity and other performance overheads which are not part of our current performance model. Thus, better pole selection remains the most cost-effective way to improve performance under many circumstances.

In Section 5.6, we define the notion of an average interpole distance which can be used as a relative quality measurement for a collection of pole sets. Several series of experiments were conducted to explore the relationship between average interpole distances and performance of multipolar mappings. The results of these experiments are presented and analyzed in this section. At the end, we derive a set of simple guidelines for better pole selection based on the results.

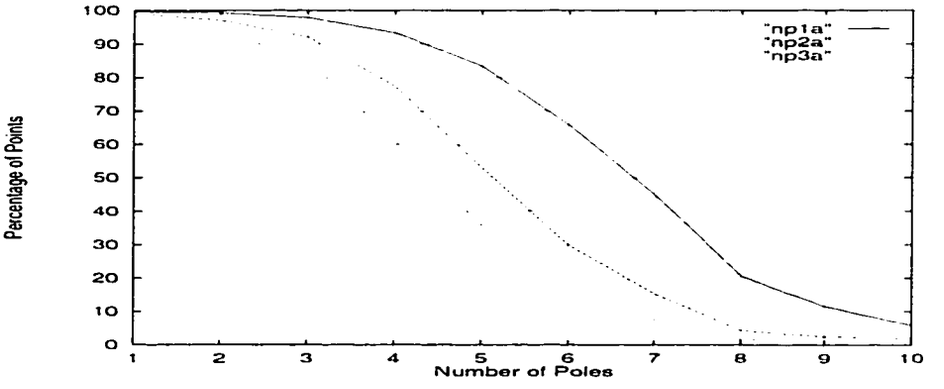
Figure 6.16 illustrates the performance of a group of 4-polar mappings with a wide



(a) Nearest neighbor queries.

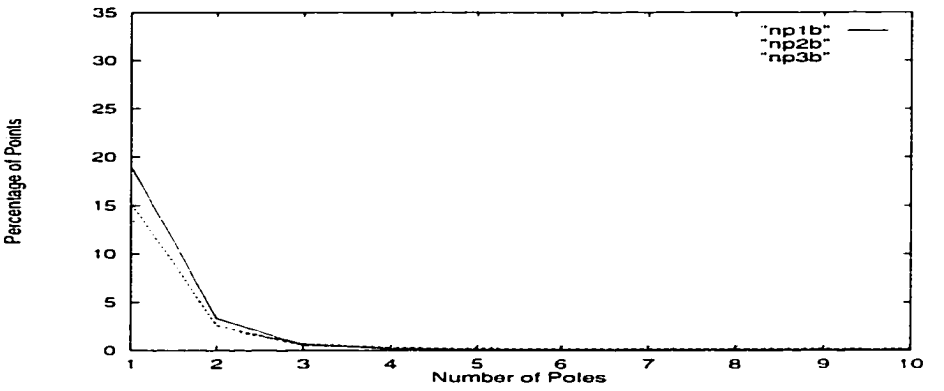


(b) 10% proximity queries.

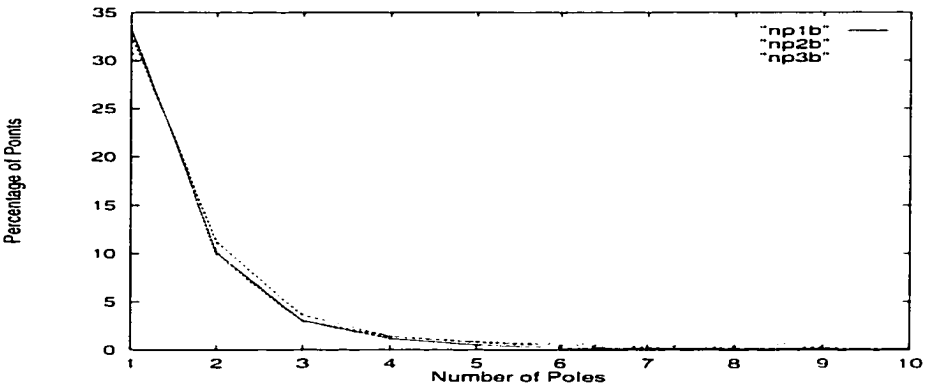


(c) False-positives for 10% proximity queries.

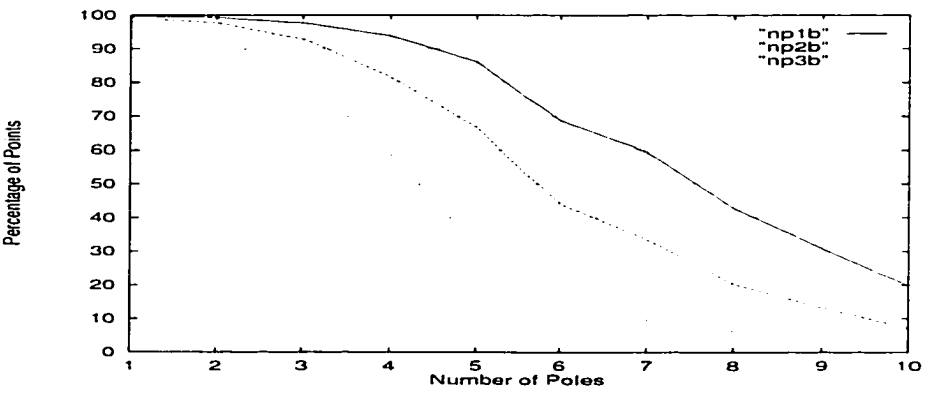
Figure 6.13: Collection 1. $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$. $p = 0.25, 0.50, 0.75$. Efficiency with respect to $1, \ldots, 10$ poles.



(a) Nearest neighbor queries.

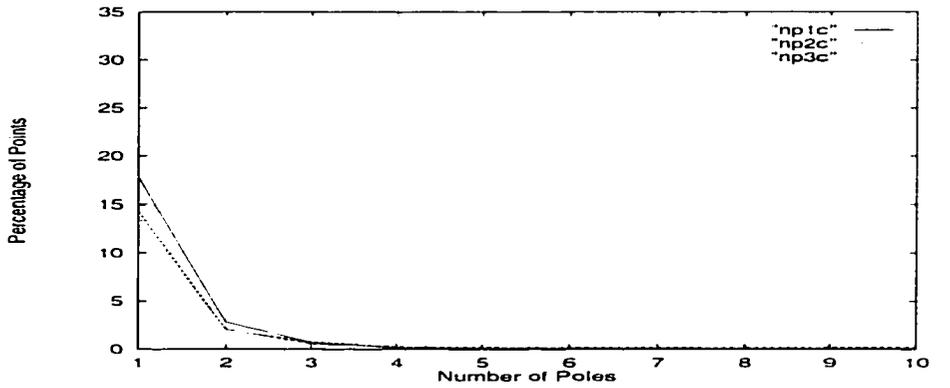


(b) 10% proximity queries.

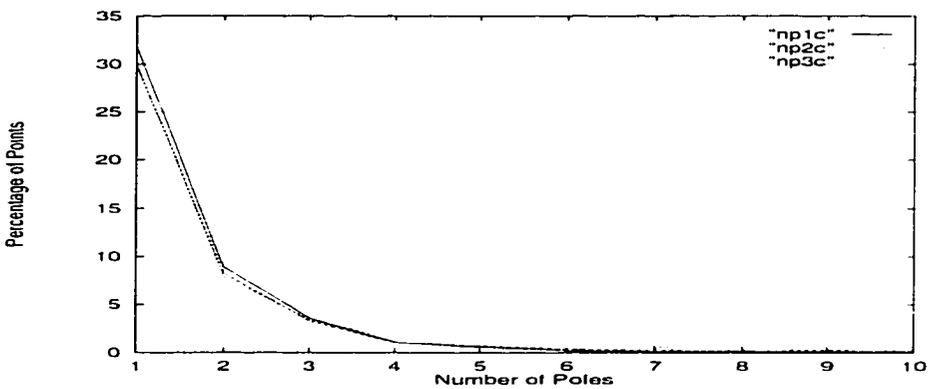


(c) False-positives for 10% proximity queries.

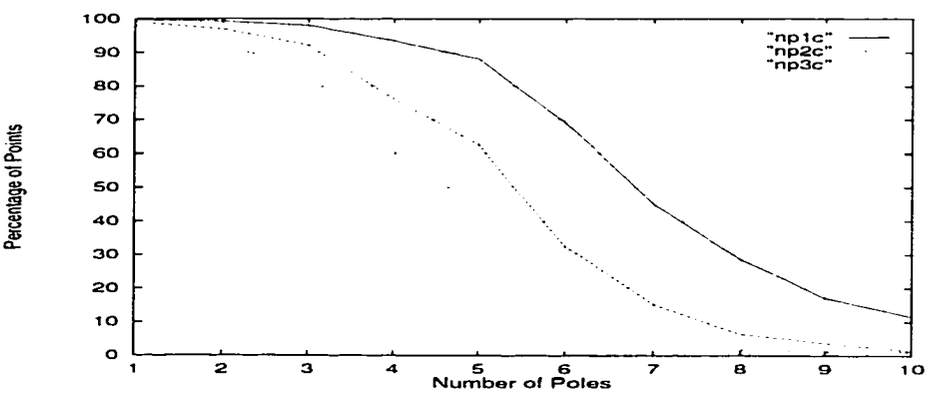
Figure 6.14: Collection 2. $(m.n.p.c.r) = (10.10^4.p.100.2)$. $p = 0.25.0.50.0.75$. Efficiency with respect to $1.....10$ poles.



(a) Nearest neighbor queries.



(b) 10% proximity queries.



(c) False-positives for 10% proximity queries.

Figure 6.15: Collection 3. $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$. $p = 0.25, 0.50, 0.75$. Efficiency with respect to $1, \ldots, 10$ poles.

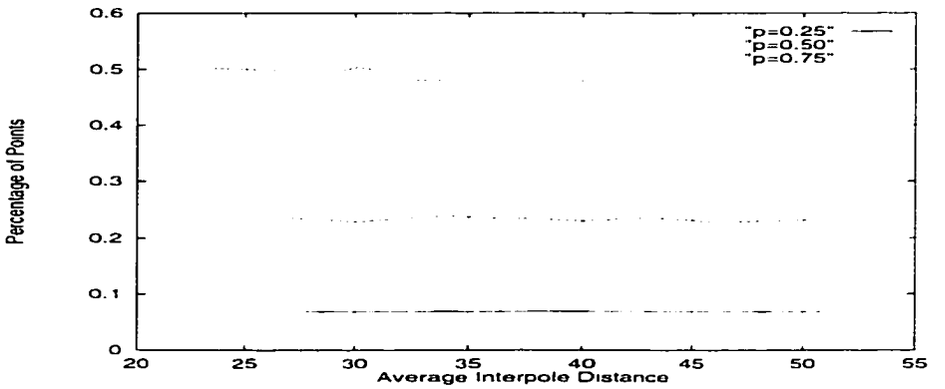
range of average interpole distances on data sets of $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$, where $p = 0.25, 0.50, 0.75$. The performance of nearest neighbor queries is very stable over the whole spectrum of average interpole distances (Figure 6.16(a)). This is hardly surprising, however: from subfigure (a) of Figures 6.13, 6.14, and 6.15, it is clear that a 4-polar mapping has enough differentiation power to provide good performance for small search radii required for nearest neighbor queries. Using more than 4 poles has little effect on performance. If the performance can not be further improved by adding poles, the variation in average interpole distances certainly does not matter.

Figure 6.16(b) and (c) indicate that for 10% proximity queries, the performance does improve as the average interpole distance increases. However, it seems that once the average interpole distance reaches the average distance of the data (approximately 38) the performance does not improve significantly.

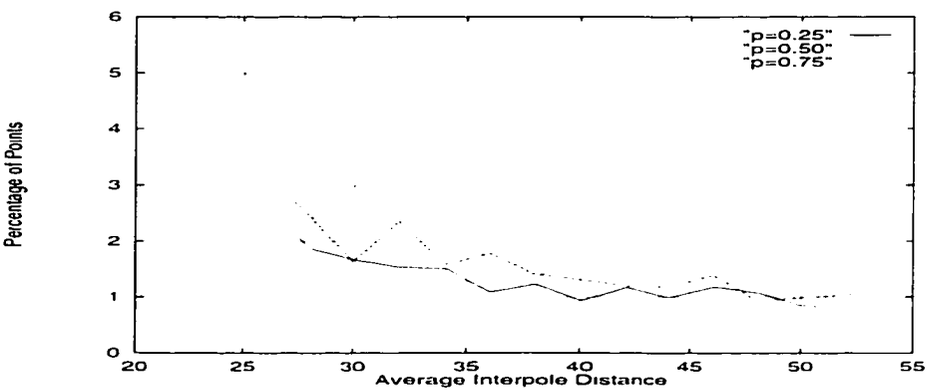
In addition to the average interpole distance, we can measure the quality of a pole set by the standard deviation of its interpole distances. Figure 6.17 illustrates the average interpole distances versus standard deviations for 1,000 randomly generated 4-pole sets on data sets of $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$, where $p = 0.25, 0.50, 0.75$ the same collection of data sets used in Figure 6.16.

Two groups of pole sets can be identified in each subfigure of Figure 6.17. The smaller group appears above and slightly to the left of the larger group. By inspection, we determined that this outlier group consists of all pole sets which contain at least two poles in the same cluster. Since the distance between these two poles is much smaller than their distances to the other poles, these pole sets have a higher than usual standard deviation. From a distance perspective, points of the same cluster share similar views toward the other points outside the cluster. This is particularly true for small clusters. Thus, a 4-pole set with two poles in the same cluster has the differentiation power roughly equivalent to that of a 3-pole set. For convenience, we call the group of pole sets with at least two poles in the same cluster as the *degenerate group*, and the other group as the *normal group*. In the degenerate group, the larger the average interpole distance, the larger the standard deviation. Also, as the value of p increases, the probability for a pole set to have at least two poles in the same cluster increases, which results in a larger degenerate group.

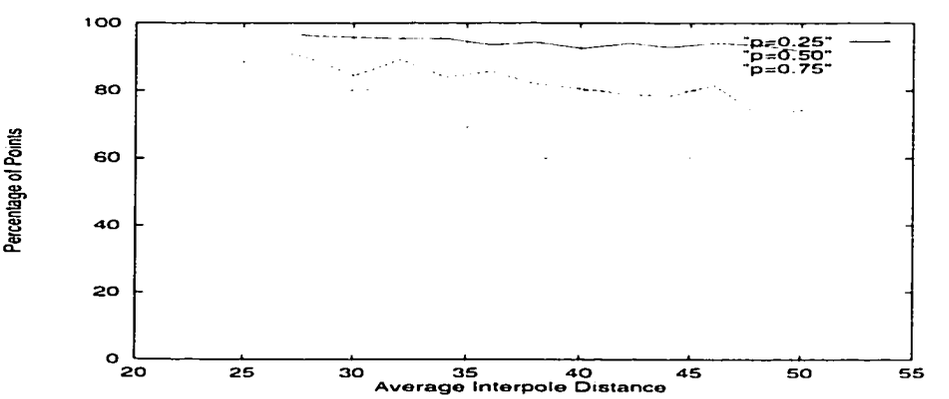
Using the same data set as Figure 6.17(c), we illustrate the results for 1,000 6-pole, 8-pole, and 10-pole sets in Figure 6.18. For clear identification, we test each pole set to see if there are two poles in the same cluster. Pole sets in the degenerate group are marked as



(a) Nearest neighbor queries.

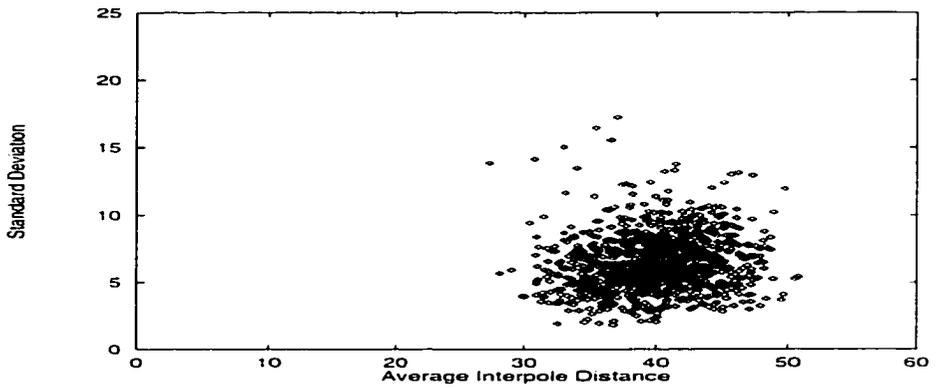


(b) 10% proximity queries.

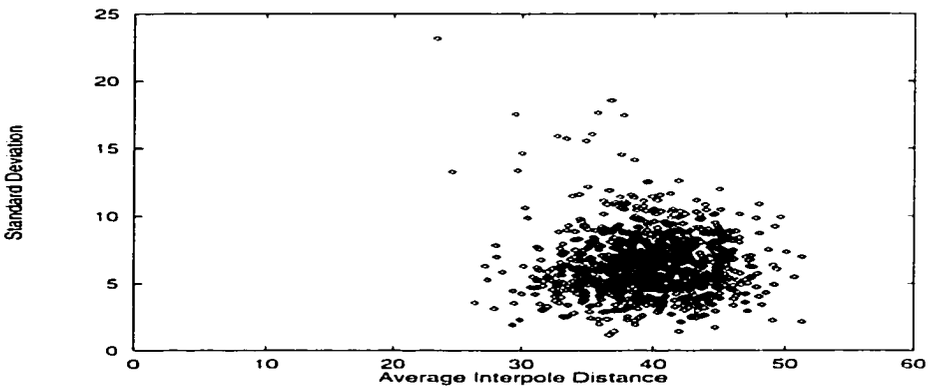


(c) False-positives for 10% proximity queries.

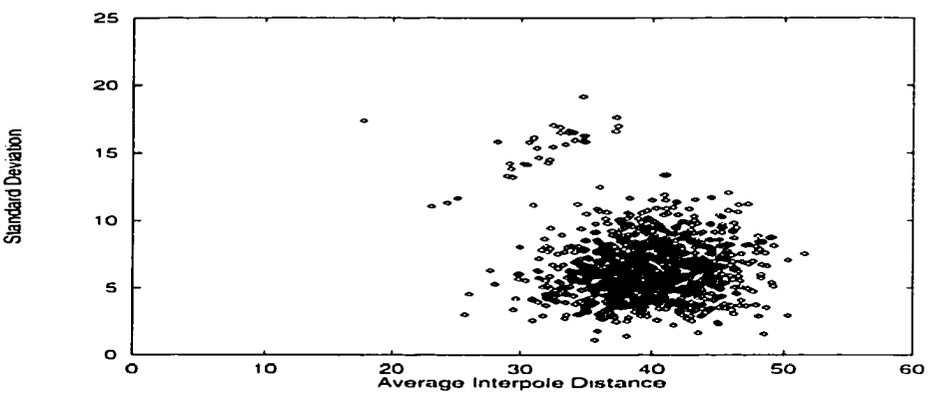
Figure 6.16: $(m, n, p, c, r) = (10, 10^4, p, 100, 2)$. $p = 0.25, 0.50, 0.75$. Efficiency of 4-polar mappings with respect to average interpole distances.



(a) $(m, n, p, c, r) = (10, 10^4, 0.25, 100, 2)$. 1,000 4-pole sets.



(b) $(m, n, p, c, r) = (10, 10^4, 0.50, 100, 2)$. 1,000 4-pole sets.



(c) $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$. 1,000 4-pole sets.

Figure 6.17: Average interpole distances versus standard deviations.

- in the figures. As the number of poles increases, it becomes harder to differentiate the two groups based on average interpole distances and standard deviations. Although the degenerate group is not as well separated from the normal group, its members still tend to have higher standard deviations than members of the normal group.

The performance of 4-pole sets on $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$ is illustrated in Figure 6.19, based on the average interpole distances. Two series of pole sets are selected from both the normal group and the degenerate group such that the whole spectrum of average interpole distance are represented. The two curves in each subfigure correspond to samples in the normal group and the degenerate group respectively. As we can see, while there is no significant performance differences in nearest neighbor queries for the two groups, the normal group performs better for 10% proximity queries.

The performance of 4-pole sets on $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$ is illustrated in Figure 6.20, based on the standard deviations of interpole distances. The results are based on all the pole sets in the normal group with an average interpole distance between 34 and 36. Clearly, standard deviation has little influence on the performance of both nearest neighbor queries and proximity queries.

From the results presented in Figures 6.16 and 6.19, we can derive the following simple guidelines for pole selection with a fixed number of poles.

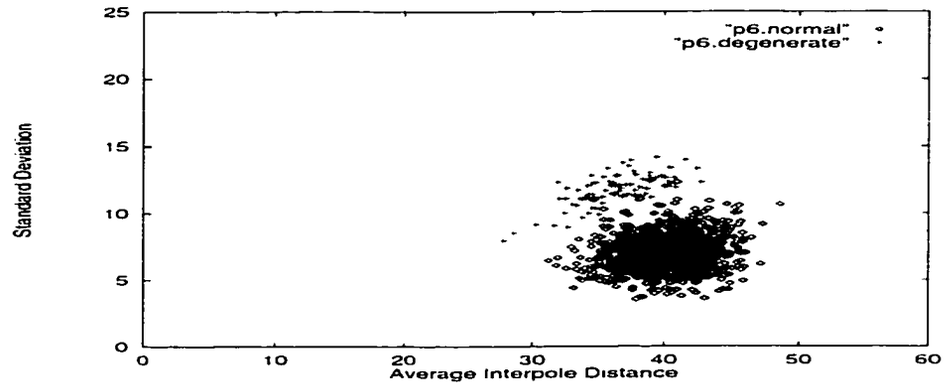
1. For nearest neighbor queries, the performance is so stable that almost any pole set provides good performance.

2. For proximity queries, there does exist appreciable performance differences among pole sets. An ideal pole set would satisfy the following two criteria:

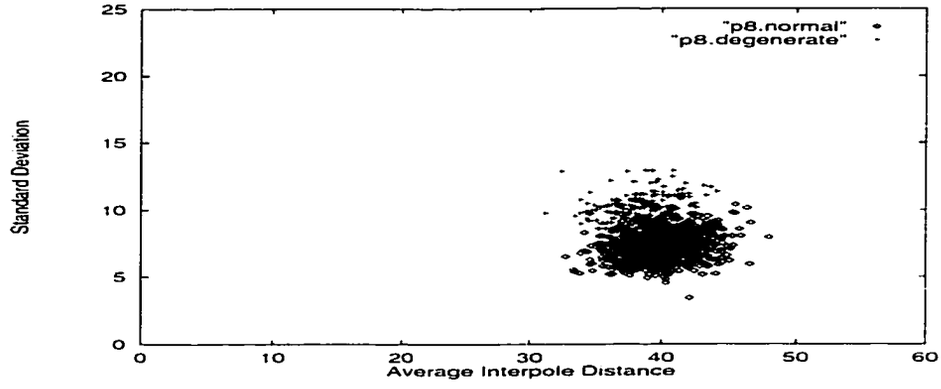
   (a) There are no two poles in the same cluster.

   (b) The average interpole distance is above the average interpoint distance.

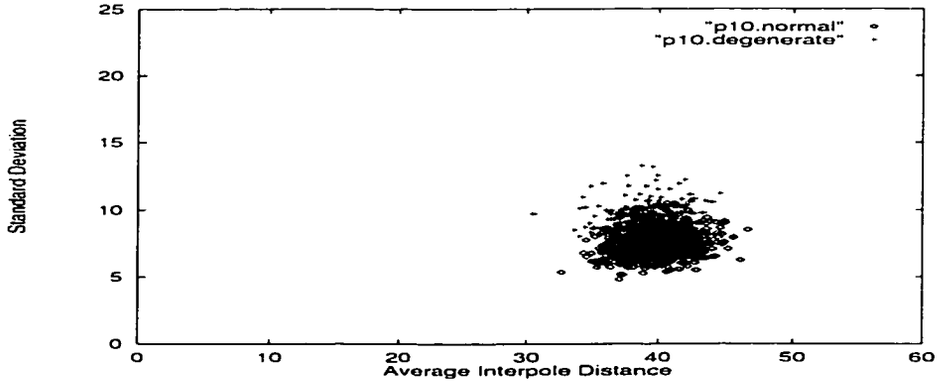
In general, we do not have *a priori* knowledge of the data set in question, e.g., the cluster sizes and average distance. However, both criteria can still be easily verified with a good confidence level through random sampling. For instance, we can randomly generated n pole sets and computed their average interpole distances. This would give us an estimate of the average interpoint distance. Among those with average interpole distance larger than the average interpoint distance, we can pick one without a pair of poles which are particularly close to each other.



(a) $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$. 1,000 6-pole sets.

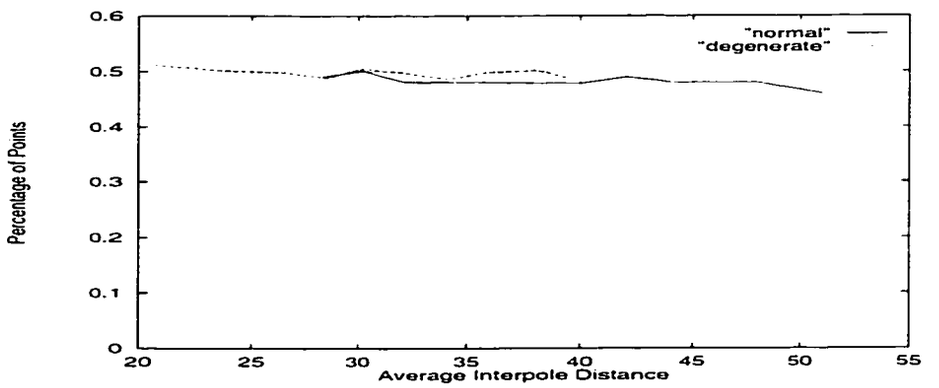


(b) $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$. 1,000 8-pole sets.

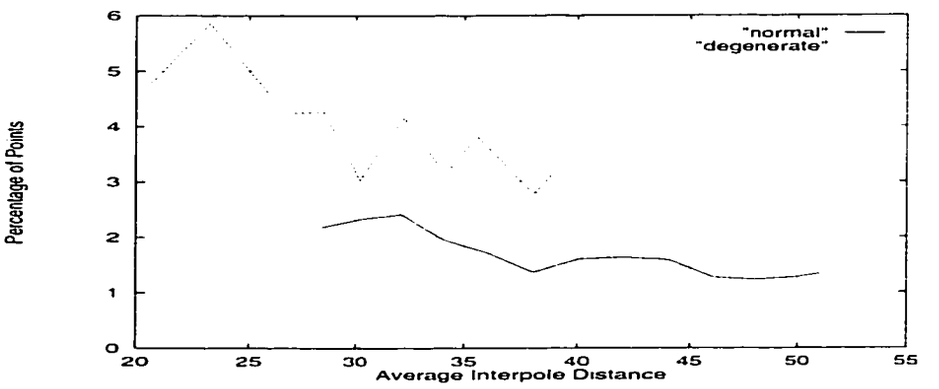


(c) $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$. 1,000 10-pole sets.

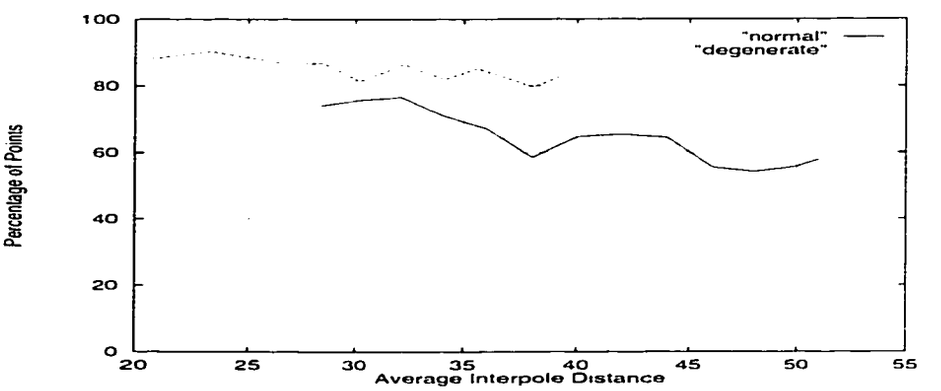
Figure 6.18: Average interpole distances versus standard deviations.



(a) Nearest neighbor queries.

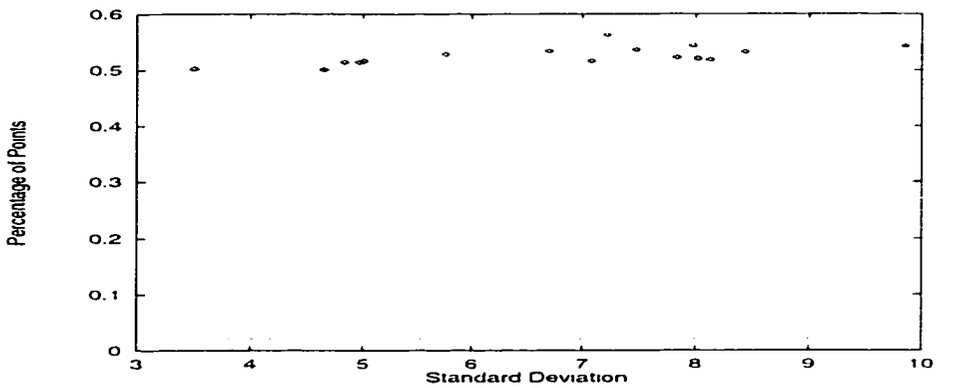


(b) 10% proximity queries.

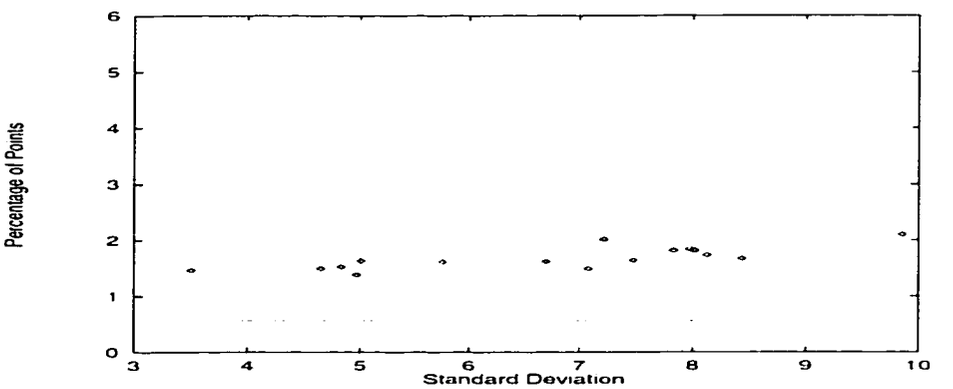


(c) False-positives for 10% proximity queries.

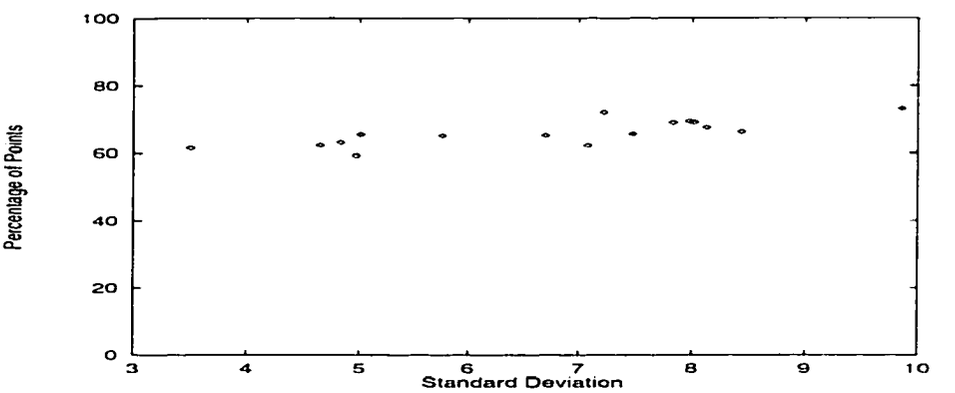
Figure 6.19: $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$. Efficiency of 4-polar mappings with respect to average interpole distances.



(a) Nearest neighbor queries.



(b) 10% proximity queries.



(c) False-positives for 10% proximity queries.

Figure 6.20: $(m, n, p, c, r) = (10, 10^4, 0.75, 100, 2)$. Efficiency for 4-polar mappings with respect to the standard deviations of interpole distances.

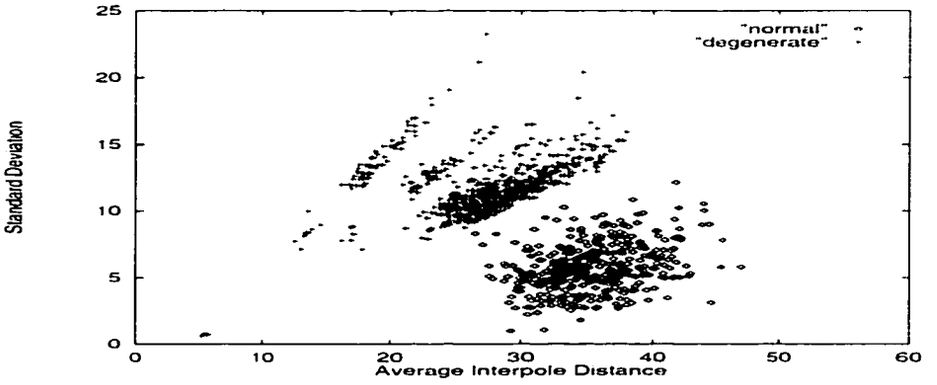


Figure 6.21: $(m, n, p, c, r) = (10, 10^4, 0.75, 4, 4)$. 1,000 4-pole sets.

6.9 Cluster Centers as Poles

Theoretically, it seems that the centers of clusters might be particularly good candidates for poles, since they provide a less biased view to the other points in the clusters. This conjecture can not be verified from the experiments described in Section 6.8, since they are all based on clustered data sets with 100 clusters. With so many clusters, whether the poles are cluster centers or not has almost nothing to do with the performance of the pole set. The objective of this section is to study the validity of our conjecture by generating and testing on data sets with a small number of clusters.

Figure 6.21 illustrates the average interpole distances versus standard deviations for 1,000 randomly selected pole sets on $(m, n, p, c, r) = (10, 10^4, 0.75, 4, 4)$. Since there are only 4 clusters in the data, there is a high probability that a pole set has two poles in the same cluster, resulting in a rather large degenerate group. In fact, the degenerate group can be further decomposed into 4 subgroups which correspond to:

1. pole sets with exactly one pair of poles in the same cluster,
2. pole sets that consist of two pairs of poles in which each pair are in the same cluster,
3. pole sets with a triplet of poles in the same cluster, and
4. pole sets with all four poles in the same cluster.

Figure 6.22 illustrates the performance of 4-polar mappings with respect to average interpole distances on $(m, n, p, c, r) = (10, 10^4, 0.75, 4, 4)$. The normal group and degenerate

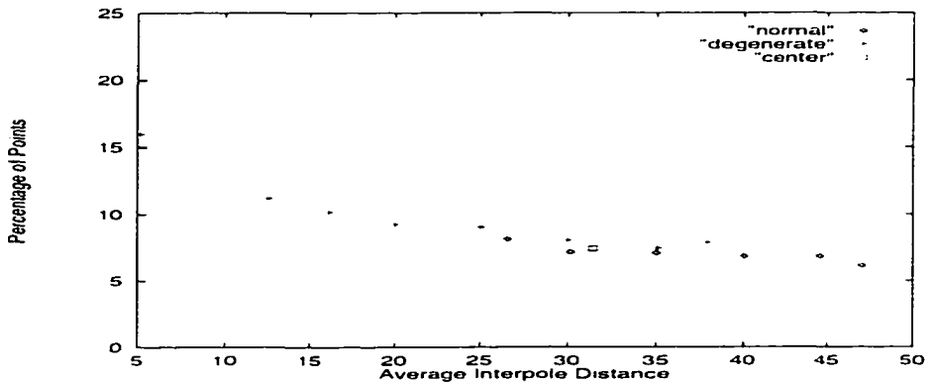
group are represented by $\circ$ and $\cdot$ respectively. The 4-pole set consisting of the 4 cluster centers is marked as $\square$. The normal group still performs better than the degenerate group in both the nearest neighbor queries and proximity queries. However, the pole set consisting of cluster centers does not seem to have a performance advantage over the other pole sets in the normal group. Overall the performance is much worse compared to Figure 6.19. This is understandable, however. Note that there are only 4 clusters, and points are distributed uniformly inside each of the clusters. While it is easy to differentiate points from different clusters, it becomes much harder to differentiate points from the same cluster. The fact that there is a significant portion of points uniformly distributed in a cluster gives rise to the curse of dimensionality at the intra-cluster level.

Another series of similar experiments was performed on a 2-dimensional data set, $(m, n, p, c, r) = (2, 10^4, 0.75, 4, 5)$, which is illustrated as a scatter plot in Figure 6.23. Compared to $(m, n, p, c, r) = (10, 10^4, 0.75, 4, 4)$, this data set has a much higher point density. Figure 6.24 shows 1,000 randomly generated 4-pole sets in which both the normal group and degenerate group are clearly marked. Again the degenerate group overlaps the normal group but tends to be slightly above and to the left of it. The performance results are presented in Figure 6.25. As we can see, the results of $(m, n, p, c, r) = (2, 10^4, 0.75, 4, 5)$ are better than the results of $(m, n, p, c, r) = (10, 10^4, 0.75, 4, 4)$. The pole set consisting of the cluster centers performs no better than the other pole sets in the normal group.

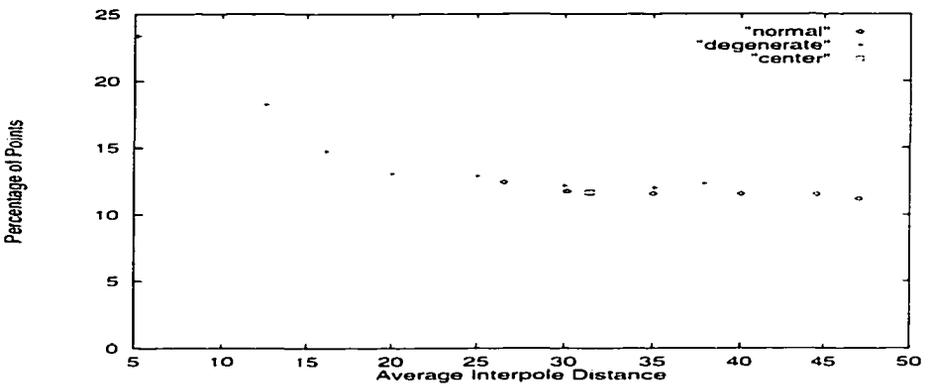
6.10 Greedy Algorithm

Based on the guidelines for pole selection in Section 6.8, a greedy algorithm can be implemented to find a "good" pole set. First, our greedy algorithm randomly generates 1,000 pole sets. Second, the algorithm sorts all the pole sets in the normal group and outputs the pole set with the highest average interpole distance in this group. The objective of this section is to evaluate the merits of such a greedy algorithm.

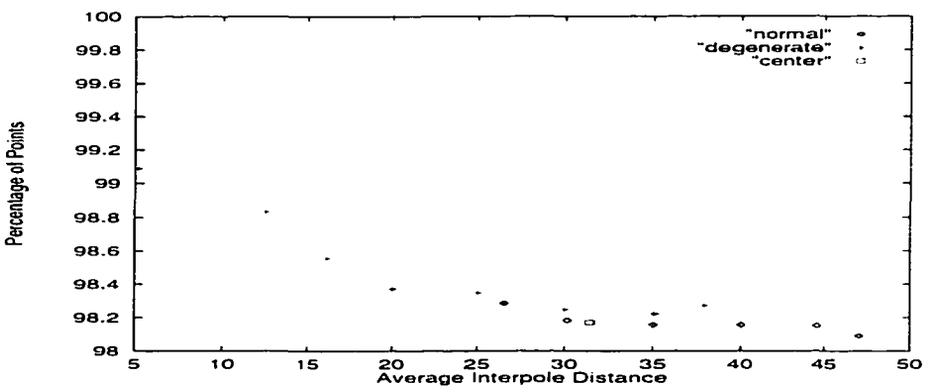
Figures 6.26, 6.27, and 6.28 duplicate the results of Figure 6.13. For each subfigure, in addition to the curve of the randomly selected pole sets used in Figure 6.13, another curve is drawn to represent the pole sets generated by the greedy algorithm. It is clear that the pole set selected by a greedy algorithm does not always guarantee better performance. Even when better performance is indeed achieved by the greedy algorithm, the improvement is very marginal. Thus, such an aggressive approach is not necessary. It suffices to follow the



(a) Nearest neighbor queries.



(b) 10% proximity queries.



(c) False-positives for 10% proximity queries.

Figure 6.22: $(m, n, p, c, r) = (10, 10^4, 0.75, 4, 4)$. Efficiency for 4-polar mappings with respect to average interpole distances.

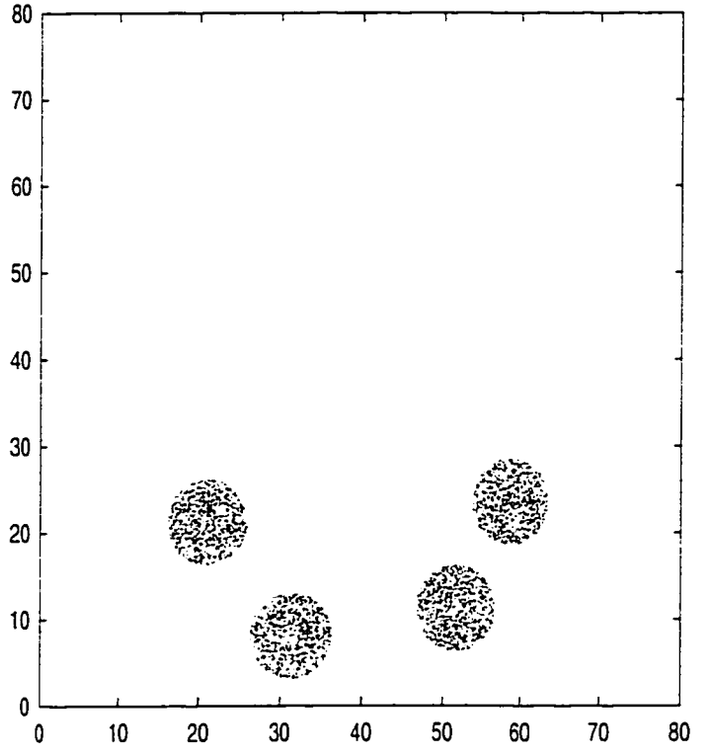


Figure 6.23: $(m.n.p.c.r) = (2.10^{4}.0.75.4.5)$. 2-dimensional scatter plot.

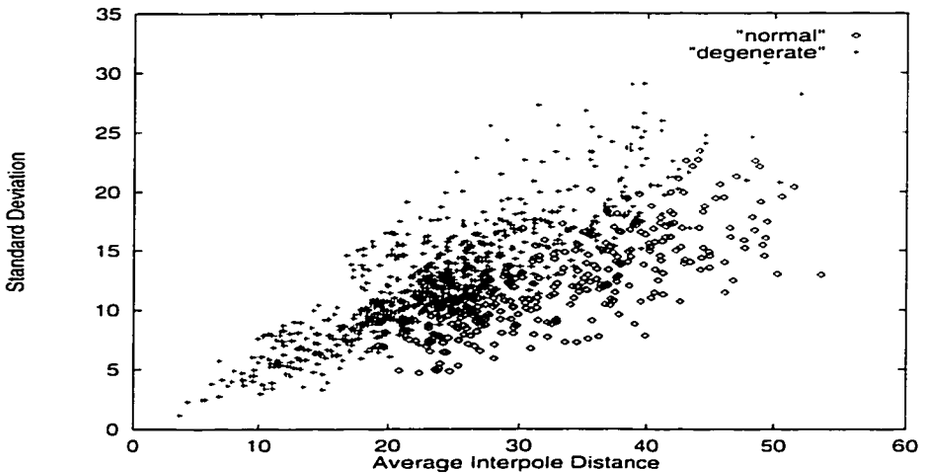
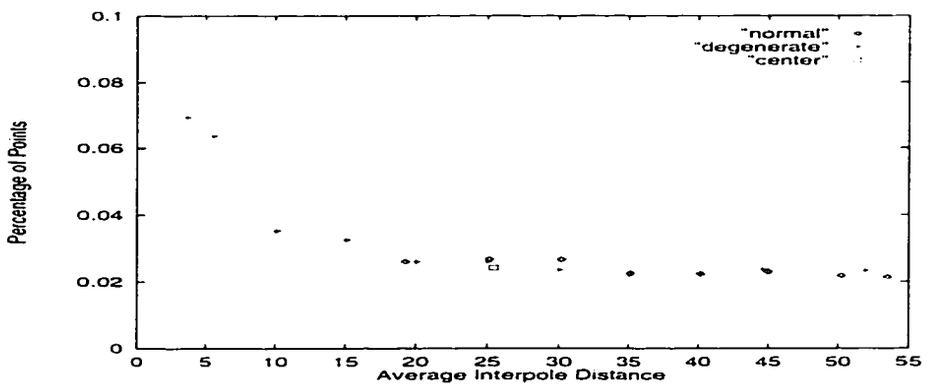
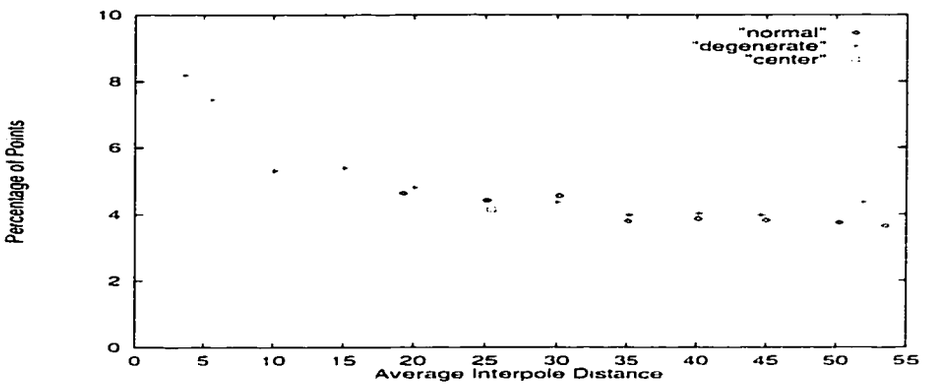


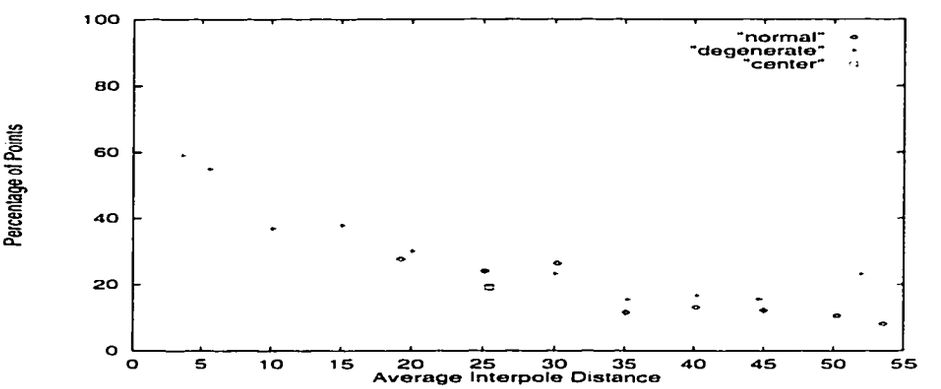
Figure 6.24: $(m.n.p.c.r) = (2.10^{4}.0.75.4.5)$. 1000 4-pole sets. Average interpole distances versus standard deviations.



(a) Nearest neighbor queries.



(b) 10% proximity queries.



(c) False-positives for 10% proximity queries.

Figure 6.25: $(m, n, p, c, r) = (2, 10^4, 0.75, 4, 5)$. Efficiency of 4-polar mappings with respect to average interpole distances.

simple guidelines (Section 6.8) to find a reasonable pole set.

Figure 6.29 illustrates the performance of multipolar mapping with respect to the number of poles. Five curves are plotted in each subfigure which correspond to the 1st, 3rd, 6th, 10th, and 1000th pole sets in 1000 randomly selected pole sets from the normal group sorted by average interpole distance in ascending order. The results are consistent with previous work. Pole sets with high average interpole distances tend to perform better than ones with low average interpole distances. However, if enough poles are used, say four poles for $(m, n, p, c, r) = (10, 10^4, 0.50, 100, 2)$, the performance can no longer be significantly improved and thus the variation in average interpole distances has very little effect on performance.

6.11 Very High Dimensionality

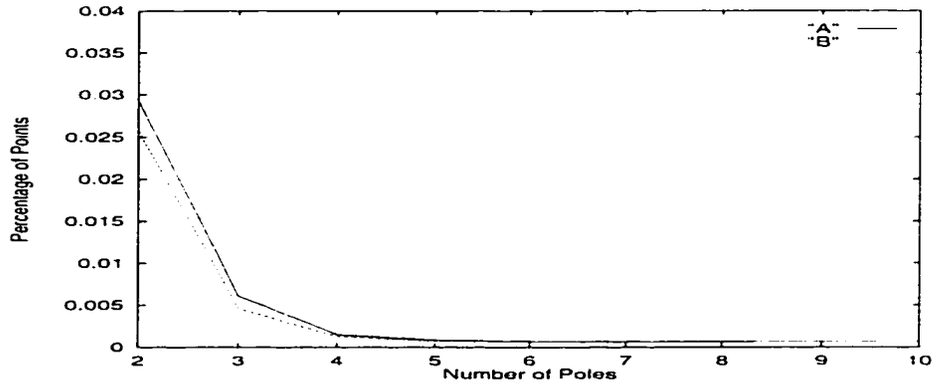
Most results presented in this chapter are based on metric data sets of 10 intrinsic dimensions. In order to evaluate the curse of dimensionality in higher dimensions, several tests on data sets of $m = 100$ and $p = 0.25$ have been performed. These preliminary experiments show that with a 5-polar mapping, less than 0.1% of points are checked for nearest neighbor queries, and almost 20% of points are checked for 10% proximity queries, compared to 0.03% and 0.3% required respectively for data sets of $m = 10$ and otherwise identical parameters (Figure 6.11). It seems that while the curse of dimensionality has a significant impact on proximity queries, the performance of nearest neighbor queries is much less affected.

6.12 Summary

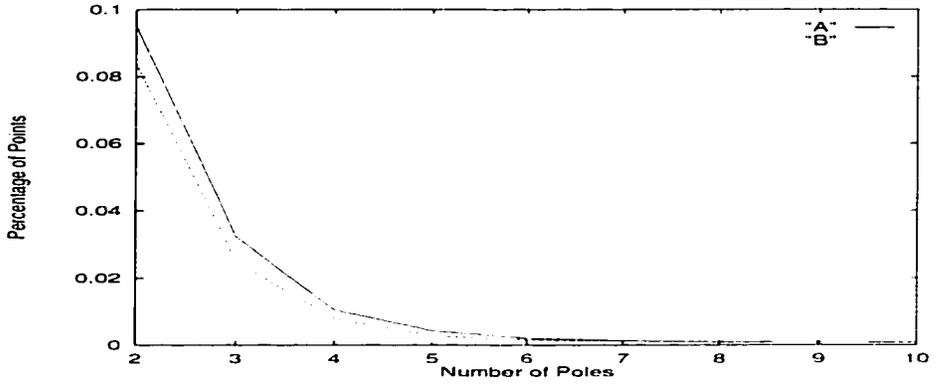
Based on the performance results presented in this chapter, we make the following general conclusions.

1. Although the curse of dimensionality puts a toll on the performance of both nearest neighbor and proximity queries, the impact on nearest neighbor queries on clustered data is not significant.

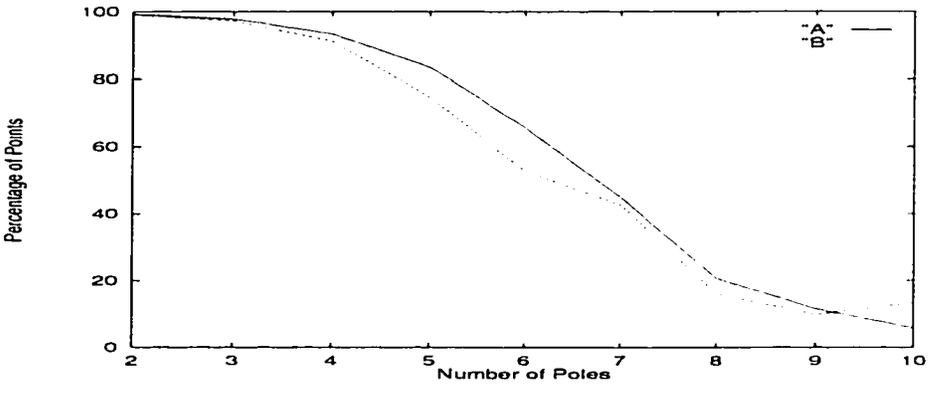
2. The multipolar approach has an almost linear time complexity for proximity queries. Time complexity for nearest neighbor queries is almost constant.



(a) Nearest neighbor queries.

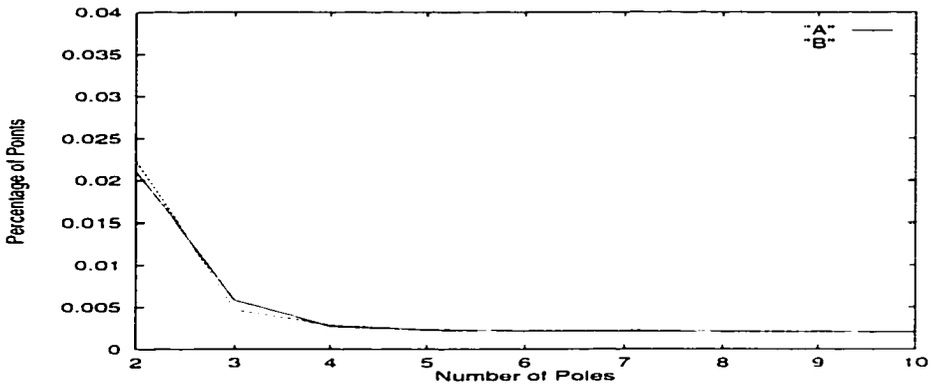


(b) 10% proximity queries.

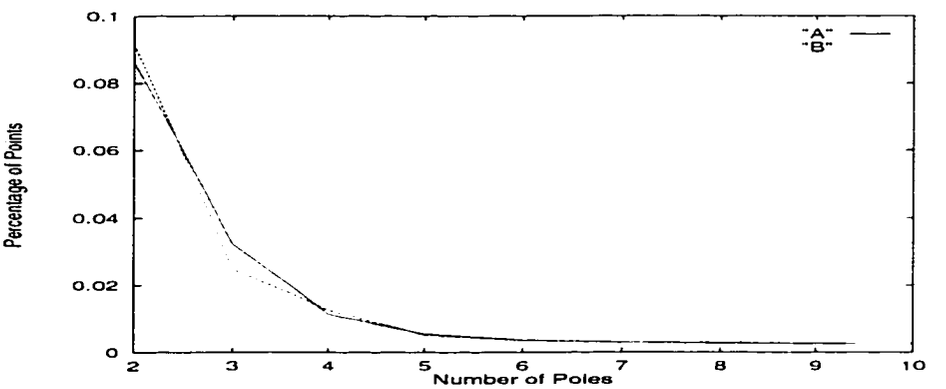


(c) False-positives for 10% proximity queries.

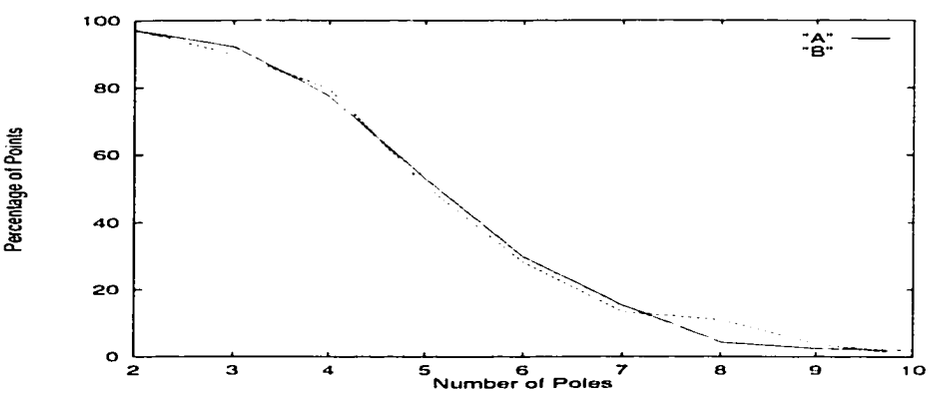
Figure 6.26: $(m, n, p, c, r) = (10, 10^4, 0.25, 100, 2)$. Efficiency for multipolar mappings with respect to the number of poles. "A" and "B" correspond to randomly selected pole sets and pole sets selected by greedy algorithm respectively.



(a) Nearest neighbor queries.

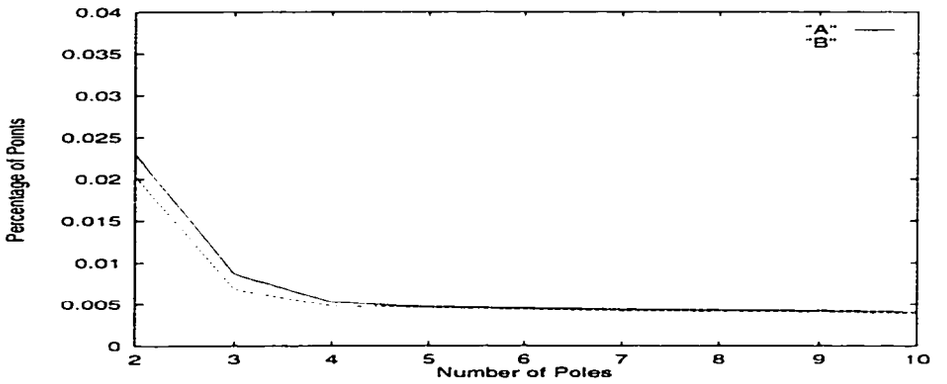


(b) 10% proximity queries.

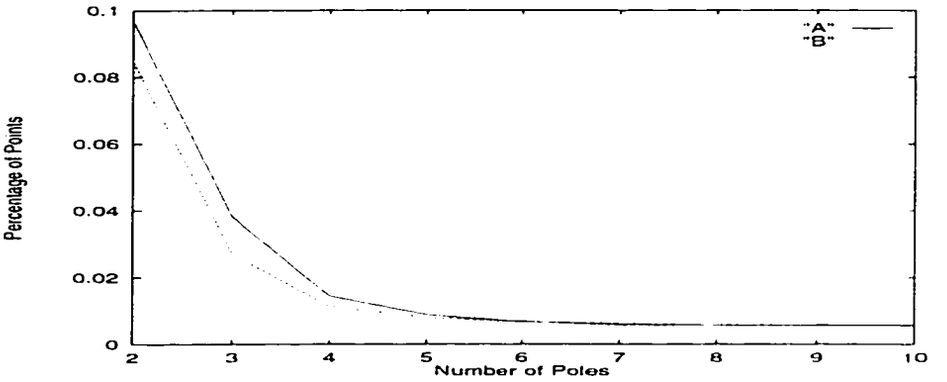


(c) False-positives for 10% proximity queries.

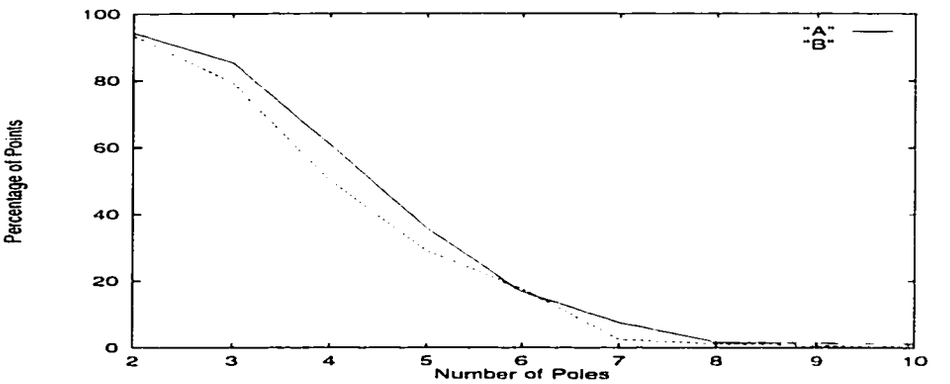
Figure 6.27: $(m, n, p, c, r) = (10, 10^4, 0.50, 100, 2)$. Efficiency for multipolar mappings with respect to the number of poles. "A" and "B" correspond to randomly selected pole sets and pole sets selected by greedy algorithm respectively.



(a) Nearest neighbor queries.

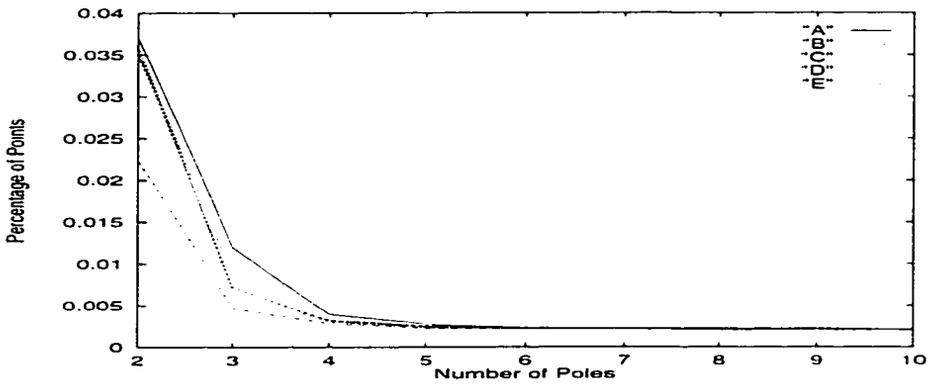


(b) 10% proximity queries.

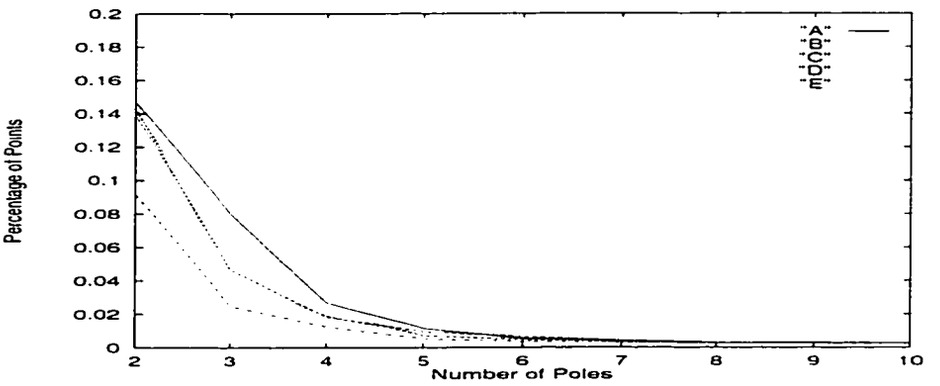


(c) False-positives for 10% proximity queries.

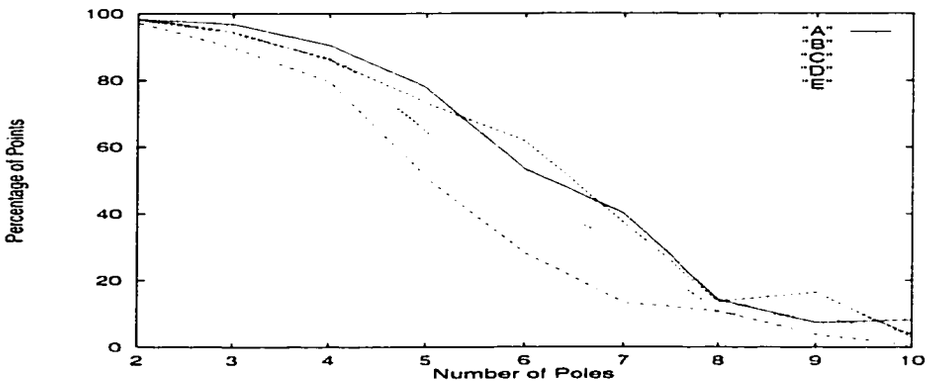
Figure 6.28: $(m.n.p.c.r) = (10.10^4.0.75.100.2)$. Efficiency for multipolar mappings with respect to the number of poles. "A" and "B" correspond to randomly selected pole sets and pole sets selected by greedy algorithm respectively.



(a) Nearest neighbor queries.



(b) 10% proximity queries.



(c) False-positives for 10% proximity queries.

Figure 6.29: $(m, n, p, c, r) = (10, 10^4, 0.50, 100, 2)$. Efficiency for multipolar mappings with respect to the number of poles. "A", "B", "C", "D", and "E" correspond to the 1st, 3rd, 6th, 10th, and 1000th pole sets in a collection of 1000 pole sets sorted by average interpole distance in ascending order.

3. Compared to intrinsic dimensionality, it seems that only a small number of poles is required to achieve good performance on clustered data. For instance, 4-polar mappings seem to be sufficient for 10-dimensional clustered data.

4. Simple and efficient guidelines are available to avoid poor pole sets which are relatively rare under most circumstances. Aggressive pole selection, such as a greedy algorithm, can not guarantee better performance most of the time. Even when there is a performance improvement, it is very marginal.

Items 2 and 3 are our conjectures based on experiments on data sets of $m \leq 10$. They are subject to further validation.

Chapter 7

Conclusions

7.1 Overview

In the recent past, there has been a growing trend in the database community to extend existing database technologies to scientific applications, such as *scientific database systems*, *multimedia database systems*, and *geographic information systems*. This trend is fueled by the ever-increasing complexity in scientific data and made possible by the availability of low-cost mass storage devices.

The objective of a database system is to maintain information and to make that information available on demand [12]. To maintain information is to have the information properly modeled and stored. To make the information available on demand requires effective and efficient data indexing. Scientific data, which include multimedia (e.g., images, audio, and video) and non-standard data (e.g., finger prints and DNA sequences), are characterized by rich interrelationships, both inter-entity and inter-instance. In order to apply database technologies to scientific data, new data models and indexing schemes are required to maintain scientific data and make scientific data available on demand.

The metric-based scientific data model presented in this thesis is designed to address the modeling need of a wide range of scientific data at the conceptual level. While DDL and DML based on the proposed model are yet to be developed, *ad hoc* metric formulations suffice as implementation models for many applications. A new class of hierarchical data structures is proposed to provide direct support of metric-modeled scientific data at the physical level. These data structures enable efficient indexing based on proximity queries.

7.2 Contributions

This thesis presents a complete framework for scientific data management and knowledge exploration based on current database technologies. This section summarizes its major contributions.

In order to provide motivation and perspective, we start with a survey of existing scientific data models. Three implementation primitives, geometry, topology, and indexable topology, are defined to establish a coherent theoretical foundation for studying and comparing the existing models.

At the conceptual level, we present a metric-based model developed from two fundamental notions, data-as-functions and pseudo-quasimetrics, which are used to model the inter-entity and inter-instance relationships respectively. In addition, a detailed approach for metric derivation based on metric theory and general topology is developed. The approach itself is useful as a paradigm for knowledge discovery from the metric perspective.

Thanks to the formal mathematical semantics of the metric-based model, the models and metrics derived can be formally validated. The notion of continuity is used as a precise mathematical tool for validating the results of the metric derivation process, for both data modeling and knowledge discovery purposes.

Compared to the spatial data models and existing scientific data models arising in computational fluid dynamics and scientific visualization, the metric-based model offers more flexibility and generality. In particular, the process for metric derivation based on observable properties can be very valuable for data mining in categorical data. From the knowledge discovery perspective, we believe the metric-based data model has tremendous potential as the foundation for developing various data mining mechanisms.

At the physical level, we have developed an innovative approach, the multipolar mapping, which enables us to utilize existing point spatial data structures for storage and indexing of metric data with pseudo-metrics. To extend this approach to pseudo-quasimetrics, a simple method, the median transformation, was developed to derive a pseudo-metric from a pseudo-quasimetric. Using the median transformation, all existing hierarchical metric data structures can also be used for metric data with pseudo-quasimetrics.

A series of experiments was carried out to measure the performance of multipolar mappings under various circumstances based on a restricted performance model which takes into account only the number of distance evaluations required for a proximity query.

The results demonstrated good performance for both the nearest neighbor and proximity queries on clustered data of intrinsic dimensionality $m \leq 10$. It is also evident that the multipolar approach is very robust and predictable with regard to many parameters of the mapping itself. Simple guidelines are available for fine tuning major parameters of multipolar mappings based on characteristics of the metric data set in question.

Limited experiments on clustered data of $m = 100$ showed a much more pronounced curse of dimensionality for proximity queries. However, the performance of nearest neighbor queries only suffers slightly compared to the performance on clustered data of $m = 10$ and otherwise identical parameters. Note that the curse of dimensionality also exerts significant performance penalty on all existing metric data structures. While it is still unclear how much performance edge the multipolar approach is able to maintain for proximity queries on data sets with very high intrinsic dimensionalities, based on the preliminary results, we are fairly confident that multipolar approach will easily outperform existing hierarchical data structures on nearest neighbor queries on a very wide range of data.

7.3 Future Research

Future research on the metric-based model can be done at all three abstraction levels. At the conceptual level, we are most interested in the extension of continuity and progressive refinements of topologies. The general notion of continuity can be further extended to incorporate probability. For instance, a data function can be continuous with respect to a particular domain geometry with a 90% probability. By extending the semantics of continuity, less-than-perfect or fuzzy interrelationships in data can also be modeled and validated. Complex interrelationships in some nonmetric data require a large set of complex observable properties to model. Property topologization can be computationally expensive in this situation. A paradigm based on progressively refined topologies can alleviate such problem. Thus, a topology can "grow" as the users identify more observable properties in the data.

From the knowledge discovery perspective, new mechanisms can be developed based on the metric derivation paradigm. Since various properties of distance functions will play important roles in developing metric data structures and data mining mechanisms, it will prove to be beneficial to identify a larger set of metric axioms and study their semantics

in the context of data management and analysis.

At the implementation level, the major tasks in the future include the development of a formal DDL and DML. In particular, we are interested in studying the formulation and implementation of pseudo-quasimetrics and proximity queries using existing implementation models such as the object-oriented model.

At the physical level, we plan to validate the three conjectures proposed in Chapter 6 either analytically or empirically with a wider range of experiments:

1. nearest neighbor queries have a constant time complexity.
2. proximity queries have a linear time complexity, and
3. only a small number of poles, with respect to intrinsic dimensionality, is required for good performance on clustered data.

We also plan to extend the performance model to include implementation parameters such that the performance of the proposed multipolar-based approach and existing hierarchical data structures can be compared and analyzed.

In addition to median transformation, we would like to develop new approaches which can enable the application of pseudo-metric based data structures to pseudo-quasimetrics. A particularly promising direction involves using two pseudo-metrics to simulate the proximity of a pseudo-quasimetric. If this is possible, a metric data set based on pseudo-quasimetrics can be stored as two instances of a pseudo-metric based data structure, and the proximity query on the pseudo-quasimetric can be carried out by a set operation on the results of two proximity queries on the two instances of data structures.

Bibliography

[1] Alan Agresti. *An Introduction to Categorical Data Analysis.* John Wiley and Sons. New York. New York. 1996.

[2] R. Daniel Bergeron and Georges G. Grinstein. A Reference Model for the Visualization of Multi-Dimensional Data. In W. Hansmann. F. R. A. Hopgood. and W. Strasser. editors. *Eurographics '89. Proceeding of the European Computer Graphics Conference and Exhibition.* pages 393–399. Hamburg. F.R.G.. September 1989. North-Holland Publishing Company. Amsterdam. Netherlands.

[3] Tolga Bozkaya and Meral Ozsoyoglu. Distance-Based Indexing for High-Dimensional Metric Spaces. In Joan M. Peckman. editor. *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data. Tucson. Arizona.* SIGMOD Record. 26(2). pages 357–368. ACM Press. June 1997.

[4] Peter Buneman and Robert E. Frankel. FQL – A Functional Query Language. In Philip A. Bernstein. editor. *Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data.* pages 52–58. Boston. Massachusetts. May 1979.

[5] George L. Cain. *Introduction to General Topology.* Addison-Wesley. Reading. Massachusetts. 1994.

[6] Ming-Syan Chen. Jiawei Han. and Philip S. Yu. Data Mining: An Overview from a Database Perspective. *IEEE Transactions on Knowledge and Data Engineering.* 8(6):866–883. December 1996.

[7] Peter Pin-Shan Chen. The Entity-Relationship Model: Toward a Unified View of Data. *ACM Transactions on Database Systems.* 1(1):9–36. January 1976.

[8] Andrew L. Comrey and Howard B. Lee. *A First Course in Factor Analysis.* Lawrence Erlbaum Associates. Hillsdale. New Jersey. 2nd edition. 1992.

[9] Trevor F. Cox and Michael A.A. Cox. *Multidimensional Scaling.* Chapman & Hall. London. United Kingdom. 1994.

[10] Michael J. Cullinane. David T. Kao. and Samuel D. Shore. Knowledge Acquisition and Representation through Neighborhoods. In Roland R. Wagner and Helmut Thoma. editors. *Seventh International Workshop on Database and Expert Systems Applications. DEXA '96. Proceedings.* pages 49–55. Zurich. Switzerland. September 1996. IEEE Computer Society Press. Los Alamitos. California.

[11] C. J. Date. *An Introduction to Database Systems.* volume 1. Addison-Wesley. Reading. Massachusetts. 5th edition. 1990.

[12] C. J. Date. *An Introduction to Database Systems.* Addison-Wesley. Reading. Massachusetts. 6th edition. 1995.

[13] Ingrid Daubechies. *Ten Lectures on Wavelets.* Number 61 in CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics. Philadelphia. Pennsylvania. 1992.

[14] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order.* Cambridge University Press. Cambridge. United Kingdom. 1990.

[15] DEC. *DEC AVS: User's Guide for Ultrix Systems.* May 1992.

[16] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems.* Benjamin/Cummings Publishing Company. Redwood City. California. 2nd edition. 1994.

[17] Gordon Erlebacher. M. Yousuff Hussaini. and Leland M. Jameson. *Wavelets: Theory and Applications.* Oxford University Press. New York. New York. 1996.

[18] Usama Fayyad. Gregory Piatetsky-Shapiro. and Padhraic Smyth. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of ACM.* 39(11):27–34. November 1996.

[19] Usama M. Fayyad. Gregory Piatetsky-Shapiro. Padhraic Smyth. and Ramasamy Uthurusamy. editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press / The MIT Press. Cambridge. Massachusetts. 1996.

[20] Bernhard Flury. *Common Principal Components and Related Multivariate Models.* John Wiley and Sons. New York. New York. 1988.

[21] E. Fredkin. Trie Memory. *CACM.* 3(9):490–499. 1960.

[22] King Sun Fu. *Syntactic Pattern Recognition and Applications.* Prentice-Hall. Englewood Cliffs. New Jersey. 1982.

[23] Larry Gelberg. David Kamins. David Parker. and Jonathan Sacks. Visualization Techniques for Structured and Unstructured Scientific Data. In *State of the Art in Data Visualization. SIGGRAPH '90 Course Notes #27.* Dallas. Texas. August 1990.

[24] Martin Gogolla. *An Extended Entity-Relationship Model. Fundamentals and Pragmatics.* Number 767 in Lecture Notes in Computer Science. Springer-Verlag. Berlin / Heidelberg. Germany. 1994.

[25] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures: In Pascal and C.* Addison-Wesley. Reading. Massachusetts. 2 edition. 1991.

[26] Gregory W. Goucher and G. Jason Mathews. A Comprehensive Look at CDF. Technical Report NSSDC/WDC-A-R&S 94-07. NASA/Goddard Space Flight Center. Greenbelt. Maryland. August 1994.

[27] Peter M. D. Gray. Krishnarao G. Kulkarni. and Norman W. Paton. *Object-Oriented Databases: A Semantic Data Model Approach.* Prentice-Hall. Englewood Cliffs. New Jersey. 1992.

[28] R. B. Haber. B. Lucas. and N. Collins. A Data Model for Scientific Visualization with Provisions for Regular and Irregular Grids. In Gregory M. Nielson and Larry Rosenblum. editors. *Visualization '91. Proceedings.* pages 298–305. San Diego. California. October 1991. IEEE Computer Society Press. Los Alamitos. California.

[29] Joseph F. Hair. Jr.. Rolph E. Anderson. Ronald L. Tatham. and William C. Black. *Multivariate Data Analysis: with Readings.* Prentice-Hall. Englewood Cliffs. New Jersey. 4th edition. 1995.

[30] William L. Hibbard. Charles R. Dyer. and Brian E. Paul. A Lattice Model for Data Display. In R. Daniel Bergeron and Arie E. Kaufman. editors. *Visualization '94. Proceedings.* pages 310–317. Washington. D.C.. October 1994. IEEE Computer Society Press. Los Alamitos. California.

[31] William Louis Hibbard. *Visualizing Scientific Computations: A System Based on Lattice-Structured Data and Display Models.* PhD thesis. Department of Computer Science. University of Wisconsin. Madison. 1995. Technical Report 1226.

[32] A. Kantorovich. *Scientific Discovery - Logic and Tinkering.* State University of New York Press. Albany. New York. 1993.

[33] David T. Kao. Topology as Logic - An Introduction. Technical Report 95-24. Department of Computer Science. University of New Hampshire. 1996.

[34] David T. Kao. R. Daniel Bergeron. Michael J. Cullinane. and Ted M. Sparr. Semantics and Mathematics of Scientific Data Sampling. In Andreas Wierse. Georges G. Grinstein. and Ulrich Lang. editors. *Database Issues for Data Visualization. IEEE Visualization '95 Workshop. Proceedings.* number 1183 in Lecture Notes in Computer Science. pages 86–100. Atlanta. Georgia. October 1995. Springer-Verlag. Berlin / Heidelberg. Germany. 1996.

[35] David T. Kao. R. Daniel Bergeron. and Ted M. Sparr. An Extended Schema Model for Scientific Data. In John P. Lee and Georges G. Grinstein. editors. *Database Issues for Data Visualization. IEEE Visualization '93 Workshop. Proceedings.* number 871 in Lecture Notes in Computer Science. pages 69–82. San Jose. California. October 1993. Springer-Verlag. Berlin / Heidelberg. Germany. 1994.

[36] David T. Kao. R. Daniel Bergeron. and Ted M. Sparr. A Topological Approach to Derive Metrics for Discrete Data. Technical Report 95-20. Department of Computer Science. University of New Hampshire. 1995.

[37] David T. Kao. R. Daniel Bergeron. and Ted M. Sparr. Geometries and Storage Topologies of Scientific Data. Technical Report 95-21. Department of Computer Science. University of New Hampshire. 1995.

[38] David T. Kao. R. Daniel Bergeron. and Ted M. Sparr. Mapping Metric Data to Multidimensional Space. Technical Report TR 97-02. Department of Computer Science. University of New Hampshire. 1997.

[39] Norio Katayama and Shin'ichi Satoh. The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries. In Joan M. Peckman. editor. *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data. Tucson. Arizona.* SIGMOD Record. 26(2). pages 369–380. ACM Press. June 1997.

[40] P. Knupp and S. Steinberg. *Fundamentals of Grid Generation.* CRC Press. Boca Raton. Florida. 1994.

[41] Donald E. Knuth. *The Art of Computer Programming. Volume 3: Sorting and Searching.* Addison-Wesley. Reading. Massachusetts. 1973.

[42] R. Kopperman. All Topologies Come from Generalized Metrics. *American Mathematical Monthly.* 95(2):89–97. February 1988.

[43] H. F. Korth and A. Silberschatz. *Database System Concepts.* McGraw-Hill. 2nd edition. 1991.

[44] J. Munkres. *Topology: A First Course.* Prentice-Hall. Englewood Cliffs. New Jersey. 1975.

[45] NASA/Goddard Space Flight Center. Greenbelt. Maryland. *CDF User's Guide.* 2.5 edition. January 1995.

[46] NCSA (National Center for Supercomputer Applications). University of Illinois. Champaign-Urbana. Illinois. *NCSA DataScope Reference Manual.* 1989.

[47] J. Nievergelt. H. Hinterberger. and K. C. Sevcik. The Grid File: An Adaptable. Symmetric Multikey File Structure. *ACM Transactions on Database Systems.* 9(1):38 71. March 1984.

[48] Davood Rafiei and Alberto Mendelzon. Similarity-Based Queries for Time Series Data. In Joan M. Peckman. editor. *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data. Tucson. Arizona.* SIGMOD Record. 26(2). pages 13–25. ACM Press. June 1997.

[49] R. K. Rew and G. P. Davis. NetCDF: An Interface for Scientific Data Access. *IEEE Computer Graphics and Applications.* 10(4). July 1990.

[50] Walter Rudin. *Principles of Mathematical Analysis.* McGraw-Hill. New York. New York. 3rd edition. 1976.

[51] Hanan Samet. *Applications of Spatial Data Structures: Computer Graphics. Image Processing. and GIS.* Addison-Wesley. Reading. Massachusetts. 1990.

[52] Hanan Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley. Reading. Massachusetts. 1990.

[53] Hanan Samet. Spatial Data Structures. In Won Kim, editor. *Modern Database Systems*, pages 361–385. ACM Press, New York, New York, 1995.

[54] Hanan Samet and Walid G. Aref. Spatial Data Models and Query Processing. In Won Kim, editor. *Modern Database Systems*, pages 338–360. ACM Press, New York, New York, 1995.

[55] David W. Shipman. The Functional Data Model and Data Language DAPLEX. *ACM Transactions on Database Systems*, 6(1):140–173, March 1981.

[56] Edgar H. Sibley and Larry Kerschberg. Data Architecture and Data Model Considerations. In *Proceedings of the AFIPS National Computer Conference*, pages 85–96, Dallas, Texas, June 1977. American Federation of Information Processing Societies.

[57] John Miles Smith and Diane C. P. Smith. Database Abstractions: Aggregation and Generalization. *ACM Transactions on Database Systems*, 2(2):105–133, June 1977.

[58] M. B. Smyth. Topology. In S. Abramsky, Dov M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science, Volume 1, Background: Mathematical Structures*, pages 641–761. Oxford University Press, Oxford, United Kingdom, 1992.

[59] M. B. Smyth. Semi-metrics, Closure Spaces and Digital Topology. *Theoretical Computer Science*, 151(1):257–276, November 1995.

[60] D. Speray and S. Kennon. Volume Probes: Interactive Data Exploration on Arbitrary Grids. *Computer Graphics*, 24(5), November 1991.

[61] Toby J. Teorey, Dongqing Yang, and James P. Fry. A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model. *ACM Computing Surveys*, 18(2):197–222, June 1986.

[62] Bernhard Thalheim. Extending the Entity-Relationship Model for a High-Level, Theory-Based Database Design. In Joachim W. Schmidt and Anatoly A. Stogny, editors, *Next Generation Information System Technology, First International East/West Database Workshop, Proceedings*, number 504 in Lecture Notes in Computer Science, pages 161–184, Kiev, USSR, October 1990. Springer-Verlag, Berlin / Heidelberg, Germany, 1991.

[63] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, Reading, Massachusetts, 1977.

[64] Jeffrey K. Uhlmann. Implementing Metric Trees to Satisfy General Proximity/-Similarity Queries. Technical report, Information Technology Division, Naval Research Laboratory, Washington, D.C., November 1991.

[65] Jeffrey K. Uhlmann. Satisfying General Proximity/Similarity Queries with Metric Trees. *Information Processing Letters*, 40(4):175–179, November 1991.

[66] James S. Walker. *Fast Fourier Transforms*. CRC Press, Boca Raton, Florida, 1991.

[67] Peter N. Yianilos. Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 311–321. Austin, Texas, January 1993. ACM, New York, New York.

[68] Pavel Zezula, Paolo Ciaccia, and Fausto Rabitti. M-tree: A Dynamic Index for Similarity Queries in Multimedia Databases. Technical Report 7, HERMES, ESPRIT LTR, 1996.

Appendix A

Mathematical Definitions

This appendix contains several technical definitions not presented in the thesis.

Definition A.1 (Manifold) *An* n-dimensional manifold M *is a set furnished with a collection* $\mathcal{P}$ *of* abstract patches *(one-to-one functions* $x : D \to M$*.* D *an open set in* E^n*) such that*

1. M *is covered by the images of the patches in the collection* $\mathcal{P}$*.*
2. *For any two patches* x *and* y *in the collection* $\mathcal{P}$*, the composite function* $y^{-1}x$ *and* $x^{-1}y$ *are Euclidean-differentiable (and defined on open sets in* E^n*).*

Definition A.2 (Bundle) *A* bundle *is a triple* $(E.p.B)$*, where* $p : E \to B$ *is a map. The space* B *is called the* base space*, the space* E *is called the* total space*, and the map* p *is called the* projection *of the bundle. For each* $b \in B$*, the space* $p^{-1}(b)$ *is called the fiber of the bundle over* b*.*

Definition A.3 (Bundle Section) *A* cross section *of a bundle* $(E.p.B)$ *is a map* $s : B \to E$ *such that* $ps = 1_B$*. In other words, a cross section is a map* $s : B \to E$ *such that* $s(b) \in p^{-1}(b)$*, the fiber over* b*, for each* $b \in B$*.*

Appendix B

Theorem Proofs

This appendix presents theorems and proofs which constitute the theoretical foundation for multipolar mappings and median transformations.

Theorem B.1 *If (X, d) is a pseudo-metric space. $M_{k,P} : (X, d) \longrightarrow (\mathbb{R}^k, E_k)$ is continuous. $\forall M_{k,P} \in \mathcal{M}(X)$.*

Proof: Since d is a pseudo-metric. $\forall p, q, r \in X$. we have

$$\begin{aligned} & d(r,p) \leq d(r,q) + d(q,p). \\ & d(r,q) \leq d(r,p) + d(p,q) \\ \Longrightarrow \quad & d(r,p) - d(r,q) \leq d(q,p). \\ & d(r,q) - d(r,p) \leq d(p,q) \\ \Longrightarrow \quad & -d(p,q) \leq d(r,p) - d(r,q) \leq d(p,q) \\ \Longrightarrow \quad & |d(r,p) - d(r,q)| \leq d(p,q) \end{aligned}$$

Let x be an arbitrary point in X and ε an arbitrary positive real number. For $y \in S_d(x, \frac{\varepsilon}{\sqrt{k}})$. we have $d(x,y) < \frac{\varepsilon}{\sqrt{k}}$.

$$\begin{aligned} E_k(M_k(x), M_k(y)) &= \left((d(p_1,x) - d(p_1,y))^2 + \cdots + (d(p_k,x) - d(p_k,y))^2\right)^{\frac{1}{2}} \\ &\leq \left(d(x,y)^2 + \cdots + d(x,y)^2\right)^{\frac{1}{2}} \\ &= \sqrt{k}\, d(x,y) \\ &< \sqrt{k}\, \frac{\varepsilon}{\sqrt{k}} \\ &= \varepsilon \end{aligned}$$

Thus, $M_k(y) \in S_{E_k}(M_k(x), \varepsilon)$ and $M_k[S_d(x, \frac{\varepsilon}{\sqrt{k}})] \subseteq S_{E_k}(M_k(x), \varepsilon)$. Since x is an arbitrary point of X. M_k is continuous everywhere in X. ∎

Corollary B.2 follows directly from the proof of Theorem B.1.

Corollary B.2 *Let $r \in \mathbb{R}$ and (X, d) be a pseudo-metric space. $\forall M_{k,P} \in \mathcal{M}(X)$. we have*

$$M_{k,P}[S_d(x, r)] \subseteq S_{E_k}(M_{k,P}(x), \sqrt{k}\, r).$$

Theorem B.3 *If (X, d) is a pseudo-metric space, $M_{k,P} : (X, d) \longrightarrow (\mathbb{R}^k, m_k)$ is continuous, $\forall M_{k,P} \in \mathcal{M}(X)$.*

Proof: The proof is similar to the one of Theorem B.1. Let x be an arbitrary point in X and ε an arbitrary positive real number. For $y \in S_d(x, \varepsilon)$, we have $d(x, y) < \varepsilon$.

$$\begin{aligned} m_k(M_k(x), M_k(y)) &= \max\{|d(p_1, x) - d(p_1, y)|, \ldots, |d(p_k, x) - d(p_k, y)|\} \\ &\leq \max\{d(x, y), \ldots, d(x, y)\} \\ &= d(x, y) \\ &< \varepsilon \end{aligned}$$

Thus, $M_k(y) \in S_{m_k}(M_k(x), \varepsilon)$ and $M_k[S_d(x, \varepsilon)] \subseteq S_{m_k}(M_k(x), \varepsilon)$. Since x is an arbitrary point of X, M_k is continuous everywhere in X. ∎

Corollary B.4 follows directly from the proof of Theorem B.3.

Corollary B.4 *Let $r \in \mathbb{R}$ and (X, d) be a pseudo-metric space, $\forall M_{k,P} \in \mathcal{M}(X)$, we have*

$$M_{k,P}[S_d(x, r)] \subseteq S_{m_k}(M_{k,P}(x), r).$$

Theorem B.5 *Let $x \in \mathbb{R}^k$ and $r \in \mathbb{R}$.*

$$S_{m_k}(x, r) \subseteq S_{E_k}(x, \sqrt{k}\, r), \forall k \in N.$$

Proof: For $y \in S_{m_k}(x, r)$, we have $m_k(x, y) < r$. Let $x = (x_1, \ldots, x_k)$ and $y = (y_1, \ldots, y_k)$. Suppose that $h \in \{1, \ldots, k\}$ and $|x_i - y_i| \leq |x_h - y_h|, \forall i \in \{1, \ldots, k\}$.

$$\begin{aligned} E_k(x, y) &= \left((x_1 - y_1)^2 + \cdots + (x_k - y_k)^2\right)^{\frac{1}{2}} \\ &\leq \left((x_h - y_h)^2 + \cdots + (x_h - y_h)^2\right)^{\frac{1}{2}} \\ &= \left(k(x_h - y_h)^2\right)^{\frac{1}{2}} \\ &= \sqrt{k}\, |x_h - y_h| \\ &= \sqrt{k}\, m_k(x, y) \\ &< \sqrt{k}\, r \end{aligned}$$

Thus, $y \in S_{E_k}(x, \sqrt{k}\, r)$. ∎

Theorem B.6 *Let (X, d) be a pseudo-metric space, $P_1, P_2 \subseteq X$, $k_1 = |P_1|$, $k_2 = |P_2|$, $x \in X$, and $r \in \mathbb{R}$. If $P_1 \subseteq P_2$, then*

$$\begin{aligned} & M_{k_2,P_2}(y) \in S_{m_{k_2}}(M_{k_2,P_2}(x), r) \\ \Longrightarrow\ & M_{k_1,P_1}(y) \in S_{m_{k_1}}(M_{k_1,P_1}(x), r), \forall y \in X. \end{aligned}$$

Proof: Let $y \in X$. $P_1 = \{p_1, \ldots, p_{k_1}\}$. and $P_2 = \{p_1, \ldots, p_{k_2}\}$. For $M_{k_2,P_2}(y) \in S_{m_{k_2}}(M_{k_2,P_2}(x), r)$. we have $m_{k_2}(M_{k_2,P_2}(x), M_{k_2,P_2}(y)) < r$.

$$\begin{aligned} & m_{k_1}(M_{k_1,P_1}(x), M_{k_1,P_1}(y)) \\ = \; & \max\{|d(p_1, x) - d(p_1, y)|, \ldots, |d(p_{k_1}, x) - d(p_{k_1}, y)|\} \\ \leq \; & \max\{|d(p_1, x) - d(p_1, y)|, \ldots, |d(p_{k_2}, x) - d(p_{k_2}, y)|\} \\ = \; & m_{k_2}(M_{k_2,P_2}(x), M_{k_2,P_2}(y)) \\ < \; & r \end{aligned}$$

Thus. $M_{k_1,P_1}(y) \in S_{m_{k_1}}(M_{k_1,P_1}(x), r)$. ∎

Theorem B.7 *If d is a pseudo-quasimetric. its median d^- is a pseudo-metric.*

Proof:

$$\begin{aligned} (1) \quad & d^-(p, p) = \frac{1}{2}(d(p, p) + d(p, p)) = 0 \\ (2) \quad & d(p, q) \leq d(p, s) + d(s, q). \\ & d(q, p) \leq d(q, s) + d(s, p) \\ \Longrightarrow \quad & d(p, q) + d(q, p) \leq (d(p, s) + d(s, p)) + (d(s, q) + d(q, s)) \\ \Longrightarrow \quad & \frac{1}{2}(d(p, q) + d(q, p)) \leq \frac{1}{2}(d(p, s) + d(s, p)) + \frac{1}{2}(d(s, q) + d(q, s)) \\ \Longrightarrow \quad & d^+(p, q) \leq d^+(p, s) + d^+(s, q) \\ (3) \quad & d^+(p, q) = \frac{1}{2}(d(p, q) + d(q, p)) = d^-(q, p) \end{aligned}$$

d^- satisfies all three axioms of pseudo-metrics. ∎

Theorem B.8 *Let (X, d) be a pseudo-quasimetric space. $r \in \mathbb{R}$. and $d(p, q) - d(q, p) \leq \alpha. \forall p, q \in X$.*

$$S_d(x, r) \subseteq S_{d^+}(x, r + \frac{\alpha}{2}). \quad \forall x \in X.$$

Proof: Let x be an arbitrary point in X . and $y \in S_d(x, r)$. From $d(p, q) - d(q, p) \leq \alpha. \forall p, q \in X$. we have $d(y, x) \leq d(x, y) + \alpha$.

$$\begin{aligned} d^-(x, y) &= \frac{1}{2}(d(x, y) + d(y, x)) \\ &\leq \frac{1}{2}(2d(x, y) + \alpha) \\ &< \frac{1}{2}(2r + \alpha) \\ &= r + \frac{\alpha}{2} \end{aligned}$$

Thus. $y \in S_{d^+}(x, r + \frac{\alpha}{2})$ and $S_d(x, r) \subseteq S_{d^+}(x, r + \frac{\alpha}{2}). \forall x \in X$. ∎

Corollary B.9 follows directly from Theorem B.8 and Corollary B.2.

Corollary B.9 *Let (X, d) be a pseudo-quasimetric space, $r \in \mathbb{R}$, and $d(p,q) - d(q,p) \leq \alpha, \forall p, q \in X$, $\forall M_{k,P} \in \mathcal{M}_{d^+}(X)$, where $\mathcal{M}_{d^+}(X)$ represents the set of all multipolar mappings on (X, d^+), we have*

$$M_{k,P}[S_d(x,r)] \subseteq S_{E_k}(M_{k,P}(x), \sqrt{k}(r + \frac{\alpha}{2})).$$