Algorithms for Clustering Data

Algorithms for Clustering Data

Anil K. Jain
Richard C. Dubes

Michigan State University

Prentice Hall
Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging-in-Publication Data

JAIN, ANIL K., 1948–
Algorithms for clustering data / Anil K. Jain, Richard C. Dubes.

p. cm. — (Prentice Hall advanced reference series)
Bibliography: p.
Includes index.
ISBN 0–13–022278–X
1. Cluster analysis—Data processing. 2. Algorithms. I. Dubes,
Richard C. II. Title.
QA278.J35 1988
519.5'3—dc19

Editorial/production supervision
and interior design: BARBARA MARTTINE
Cover design: KAREN STEPHENS
Manufacturing buyer: LORRAINE FUMOSO

Prentice Hall Advanced Reference Series

Printed in the United States of America

10 9 8 7 6 5 4 3 2

ISBN 0-13-022278-X

PRENTICE-HALL INTERNATIONAL (UK) LIMITED, *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
PRENTICE-HALL CANADA INC., *Toronto*
PRENTICE-HALL HISPANOAMERICANA, S.A., *Mexico*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
SIMON & SCHUSTER ASIA PTE. LTD., *Singapore*
EDITORA PRENTICE-HALL DO BRASIL, LTDA., *Rio de Janeiro*

TO NANDITA AND MARYLYN

Contents

Preface

Cluster analysis is an important technique in the rapidly growing field known as exploratory data analysis and is being applied in a variety of engineering and scientific disciplines such as biology, psychology, medicine, marketing, computer vision, and remote sensing. Cluster analysis organizes data by abstracting underlying structure either as a grouping of individuals or as a hierarchy of groups. The representation can then be investigated to see if the data group according to preconceived ideas or to suggest new experiments. Cluster analysis is a tool for exploring the structure of the data that does not require the assumptions common to most statistical methods. It is called "unsupervised learning" in the literature of pattern recognition and artificial intelligence.

This book will be useful for those in the scientific community who gather data and seek tools for analyzing and interpreting data. It will be a valuable reference for scientists in a variety of disciplines and can serve as a textbook for a graduate course in exploratory data analysis as well as a supplemental text in courses on research methodology, pattern recognition, image processing, and remote sensing. The book emphasizes informal algorithms for clustering data, and interpreting results. Graphical procedures and other tools for visually representing data are introduced both to evaluate the results of clustering and to explore data. Mathematical and statistical theory are introduced only when necessary.

Most existing books on cluster analysis are written by mathematicians, numerical taxonomists, social scientists, and psychologists who emphasize either the methods that lend themselves to mathematical treatment or the applications in their particular area. Our book strives for a sense of completeness and for a balanced presentation. We bring together many results that are scattered through the literature of several fields. The most unique feature of this book is its thorough, understandable treatment of cluster validity, or the objective validation of the results of cluster analysis, from the application viewpoint.

This book resulted from class notes that the authors have used in a graduate course on clustering and scaling algorithms in the Department of Computer Science at Michigan State University. We owe a debt of gratitude to the many graduate students who have helped develop this book. Special thanks are due to Karl Pettis, Tom Bailey, Neal Wyse, Steve Smith, Gautam Biswas, George Cross, James Coggins, Phil Nagan, Rick Hoffman, Xiaobo Li, Pat Flynn, and C. C. Chen. The prerequisite for this course is probability theory, matrix algebra, computer programming, and data structures. In addition to homework problems and an exam, the students in this course work on a project which can range from the analysis of a real data set to comparative analysis of various algorithms. This course is particularly useful for students who wish to pursue research in pattern recognition, image processing, and artificial intelligence. Interested readers may contact the authors for homework problems for this course.

We have a long-standing interest in cluster analysis, especially in the problems of cluster validity and cluster tendency. Our research in this area has been funded by the National Science Foundation. We are grateful to NSF for this support. We also wish to acknowledge the support and the facilities provided by the Department of Computer Science, Michigan State University, which were essential for the completion of this book.

A. K. Jain
R. C. Dubes

1

Introduction

The practice of classifying objects according to perceived similarities is the basis for much of science. Organizing data into sensible groupings is one of the most fundamental modes of understanding and learning. Cluster analysis is the formal study of algorithms and methods for grouping, or classifying, objects. An object is described either by a set of measurements or by relationships between the object and other objects. Cluster analysis does not use category labels that tag objects with prior identifiers. The absence of category labels distinguishes cluster analysis from discriminant analysis (and pattern recognition and decision analysis). The objective of cluster analysis is simply to find a convenient and valid organization of the data, not to establish rules for separating future data into categories. Clustering algorithms are geared toward finding structure in the data.

A cluster is comprised of a number of *similar* objects collected or grouped together. Everitt (1974) documents some of the following definitions of a cluster:

1. "A cluster is a set of entities which are *alike*, and entities from different clusters are not alike."
2. "A cluster is an aggregation of points in the test space such that the *distance* between any two points in the cluster is less than the distance between any point in the cluster and any point not in it."
3. "Clusters may be described as connected regions of a multi-dimensional space containing a relatively *high density* of points, separated from other such regions by a region containing a relatively low density of points."

The last two definitions assume that the objects to be clustered are represented as points in the measurement space. We recognize a cluster when we see it in the plane, although it is not clear how we do it. While it is easy to give a functional definition of a cluster, it is very difficult to give an operational definition of a cluster. This is due to the fact that objects can be grouped into clusters with different purposes in mind. Data can reveal clusters of differing "shapes" and "sizes." To compound the problem further, cluster membership can change over time, as is the case with star clusters (Dewdney, 1986), and the number of clusters often depends on the resolution (fine versus coarse) with which we view the data. Figure 1.1 illustrates some of these concepts for two-dimensional point clusters. How many clusters are there in Figure 1.1? At the global or higher level of similarity, we perceive four clusters in these data, but at the local level or a

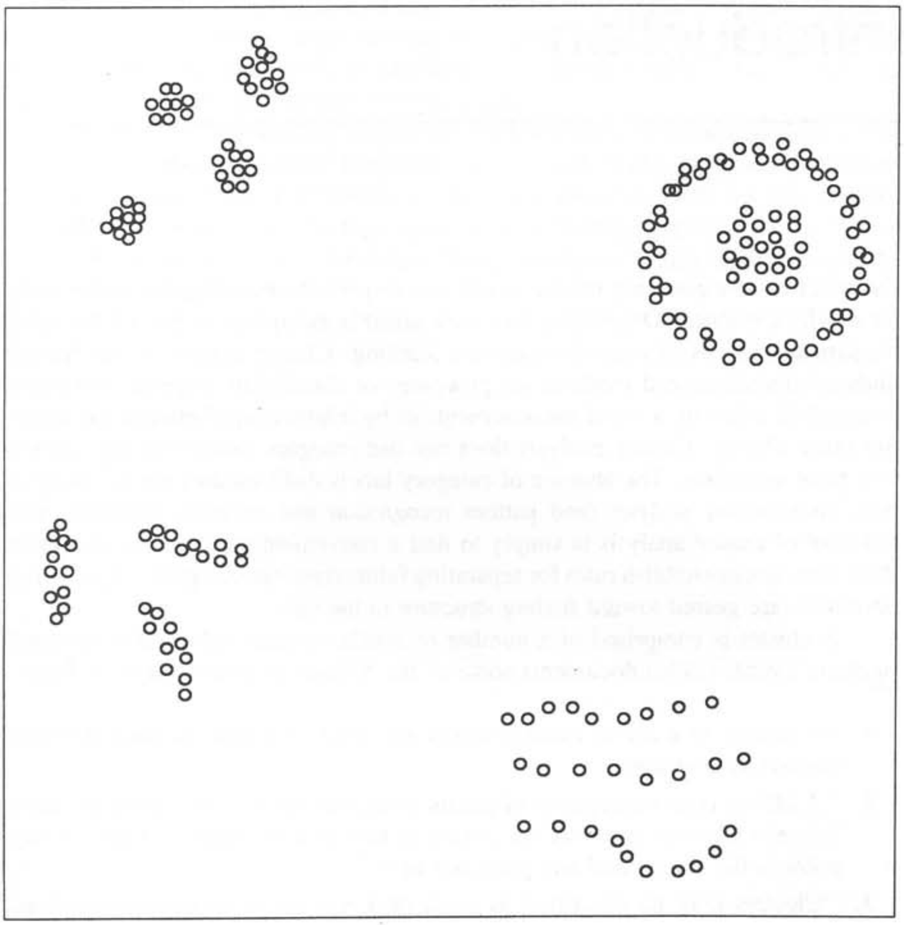


Figure 1.1 Clusters of point patterns in two dimensions.

lower similarity threshold, we perceive ~~nine~~ 12 clusters. Which answer is correct? Looking at the data at multiple scales may actually help in analyzing its structure. Thus the crucial problem in identifying clusters in data is to specify what proximity is and how to measure it. As is to be expected, the notion of proximity is problem dependent.

Clustering techniques offer several advantages over a manual grouping process. First, a clustering program can apply a specified objective criterion consistently to form the groups. Human beings are excellent cluster seekers in two and often in three dimensions, but different individuals do not always identify the same clusters in data. The proximity measure defining similarity among objects depends on an individual's educational and cultural background. Thus it is quite common for different human subjects to form different groups in the same data, especially when the groups are not well separated. Second, a clustering algorithm can form the groups in a fraction of time required by a manual grouping, particularly if a long list of descriptors or features is associated with each object. The speed, reliability, and consistency of a clustering algorithm in organizing data together constitute an overwhelming reason to use it. A clustering algorithm relieves a scientist or data analyst of the treacherous job of ''looking'' at a pattern matrix or a similarity matrix to detect clusters. A data analyst's time is better spent in analyzing or interpreting the results provided by a clustering algorithm.

Clustering is also useful in implementing the ''divide and conquer'' strategy to reduce the computational complexity of various decision-making algorithms in pattern recognition. For example, the nearest-neighbor decision rule is a popular technique in pattern recognition (Duda and Hart, 1973). However, finding the nearest neighbor of a test pattern can be very time consuming if the number of training patterns or prototypes is large. Fukunaga and Narendra (1975) used the well-known partitional clustering algorithm, ISODATA (Chapter 3), to decompose the patterns, and then in conjunction with the branch-and-bound method obtained an efficient algorithm to compute nearest neighbors. Similarly, Fukunaga and Short (1978) used clustering for problem localization, whereby a simple decision rule can be implemented in local regions or clusters of the pattern space. The applications of clustering continue to grow.

Consider the problem of grouping various colleges and universities in the United States to illustrate the factors in clustering problems. Schools can be clustered based on their geographical location, size of the student body, size of the campus, tuition fee, or offerings of various professional graduate programs. The factors depend on the goal of the analysis. The shapes and sizes of the clusters formed will depend on which particular attribute is used in defining the similarity between colleges. Interesting and challenging clustering problems arise when several attributes are taken together to construct clusters. One cluster could represent private, midwestern, and primarily liberal arts colleges with fewer than 1000 students and another can represent large state universities. The features or attributes that we have mentioned so far can easily be measured. What about such attributes as quality of education, quality of faculty, and the quality of campus life, which cannot be measured easily? One can poll alumni or a panel of experts to get either a numerical score (on a scale of, say, 1 to 10) for these factors or similarity measures for all pairs of universities. These scores or similarities must be averaged over all respondents because individual opinions differ. One can also measure subjective attributes indirectly. For example, faculty excellence in a graduate program can be estimated from the number of professional papers written and number of Ph.D. degrees awarded.

The example above illustrates the difference between decision making and clustering. Suppose that we want to partition computer science graduate programs in the United States into two categories based on such attributes as size of faculty, computing resources, external research support, and faculty publications. In the decision-making paradigm, an "expert" must first define these two categories by identifying *some* computer science programs from each of the two categories (these are the training samples in pattern recognition terminology). The attributes of these training samples will be used to construct decision boundaries (or simply thresholds on attribute values) that will separate the two types of programs. Once the decision boundary is available, the remaining computer science programs (those that were not labeled by the expert) will be assigned to one of the two categories. In the clustering paradigm, no expert is available to define the categories. The objective is to determine whether a two-category partition of the data, based on the given attributes, is reasonable, and if so, to determine the memberships of the two clusters. This can be achieved by forming similarities between all pairs of computer science graduate programs based on the given attributes and then constructing groups such that the within-group similarities are larger than the between-group similarities.

Cluster analysis is one component of exploratory data analysis, which means sifting through data to make sense out of measurements by whatever means are available. The information gained about a set of data from a cluster analysis should prod one's creativity, suggest new experiments, and provide fresh insight into the subject matter. The modern digital computer makes all this possible.

Cluster analysis is a child of the computer revolution and frees the analyst from time-honored statistical models and procedures conceived when the human brain was aided only by pencil and paper. The development of clustering methodology has been truly interdisciplinary. Researchers in almost every area of science that collects data have contributed, such as taxonomists, psychologists, biologists, statisticians, social scientists, and engineers. I. J. Good (1977) has suggested the new name *botryology* for the discipline of cluster analysis, from the Greek word for a cluster of grapes.

One objective of this book is to encourage communication among disciplines. All too often, the same procedures are developed in different disciplines but are so clothed in the language of the individual disciplines that cross fertilization is severely hindered. A casual scan of the bibliography for this book reveals citations from almost 100 different journals. Only the *Journal of Classification*, a publication of the Classification Society of North America which first appeared in 1984, is devoted to theoretical and practical issues in exploratory data analysis itself. We have tried to provide an understandable but complete exposition of cluster analysis that uses only enough mathematical detail to make the material precise. The space limitation makes it difficult for us to cover every aspect of cluster analysis in great detail. We emphasize informal algorithms for clustering methods, and analysis of results. However, a reader may have difficulty in implementing the algorithms from the description given in Chapter 3. Most of the well-known clustering algorithms have been implemented and are available as part of clustering and statistical software packages (see Section 3.4). We see cluster analysis as a tool to be used, not as a theory to be developed.

In Chapter 2 we present our idea of data with emphasis on ways of viewing data, such as projections based on eigenvectors and multidimensional scaling. Four ways in which data are analyzed that are related to cluster analysis are reviewed so as to clarify the role of cluster analysis. Clustering methods and algorithms themselves are described in Chapter 3. The primary division among clustering methods is between hierarchical and partitional methods. Both approaches are carefully developed and several examples are provided. The availability of clustering software, methodology by which cluster analysis can be applied, and comparative studies of various clustering techniques are also summarized in Chapter 3. A comparative analysis of clustering methods is useful since empirical evidence seems to be the only practical guide to the selection of clustering methods.

The crucial step in applications of cluster analysis is the interpretation of the results. In Chapter 4 we present a comprehensive summary of procedures for quantitatively verifying the results of cluster analysis. Monte Carlo techniques along with the method of bootstrapping are also introduced in Chapter 4, because they are useful for estimating the distributions of various cluster statistics. Applications of clustering to an engineering domain (image processing and computer vision) are discussed in Chapter 5. The book also contains eight appendices to review briefly related topics of pattern recognition, commonly used Gaussian and hypergeometric distributions, linear algebra, scatter matrices, factor analysis, multivariate analysis of variance, and graph theory. An algorithm to generate clustered data is also given in one of the appendices. We hasten to add that these appendices contain only elementary material provided for the convenience of the reader. The reader should consult standard textbooks for detailed coverage of these topics.

This is not the first book on cluster analysis. Anderberg (1973) has written the most comprehensive book for those who want to use cluster analysis. We refer frequently to Anderberg's excellent exposition. Everitt (1974) explains cluster analysis in a very readable way but contains fewer details than we feel are necessary. Tryon and Bailey (1970) wrote one of the first books on cluster analysis, but it is restricted to a single approach. Jardine and Sibson (1971) concentrate on mathematical foundations. Other early books are those of Duran and Odell (1974) and Clifford and Stephenson (1975). Sneath and Sokal (1973) include an excellent chapter on hierarchical clustering. Hartigan (1975) provides a number of interesting projects, and Lorr (1983) presents cluster analysis especially for social scientists.

Van Ryzin (1977) provides a good collection of interesting papers. The book by Gordon (1981) is another excellent reference. As cluster analysis has become an accepted methodology, chapters in books and long summary papers have appeared. Enslein et al. (1977) emphasize computational matters in their chapter on clustering. Cormack (1971) wrote one of the first summaries of cluster analysis. Three papers appear in Volume 2 of *Handbook of Statistics* (Krishnaiah and Kanal, 1982). Lee (1981) provides a long chapter in Volume 8 of *Advances in Information System Science*, and Dubes and Jain (1980) contributed a chapter to Volume 19 of *Advances in Computers*. The chapter by Hawkins et al. (1982) in *Topics in Applied Multivariate Analysis* emphasizes model fitting and tests for number of clusters. Diday and Simon (1976) and Jain (1986) cover clustering algorithms in the context of pattern recognition and image processing. We owe an intellectual debt to all of the precursors of this book.

The treatment in this book is not mathematical but uses mathematical notation and concepts. Chapter 4 requires more mathematical sophistication than do the other chapters. For the clearest understanding of the material, it is best to read the book in the order presented. However, each chapter contains many details that can be omitted on first reading. We have tried to list all applicable current literature and apologize for inadvertently omitting any important papers. We hope that some parts of this book will appeal to all who use cluster analysis and that the book will help those first encountering cluster analysis to get to the important issues in a timely manner.

2

Data Representation

The first prerequisite for a rational application of cluster analysis is an appreciation for the basic factors required to represent the data. Clustering algorithms are matched to data type, and unless factors such as scale, normalization, and types of proximity measures are understood, one can be misled when interpreting the results of a clustering algorithm. The first three sections discuss these issues and define many of the terms used throughout the book.

Cluster analysis is a tool for exploring data and must be supplemented by techniques for visualizing data. The most direct visualization is a two-dimensional plot showing the objects to be clustered as points. Multivariate data cannot always be faithfully reproduced in two dimensions but when valid, such a representation is helpful in verifying the results of a clustering algorithm. Sections 2.4 and 2.5 cover some common linear and nonlinear techniques for projecting and representing multivariate data. Some mathematical and statistical concepts needed to understand linear projections are summarized in Appendices B, C, and D. The equations derived to explain linear projections in Section 2.4 bear some resemblance to equations from multivariate statistics. Thus Appendices E and F review two related topics from statistical data analysis: factor analysis, and multivariate analysis of variance.

The intrinsic, or topological, dimensionality of a data set dictates the smallest number of factors needed to represent the data. Ways of estimating intrinsic dimensionality are reviewed in Section 2.6. Multidimensional scaling, the topic of Section 2.7, is a procedure for representing qualitative data in a space of few dimensions and supplements cluster analysis.

2.1 DATA TYPES AND DATA SCALES

Clustering algorithms group objects, or data items, based on indices of proximity between pairs of objects. The objects themselves have been called individuals, cases, subjects, and OTUs (operational taxonomic units) in various applications. This book uses pattern recognition terminology (Appendix A). A set of objects comprises the raw data for a cluster analysis and can be described by two standard formats: a pattern matrix and a proximity matrix.

2.1.1 Pattern Matrix

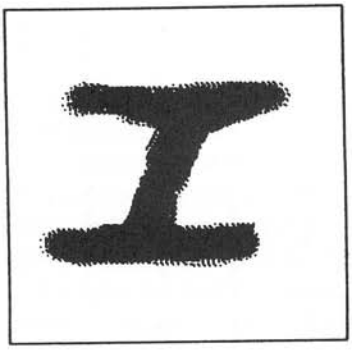
If each object in a set of n objects is represented by a set of d measurements (or attributes or scores), each object is represented by a pattern, or d-place vector. The set itself is viewed as a $n \times d$ *pattern matrix*. Each row of this matrix defines a pattern and each column denotes a *feature*, or measurement. For example, when clustering time functions such as biological signals or radar echoes, a feature could be a sample value taken at a particular time; the average value of the signal could also be a feature. The set of feature values for a signal is a pattern. We require that the same features be measured for all patterns. If patients in a hospital are to be clustered, each row in the pattern matrix would represent one individual. The features, or columns in the pattern matrix, could represent responses to questions on an admission form or the results of diagnostic tests. The same questions must be asked of every patient and the same diagnostic tests must be performed on all patients in a particular experiment. Categorical, or extrinsic, information, such as age, sex, religion, or hair color, is normally used to interpret the results of a cluster analysis but is not part of the pattern matrix.

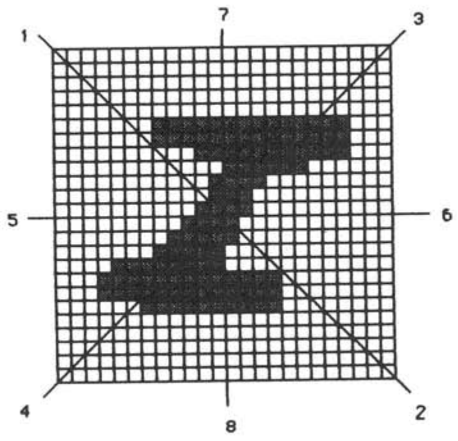
The d features are usually pictured as a set of orthogonal axes. The n patterns are then points embedded in a d-dimensional space called a *pattern space*. We use the word "pattern" in the technical sense as a point in a pattern space, not to describe the topological arrangement of objects. A cluster can be visualized as a collection of patterns which are close to one another or which satisfy some spatial relationships. The task of a clustering algorithm is to identify such natural groupings in spaces of many dimensions. Although visual perception is limited to three dimensions, one must be careful not to think automatically of clustering problems as two- or three-dimensional. The real benefit of cluster analysis is to organize multidimensional data where visual perception fails.

Example 2.1

This example shows the pattern matrix representation of a data set, called the 8OX data, that will be used to demonstrate several projection and clustering methods. This data set was derived from the Munson handprinted FORTRAN character set, which has been used extensively in pattern recognition studies and consists of handwritten characters from several authors, each of whom wrote three alphabets of 46 characters. The handwritten characters were digitized on a 24 × 24 grid and the gray values were quantized to two levels, resulting in a binary image. The data for this example use the characters 8, O (letter "oh"), and X from each of the three alphabets written by the first five authors for a total of 45 patterns. Each pattern is represented by eight features, which count the number of squares from the perimeter to the character, as shown in Figure 2.1. The actual feature values in integer format for this 45 × 8 pattern matrix are shown in Table 2.1. The first 15 patterns (rows) belong to category 8, the next 15 patterns belong to category O, and the last 15 patterns belong to category X.

The discussion above does not eliminate the possibility of clustering the features, or columns of the pattern matrix. We simply transpose the pattern matrix, being sure to choose a proper measure of similarity for the features (Section 2.2). The names "Q-mode clustering" and "R-mode clustering" have been applied to the clustering of patterns and features, respectively (Sneath and Sokal, 1973).





Pattern: (11, 11, 5, 6, 10, 10, 5, 5)

Figure 2.1 Binary representation of a handwritten character.

TABLE 2.1 Pattern Matrix for 8OX Data

7	13	5	5	6	13	2	3
5	13	6	4	6	13	3	13
9	10	6	6	8	10	2	3
7	7	6	6	8	7	2	3
8	7	6	6	8	7	2	0
7	7	6	7	7	7	1	1
6	10	7	8	8	9	4	4
6	7	7	7	9	8	4	5
5	5	5	12	10	7	2	3
7	8	4	4	7	6	2	3
8	7	5	4	6	10	1	0
6	10	5	2	6	8	1	2
7	10	5	5	8	7	1	20
7	10	6	6	6	8	3	3
6	6	7	7	8	8	3	2
7	7	5	6	3	3	4	6
7	6	7	6	3	4	6	5
6	6	5	5	4	3	4	5
8	8	7	6	5	7	5	5
6	8	7	5	5	6	2	2
7	7	7	7	6	6	3	2
7	7	7	7	5	6	5	5
7	7	8	7	5	6	4	5
8	7	7	6	6	5	4	5
7	8	6	5	4	6	2	4
9	7	7	6	8	6	4	3
8	8	7	6	7	6	4	4
7	6	6	5	3	2	5	4
7	6	7	5	5	5	4	3
0	8	7	6	6	7	2	4
10	7	6	6	9	9	7	10
10	4	4	4	8	8	3	10
10	7	4	4	9	9	3	9
7	7	6	5	10	10	10	8
6	10	6	10	8	8	13	4
8	10	7	10	9	8	11	4
5	7	8	7	8	9	9	8
5	6	7	7	9	9	9	7
6	7	11	6	8	11	7	10
6	12	10	4	8	11	8	3
7	12	8	6	9	11	9	1
11	8	7	10	11	10	6	9
9	5	6	7	10	9	7	5
10	5	6	4	9	9	6	11
0	5	11	6	9	11	5	9

2.1.2 Proximity Matrix

Clustering methods require that an index of proximity, or alikeness, or affinity, or association be established between pairs of patterns. This index can be computed from a pattern matrix, as discussed in Section 2.2, or can be formed from raw data. The data in some psychometric applications are collected as proximities. For example, several individuals can be asked to rank their preference for brands of soap and the proximity between two brands can be computed by averaging over individuals. An individual can also be asked to provide proximities directly by judging similarity between brands on a scale from 1 to 10. A *proximity matrix* $[d(i, j)]$ accumulates the pairwise indices of proximity in a matrix in which each row and column represents a pattern. We ignore the diagonal entries of a proximity matrix since all patterns are assumed to have the same degree of proximity with themselves. We also assume that all proximity matrices are symmetric, so all pairs of objects have the same proximity index, independent of the order in which they are written. Hubert (1973) and Gower (1977) consider nonsymmetric proximity matrices.

A proximity index is either a *similarity* or a *dissimilarity*. The more the ith and jth objects resemble one another, the larger a similarity index and the smaller a dissimilarity index. For example, Euclidean distance between two patterns in a pattern space is a dissimilarity index, whereas the correlation coefficient is a similarity index. Several proximity indices are described in Section 2.2. Note that a pattern matrix can easily be converted to a proximity matrix with proximity indices, but projection algorithms (Sections 2.4 and 2.5) or multidimensional scaling techniques (Section 2.7) are needed to convert a proximity matrix into a pattern matrix.

Example 2.2

We present an example of a proximity matrix that was used by Levine (1977a) to study the perceived similarity of numerical digits by subjects. The subjects were eight graduate students who observed a single numerical digit (0–9) as a 7 × 9 dot matrix character for

TABLE 2.2 Confusion Matrix for Stimulus–Response Combination

		Response									
		0	1	2	3	4	5	6	7	8	9
	0	45	10	68	68	19	39	42	27	59	32
	1	16	269	26	9	40	7	10	12	5	6
	2	7	5	330	13	5	10	4	9	6	7
	3	16	8	40	232	11	33	17	13	13	6
Stimulus	4	7	19	39	13	290	7	3	11	4	6
	5	18	6	14	17	12	280	20	13	10	6
	6	19	7	42	47	13	46	152	21	42	13
	7	11	5	73	14	9	6	1	270	3	6
	8	27	2	43	71	10	46	37	11	120	30
	9	21	2	24	42	6	56	18	8	34	196

variable time on a CRT display system. A noise field was immediately displayed on the CRT so that the digit was not clearly visible. The subjects had to respond what digit was present in the noisy image. Each student looked at only 50 stimuli, so Table 2.2 shows the aggregate response of all eight students. The table shows the confusion for each possible stimulus–response combination. The entry in the second row of the matrix in Table 2.2 indicates that of the 400 stimuli presented for digit 1, 269 correct responses were made by the subjects. Levine (1977a) defined the frequency of confusion between stimuli to be the measure of similarity. Thus digit pair 9 and 3 are considered more similar than digit pair 9 and 1. Notice that this similarity matrix is nonsymmetric. Multidimensional scaling and hierarchical clustering algorithms were applied to this matrix by Levine to study the evidence of hierarchical structure in the organization of visual stimuli.

2.1.3 Data Types and Scales

Now that the two primary formats for representing data—the pattern matrix and the proximity matrix—have been established, we turn to the characteristics of the data themselves. Anderberg (1973) outlines a categorization of data types and data scales appropriate for cluster analysis that is summarized below. Recognizing the type and scale of data will help in selecting a clustering algorithm.

Data *type* refers to the degree of quantization in the data. A single feature can be typed as binary, discrete, or continuous. *Binary* features have exactly two values and occur, for example, in "yes–no" responses on a questionnaire. A *discrete* feature has a finite, usually small, number of possible values. For example, samples of a speech signal can be quantized to 16, or 2^4, levels, so a feature representing the sample can be coded into 4 bits. All measurements and all numbers stored in computers have a finite number of significant digits, so, strictly speaking, all features are discrete. However, it is often convenient to think of a feature value as a point on the real line that can take on any real value in a fixed range of values. Such a feature is called *continuous*.

Proximity indices can also be binary, discrete, or continuous. For example, suppose that a set of objects is partitioned into mutually exclusive, all-inclusive subsets. One binary index of similarity assigns zero to a pair of objects that fall in different subsets and one to a pair in the same subset. A rank order proximity index is an integer from 1 to $n(n-1)/2$, where n is the number of objects. The integers represent the relative order of the proximities. Such an index is discrete. The Euclidean distance proximity index, defined for patterns in a pattern space, is typed continuous.

The second trait of a feature and of a proximity index is the data *scale*, which indicates the relative significance of numbers. Data scales can be dichotomized into *qualitative* (nominal and ordinal) scales and *quantitative* (interval and ratio) scales. A *nominal* scale is not really a scale at all because numbers are simply used as names. For example, a (yes, no) response could be coded as (0, 1) or (1, 0) or (50, 100); the numbers themselves are meaningless in any quantitative sense. The other qualitative scale, and the weakest numerical scale, is the *ordinal* scale; the numbers have meaning only in relation to one another. For example, the scales (1, 2, 3), (10, 20, 30), and (1, 20, 300) are all equivalent from an ordinal viewpoint. Binary and discrete features and proximity indices can be coded on these qualitative scales.

The separation between numbers has meaning on an *interval* scale. A unit of measurement exists, and the interpretation of the numbers depends on this unit. For example, a person can be asked to judge satisfaction with politicians on a scale from 0 to 100. The pair of scores (45, 55) and the pair (10, 90) on two politicians would indicate very different perceptions. Before the number 10 could be interpreted, one would need to know that the scale was 0 to 100 or 1 to 10 or 10 to 100. Temperature provides another example of an interval scale. A reading of 90° Fahrenheit has a very different implication for comfort than does a temperature of 90° Celsius.

The strongest scale is the *ratio* scale, on which numbers have an absolute meaning. This implies that an absolute zero exists along with a unit of measurement, so the ratio between two numbers has meaning. For example, the distance between two cities can be measured in meters, miles, or inches, but doubling the distance always has the same significance when driving from one to the other. Similarly, doubling one's income should double purchasing power, no matter what unit of currency is used. Degrees Kelvin establishes a ratio temperature scale because it has a natural zero. All three data types can be coded on the two quantitative scales.

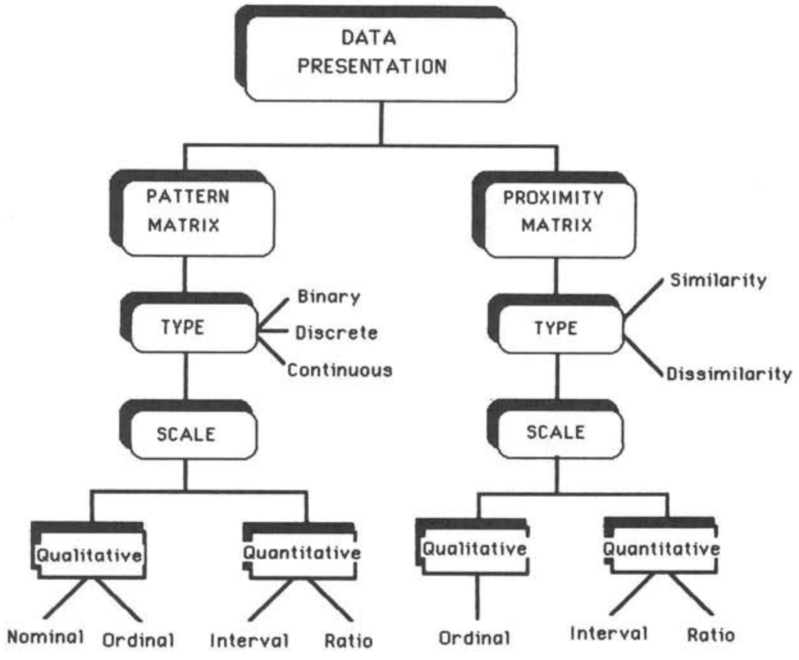


Figure 2.2 Formats, types, and scales for data.

Data type and scale are not always of one's choosing. Recognizing type and scale is important in both forming proximity indices and interpreting the results of a cluster analysis. For example, one should realize that human subjects are good at generating binary, qualitative data but that instruments are required to produce continuous, quantitative data. A human subject required to generate discrete, interval data will be under greater stress than one asked to provide binary, ordinal data, so the reliability of data can depend on type and scale. Anderberg (1973) explains conversions from one scale to another. Clustering methods (Chapter 3) use quantitative indices of proximity to assign a cluster label, or name, to each object, so a nominal scale can be generated from a quantitative scale. Multidimensional scaling (Section 2.7) changes ordinal scales into ratio scales. The various formats, types, and scales for data are summarized in Figure 2.2.

2.2 PROXIMITY INDICES

This section explains some of the more common proximity indices. Anderberg (1973) provides a thorough review of measures of association and their interrelationships. A proximity index between the *i*th and *k*th patterns is denoted $d(i, k)$ and must satisfy the following three properties:

1. (a) For a dissimilarity: $d(i, i) = 0$, all i

   (b) For a similarity: $d(i, i) \geq \max_k d(i, k)$, all i
2. $d(i, k) = d(k, i)$, all (i, k)
3. $d(i, k) \geq 0$, all (i, k)

Ratio and nominal proximity indices are discussed in separate sections.

2.2.1 Ratio Types

A proximity index can be determined in several ways. Suppose that we begin with a pattern matrix $[x_{ij}]$, where x_{ij} is the *j*th feature for the *i*th pattern. All features are continuous and measured on a ratio scale. The most common proximity index for such patterns is the Minkowski metric, which measures dissimilarity. The *i*th pattern, which is the *i*th row of the pattern matrix, is denoted by the column vector $\mathbf{x}_i$.

$$\mathbf{x}_i = (x_{i1}\, x_{i2} \ldots x_{id})^T, \; i = 1, 2, \ldots, n$$

Here d is the number of features, n the number of patterns, and T denotes vector transpose. The Minkowski metric is defined by

$$d(i, k) = \left(\sum_{j=1}^{d} |x_{ij} - x_{kj}|^r \right)^{1/r} \qquad \text{where } r \geq 1$$

All Minkowski metrics satisfy the additional metric properties stated below. Property 5 is called the *triangle inequality*.

4. $d(i, k) = 0$ only if $\mathbf{x}_i = \mathbf{x}_k$
5. $d(i, k) \leqq d(i, m) + d(m, k)$, all (i, k, m)

Gower and Legendre (1986) show that for a metric dissimilarity matrix $[d(i, j)]$, only properties 1 and 4 are required; other properties can be derived from these two.

The three most common Minkowski metrics are defined below and are illustrated in Figure 2.3.

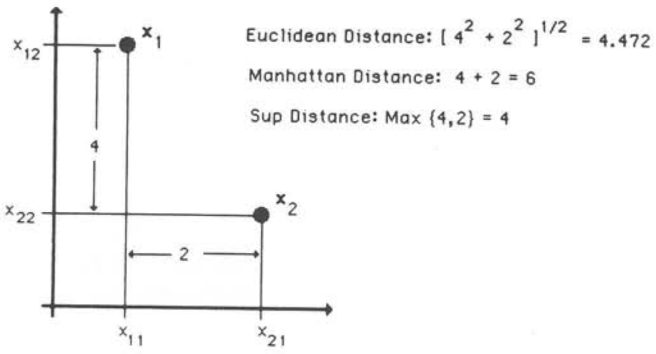

x_1
x_2
x_{12}
x_{22}
x_{11}
x_{21}
4
2
Euclidean Distance: $[4^2 + 2^2]^{1/2} = 4.472$
Manhattan Distance: $4 + 2 = 6$
Sup Distance: Max $\{4,2\} = 4$

Figure 2.3 Minkowski metrics.

1. $r = 2$ (Euclidean distance)

$$d(i, k) = \left[\sum_{j=1}^{d} (x_{ij} - x_{kj})^2\right]^{1/2} = [(\mathbf{x}_i - \mathbf{x}_k)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_k)]^{1/2}$$

2. $r = 1$ (Manhattan, or taxicab, or city block distance)

$$d(i, k) = \sum_{j=1}^{d} |x_{ij} - x_{kj}|$$

3. $r \rightarrow \infty$ (''sup'' distance)

$$d(i, k) = \max_{1 \leq j \leq d} |x_{ij} - x_{kj}|$$

Euclidean distance is the most common of the Minkowski metrics. The familiar geometric notions of invariance to translations and rotations of the pattern space are valid only for Euclidean distance. Accepted practice in the application area strongly affects the choice of proximity index. Euclidean distance seems to be preferred in engineering work. When all features are binary, the Manhattan metric is called the *Hamming distance*, or the number of features in which two patterns differ. Not all proximities encountered in applications are metrics. Tversky (1977) gives several examples to illustrate why a similarity is not always symmetric or transitive.

The squared Mahalanobis distance has also been used as a distance measure in cluster analysis (Everitt, 1974). The expression for the squared Mahalanobis distance between patterns $\mathbf{x}_i$ and $\mathbf{x}_k$ is

$$d(i, k) = (\mathbf{x}_i - \mathbf{x}_k)^{\mathrm{T}} \mathcal{S}^{-1} (\mathbf{x}_i - \mathbf{x}_k)$$

where the matrix $\mathcal{S}$ is the pooled sample covariance matrix, defined in Appendix D. The Mahalanobis distance incorporates the correlation between features and standardizes each feature to zero mean and unit variance. If $\mathcal{S}$ is the identity matrix, the squared Mahalanobis distance is the same as the squared Euclidean distance.

The sample correlation coefficient defined below is an index of similarity for continuous, ratio data that can be used with patterns but is more frequently used to measure the degree of linear dependency between two features.

$$d(j, r) = \left| \frac{(1/n) \sum_{i=1}^{n} (x_{ij} - m_j)(x_{ir} - m_r)}{s_j s_r} \right|$$

where m_j and s_j^2 are the sample mean and sample variance, respectively, for feature j and are defined in Section 2.3. The absolute value is required because a negative and a positive correlation that differ in sign but not in absolute value have the same significance when measuring similarity. If $d(j, r) = 0$, then features j and r are linearly independent. One of the features is usually discarded if $d(j, r)$ is close to 1. When data are on an ordinal scale, measures of rank correlation (Conover, 1971; Anderberg, 1973; Goodman and Kruskal, 1954) can be applied.

2.2.2 Nominal Types

If continuous, ratio-scaled data are considered to be the "strongest" type of data, then binary, nominal-scaled data are the "weakest" type. Many actual measurements, especially data collected from human subjects, are binary and nominal. Matching coefficients are proximity indices for such data. For convenience, all feature values are taken to be either 0 or 1. These symbols should be assigned consistently; if "1" means "large" for the first feature and "0" means "small," "1" must also denote "large" for all other features measuring size. Proximity indices between the ith and kth patterns are derived from the following contingency table. For example, a_{11} is the number of features that are 1 for both patterns, and a_{10} is the number of features that are 1 for pattern $\mathbf{x}_i$ and zero for pattern $\mathbf{x}_k$. The four entries sum to d, the number of features.

$$\begin{array}{ccc|cc|} & & & \multicolumn{2}{c}{\mathbf{x}_k} \\ & & & 1 & 0 \\ \cline{4-5} \mathbf{x}_i & & 1 & a_{11} & a_{10} \\ & & 0 & a_{01} & a_{00} \\ \cline{4-5} \end{array}$$

Several measures of proximity can be defined from the four numbers $\{a_{00}, a_{01}, a_{10}, a_{11}\}$ in the contingency table for two binary vectors. Anderberg (1973) reviews most of them and puts them into context. Gower (1971) discusses the properties of general coefficients based on weighted combinations of these four numbers and shows the conditions under which proximity matrices formed from them are positive-definite matrices. Gower's index can also be used with a mixture of binary, qualitative, and quantitative features. Measures of proximity for discrete data have been proposed by Hall (1967), who described a heterogeneity function, and Bartels et al. (1970), who introduc~~t~~ed the Calhoun distance as the percentages of patterns "between" two given patterns. Many other proximity measures have been defined for particular problems. Hubalek (1982) summarizes and evaluates proximity measures for binary vectors.

Two common matching coefficients between $\mathbf{x}_i$ and $\mathbf{x}_k$ are defined below:

1. Simple matching coefficient

$$d(i, k) = \frac{a_{00} + a_{11}}{a_{00} + a_{11} + a_{01} + a_{10}} = \frac{a_{00} + a_{11}}{d}$$

2. Jaccard coefficient

$$d(i, k) = \frac{a_{11}}{a_{11} + a_{01} + a_{10}} = \frac{a_{11}}{d - a_{00}}$$

The simple matching coefficient weights matches of 0's the same as matches of 1's, whereas the Jaccard coefficient ignores matches of 0's. The value 1 means "presence of effect" in some applications, so 1–1 matches are much more important than 0 – 0 matches. One example is that of questionnaire data. These two matching coefficients take different values for the same data and their meanings and interpretations are not obvious. Accepted practice in the area of application seems to be the best guide to a choice of proximity index.

Example 2.3

Suppose that two individuals are given psychological tests consisting of lists of 20 questions to which "yes" (1) and "no" (0) responses are required. Assuming that the questions are phrased so that "yes" and "no" have consistent interpretations, meaningful matching coefficients can be computed from the two patterns.

	Feature Number																			
	1	2	3	4	5					10					15					20
Pattern 1 ($\mathbf{x}_1$)	0	1	1	0	0	1	0	0	1	0	0	1	1	1	0	0	1	0	1	0
Pattern 2 ($\mathbf{x}_2$)	0	1	1	1	0	0	0	0	1	1	1	1	1	1	0	1	1	0	1	0

The matching coefficients are derived from the following table:

$$\begin{array}{cc|cc|}
 & & \multicolumn{2}{c}{\mathbf{x}_2} \\
 & & 1 & 0 \\
\cline{3-4}
\mathbf{x}_1 & 1 & 8 & 1 \\
 & 0 & 4 & 7 \\
\cline{3-4}
\end{array}$$

Simple matching coefficients: 15/20 = 0.75

Jaccard coefficient: 8/13 = 0.615

A value of 1 for either coefficient would mean identical patterns. However, other values are not as easily interpreted.

Example 2.4

Suppose that two partitions of nine numerals are given and a measure of their proximity is desired.

$$\mathscr{C}_1 = \{(1, 3, 4, 5), (2, 6), (7), (8, 9)\}$$

$$\mathscr{C}_2 = \{(1, 2, 3, 4), (5, 6), (7, 8, 9)\}$$

The characteristic function for a partition assigns the number 1 or 0 to a pair of numerals as follows.

$$T(i, j) = \begin{cases} 1 & \text{if numerals } i \text{ and } j \text{ are in the same subset in the partition} \\ 0 & \text{if not} \end{cases}$$

The characteristic functions T_1 and T_2, for partitions $\mathscr{C}_1$ and $\mathscr{C}_2$, respectively, are listed below in matrix form; T_1 is shown above the diagonal and T_2 is shown below the diagonal.

	1	2	3	4	5	6	7	8	9
1	—	0	1	1	1	0	0	0	0
2	1	—	0	0	0	1	0	0	0
3	1	1	—	1	1	0	0	0	0
4	1	1	1	—	1	0	0	0	0
5	0	0	0	0	—	0	0	0	0
6	0	0	0	0	1	—	0	0	0
7	0	0	0	0	0	0	—	0	0
8	0	0	0	0	0	0	1	—	1
9	0	0	0	0	0	0	1	1	—

The two characteristic functions are matched term by term to obtain the following table and coefficients. The relative significance of these values is discussed in the next section.

$$\begin{array}{cc|cc|} & & \multicolumn{2}{c}{T_1} \\ & & 1 & 0 \\ \cline{3-4} T_2 & 1 & 4 & 6 \\ & 0 & 4 & 22 \\ \cline{3-4} \end{array}$$

Simple matching coefficient: 26/36 = 0.722
Jaccard coefficient: 4/14 = 0.286

2.2.3 Missing Data

The problem of missing observations occurs often in practical applications. Suppose that some of the pattern vectors have missing feature values, as in

$$\mathbf{x}_i = (x_{i1}\ x_{i2}\ ?\ x_{i4}\ ?\ x_{i6})^{\mathrm{T}}$$

where the third and fifth features have not been recorded for the ith pattern. Missing values occur because of recording error, equipment failure, the reluctance of subjects to provide information, carelessness, and unavailability of information. Should incomplete pattern vectors be discarded? Should missing values be replaced by averages or nominal values? Answers to these questions depend on the size of the data set and the type of analysis. Sneath and Sokal (1973), Kittler (1978), Dixon (1979), and Zagoruiko and Yolkina (1982) all treat the problem of missing data.

Dixon (1979) describes several simple, inexpensive, easy to implement, and general techniques for handling missing values. These techniques either eliminate part of the data, estimate the missing values, or compute an estimated distance between two vectors with missing values. We summarize some of these techniques here.

1. Simply delete the pattern vectors or features that contain missing values. This technique does not lead to the most efficient utilization of the data and should be used only in situations where the number of missing values is very small.
2. Suppose that the jth feature value in the ith pattern vector is missing. Find the K nearest neighbors of $\mathbf{x}_i$ and replace the missing value x_{ij} by the average of the jth feature of the K nearest neighbors. The value of K should be a function of the size of the pattern matrix.
3. The distance between two vectors $\mathbf{x}_i$ and $\mathbf{x}_k$ containing missing values is computed as follows. First define the distance d_j between the two patterns along the jth feature.

$$d_j = \begin{cases} 0 & \text{if } x_{ij} \text{ or } x_{kj} \text{ is missing} \\ x_{ij} - x_{kj} & \text{otherwise} \end{cases}$$

Then the distance between $\mathbf{x}_i$ and $\mathbf{x}_k$ is written as

$$d(i, k) = \frac{d}{d - d_0} \sum d_j^2$$

where d_0 is the number of features missing in $\mathbf{x}_i$ or $\mathbf{x}_k$ or both. Note that if there are no missing values, then $d(i, k)$ defined above is the squared Euclidean distance.

4. Let $\bar{d}_j$ denote the average distance between all pairs of patterns along the jth feature defined as follows:

$$\bar{d}_j = \frac{2}{n(n-1)} \sum_{i=2}^{n} \sum_{k=1}^{i-1} |x_{ij} - x_{kj}|$$

where n is the number of patterns. Now define the distance between two patterns along the jth feature as

$$d_j = \begin{cases} \bar{d}_j & \text{if } x_{ij} \text{ or } x_{kj} \text{ is missing} \\ x_{ij} - x_{kj} & \text{otherwise} \end{cases}$$

Finally, the distance between patterns $\mathbf{x}_i$ and $\mathbf{x}_k$ is written as

$$d(i, k) = \sum d_j^2$$

Based on experimental results, Dixon (1979) recommends method 3 as the best overall method.

2.2.4 Probabilistic Indices

Goodall (1966) proposed an index of similarity that has a uniform distribution when the data are "random." The idea of using a probability scale to assess the significance of a proximity measure appears in Hamdan and Tsokos (1971), who define an information measure for a contingency table, and Brockett et al. (1981), who used the asymptotic distribution of an information-theoretic measure on questionnaire data. Li (1984) provided the most recent example of this type of measure. Before explaining the proximity measure, we reexamine the simple matching and Jaccard coefficients in light of their distributions under "random" data.

Matching coefficients measure the degree of similarity between objects. We know that their value is between 0 and 1 but do not know how large a value is required before two objects can be called "close." We now examine baseline distributions for the simple matching coefficient and the Jaccard coefficient. A baseline distribution describes a state of "randomness," or the absence of structure, for gauging the magnitude of a matching coefficient. Baseline distributions are used extensively in Chapter 4. Two vectors will be called "close" if a similarity as large as the one observed is unlikely under a baseline distribution.

The simple matching coefficient between two d-position binary vectors $\mathbf{a}$ and $\mathbf{b}$ can be expressed as

$$\text{SMC}(\mathbf{a}, \mathbf{b}) = (1/d)(\text{number of positions in which } \mathbf{a} \text{ and } \mathbf{b} \text{ match})$$

A value z of SMC can be considered to be "unusually" large if the probability of achieving a value of z or more is sufficiently low under some baseline distribution, such as the distribution of SMC for two randomly selected d-vectors. The choice of a baseline distribution is a matter of taste and depends on the application. The population Ω_{01} is defined as the set of all 4^d possible pairs of d-vectors. The probability function P_{01} assigns probability 4^{-d} to each pair in Ω_{01} and provides an obvious baseline distribution. This is equivalent to filling in the d-vectors by choosing the 1's and 0's independently with probability 1/2, as in d flips of a true coin. It is easy to show that the distribution of SMC follows the binomial distribution (Appendix B). Thus the probability that SMC is k/d or more can be written as

$$P_{01}[\text{SMC}(\mathbf{a}, \mathbf{b}) \geq k/d] = \sum_{j=k}^{d} \binom{d}{j} (1/2)^d$$

The notation

$$\binom{d}{m}$$

denotes the binomial coefficient or

$$\binom{d}{m} = \frac{d!}{m!\,(d-m)!}$$

For example, the probability that two six-position, randomly chosen binary vectors match in four or more positions (i.e., SMC $\geq 2/3$) is 0.3438, while the chance that SMC is 5/6 or more is 0.1094. Thus a value as large as 2/3 for SMC when $d = 6$ is not too unlikely even when there is no inherent correspondence between the vectors. A value of 5/6 might be required before calling the vectors unusually close under this baseline distribution. Even a perfect match has probability 0.0156, so one can never be absolutely sure that a large similarity is not a purely random event.

The Jaccard coefficient for two d-position vectors is given below, where $\mathbf{0}$ is the vector containing all zeros.

$$J(\mathbf{a}, \mathbf{b}) = \frac{\text{number of 1–1 matches}}{d - \text{Number of 0–0 matches}} \qquad \text{if } (\mathbf{a}, \mathbf{b}) \neq (\mathbf{0}, \mathbf{0})$$
$$J(\mathbf{0}, \mathbf{0}) = 1$$

The baseline distribution for J cannot be stated as compactly as that for SMC.

$$P_{01}(J \geq z) = \sum_{x \geq z} \sum{}^{*} P_{01}(k, m)$$

where

$$P_{01}(k, m) = \binom{d}{m}\binom{d-m}{k}(1/2)^{d+m+k} \qquad \text{if } 0 \le m \le d \text{ and } 0 \le k \le d-m$$

The starred sum is over the set

$$\{(k, m) : \frac{k}{d-m} = x \text{ and } 0 \le m \le d \text{ and } 0 \le k \le d-m\}$$

When $d = 6$, the chance that J is 5/6 or more is 0.0186, while the chance that SMC is 5/6 or more is 0.1094. Thus a Jaccard value of 5/6 or more is more unusual than an SMC value of 5/6 or more. Figure 2.4 shows the probabilities that the two coefficients are x or more when $d = 6$. The Jaccard coefficient can take on more values than the simple matching coefficient.

One can argue that choosing two binary vectors purely at random does not provide a sharp test of SMC or J because the population of all pairs of vectors is too large. For example, if the number of 1's is fixed in each vector, the vectors [10111011] and [01001111] have three 1–1 matches no matter how the entries of the two vectors are rearranged. The permutation statistic proposed by Li (1984) overcomes this limitation. It measures the correspondence between two binary vectors, just as SMC and J. However, it can be interpreted directly because it has a uniform distribution over the interval [0, 1] under a baseline distribution, shown as S in Figure 2.4. Note that the distributions of SMC and J in Figure 2.4 are under P_{01}, while the distribution of S is under a baseline distribution based on random permutations and described below.

Suppose that a measure of correspondence between binary d-vectors **a** and **b** is to be defined and the vectors are treated as dichotomous, so 0–0 matches

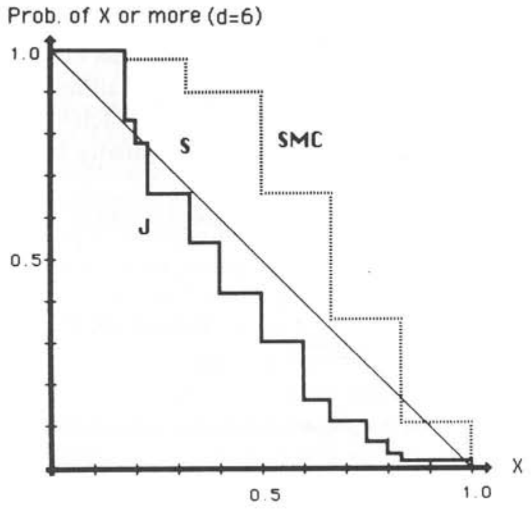


Figure 2.4 Baseline distributions of matching coefficients.

are not as important as 1–1 matches. Consider the population Ω_{02} of all $d!$ pairs of vectors that can be obtained by permuting the entries of one of the vectors. Not all pairs of vectors are distinct. Probability function P_{02} assigns each pair of vectors probability mass $1/d!$, thus establishing a new baseline distribution.

Let A_{11} be the number of 1–1 matches in a randomly selected pair of vectors from population Ω_{02}. Let N_a be the number of 1's in **a** and let N_b be the number of 1's in **b.** All pairs of vectors in Ω_{02} have N_a and N_b 1's. For example, there are six 1's in [10111011]. The probability that $A_{11} = k$ can be obtained from the hypergeometric distribution (Appendix B) under P_{02}. In the notation of Appendix B, we have a population of size d with N_b defectives and we take a sample of size N_a. Of course, the roles of N_a and N_b can be reversed. The probability of exactly k matches between pairs of 1's is

$$P_{02}(A_{11} = k) = \frac{\binom{N_b}{k}\binom{d - N_b}{N_a - k}}{\binom{d}{N_a}} = H(k, N_a, N_b, d)$$

This probability expression requires that

$$\max\{0, N_a + N_b - d\} \le k \le \min\{N_a, N_b\}$$

The S-statistic defined below is essentially the inverse of the hypergeometric cumulative density function. Such statistics have been used elsewhere (Kempthorne, 1952). The additive factor ensures that S has a (continuous) uniform distribution over the unit interval since U is a continuous uniform random variable over the unit interval. If t is the number of 1–1 matches observed between d-vectors **a** and **b,** the S-measure of proximity is

$$S(\mathbf{a}, \mathbf{b}) = \sum_{k<t} H(d, N_a, N_b, k) + H(d, N_a, N_b, t)U$$

Since the distribution of S is uniform under P_{02}, the value of S is implicitly meaningful. For example, the probability that S is z or more is $1 - z$ for z between 0 and 1, as shown in Figure 2.4. This proximity has been used in the analysis of questionnaire data (Li and Dubes, 1984) and in a template-matching problem (Li and Dubes, 1985). The additive factor does not contribute much to the value of S except when d is small.

2.3 NORMALIZATION

Suppose that the raw data consist of an $n \times d$ pattern matrix in which all features are continuous and on a ratio scale. Raw data, or the actual measurements, are seldom used just as they are recorded unless a probabilistic model for pattern generation is available. Some normalization is usually employed based on the requirements of the analysis. Preparing the data for a cluster analysis requires some sort of normalization that takes into account the measure of proximity. For example, Euclidean distance is a popular and familiar index of dissimilarity, but it implicitly assigns more weighting to features with large ranges than to those with small ranges. Scaling one feature in miles and a second feature in inches makes the second feature numerically overpower the first. We present a normalization scheme that remedies some of these problems.

As explained earlier in this section, the basic unit of data is called a pattern, denoted by a *d*-vector, whose components are scalars called features. The *i*th pattern is denoted by the (column) vector $\mathbf{x}_i^*$ in this section and the *j*th feature value for the *i*th pattern is denoted by x_{ij}^*. The asterisk denotes "raw" or unnormalized data. If n is the number of patterns in the analysis, the pattern matrix is the $n \times d$ matrix $\mathcal{A}^*$:

$$\mathcal{A}^* = [\mathbf{x}_1^* \quad \mathbf{x}_2^* \quad \cdots \quad \mathbf{x}_n^*]^{\mathrm{T}} = \begin{bmatrix} x_{11}^* & x_{12}^* & \cdots & x_{1d}^* \\ x_{21}^* & x_{22}^* & \cdots & x_{2d}^* \\ \vdots & \vdots & & \vdots \\ x_{n1}^* & x_{n2}^* & \cdots & x_{nd}^* \end{bmatrix}$$

Each row of $\mathcal{A}^*$ is a pattern. Each point in the pattern space is a potential pattern. We treat the case when $n > d$, so the patterns are visualized as a number of points scattered around the pattern space.

The *j*th feature average, m_j, and *j*th feature variance, s_j^2, are defined as the sample mean and the sample variance for the *j*th feature.

$$m_j = (1/n) \sum_{i=1}^{n} x_{ij}^*$$

$$s_j^2 = (1/n) \sum_{i=1}^{n} (x_{ij}^* - m_j)^2$$

The simplest type of normalization subtracts the feature means:

$$x_{ij} = x_{ij}^* - m_j \tag{2.1}$$

This normalization makes feature values invariant to rigid displacements of the coordinates. The second type of normalization translates and scales the axes so that all the features have zero mean and unit variance:

$$x_{ij} = \frac{x_{ij}^* - m_j}{s_j} \tag{2.2}$$

Removing the asterisk indicates that the pattern has been normalized, but the type of normalization must be clear from the context. Other types of normalization include scaling by the range (Carmichael et al., 1968) and a heterogeneity measure (Hall, 1969). Lumelsky (1982) incorporates the normalization into the clustering procedure. Normalization or scaling is not always desirable. For example, if the spread among the patterns is due to the presence of clusters, the normalization in Eq. (2.2) can change the interpoint distances and can alter the separation between natural clusters as demonstrated in Figure 2.5.

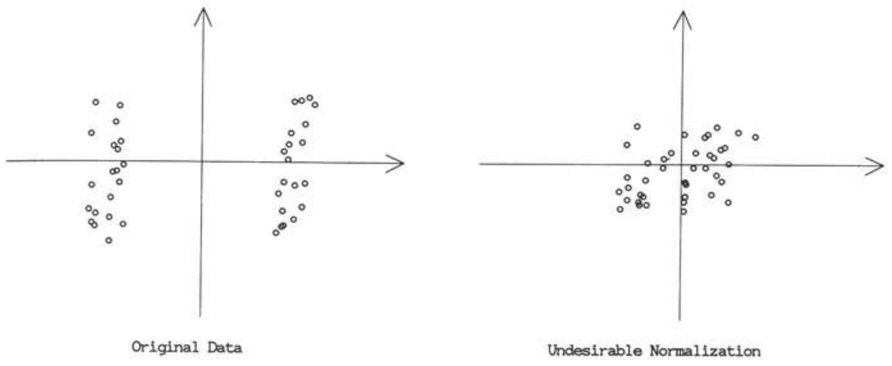


Figure 2.5 Effect of normalization.

The $d \times d$ matrix $\mathcal{R} = [r_{ij}]$ is defined below in terms of normalized data.

$$\mathcal{R} = (1/n)\, \mathcal{A}^{\mathrm{T}}\mathcal{A}$$

where

$$r_{ij} = (1/n) \sum_{k=1}^{n} x_{ki}x_{kj} \tag{2.3}$$

Under Eq. (2.1), $\mathcal{R}$ is a (sample) *covariance matrix*. Under Eq. (2.2), r_{ij} is the sample correlation coefficient between features i and j and $r_{jj} = 1$ for all j; $\mathcal{R}$ is then called a *correlation matrix*. The entries of $\mathcal{R}$ can be interpreted as relative spreads in the following sense. Each pattern is pictured as a unit of mass placed in the pattern space. The matrix $\mathcal{A}$ is pictured as a swarm of points in the pattern space (Figure 2.5), each point having the same mass and representing a row of $\mathcal{A}$. The normalizations of Eqs. (2.1) and (2.2) make the diagonal elements of $\mathcal{R}$ the moments of inertia of the swarm about the coordinate axes and force the origin of the coordinate system to coincide with the sample mean. Equation (2.2) makes all the moments of inertia unity. Another type of normalization that rotates the coordinate axes is discussed in Appendix C and is used in linear projection of the data (Section 2.4).

2.4 LINEAR PROJECTIONS

Projection algorithms map a set of n d-dimensional patterns onto an m-dimensional space, where $m < d$. The main motivation for studying projection algorithms in the context of cluster analysis is to permit visual examination of multivariate data, so $m = 2$ in our discussion. When a reasonably accurate two-dimensional representation of a set of patterns can be obtained, one can cluster by eye and qualitatively validate conclusions drawn from clustering algorithms. This search for a two-dimensional projection is closely related to problems in multivariate analysis of variance and factor analysis (see Appendices E and F).

A linear projection expresses the m new features as linear combinations of the original d features.

$$\mathbf{y}_i = \mathcal{H}\mathbf{x}_i \qquad \text{for } i = 1, \ldots, n$$

Here, $\mathbf{y}_i$ is an m-place column vector, $\mathbf{x}_i$ is a d-place column vector, and $\mathcal{H}$ is an $m \times d$ matrix. Linear projection algorithms are relatively simple to use, tend to preserve the character of the data, and have well-understood mathematical properties. The type of linear projection used in practice is influenced by the availability of category information about the patterns in the form of labels on the patterns. If no category information is available, the eigenvector projection (also called the Karhunen–Loeve method, or principal component method) is commonly used. Discriminant analysis is a popular linear mapping technique when category labels are available. We now describe these two popular linear projection algorithms. Readers will find some useful background information on linear algebra and scatter matrices in Appendices C and D.

2.4.1 Eigenvector Projection

The eigenvectors of the covariance matrix $\mathcal{R}$ in Eq. (2.3) define a linear projection that replaces the features in the raw data with uncorrelated features. These eigenvectors also provide a link between cluster analysis and factor analysis (Appendix E). Since $\mathcal{R}$ is a $d \times d$ positive definite matrix, its eigenvalues are real and can be labeled so that

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$$

A set of corresponding eigenvectors, $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_d$, is labeled accordingly. The $m \times d$ matrix of transformation $\mathcal{H}_m$ is defined from the eigenvectors of the covariance matrix (or correlation matrix) $\mathcal{R}$ as follows. The eigenvectors are also called principal components.

$$\mathcal{H}_m = \begin{bmatrix} \mathbf{c}_1^{\mathrm{T}} \\ \mathbf{c}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{c}_m^{\mathrm{T}} \end{bmatrix}$$

The rows of $\mathcal{H}_m$ are eigenvectors, as are the rows of $\mathcal{C}_R$ defined in Appendix C. This matrix projects the pattern space into an m-dimensional subspace (hence the subscript m on $\mathcal{H}_m$) whose axes are in the directions of the largest eigenvalues of $\mathcal{R}$ as follows. The derivation is given in Section 2.4.2.

$$\mathbf{y}_i = \mathcal{H}_m\mathbf{x}_i \qquad \text{for } i = 1, \ldots, n \tag{2.4}$$

The projected patterns can be written as follows:

$$\mathcal{B}_m = \begin{bmatrix} \mathbf{y}_1^{\mathrm{T}} \\ \mathbf{y}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{y}_n^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^{\mathrm{T}} \\ \mathbf{x}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{x}_n^{\mathrm{T}} \end{bmatrix} \mathcal{H}_m^{\mathrm{T}} = \mathcal{A}\mathcal{H}_m^{\mathrm{T}}$$

Note that $\mathbf{x}_i$ is the original pattern and $\mathbf{y}_i$ is the corresponding projected pattern. Equation (2.4) will be called the *eigenvector transformation*.

The covariance matrix in the new space can be defined with Eq. (C.1) as follows:

$$(1/n)\mathcal{B}_m^{\mathrm{T}}\mathcal{B}_m = (1/n)\sum_{i=1}^{n} \mathbf{y}_i\mathbf{y}_i^{\mathrm{T}} = \mathcal{H}_m\mathcal{R}\mathcal{H}_m^{\mathrm{T}} = \mathcal{H}_m\mathcal{C}_R^{\mathrm{T}}\Lambda_R(\mathcal{H}_m\mathcal{C}_R^{\mathrm{T}})^{\mathrm{T}}$$

The matrix $\mathcal{H}_m\mathcal{C}_R^{\mathrm{T}}$ can be partitioned as follows, where $\mathcal{I}$ is an $m \times m$ identity matrix and $\mathbb{O}$ is an $m \times (d - m)$ zero matrix.

$$\mathcal{H}_m\mathcal{C}_R^{\mathrm{T}} = [\mathcal{I}|\mathbb{O}]$$

Thus the covariance matrix in the new space, Λ_m, becomes a diagonal matrix as shown below.

$$(1/n)\mathcal{B}_m^{\mathrm{T}}\mathcal{B}_m = \Lambda_m = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m) \tag{2.5}$$

This implies that the m new features obtained by applying the linear transformation defined by $\mathcal{H}_m$ are uncorrelated.

Techniques for choosing an appropriate value for m in Eq. (2.4) are based on the eigenvalues of $\mathcal{R}$. Comparing Eqs. (C.2) and (2.5) shows that the sum of the first m eigenvalues is the "variance" retained in the new space. That is, the eigenvalues of $\mathcal{R}$ are the sample variances in the new space, while the sum of the d eigenvalues is the total variance in the original pattern space. Since the eigenvalues are ordered largest first, one could choose m so that

$$r_m = \sum_{i=1}^{m} \lambda_i \Big/ \sum_{i=1}^{d} \lambda_i \geq 0.95$$

which would assure that 95% of the variance is retained in the new space. Thus a "good" eigenvector projection is that which retains a large proportion of the variance present in the original feature space with only a few features in the transformed space. Krzanowski (1979) provides a table for the distribution of this ratio for $m = 1$, $d = 3$ and 4, and several values of n under the assumption that all components of all patterns are samples from independent standard normal distributions. These tables should help determine whether it is reasonable to say that the size of the largest eigenvalue could have been achieved by chance. Another technique for choosing m is to plot r_m as a function of m and look for a "knee" in the plot.

Example 2.5

This example shows the eigenvector projection on the 8OX data set described in Example 2.1. The pattern matrix is normalized by subtracting the sample means [Eq. (2.1)]. The normalized data are projected to two dimensions ($m = 2$) with Eq. (2.4) as shown in Figure 2.6, where category labels are given for the projected patterns. The percentage variance retained in two dimensions is 43.1%. A factor analysis (Appendix E) shows that features 5 to 8 most strongly affect the first coordinate (abscissa in Figure 2.6). Since these features are horizontal and vertical measurements (Figure 2.1), we could name the abscissa in Figure 2.6 as ''horizontal–vertical.'' Features 2, 3, 6, 7, and 8 most strongly affect the second projected feature (ordinate in Figure 2.6), so we could name the axis ''rightside.''

Subsequent clusterings of the 8OX data will show that the patterns are separated quite well according to category. Some of this separation is evident in Figure 2.6. To demonstrate how the first two principal components of the 8OX data exhibit more structure than other principal components, we show the projection defined by the third and fourth principal components, or the eigenvectors corresponding to the third and fourth largest eigenvalues of $\mathcal{R}$, in Figure 2.7. Little separation

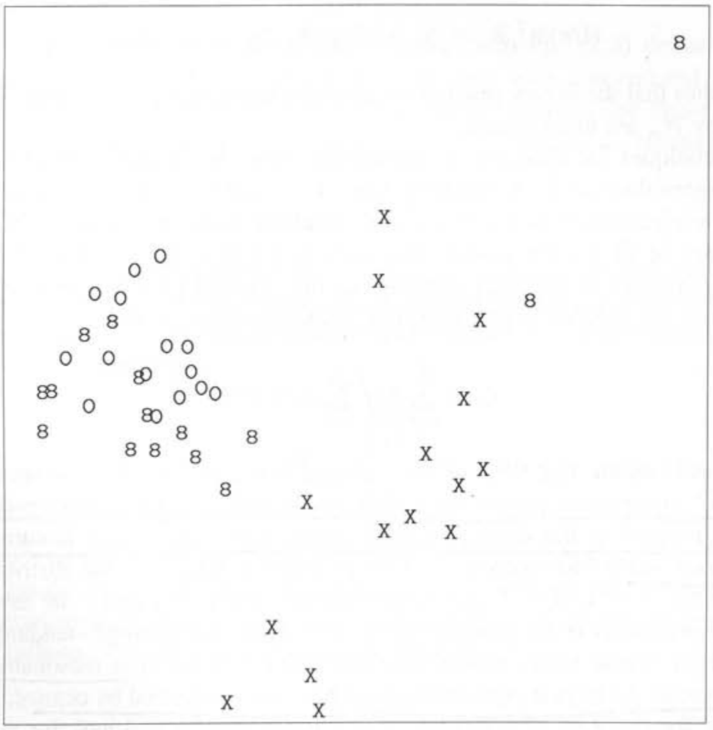


Figure 2.6 Two-dimensional representation of 80X data on the first two principal components.

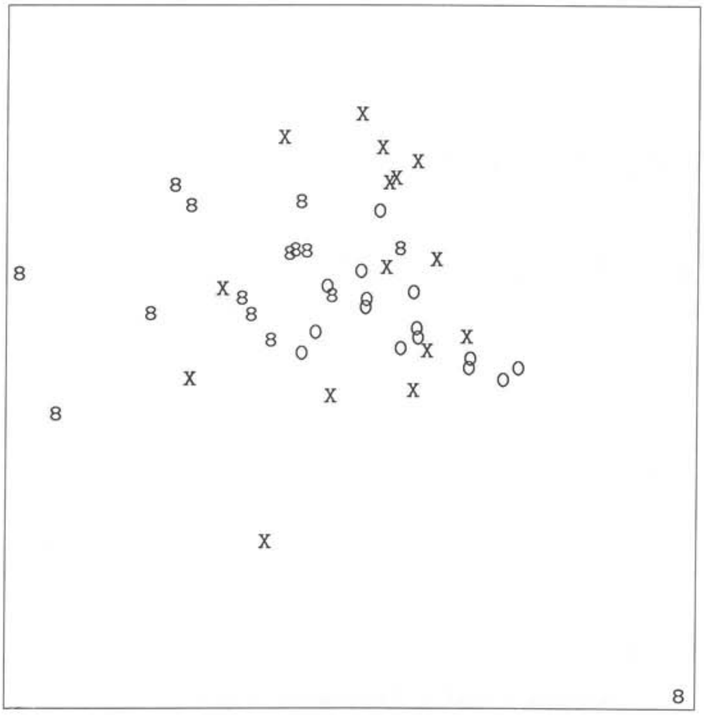
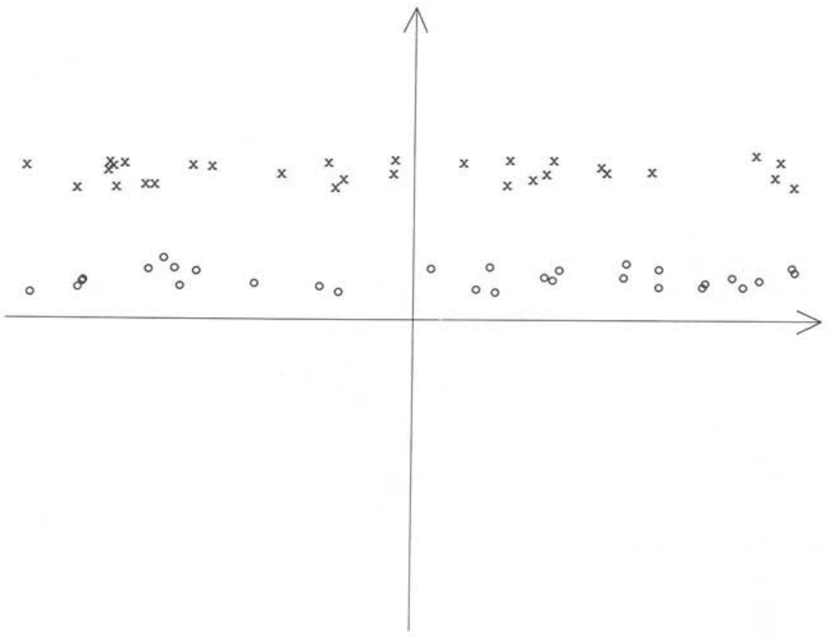


Figure 2.7 Two-dimensional projection of 80X data on the third and fourth principal components.

among categories is evident. However, there is no guarantee that the features with the largest eigenvalues will be the best for preserving the separation among categories. Figure 2.8 shows two "cigar-shaped" clusters. The eigenvector of $\mathcal{R}$ corresponding to the smallest eigenvalue instead of the largest eigenvalue is the best for projecting the data to one dimension. Chang (1983) also demonstrates that the practice of reducing dimensionality by choosing new features with the largest eigenvalues is not always justified.

Example 2.6

This example demonstrates the value of two-dimensional projections when observing data in four and six dimensions. Artificial data were generated for this study by computer with a random number generator. These data sets will be used throughout the book to illustrate clustering methods. The first data set, called DATA1, contains 100 patterns inside a four-dimensional hypercube arranged into four clusters. The Neoclus program described in Appendix H was used to generate this data set. Each cluster consists of at least 20 patterns spread around a cluster center according to a normal distribution whose mean is the cluster center and whose components are samples of independent Gaussian random variables with standard deviation 0.12. The cluster centers are chosen at random in the hypercube, but



```
Eigenvalue 1 = 13.2  Eigenvector = (1.00,0.05)
Eigenvalue 2 =  2.4  Eigenvector = (-.05,1.00)
```

Figure 2.8 First principal component not best for separating categories.

the clusters are not allowed to overlap significantly. In this data set, the numbers of patterns in the four clusters are 24, 35, 21, and 20, respectively.

Figure 2.9 provides a projection of DATA1 onto the eigenvectors corresponding to the two largest eigenvalues of $\mathcal{R}$. The percent variance retained is 78.6%. Since the patterns were generated in a hypercube, normalizations by Eqs. (2.1) and (2.2) should have little effect on the spread of the data. The locations of dots, or projected patterns, in Figure 2.9 suggest three clusters with the large cluster at the top divided into two overlapping clusters. One task of cluster analysis is to establish, in the original four-dimensional space, whether a three-cluster model or a four-cluster model is more appropriate for this data set. Beware of the limitations of two-dimensional projections. The large cluster in Figure 2.9 might be well separated into two individual clusters in four dimensions; we simply might not be able to see the separation in two dimensions. Figure 2.10 is identical to Figure 2.9 except that the circles of Figure 2.9 have been replaced by category labels. We see the four clusters generated by the program clearly from the labels. Clustering methods do not use category labels to organize the data. However, the interpretation of that organization often depends on the category labels.

A second data set, DATA2, provides a contrast to the structured data in DATA1 and consists of 100 patterns generated uniformly and independently over a hypercube in six dimensions. Figure 2.11 shows the projection to the two eigenvectors with largest eigenvalues. The percent variance retained is 41.9%. The human brain tries to make sense

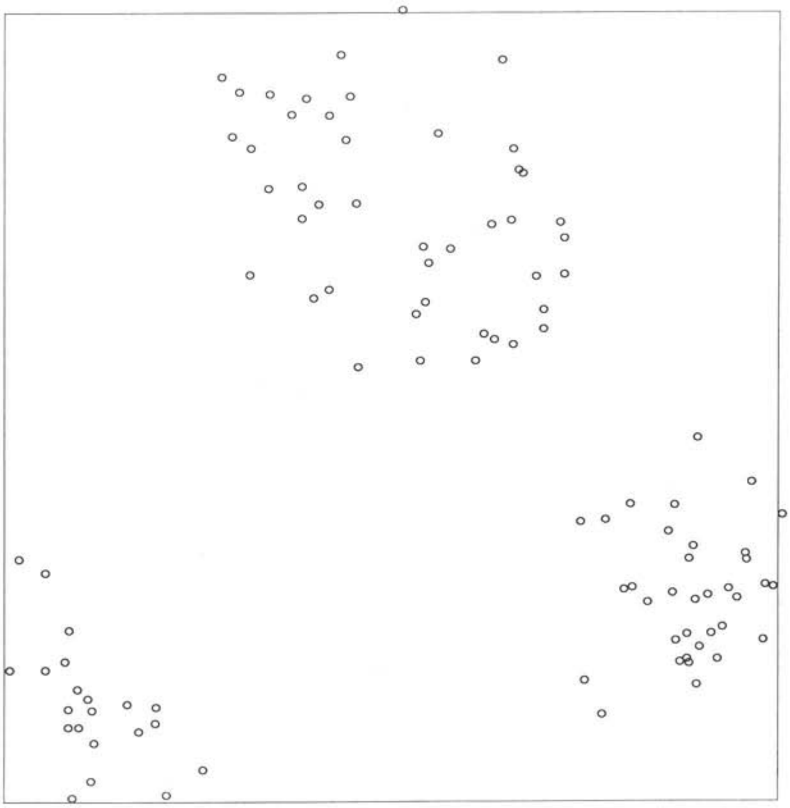


Figure 2.9 Two-dimensional projection of DATA1 onto first two principal components.

out of dot patterns, so one might be convinced that the projection contains clusters or other structure. The fact of the matter is that the data were generated purely at random, so any accumulations of points or large empty spaces in Figure 2.11 are artifacts. Unfortunately, a clustering method, when applied to random data, will impose some type of clustered structure. The task of cluster validity is to recognize bogus structures.

The projection in Eq. (2.4) is the most common means of representing patterns in two dimensions and of reducing the dimensionality of the set of patterns. Two reasons for the central role played by Eq. (2.4) are that it provides the best approximation to the data (in a mean-square-error sense) and that it maximizes scatter, as shown in Section 2.4.2.

2.4.2 Derivation

This section approaches linear projection from a minimum mean-square-error point of view, which means approximating the d-feature patterns in m dimensions by minimizing a square-error criterion function. Assume that an $m \times d$ matrix,

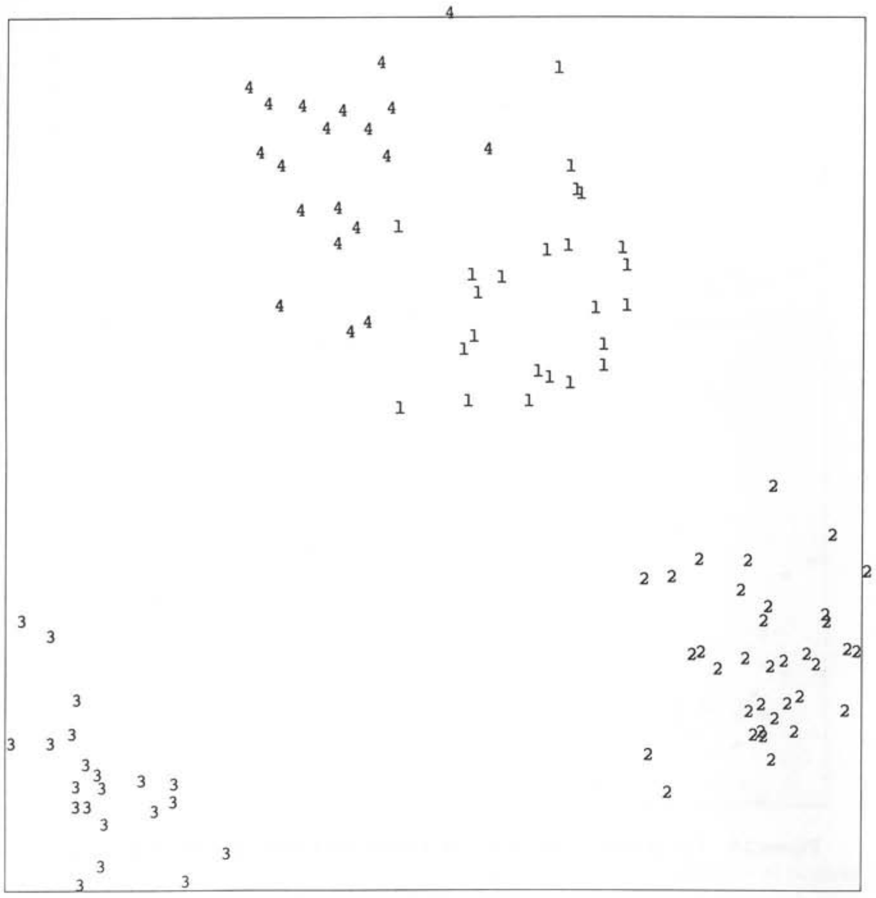


Figure 2.10 Two-dimensional projection of DATA1 showing category labels.

$\mathcal{H}$, is to be found which projects each d-dimensional pattern $\mathbf{x}_i$ into a point $\mathbf{y}_i = \mathcal{H}\mathbf{x}_i$ in such a way that the square error, explained below, is minimized.

A projection is defined in terms of a set of basis vectors. A set of orthonormal basis vectors in d dimensions $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_d\}$ is defined by the property

$$\mathbf{e}_i^T\mathbf{e}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Pattern $\mathbf{x}_i$ is expressed in terms of the basis vectors as follows:

$$\mathbf{x}_i = \sum_{k=1}^{d} \psi_{ik}\mathbf{e}_k$$

where $\psi_{ij} = \mathbf{x}_i^T\mathbf{e}_j$.

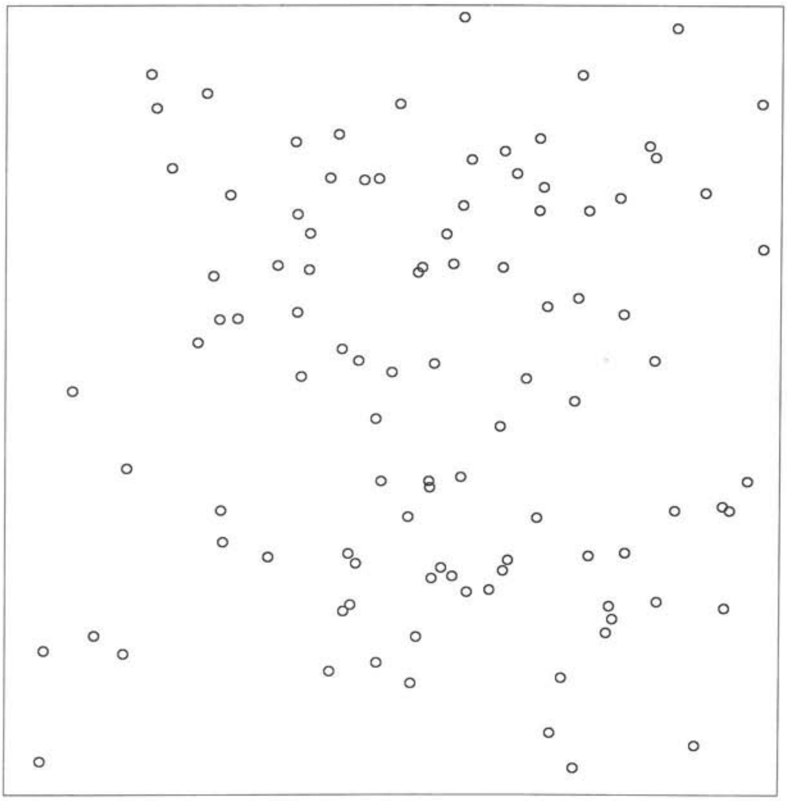


Figure 2.11 Two-dimensional projection of six-dimensional random data (DATA2).

If $\{\mathbf{e}'_1, \mathbf{e}'_2, \ldots, \mathbf{e}'_m\}$ is a subset of $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_d\}$, then the d-vector $\mathbf{y}'_i$ defined below approximates $\mathbf{x}_i$.

$$\mathbf{y}'_i = \sum_{k=1}^{m} \psi'_{ik}\mathbf{e}'_k$$

where $\psi'_{ij} = \mathbf{x}_i^{\mathrm{T}}\mathbf{e}'_j$. The goodness of the approximation can be judged by the average square-error measure SE(m), which depends on m and on the set of basis vectors chosen.

$$\mathrm{SE}(m) = (1/n) \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{y}'_i)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{y}'_i)$$

Since the m scalars $(\psi'_{i1}, \ldots, \psi'_{im})$ define the projection, $\mathbf{y}'_i$ is really a point in the m-dimensional space obtained by projecting each pattern to the space spanned by $(\mathbf{e}'_1 \ldots, \mathbf{e}'_m)$; that is, $\mathbf{y}'_i$ can also be expressed as in Eq. (2.4). Writing $\mathbf{y}'_i$ as a sum of d-vectors permits a meaningful expression of the square-error criterion.

A set of basis vectors that minimizes SE(m) has been shown to be a set of eigenvectors ($\mathbf{c}_1, \ldots, \mathbf{c}_m$) for the covariance matrix $\mathcal{R}$ (Tou and Heydorn, 1967; Sebestyen, 1962; Wilks, 1963; Fukunaga, 1972). In addition, the basis vectors that minimize SE(m) correspond to the m largest eigenvalues of $\mathcal{R}$. Projecting $\mathbf{x}_i$ into this optimal subspace is precisely the operation in Eq. (2.4). The degree of approximation can be determined by expressing the minimum square error in terms of eigenvalues of $\mathcal{R}$.

$$\mathrm{SE}(m)_{\min} = \sum_{j=1}^{d} \lambda_j - \sum_{j=1}^{m} \lambda_j = \sum_{j=m+1}^{d} \lambda_j$$

This result justifies the rule suggested earlier for choosing m. Retaining only those new features that provide the largest spreads minimizes the square error. In other words, the importance, in a square-error sense, of each prospective new feature is measured by an eigenvalue of $\mathcal{R}$. The normalization of Eq. (2.2) scales the pattern space so that $\Sigma_1^d \lambda_i = d$, but the development given above is wholly applicable.

The second reason for the importance of Eq. (2.4) is that the eigenvector projection maximizes scatter. This interpretation of Eq. (2.4) has its roots in theoretical statistics (Wilks, 1963). Assuming normalization by Eq. (2.1) or (2.2), the scatter of the set of patterns is given by

$$|\mathcal{S}| = n^d |\mathcal{R}|$$

where $|\mathcal{R}|$ is the determinant of the covariance matrix, $\mathcal{R}$.

Wilks (1963) has provided a geometrical interpretation of scatter. Another interpretation follows from the relation between $\mathcal{S}$ and $\mathcal{R}$. From Eq. (C.2),

$$|\mathcal{S}| = n^d \prod_{i=1}^{d} \lambda_i = \prod_{i=1}^{d} (n\lambda_i)$$

This shows that the scatter is invariant through a rotation of the pattern space, such as Eq. (2.4) when $m = d$. A set of eigenvalues for $\mathcal{R}$ is also a set for $\mathcal{S}$. More important, it shows that the scatter is proportional to the product of the sample variances ($\lambda_1, \ldots, \lambda_d$) along the rotated axes.

If the patterns are projected into an m-dimensional space by Eq. (2.4), their scatter is maximized in the m-dimensional space with respect to all other orthogonal m-dimensional projections because Eq. (2.4) uses eigenvectors corresponding to the m largest eigenvalues of $\mathcal{R}$ (or of $\mathcal{S}$). From an intuitive point of view, maximizing scatter might not appear to be as worthy a goal as minimizing square error, although both are achieved with the same transformation.

2.4.3 Discriminant Analysis

Classical discriminant analysis (Wilks, 1963; Friedman and Rubin, 1967; Fortier and Solomon, 1966; Lachenbruch and Goldstein, 1979) attempts to project patterns into a space having fewer dimensions than the original pattern space.

The discriminant analysis projection maximizes the between-group scatter while holding the within-group scatter constant. The scatter matrices have been defined in Appendix D. This projection requires that all patterns have pattern class, or category, labels. The dependence of discriminant analysis on the number of categories, K, makes this approach fundamentally different from the eigenvector projection in Section 2.4.1.

An important result of discriminant analysis demonstrates the existence of a $(K - 1) \times d$ matrix, $\mathcal{H}_0$, with a very interesting property. Letting $\mathbf{x}_j^{(i)}$ denote the jth pattern in class i, $\mathcal{H}_0$ projects each pattern into a $(K - 1)$-dimensional subspace by

$$\mathbf{y}_j^{(i)} = \mathcal{H}_0 \mathbf{x}_j^{(i)}, \qquad j = 1, \ldots, n_i \quad \text{and} \quad i = 1, \ldots, K \tag{2.6}$$

and the ratio of scatters $|\mathcal{S}_W|/|\mathcal{S}|$ remains constant. This ratio is called *Wilks's lambda statistic*. The rows of $\mathcal{H}_0$ are the eigenvectors corresponding to the $(K - 1)$ nonzero eigenvalues of $\mathcal{S}_W^{-1}\mathcal{S}_B$. This assumes that $n - K \geq d$, $d \geq K$, and that $\mathcal{S}_W$ is nonsingular. A tremendous reduction in the number of features can be realized with no increase in the scatter ratio $|\mathcal{S}_W|/|\mathcal{S}|$. For example, three groups of patterns in a 20-dimensional pattern space can be projected into the plane while maintaining the scatter ratio. In addition, the value of the scatter ratio can be expressed in terms of the eigenvalues

$$(\zeta_1, \zeta_2, \ldots, \zeta_{K-1})$$

of $\mathcal{S}_W^{-1}\mathcal{S}_B$ as follows, where the ordering $\zeta_1 \geq \zeta_2 \geq \cdots \geq \zeta_{K-1} \geq 0$ is assumed.

$$|\mathcal{S}_W|/|\mathcal{S}| = [(1 + \zeta_1)(1 + \zeta_2) \cdots (1 + \zeta_{K-1})]^{-1} \tag{2.7}$$

If the patterns are to be projected into a t-dimensional space, $t < (K - 1)$, and this scatter ratio is to be minimized in t dimensions, only the eigenvectors corresponding to the t largest eigenvalues $(\zeta_1, \ldots, \zeta_t)$ should be employed. Dividing the scatter ratio $|\mathcal{S}_W|/|\mathcal{S}|$ in the original pattern space by its counterpart in the t-dimensional space results in $[(1 + \zeta_{t+1}) \cdots (1 + \zeta_{K-1})]^{-1}$. If the $(K - t - 1)$ smallest eigenvalues of $\mathcal{S}_W^{-1}\mathcal{S}_B$ are sufficiently close to zero, the scatter ratio is increased very little by projecting into t dimensions.

The rows of $\mathcal{H}_0$ in Eq. (2.6) are eigenvectors of $\mathcal{S}_W^{-1}\mathcal{S}_B$, while the rows of $\mathcal{H}_m$ in Eq. (2.4) are eigenvectors of $\mathcal{R}$. Thus discriminant analysis requires more than a correlation matrix, but not much more. When the number of groups or pattern classes is two $(K = 2)$, the data can be transformed to a line without any increase in the scatter ratio. This one-dimensional projection is also called the Fisher linear discriminant.

Recall that the eigenvector projection [Eq. (2.4)] does not require category information, whereas the projection based on discriminant analysis in Eq. (2.6) demands category labels. Even when no extrinsic category labels are available, the patterns can be clustered (Chapter 3) and cluster labels used as category information for projection purposes. The adequacy of clusters in representing the structure of the data must be determined to interpret the results of such projections. Our emphasis is on projections involving no category labels.

Two items should be noted concerning the computation of $\mathcal{H}_0$ in Eq. (2.6). First, although both $\mathcal{S}_W^{-1}$ and $\mathcal{S}_B$ are symmetric, the product $\mathcal{S}_W^{-1}\mathcal{S}_B$ is not generally symmetric. Most simple computer programs for finding eigenvectors and eigenvalues require symmetry. Second, $\mathcal{S}_B$ involves only K vectors and a mean is subtracted so that the rank of $\mathcal{S}_B$ cannot be greater than $(K - 1)$. The rank of a product of two matrices cannot exceed the ranks of the components, which explains why $\mathcal{S}_W^{-1}\mathcal{S}_B$ has, at most, $(K - 1)$ nonzero eigenvalues.

The number of new coordinates in discriminant analysis depends on the number of categories. Some attempts have been made to alter this requirement. Okada and Tomita (1985) derive a set of coordinate axes by selecting axes one at a time under a constraint of orthonormality. Numerical examples show that the new discriminant axes are superior to those in Eq. (2.6) in some respects.

Example 2.7

A discriminant analysis was performed for the 8OX character data discussed in Example 2.1. Here we have three groups of 15 patterns each, each group corresponding to a different character. Thus the two-dimensional projection defined by Eq. (2.6) and shown in Figure 2.12 retains the scatter ratio from the original eight-dimensional pattern space. Separation among pattern classes is much more evident than in the eigenvector transformation of Figure 2.6. Both are linear transformations, but Figure 2.12 uses the intrinsic pattern class information to separate the patterns according to category.

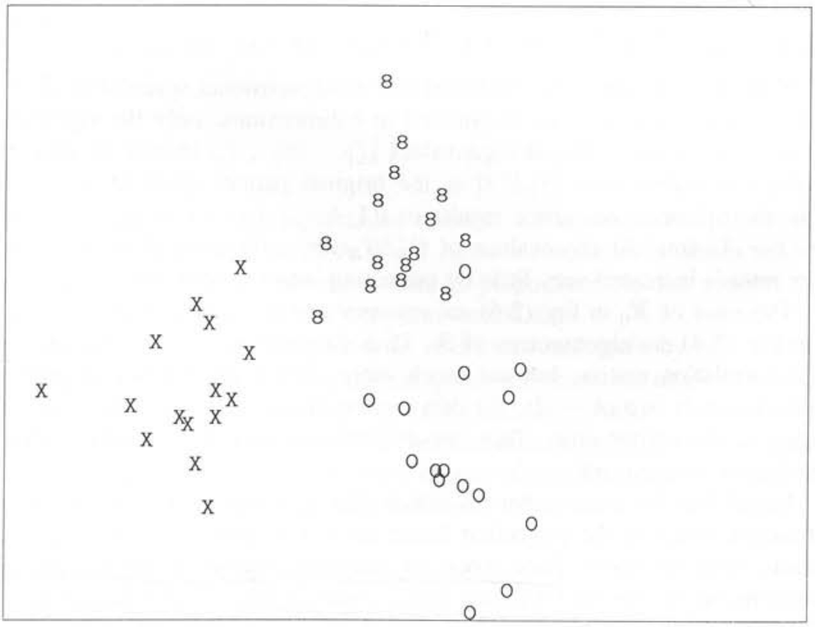


Figure 2.12 Two-dimensional projection of 80X data using discriminant analysis.

2.5 NONLINEAR PROJECTIONS

The inability of linear projections to preserve "complex" data structures has made nonlinear projections more popular in recent years. By "complex data structures" we mean situations where patterns lie on a curved surface. For example, the principal component method will not be able to obtain a good two-dimensional representation of a data set in which patterns lie along a helix in three dimensions. The notion of intrinsic dimensionality, discussed in Section 2.6, addresses this issue in detail. Most nonlinear projection algorithms are based on maximizing or minimizing a function of a large number of variables. This optimization problem is data dependent and does not involve an explicit mapping function. Therefore, a change in the number of patterns requires recomputing the entire projection. Nonlinear projection algorithms are expensive to use, so several heuristics are employed to reduce the search time for the optimal solution. For example, the best principal component projection could be used as the starting configuration for a nonlinear mapping algorithm.

Nonlinear projection algorithms can be derived from two viewpoints, depending on the prior information available about the patterns. If category information, or extrinsic pattern class labels, are known, the aim is to find a nonlinear projection that reduces dimensionality yet maximizes separability between categories. In the absence of category information, the goal is to project the patterns onto a low-dimensional space so as to retain as much structure as possible. For example, the goal could be to maintain all the interpattern distances. In statistical pattern recognition (Appendix A) the training samples are labeled according to pattern class and nonlinear projection algorithms are used to find an "effective" subset of m features. For example, between-class separability can be maximized and within-class variability minimized (Calvert, 1970; Koontz and Fukunaga, 1972; Fukunaga and Ando, 1977; Mantock and Fukunaga, 1980). In exploratory data analysis, we seek two-dimensional projections to visually perceive the structure present in the data. The selection of a nonlinear projection algorithm for exploratory data analysis is affected by the form of the available data. Here we assume that the data are presented as in a pattern matrix. If the analysis starts with a proximity matrix, multidimensional scaling (Section 2.7) is applicable.

2.5.1 Graphical Methods

We distinguish nonlinear projection to two dimensions from graphical representation of multidimensional patterns. While a projection algorithm seeks two coordinates that preserve structural information contained in the original d features, a graphical representation tries to preserve this information exactly by utilizing all the features as attributes of graphical representation for the patterns. For example, Chernoff (1973) represents each pattern as a cartoon face whose facial characteristics, such as nose length, mouth curvature, and eye size, are made to correspond to individual features. Chernoff's (1973) program can vary 18 different parameters in generating the cartoon faces, and hence n d-dimensional patterns can be represented by n cartoon faces, for $d \leq 18$.

Andrews (1972) represents each d-dimensional pattern vector by a Fourier series representation, where the coefficients in the series are the feature values, so that each pattern becomes a periodic time function. Studying the set of functions permits a grouping of the patterns in some situations. This approach is especially useful in identifying outliers, or patterns far removed from the main mass of patterns. Other graphical techniques include representing multidimensional points as stars, trees, and castles (Kleiner and Hartigan, 1981). Chernoff (1978), Chien (1978), and Everitt (1978) evaluate several techniques for graphically representing multivariate data. The development of computer graphics has influenced the popularity of graphical techniques in multivariate statistics. However, these techniques are valuable only when the number of patterns and the number of features are small ($n < 50$, $d < 10$).

2.5.2 Iterative Procedures

Sammon (1969) proposed a nonlinear technique that tries to create a two-dimensional configuration of points in which interpattern distances are preserved. Let $\{\mathbf{x}_i\}$ denote a set of n d-dimensional patterns and let $d(i, j)$ denote the distance between patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ in the d-dimensional pattern space. Suppose that a configuration of n points, one per pattern, is available in $m < d$ dimensions and let $D(i, j)$ be the distance between the points corresponding to $\mathbf{x}_i$ and $\mathbf{x}_j$ in this m-dimensional space. Each pattern is represented by one point in the configuration space. Sammon (1969) defined the mean-square-error between the two sets of distances as follows. This square-error formula is similar to the "stress" criterion from multidimensional scaling (Section 2.7), so E is sometimes called "stress." Both double sums are over the set $\{(i, j) : 1 \leq i < j \leq n\}$.

$$E = \frac{1}{\sum\sum_{i<j} d(i, j)} \sum\sum_{i<j} \frac{[d(i, j) - D(i, j)]^2}{d(i, j)}$$

The objective of Sammon's (1969) algorithm is to find, for a given m, a configuration of patterns that minimizes E. Note that the error, E, is invariant to scale changes and sample size. However, the error is a function of nm variables, namely the coordinates of the points in the m-dimensional projected space, so the minimization problem is not trivial. Sammon's algorithm starts with a random configuration of n patterns in m dimensions and uses the method of steepest descent to reconfigure the patterns so as to minimize E in an iterative fashion. Some details of this algorithm are discussed in Section 2.7.1. The algorithm terminates when the error, E, falls below an acceptable level or if the change in the value of E between two successive iterations of the steepest descent algorithm is negligible. The algorithm should be applied with several initial configurations to ensure a global minimum for E. Other procedures for minimizing functions of several variables, such as simulated annealing (Kirkpatrick, 1984), can also be brought to bear on this problem.

Kruskal (1971) shows that a configuration very similar to Sammon's can be generated by the MDSCAL algorithm (Section 2.7). Cormack (1971) provides several other expressions for square error and measures of distortion that can be used in place of E. A two-dimensional configuration for the 8OX data set (Example 2.1) using Sammon's method is shown in Figure 2.13. The 45 patterns in Figure 2.13 have been labeled by category to highlight the separation of the three categories. Other projections of the 8OX data are in Figures 2.6 and 2.12. Figure 2.14 shows the projection of 100 random patterns in six dimensions (DATA2) by Sammon's method. Comparing Figure 2.14 to the eigenvalue projection in Figure 2.11 suggests that Sammon's method attempts to spread the data as much as possible. The projection in Figure 2.14 appears more "regular" than that in Figure 2.11. For most data sets this algorithm converges in fewer than 20 steps. However, the mapping error may be too large for complex high-dimensional data with $m = 2$.

Several modifications to Sammon's algorithm have been proposed. Niemann and Weiss (1979) generalized the square error, E, in such a way that smaller interpattern distances influence E more or less than large interpattern distances, depending on a parameter. A procedure for computing an optimal step size at every iteration in the steepest descent algorithm, which guarantees the convergence of the algorithm, was also proposed. Unfortunately, this procedure imposes an extra computational burden. Pykett (1978) suggests projecting only "archetypes,"

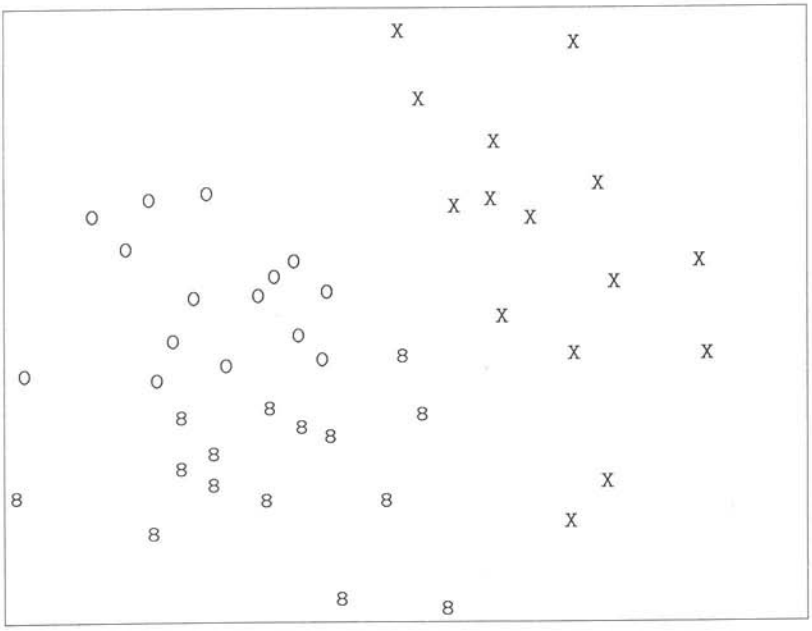


Figure 2.13 Projection of 80X data by Sammon's method.

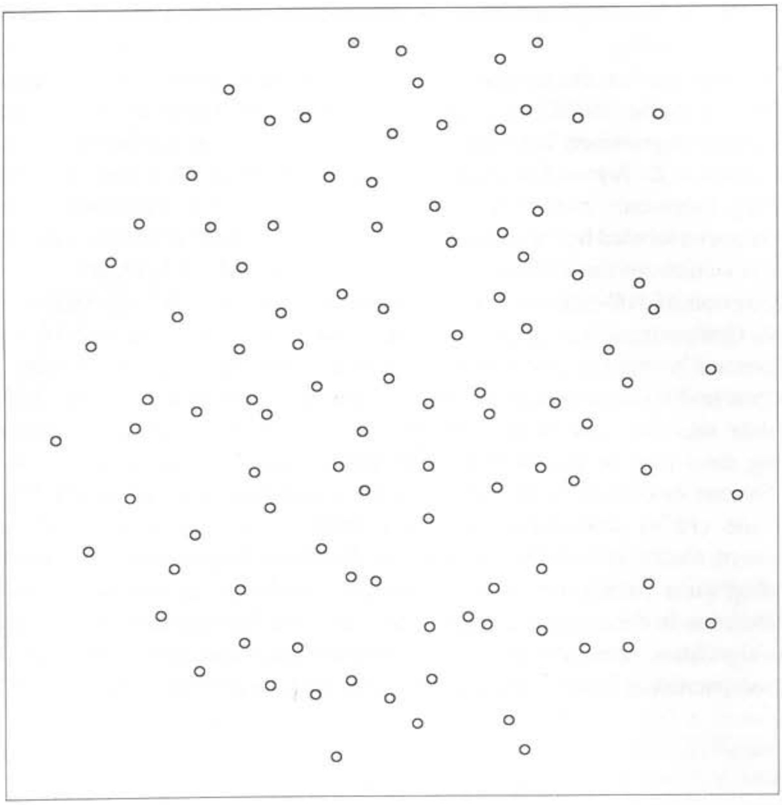


Figure 2.14 Projection of DATA2 by Sammon's method.

or centers of clusters of patterns, to two dimensions and drawing circles around each projected archetype to indicate the spread of the cluster. Such a representation requires an initial clustering of the patterns and projects cluster centers, not patterns, but does save a significant amount of computation. Schachter (1978) proposed a similar modification and used it to project multiband imagery data.

The gradient descent procedure in Sammon's algorithm moves all patterns in the configuration space simultaneously to minimize the square error. Chang and Lee (1973) suggested minimizing the square error by moving the patterns two at a time. We have found that the amount of computation of this procedure becomes prohibitive, even when only a moderate number of patterns are involved, and the final projection depends on the order in which the patterns are paired (Biswas et al., 1981). The *frame method*, also proposed by Chang and Lee, overcomes the computational problem. This heuristic method defines a frame from a representative number, n', of patterns and creates a two-dimensional configuration of the frame. The remaining $(n - n')$ patterns are then projected one by one, adjusting their distances only with respect to this fixed frame. The frame method achieves considerable savings in computation time and memory requirements because the interpattern distances among the $(n - n')$ patterns are not maintained. While Sammon's algorithm tries to preserve all $n(n - 1)/2$ interpattern distances, the frame method attempts to preserve only $[n'(n' - 1)/2 + n'(n - n')]$ interpattern distances. Data can be first clustered and a few patterns from each of the clusters can be selected to form a frame (Biswas et al., 1981).

2.5.3 Triangulation

The algorithms discussed above are iterative and seek to preserve the global structure of the d-dimensional patterns. The *triangulation method* (Lee et al., 1977) exactly preserves only local structure in the data. The patterns are represented in the plane sequentially in such a way that exactly $(2n - 3)$ of the $n(n - 1)/2$ possible interpattern distances are preserved. Whenever a point representing a pattern is placed in the plane, its distance to two patterns represented previously can be exactly preserved: hence the name "triangulation method." This approach can easily be extended to projection onto an m-dimensional space with the corresponding increase in the number of interpattern distances which can be preserved exactly.

The triangulation method preserves the $(n - 1)$ distances in the minimum spanning tree (MST, Appendix G) of the n d-dimensional patterns. That is, the MST of the patterns in the d-dimensional space is the same as the MST of the projected patterns in two dimensions. This is a desirable property because the perceptual grouping in a set of patterns is based in part on the structure inferred from its MST (Zahn, 1971). The remaining $(n - 2)$ distances can be preserved by either one of the following two approaches. In the *reference point approach*, a reference pattern is chosen and the distances from $(n - 2)$ patterns to this reference are preserved. In the *second-nearest-neighbor approach*, the distances from each pattern to its first two nearest neighbors among the patterns already projected are preserved.

The triangulation algorithm is much faster than Sammon's algorithm and the frame method since it exactly preserves only two interpattern distances in projecting each pattern (Biswas et al., 1981). On the other hand, the triangulation method may have the undesirable effect of ignoring global structure in the data, which is well preserved by algorithms that minimize square-error. The configuration generated by the triangulation method is sensitive to the choice of the reference pattern and the order in which the patterns are projected. Biswas et al. (1981) propose a new mapping algorithm which combines the desirable characteristics of the triangulation method and Sammon's method. They also compare several of these projection algorithms and point out their relative merits and drawbacks. Wismath et al. (1981) have used the triangulation method to select subsets of the features. Each of the d features is assumed to be represented as a point in an n-dimensional space, where n is the number of patterns. The pairwise similarity between the d points is 1 minus the square of the correlation coefficient. The features are projected to a two-dimensional space and visual clustering of the features determines redundant features.

2.5.4 Other Methods

Square error, or stress between the original and the projected configuration, is not the only mapping criterion that has been examined. For example, Shepard and Carroll (1966) define a measure of continuity between two spaces as a means of projection that is used by Calvert (1970). Many ideas of this sort are closely related to notions of intrinsic dimensionality (Section 2.6). Wang et al. (1983) require that the MST of the projected patterns be topologically similar to the original MST. A measure of the overall similarity between the two MSTs is the ratio of the number of common edges in the MSTs to the total number of MST edges.

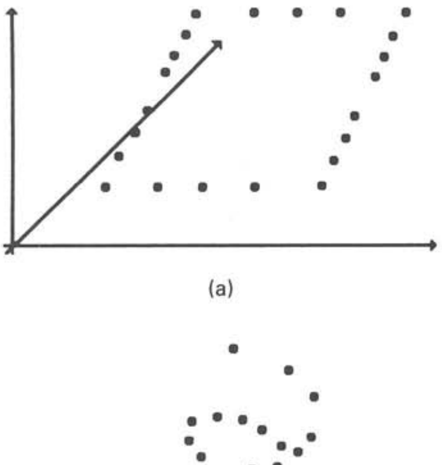
Another projection algorithm that deserves attention is the projection pursuit algorithm (Friedman and Tukey, 1974). The principal component method is a special case of projection pursuit. This linear algorithm overcomes the tendency of the principal component method to be sensitive only to the global property of the data. The projection pursuit algorithm defines a projection index for each direction in the multidimensional space which is a product of the ''trimmed'' standard deviation and the local density of the projected patterns along that direction. This interactive algorithm then finds the direction (or pair of directions for two-dimensional projection) that maximizes this index. The user can decide whether the projection axis or the plane picked out by the algorithm is ''interesting.'' For example, if the objective is to look for some structure or clusters in the data, the user can look at several projections and pick out the best one. The starting direction for the hill-climbing algorithm can include the principal coordinates, the original features, and randomly chosen directions. This algorithm has been found to be useful in detecting certain kinds of clusters present in multidimensional data. Recently, the projection pursuit technique has been extended to other problems (regression, classification, and density estimation) in multivariate data analysis (Huber, 1985).

2.6 INTRINSIC DIMENSIONALITY

The intrinsic, or topological, dimensionality of n patterns in a d-dimensional space refers to the minimum number of ''free'' parameters needed to generate the patterns. Intrinsic dimensionality essentially determines whether the d-dimensional patterns can be described adequately in a subspace of dimensionality less than d. The adjective ''intrinsic'' emphasizes that we seek a property only of the data that does not depend on the dimensionality of the pattern space. For example, d-dimensional patterns along a reasonably smooth curve have intrinsic dimensionality 1, irrespective of the value of d. Similarly, d-dimensional patterns that define a plane or a smooth surface are defined to have intrinsic dimensionality 2. The notion of intrinsic dimensionality differs substantially from that of linear dimensionality of a subspace containing all the n patterns. The linear dimensionality is a global property of the data and usually refers to the number of significant eigenvalues of the covariance matrix of the data. Another notion of dimensionality is described by fractal dimensions, which have become very popular in computer graphics (Mandelbrot, 1977).

Figure 2.15(a) shows 22 points represented in the x-y plane of three-dimensional Euclidean space. The intrinsic dimensionality is 1 and the linear dimensionality is 2. The intrinsic dimensionality of 20 points that lie on a helix, shown in Figure 2.15(b), is 1, whereas the linear dimensionality is 3. Noise in the data can mask the true shape of the underlying surface. In the presence of measurement noise, a curve may be replaced with a ''fat'' tube and the estimated intrinsic dimensionality may change from 1 to 2.

Intrinsic dimensionality is an important characteristic of a data set since it can determine an appropriate number of features for representing the data. A knowledge of intrinsic dimensionality can provide a goal for dimensionality reduction as well as determine an appropriate number of dimensions for the mapping



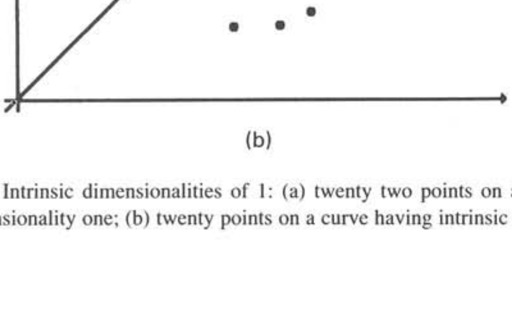


Figure 2.15 Intrinsic dimensionalities of 1: (a) twenty two points on a plane having intrinsic dimensionality one; (b) twenty points on a curve having intrinsic dimensionality one.

methods discussed in Sections 2.4 and 2.5. The notion of intrinsic dimensionality has also been used to determine the number of parameters needed to characterize a complex system when the response of the system is available for a limited number of inputs (Schwartzmann and Vidal, 1975).

There are two primary approaches to estimating the intrinsic dimensionality of a given pattern matrix. The first approach is dynamic in nature and attempts to "unfold" the data, or flatten the swarm of patterns in the d-dimensional space. An estimate of intrinsic dimensionality can then be obtained by finding the linear dimensionality from the number of significant eigenvalues of the covariance matrix of the flattened data (Section 2.4). This method for estimating intrinsic dimensionality is global in nature and does not always work well, especially when the output of the data-flattening step is highly warped (Pettis et al., 1979). The second approach does not move the patterns but estimates the intrinsic dimensionality directly from information in a local neighborhood of patterns. This is also referred to as a static approach. Some details are given below. Wyse et al. (1980) summarize and compare algorithms for intrinsic dimensionality.

2.6.1 Global Approach

Bennett (1969) estimated the intrinsic dimensionality globally. His method is based on the observation that the variance of the distance between points chosen randomly in a hypersphere is inversely proportional to the dimensionality. Let $\mathbf{X}_1$ and $\mathbf{X}_2$ be independent, vector-valued random variables having uniform distributions inside a sphere of radius r in a d-dimensional space. The normalized Euclidean distance between $\mathbf{X}_1$ and $\mathbf{X}_2$ is given by

$$D = \frac{\left[\sum_{i=1}^{d} (X_{1i} - X_{2i})^2\right]^{1/2}}{2r}$$

The probability density function for D (Alagar, 1976; Hammersley, 1950) is given by

$$f_D(z) = 2^d d z^{(d-1)} I_{1-z^2}\left(\frac{d+1}{2}, \frac{1}{2}\right)$$

where $I_a(p, q)$ is the incomplete beta function. If var (D) denotes the variance of D, Bennett shows that

$$d \times \text{var}(D) \approx \text{constant}$$

This relationship has been established only for points uniformly distributed inside a hypersphere and may not be true for other distributions.

Bennett's algorithm utilizes the foregoing property to "flatten" the swarm of patterns. First the patterns (in the original pattern space) are perturbed to increase the variance of the interpoint distances, thereby reducing d. Then the positions of the patterns are adjusted to preserve the rank orders of interpoint distances in local regions. These two processes are repeated, with suitable normalizations, until there is no significant increase in the variance of the interpoint distances. The intrinsic dimensionality is determined by the number of significant eigenvalues of the covariance matrix of the flattened data (Section 2.4.1). The success of Bennett's algorithm depends on three parameters that must be set heuristically.

The main drawback of Bennett's algorithm is that local structure is not preserved very well during the unfolding process. Local structure is preserved by penalizing misrankings of local interpoint distances more heavily than for distances between distant points. Chen and Andrews (1974) introduced a cost function to make Bennett's rank-order criterion more sensitive to local regions. The cost function controls the extent to which the rank ordering of the interpoint distances is violated. A zero cost function imposes a very rigid requirement on the movement of points and the data are flattened very little. Chen and Andrews's algorithm also requires the heuristic determination of several parameters.

The process of unfolding the data needs to be constrained by the neighborhood over which local structure can be preserved. Schwartzmann and Vidal (1975) require that the minimum spanning tree (MST) of the patterns be preserved during the linearization of the data. A ''barycentric'' transformation replaces each pattern by a weighted average (called a barycenter) of the pattern and all the patterns connected to it in the MST. The barycentric transformation smoothes the original MST and shortens the total length of the MST, so a uniform scaling is applied to restore the original MST length. The barycentric transformation is applied repeatedly until either the connectivity pattern of the MST is violated or the variance of the interpoint distances stabilizes. Once again, the actual intrinsic dimensionality is determined from the number of significant eigenvalues of the covariance matrix. One problem with this approach is that the MST often breaks during the first iteration on real data sets, before any significant smoothing takes place. A possible remedy is to continue the algorithm until a significant number of breaks occur in the MST. Srivastava (1973) proposed another method based on entropy.

2.6.2 Local Approach

The global nature of the eigenvalue method for estimating intrinsic dimensionality have led to three methods for maintaining local properties of the data set. These methods do not generate a configuration of points or project the patterns to a space of few dimensions. Instead, they simply estimate an appropriate number of dimensions for representing the data. Fukunaga and Olsen (1971) partition the pattern space and estimate eigenvalues in local regions rather than computing global eigenvalues. The numbers of significant eigenvalues of covariance matrices computed over each local region are determined and a summary table is formed which indicates the number of regions corresponding to each dimensionality. The procedure is interactive, with the final choice of dimensionality left to the judgment of the investigator. The size of local neighborhoods can substantially affect the estimate of intrinsic dimensionality.

Pettis et al. (1979) propose a straightforward estimator for intrinsic dimensionality which is based on near-neighbor information. Let $r_{k,\mathbf{x}}$ be the distance from pattern **x** to its *k*th nearest neighbor. Pettis et al. (1979) model the local properties of the data by a Poisson, or purely random, spatial point process (Section 4.6) to obtain the density function for $r_{k,\mathbf{x}}$. Define a sample-averaged distance to the *k*th nearest neighbor over the set of patterns as

$$\bar{r}_k = (1/n) \sum_{i=1}^{n} r_{k,\mathbf{x}_i}$$

It can be shown that a plot of log $(\bar{r}_k)$ as a function of log (k) has a slope of $(1/\hat{d})$, where $\hat{d}$ is an estimator of the intrinsic dimensionality. Pettis et al. (1979) solve for $\hat{d}$ iteratively by fitting a least-squares regression line to a plot of log $(\bar{r}_k)$ versus log (k) for $k = 1, \ldots, K$. In the special case when only the *k*th and $(k + 1)$st near neighbors are used, the estimate of the intrinsic dimensionality can be written as

$$\hat{d} = \frac{\bar{r}_k}{k(\bar{r}_{k+1} - \bar{r}_k)}$$

This method works well in identifying the true dimensionality for a variety of data sets and is fairly insensitive to the number of samples and to the algorithmic parameters. For example, the intrinsic dimensionalities of two ''surface'' data sets, uniformly distributed points on the surface of a sphere and uniformly distributed points on a circle, were correctly identified as 2 and 1, respectively, even for $K = 2$ and number of patterns $n = 20$. The global eigenvalue method will give estimates of 3 and 2, respectively, for intrinsic dimensionality of these two data sets.

Trunk's method (1976) also uses the near-neighbor information to estimate the intrinsic dimensionality. He computes the average of the angles between the $(k + 1)$st nearest neighbor of $\mathbf{x}_i$ and the subspace that spans the vectors from $\mathbf{x}_i$ to its k nearest neighbors. This average angle is compared against some thresholds which are known only for intrinsic dimensionalities of 1, 2, and 3.

2.7 MULTIDIMENSIONAL SCALING

Multidimensional scaling is a generic name for a body of procedures and algorithms that start with an ordinal proximity matrix and generate configurations of points in one, two or three dimensions. The objectives of multidimensional scaling are the same as those of the iterative procedures for nonlinear projection in Section 2.5.2. However, those methods begin with a distance matrix that contains ratio-type data. Multidimensional scaling translates an ordinal scale to a set of ratio scales and is an example of *ordination*. The configuration of points provides a concrete model of the proximity matrix that can be observed and interpreted.

The literature of multidimensional scaling is large and diverse, with almost all theoretical developments and applications being in the behavioral and social sciences. Most engineering applications begin with patterns in a pattern space, so metric properties can be exploited when projecting the patterns to a low-dimensional space (Sections 2.4 and 2.5). Several excellent surveys of multidimensional scaling methodology and practice exist (Green and Carmone, 1970; Kamenskii, 1977; Kruskal and Carroll, 1969; Romney et al., 1972; Shepard et al., 1972; Shepard, 1974; Young, 1970). In this section we describe a basic multidimensional scaling algorithm and mention a few applications. Our goal here is to clarify the relation between multidimensional scaling and cluster analysis (Kruskal, 1977b).

Kruskal (1977a) explains that Torgerson (1958) invented the name multidimensional scaling. Shepard's (1962) pioneering work on the analysis of (ordinal) proximities and Kruskal's (1964a,b) development of MDSCAL, a widely distributed FORTRAN program for multidimensional scaling, popularized this approach to data analysis and established it as an important tool in behavioral and social science research. Since the objective of a multidimensional scaling method is to create a set of scales, or dimensions, that represent the data, natural links exist between multidimensional scaling, intrinsic dimensionality (Section 2.6), and nonlinear projections (Section 2.5). These links should become apparent in the following outline of Kruskal's (1964a,b) MDSCAL technique.

2.7.1 The Basic MDSCAL Algorithm

We begin with the $n \times n$ ordinal proximity matrix of dissimilarities $[d(i, j)]$ and search for an m-dimensional configuration of points $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, where

$$\mathbf{x}_i = [x_{i1} \quad x_{i2} \quad \cdots \quad x_{im}]^{\mathrm{T}}$$

for which the rank order of the distances $[D(i, j)]$ in the configuration "match" the proximities. The measure of distance in the configuration space is the Minkowski metric (Section 2.2.1), or

$$D(i, j) = \left(\sum_{k=1}^{m} |x_{ik} - x_{jk}|^r \right)^{1/r}, \qquad r \geq 1$$

A perfect match occurs when the rank orders of the entries in the matrix $[D(i, j)]$ match those in the matrix $[d(i, j)]$. The degree to which the two sets of rank orders agree is measured by Kruskal's *stress*, which resembles the square error E (Section 2.5.2). Before defining stress, we consider briefly the problem of embedding points in a space.

It is clear that two points can be placed on a line to match the dissimilarity between two objects. Three points in a metric space define a plane, so a configuration of three points can always be defined in two dimensions in such a way that interpoint distances exactly match the dissimilarities among three objects. In fact, n points in a metric space can always be embedded in an $(n-1)$-dimensional space so as to exactly recreate the proximities among objects. Shepard (1962) shows that an $(n-1)$-dimensional configuration exists in which the rank order of the distances matches that of the given ordinal proximities. Lingoes (1971) provides conditions under which only $(n-2)$ dimensions are needed for this type of match. Cunningham and Shepard (1974) study the embedding problem in some detail. It is clear that if we are to represent even 10 patterns faithfully in two dimensions, the proximities must be interrelated in a structured fashion. Levine (1978) suggests using stress to evaluate the randomness in the data.

To define stress, we begin with a fixed configuration of points in m dimensions with interpoint distance matrix $[D(i,j)]$. Each point represents an object in the given ordinal dissimilarity matrix $[d(i,j)]$. Label the rank-ordered distances as

$$D(i_1, j_1) \le D(i_2, j_2) \le \cdots \le D(i_M, j_M)$$

and label the rank-ordered dissimilarities correspondingly as

$$[d(i_1, j_1), d(i_2, j_2), \ldots, d(i_M, j_M)]$$

where $M = n(n-1)/2$ is the number of interpoint distances and dissimilarities. Stress can be understood in terms of a Shepard diagram, which is a plot of M points, each representing the (distance, dissimilarity) values for one pair of patterns. The abscissa of the Shepard diagram is distance and the ordinate is dissimilarity. The ordinal dissimilarities are spaced evenly on the y-axis.

The match between dissimilarity matrix and distance matrix is perfect if the M points can be connected by a sequence of straight lines all having nonnegative slopes. Consider first any curve consisting of a connected sequence of straight lines, all segments of which have nonnegative slope. Let $\hat{D}(i,j)$ be the abscissa of the intersection of a horizontal line through ordinate $d(i,j)$ with the curve. Then $|D(i,j) - \hat{D}(i,j)|$ measures, in distance units, the amount by which $D(i,j)$ is off the curve. The stress for such a curve is defined below and indicates how well the rank orders of the dissimilarities among objects match the distances among points in the configuration. The stress equation involves only the x-axis values because they are on a ratio scale whereas y-axis values are on an ordinal scale.

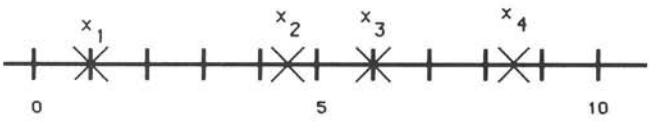
$$\text{Stress(curve)} = \left[\frac{\sum\sum_{i<j} |D(i,j) - \hat{D}(i,j)|^2}{\sum\sum_{i<j} D^2(i,j)} \right]^{1/2}$$

Example 2.8

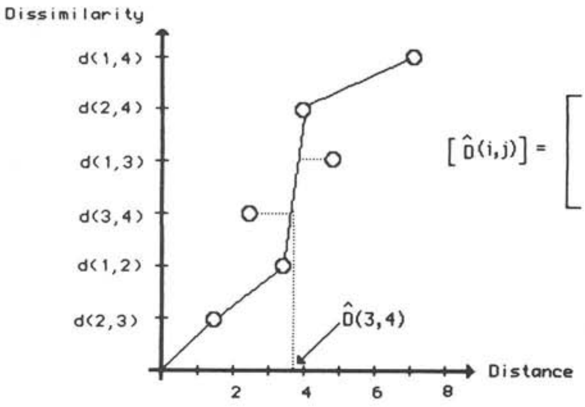
A 4×4 rank-order dissimilarity matrix $[d(i,j)]$ is given below. Figure 2.16(a) shows a configuration of four points in one dimension and the corresponding Shepard diagram. The sequence of straight lines in Figure 2.16(b) was drawn to minimize stress for the configuration. The points on the Shepard diagram corresponding to interpattern dissimilarities $d(2,3)$, $d(1,2)$, $d(2,4)$, and $d(1,4)$ lie on the connected sequence of straight lines having

$$[D(i,j)] = \begin{bmatrix} - & 3.5 & 5.0 & 7.5 \\ - & - & 1.5 & 4.0 \\ - & - & - & 2.5 \\ - & - & - & - \end{bmatrix}$$

Interpoint Distance Matrix
From configuration below



(a)



$$[\hat{D}(i,j)] = \begin{bmatrix} - & 3.5 & 3.83 & 7.5 \\ - & - & 1.5 & 4.0 \\ - & - & - & 3.67 \\ - & - & - & - \end{bmatrix}$$

$$\text{Stress(curve)} = \left[\frac{(1.17)^2 + (1.17)^2}{(3.5)^2 + (5.0)^2 + (7.5)^2 + (1.5)^2 + (4.0)^2 + (2.5)^2} \right]^{\frac{1}{2}}$$
$$= 0.152$$

(b)

Figure 2.16 Shepard diagram: (a) configuration of four points; (b) Shepard diagram and stress computation for one curve.

positive slope. The entries of the matrix $[\hat{D}(i,j)]$ are found from intersections between lines of constant dissimilarity and the straight-line segments.

$$[d(i,j)] = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & — & 2 & 4 & 6 \\ 2 & — & — & 1 & 5 \\ 3 & — & — & — & 3 \\ 4 & — & — & — & — \end{array}$$

The sequence of straight lines in Figure 2.16(b) solves the monotone regression problem. In other words, the lines define an ascending, monotonic function f that minimizes

$$\sum_{i=1}^{3} \sum_{j=i+1}^{4} [d(i,j) - f(D(i,j))]^2$$

This minimum value, 0.152, is the stress for the configuration. The MDSCAL procedure seeks the configuration whose stress is minimum.

The MDSCAL program begins with a randomly chosen configuration of n points, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ in m dimensions, where m is a fixed integer between 1 and 6 and moves the points so as to minimize stress. The value of m is usually 2, so that the final configuration can be plotted on paper. Configurations in $m > 2$ dimensions are often projected to two dimensions (Section 2.4). Stress is pictured as a function of the nm coordinates of the n points in the configuration. A gradient descent, or steepest descent, procedure is applied. The jth coordinate of the ith point, x_{ij}, is updated according to

$$x_{ij} \leftarrow x_{ij} + \frac{\alpha}{|g|} g_{ij}$$

where g_{ij} is a numerical approximation to the negative partial derivative of stress with respect to x_{ij} and

$$|g| = \left(\sum_{i<j} \sum g_{ij}^2 \right)^{1/2}$$

The step size, α, is changed at each iteration by an empirical formula. The iteration stops when the magnitude of the gradient, $|g|$, is small. An outline of the MDSCAL algorithm is listed below.

ALGORITHM FOR MULTIDIMENSIONAL SCALING

Step 1. Create a random configuration of n points in m dimensions. Normalize the configuration.

Repeat steps 2 to 5 until $|g|$ is "small."

Step 2. Compute the $n \times n$ distance matrix $[D(i,j)]$.

Step 3. Compute the stress for the configuration.

Step 4. Compute $|g|$, the magnitude of the gradient.

Step 5. Compute the step size and update the configuration. Normalize the configuration.

The MDSCAL program has several options, such as the procedure for minimizing stress, alternative stress formulas, provisions for missing data, and various stopping criteria that are not discussed here. The main outputs of the program are the configuration itself and the final stress. All configurations are normalized (step 1) by Eq. (2.2). The final Shepard diagram and various statistics are also generated.

2.7.2 Interpreting MDSCAL Configurations

How does one determine whether or not the interpoint distances in a configuration are reasonably accurate reflections of the ordinal proximities? Answering this question will also help establish a minimal dimensionality for the configuration. Kruskal (1964a,b) suggests a threshold of 0.05 for the stress of a "good" configuration. Some confusion results from the existence of two formulas for stress which treat ties in the proximity matrix differently. Since the stress of a minimum-stress configuration tends to decrease as the dimensionality of the configuration increases, one can run the MDSCAL program for increasing dimensionalities, m, and look for a "knee" in the plot of final stress versus m, which indicates an abrupt decrease in stress. This prevents choosing an unnecessarily large dimensionality.

Several Monte Carlo studies of the distribution of stress have been conducted. The three studies reported here generate empirical distributions for stress, given the dimensionality (m) and the number of points (n), under the null hypothesis that all (ordinal) proximity matrices are equally likely. The entries in each proximity matrix are chosen independently and at random from a uniform distribution. Since only the rank order of the proximities is used in MDSCAL, this is equivalent to the random graph hypothesis (Section 4.1.1). Stenson and Knoll (1969) computed the final stress for configurations of 10, 20, 30, 40, 50, and 60 points, each with configuration dimensionalities of 1, 2, . . . , 10. Only three repetitions were run and the average stress was plotted against m for each n. Klahr (1969) ran this experiment for 6, 7, 8, 10, 12, and 16 points with dimensionalities of 1, 2, 3, 4, and 5; 100 Monte Carlo trials were employed for the four smallest numbers of points and 50 were used for the two largest numbers of points. Klahr (1969) found several "good" solutions (stress < 0.05) for 6, 7, and 8 points in three dimensions, but none of the configurations yielded "good" stress for more than 10 points. Klahr (1969) also showed that the "elbow rule," discussed above, yielded spurious results for small numbers of points. Levine (1978) repeated this experiment. These stress distributions should provide a better indication of acceptable stress than Kruskal's "5% rule."

The relation between n, the number of points, and m, the dimensionality of the configuration, is an important consideration when interpreting an MDSCAL configuration. If n is not much larger than m, finding a low-stress configuration is relatively easy. The configuration is "loose" in the sense that moving the points a bit should not alter the stress greatly and several configurations will have nearly minimum stress. When n is much larger than m, say by a factor of at least 5, the fit is "tight." A low-stress configuration is more trustworthy when n is large than when n is small.

A second important issue is that the MDSCAL program is designed to capture global, not local, structure. For example, when the patterns are clustered into a few tight clusters, the MDSCAL configuration will reproduce the dissimilarities between the clusters, but the distances among patterns within clusters might not have much meaning. The reason for this behavior is that stress weights large distances more heavily than small ones. This fact also permits a few outliers to seriously distort the configuration. Thus a preliminary data analysis that removes outliers and checks for strong clustering effects might be needed (see Kruskal, 1977a).

The "local minimum" problem can be irritating in practice. The stress function depends on *nm* vairables and has several relative minima in the search space. Steps are taken in MDSCAL to ignore local minima when seeking a global minimum, but the program can get trapped and stop with an unreasonably poor configuration. Thus MDSCAL should be run several times with different randomly chosen starting configurations.

The axes in the final configuration can sometimes be named from pattern class labels on the points in the configuration. If, for example, the points represent political figures as seen by a subject and if conservatives tend to fall on the north side of a two-dimensional configuration and liberals on the south side, the north/south axis might be named "ideology." Similarly, urban politicians on the east side of the configuration and rural politicians on the west side could lead to a name "urbanization" for the other axis. In this case, one might conclude, assuming that the two-dimensional configuration had low stress, that the subject views politicians by ideology and constituency. Of course, there is no assurance that one will be able to interpret the axes. Another way of interpreting the configuration is to impose a hierarchical clustering (Section 3.2) on the configuration and studying the clusters.

Multidimensional scaling has been applied in numerous circumstances (Romney et al., 1972). For example, Levine (1977b) provides a straightforward application in a study of the way undergraduates perceive sports. Kruskal and Hart (1966) reported an application of multidimensional scaling to the analysis of 50,000 possible computer malfunctions and the results of 1000 pass/fail tests for each malfunction.

Example 2.9

Levine (1977a) used multidimensional scaling to study the stimulus-response data described in Example 2.2. The original asymmetric confusion matrix was converted to a symmetric matrix. The two-dimensional solution obtained by MDSCAL algorithm is shown in Figure 2.17. The horizontal axis in Figure 2.17 separates purely straight numbers (1 and 4) from the curved numbers. The vertical axis in the two-dimensional configuration is named "locus of curvature." Additional results obtained with hierarchical clustering suggest four major groups in the 10 digits: purely straight numbers (1 and 4), combined straight and curved numbers (2 and 7), rounded curved numbers (0, 3, 6, 8), and complex curved numbers (5 and 9). The stress values in one, two, three, and four dimensions are 0.334, 0.207, 0.117, and 0.033, respectively. To avoid local minima problems the MDSCAL algorithm was run with 10 different starting configurations.

The basic idea of multidimensional scaling has been generalized and extended in many ways. We mention a few here. Ramsay (1978) studies the process of defining an ellipse about each point of the configuration which defines the region that contains the "true" population point with a confidence of 0.95. Carroll and

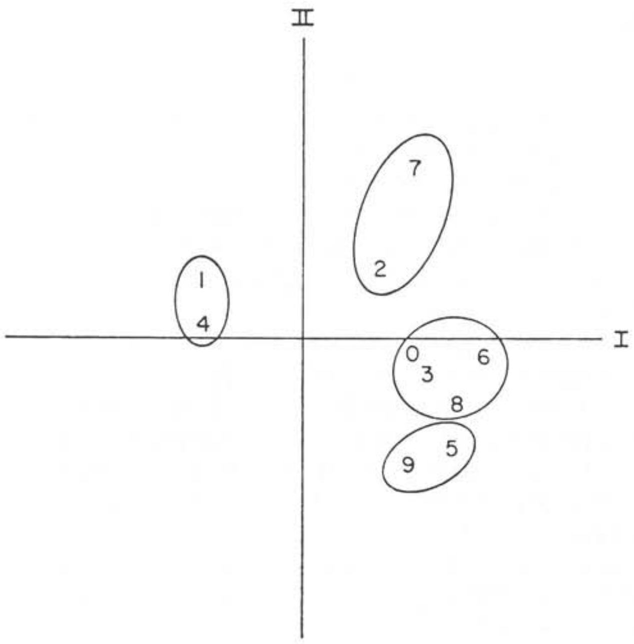


Figure 2.17 Two-dimensional solution obtained by MDSCAL analysis of the 10 numbers. (Source: David M. Levine, "Multivariate Analysis of the visual information processing of numbers," *Journal of Multivariate Behavioral Research* 12, 1977, 347–355.)

Wish (1971) introduced a program called INDSCAL (Individual Differences Scaling) which analyzes proximity matrices from several subjects simultaneously by trying to generate a configuration faithful to the individual subjects as well as to the group of subjects. Kruskal (1977a) provides computational details. Tzeng and Landis (1978) suggest some generalizations. Bentler and Weeks (1978) probe ways of fitting a model to the data by estimating parameters of the model during multidimensional scaling. Cunningham and Shepard (1974) search for a functional relationship between dissimilarities and distances. Hubert and Busk (1976) attack the problem of adding points to an existing configuration. Lindman and Caelli (1978) use Riemannian spaces rather than Minkowski metrics in multidimensional scaling. MacCallum (1974) relates multidimensional scaling to factor analysis. MacCallum and Cornelius (1977) compare MDSCAL and INDSCAL to ALSCAL, another procedure for modeling proximity data, and provide a Monte Carlo study of the accuracy with which ALSCAL recovers data structure. Ramsay (1977) proposes an alternative to multidimensional scaling that fits a log-normal distribution to the proximities themselves. Kakusho and Mizoguchi (1983) have extended nonmetric multidimensional scaling by defining a new local criterion function which requires that the first K nearest neighbors of each object in the original proximity matrix remain nearest neighbors in the configuration.

2.8 SUMMARY

Several factors that must be appreciated during a cluster analysis have been reviewed in this chapter. The form, type, and scale of data are often dictated by the problem domain and measuring instruments, but one should realize the influences of these factors on clustering options. The properties of the proximity index chosen must be readily understood and accepted in the area of application. The type of normalization applied to the data depends on the manner in which the data are recorded. Our tutorial exposition of the eigenvector projection reflects the key role of this linear transformation.

One way of exploring data is to look at it. Thus we have discussed several linear and nonlinear projections in Sections 2.4 and 2.5, along with ways of picturing multivariate data. Intrinsic dimensionality algorithms are often ignored in clustering applications, but establishing an appropriate number of factors for representing data is a persistent problem. Our treatment in Section 2.6 is more a guide to the literature than a handbook of practical procedures.

Multidimensional scaling, treated briefly in Section 2.7, may be considered a nonlinear representation technique and a means for estimating intrinsic dimensionality. It is unique in its starting point—an ordinal proximity matrix. A wide variety of scaling procedures has been developed in the psychometric literature that should interest researchers in other areas. The information gleaned from representing the data in various ways and studying its intrinsic dimensionality supplements and supports a cluster analysis and can even make a cluster analysis unnecessary.

3

Clustering Methods and Algorithms

Cluster analysis is the process of classifying objects into subsets that have meaning in the context of a particular problem. The objects are thereby organized into an efficient representation that characterizes the population being sampled. In this chapter we present the clustering methods themselves and explain algorithms for performing cluster analysis. Section 3.1 lists the factors involved in classifying objects, and in Sections 3.2 and 3.3 we explain the two most common types of classification. Computer software for cluster analysis is described in Section 3.4. Section 3.5 outlines a methodology for using clustering algorithms to the best advantage. This chapter focuses on the act of clustering itself by concentrating on the inputs to and outputs from clustering algorithms. The need for the formal validation methods in Chapter 4 will become apparent during the discussion.

3.1 GENERAL INTRODUCTION

A clustering is a type of classification imposed on a finite set of objects. As explained in Section 2.2, the relationship between objects is represented in a proximity matrix in which rows and columns correspond to objects. If the objects are characterized as patterns, or points in a d-dimensional metric space, the proximities can be distance between pairs of points, such as Euclidean distance. Unless a meaningful measure of distance, or proximity, between pairs of objects has been established, no meaningful cluster analysis is possible. The proximity matrix is the one and only input to a clustering algorithm.

Clustering is a special kind of classification. See Kendall (1966) for discussion on the relationship between classification and clustering. Figure 3.1 shows a tree of classification problems as suggested by Lance and Williams (1967). Each leaf in the tree in Figure 3.1 defines a different genus of classification problem. The nodes in the tree of Figure 3.1 are defined below.

a. *Exclusive versus nonexclusive.* An exclusive classification is a partition of the set of objects. Each object belongs to exactly one subset, or cluster. Nonexclusive, or overlapping, classification can assign an object to several classes. For example, a grouping of people by age or sex is exclusive, whereas a grouping by disease category is nonexclusive because a person can have several diseases simultaneously. Shepard and Arabie (1979) provide a review of nonexclusive or overlapping clustering methods. This chapter treats only exclusive classification. Fuzzy clustering is a type of nonexclusive classification in which a pattern is assigned a degree of belongingness to each cluster in a partition and is explained in Section 3.3.8.

b. *Intrinsic versus extrinsic.* An intrinsic classification uses only the proximity matrix to perform the classification. Intrinsic classification is called "unsupervised learning" in pattern recognition because no category labels denoting an a priori partition of the objects are used. (See Appendix A for an introduction to pattern recognition.) Extrinsic classification uses category labels on the objects as well

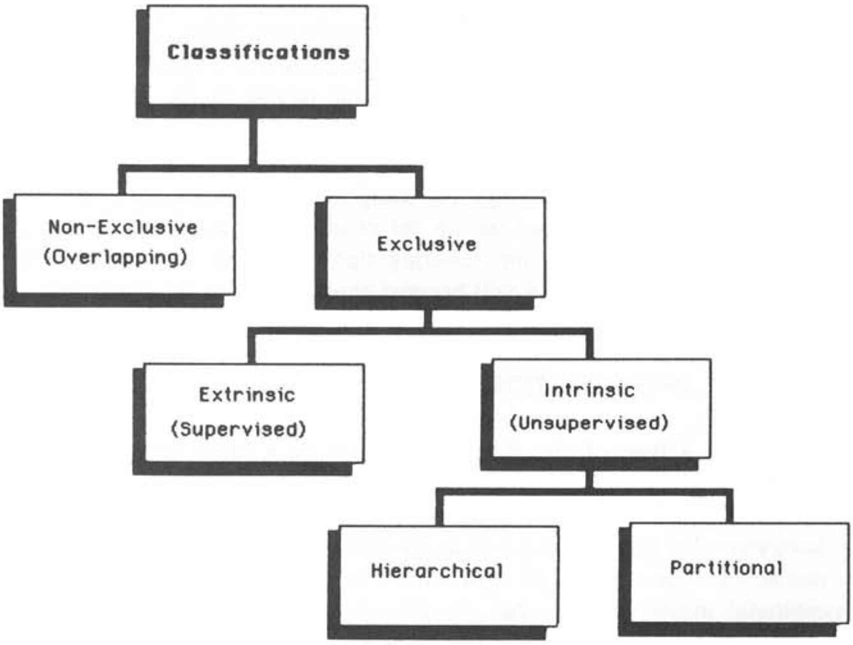


Figure 3.1 Tree of classification types.

as the proximity matrix. The problem is then to establish a discriminant surface that separates the objects according to category. In other words, an extrinsic classifier relies on a "teacher," whereas an intrinsic classifier has only the proximity matrix.

One way to evaluate an intrinsic classification is to see how the cluster labels, assigned to objects during clustering, match the category labels, assigned a priori. For example, suppose that various indices of personal health were collected from smokers and nonsmokers. An intrinsic classification would group the individuals based on similarities among the health indices and then try to determine whether smoking was a factor in the propensity of individuals toward various diseases. An extrinsic classification would study ways of discriminating smokers from nonsmokers based on health indices. We are concerned only with intrinsic classification in this book; intrinsic classification is the essence of cluster analysis.

c. *Hierarchical versus partitional*. Exclusive, intrinsic classifications are subdivided into hierarchical and partitional classifications by the type of structure imposed on the data. A hierarchical classification is a nested sequence of partitions and is explained in Section 3.2, whereas a partitional classification is a single partition and is defined in Section 3.3. Thus a hierarchical classification is a special sequence of partitional classifications. We will use the term *clustering* for an exclusive, intrinsic, partitional classification and the term *hierarchical clustering* for an exclusive, intrinsic, hierarchical classification. Sneath and Sokal (1973) apply the acronym SAHN (Sequential, Agglomerative, Hierarchical, Nonoverlapping) to exclusive, intrinsic, hierarchical, agglomerative algorithms. The differences and similarities between algorithms for generating these two types of classifications are the topics of this chapter.

Several algorithms can be proposed to express the same exclusive, intrinsic classification. One frequently uses an algorithm to express a clustering method, then examines various computer implementations of the method. The primary algorithmic options in common use are explained below.

1. *Agglomerative versus divisive*. An agglomerative, hierarchical classification places each object in its own cluster and gradually merges these atomic clusters into larger and larger clusters until all objects are in a single cluster. Divisive, hierarchical classification reverses the process by starting with all objects in one cluster and subdividing into smaller pieces. Thus this option corresponds to a choice of procedure rather than to a different kind of classification. Partitional classification can be characterized in the same way. A single partition can be established by gluing together small clusters (agglomerative) or by fragmenting a single all-inclusive cluster (divisive).
2. *Serial versus simultaneous*. Serial procedures handle the patterns one by one, whereas simultaneous classification works with the entire set of patterns at the same time (see Clifford and Stephenson, 1975).
3. *Monothetic versus polythetic*. This option is most applicable to problems in taxonomy, where the objects to be clustered are represented as patterns, or

points in a space. A monothetic clustering algorithm uses the features one by one, whereas, a polythetic procedure uses all the features at once. For example, a different feature can be used to form each partition in a hierarchical classification under a monothetic algorithm. We will consider only polythetic algorithms.

4. *Graph theory versus matrix algebra*. What is the appropriate mathematical formalism for expressing a clustering algorithm? We will express some algorithms in terms of graph theory, using properties such as connectedness and completeness to define classifications, and express other algorithms in terms of algebraic constructs, such as mean-square-error. The choice is one of clarity, convenience, and personal choice. When implementing an algorithm on a computer, attention must be paid to questions of computational efficiency. This issue is not related to human understanding of the classification method. Some algorithms have convenient expressions under both options.

3.2 HIERARCHICAL CLUSTERING

A hierarchical clustering method is a procedure for transforming a proximity matrix into a sequence of nested partitions. A hierarchical clustering algorithm is the specification of steps for performing a hierarchical clustering. It is often convenient to characterize a hierarchical clustering method by writing down an algorithm, but the algorithm should be separated from the method itself. In addition to defining algorithms and methods in this section, we define the type of mathematical structure a hierarchical clustering imposes on data and describe ways of viewing that structure.

First comes the notion of a sequence of nested partitions. The n objects to be clustered are denoted by the set $\mathscr{X}$.

$$\mathscr{X} = \{x_1, x_2, \ldots, x_n\}$$

where x_i is the ith object. A partition, $\mathscr{C}$, of $\mathscr{X}$ breaks $\mathscr{X}$ into subsets $\{C_1, C_2, \ldots, C_m\}$ satisfying the following:

$$C_i \cap C_j = \Phi \qquad \text{for } i \text{ and } j \text{ from 1 to } m, \quad i \neq j$$

$$C_1 \cup C_2 \cup \ldots \cup C_m = \mathscr{X}$$

In this notation, "$\cap$" stands for set intersection, "$\cup$" stands for set union, and Φ is the empty set. A clustering is a partition; the components of the partition are called clusters. Partition $\mathscr{B}$ is nested into partition $\mathscr{C}$ if every component of $\mathscr{B}$ is a ~~proper~~ subset of a component of $\mathscr{C}$. That is, $\mathscr{C}$ is formed by merging components of $\mathscr{B}$. For example, if the clustering $\mathscr{C}$ with three clusters and the clustering $\mathscr{B}$ with five clusters are defined as follows, then $\mathscr{B}$ is nested into $\mathscr{C}$. Both $\mathscr{C}$ and $\mathscr{B}$ are clusterings of the set of objects $\{x_1, x_2, \ldots, x_{10}\}$.

$$\mathscr{C} = \{(x_1, x_3, x_5, x_7), (x_2, x_4, x_6, x_8), (x_9, x_{10})\}$$

$$\mathscr{B} = \{(x_1, x_3), (x_5, x_7), (x_2), (x_4, x_6, x_8), (x_9, x_{10})\}$$

Neither $\mathscr{C}$ nor $\mathscr{B}$ is nested into the following partition, and this partition is not nested into $\mathscr{C}$ or $\mathscr{B}$.

$$\{(x_1, x_2, x_3, x_4), (x_5, x_6, x_7, x_8), (x_9, x_{10})\}$$

A hierarchical clustering is a sequence of partitions in which each partition is nested into the next partition in the sequence. An *agglomerative* algorithm for hierarchical clustering starts with the disjoint clustering, which places each of the n objects in an individual cluster. The clustering algorithm being employed dictates how the proximity matrix should be interpreted to merge two or more of these trivial clusters, thus nesting the trivial clustering into a second partition. The process is repeated to form a sequence of nested clusterings in which the number of clusters decreases as the sequence progresses until a single cluster containing all n objects, called the conjoint clustering, remains. A *divisive* algorithm performs the task in the reverse order.

A picture of a hierarchical clustering is much easier for a human being to comprehend than is a list of abstract symbols. A *dendrogram* is a special type of tree structure that provides a convenient picture of a hierarchical clustering. A dendrogram consists of layers of nodes, each representing a cluster. Lines connect nodes representing clusters which are nested into one another. Cutting a dendrogram horizontally creates a clustering. Figure 3.2 provides a simple example. Section 3.2.2 explains the role of dendrograms in hierarchical clustering.

Other pictures can also be drawn to visualize a hierarchical clustering (Kleiner and Hartigan, 1981; Friedman and Rafsky, 1981; Everitt and Nicholls, 1975). Information other than the sequence in which clusterings appear will be of interest. The level, or proximity value, at which a clustering is formed can also be recorded. If objects are represented as patterns, or points in a space, the centroids of the clusters can be important, as well as the spreads of the clusters.

Two specific hierarchical clustering methods are now defined called the *single-link* and the *complete-link* methods. Section 3.2.1 explains algorithms for these two commonly used hierarchical clustering methods. The sequences of clusterings created by these two methods depend on the proximities only through their rank

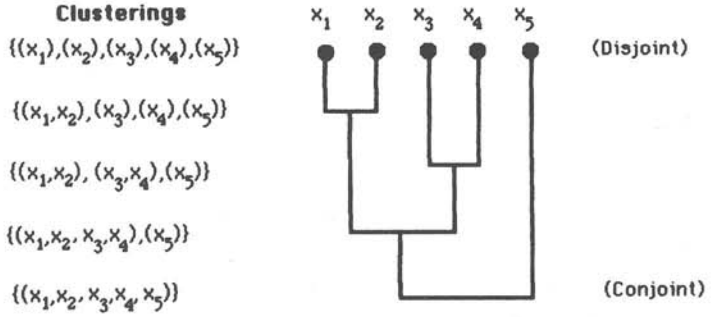


Figure 3.2 Example of dendrogram.

order. Thus we first assume an ordinal scale for the proximities and use graph theory to express algorithms. Single-link and complete-link hierarchical methods are not limited to ordinal data. Sections 3.2.4 and 3.2.5 examine algorithms for these two methods in terms of interval and ratio data. The effects of proximity ties on hierarchical clustering are discussed in Section 3.2.6, while algorithms defined for single-link and complete-link clustering are generalized in Sections 3.2.7 and 3.2.9 to establish new clustering methods. The issues in determining whether or not a hierarchical classification is appropriate for a given proximity matrix are postponed until Chapter 4.

3.2.1 Single-Link and Complete-Link Algorithms from Graph Theory

We begin with a symmetric $n \times n$ proximity matrix $\mathcal{D} = [d(i, j)]$, as defined in Section 2.2. The $n(n-1)/2$ entries on one side of the main diagonal are assumed to contain a permutation of the integers from 1 to $n(n-1)/2$ with no ties. That is, the proximities are on an ordinal scale. We take the proximities to be dissimilarities; $d(1, 2) > d(1, 3)$ means that objects 1 and 3 are more like one another than are objects 1 and 2.

Example 3.1

An example of an ordinal proximity matrix for $n = 5$ is given as matrix $\mathcal{D}_1$.

$$\mathcal{D}_1 = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array} \begin{array}{c} \begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 0 & 6 & 8 & 2 & 7 \\ 6 & 0 & 1 & 5 & 3 \\ 8 & 1 & 0 & 10 & 9 \\ 2 & 5 & 10 & 0 & 4 \\ 7 & 3 & 9 & 4 & 0 \end{bmatrix} \end{array}$$

A *threshold graph* is an undirected, unweighted graph on n nodes without self-loops or multiple edges. Each node represents an object. See Appendix G for a brief review of terms in graph theory. A threshold graph $G(v)$ is defined for each dissimilarity level v by inserting an edge (i, j) between nodes i and j if objects i and j are less dissimilar than v. That is,

$$(i, j) \in G(v) \qquad \text{if and only if} \quad d(i, j) \leq v$$

As discussed in Section 2.2, we assume that $d(i, i) = 0$ for all i. Thus $G(v)$ defines a binary relation for any real number v that is reflexive and symmetric. A binary relation is a subset of the product set $\mathcal{X} \times \mathcal{X}$, where $\mathcal{X}$ is the set of objects. Objects x_i and x_j are "related" if their dissimilarity is below the threshold v. Reflexive, symmetric binary relations are pictured in a natural fashion by a threshold graph. Figure 3.3 shows the binary relation obtained from proximity matrix $\mathcal{D}_1$ above for a threshold of 5. The symbol "$*$" in position (i, j) of the matrix means that the pair (x_i, x_j) belongs to the binary relation.

$$\begin{array}{c|ccccc} & x_1 & x_2 & x_3 & x_4 & x_5 \\ \hline x_1 & * & & & * & \\ x_2 & & * & * & * & * \\ x_3 & & * & * & & \\ x_4 & * & * & & * & * \\ x_5 & & * & & * & * \end{array}$$

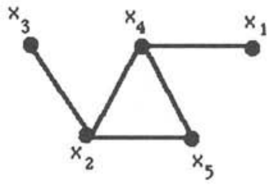


Figure 3.3 Binary relation and threshold graph for threshold 5.

Simple algorithms for the single-link and complete-link clustering methods based on threshold graphs are listed below. These algorithms should help one conceptualize the way in which the two hierarchies are formed and can easily be applied to small problems. Other algorithms are given later in this chapter that are appropriate for computer implementation. Both algorithms assume an ordinal dissimilarity matrix containing no tied entries and produce a nested sequence of clusterings that can be pictured on a dendrogram.

AGGLOMERATIVE ALGORITHM FOR SINGLE-LINK CLUSTERING

Step 1. Begin with the disjoint clustering implied by threshold graph $G(0)$, which contains no edges and which places every object in a unique cluster, as the current clustering. Set $k \leftarrow 1$.

Step 2. Form threshold graph $G(k)$.

If the number of components (maximally connected subgraphs) in $G(k)$ is less than the number of clusters in the current clustering, redefine the current clustering by naming each component of $G(k)$ as a cluster.

Step 3. If $G(k)$ consists of a single connected graph, stop. Else, set $k \leftarrow k + 1$ and go to step 2.

AGGLOMERATIVE ALGORITHM FOR COMPLETE-LINK CLUSTERING

Step 1. Begin with the disjoint clustering implied by threshold graph $G(0)$, which contains no edges and which places every object in a unique cluster, as the current clustering. Set $k \leftarrow 1$.

Step 2. Form threshold graph $G(k)$.

If two of the current clusters form a clique (maximally complete subgraph) in $G(k)$, redefine the current clustering by merging these two clusters into a single cluster.

Step 3. If $k = n(n - 1)/2$, so that $G(k)$ is the complete graph on the n nodes, stop. Else, set $k \leftarrow k + 1$ and go to step 2.

These algorithms can be extended to dissimilarity matrices on interval and ratio scales as long as no entries are tied. Simply view $G(k)$ as the threshold graph containing edges corresponding to the k smallest dissimilarities. A *threshold dendrogram* records the clusterings in the order in which they are formed, irrespective of the dissimilarity level at which the clusterings first appear. A *proximity dendrogram* lists the dissimilarity level at which each clustering forms and, in effect, is a nonlinear transformation of the scale used with a threshold dendrogram. Examples of proximity dendrograms are given in Section 3.2.2.

The single-link clustering on $G(v)$ is defined in terms of connected subgraphs in $G(v)$; the complete-link clustering uses complete subgraphs. However, not all maximally complete subgraphs in a threshold graph need be complete-link clusters. The order in which the clusters are formed is crucial. Figure 3.4 exhibits the single-link and complete-link hierarchical clusterings for the proximity matrix $\mathcal{D}_1$ of Example 3.1. The first seven threshold graphs in the sequence of 10 threshold graphs are shown with nodes labeled so that node j denotes object x_j.

Please note the following peculiarities about forming hierarchical clusterings from threshold graphs. The entire single-link hierarchy is defined by the first

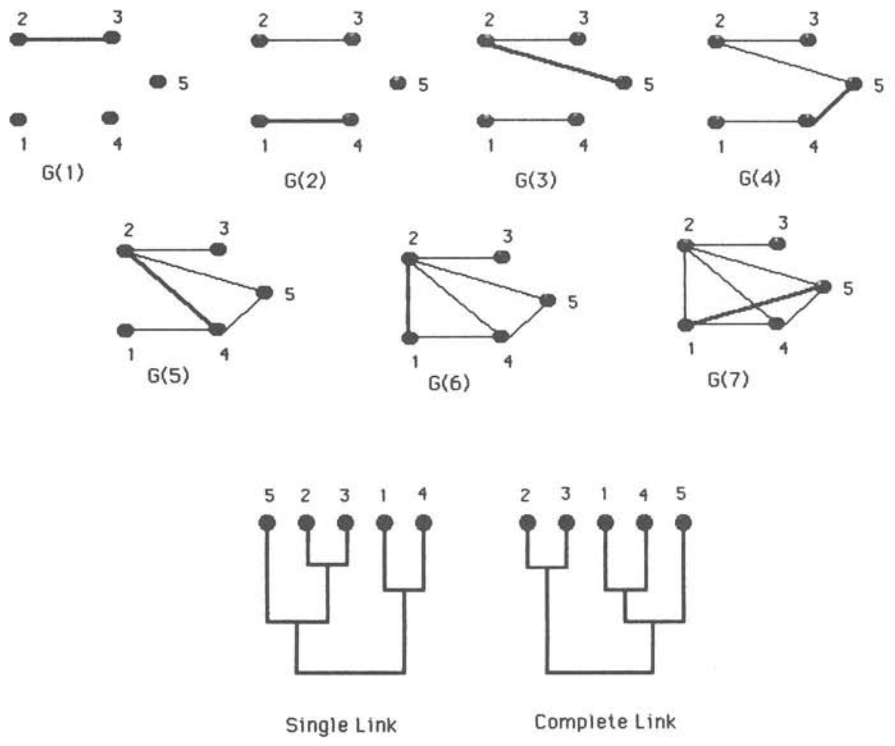


Figure 3.4 Threshold graphs and dendrograms for single-link and complete-link hierarchical clusterings.

four threshold graphs in Figure 3.4. However, the first seven threshold graphs are needed to determine the complete-link hierarchy. Once the two-cluster complete-link clustering has been obtained, no more explicit threshold graphs need be drawn because the two clusters will merge into the conjoint clustering only when all $n(n - 1)/2$ edges have been inserted. This example demonstrates the significance of nesting in the hierarchy. Objects $\{x_2, x_5, x_4\}$ form a clique, or maximally complete subgraph, in threshold graph $G(5)$, but the three objects are not a complete-link cluster. Once complete-link clusters $\{x_2, x_3\}$ and $\{x_1, x_4\}$ have been established, object x_5 must merge with one of the two established clusters; once formed, clusters cannot be dissolved and clusters cannot overlap. The dendrograms themselves are drawn with each clustering shown on a separate level, even though, for example, the two-cluster single-link clustering is obtained from $G(3)$ and the two-cluster complete-link clustering is obtained from $G(7)$.

The interpretation of the dendrograms is not under consideration in this chapter, but the two dendrograms in Figure 3.4 do raise a question about object x_5. Does it belong to the cluster $\{x_2, x_3\}$ or to the cluster $\{x_1, x_4\}$? A case can also be made for calling $\{x_2, x_4, x_5\}$ a cluster. Perhaps a hierarchical structure is not appropriate for this proximity matrix. These issues are examined in Chapter 4.

Hubert (1974a) provides the following algorithms for generating hierarchical clusterings by the single-link and complete-link methods. When the proximity matrix contains no ties, clusterings are numbered 0, 1, . . . , $(n - 1)$ and the mth clustering, $\mathscr{C}_m$, contains $n - m$ clusters.

$$\mathscr{C}_m = \{C_{m1}, C_{m2}, \ldots, C_{m(n-m)}\}$$

HUBERT'S ALGORITHM FOR SINGLE-LINK AND COMPLETE-LINK METHODS

Step 1. Set $m \leftarrow 0$. Form the disjoint clustering with clustering number m.

$$\mathscr{C}_0 = \{(x_1), (x_2), \ldots, (x_n)\}$$

Step 2a. To find the next clustering (with clustering number $m + 1$) by the *single-link* method, define the function Q_s for all pairs (r, t) of clusters in the current clustering as follows.

$$Q_s(r, t) = \min \{d(i, j) : \text{the maximal subgraph of } G(d(i, j)) \text{ defined by } C_{mr} \cup C_{mt} \text{ is connected}\}$$

Clusters C_{mp} and C_{mq} are merged to form the next clustering in the single-link hierarchy if

$$Q_s(p, q) = \min \{Q_s(r, t)\}$$

Step 2b. The function Q_c is used to find clustering number $m + 1$ by the *complete-link* method and is defined for all pairs (r, t) of clusters in the current clustering.

$$Q_c(r, t) = \min \{d(i, j) : \text{the maximal subgraph of } G(d(i, j)) \text{ defined by } C_{mr} \cup C_{mt} \text{ is complete}\}$$

Cluster C_{mp} is merged with cluster C_{mq} under the complete-link method if

$$Q_c(p, q) = \min \{Q_c(r, t)\}$$

Step 3. Set $m \leftarrow m + 1$ and repeat step 2. Continue until all objects are in a single cluster.

The word "maximal" in the definitions of functions Q_c and Q_s means that all nodes of the two clusters C_{mr} and C_{mt} must be considered when establishing connectedness or completeness. Only existing clusters can be merged at the next level.

Example 3.2

One way to understand the functions Q_s and Q_c is to consider the sequence of threshold graphs even though the threshold graphs are not necessary to the evaluation of these functions. For example, the first seven threshold graphs for the proximity matrix $\mathcal{D}_1$ (Example 3.1) are given in Figure 3.4. The third clustering ($m = 2$) can be numbered as follows.

$$\mathcal{C}_2 = \{C_{21}, C_{22}, C_{23}\}$$

The three clusters are defined as

$$C_{21} = \{x_5\},\ C_{22} = \{x_2, x_3\},\ C_{23} = \{x_1, x_4\}$$

To evaluate Q_s when m is 2, find the smallest proximity that will connect two of the existing clusters. Clusters C_{21} and C_{22} become connected in threshold graph $G(3)$. Therefore, (p, q) is (1, 2) and $Q_s(p, q)$ is 3. Another way of understanding this function is to realize that $Q_s(r, t)$ is the smallest dissimilarity that connects clusters C_{mr} and C_{mt} and the smallest of the dissimilarities so found defines the next clustering. In this case, clusters C_{21} and C_{22} first connect at level 3, or in $G(3)$, clusters C_{21} and C_{23} first connect in $G(4)$ and clusters C_{22} and C_{23} first form a connected subgraph in $G(5)$. The minimum of the levels (3, 4, 5) is 3.

The interpretation of Q_c is much the same, with completeness replacing connectedness. For example, Q_c is found when $m = 2$ by searching the threshold graphs in sequence until one is found that merges existing clusters from $\mathcal{C}_2$ into a complete subgraph. This does not happen until $G(7)$, so (p, q) is (1, 3) and $Q_c(p, q)$ is 7. Clusters C_{21} and C_{22} first form a complete subgraph in threshold graph $G(9)$. Clusters C_{21} and C_{23} first merge into a complete subgraph in $G(7)$ and clusters C_{22} and C_{23} first form a complete subgraph in $G(10)$. The minimum of the levels (7, 9, 10) is 7, so the fourth complete link clustering ($m = 3$) is achieved at threshold 7 and merges clusters C_{21} and C_{23}. The fact that other complete subgraphs are formed in the process, such as $\{x_2, x_4, x_5\}$, is immaterial.

Single-link clusters are characterized as maximally connected subgraphs, whereas complete-link clusters are cliques, or maximally complete subgraphs. Jardine and Sibson (1971) have demonstrated several desirable theoretical properties of single-link clusterings, but several authors (e.g., Wishart, 1969; Hubert, 1974b) have objected to certain practical difficulties with clusters formed by the single-link method. For example, single-link clusters easily chain together and are often "straggly." Only a single edge between two large clusters is needed to merge the clusters. On the other hand, complete-link clusters are conservative. All pairs of objects must be related before the objects can form a complete-link cluster. Completeness is a much stronger property than connectedness. Perceived deficiencies in these two clustering methods have led to a large number of alternatives, some of which are explained in Sections 3.2.7 and 3.2.9. For example, Hansen and DeLattre (1978) noted that single-link clusters may chain and have little homogeneity, while complete-link clusters may not be well separated.

Every connected subgraph of a threshold graph is a single-link cluster but not every clique is a complete-link cluster. Peay (1975) proposed an exclusive, overlapping, hierarchical clustering method based on cliques and extended it to asymmetric proximity matrices. Matula (1977) noted that the number of possible cliques is huge, so clustering based on cliques is practical only for small n.

Suppose that the latest clustering of $\{x_1, x_2, \ldots, x_n\}$ in one of the hierarchies has been formed by merging clusters C_{mp} and C_{mq} in the clustering

$$\{C_{m1}, C_{m2}, \ldots, C_{m(n-m)}\}$$

The following characterizations may help to distinguish the two clustering methods. If the clustering was by the single-link method, we would know that

$$\min_{x_i \in C_{mp}, x_j \in C_{mq}} \{d(i,j)\} = \min_{r \neq s} \{ \min_{x_i \in C_{mr}, x_j \in C_{ms}} \{d(i,j)\}\}$$

If the clustering was by the complete-link method, we have that

$$\max_{x_i \in C_{mp}, x_j \in C_{mq}} \{d(i,j)\} = \min_{r \neq s} \{ \max_{x_i \in C_{mr}, x_j \in C_{ms}} \{d(i,j)\}\}$$

These characterizations show why the single-link method has been called the "minimum" method and the complete-link method has been named the "maximum" method (Johnson, 1967). However, if the proximities are similarities instead of dissimilarities, this terminology would be confusing. This characterization also explains why the complete-link method is referred to as the "diameter" method. The diameter of a complete subgraph is the largest proximity among all proximities for pairs of objects in the subgraph. Although the complete-link method does not generate clusters with minimum diameter, the diameter of a complete-link cluster is known to equal the level at which the cluster is formed. By contrast, single-link clusters are based on connectedness and are characterized by minimum path length among all pairs of objects in the cluster.

3.2.2 Dendrograms and Recovered Structure

An important objective of hierarchical cluster analysis is to provide a picture of the data that can easily be interpreted, such as the dendrograms in Figure 3.4. Dendrograms list the clusterings one after another. Cutting a dendrogram at any level defines a clustering and identifies clusters. The level itself has no meaning in terms of the scale of the proximity matrix.

A *proximity graph* is a threshold graph in which each edge is weighted according to its proximity. The proximities used in Section 3.2.1 are ordinal, so the weights are integers from 1 to $n(n-1)/2$. The dendrogram drawn from a proximity graph is called a *proximity dendrogram* and records both the clusterings and the proximities at which they are formed. Proximity dendrograms are especially useful when the proximities are on an interval or ratio scale.

Example 3.3

A ratio proximity matrix is given below as $\mathcal{D}_2$. The threshold and proximity dendrograms are given in Figure 3.5. Also shown is the sequence of proximity graphs which provides the actual dissimilarity values at which clusters are formed.

$$\mathcal{D}_2 = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 5.8 & 4.2 & 6.9 & 2.6 \\ & 6.7 & 1.7 & 7.2 \\ & & 1.9 & 5.6 \\ & & & 7.6 \end{bmatrix} \end{array}$$

A proximity dendrogram is drawn on a proximity scale from a sequence of proximity graphs and highlights clusters that are "born" early and "last" a long time in the dendrogram. These observations are the basis for formal measures of cluster validity in Chapter 4.

Any hierarchical clustering algorithm can be seen as a way of transforming a proximity matrix into a dendrogram. Only the single-link and complete-link methods of clustering have been discussed so far, but the statement applies to hierarchical clustering methods defined in Sections 3.2.7 and 3.2.9 as well. Threshold and proximity dendrograms represent the structure that the hierarchical clustering method is imposing on the data. This imposed structure can be captured in another proximity matrix called the cophenetic matrix. The agreement between the given proximity matrix and the cophenetic matrix measures the degree to which the hierarchical clustering method captures the actual structure of the data. Formal methods for measuring this agreement are discussed in Chapter 4. Here the cophenetic matrix is defined to help explain the difference between the single-link and complete-link methods.

We begin with a hierarchical clustering:

$$\{\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_{n-1}\}$$

where the mth clustering contains $n - m$ clusters:

$$\mathcal{C}_m = \{C_{m1}, C_{m2}, \ldots, C_{m(n-m)}\}$$

A level function, L, records the proximity at which each clustering is formed. For a threshold dendrogram $L(k) = k$, because the levels in the dendrogram are evenly spaced. In general,

$$L(m) = \min \{d(x_i, x_j) : \mathscr{C}_m \text{ is defined}\}$$

The *cophenetic proximity* measure d_C on the n objects is the level at which objects x_i and x_j are first in the same cluster.

$$d_C(i, j) = L(k_{ij})$$

where

$$k_{ij} = \min \{m : (x_i, x_j) \in C_{mq}, \text{ some } q\}$$

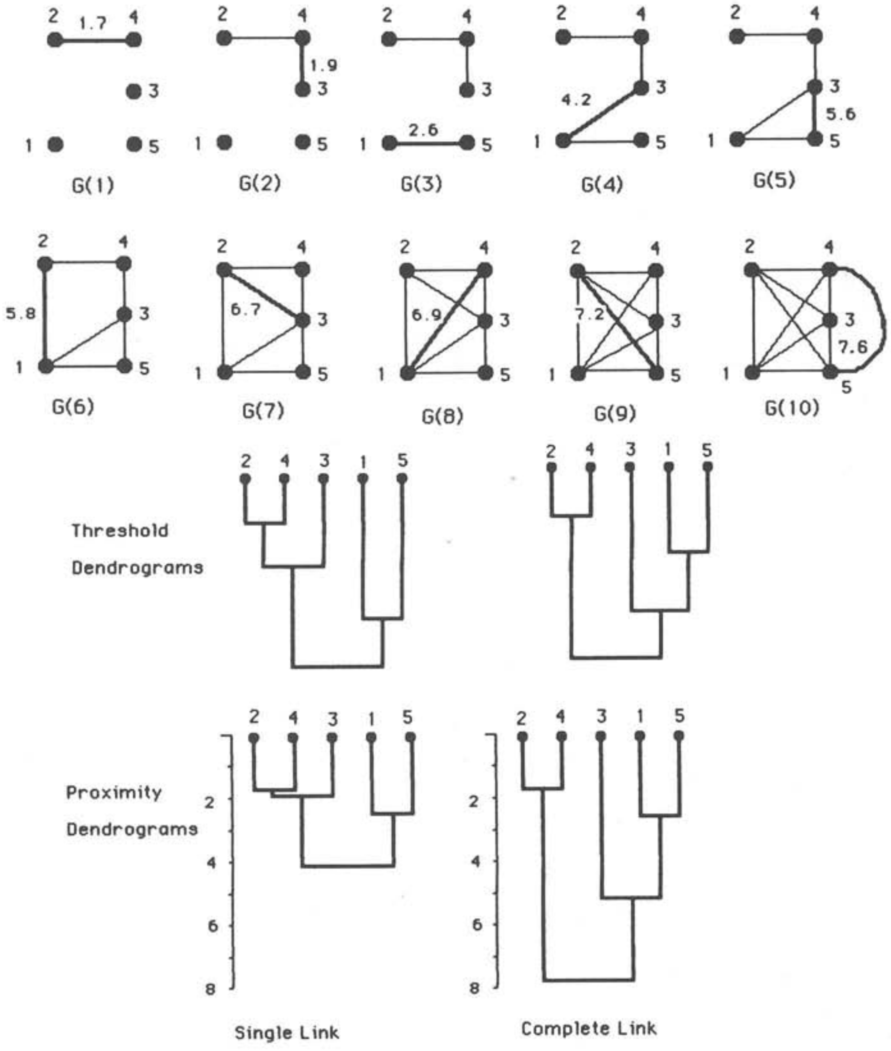


Figure 3.5 Examples of threshold and proximity graphs with corresponding dendrograms.

The matrix of values $[d_C(x_i, x_j)]$ is called the *cophenetic matrix*. The closer the cophenetic matrix and the given proximity matrix, the better the hierarchy fits the data. There can be no more than $(n - 1)$ levels in a dendrogram, so there can be no more than $(n - 1)$ distinct cophenetic proximities. Since the cophenetic matrix has $n(n - 1)/2$ entries, it must contain many ties.

The cophenetic matrix for the single-link dendrogram in Figure 3.5 is shown below as $\mathcal{D}_{Cs}$ and will be used to demonstrate some interesting properties of cophenetic matrices.

$$\mathcal{D}_{Cs} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 4.2 & 4.2 & 4.2 & 2.6 \\ & 1.9 & 1.7 & 4.2 \\ & & 1.9 & 4.2 \\ & & & 4.2 \end{bmatrix} \end{array}$$

Applying the single-link clustering method to $\mathcal{D}_{Cs}$ reproduces the single-link dendrogram in Figure 3.5. This might be expected. However, applying the complete-link method to $\mathcal{D}_{Cs}$ generates the same (single-link) dendrogram. The complete-link method is usually ambiguous when the proximity matrix contains ties, as discussed in Section 3.2.6. However, the cophenetic matrix is so arranged that tied proximities form complete subgraphs and no ambiguity occurs under complete-link clustering. A cophenetic matrix is an example of a proximity matrix with perfect hierarchical structure. Both the single-link and the complete-link methods generate exactly the same dendrogram when applied to a cophenetic matrix. Repeating this exercise by starting with the complete-link dendrogram generates the cophenetic matrix $\mathcal{D}_{Cc}$.

$$\mathcal{D}_{Cc} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 7.6 & 5.6 & 7.6 & 2.6 \\ & 7.6 & 1.7 & 7.6 \\ & & 7.6 & 5.6 \\ & & & 7.6 \end{bmatrix} \end{array}$$

The cophenetic matrix $\mathcal{D}_{Cc}$ also has perfect hierarchical structure. The complete-link and single-link clustering methods will produce exactly the same dendrogram when applied to $\mathcal{D}_{Cc}$, and that dendrogram will be identical to the complete-link dendrogram in Figure 3.5. An important question in applications is: Which dendrogram better describes the true structure of the data?

3.2.3 Hierarchical Structure and Ultrametricity

The fact that both the single-link and the complete-link methods generate exactly the same proximity dendrogram when applied to a cophenetic matrix suggests that the cophenetic matrix captures "true" or "perfect" hierarchical structure. Whether or not the hierarchical structure is appropriate for a given data set has yet to be determined, but the type of structure exemplified by the cophenetic matrix is very special. The justification for calling the cophenetic matrix "true" hierarchical structure comes from the fact that a cophenetic proximity measure d_C defines the following equivalence relation, denoted R_C, on the set of objects:

$$R_C(a) = \{(x_i, x_j) : d_C(i, j) \leq a\}$$

Relation $R_C(a)$ can be shown to be an equivalence relation for any $a \geq 0$ by checking the three conditions necessary for an equivalence relation. Since $d_C(i, i) = 0$ for all i,

$$(x_i, x_i) \in R_C(a) \qquad \text{for all } a \geq 0$$

so $R_C(a)$ is reflexive. Since $d_C(i, j) = d_C(j, i)$ for all (i, j),

$$(x_j, x_i) \in R_C(a) \quad \text{if} \quad (x_i, x_j) \in R_C(a) \qquad \text{for all } a \geq 0$$

so $R_C(a)$ is symmetric. The final condition, transitivity, requires that for all $a \geq 0$,

$$\text{if } (x_i, x_k) \in R_C(a) \quad \text{and if} \quad (x_k, x_j) \in R_C(a), \qquad \text{then } (x_i, x_j) \in R_C(a)$$

This condition must be satisfied for all triples (x_i, x_j, x_k) of objects and all a. It can also be restated as

$$d_C(i, j) \leq \max \{d_C(i, k), d_C(k, j)\} \qquad \text{for all } (i, j, k)$$

When stated in this way, the requirement is called the *ultrametric inequality*. A close inspection of the cophenetic matrices for Figure 3.5 shows that they satisfy the ultrametric inequality, so $R_C(a)$ is, indeed, an equivalence relation for any $a \geq 0$. The nesting of the clusterings forming the hierarchy assures transitivity. The only way that the very restrictive ultrametric inequality can be satisfied is to have many ties in the cophenetic proximity. Recall that, at most, only $n - 1$ of the $n(n - 1)/2$ cophenetic proximities can be distinct. Since cophenetic proximity measures represent perfect hierarchical structure, proximity measures without ties seldom reflect true hierarchical structure. The concept of ultrametricity has been developed separately in mathematics and has applications in physics. See Rammal et al. (1986) for an excellent review of ultrametricity and Schikhof (1984) for a mathematical treatment of ultrametricity in the realm of "p-adic" analysis.

Two items should be noted with regard to the ultrametric inequality. First, the cophenetic proximities derived from single-link and complete-link clusterings always satisfy the ultrametric inequality. However, Section 3.2.7 will introduce some hierarchical clustering methods whose cophenetic proximities are not ultrametric. Second, a geometric interpretation of the ultrametric inequality demonstrates why proximities measured in applications are very seldom ultrametric. Suppose that each object is a pattern in a d-dimensional space. If Euclidean distance is the measure of proximity and if the proximity matrix is to be ultrametric, the triangles formed by *all* triples of points must be isosceles triangles with the unequal leg no longer than the two legs of equal length.

Jardine and Sibson (1971) characterize hierarchical clustering methods as mappings from the class of proximity matrices to the class of ultrametric proximity measures. That is, a hierarchical clustering method imposes a dendrogram on the given proximity matrix and this establishes the cophenetic proximity matrix, which satisfies the ultrametric inequality. Measures of fit between proximity measures and cophenetic proximity measures are discussed in Chapter 4. The property of ultrametricity is also called *monotonicity*; a cophenetic proximity measure satisfies the ultrametric inequality only if the clusters form in a monotonic manner as dissimilarity increases. In other words, the clusterings are nested in the hierarchy. Single-link and complete-link clusterings are always monotonic, but other common clustering methods defined in Section 3.2.7 can create the next clustering at a *smaller* dissimilarity than the present one. This issue is discussed in Section 3.2.8.

3.2.4 Other Graph Theory Algorithms for Single-Link and Complete-Link

The algorithms for single-link and complete-link hierarchical clusterings described thus far establish step-by-step procedures for forming dendrograms. In this section we present other algorithms for these clustering methods that provide insight into the clustering methods and can be computationally attractive.

An algorithm for single-link clustering begins with the minimum spanning tree (MST) for $G(\infty)$, which is the proximity graph containing all $n(n - 1)/2$ edges. Although the single-link hierarchy can be derived from the MST, the MST cannot be found from a single-link hierarchical clustering. For convenience, we assume that no two edges in the MST have the same weight, even though Section 3.2.6 shows that ties in proximity pose no problem with single-link clustering. An agglomerative algorithm for single-link clustering is given below that assumes a dissimilarity matrix.

GRAPH THEORY ALGORITHM FOR SINGLE-LINK CLUSTERING

Step 1. Begin with the disjoint clustering, which places each object in its own cluster. Find an MST on $G(\infty)$.

Repeat steps 2 and 3 until all objects are in one cluster.

Step 2. Merge the two clusters connected by the MST edge with the smallest weight to define the next clustering.

Step 3. Replace the weight of the edge selected in step 2 by a weight larger than the largest proximity.

This algorithm follows from the characterization for single-link clustering given in Section 3.2.1 and the definition of MST. A divisive algorithm is just as simple. Cut the edges in the MST in the order of weight, cutting the largest first. Each cut defines a new clustering, with those objects connected in the MST at any stage belonging to the same cluster. As long as no proximity ties occur, these algorithms generate the same single-link clusterings as the algorithms presented earlier. Gower and Ross (1969) first proposed this algorithm. Rohlf (1973) provided an implementation that examines each proximity value only once.

Example 3.4

Examples of the two algorithms are given in Figure 3.6 for the proximity matrix $\mathcal{D}_3$ defined below.

$$\mathcal{D}_3 = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 2.3 & 3.4 & 1.2 & 3.7 \\ & 2.6 & 1.8 & 4.6 \\ & & 4.2 & 0.7 \\ & & & 4.4 \end{bmatrix} \end{array}$$

A node coloring of a threshold graph $G(v)$ is an assignment of "colors," or labels, to the n nodes in such a way that no two nodes connected by an edge

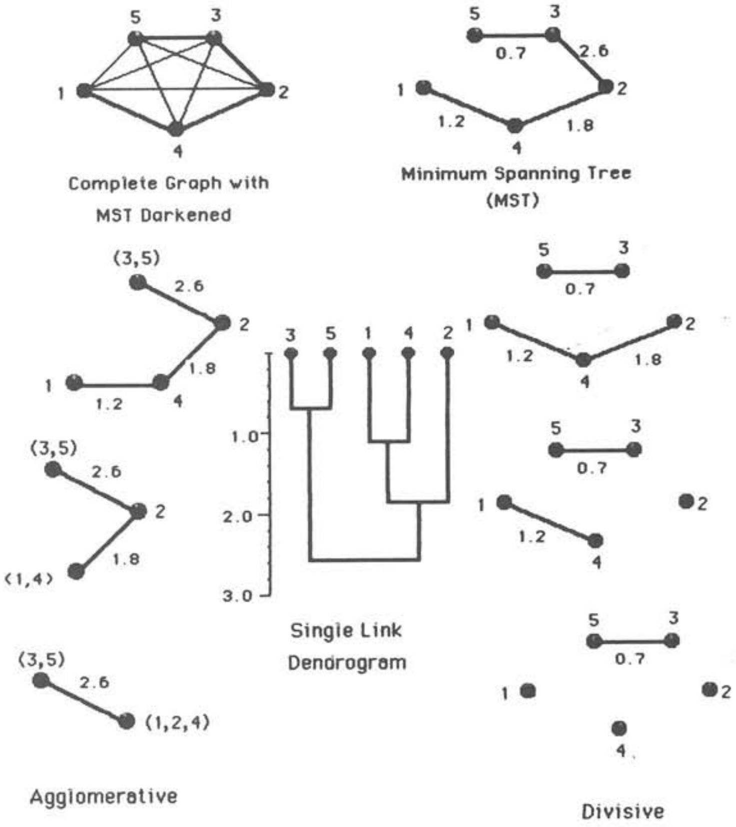


Figure 3.6 Examples of agglomerative and divisive single-link algorithms based on the MST.

in $G(v)$ are colored the same. Baker and Hubert (1976) show how the set of node colorings is related to hierarchical clustering. The connection between node coloring and complete-link clustering is not as simple as is the relation between single-link clustering and the MST. The last complete-link clustering achieved for a given threshold graph $G(v)$ corresponds to a coloring of the nodes of the complement of $G(v)$. Hansen and DeLattre (1978) provide other algorithms from graph coloring.

3.2.5 Matrix Updating Algorithms for Single-Link and Complete-Link

In this section we discuss algorithms for single-link and complete-link clustering in terms of a scheme for updating the proximity matrix. This approach was suggested by King (1967) and popularized by Johnson (1967), who formalized the procedure. The algorithm is an agglomerative scheme that erases rows and columns in the proximity matrix as old clusters are merged into new ones. We again simplify the algorithm by assuming no ties in the proximity matrix. Figure 3.7 provides examples of this algorithm for the proximity matrix $\mathcal{D}_3$ (Example 3.4).

The $n \times n$ proximity matrix is $\mathcal{D} = [d(i, j)]$. The clusterings are assigned sequence numbers $0, 1, \ldots, (n - 1)$ and $L(k)$ is the level of the kth clustering. A cluster with sequence number m is denoted (m) and the proximity between clusters (r) and (s) is denoted $d[(r), (s)]$.

JOHNSON'S ALGORITHM FOR SINGLE-LINK AND COMPLETE-LINK CLUSTERING

Step 1. Begin with the disjoint clustering having level $L(0) = 0$ and sequence number $m = 0$.

Step 2. Find the least dissimilar pair of clusters in the current clustering, say pair $\{(r), (s)\}$, according to

$$d[(r), (s)] = \min \{d[(i), (j)]\}$$

where the minimum is over all pairs of clusters in the current clustering.

Step 3. Increment the sequence number: $m \leftarrow m + 1$. Merge clusters (r) and (s) into a single cluster to form the next clustering m. Set the level of this clustering to

$$L(m) = d[(r), (s)]$$

Step 4. Update the proximity matrix, $\mathcal{D}$, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The proximity between the new cluster, denoted (r, s) and old cluster (k) is defined as follows. For the single-link method,

$$d[(k), (r, s)] = \min \{d[(k), (r)], d[(k), (s)]\}$$

For the complete-link method,

$$d[(k), (r, s)] = \max \{d[(k), (r)], d[(k), (s)]\}$$

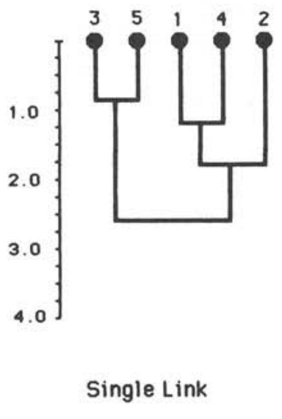
Step 5. If all objects are in one cluster, stop. Else, go to step 2.

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 2.3 & 3.4 & 1.2 & 3.7 \\ 2 & & 0 & 2.6 & 1.8 & 4.6 \\ 3 & & & 0 & 4.2 & \boxed{0.7} \\ 4 & & & & 0 & 4.4 \\ 5 & & & & & 0 \end{array}$$

$$\begin{array}{c|cccc} & 1 & 2 & 3,5 & 4 \\ \hline 1 & 0 & 2.3 & 3.4 & \boxed{1.2} \\ 2 & & 0 & 2.6 & 1.8 \\ 3,5 & & & 0 & 4.2 \\ 4 & & & & 0 \end{array} \qquad \begin{array}{c|cccc} & 1 & 2 & 3,5 & 4 \\ \hline 1 & 0 & 2.3 & 3.7 & \boxed{1.2} \\ 2 & & 0 & 4.6 & 1.8 \\ 3,5 & & & 0 & 4.4 \\ 4 & & & & 0 \end{array}$$

$$\begin{array}{c|ccc} & 1,4 & 2 & 3,5 \\ \hline 1,4 & 0 & \boxed{1.8} & 3.4 \\ 2 & & 0 & 2.6 \\ 3,5 & & & 0 \end{array} \qquad \begin{array}{c|ccc} & 1,4 & 2 & 3,5 \\ \hline 1,4 & 0 & \boxed{2.3} & 4.4 \\ 2 & & 0 & 4.6 \\ 3,5 & & & 0 \end{array}$$

$$\begin{array}{c|cc} & 1,2,4 & 3,5 \\ \hline 1,2,4 & 0 & \boxed{2.6} \\ 3,5 & & 0 \end{array} \qquad \begin{array}{c|cc} & 1,2,4 & 3,5 \\ \hline 1,2,4 & 0 & \boxed{4.6} \\ 3,5 & & 0 \end{array}$$



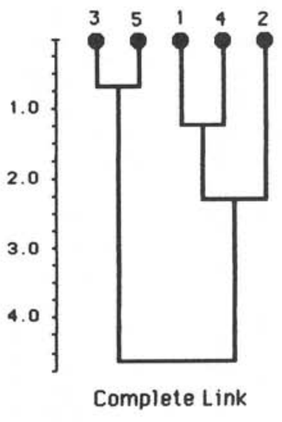


Figure 3.7 Examples of matrix updating algorithms for single-link and complete-link clusterings.

Anderberg (1973) discusses three computational approaches to implementing the algorithm above, called the stored matrix, sorted matrix, and stored data approaches. The three approaches differ as to whether the pattern matrix or the dissimilarity matrix is stored in random access memory or in auxiliary storage, such as disk. Note that the dissimilarity matrix requires more storage than the pattern matrix when $n \gg d$. The stored matrix approach, where the entire dissimilarity matrix is stored in random access memory, is fastest.

Example 3.5

The computational examples in Figure 3.7 demonstrate the construction of single-link and complete-link hierarchies. This example demonstrates the qualitative differences between the single-link and complete-link hierarchies for the two artificial data sets defined in Section 2.4. The first data set, called DATA1, consists of 100 patterns in a four-dimensional pattern space generated so as to have four categories, or true clusters. Patterns 1 through 24 were generated from category 1, patterns 25 through 59 from category 2, patterns 60 through 80 from category 3, and patterns 81 through 100 were generated in category 4. An eigenvector projection is given in Figure 2.9. The proximity measure is squared Euclidean distance in the pattern space. Since the proximity measure is Euclidean distance and since the data were generated to several decimal places on a computer, we feel safe in assuming that no proximity ties exist, so the hierarchies are both unique (see Section 3.2.6). Figures 3.8 and 3.9 show the proximity dendrograms for the single-link and complete-link hierarchies, respectively.

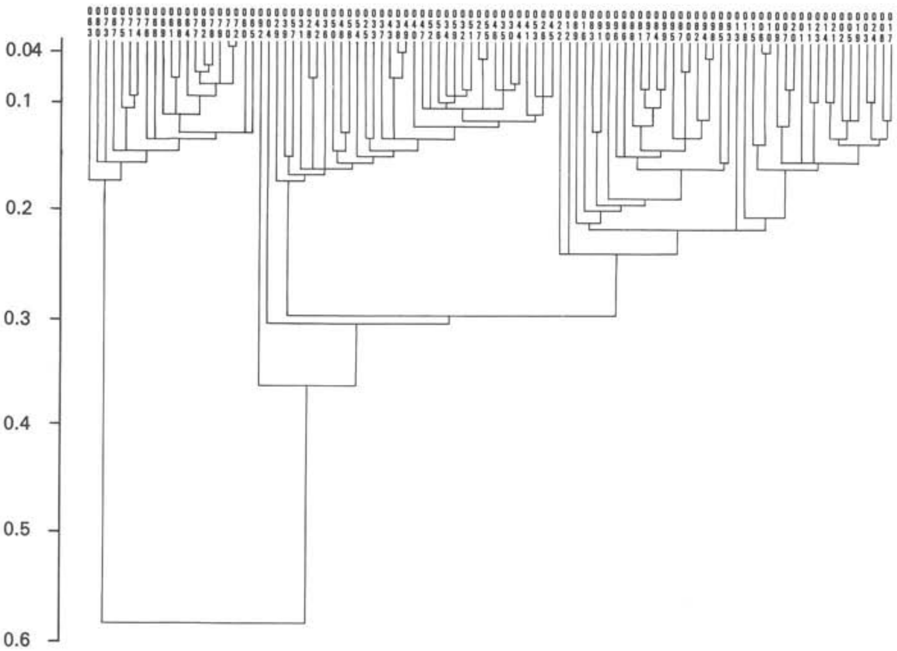


Figure 3.8 Single-link hierarchy for 100 clustered patterns in four dimensions.

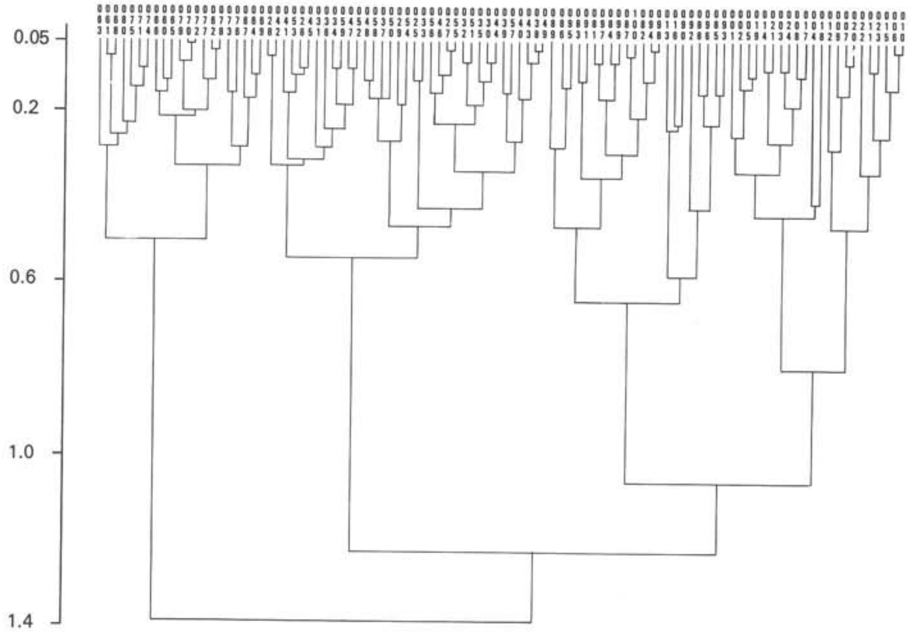


Figure 3.9 Complete-link hierarchy for 100 clustered patterns in four dimensions.

This example demonstrates the difficulty in comparing two dendrograms and motivates the development of methods for automatically isolating significant clusters that are presented in Chapter 4. The complete-link dendrogram in Figure 3.9 can be cut at level 1.0 to generate four clusters. These clusters recover the original four categories in the data perfectly. The four-category structure is not at all apparent in the single-link hierarchical clustering of Figure 3.8.

Clustering methods have the nasty habit of creating clusters in data even when no natural clusters exist, so hierarchies and clusterings must be viewed with extreme suspicion. Figures 3.10 and 3.11 demonstrate this statement on the two hierarchies for a data set, called DATA2, consisting of 100 points uniformly distributed over a unit hypercube in six dimensions (see Section 2.4). The patterns are positioned at random, so it is barely possible that they have arranged themselves into meaningful clusters; however, it is unlikely that real clusters exist, especially considering the two-dimensional projections in Figure 2.11. We thus interpret Figures 3.10 and 3.11 as hierarchies in which no true clusters exist. The single-link dendrogram in Figure 3.10 exhibits the chaining that is characteristic of single-link hierarchies. This chaining can occur even when valid clusters exist, as in Figure 3.8. The complete-link hierarchy in Figure 3.11 suggests some meaningful clusters; it looks more clustered than the single-link hierarchy, and this is the lure of complete-link clustering. It tends to produce dendrograms that form small clusters which combine nicely into larger clusters even when such a hierarchy is not warranted, as with random data. This example should demonstrate the difficulties inherent in letting the human eye scan over the dendrogram to pick out believable clusters and clusterings.

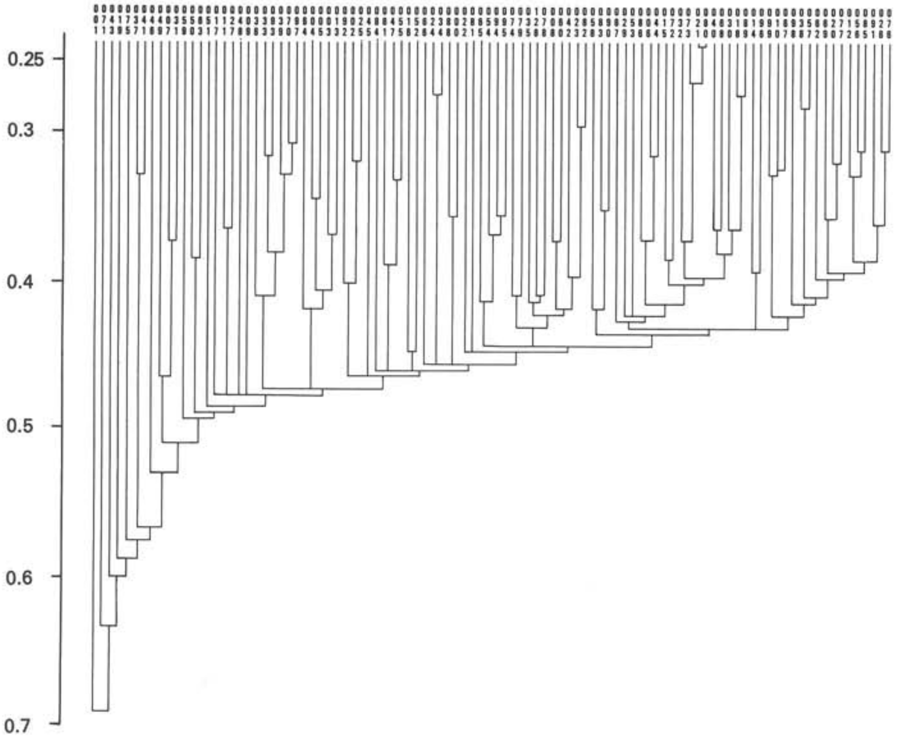


Figure 3.10 Single-link hierarchy for 100 random patterns in six dimensions.

3.2.6 Ties in Proximity

The computational complexity of competing algorithms for implementing a particular clustering method and the availability of software should determine which algorithm is appropriate for a given application. The problem of choosing between the single-link and complete-link methods is much more difficult than choosing an algorithm for one of the methods. No list of characteristics exists that lets us choose between the two methods in a calm, rational manner. Some theoretical and practical information about the two methods is summarized in this section, especially the effects of ties in the proximity matrix.

The single-link and complete-link methods differ in many respects, such as in the graph structures recognized and the updating procedure. The two methods produce the same clusterings when the proximity matrix satisfies the ultrametric inequality, as discussed in Section 3.2.3. This section demonstrates that the two methods differ in the way they treat ties in the proximity matrix. Up to now, we have assumed that the proximity matrix contains no ties so that two new clusters are never formed at the same level and the algorithms defined thus far produce unique dendrograms. A tie implies that two or more edges are added to the proximity

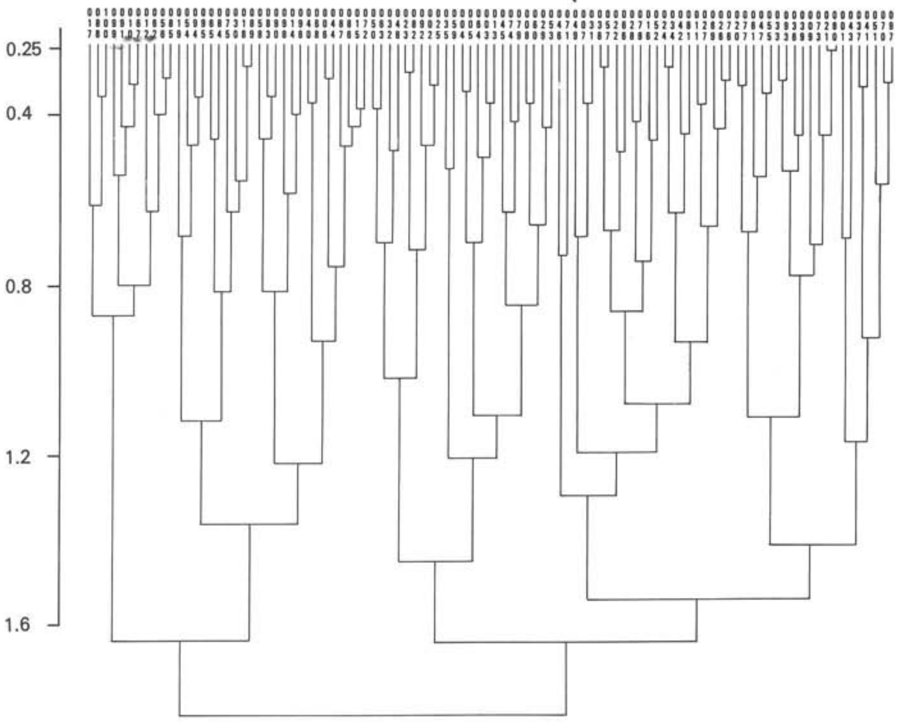


Figure 3.11 Complete-link hierarchy for 100 random patterns in six dimensions.

graph at once and that the minimum and maximum functions required in matrix updating are not unique.

Jardine and Sibson (1971) showed that the single-link method does not suffer from ambiguities due to ties because it has a continuity property. If the ties are broken in the proximities by adding or subtracting a small amount from the tied proximities, the resulting single-link dendrograms will merge smoothly into the same dendrogram as the added amount tends to zero, no matter how the ties are broken. This statement applies to all single-link algorithms as long as the rank orders of the proximities are not changed by the added amounts. By contrast, several complete-link dendrograms can be obtained by breaking ties in this way, as demonstrated in Figure 3.12.

Figure 3.12(a) shows the first three threshold graphs for the given proximity matrix. Two edges are added at once in $G(3)$. The single-link hierarchy is the same whether edge (2, 3) is inserted first or edge (3, 4) is inserted first. The proximity dendrogram for the single-link method in Figure 3.12(b) is unique even though more than one cluster can be formed at the same level. Algorithms based on the MST or on matrix updating produce the same results. The situation is very different with complete link clustering. Figure 3.12(b) shows the hierarchy defined by adding edge (2, 3) first and that formed when adding edge (3, 4)

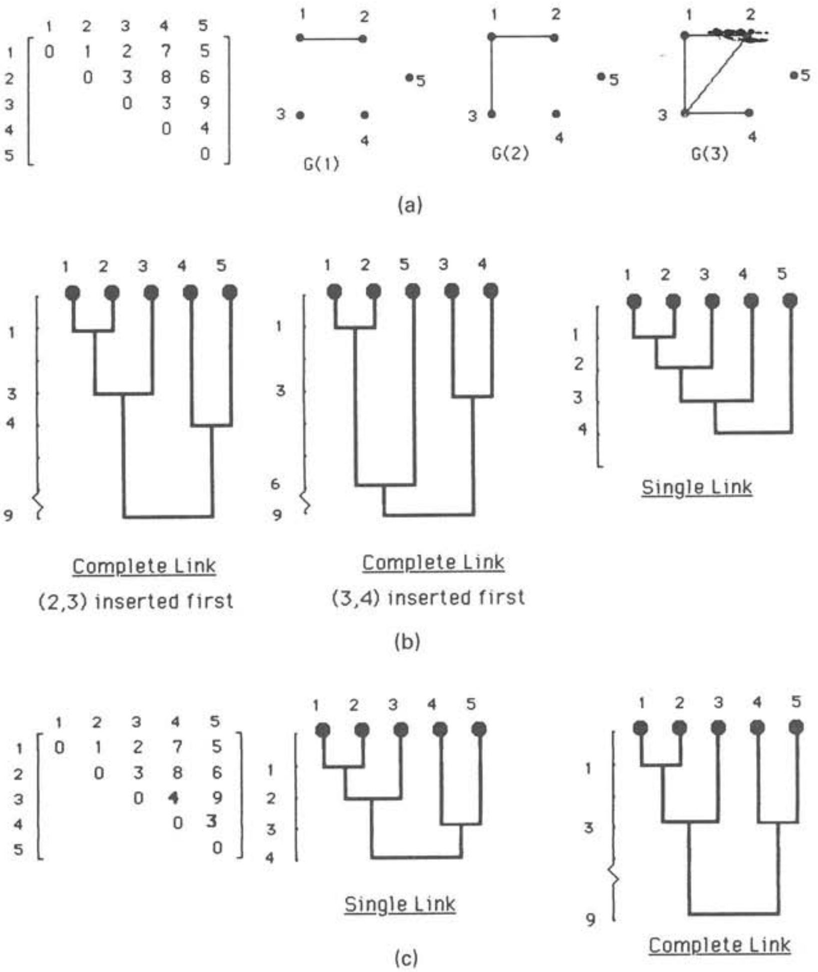


Figure 3.12 Effects of ties in proximity on single-link and complete-link clustering: (a) proximity matrix and threshold graphs; (b) proximity dendrograms; (c) altered proximity matrix and dendrograms.

first. The two clustering structures are very different. This effect can also be observed on the matrix updating algorithm and with Hubert's algorithm (Section 3.2.1). Adding the two edges (2, 3) and (3, 4) simultaneously does not solve the problem because the resulting four-edge threshold graph is not a complete graph. In fact, the next complete graph would be the one on all five nodes and the hierarchical clustering would have only three levels.

Figure 3.12(c) emphasizes the seriousness of ties. The given proximity matrix differs from that in Figure 3.12(a) in only two entries; the (3, 4) and (4, 5) entries are interchanged, as might occur through a typing error when entering data. In this case a unique complete-link hierarchy is obtained because the two edges with the same proximity can be added in arbitrary order. However, the hierarchy has two clusters forming at level 3. The single-link hierarchy is also shown. The single- and complete-link dendrograms in Figure 3.12(c) resemble one another much more closely than do those in Figure 3.12(b), which might lead one to believe that the proximity matrix in Figure 3.12(c) had a good hierarchical structure, whereas that in Figure 3.12(a) has a poor hierarchical structure. This example raises the issue of sensitivity. It appears that the hierarchical structure can change dramatically with small changes in the rank orders of the proximities.

The havoc that ties can create in a complete-link hierarchy has been noted by several researchers. Sibson (1971) and Williams et al. (1971) argue against the complete-link method as a feasible clustering procedure (see also Hubert, 1974a). The practical problem of ties is subtle. Software packages do not typically check for ties. The order in which an edge is added from a set of edges with the same proximity is at the whim of the programmer. The program will generate only one complete-link clustering, even though a number of clusterings might be equally justifiable. This problem is compounded when the proximity matrix contains several ties. The comparative studies in Section 3.5.2 suggest that the complete-link method produces more useful hierarchies in many applications than does the single-link method, even though proximity ties make it ambiguous.

3.2.7 General Matrix Updating Algorithms and Monotonicity

This section generalizes the algorithms in Section 3.2.5 and discusses issues in the computation and application of these algorithms. Questions of the validity of cluster structures are taken up in Chapter 4. The general paradigm for expressing SAHN (Sequential, Agglomerative, Hierarchical, Nonoverlapping) clustering methods is given in Section 3.2.5. Step 4 of that algorithm specifies how the dissimilarity matrix is to be updated by defining the formula for the dissimilarity between a newly formed cluster, (r, s), and an existing cluster, (k) with n_k objects. The single-link and complete-link algorithms use the minimum and maximum, respectively, of the dissimilarities between the pairs $\{(k), (r)\}$ and $\{(k), (s)\}$. Other clustering methods can be defined by specifying different combinations of the distances involved. A general formula for step 4 that includes most of the commonly referenced hierarchical clustering methods is given below.

$$\begin{aligned} d[(k), (r, s)] \\ &= \alpha_r d[(k), (r)] + \alpha_s d[(k), (s)] + \beta d[(r), (s)] + \gamma \lvert d[(k), (r)] - d[(k), (s)] \rvert \end{aligned}$$

This formula was first proposed by Lance and Williams (1967). Table 3.1 shows the parameter values for the most common algorithms. This table is also given in Milligan (1979) and in Day and Edelsbrunner (1984).

The acronym "PGM" refers to the "pair group method"; the prefixes "U" and "W" refer to unweighted and weighted, respectively. An "unweighted" method

TABLE 3.1 Coefficient Values for SAHN Matrix Updating Algorithms

Clustering Method	α_r	α_s	β	γ
Single-link	1/2	1/2	0	−1/2
Complete-link	1/2	1/2	0	1/2
UPGMA (group average)	$\frac{n_r}{n_r + n_s}$	$\frac{n_s}{n_r + n_s}$	0	0
WPGMA (weighted average)	1/2	1/2	0	0
UPGMC (unweighted centroid)	$\frac{n_r}{n_r + n_s}$	$\frac{n_s}{n_r + n_s}$	$\frac{-n_r n_s}{(n_r + n_s)^2}$	0
WPGMC (weighted centroid)	1/2	1/2	−1/4	0
Ward's method (minimum variance)	$\frac{n_r + n_k}{n_r + n_s + n_k}$	$\frac{n_s + n_k}{n_r + n_s + n_k}$	$\frac{-n_k}{n_r + n_s + n_k}$	0

treats each object in a cluster equally, regardles of the structure of the dendrogram. A ''weighted'' method weights all clusters the same, so objects in small clusters are weighted more heavily than objects in large clusters. The suffixes ''A'' and ''C'' refer to ''arithmetic averages'' and ''centroids.'' Thus ''UPGMA'' stands for ''unweighted pair group method using arithmetic averages'' and ''WPGMC'' refers to ''weighted pair group method using centroids.'' Rohlf (1970) and Sneath and Sokal (1973) have used this terminology. The UPGMC method has also been called, simply, the centroid method, while the WPGMC method has been called the median method (see Lance and Williams, 1967).

Sneath and Sokal (1973) provide a good discussion of the backgrounds of these methods and define other SAHN algorithms. Arithmetic averaging attempts to avoid the extremes of the single-link and complete-link methods. When measuring the dissimilarity between an existing cluster and a prospective cluster, the single-link method finds the closest pair of objects in the two clusters, the complete-link method finds the most distant pair, and the UPGMA and WPGMA methods use arithmetic averages of the dissimilarities. The arithmetic averaging methods have no simple geometric interpretation. In contrast, the UPGMC and WPGMC methods have direct geometric interpretations when the objects are represented as patterns in a *d*-dimensional space. The centroid methods assess the dissimilarity between two clusters by the distance between centroids. The UPGMC method measures distance in terms of the centroid computed from all patterns in each cluster. The WPGMC method computes centroids from the centroids of the two clusters that merge to form a new cluster. The UPGMA weights the contribution of each pattern equally by taking into account the sizes of the clusters, while the WPGMC weights the patterns in small clusters more heavily than the patterns in large clusters. Centroid methods should only be used when the objects are represented as patterns and the proximity measure is squared Euclidean distance. An important distinction between centroid methods and other SAHN algorithms is in monotonicity, as explained later in this section.

Example 3.6

Dendrograms for the seven algorithms in Table 3.1 are drawn in Figure 3.13. The six objects involved are the six pattern vectors defined below in a three-dimensional space.

$$\mathbf{x}_1 = (1.0 \quad 2.0 \quad 2.0)^T \qquad \mathbf{x}_4 = (3.0 \quad 4.0 \quad 3.0)^T$$

$$\mathbf{x}_2 = (2.0 \quad 1.0 \quad 2.0)^T \qquad \mathbf{x}_5 = (0.0 \quad 3.5 \quad 3.5)^T$$

$$\mathbf{x}_3 = (0.0 \quad 1.0 \quad 3.0)^T \qquad \mathbf{x}_6 = (2.0 \quad 2.5 \quad 2.5)^T$$

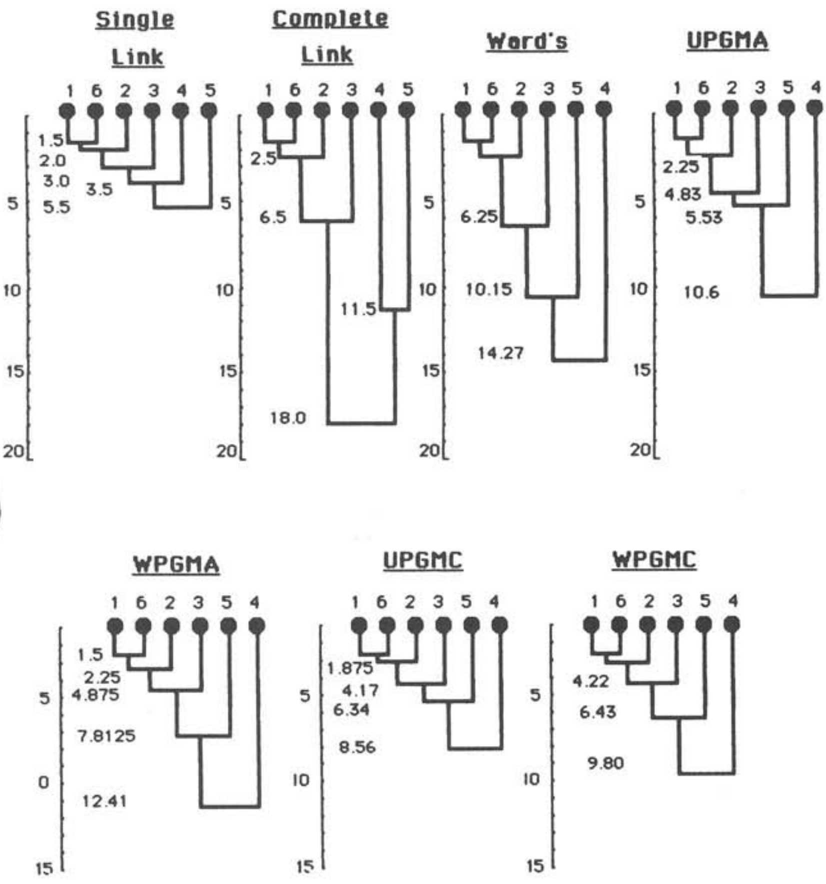
Under a squared Euclidean distance measure of dissimilarity, the proximity matrix is given below.

$$\begin{array}{c|cccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
1 & 0 & 2.0 & 3.0 & 9.0 & 5.5 & 1.5 \\
2 & & 0 & 5.0 & 11.0 & 12.5 & 2.5 \\
3 & & & 0 & 18.0 & 6.5 & 6.5 \\
4 & & & & 0 & \bigcirc\!\!\!\!\!\!\!\!\;11.5 & 3.5 \\
5 & & & & & 0 & 6.0 \\
6 & & & & & & 0 \\
 & & & & 9.5 & &
\end{array}$$

The tie in proximity between pairs of patterns (x_3, x_5) and (x_3, x_6) causes no ambiguity in any of the dendrograms. The dendrograms for Ward's method, the two arithmetic average methods, and the two centroid methods all have the same topology and differ only in levels. They suggest that x_4 is an outlier because it joins the cluster of the other five patterns last and the gap between the formation of the five-pattern cluster and the singleton cluster is large. The single-link dendrogram has much the same topology, except that x_5 now appears to be the outlier. The complete-link dendrogram establishes cluster (x_4, x_5). All dendrograms agree that (x_1, x_2, x_6) is a strong cluster. Quantitative measures of the strength and quality of clusters and clusterings, defined in Chapter 4, should help answer such questions as: If one of the dendrograms were to be cut to define a partition, where is the best cutting level? Is (x_1, x_2, x_6, x_3) a good cluster?

Several of the comparative studies discussed in Section 3.5.2 conclude that Ward's method (Ward, 1963), also called the minimum variance method, outperforms other hierarchical clustering methods. This method is based on notions of square error popularized in analysis-of-variance and other statistical procedures (Wilks, 1963; Cooley and Lohnes, 1971). Square-error criteria are also used in partitional clustering algorithms (Section 3.3.1). Ward's method is implemented by the standard algorithm using the constants in Table 3.1. These constants are derived below to see how square error is minimized.

Suppose that a clustering has been achieved with Ward's method and that the next clustering in the hierarchy is to be obtained with the matrix updating algorithm. Ward's method is designed for the situation when the data appear as patterns. Thus we begin with a set of n patterns in a d-dimensional space. Let $x_{ij}^{(k)}$ be the value for feature j of pattern i when pattern i is in cluster k for i from 1 to n_k and j from 1 to d. The centroid of cluster k, denoted $[m_1^{(k)}, \ldots, m_d^{(k)}]$, is the cluster center, or the average of the n_k patterns in cluster k.



ew figure

Figure 3.13 Examples of dendrograms for matrix updating algorithms.

$$m_j^{(k)} = (1/n_k) \sum_{i=1}^{n_k} x_{ij}^{(k)}$$

The square-error for cluster k is the sum of squared distances to the centroid for all patterns in cluster k.

$$e_k^2 = \sum_{i=1}^{n_k} \sum_{j=1}^{d} [x_{ij}^{(k)} - m_j^{(k)}]^2$$

The square-error for the entire clustering, which contains K clusters, is the sum of the square-errors for the individual clusters.

$$E_K^2 = \sum_{k=1}^{K} e_k^2$$

Ward's method merges the pair of clusters that minimizes ΔE_{pq}^2, the change in E_K^2 caused by the merger of clusters p and q into cluster t to form the next clustering.

Since the square-errors for all clusters except for the three clusters involved remain the same,

$$\Delta E_{pq}^2 = e_t^2 - e_p^2 - e_q^2$$

After a bit of algebra, we find that the change in square-error depends only on the centroids.

$$\Delta E_{pq}^2 = \frac{n_p n_q}{n_p + n_q} \sum_{j=1}^{d} [m_j^{(p)} - m_j^{(q)}]^2$$

The clusters p and q selected for merger are the clusters that minimize this quantity. The square-error must increase as the number of clusters decreases, but the increase is as small as possible in Ward's method. Once clusters p and q are merged into cluster t, the proximity between all other clusters and the new cluster t must be updated. Letting cluster r represent a cluster other than p, q, or t, the following formula can be applied to find $d[(r), (t)]$:

$$d[(r), (t)] = \frac{n_r + n_p}{n_r + n_t} d[(r), (p)] + \frac{n_r + n_q}{n_r + n_t} d[(r), (q)] - \frac{n_r}{n_r + n_t} d[(p), (q)]$$

The choice of a suitable hierarchical clustering method is an important matter in applications, but theory provides few guidelines for optimizing the choice. Square-error is a familiar criterion in engineering, so one might feel comfortable with a procedure that minimizes square-error, such as Ward's method. However, the objective of cluster analysis is to investigate the structure of the data, so the imposition of an apriori criterion, such as square-error, might not be appropriate. All data do not occur as patterns, so we cannot limit our thinking to geometrical constructs. Section 3.5.2 reviews several empirical studies that compare hierarchical clustering methods and that guide the choice of a clustering method.

3.2.8 Crossovers and Monotonicity in Dendrograms

Section 3.2.2 defined perfect hierarchical structure as a proximity matrix that satisfies the ultrametric inequality. The rationale was that the single-link and complete-link methods produced the same dendrograms for an ultrametric proximity matrix, and since these two methods search for very different types of structure, the fact that they exhibit the same exact structure is meaningful. It is clear from the single-link and complete-link algorithms based on threshold and proximity graphs that these methods are *monotonic*. That is, the level at which the next cluster forms is always larger, on a dissimilarity scale, than the level of the current clustering. Monotone methods induce ultrametric cophenetic matrices. Monotonicity can be expressed in mathematical terms by referring to the matrix updating formula in Section 3.2.7. If a clustering method merges clusters (r) and (s) into cluster (r, s), monotonicity demands that

$$d[(k), (r, s)] \geq d[(r), (s)]$$

$$\begin{array}{c|cccc}
 & 1 & 2 & 3 & 4 \\
1 & 0 & 1.2 & 2.3 & 2.2 \\
2 & & 0 & 2.4 & 2.6 \\
3 & & & 0 & 2.0 \\
4 & & & & 0
\end{array}$$

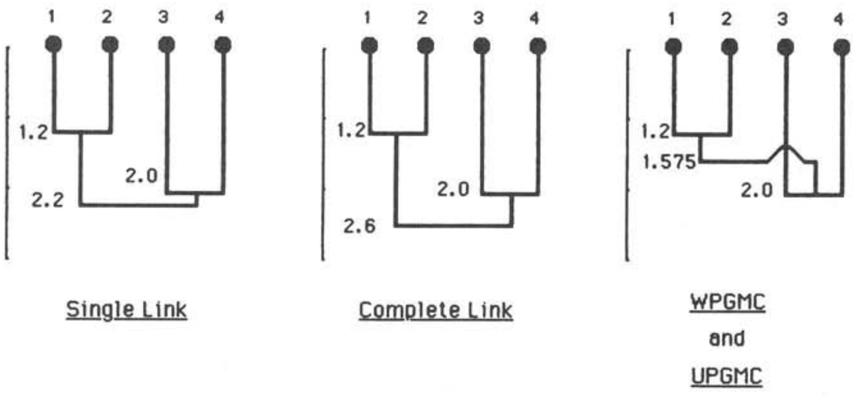


Figure 3.14 Examples of crossover or reversal in dendrogram.

for all clusters (k) distinct from (r) and (s). That is, no dissimilarity in the updated matrix can be smaller than the smallest entry in the previous matrix. Another way of saying this is that the cophenetic matrix generated by these two methods satisfies the ultrametric inequality.

What can be said about the monotonicity of SAHN algorithms expressed through matrix updating, especially those defined by the matrix updating algorithm in Table 3.1? Figure 3.14 provides a simple example of a dissimilarity matrix and the dendrograms generated by the single-link, complete-link, UPGMC, and WPGMC methods. The dendrograms from the centroid methods (UPGMC and WPGMC) are not monotone and exhibit what is called a ''crossover'' or a ''reversal'' since clusters (x_1, x_2) and (x_3, x_4) merge at a level lower than the level at which (x_3, x_4) is first defined.

Monotonicity is clearly a property of the clustering method and has nothing to do with the proximity matrix. The advantage of the matrix updating formula is that the monotonicity of any SAHN algorithm that can be expressed in terms of this updating formula can be predicted from the coefficients. Assuming that $\alpha_r > 0$, and $\alpha_s > 0$, Milligan (1979) provided the following results. The matrix updating formula for step 4 of the SAHN algorithm is repeated below for easy reference. Clusters (r) and (s) are being merged into cluster (r, s) and the dissimilarity between distinct cluster (k) and the newly formed cluster is being established.

$$d[(k), (r, s)] = \alpha_r d[(k), (r)] + \alpha_s d[(k), (s)] + \beta d[(r), (s)] + \gamma |d[(k), (r)] - d[(k), (s)]|$$

Result 1. If $\alpha_r + \alpha_s + \beta \geq 1$ and $\gamma \geq 0$, the clustering method is monotone.

This result is easily demonstrated. The first inequality can be rewritten as

$$\beta \geq 1 - \alpha_r - \alpha_s$$

and substituting in the matrix updating formula shows that

$$d[(k), (r, s)] \geq d[(r), (s)] + \alpha_r\{d[(k), (r)] - d[(r), (s)]\} + \alpha_s\{d[(k), (s)] - d[(r), (s)]\} + \gamma |d[(k), (r)] - d[(k), (s)]|$$

SAHN algorithms require that $d[(r), (s)]$ be no greater than either $d[(k), (r)]$ or $d[(k), (s)]$ for any distinct (k). Thus the condition that γ be nonnegative implies that

$$d[(k), (r, s)] \geq d[(r), (s)]$$

for all clusters (k) other than (r) and (s), which implies monotonicity.

Result 2. If $\alpha_r + \alpha_s + \beta \geq 1$ and $0 > \gamma \geq \max\{-\alpha_r, -\alpha_s\}$, the clustering method is monotone.

To demonstrate this result, first consider the case when $d[(k), (r)] > d[(k), (s)]$. Using the first inequality and recalling that γ is negative, the matrix updating equation can be rewritten as

$$d[(k), (r, s)] \geq d[(r), (s)] + (\alpha_r - |\gamma|)\{d[(k), (r)] - d[(r), (s)]\} + (\alpha_s + |\gamma|)\{d[(k), (s)] - d[(r), (s)]\}$$

Since $\alpha_r \geq |\gamma|$, the second term on the right is nonnegative for the case under consideration. The last term on the right is nonnegative because of the way SAHN algorithms are defined. Thus

$$d[(k), (r, s)] \geq d[(r), (s)]$$

as in Result 1, and the clustering method is monotone. The case when $d[(k), (s)] > d[(k), (r)]$ can be proved in a similar fashion.

The inequality in Result 1 is not satisfied for either of the centroid methods (UPGMC and WPGMC). It is easy to create examples for which these methods do not produce monotone hierarchies, as indicated in Figure 3.14. However, all other methods in Table 3.1 are monotone. Note that only single-link and complete-link clusterings are invariant under monotone transformation of the dissimilarities.

Figure 3.13 demonstrates that nonmonotone clustering methods do not necessarily produce crossovers. ~~The discussion of the centroid methods suggests that they should only be applied when objects are patterns in a pattern space so crossovers~~
Crossovers can occur even when the object are patterns in a pattern space and Euclidean distance is the measure of proximity.

~~will be eliminated.~~ One is tempted to reject nonmonotone methods out of hand. Williams and Lance (1977) call them obsolete. Anderberg (1973) claims that they lack a useful interpretation for general proximities. Sneath and Sokal (1973) claim that "the frequency of reversals and the relatively high degree of its distortion of the original [proximity] matrix has led to the abandonment of this [UPGMC] technique." On the other hand, the performances of nonmonotone methods in several comparative studies of clustering methods discussed in Section 3.5.2 do not suggest that such methods be abandoned. Williams et al. (1971) argue that monotonicity is not essential for the proper performance of hierarchical clustering.

3.2.9 Clustering Methods Based on Graph Theory

The statements of the single-link and complete-link algorithms in terms of graph theory in Section 3.2.1 suggest that properties other than connectedness and completeness can be used to define clustering methods. The idea is to watch the sequence of threshold graphs or proximity graphs for the appearance of a suitable property. Hubert (1974a) suggests the following expression of algorithms that define hierarchical clustering methods. Ties in the proximities can affect the clusterings in unexpected ways, so we assume that no ties exist in the proximity matrix.

New hierarchical clustering algorithms are formed by changing step 2 in the algorithm of Section 3.2.1. The function $Q_{\rho(k)}$ is defined as follows for all pairs of clusters $\{C_{mr}, C_{mt}\}$ in the clustering $\{C_{m1}, \ldots, C_{m(n-m)}\}$:

$$Q_{\rho(k)}(r, t) = \min \{d(i, j) : \text{the maximal subgraph of } G[d(i, j)] \text{ defined by } C_{mr} \cup C_{mt} \text{ is connected and either has property } \rho(k) \text{ or is complete}\}$$

Following the algorithm, clusters C_{mp} and C_{mq} are merged to form the next clustering in the sequence if

$$Q_{\rho(k)}(p, q) = \min \{Q_{\rho(k)}(r, t)\}$$

Some examples of property ρ are given below. Integer k is a parameter, so, for example, $\rho(k)$ could mean a node connectivity of k or a node degree of k.

Node connectivity. The node connectivity of a connected subgraph is the largest number n_c such that all pairs of nodes are joined by at least n_c paths having no nodes in common.

Edge connectivity. The edge connectivity of a connected subgraph is the largest integer n_e such that all pairs of nodes are joined by at least n_e paths having no edges in common.

Node degree. The degree of a connected subgraph is the largest integer n_d such that each node has at least n_d incident edges.

Diameter. The diameter of a connected subgraph is the maximum "distance" between two nodes in the subgraph. The distance between two nodes is the number of edges in the shortest path joining them.

Radius. The radius of a connected subgraph is the smallest integer n_r such that at least one node is within distance n_r of all other nodes in the subgraph.

Specifying parameter k and property ρ defines a new clustering method. Every cluster must at least be connected. Once all the edges have been inserted into the subgraph, it is complete and no further properties can be applied. Certain practical difficulties arise when trying to select a suitable property. Few guidelines exist other than intuition and experience. Theorems from mathematics provide some insight into these methods. For example, a node connectivity of k implies an edge connectivity of k, but the reverse is not true. Similarly, an edge connectivity of k implies a minimum degree of k, but the reverse does not hold. A compelling reason must appear before one of these methods is used in place of the single-link, complete-link, or other SAHN algorithms.

Example 3.7

Figure 3.15 demonstrates hierarchical clustering methods defined by graph properties. An ordinal proximity matrix is given below on eight objects. Threshold graph $G(13)$ is pictured in Figure 3.15(a) to help in establishing the dendrograms for several methods, as well as for the single-link and complete-link methods. Proximity dendrograms are shown in Figure 3.15(b)–(h). A simple way to find these hierarchies with pencil and paper is first to list the pairs of objects in rank order by proximity. Then construct a sequence of threshold graphs and find the first threshold graph at which a property is satisfied. It is important to check the property only on the subgraph formed by the union of the subgraphs for two *existing* clusters.

$$\begin{array}{c|cccccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\hline
1 & 0 & 13 & 21 & 18 & 4 & 8 & 7 & 28 \\
2 & - & 0 & 9 & 19 & 15 & 14 & 10 & 16 \\
3 & - & - & 0 & 22 & 20 & 12 & 11 & 17 \\
4 & - & - & - & 0 & 3 & 23 & 27 & 1 \\
5 & - & - & - & - & 0 & 5 & 24 & 2 \\
6 & - & - & - & - & - & 0 & 6 & 25 \\
7 & - & - & - & - & - & - & 0 & 26 \\
8 & - & - & - & - & - & - & - & 0
\end{array}$$

Ling (1972) examined hierarchical clustering based on notions of connectivity and compactness that are particularly appropriate for ordinal proximity matrices. He assumed ordinal proximities with no ties, but his clustering method can also be applied to interval and ratio proximity matrices. The properties ρ that Ling proposed are defined below.

Consider a proximity graph on n nodes. A subgraph is *r-connected* if all pairs of nodes in the subgraph are connected by r-chains. An *r-chain* between two nodes is a sequence of nodes having $d(i, j) \leq r$ for all pairs (i, j) of nodes in the sequence. A subgraph is *(k, r)-bonded* if every node in the subgraph is directly connected to at least k nodes and if $d(i, j) \leq r$ for all k connections. Finally, a subgraph is *(k, r)-connected* if it is both r-connected and (k, r)-bonded.

A subgraph becomes a (k, r)-*cluster* in the algorithm of Section 3.2.1 as soon as it becomes (k, r)-connected. Note that $(1, r)$ clusters are single-link clusters. Ling (1972, 1973a) proposed the (k, r)-cluster as a way of identifying significant clusters. Given any proximity graph, a (k, r)-cluster can be defined independent of the hierarchical clustering algorithm. A subgraph is a (k, r)-cluster if r is the smallest value of s for which the subgraph is (k,s)-connected for some s and the subgraph is not properly contained in any other (k, t)-connected subgraph for $t > r$. Such clusters have several attractive mathematical properties and are intuitively appealing since both connectedness and compactness are involved in the definition.

We emphasize that no theory exists for choosing among the various properties of graphs to select the ''best'' clustering method for a particular application. Section 3.5 provides some guidance, but familiarity with a method and confidence in the results of previous applications of the method are the only practical ways of choosing a method.

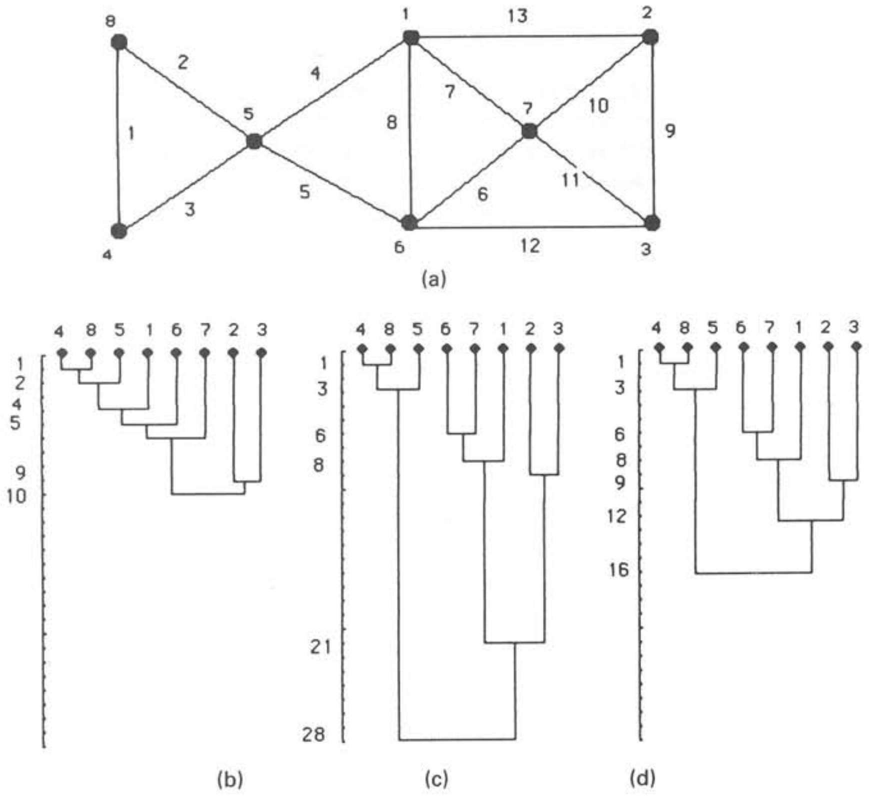


Figure 3.15 Examples of dendrograms from graph theory: (a) threshold graph G(13) for proximity matrix in example 3.7; (b) single-link; (c) complete-link; (d) 2-node connected; (e) 2-edge connected; (f) 2-degree; (g) 2-diameter; (h) 2-radius.

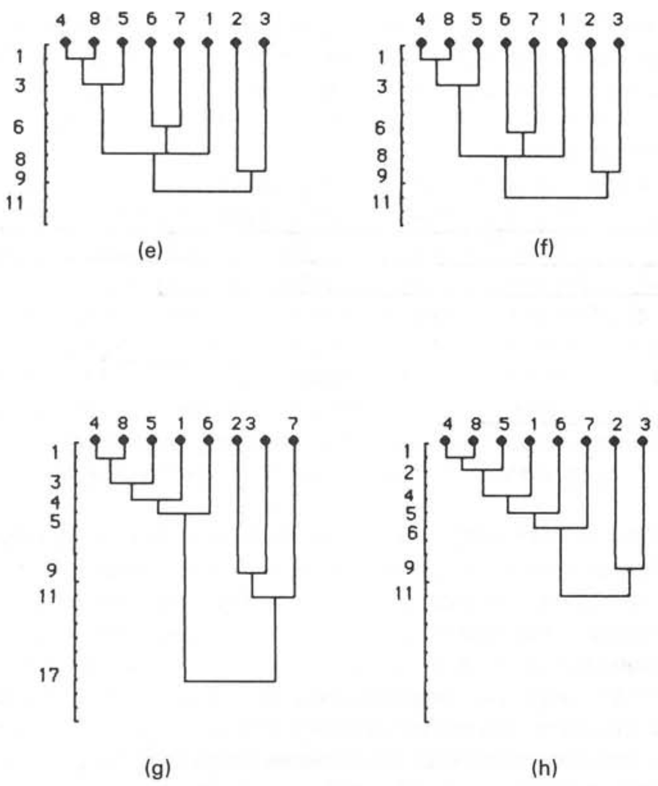


Figure 3.15 *(continued)*

3.3 PARTITIONAL CLUSTERING

Hierarchical clustering techniques organize the data into a nested sequence of groups. An important characteristic of hierarchical clustering methods is the visual impact of the dendrogram, which enables a data analyst to see how objects are being merged into clusters or split at successive levels of proximity. The data analyst can then try to decide whether the entire dendrogram describes the data or can select a clustering, at some fixed level of proximity, which makes sense for the application in hand. We refer to nonhierarchical clustering methods as *partitional* clustering methods. They generate a single partition of the data in an attempt to recover natural groups present in the data. Both clustering strategies have their appropriate domains of applications. Hierarchical clustering methods generally require only the proximity matrix among the objects, whereas partitional techniques expect the data in the form of a pattern matrix. It is generally assumed that the features have been measured on a ratio scale.

Hierarchical techniques are popular in biological, social, and behavioral sciences because of the need to construct taxonomies. Partitional techniques are used frequently in engineering applications where single partitions are important. Partitional clustering methods are especially appropriate for the efficient representation and compression of large data bases. Dendrograms are impractical with more than a few hundred patterns.

The problem of partitional clustering can be formally stated as follows. Given n patterns in a d-dimensional metric space, determine a partition of the patterns into K groups, or clusters, such that the patterns in a cluster are more similar to each other than to patterns in different clusters. The value of K may or may not be specified. A clustering criterion, such as square-error, must be adopted. Criteria can be classified as global or local. A *global* criterion represents each cluster by a prototype and assigns the patterns to clusters according to most similar prototypes. A *local* criterion forms clusters by utilizing local structure in the data. For example, clusters can be formed by identifying high-density regions in the pattern space or by assigning a pattern and its k nearest neighbors to the same cluster.

The theoretical solution to this partitional problem is straightforward. Simply select a criterion, evaluate it for all possible partitions containing K clusters, and pick the partition that optimizes the criterion. The first difficulty encountered is selecting a criterion that translates one's intuitive notions about "cluster" into a reasonable mathematical formula. Criteria are highly dependent on problem parameters and must be simple for computational reasons but complex enough to reflect various data structures. The second difficulty with this approach is that the number of partitions is astronomical, even for moderate numbers of patterns, so evaluating even the simplest criterion over all partitions is impractical.

Let $S(n, K)$ denote the number of clusterings of n objects into K clusters. The order of the objects in each cluster and the order of the clusters themselves are immaterial. Empty clusters are not counted. A partial difference equation can be written for $S(n, K)$ as follows. Suppose that all clusterings of $n - 1$ objects have been listed. A clustering of n objects can be formed from this list in two ways.

1. The nth object can be added as a singleton cluster to each member of the list with exactly $(K - 1)$ clusters.
2. The nth object can be added to each cluster of any member of the list with exactly K clusters.

Thus

$$S(n, K) = S(n - 1, K - 1) + KS(n - 1, K)$$

The boundary conditions on this equation are

$$S(n, 1) = 1,\ S(n, n) = 1,\ S(n, K) = 0 \qquad \text{if } K > n$$

The solution to this equation for $S(n, K)$ requires that values $\{S(j, p)\}$ be known for the set $\{(j, p) : 1 \leq j \leq n - 2,\ 1 \leq p \leq K\}$.

Solutions to the partial difference equation are called Stirling numbers of the second kind (Fortier and Solomon, 1966; Jensen, 1969):

$$S(n, K) = \frac{1}{K!} \sum_{i=1}^{K} (-1)^{K-i} \binom{K}{i} (i)^n$$

There are only 34,105 distinct partitions of 10 objects into four clusters, but this number explodes to approximately 11,259,666,000 if 19 objects are to be partitioned into four clusters. Clearly, exhaustive enumeration of all possible partitions is not computationally feasible even for small numbers of patterns. In addition, K, the number of clusters, must be selected a priori. Chapter 4 discusses this problem.

To avoid this combinatorial explosion, a criterion function is evaluated only for a small set of "reasonable" partitions. How to identify a small subset of partitions that has a good chance of containing the optimal partition? The most common approach is to optimize the criterion function using an iterative, hill-climbing technique. Starting with an initial partition, objects are moved from one cluster to another in an effort to improve the value of the criterion function. Thus each successive partition is a perturbation of the previous one and, therefore, only a small number of partitions is examined. Algorithms based on this technique are computationally efficient but often converge to local minima of the criterion function. Several heuristics for choosing the initial partition, moving objects from one cluster to the other, and for merging and splitting clusters will be discussed later.

Another way of avoiding the combinatorial explosion is somehow to identify and reject a large number of partitions which are not likely to be of interest. Jensen (1969) uses a dynamic programming approach to eliminate many partitions and is still able to achieve an optimal solution. A significant computational savings is realized, especially for large clustering problems, at the expense of algorithmic complexity. For example, to partition 19 objects into four clusters, less than 2% of the total number of partitions need to be evaluated using the dynamic programming formulation. Even this reduction is not enough to make this approach computationally feasible for practical clustering problems. Several related approaches are described in the literature (Edwards and Cavalli-Sforza, 1965; Vinod, 1969; Rao, 1971; Koontz et al., 1975; Lefkovitch, 1980).

There is no single "best" criterion for obtaining a partition because no precise and workable definition of "cluster" exists. Clusters can be of arbitrary shapes and sizes in a multidimensional pattern space. Each clustering criterion imposes a certain structure on the data, and if the data happen to conform to the requirements of a particular criterion, the true clusters are recovered. Only a small number of independent clustering criteria can be understood both mathematically and intuitively. Thus the hundreds of criterion functions proposed in the literature are related and the same criterion appears in several disguises. Shaffer et al. (1979) demonstrate the similarity of a mode-seeking partitional algorithm (Kittler, 1976) and the MST-based algorithm of Zahn (1971). Similarly, Urquhart (1982) shows that partitions obtained from a relative neighborhood graph are identical to those generated by mutual near-neighbor clustering (Gowda and Krishna, 1978). The literature of cluster analysis is spread so widely and over so many areas of science that a single criterion function is rediscovered repeatedly.

In this section we present some of the most common partitional clustering methods. Several popular criteria are versions of square-error and are discussed in Sections 3.3.1 to 3.3.3. Another global criterion obtains a partition by fitting a mixture density model to the patterns (Section 3.3.4). These clustering criteria create clusters having hyperellipsoidal shapes. Sections 3.3.5 to 3.3.7 describe several partitional clustering methods based on local criteria of density or mode estimation, graph connectivity, and near-neighbor relationships. Finally, Section 3.3.8 briefly covers the topic of fuzzy clustering, where each object is permitted to belong to more than one cluster with a grade of membership.

Most of the partitional clustering techniques presented here implicitly assume continuous-valued feature vectors so that the patterns can be viewed as being embedded in a metric space. If the features are on a nominal or ordinal scale, Euclidean distances and cluster centers are not very meaningful, so hierarchical clustering methods are normally applied. Wong and Wang (1979) proposed a clustering algorithm for discrete-valued data. The approach is similar to the mode estimation procedure for continuous data but approximates the high-order discrete probability distribution by a second-order product and uses Hamming distance (Section 2.2) between patterns.

The technique of conceptual clustering or learning from examples (Michalski and Stepp, 1983) can be used with objects represented by nonnumeric or symbolic descriptors. The objective here is to group objects into conceptually simple classes. Clustering of trains using attributes such as number of cars, number of wheels, colors of wheels, and number of items carried, and the clustering of microcomputers using attributes such as CPU speed, memory size, and type of processor are more appropriately handled by associating each cluster with a simple "concept." Concepts are defined in terms of attributes. For example, in the train classification problem, trains with two red cars is a concept. Objects are arranged into a hierarchy of classes described by concepts.

3.3.1 Square-Error Clustering Criteria

The most commonly used partitional clustering strategy is based on the square-error criterion. The general objective is to obtain that partition which, for a fixed number of clusters, minimizes the square-error. Ward's method of hierarchical clustering (Section 3.2.7) uses square-error in a different way. Minimizing square-error, or within-cluster variation, will be shown to be equivalent to maximizing the between-cluster variation.

Suppose that the given set of n patterns in d dimensions has somehow been partitioned into K clusters $\{C_1, C_2, \ldots, C_K\}$ such that cluster C_k has n_k patterns and each pattern is in exactly one cluster, so that

$$\sum_{k=1}^{K} n_k = n$$

The mean vector, or center, of cluster C_k is defined as the centroid of the cluster, or

$$\mathbf{m}^{(k)} = (1/n_k) \sum_{i=1}^{n_k} \mathbf{x}_i^{(k)}$$

where $\mathbf{x}_i^{(k)}$ is the *i*th pattern belonging to cluster C_k. The square-error for cluster C_k is the sum of the squared Euclidean distances between each pattern in C_k and its cluster center $\mathbf{m}^{(k)}$. This square-error is also called the within-cluster variation.

$$e_k^2 = \sum_{i=1}^{n_k} (\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)})^{\mathrm{T}}(\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)})$$

The Mahalanobis distance (Section 2.2) can also be used to define square-error.

The square-error for the entire clustering containing K clusters is the sum of the within-cluster variations:

$$E_K^2 = \sum_{k=1}^{K} e_k^2$$

The objective of a square-error clustering method is to find a partition containing K clusters that minimizes E_K^2 for fixed K. The resulting partition has also been referred to as the minimum variance partition. Figure 3.16 illustrates that the square-error criterion views the centroids of clusters as prototypes. The error represents deviations of the patterns from the centroids. In other words, the patterns are viewed as a collection of K spherically shaped swarms. Square-error clustering tries to make the K swarms as compact and separated as possible.

Gordon and Henderson (1977) also define the clustering problem in terms of minimizing the within-cluster sum of square distances. However, they write their criterion function in such a way that the clustering problem can be formulated as a nonlinear programming problem. Let x_{ij} denote the *j*th feature of the *i*th

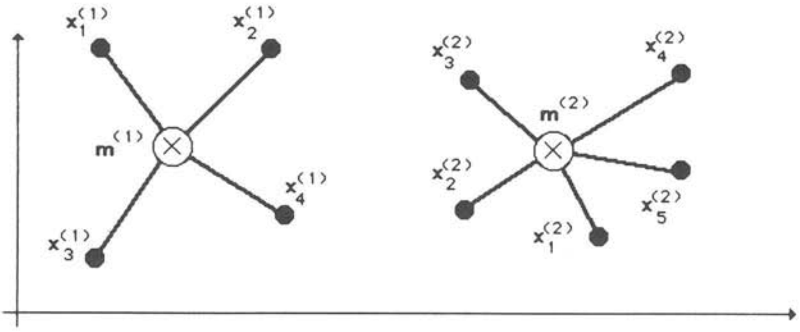


Figure 3.16 Distances used in computing square-error.

pattern, $i = 1, \ldots, n; j = 1, \ldots, d$. Let $y_{ik} = 1$ if the ith pattern belongs to the kth cluster, and 0 if the ith pattern does not belong to the kth cluster, $k = 1, \ldots, K$. The centroid of the kth cluster, $\mathbf{z}_k$, is written as $\mathbf{z}_k = (z_{k1}, \ldots, z_{kd})$, where

$$z_{kj} = \sum_{i=1}^{n} (y_{ik}x_{ij}) \Big/ \sum_{i=1}^{n} y_{ik}$$

The total within-cluster variation, denoted as S_T, can be written as

$$E_K^2 = \quad S_T = \sum_{i=1}^{n} \sum_{k=1}^{K} y_{ik} \sum_{j=1}^{d} (x_{ij} - z_{kj})^2$$

Gordon and Henderson (1977) proposed the following two formulations for minimizing S_T.

1. Given the pattern matrix, $\mathcal{X}$, and the number of clusters, K, find the $n \times K$ matrix $\mathcal{Y} = [y_{ik}]$ that minimizes S_T. Note that $\mathcal{Y}$ is an $n \times K$ matrix of 0's and 1's with exactly one 1 in each row and at least one 1 in each column.
2. A more general formulation minimizes S_T under the assumption that $y_{ik} \in [0, 1]$ subject to the constraints

$$\sum_{k=1}^{K} y_{ik} = 1 \qquad \text{and} \qquad y_{ik} \geq 0$$

The term y_{ik} denotes the fraction of the ith pattern that is assigned to the kth cluster. This concept is similar to fuzzy clustering (Section 3.3.8), where a pattern belongs to a cluster with a "grade of membership." A lemma by Gordon and Henderson shows that the matrix $\mathcal{Y}$ which minimizes S_T under these constraints must contain only 0's and 1's. Thus the first formulation is a special case of the second formulation. The algorithm used to minimize S_T is based on the method of steepest descent.

A number of clustering criteria related to square-error have been derived from scatter matrices used in discriminant analysis. They are based on the following decomposition of the total scatter matrix, $\mathcal{S}$, into the within-cluster scatter matrix, $\mathcal{S}_W$, and the between-cluster scatter matrix, $\mathcal{S}_B$ (see Appendix D).

$$\mathcal{S} = \mathcal{S}_B + \mathcal{S}_W$$

The total scatter matrix $\mathcal{S}$ is fixed, no matter how the given patterns are partitioned. These matrices are scalars in one dimension, so it should be clear that increasing the between-cluster scatter $\mathcal{S}_B$ decreases the within-cluster scatter $\mathcal{S}_W$, and vice versa. To define clustering criteria in terms of scatter matrices, we need to represent the "size" of clusters by the trace and the determinant operators. A number of clustering criteria are described below in terms of scatter (Friedman and Rubin, 1967).

We first show that a clustering criterion defined by either the trace of $\mathcal{S}_W$ or $\mathcal{S}_B$ is identical to the square-error criterion.

$$\operatorname{tr}(\mathcal{S}_W) = \sum_{k=1}^{K} \operatorname{tr}(\mathcal{S}^{(k)}) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} (\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)})^{\mathrm{T}}(\mathbf{x}_i^{(k)} - \mathbf{m}^{(k)}) = E_K^2$$

Note that tr $(\mathcal{S}^{(k)})$ is the sum of variances along the feature directions for cluster k and measures the compactness of cluster C_k. Minimizing tr $(\mathcal{S}_W)$ is identical to maximizing tr $(\mathcal{S}_B)$ because

$$\operatorname{tr}(\mathcal{S}) = \operatorname{tr}(\mathcal{S}_W) + \operatorname{tr}(\mathcal{S}_B)$$

and tr $(\mathcal{S})$ is the same for any partition. The trace criterion, and the equivalent square-error criterion, are invariant under orthogonal transformations (Appendix D) of the pattern space, such as rotations, but are not invariant under nonsingular linear transformations. That is, the minimum square-error partition may change if the coordinate axes are scaled.

The determinant of $\mathcal{S}_B$ is not a good clustering criterion because $\mathcal{S}_B$ becomes singular when the number of clusters is less than the number of features, $K < d$. Minimizing the determinant of $\mathcal{S}_W$ has an advantage over the square-error criterion in that $\mathcal{S}_W$ is invariant to nonsingular linear transformations (Appendix D) of the patterns. However, $\mathcal{S}_W$ becomes singular if $(n - K) < d$ or if the patterns lie in a subspace of the feature space. A linear transformation (Sections 2.4.1 and 2.4.3) can be used to reduce the dimensionality of the data when $\mathcal{S}_W$ is singular.

Section 2.4.3 showed that the eigenvectors of $\mathcal{S}_W^{-1}\mathcal{S}_B$ define a projection of the patterns to a space of $K - 1$ dimensions. These eigenvectors are invariant under nonsingular linear transformations of the pattern matrix. Further, since the eigenvalues of $\mathcal{S}_W^{-1}\mathcal{S}_B$ determine the ratio of between-cluster to within-cluster scatter, we can define two additional clustering criterion. Assuming that there are $m \le \min\{d, K - 1\}$ significant eigenvalues, $\{\zeta_1, \zeta_2, \ldots, \zeta_m\}$ of $\mathcal{S}_W^{-1}\mathcal{S}_B$, these criteria can be expressed as

$$\operatorname{tr}(\mathcal{S}_W^{-1}\mathcal{S}_B) = \sum_{i=1}^{m} \zeta_i \qquad \text{and} \qquad |\mathcal{S}|/|\mathcal{S}_W| = \prod_{i=1}^{m} (1 + \zeta_i)$$

Clustering methods based on these criteria choose that partition for which tr $(\mathcal{S}_W^{-1}\mathcal{S}_B)$ or $|\mathcal{S}|/|\mathcal{S}_W|$ is maximized. Note that maximizing $|\mathcal{S}|/|\mathcal{S}_W|$ is the same as minimizing $|\mathcal{S}_W|$ because $|\mathcal{S}|$ is independent of the partition of the data.

The clustering criteria above look for globular or hyperellipsoidal clusters. Different criterion lead to different clusterings. The square-error criterion is less demanding computationally than the criteria based on scatter ratios because the latter require the computation of eigenvalues after every partition. Unfortunately, there is no general guideline available for choosing one criterion over the other. In practice, one should generate partitions using different criterion functions and then choose the "best" one under some validation scheme (Chapter 4).

Example 3.8

Suppose that the four two-dimensional pattern vectors shown in Figure 3.17 are to be clustered (Duda and Hart, 1973). It is easy to compute the scatter matrices for the following three partitions (Duda and Hart, 1973).

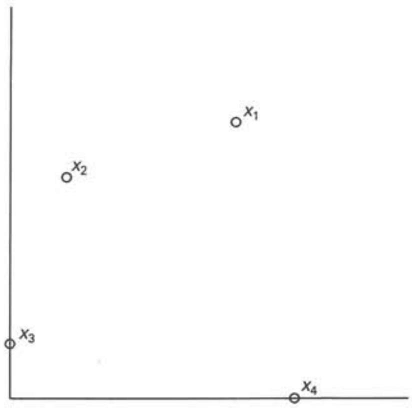


Figure 3.17 Clustering of four patterns in two dimensions.

$(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4\})$	$\text{tr}(\mathcal{S}_W) = 18$	$\lvert\mathcal{S}_W\rvert = 16$
$(\{\mathbf{x}_1, \mathbf{x}_4\}, \{\mathbf{x}_2, \mathbf{x}_3\})$	$\text{tr}(\mathcal{S}_W) = 18$	$\lvert\mathcal{S}_W\rvert = 16$
$(\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}, \{\mathbf{x}_4\})$	$\text{tr}(\mathcal{S}_W) = 52/3$	$\lvert\mathcal{S}_W\rvert = 64/3$

Thus the third partition is best of the three according to the tr $(\mathcal{S}_W)$ criterion, which is equivalent to the square-error criterion. However, the first two partitions are selected by the $|\mathcal{S}_W|$ criterion.

3.3.2 Square-Error Clustering Methods

The basic idea of an iterative clustering algorithm is to start with an initial partition and assign patterns to clusters so as to reduce square-error. The square-error tends to decrease as the number of clusters increases and can be minimized only for a fixed number of clusters. An iterative partitional clustering method can be implemented in several different ways. Different implementations can lead to different partitions. Dubes and Jain (1976) emphasize the distinction between clustering methods and clustering algorithms. A clustering method specifies the general strategy for grouping the patterns into clusters such as minimizing square-error or maximizing tr $(\mathcal{S}_W^{-1}\mathcal{S}_B)$. A clustering algorithm, on the other hand, is a computer program that implements a strategy and incorporates various heuristics. A general algorithm for iterative partitional clustering method is given below. Anderberg (1973) provides an extensive discussion of several details of this approach.

ALGORITHM FOR ITERATIVE PARTITIONAL CLUSTERING

Step 1. Select an initial partition with K clusters.

Repeat steps 2 through 5 until the cluster membership stabilizes.

Step 2. Generate a new partition by assigning each pattern to its closest cluster center.

Step 3. Compute new cluster centers as the centroids of the clusters.

Step 4. Repeat steps 2 and 3 until an optimum value of the criterion function is found.

Step 5. Adjust the number of clusters by merging and splitting existing clusters or by removing small, or outlier, clusters.

The details of the steps in this algorithm must either be supplied by the user as parameters or be implicitly hidden in the computer program. However, these details are crucial to the success of the program. A big frustration in using clustering programs is the lack of guidelines available for choosing details. We briefly review some of the crucial parameters and the options available (Anderberg, 1973; Dubes and Jain, 1980).

Initial partition. An initial partition can be formed by first specifying a set of K seed points. Seed points can be the first K patterns or K patterns chosen randomly from the pattern matrix. A set of K patterns that are well separated from each other can be obtained by taking the centroid of the data as the first seed point and selecting successive seed points which are at least a certain distance away from the seed points already chosen. The initial partition or clustering is formed by assigning each pattern to the closest seed point. The centroids of the resulting clusters are the initial cluster centers. Hierarchical clustering of the data can also be used to select an initial partition.

Different initial partitions can lead to different final clusterings because algorithms based on square-error can converge to local minima. This is especially true if the clusters are not separated well. One way to overcome local minima is to run the partitional algorithm with several different initial partitions. If they all lead to the same final partition, we have some confidence that the global minimum of square-error has been achieved.

Updating the partition. Partitions are updated by reassigning patterns to clusters in an attempt to reduce the square-error. The term "pass" or "cycle" refers to the process of examining the cluster label of every pattern once. McQueen (1967) defined a *K-means pass* as an assignment of all patterns to the closest cluster center. The center of the gaining cluster is recomputed after each new assignment in McQueen's K-means method. Forgy's method (Forgy, 1965) recomputes cluster centers after all patterns have been examined. The Euclidean metric is the most common metric for computing the distance between a pattern and a cluster center but Mahalanobis distance (Section 2.2.1) is also used. However, Mahalanobis distance requires computation of the inverse of the sample covariance matrix every time a pattern changes its cluster label.

Friedman and Rubin (1967) define a *hill-climbing pass* and a *forcing pass* in their clustering algorithm based on an invariant criteria using scatter matrices. A hill-climbing pass changes the cluster label of a pattern only to improve the criterion function. Remember that a *K*-means pass assigns every pattern to its closest cluster center. A forcing pass perturbs the partition to avoid getting trapped at a local minimum of the criterion function. A forcing pass tries each pattern of a cluster in a different cluster. The criterion function is recalculated after each test, the best partition found is retained, and the forcing pass is repeated for the next cluster. These passes are applied repeatedly until convergence is obtained.

Adjusting the number of clusters. Some clustering algorithms can create new clusters or merge existing clusters if certain conditions are met. This capability allows an algorithm to recover from poor initial partitions and lets it select a ''natural'' or ''suitable'' number of clusters, especially if the number of clusters desired is not appropriate. In one of the popular partitional clustering algorithms called ISODATA (Ball and Hall, 1964), these conditions are determined from parameters specified by the user of the program. A cluster is split if it has too many patterns and an unusually large variance along the feature with largest spread. Two clusters are merged if their cluster centers are sufficiently close, again based on a parameter supplied by the user.

An outlier is a pattern that is sufficiently far removed from the rest of the data to suspect that it was included by error, such as a mistake in data entry. Quite often an outlier is due to noise in the measurement process or error in data coding. Outliers can provide useful information about the underlying data generation process, but forcing an outlier to belong to a cluster distorts the shape of that cluster. Figure 3.18 demonstrates that an outlier can force a partitional clustering

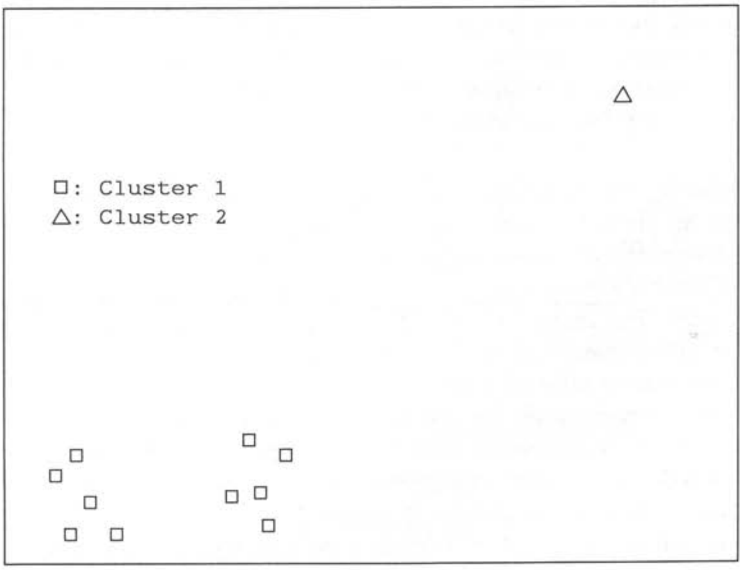


Figure 3.18 Effect of outlier in distorting a clustering.

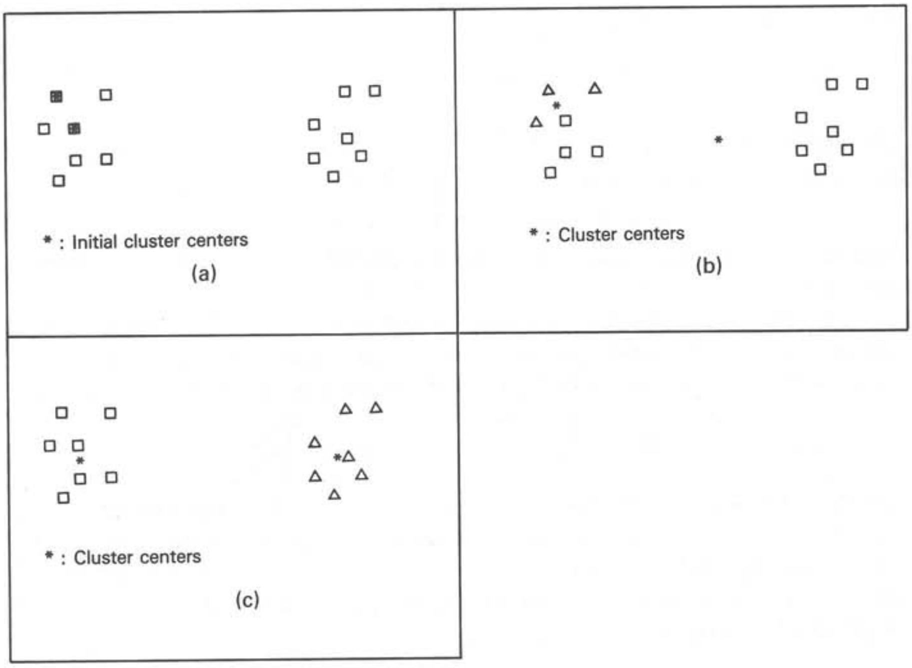


Figure 3.19 Convergence of K-means clustering: (a) initial data; (b) cluster membership after first loop; (c) cluster membership after second loop.

algorithm to put two compact and well-separated groups into the same cluster. Thus it is best to identify an outlier and remove it from further consideration. Some clustering algorithms also treat ''small'' clusters as outliers.

Convergence. When does the algorithm stop? Partitional algorithms terminate when the criterion function cannot be improved. There is no guarantee that an iterative algorithm will reach a global minimum. Some algorithms stop when the cluster labels for all the patterns do not change between two successive iterations. A maximum number of iterations can be specified to prevent endless oscillations. In practice, *K*-means type algorithms converge rapidly. Figure 3.19 shows two well-separated clusters in two dimensions. Even though the two initial seed points belong to the same cluster, the convergence of the *K*-means algorithm to the correct partition requires only two iterations.

Selim and Ismail (1984) rigorously prove convergence of the *K*-means algorithm. The problem of partitioning n d-dimensional patterns into K clusters can be formulated as the following mathematical programming problem. Minimize the weighted sum of Euclidean distances between patterns and cluster centers,

$$f(\mathcal{W}, \mathcal{M}) = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ki} d(\mathbf{x}_i, \mathbf{m}^{(k)})$$

subject to the constraint

$$\sum_{k=1}^{K} w_{ki} = 1, \qquad i = 1, 2, \ldots, n \quad \text{and} \quad w_{ki} \in \{0, 1\}$$

The matrix $\mathcal{W} = [w_{ki}]$ is a $K \times n$ matrix of weights for each pattern in each cluster and $\mathcal{M}$ is the $d \times K$ matrix of cluster centers.

$$\mathcal{M} = [\mathbf{m}^{(1)} \quad \mathbf{m}^{(2)} \quad \cdots \quad \mathbf{m}^{(K)}]$$

This is similar to the formulation of Gordon and Henderson (1977) discussed in Section 3.3.1.

The function $f(\mathcal{W}, \mathcal{M})$ is nonconvex and its local minimum need not be a global minimum. Frieze (1980) has also studied similar optimization problems. Consider the data set shown in Figure 3.20. If the two initial cluster centers are

$$\mathbf{m}^{(1)} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \qquad \text{and} \qquad \mathbf{m}^{(2)} = \begin{bmatrix} 6 \\ 2 \end{bmatrix}$$

then the K-means algorithm will stop after one iteration and converge to the solution $w_{11} = w_{12} = w_{23} = w_{24} = 1$, $w_{ij} = 0$ otherwise, which yields two clusters $\{\mathbf{x}_1, \mathbf{x}_2\}$ and $\{\mathbf{x}_3, \mathbf{x}_4\}$ and $f(\mathcal{W}, \mathcal{M}) = 8$. Selim and Ismail show that this solution is not even a local minimum because by slightly perturbing $\mathcal{M}$, $f(\mathcal{W}, \mathcal{M}) = 6$ can be achieved with the cluster centers

$$\mathbf{m}^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \text{and} \qquad \mathbf{m}^{(2)} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

Pollard (1981) also provides conditions for the almost-sure convergence of the cluster centers in K-means clustering as the number of patterns increases.

Computation. The computational complexity of this algorithm is of the order $O(ndKT)$, where n is the number of patterns, d the number of features, K

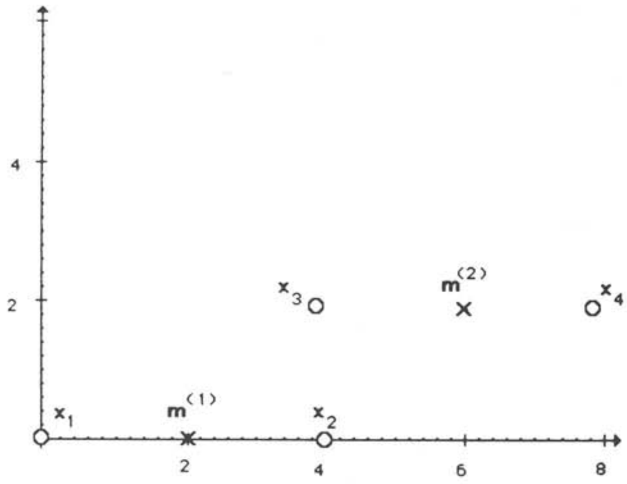


Figure 3.20 Convergence of square-error clustering algorithm.

the number of clusters desired, and T the number of iterations. The value of T depends on the initial cluster centers, distribution of patterns, and the size of the clustering problem. In practice, however, the user specifies an upper bound on the value of T. The iterative nature of the square-error clustering methods demands substantial processing time, even for a few hundred patterns. Two approaches have recently been taken to speed up square-error clustering algorithms that utilize advances in microelectronics technology: parallel processing and VLSI architecture. Tilton and Strong (1984) describe the performance of the ISODATA clustering algorithm when implemented on the MPP (Massively Parallel Processor), which contains an array of 128×128 microprocessors. A clustering problem involving a 512×512 digital image and 16 clusters required 20 seconds on the MPP compared to 7 hours on a VAX-11/780 superminicomputer. (See Chapter 5 for a brief review of digital image processing and image segmentation.) Ni and Jain (1985) present a systolic architecture for a square-error pattern clustering whose design has a potential performance gain of 1300 times over a serial processor.

3.3.3 Square-Error Clustering Programs

We now explain two examples of square-eror clustering programs, called FORGY and CLUSTER and compare their performance on two data sets. Clustering software is discussed in Section 3.4.

FORGY is the simplest and most straightforward square-error clustering program (Forgy, 1965). It uses only the K-means pass. The cluster centers are updated by recomputing the centroids of all patterns having the same cluster label at the end of the pass. The seed points are K patterns chosen at random, where K is specified by the user as the number of clusters desired. Our implementation of FORGY allows the user to specify a heuristic that creates additional clusters (Dubes and Jain, 1976). After the square-error has converged for a fixed K, a new cluster is created when a pattern is found that is sufficiently far removed from the existing cluster centers. The average distance between pattern $\mathbf{x}_i$ and the K cluster centers is given by

$$\bar{d}_i = (1/K) \sum_{k=1}^{K} d(\mathbf{x}_i, \mathbf{m}^{(k)})$$

A new cluster is created centered at pattern $\mathbf{x}_i$ if

$$|d(\mathbf{x}_i, \mathbf{m}^{(q)}) - \bar{d}_i| \leq \bar{d}_i T_1$$

where the qth cluster center is the cluster center closest to pattern $\mathbf{x}_i$ and T_1 is a user-specified parameter, $0 < T_1 < 1$. The left side of the above inequality is roughly $\bar{d}_i$ for patterns close to an existing cluster and is small for patterns far removed from all existing clusters. The larger T_1, the more new clusters are created. FORGY also detects outliers. If the number of patterns in any cluster falls below another user-specified parameter, T_2, then all the patterns belonging to that cluster are considered as outliers and are ignored.

To run FORGY, a user specifies threshold T_1 for creating new clusters, threshold T_2 for deleting clusters, and the maximum allowable number of iterations. The most crucial parameter is T_1 and a few runs of the program with different values of T_1 and T_2 may be necessary to obtain a reasonable grouping of the data. FORGY should be run several times with different starting configurations.

The output of FORGY provides cluster labels for all patterns and several statistics such as the square error for each cluster, a table of distances between cluster centers, and the ratio of within-cluster to between-cluster distance. A table showing the number of patterns in each category is also printed when a priori category labels for all patterns are available. Since FORGY is based on square-error, it also generates some statistics that can be applied to analyze the separations among patterns. Since FORGY tries to group the patterns, an analysis of variance cannot be applied to these statistics. This point is treated at length in Chapter 4. Example 3.9 demonstrates the expected output of FORGY. The statistics are explained below.

The square-error for a clustering, E_K^2, can be decomposed into a feature-by-feature sum as shown below.

$$E_K^2 = \sum_{j=1}^{d} f_j^2$$

The contribution of feature j to the square-error, f_j^2, is the sum of the squared differences between each pattern and its cluster center using only feature j.

$$f_j^2 = \sum_{k=1}^{K} \sum_{i=1}^{n_k} [x_{ij}^{(k)} - m_j^{(k)}]^2$$

In terms of the standard decomposition (Appendix D), f_j^2 is the "within-cluster" variation in feature j. The "between-cluster" component, b_j^2, can be expressed in terms of the cluster centers and m_j, the jth coordinate of the centroid of all patterns.

$$b_j^2 = \left[\sum_{k=1}^{K} n_k (m_j^{(k)})^2\right] - n(m_j)^2$$

$$m_j = \frac{1}{n} \sum_{k=1}^{K} \sum_{i=1}^{n_k} x_{ij}^{(k)} = \frac{1}{n} \sum_{k=1}^{K} n_k m_j^{(k)}$$

Discriminant analysis (Appendix D) shows that the total square-error contribution from feature j, a_j^2, can be written as

$$a_j^2 = f_j^2 + b_j^2$$

The F-ratio is defined as

$$F\text{-ratio} = \frac{b_j^2/(K-1)}{f_j^2/(n-K)}$$

The name "F-ratio" comes from analysis of variance (Appendix F). When the samples are independent and come from Gaussian distributions and when the variances of all groups are the same and when the group labels are assigned a priori, the F-ratio has an F distribution with $K - 1$ and $n - K$ degrees of freedom. Large values of the F-ratio, when measured on the scale of an F distribution, indicate a grouping in which the separation among clusters is significantly large with respect to the separations among patterns in individual clusters. Unfortunately, this distribution cannot be applied to determine whether a particular feature contributes significantly to the clustering because the cluster labels are assigned *after* looking at the data. For example, a clustering algorithm labels the patterns so as to separate the clusters maximally. The null distribution of the F-ratio printed out by the program is not the standard distribution published in textbooks.

A few other quantities defined in FORGY's output need clarification. For cluster number k, the "squared error" is e_k^2, "S.E./(N(k) − 1)" is $e_k^2/(n_k - 1)$, and "CLAVGD(k)" is $(e_k^2/n_k)^{1/2}$. Other terms are self-explanatory. Note that "distance" means "squared Euclidean distance."

Example 3.9

FORGY was applied to data sets DATA1 and DATA2 (Example 2.6). DATA1 consists of 100 patterns arranged into four distinct clusters in a four-dimensional unit hypercube. Patterns are arranged by category. The first 24 patterns are from category 1, the next 35 from category 2, the next 21 from category 3, and the last 20 from category 4. DATA2 consists of 100 patterns uniformly generated in a six-dimensional unit hypercube. The following parameter settings were tried for both sets of data:

$$K = 2, 4, 6 \qquad T_1 = 0.5, \quad T_2 = 1 \qquad \text{maximum number of iterations} = 20$$

The objective here was not to finely tune the parameters to get the best clustering, but to see if reasonable clusterings can be obtained. FORGY converged in fewer than 10 iterations on both data sets. The two-cluster solution took about 12 seconds of CPU time on a Harris 500 superminicomputer. The execution time increased to 35 seconds when the number of clusters was increased to six.

We first summarize the results for DATA1, where the true cluster numbers generated by computer are treated as category information. Part of the output for the two-cluster solution is shown in Figure 3.21. The output shows that categories 1 and 2 are grouped into one cluster and categories 3 and 4 in the other cluster. This grouping is surprising in light of the two-dimensional representation in Figure 2.10 and might be explained by the observation that the clusters are hyperellipsoidal with nearly identical covariance structure. The F-ratio for feature 4 is largest, so the patterns cluster better in feature 4 than in the other 3 features. We cannot say whether the clustering in feature 4 is significantly best. Only patterns 13, 16, and 89 fail to cluster properly. The main output of the program is the locations of the two cluster centers. The total square error for the two-cluster solution is 17.406 and seems evenly divided between the two clusters.

Only the most useful information in FORGY output is displayed for the four-cluster (Figure 3.22) and six-cluster (Figure 3.23) solutions. The total square-error reduces to 6.8396 for the four-cluster solution and to 3.4096 for the six-cluster solution. Categories 1 and 4 are confused in the four-cluster solution, whereas they are separated in the two-cluster solution. ~~Feature 4 is not as important to the four-cluster solution as it was to the two-cluster solution according to the F-ratio.~~ The six-cluster solution separates patterns

Feature 4 is more important to the two-cluster solution than feature 2 according to the F-ratio.

```
Results of FORGY algorithm after 6 iterations
  0 Patterns were removed.
  2 Clusters were obtained.
Cluster No. for each pattern—LABEL array
  1  1  1  1  1  1  1  1  1  1    1  1  2  1  1  2  1  1  1  1
  1  1  1  1  1  1  1  1  1  1    1  1  1  1  1  1  1  1  1  1
  1  1  1  1  1  1  1  1  1  1    1  1  1  1  1  1  1  1  1  2
  2  2  2  2  2  2  2  2  2  2    2  2  2  2  2  2  2  2  2  2
  2  2  2  2  2  2  2  2  1  2    2  2  2  2  2  2  2  2  2  2
Variation about cluster means
Feature    Squared error    Between clusters    F-Ratio
    1          2.5197            18.456          717.80
    2          7.3108            41.203          552.32
    3          5.1391            37.489          714.89
    4          2.4365            35.665         1434.5
Squared error per cluster—K is the cluster No.
       K      N(K)    Squared error    S.E./(N(K)-1)    CLAVGD(K)
       1       58        8.1296           0.14262         0.37439
       2       42        9.2766           0.22626         0.46997
   TOTAL      100       17.406
Squared error divided by (No. of patterns in clustering—No. of clusters) =
  0.17761
Average error or root of (Squared error divided by NPAT-NDELE) = 0.41721

Distances between cluster centers
          1          2
  1    0.0000     0.51911
  2    0.51911    0.0000
Average distance between each cluster center and the other centers
  0.51911    0.51911
Average of the Avg. distances listed directly above      =0.51911
Minimum of the distances between each cluster center and the other cluster centers
  0.51911    0.51911
For each cluster, the ratio of the average distance between that cluster center and
all others to the average within-cluster distance
  1.3866     1.1046
For each cluster, the ratio of the minimum of the distances between that
cluster center and all others to the average within-cluster distance
  1.3866     1.1046
Cluster membership according to category:
Rows are clusters and columns are categories
        1     2     3     4
  1    22    35     0     1
  2     2     0    21    19
CLUSTER CENTERS
  1    0.509    0.727    0.736    0.771
  2    0.301    0.520    0.402    0.204
```

Figure 3.21 FORGY clustering on DATA1: two clusters.

```
Results of  FORGY  algorithm after  6  iterations

  0 Patterns were removed.
  4 Clusters were obtained.


Cluster No. for each pattern - LABEL array

  2  2  2  2  2  2  2  2  2  2          2  2  2  2  2  2  2  2  2  2
  2  2  2  2  3  3  3  3  3  3          3  3  3  3  3  3  3  3  3  3
  3  3  3  3  3  3  3  3  3  3          3  3  3  3  3  3  3  3  3  1
  4  1  4  1  1  4  1  4  1  1          4  1  1  4  4  1  1  1  1  4
  2  2  2  2  2  2  2  2  2  2          2  2  2  2  2  2  2  2  2  2


Squared error per cluster - K is the cluster No.

      K    N(K)     Squared error     S.E./(N(K)-1)     CLAVGD(K)
      1      13      0.20735           0.17279E-01       0.12629
      2      44      5.1956            0.12083           0.34363
      3      35      1.2852            0.37799E-01       0.19162
      4       8      0.15144           0.21634E-01       0.13759
  TOTAL     100      6.8396

Squared error divided by (No  of patterns in  clustering - No. of clusters) =
    0.71246E-01


Distances between cluster centers

           1              2             3              4
  1    0.0000         0.76497        1.0242        0.53347E-01
  2   0.76497         0.0000        0.52729        0.73402
  3    1.0242         0.52729        0.0000         1.2203
  4   0.53347E-01 0.73402            1.2203         0.0000


Cluster membership according to category:
Rows are clusters and columns are categories
          1     2     3     4
  1       0     0    13     0
  2      24     0     0    20
  3       0    35     0     0
  4       0     0     8     0
```

Figure 3.22 FORGY clustering on DATA1: four clusters.

into clusters according to category except for patterns 13 and 16. The origins of these two patterns should probably be inspected. Categories 2 and 4 are split into two clusters, so a bimodal structure might be appropriate for describing these categories.

How many clusters are appropriate for these data? One of the heuristics for deciding the number of clusters is to look for a ''knee'' in the plot of total square-error versus the number of clusters (see Section 4.4.2). This heuristic suggests that DATA1 has four clusters. The problem of fixing the ''correct'' number of clusters is one of the most fundamental and unsolved problems in cluster analysis.

One useful statistic printed by FORGY is the ratio of the distances between a cluster center and all others to the average within-cluster distance. Large values of this statistic suggest that the clusters are compact and well separated. These values are substantially higher for both the four-cluster and the six-cluster solution than for the two-cluster solution, although these statistics naturally increase with the number of clusters. In the four-cluster solution, only cluster 2, which is a mixture of categories 1 and 4, has a low value for this statistic.

Ag applie [illegible]
Figs. 3.22-3

```
Results of  FORGY  algorithm after  9  iterations

   0 Patterns were removed.
   6 Clusters were obtained.


Cluster No. for each pattern - LABEL array

   6  6  6  6  6  6  6  6  6  6        6  6  4  6  6  4  6  6  6  6
   6  6  6  6  2  5  2  5  5  5        5  5  2  5  5  5  5  2  2  2
   5  5  2  5  5  2  5  5  2  5        5  5  5  5  2  2  2  5  5  3
   3  3  3  3  3  3  3  3  3  3        3  3  3  3  3  3  3  3  3  3
   1  1  1  1  4  1  1  1  1  4        1  4  4  1  1  4  1  1  1  1


Squared error per cluster - K is the cluster No.

       K   N(K)    Squared error    S.E./(N(K)-1)     CLAVGD(K)
       1     15       0.43799        0.31285E-01       0.17088
       2     12       0.25361        0.23056E-01       0.14538
       3     21       0.62299        0.31149E-01       0.17224
       4      7       0.30889        0.51482E-01       0.21006
       5     23       0.71969        0.32713E-01       0.17689
       6     22        1.0664        0.50783E-01       0.22017
   TOTAL    100        3.4096

Squared error divided by (No  of patterns in  clustering - No. of clusters) =
    0.36272E-01


Distances between cluster centers

           1            2            3            4            5            6
  1     0.0000      0.88037      0.80797      0.85551E-01  0.84843      0.32988
  2    0.88037       0.0000      0.97198      0.74127      0.39550E-01  0.39932
  3    0.80797      0.97198       0.0000      0.61857       1.1596      0.88723
  4    0.85551E-01  0.74127      0.61857       0.0000      0.70367      0.20117
  5    0.84843      0.39550E-01   1.1596      0.70367       0.0000      0.41891
  6    0.32988      0.39932      0.88723      0.20117      0.41891       0.0000


Cluster membership according to category:
Rows are clusters and columns are categories

          1     2     3     4
  1       0     0     0    15
  2       0    12     0     0
  3       0     0    21     0
  4       2     0     0     5
  5       0    23     0     0
  6      22     0     0     0
```

Figure 3.23 FORGY clustering on DATA1: six clusters.

The patterns in DATA2 have no category information, so FORGY cannot construct cluster by category tables to judge the performance of the clustering program. We can contrast the output of FORGY for these random data with the clustered data in DATA1 but must be aware that the numbers of features is not the same. Figures 3.24 to 3.26 show that the square-error for DATA2 is substantially higher than for DATA1, and the total square-error for DATA2 does not fall as sharply as for DATA1 when the number of clusters is increased. Both sets of data have the same number of patterns, but DATA2 is in six dimensions. Again, the complete output for FORGY on DATA2 is shown only for the two-cluster solution (Figure 3.24).

Another noticeable difference between clustering for the two data sets appears in

```
Results of FORGY algorithm after 6 iterations
  0 Patterns were removed.
  2 Clusters were obtained.
Cluster No. for each pattern—LABEL array
  1  1  2  2  2  1  2  1  1  1     1  1  2  2  1  1  2  1  2  2
  2  1  1  2  1  1  1  1  2  1     2  1  1  2  1  1  1  1  1  1
  1  1  1  1  2  2  2  1  1  1     1  1  1  1  2  1  1  2  1  1
  2  2  1  2  1  1  1  1  1  1     2  1  2  2  2  1  2  1  2  2
  1  1  2  1  1  1  1  2  1  2     1  1  1  2  1  2  1  2  1  2
Variation about cluster means
Feature     Squared error     Between clusters     F-Ratio
   1           7.5990              22.963           296.14
   2           7.4116              22.134           292.67
   3           5.9815              28.352           464.52
   4           7.1100              26.380           363.61
   5           8.4855              27.928           322.54
   6           6.4006              23.071           353.24
Squared error per cluster—K is the cluster No.
      K      N(K)     Squared error     S.E./(N(K)-1)     CLAVGD(K)
      1       64         27.509            0.43665          0.65561
      2       36         15.479            0.44226          0.65572
  TOTAL      100         42.988
Squared error divided by (No. of patterns in clustering—No. of clusters) =
  0.43865

Average error or root of (Squared error divided by NPAT-NDELE) = 0.65565

Distances between cluster centers
           1          2
  1     0.0000     0.26452
  2     0.26452    0.0000
Average distance between each cluster center and the other centers
  0.26452    0.26452
Average of the Avg. distances listed directly above     =0.26452
Minimum of the distances between each cluster center and the other cluster centers
  0.26452    0.26452
For each cluster, the ratio of the average distance between that cluster center and
all others to the average within-cluster distance
  0.40348    0.40341
For each cluster, the ratio of the minimum of the distances between that
cluster center and all others to the average within-cluster distance
  0.40348    0.40341
CLUSTER CENTERS
  1     0.368     0.525     0.405     0.564     0.558
        0.551
  2     0.640     0.369     0.714     0.426     0.487
        0.336
```

Figure 3.24 FORGY clustering on DATA2: two clusters.

```
Results of  FORGY  algorithm after  10  iterations

  0 Patterns were removed.
  4 Clusters were obtained.


Cluster No. for each pattern - LABEL array

  1  4  4  1  1  2  1  4  3  4        3  1  4  1  2  4  1  2  3  1
  3  4  4  1  4  2  1  4  4  2        1  2  3  4  4  3  2  1  3  2
  2  4  2  2  4  1  1  2  4  2        2  4  2  3  1  1  1  1  2  4
  1  3  2  4  3  1  4  2  1  3        1  2  3  4  3  1  1  4  4  3
  4  4  1  3  1  2  2  1  2  1        1  4  3  3  3  1  3  1  4  1


Squared error per cluster - K is the cluster No.

      K   N(K)     Squared error    S.E./(N(K)-1)     CLAVGD(K)
      1     32       12.332           0.39782           0.62079
      2     22       6.1898           0.29475           0.53043
      3     19       5.1512           0.28618           0.52069
      4     27       9.6708           0.37195           0.59848
  TOTAL    100       33.344

Squared error divided by (No  of patterns in  clustering - No. of clusters) =
    0.34733


Distances between cluster centers

          1            2            3            4
  1    0.0000      0.44283      0.39219      0.42436
  2   0.44283       0.0000      0.30128      0.31992
  3   0.39219      0.30128       0.0000      0.66478
  4   0.42436      0.31992      0.66478       0.0000
```

Figure 3.25 FORGY clustering on DATA2: four clusters.

the ratio of the average distance between a cluster center and all others to the average within-cluster distance. This statistic has substantially lower values for random data than for the clustered data. Clustering tendency (Section 4.6) deals with the issue of whether a given data set is random.

Example 3.10

Figure 3.27(a) demonstrates a difficulty with square-error clustering. It shows the two-cluster solution generated by FORGY for a set of patterns in the plane which clearly contains two well-separated groups. Unfortunately, the partition boundary between the two clusters does not lie in the sparse region as expected but cuts one of the two "natural" clusters in half. Similarly, the cigar-shaped data in Figure 3.27(b) are not clustered correctly by FORGY. These examples demonstrate that the square-error criterion, which seeks compact hyperellipsoidal clusters, can produce misleading results when the data do not occur in compact, hyperellipsoidal boundaries.

The second square-error clustering program to be examined is called CLUSTER (Dubes and Jain, 1976). It has the same objective as FORGY but generates a nonhierarchical sequence of clusterings rather than a single clustering. This program utilizes both a *K*-means pass and a forcing pass (Friedman and Rubin, 1967). CLUSTER attempts to find the "best" clusterings containing 1, 2, . . . ,

```
Results of  FORGY  algorithm after  6  iterations

  0 Patterns were removed.
  6 Clusters were obtained.


Cluster No. for each pattern - LABEL array

  1  3  1  1  1  4  1  3  5  6        1  6  1  5  4  3  6  4  5  1
  5  1  3  6  3  4  1  3  3  4        6  2  5  2  3  5  4  1  5  4
  4  2  4  4  3  1  6  2  3  2        1  3  2  5  1  6  1  4  4  3
  5  6  2  3  5  1  6  2  1  5        1  4  5  3  4  6  4  3  3  5
  3  3  4  5  6  4  4  6  4  4        6  3  5  5  2  6  5  4  6  6


Squared error per cluster - K is the cluster No.

         K    N(K)     Squared error    S.E./(N(K)-1)     CLAVGD(K)
         1     18        5.7175           0.33633          0.56360
         2      9        1.6167           0.20209          0.42383
         3     19        6.0932           0.33851          0.56630
         4     21        6.9902           0.34951          0.57694
         5     17        4.5053           0.28158          0.51480
         6     16        3.8947           0.25965          0.49338
     TOTAL    100        28.818

Squared error divided by (No  of patterns in  clustering - No. of clusters) =
    0.30657


Distances between cluster centers

          1           2           3           4           5           6
  1    0.0000     0.65279     0.41074     0.44736     0.46324     0.49531
  2   0.65279      0.0000     0.32856     0.39610     0.34615     0.51358
  3   0.41074     0.32856      0.0000     0.46140     0.83177     0.61105
  4   0.44736     0.39610     0.46140      0.0000     0.36814     0.39783
  5   0.46324     0.34615     0.83177     0.36814      0.0000     0.60731
  6   0.49531     0.51358     0.61105     0.39783     0.60731      0.0000
```

Figure 3.26 FORGY clustering on DATA2: six clusters.

K clusters and prints a history of the *K* clusterings achieved, one for each number of clusters.

CLUSTER involves two phases which are repeated until a pass through both phases does not decrease the square-error. Phase 1 creates a sequence of clusterings containing 2, 3, ... , *K* clusters, where *K* is specified by the user. The initial two cluster centers are the centroid of the patterns and the pattern farthest removed from the centroid, not counting the outliers. Given a clustering with *k* clusters, the pattern farthest removed from the existing clustering is identified as the $(k + 1)$st cluster center. The *K*-means pass is repeated until no patterns change clusters or until a maximum number of iterations have been completed.

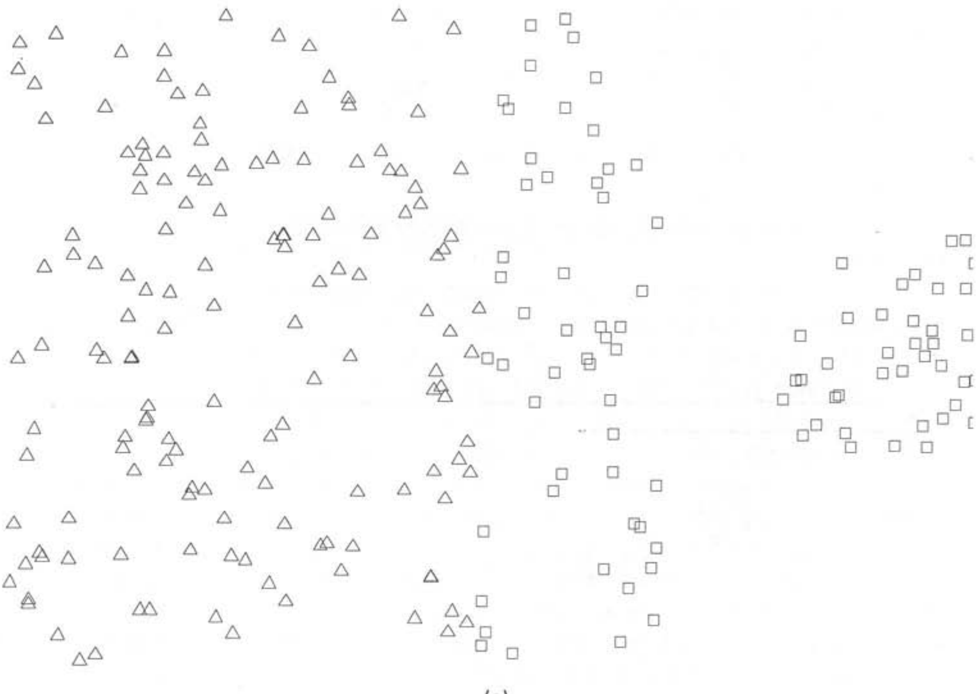
The first pass through phase 1 gives a set of *K* clusterings, each containing a different number of clusters. Phase 2 then creates another set of clusterings by merging existing clusters two at a time to see if a better clustering can be achieved (forcing pass). After each pass through phases 1 and 2, the square-errors of the clusterings are compared with those of the clusterings (having the same number of clusters) that existed before that pass. If any of the square-errors are smaller than before, another pass through phases 1 and 2 is initiated. This continues until square-error cannot be decreased.

One of the main advantages of CLUSTER over other clustering programs is that the user need not specify parameters. Only the maximum number of clusters desired is required. The clustering statistic computed in CLUSTER is a purely heuristic number.

$$(1/K) \sum_{k=1}^{K} [(n_k/e_k^2)^{1/2} (n - n_k)^{-1} \sum_{\substack{r=1 \\ k \neq r}}^{K} n_r d^2(\mathbf{m}^{(k)}, \mathbf{m}^{(r)})]$$

That is, the ratio of the "average" distance from cluster k to all other clusters is divided by the average within-cluster distance for cluster k. This quantity is averaged over all clusters. Intuition dictates that the larger this number, the better the clustering. But how large is large? How does this statistic depend on problem parameters? These questions have not been answered. We now present the results of CLUSTER on DATA1 and DATA2 and compare these results with those given by program FORGY.

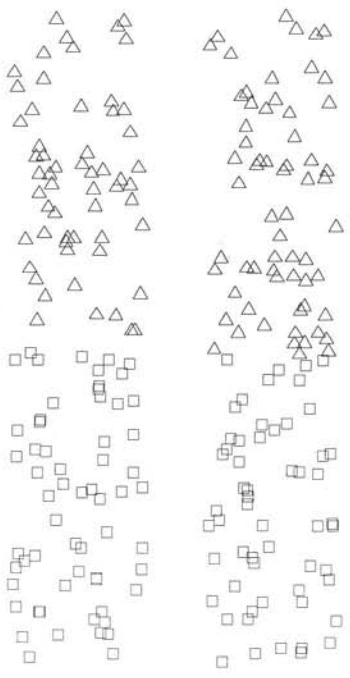
```
No. of patterns = 240(200,40)  No. of ini. clu. centers = 2
Min clus. size  = 1            Max  no of iterations    = 5
Max no of inner loop= 10
Threshold for forming new cluster= 0.4
```



(a)

Figure 3.27 Inadequacies of square-error clustering.

```
No. of patterns = 240(120,120) No. of ini. clu. centers = 2
Min clus. size  = 1            Max  no of iterations    = 5
Max no of inner loop= 10
Threshold for forming new cluster= 0.4
```



(b)

Figure 3.27 (*continued*)

Example 3.11

The partitions obtained from CLUSTER for data sets DATA1 and DATA2 are given in Figures 3.28 and 3.29. They appear to be better than those generated by FORGY. For example, the four-cluster solution from CLUSTER for DATA1 has a total square-error of 4.0677 (Figure 3.28) compared to 6.8396 by FORGY. The partition from CLUSTER uniquely associates a cluster with each category with the exception of a single pattern from category 1. FORGY's four-cluster solution merged categories 1 and 4 in the same cluster. The performance of CLUSTER on the random data of DATA2 is comparable to that of FORGY.

In summary, clustering programs that minimize square-error are very practical. They try to define clusters that are hyperellipsoidal in shape. The square-error criterion is equivalent to several other criteria involving the scatter matrices used in discriminant analysis. The numerous square-error programs available differ both in computational details and in the approach taken to minimize the square error. Square-error clustering methods do exhibit inadequacies as when the Euclidean metric is used to measure distance but the features are not on comparable scales.

```
Program CLUSTER will group 100 patterns using
 4 features. There are 4 categories.
Mean vector for raw data:
     0.4214      0.6402      0.5956      0.5330
Standard deviations for raw data:
     0.1890      0.2890      0.2805      0.3204
```

```
 Begin output for clustering number 1
Clusters by category:
CLUSTER     1     2     3     4
   1       24    35    21    20
Squared error for cluster number J
   J  N(J)    SQUARED ERROR     S. E./(N(J)-1)     CLAVGD(J)
   1  100     30.052            0.30355            0.54819
Squared error divided by (No. patterns - 1) = 0.30355
```

```
 Begin output for clustering number 2
 2 Clusters: Cluster Membership Table
 1 1 1 1 1 1 1 1 1 1    1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1    1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1    1 1 1 1 1 1 1 1 1 2
 2 2 2 2 2 2 2 2 2 2    2 2 2 2 2 2 2 2 2 2
 1 1 1 1 1 1 1 1 1 1    1 1 1 1 1 1 1 1 1 1
Clusters by category:
CLUSTER     1     2     3     4
   1       24    35     0    20
   2        0     0    21     0
```

```
Squared error for cluster number J
   J  N(J)    SQUARED ERROR     S. E./(N(J)-1)     CLAVGD(J)
   1   79     16.760            0.21487            0.46059
   2   21      0.62299          0.31149E-01        0.17224
Total  100    17.383
Sq. error divided by (No. patterns - No. of clusters) = 0.17737
Table of distances between cluster centers
CLUS      1        2
  1      0.00     0.874
  2      0.874    0.00


The clustering statistic: 3.0112
```

```
 Begin output for clustering number 3
 3 Clusters: Cluster Membership Table
 3 3 3 3 3 3 3 3 3 3    3 3 3 3 3 3 3 3 3 3
 3 3 3 3 1 1 1 1 1 1    1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1    1 1 1 1 1 1 1 1 1 2
 2 2 2 2 2 2 2 2 2 2    2 2 2 2 2 2 2 2 2 2
 3 3 3 3 3 3 3 3 3 3    3 3 3 3 3 3 3 3 3 3
Clusters by category:
CLUSTER     1     2     3     4
   1        0    35     0     0
   2        0     0    21     0
   3       24     0     0    20
```

Figure 3.28 Results of CLUSTER on DATA1.

```
Squared error for cluster number J
     J   N(J)    SQUARED ERROR       S. E./(N(J)-1)    CLAVGD(J)
     1    35     1.2852              0.37799E-01       0.19162
     2    21     0.62299             0.31149E-01       0.17224
     3    44     5.1956              0.12083           0.34363
Total    100     7.1038
Sq. error divided by (No. patterns - No. of clusters) = 0.73235E-01
Table of distances between cluster centers
CLUS        1        2        3
   1     0.00     1.04     0.726
   2     1.04     0.00     0.861
   3     0.726    0.861    0.00
The clustering statistic: 1.5205
```

```
 Begin output for clustering number 4
 4 Clusters: Cluster Membership Table
 4  4  4  4  4  4  4  4  4  4    4  4  4  4  4  3  4  4  4  4
 4  4  4  4  1  1  1  1  1  1    1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1    1  1  1  1  1  1  1  1  1  2
 2  2  2  2  2  2  2  2  2  2    2  2  2  2  2  2  2  2  2  2
 3  3  3  3  3  3  3  3  3  3    3  3  3  3  3  3  3  3  3  3
Clusters by category:
CLUSTER       1      2      3      4
   1          0     35      0      0
   2          0      0     21      0
   3          1      0      0     20
   4         23      0      0      0
Squared error for cluster number J
     J   N(J)    SQUARED ERROR       S. E./(N(J)-1)    CLAVGD(J)
     1    35     1.2852              0.37799E-01       0.19162
     2    21     0.62299             0.31149E-01       0.17224
     3    21     0.99892             0.49946E-01       0.21810
     4    23     1.1606              0.52756E-01       0.22464
Total    100     4.0677
Sq. error divided by (No. patterns - No. of clusters) = 0.42372E-01

Table of distances between cluster centers
CLUS        1        2        3        4
   1     0.00     1.04     0.900    0.633
   2     1.04     0.00     0.867    0.929
   3     0.900    0.867    0.00     0.526
   4     0.633    0.929    0.526    0.00

The clustering statistic: 1.0908
```

```
 Begin output for clustering number 5
 5 Clusters: Cluster Membership Table
 4  4  4  4  4  4  4  4  4  4    4  4  5  4  4  5  4  4  4  4
 4  4  4  4  1  1  1  1  1  1    1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1    1  1  1  1  1  1  1  1  1  2
 2  2  2  2  2  2  2  2  2  2    2  2  2  2  2  2  2  2  2  2
 3  3  3  3  5  3  3  3  3  5    3  5  5  3  3  5  3  3  3  3
Clusters by category:
CLUSTER       1      2      3      4
   1          0     35      0      0
   2          0      0     21      0
   3          0      0      0     15
   4         22      0      0      0
   5          2      0      0      5
Squared error for cluster number J
     J   N(J)    SQUARED ERROR       S. E./(N(J)-1)    CLAVGD(J)
     1    35     1.2852              0.37799E-01       0.19162
     2    21     0.62299             0.31149E-01       0.17224
     3    15     0.43799             0.31285E-01       0.17088
     4    22     1.0664              0.50783E-01       0.22017
     5     7     0.30889             0.51482E-01       0.21006
Total    100     3.7215
Sq. error divided by (No. patterns - No. of clusters) = 0.39174E-01
```

Figure 3.28 ***(continued)***

```
Table of distances between cluster centers
CLUS      1       2       3       4       5
 1      0.00    1.04    0.922   0.635   0.841
 2      1.04    0.00    0.899   0.942   0.786
 3      0.922   0.899   0.00    0.574   0.292
 4      0.635   0.942   0.574   0.00    0.449
 5      0.841   0.786   0.292   0.449   0.00
The clustering statistic: 0.78831
```

```
 Begin output for clustering number 6
 6 Clusters: Cluster Membership Table
 6 4 6 6 4 4 4 6 4 4   4 4 5 6 4 5 6 6 6 4
 6 4 4 6 1 1 1 1 1 1   1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1   1 1 1 1 1 1 1 1 1 2
 2 2 2 2 2 2 2 2 2 2   2 2 2 2 2 2 2 2 2 2
 3 3 3 3 5 3 3 3 3 5   3 5 5 3 3 5 3 3 3 3
Clusters by category:
CLUSTER    1    2    3    4
  1        0   35    0    0
  2        0    0   21    0
  3        0    0    0   15
  4       12    0    0    0
  5        2    0    0    5
  6       10    0    0    0
```

```
Squared error for cluster number J
   J  N(J)   SQUARED ERROR    S. E./(N(J)-1)    CLAVGD(J)
   1   35    1.2852           0.37799E-01       0.19162
   2   21    0.62299          0.31149E-01       0.17224
   3   15    0.43799          0.31285E-01       0.17088
   4   12    0.36424          0.33113E-01       0.17422
   5    7    0.30889          0.51482E-01       0.21006
   6   10    0.33110          0.36789E-01       0.18196
Total  100   3.3504
Sq. error divided by (No. patterns - No. of clusters) = 0.35642E-01
Table of distances between cluster centers
CLUS      1       2       3       4       5       6
 1      0.00    1.04    0.922   0.674   0.841   0.615
 2      1.04    0.00    0.899   0.961   0.786   0.939
 3      0.922   0.899   0.00    0.606   0.292   0.568
 4      0.674   0.961   0.606   0.00    0.431   0.261
 5      0.841   0.786   0.292   0.431   0.00    0.506
 6      0.615   0.939   0.568   0.261   0.506   0.000
The clustering statistic: 0.68259
```

Figure 3.28 ***(continued)***

```
Program CLUSTER will group 100 patterns using
 6 features. There are 0 categories.
Mean vector for raw data:
    0.4657     0.4692     0.5166     0.5145     0.5327
    0.4738
Standard deviations for raw data:
    0.3051     0.2824     0.2860     0.2747     0.2933
    0.2733
```

```
 Begin output for clustering number 1
Squared error for cluster number J
   J  N(J)    SQUARED ERROR      S. E./(N(J)-1)     CLAVGD(J)
   1  100     49.083             0.49578            0.70059
Squared error divided by (No. patterns - 1) = 0.49578
```

```
 Begin output for clustering number 2
 2 Clusters: Cluster Membership Table
 2 1 2 2 2 1 2 2 1 2     1 2 2 2 1 2 2 1 1 2
 1 2 1 2 1 1 2 2 1 1     2 1 1 2 2 1 1 2 1 1
 1 1 2 1 2 2 2 1 2 1     1 2 1 1 2 2 2 2 2 1
 2 1 1 2 1 2 2 1 2 1     2 1 1 2 1 2 2 2 2 1
 2 1 2 1 2 1 1 2 1 2     2 1 1 1 1 2 1 2 2 2
Squared error for cluster number J
   J  N(J)    SQUARED ERROR      S. E./(N(J)-1)     CLAVGD(J)
   1   47     17.783             0.38658            0.61510
   2   53     23.387             0.44975            0.66428
Total  100    41.170
Sq. error divided by (No. patterns - No. of clusters) = 0.42010
Table of distances between cluster centers
CLUS        1          2
  1       0.00       0.564
  2       0.564      0.00
The clustering statistic: 0.65331
```

```
 Begin output for clustering number 3
 3 Clusters: Cluster Membership Table
 3 1 2 2 2 1 3 2 1 3     1 3 2 3 1 2 3 1 1 3
 1 2 1 3 1 1 3 2 2 1     3 1 1 2 2 1 1 3 1 1
 1 1 3 1 2 2 3 1 2 1     1 2 1 1 3 3 3 2 3 2
 3 3 1 2 1 2 3 1 3 1     2 3 1 2 1 3 3 2 2 1
 2 1 3 1 3 1 1 3 1 3     3 1 1 1 1 3 1 3 3 3
Squared error for cluster number J
   J  N(J)    SQUARED ERROR      S. E./(N(J)-1)     CLAVGD(J)
   1   43     15.884             0.37820            0.60779
   2   24      8.3599            0.36348            0.59020
   3   33     12.054             0.37668            0.60438
Total  100    36.298
Sq. error divided by (No. patterns - No. of clusters) = 0.37421
Table of distances between cluster centers
CLUS        1          2          3
  1       0.00       0.669      0.606
  2       0.669      0.00       0.610
  3       0.606      0.610      0.00
The clustering statistic: 0.49489
```

```
 Begin output for clustering number 4
 4 Clusters: Cluster Membership Table
 4 1 2 2 2 1 3 2 4 3     4 3 2 3 1 2 3 1 1 4
 1 2 1 3 1 4 4 2 2 1     3 1 4 3 2 4 4 3 4 1
 1 1 4 1 2 4 3 1 2 1     4 2 1 1 3 3 3 2 3 2
 4 3 1 2 1 4 3 1 4 4     4 1 4 2 1 3 4 2 2 1
 2 1 2 4 3 1 1 3 1 3     3 1 4 1 1 3 4 3 3 3
```

Figure 3.29 Results of CLUSTER on DATA2.

```
Squared error for cluster number J
   J  N(J)   SQUARED ERROR     S. E./(N(J)-1)    CLAVGD(J)
   1   31    10.590            0.35301           0.58449
   2   21     6.3966           0.31983           0.55190
   3   25     7.9114           0.32964           0.56254
   4   23     8.1612           0.37097           0.59568
Total  100   33.060
Sq. error divided by (No. patterns - No. of clusters) = 0.34437
Table of distances between cluster centers
CLUS      1        2        3        4
 1      0.00     0.659    0.659    0.583
 2      0.659    0.00     0.654    0.764
 3      0.659    0.654    0.00     0.638
 4      0.583    0.764    0.638    0.00


The clustering statistic: 0.36890
```

```
 Begin output for clustering number 5
 5 Clusters: Cluster Membership Table
 4  5  2  2  2  1  3  2  4  3    4  3  2  4  1  5  3  1  1  4
 1  2  5  3  5  4  4  2  2  1    3  1  4  3  2  4  4  3  4  1
 1  5  2  1  2  4  3  1  5  1    4  5  1  4  3  3  3  2  3  5
 4  3  1  2  4  4  3  1  4  4    4  1  4  2  1  3  4  2  2  1
 5  5  2  4  3  1  1  3  1  2    3  5  4  1  1  3  4  2  3  3
Squared error for cluster number J
   J  N(J)   SQUARED ERROR     S. E./(N(J)-1)    CLAVGD(J)
   1   23     6.6293           0.30133           0.53687
   2   19     5.9713           0.33174           0.56061
   3   22     6.2765           0.29888           0.53413
   4   25     8.5578           0.35658           0.58507
   5   11     2.4206           0.24206           0.46910
Total  100   29.856
Sq. error divided by (No. patterns - No. of clusters) = 0.31427
```

```
Table of distances between cluster centers
CLUS      1        2        3        4        5
 1      0.00     0.718    0.667    0.567    0.695
 2      0.718    0.00     0.681    0.760    0.765
 3      0.667    0.681    0.00     0.654    0.796
 4      0.567    0.760    0.654    0.00     0.823
 5      0.695    0.765    0.796    0.823    0.00
The clustering statistic: 0.38778
```

```
 Begin output for clustering number 6
 6 Clusters: Cluster Membership Table
 6  5  2  2  2  3  3  2  1  3    4  3  2  4  6  5  3  1  1  4
 1  2  5  3  5  6  4  2  2  1    3  1  4  1  2  4  6  4  4  6
 6  5  6  3  2  4  4  1  3  5    6  5  1  1  4  3  3  2  6  5
 4  3  5  2  1  4  3  1  4  4    4  6  1  2  1  3  4  2  2  1
 5  5  2  4  3  6  6  3  1  2    3  5  4  1  1  4  4  2  3  3
Squared error for cluster number J
   J  N(J)   SQUARED ERROR     S. E./(N(J)-1)    CLAVGD(J)
   1   18     4.3726           0.25721           0.49287
   2   18     5.2696           0.30998           0.54107
   3   19     5.1522           0.28263           0.52074
   4   21     6.7206           0.33603           0.56571
   5   12     2.3807           0.21643           0.44541
   6   12     3.0590           0.27809           0.50489
Total  100   26.955
Sq. error divided by (No. patterns - No. of clusters) = 0.28675
Table of distances between cluster centers
CLUS      1        2        3        4        5        6
 1      0.00     0.759    0.742    0.612    0.736    0.717
 2      0.759    0.00     0.736    0.780    0.816    0.697
 3      0.742    0.736    0.00     0.647    0.748    0.736
 4      0.612    0.780    0.647    0.00     0.874    0.676
 5      0.736    0.816    0.748    0.874    0.00     0.736
 6      0.717    0.697    0.736    0.676    0.736    0.000
The clustering statistic: 0.29051
```

Figure 3.29 ***(continued)***

Other data sets that cannot be adequately clustered by square-error programs, such as CLUSTER and FORGY, are demonstrated by Zahn (1971) and Hall et al. (1973).

3.3.4 Clustering by Mixture Decomposition

A popular approach to clustering is based on the notion of a *mixture density*. Each pattern is assumed to be drawn from one of K underlying populations, or clusters. The clustering problem is to allocate each pattern to its correct population. Unlike the density estimation or the mode-seeking clustering algorithms discussed in Section 3.3.5, the form and the number of underlying population densities are assumed to be known here. The patterns are not labeled by population. If the parameters of the population densities can be estimated from the patterns, each pattern can be assigned to its appropriate cluster based on estimated probability densities. This model of clustering is identical to the problem of unsupervised learning in statistical pattern recognition (Appendix A) and has been used to estimate crop acreages from remote-sensing data (Odell and Basu, 1976).

The patterns are drawn from a population with a known number of clusters, or classes. The underlying probability density function for class ω_i is denoted $p(\mathbf{x}|\omega_i,\boldsymbol{\theta}_i)$, where $\boldsymbol{\theta}_i$ is a vector of unknown parameters for ω_i. If $P(\omega_i)$ is the a priori probability of class ω_i, or the chance that a pattern comes from ω_i, the mixture density can be written as

$$p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{i=1}^{K} p(\mathbf{x}|\omega_i, \boldsymbol{\theta}_i)P(\omega_i)$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K)$. The class-conditional densities $p(\mathbf{x}|\omega_j, \boldsymbol{\theta}_j)$ are called the *component densities*, and the a priori probabilities $P(\omega_j)$ are called the *mixing parameters*. Note that

$$\sum_{i=1}^{K} P(\omega_i) = 1$$

We would like to use the patterns to estimate the parameter vector $\boldsymbol{\theta}$ so that the mixture can be decomposed into its component clusters, assuming that the mixture is "identifiable" (Duda and Hart, 1973; Titterington et al., 1985). If one cluster is to be assigned to each category, the parameters of each component density can be estimated separately. This is estimation, not clustering.

The general formulation given above is applicable for arbitrary density functions. In the absence of prior knowledge about the shape and size of the clusters present in the data, it is a common practice to assume that the component densities are multivariate normal with different mean vectors and, perhaps, different covariance matrices (McLachlan, 1982; Symons, 1981; Sclove, 1977; Scott and Symons, 1971; Wolfe, 1970; Day, 1969). This approach to clustering is model based, but the model is Gaussian. The parameter vector $\boldsymbol{\theta}$ is usually estimated by the maximum likelihood approach, although the Bayesian approach has also been used (Binder, 1978; Symons, 1981). Duda and Hart (1973) lucidly illustrate the practical difficulties associated with obtaining the maximum likelihood estimates of $\boldsymbol{\theta}$. One difficulty is that no explicit solution for the maximum likelihood estimator of $\boldsymbol{\theta}$ exists, so an iterative estimation scheme must be employed. Starting with an initial estimate of $\boldsymbol{\theta}$, a hill-climbing or gradient-descent procedure maximizes the log-likelihood function. Problems such as the rate of convergence, singular solutions, dependence on the starting point, and local versus global maximum are inherent in this procedure. These problems are further compounded as the number of unknown parameters increases. It is commonly assumed that the K covariance matrices are equal to limit the size of the problem.

It turns out that this maximum likelihood approach of mixture decomposition is related to a well-known clustering method (Symons, 1981). If the K covariance matrices are equal, then the maximum likelihood criterion is a simple modification of Friedman and Rubin's invariant criterion of minimizing $|\mathcal{S}_W|$. When the covariance matrices or cluster shapes are different, the maximum likelihood grouping minimizes

$$\prod_{i=1}^{K} |\mathcal{S}_W^{(i)}|^{n_i}$$

Symons (1981) has compared several of these criteria on real as well as synthetic data sets. His empirical results show that the choice of the most appropriate criterion depends on the similarity of the component covariance matrices and the relative sizes of the clusters. A suboptimal but practical approach is to take the clusters generated in a square-error clustering program, such as CLUSTER (Section 3.3.3), and use the cluster centers as estimates of mean vectors and sample covariance matrices from each cluster as estimates of the covariance matrix.

3.3.5 Clustering by Density Estimation and Mode Seeking

Clusters can be viewed as regions of the pattern space in which the patterns are dense, separated by regions of low pattern density. Clusters can be identified by searching for regions of high density, called modes, in the pattern space. Each mode is associated with a cluster center and each pattern is assigned to the cluster with the closest center. The probability density estimate at a point $\mathbf{x}$ is proportional to the number of patterns, k_n, falling in a small region of volume V_n around $\mathbf{x}$ (Duda and Hart, 1973; Silverman, 1986).

$$\hat{p}_n(\mathbf{x}) = \frac{k_n/n}{V_n}$$

where n is the total number of patterns. For a fixed V_n, k_n will be large for points lying in a dense region, resulting in a large estimate $\hat{p}_n(\mathbf{x})$. The choice of V_n is critical when n is small and is governed either by the Parzen window approach or by the nearest-neighbor approach (Duda and Hart, 1973).

The volume V_n in the Parzen window approach is specified as a function of

n. In the nearest-neighbor approach, k_n is specified as a function of n. The region around each pattern is examined to capture its k_n nearest neighbors. In both approaches, convergence arguments and other heuristics suggest that V_n be inversely proportional to $\sqrt{n}$. Once more, k_n is usually taken as proportional to $\sqrt{n}$. The primary difference between these two approaches is that the window around each point in the Parzen-window approach has the same volume, whereas the window size depends on the location of the pattern in the pattern space in the nearest-neighbor approach.

The simplest way to identify modes in the data is to construct a histogram by partitioning the pattern space into a number of nonoverlapping regions or cells. Cells with relatively high frequency counts are the potential modes or cluster centers and the boundaries between clusters fall in the "valleys" of the histogram. This method has the capability of identifying unimodal clusters of any shape. However, the number of patterns must be sufficiently large (compared to the number of features) in order for the histogram to be a good estimate of the density function. Even if the sample-size requirement is met, the success of such an approach depends on two factors. First, cells of small volume will give a very "noisy" estimate of the density, whereas large cells tend to overly smooth the density estimate. Second, the procedure for locating peaks and valleys in the histogram must be performed over a neighborhood whose size is known. These factors are difficult to handle in more than a few dimensions.

The general concept of identifying modes for clustering has been proposed by a number of researchers (Torn, 1977; Wong and Liu, 1977; Kittler, 1976; Koontz et al., 1976; Eigen et al., 1974; Katz and Rholf, 1973; Gitman and Levine, 1970; Mucciardi and Gose, 1972; Sebestyen and Edie, 1966). This approach has been quite popular in the clustering of multispectral data in remote sensing, where the large-sample-size requirement is easily met and the size of the histogram cells is naturally defined because of the gray-level quantization (Narendra and Goldberg, 1977; Goldberg and Shlien, 1978; Wharton, 1983). Wharton (1983) shows that the performance of clustering based on histograms is comparable to that of K-means clustering. The surprising result of this study was that even for moderately sized data sets (100 observations per category), the histograms gave a reasonable estimate of the density function.

The memory and run-time requirements of storing and searching multidimensional histograms can be enormous. Chhikara and Register (1979) get around this problem by constructing one histogram for each feature. Histograms with only one mode are eliminated and the set of patterns is dichotomized on the feature whose histogram has the smallest number of modes. This sequential splitting procedure is repeated by recomputing histograms for individual clusters. A merging procedure is invoked whenever the clusters overlap in the pattern space. The splitting and merging stop when all the clusters have unimodal frequency distributions for every feature. This algorithm requires the user to interactively specify the valleys in the histogram for data splitting. Eigen et al. (1974) also use equal-interval histograms for each dimension and identify modes by recording sign changes in the finite differences of the cell counts.

The mode separation procedure of Kittler (1976) uses a Parzen window estimate of the density function with a hypercubic "kernel function" (Duda and Hart, 1973). Unimodal regions of the pattern space are identified as follows. The starting pattern is chosen randomly and corresponds to the first point in the sequence. The second point in the sequence is that pattern which has a maximum density in a hypercubic window around the first pattern. The pattern with the maximum density in the region which is the union of the windows around the first two patterns is selected for the third point. A one-dimensional sequence of density estimates is thus obtained in which each pattern is represented once and only once. The regions of dense patterns correspond to the peaks in this plot. Shaffer et al. (1979) have analyzed this algorithm and demonstrate that the results of this mode-seeking algorithm are always identical to the results of single-link clustering. Thus, seemingly different clustering algorithms can give the same results.

The underlying density of patterns can also be estimated by the k_n-nearest-neighbor method (Wong and Lane, 1983). Two patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ are said to be neighbors if $\mathbf{x}_i$ is one of the k_n nearest neighbors of $\mathbf{x}_j$ and if $\mathbf{x}_j$ is among the k_n patterns closest to $\mathbf{x}_i$. The dissimilarity between neighboring patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by

$$d(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2\hat{p}_n(\mathbf{x}_i)} + \frac{1}{2\hat{p}_n(\mathbf{x}_j)}$$

Pairs of patterns that are not neighbors are assigned arbitrarily large dissimilarities. A single-link clustering algorithm (Section 3.2.1) is then applied to this dissimilarity matrix to generate hierarchical clusters. The parameter k_n is a function of n and is usually taken to be $\log_2 n$. However, different values of k_n can lead to different clusterings. Wong and Lane (1983) demonstrate that their clustering algorithm is "strongly set consistent" for high-density clusters; that is, the resulting single-linkage clusters are maximally connected sets of the form

$$\{\mathbf{x} \mid \hat{p}_n(\mathbf{x}) \geq p^*\}$$

for some fixed density level p^*. Mode-seeking clustering methods have been used extensively in the engineering literature, particularly in remote sensing applications.

3.3.6 Clustering by Graph Theory

Various kinds of geometric structures or graphs for analyzing multidimensional patterns have led to some useful algorithms which can identify irregularly shaped or nonglobular clusters. Section 3.2 has exhibited a number of algorithms for hierarchical clustering based on graph theory. This section treats objects as points in a pattern space, so distances are available between all pairs of objects. The methods in this section seek single partitions, not hierarchies.

A graph is constructed whose nodes represent the patterns to be clustered and whose edges represent relations between the nodes (see Appendix G). In the simplest case, every node is connected to the remaining $(n - 1)$ nodes, resulting in the complete graph. The edge weights are distances between pairs of patterns. For the purpose of clustering, it is the relative positions of the points that is important; pairs of patterns in the same cluster should be closer than pairs of patterns belonging to different clusters.

Several graph structures, such as minimum spanning trees, relative neighborhood graphs, and Gabriel graphs, have been imposed on the set of patterns to capture perceptual grouping. These graphs choose a subset of the $n(n - 1)/2$ edges in the complete graph to reflect the "structure" or the inherent separation among clusters. The edges in these graphs mostly correspond to small interpoint distances. These graphs depend only on the ordering of the lengths of the edges. Clustering methods decompose the graphs into connected components by identifying and deleting "inconsistent" edges. Each component represents a cluster.

Zahn (1971) demonstrated how the minimum spanning tree (MST) can be used to detect clusters. His choice of MST was influenced by the Gestalt principle, which favors that grouping of patterns which represents smaller interpoint distances. The basic idea of Zahn's clustering algorithm is very simple and consists of the following steps.

ZAHN'S CLUSTERING ALGORITHM

Step 1. Construct the MST for the set of n patterns given.

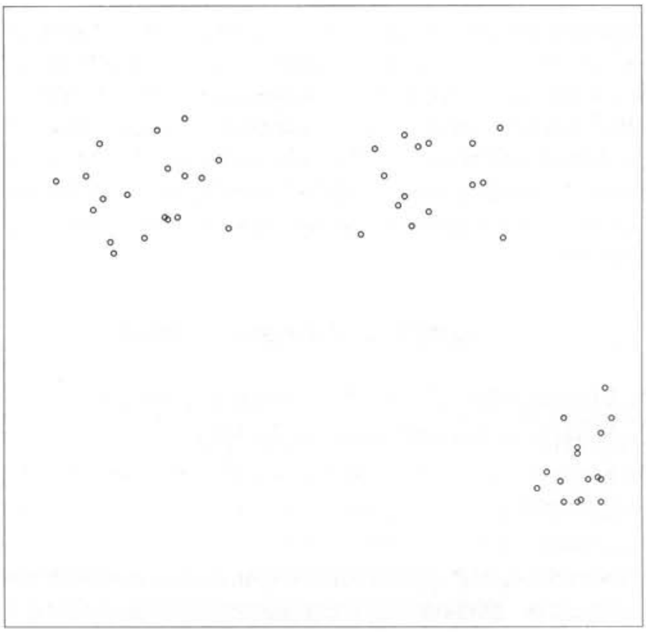
Step 2. Identify inconsistent edges in the MST.

Step 3. Remove the inconsistent edges to form connected components and call them clusters.

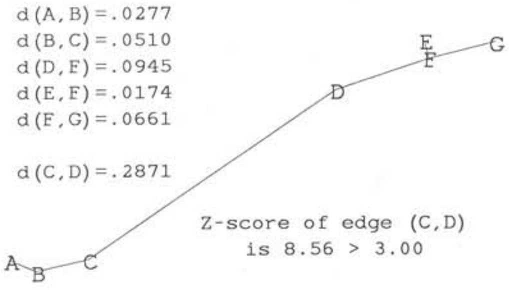
Zahn's algorithm can be applied iteratively to each of the resulting components to identify subclusters. Section 3.2.4 explains the relation between single-link clustering and the MST. The crucial step in the algorithm is the definition of inconsistency. Zahn considers several criteria for inconsistency. In one, an edge is inconsistent if its weight (interpoint distance) is significantly larger than the average of nearby edge weights. Thus the inconsistent edges are related to cluster separation. The number of standard deviations by which an edge weight differs from the average of nearby edge weights and the ratio of the edge weight to the average of nearby edge weights are two means for identifying inconsistent edges. An edge with a factor of inconsistency of two usually links two clusters and can be deleted.

Figure 3.30 illustrates inconsistent edges for a two-dimensional data set. Figure 3.30(a) shows a set of two-dimensional patterns containing three well-separated clusters. Figure 3.30(b) demonstrates an inconsistent edge in the MST for a set of seven patterns. Since the z-score of edge (C, D) is greater than 3, it is unusually long compared to its neighboring edges, and hence is labeled as an inconsistent edge. Figure 3.30(c) identifies inconsistent edges in the MST for the data in Figure 3.30(a). The two intercluster edges have been correctly identified, but an additional inconsistent edge breaks the cluster in the upper left corner into two components.

Zahn has applied his algorithm to a number of data sets consisting of clusters with different shapes and properties, including touching clusters, clusters with smoothly varying point densities, smoothly varying nonhomogeneous clusters, and line-like clusters. The results show that while the above-mentioned notion of inconsistent edges works well for disjoint clusters, special heuristics are needed for

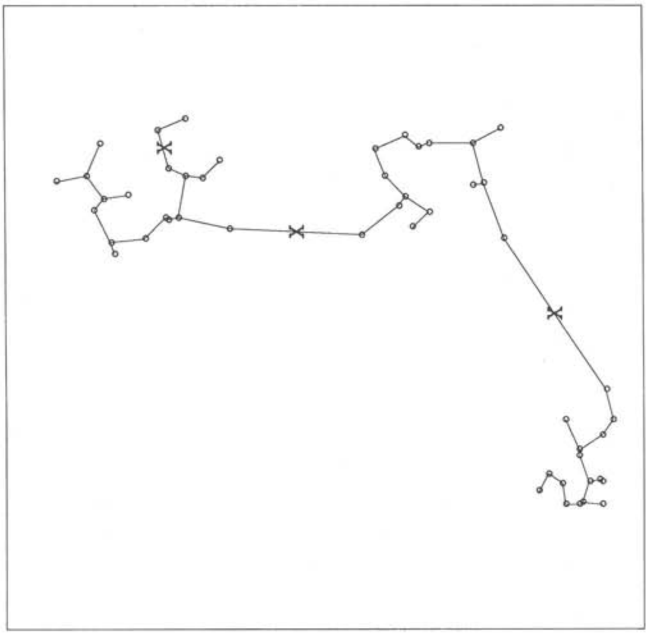


(a)



(b)

Figure 3.30 Inconsistent edges in two-dimensional data: (a) three well-separated clusters; (b) an example of an inconsistent edge; (c) MST of (a) with inconsistent edges marked X.



(c)

Figure 3.30 (*continued*)

more complex situations. For example, in the case of two fairly homogeneous clusters of different point density shown in Figure 3.31, several inconsistent edges will be found in the sparse cluster. Zahn suggests first detecting and deleting the denser cluster and then analyzing the remaining data. The histogram of the edge lengths in the MST helps us in identifying the intercluster edges; intercluster edges occupy the region between the two peaks (corresponding to within-cluster edge lengths) in the histogram of edge lengths.

Prior knowledge of the shapes of the clusters is needed to select the proper heuristic to identify inconsistent edges. This is the greatest deficiency of the MST-based approach in more than two dimensions. The very reason for applying cluster analysis is often to estimate the shapes of clusters and their structure. Nevertheless, MST-based clustering is an important technique which complements the square-error partitional technique. Some other variations of Zahn's idea have also been reported in the literature (Fehlauer and Eisenstein, 1978; Page, 1974; Magnuski, 1975). Koontz et al. (1976) and Mizoguchi and Shimura (1980) base their clustering algorithms on directed trees.

Two other geometric structures, the relative neighborhood graph (RNG) and the Gabriel graph (GG), have also been used in cluster analysis (Urquhart, 1982; Matula and Sokal, 1980). These connected graphs are based on a region of influence

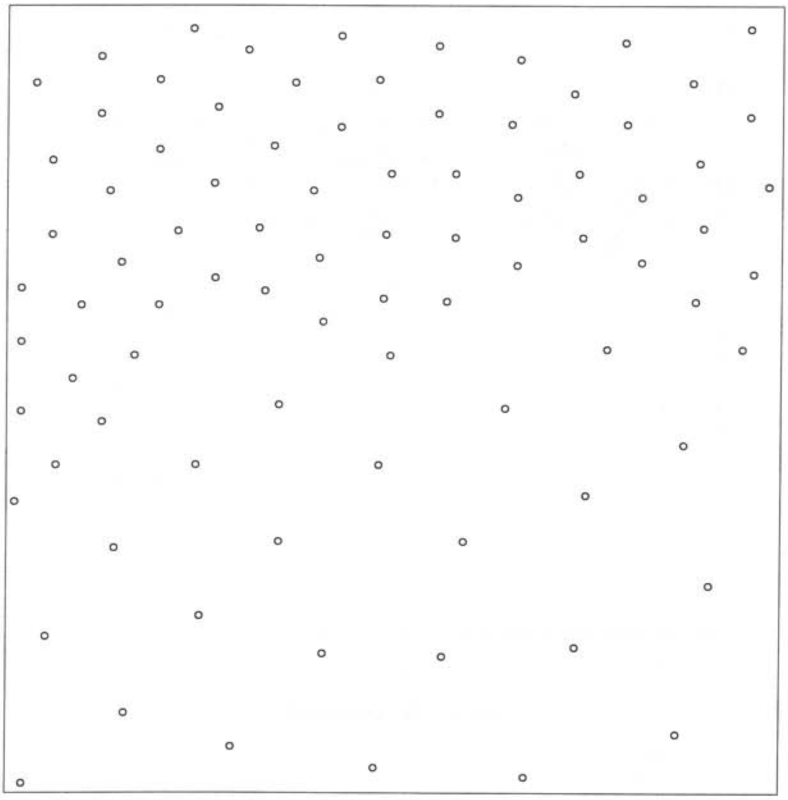


Figure 3.31 Two homogeneous clusters with different density.

(Toussaint, 1980). Patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ are defined to be relative neighbors, and are connected in the RNG, if and only if

$$d(\mathbf{x}_i, \mathbf{x}_j) \le \max\{d(\mathbf{x}_i, \mathbf{x}_k), d(\mathbf{x}_j, \mathbf{x}_k)\} \qquad \text{for all } k, \quad k \ne i \text{ and } k \ne j$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ denotes the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. Or, we can say that $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected in RNG if and only if no other point falls in LUNE($\mathbf{x}_i$, $\mathbf{x}_j$), where LUNE($\mathbf{x}_i$, $\mathbf{x}_j$) is the intersection of the two disks of radius $d(\mathbf{x}_i, \mathbf{x}_j)$ centered at $\mathbf{x}_i$ and $\mathbf{x}_j$. LUNE is the region of influence of RNG and is shown in Figure 3.32(a).

The Gabriel graph (GG) is defined as follows. Points $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected in GG if and only if

$$d^2(\mathbf{x}_i, \mathbf{x}_j) < d^2(\mathbf{x}_i, \mathbf{x}_k) + d^2(\mathbf{x}_j, \mathbf{x}_k) \qquad \text{for all } k, \quad k \ne i \text{ and } k \ne j$$

This is equivalent to the condition that two points $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected in GG if and only if no other point lies in DISK($\mathbf{x}_i$, $\mathbf{x}_j$), where DISK($\mathbf{x}_i$, $\mathbf{x}_j$) is the disk with diameter $d(\mathbf{x}_i, \mathbf{x}_j)$ as shown in Figure 3.32(b). We say that DISK is the

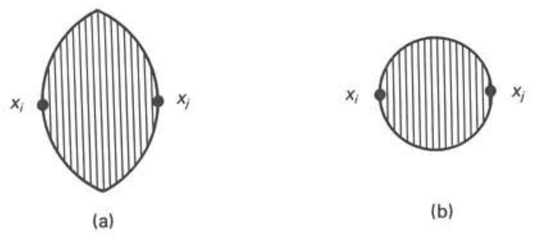


Figure 3.32 Regions of influence for RNG and GG.

region of influence for this graph. The construction of RNG and GG is well understood for two-dimensional data, but the construction of these graphs in high dimensions is a difficult problem.

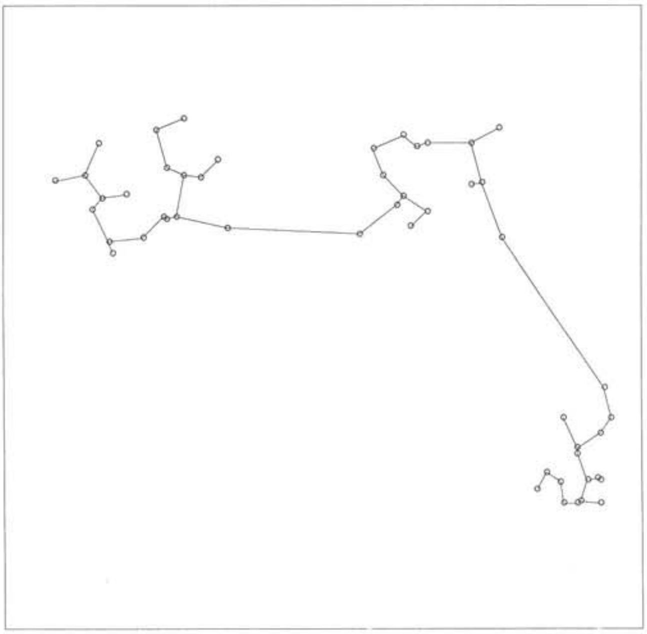
Delaunay triangulation (DT) is another graph structure that is useful in point pattern analysis and plays a prominent role in many algorithms that compute GG and RNG; efficient algorithms for computing DT are available and RNG and GG can easily be obtained from DT by deleting some of the edges in DT. The definition of DT is best made in terms of its dual structure, the Dirichlet tessellation. The Dirichlet tessellation, also well known as the Voronoi diagram, of a set of patterns $\mathcal{X}$ in R^d (d-dimensional Euclidean space) is a partition of R^d into "cells" about each pattern vector $\mathbf{x}_i$ such that each cell consists of those points of R^d lying closer to $\mathbf{x}_i$ than to any other pattern in $\mathcal{X}$. Cell boundaries are intersections of the perpendicular bisectors of the lines connecting $\mathbf{x}_i$ to each of the $(n - 1)$ other patterns in $\mathcal{X}$. Thus each cell is a convex polygon.

The Delaunay triangulation is defined as follows. The edge connecting points $\mathbf{x}_i$ and $\mathbf{x}_j$ is in the DT if and only if the two cells of the Dirichlet tessellation containing $\mathbf{x}_i$ and $\mathbf{x}_j$ share a common boundary. A large body of applications of DT resides in such varied disciplines as biology and geography. For example, DT has been used as a model of territories of breeding bird species (Sibson, 1980). Ahuja (1982) outlines applications of DT to problems in clustering, matching, and segmentation. (See Chapter 5 for segmentation and matching in the image processing context.) An implementation of an agglomerative clustering algorithm based on DT has been made by Howe (1978) and applied to observations of pollen in lake sediments with the goal of partitioning the region with respect to forest type. However, these applications have been developed only for two-dimensional data.

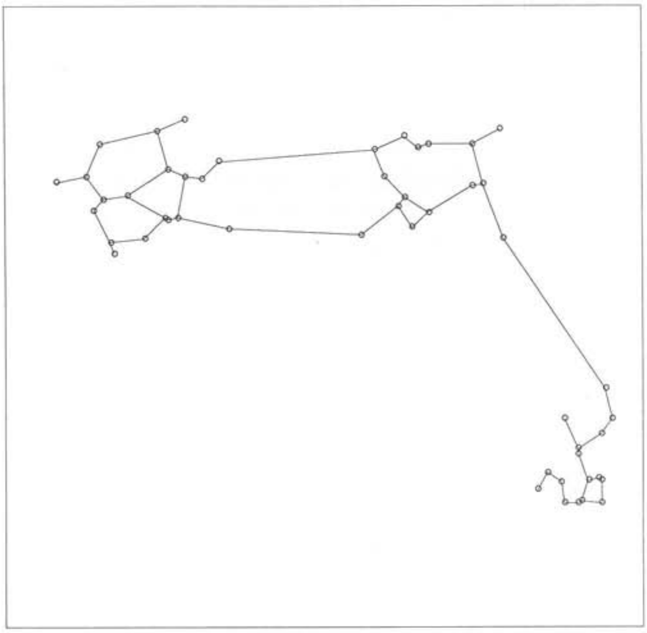
A minimum spanning tree and a Delaunay triangulation (DT) play important roles as "bounds" on RNG and GG. It can be shown that

$$E(\text{MST}) \subseteq E(\text{RNG}) \subseteq E(\text{GG}) \subseteq E(\text{DT})$$

where E denotes the edge set of a graph. The first set inclusion guarantees that the RNG is a supergraph of its MST. Thus every RNG is connected. Lee and Preparata (1984) define a method for efficiently computing the RNG or GG. Figure 3.33 demonstrates the MST, RNG, GG, and Delaunay triangulation for the data in Figure 3.30(a).

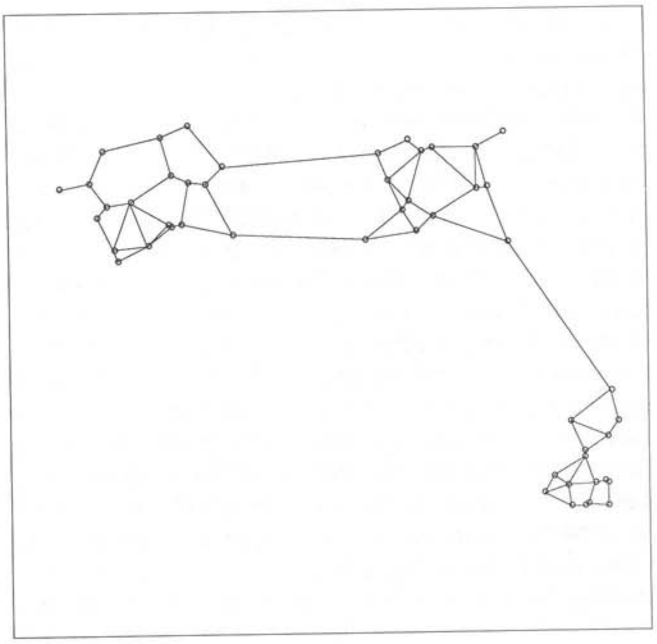


(a)

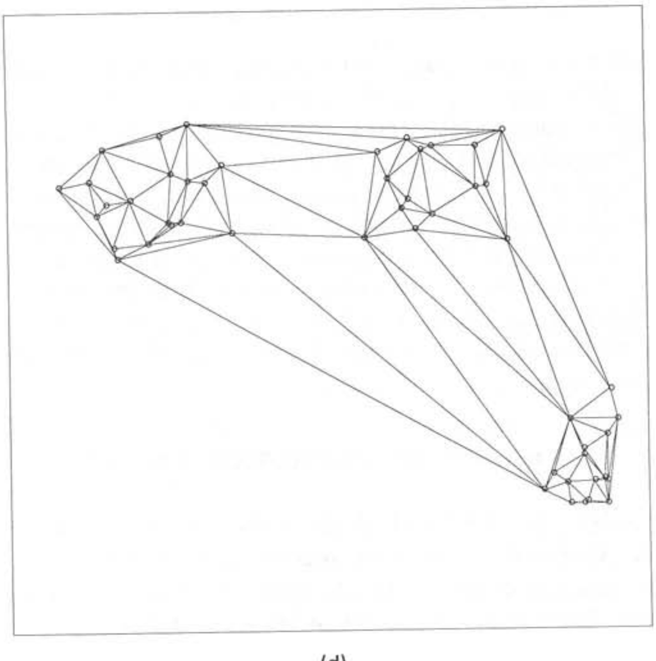


(b)

Figure 3.33 MST, RNG, GG, and Delaunay triangulation for the data in Figure 3.30(a): (a) MST; (b) RNG; (c) GG; (d) Delaunay triangulation.



(c)



(d)

Figure 3.33 *(continued)*

Clustering algorithms based on RNG, GG, and DT are very similar to Zahn's MST-based clustering algorithm. Only the first step in his algorithm needs to be changed: from "Construct the MST" to "Construct the appropriate graph structure." The heuristics needed to define inconsistent edges becomes more complex with larger sets of edges. In the case of the MST, removing a single edge always results in two components. This is not always true with the RNG, GG, and DT because there can be more than one path between two patterns. Applications of the RNG, GG, and DT have been limited so far to two- and three-dimensional patterns primarily due to computational difficulties in higher dimensions.

Urquhart (1982) favors the use of GG and RNG over the MST for clustering problems for the following reasons. First, RNG and GG are less sensitive to changes in the positions of patterns than the MST. Second, since the GG and RNG are supergraphs of the MST, they exhibit a greater degree of interconnectedness of patterns and so may be more appropriate in capturing the cluster structure in the data than the MST. Indeed, the RNG-based clustering algorithms developed by Urquhart avoid some of the problems of Zahn's MST algorithm. Ahuja (1982) and Tuceryan (1986) argue for the intuitively appealing characteristics of DT over other graph structures in processing patterns. They have demonstrated that DT is useful for grouping or clustering two-dimensional patterns. However, we again emphasize that the performance of a clustering algorithm is data dependent.

3.3.7 Nearest-Neighbor Clustering

A natural way to define clusters is by utilizing the property of nearest neighbors; a pattern should usually be put in the same cluster as its nearest neighbor. Two patterns should be considered similar if they share neighbors. The notion of nearest neighbors is inherent in the construction of various graphs, particularly the graphs discussed in Section 3.3.6, so graph-theoretic clustering methods are closely related to nearest-neighbor clustering methods. However, they differ significantly in how the clusters are formed and, most important, in the final partition.

A very simple clustering algorithm which is based on the nearest neighbor rule is given below (Lu and Fu, 1978). A set of patterns $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ is to be partitioned into K clusters. The user specifies a threshold, t, on the nearest-neighbor distance.

NEAREST-NEIGHBOR CLUSTERING ALGORITHM

Step 1. Set $i \leftarrow 1$ and $k \leftarrow 1$. Assign pattern $\mathbf{x}_1$ to cluster C_1.

Step 2. Set $i \leftarrow i + 1$. Find the nearest neighbor of $\mathbf{x}_i$ among the patterns already assigned to clusters. Let d_m denote the distance from $\mathbf{x}_i$ to its nearest neighbor. Suppose that the nearest neighbor is in cluster m.

Step 3. If $d_m \leq t$, then assign $\mathbf{x}_i$ to C_m. Otherwise, set $k \leftarrow k + 1$ and assign $\mathbf{x}_i$ to a new cluster C_k.

Step 4. If every pattern has been assigned to a cluster, stop. Else, go to step 2.

The number of clusters generated, K, is a function of the parameter t. As the value of t increases, fewer clusters are generated. The nearest neighbor distance in step 2 can be replaced by the average distance between $\mathbf{x}_i$ and its p nearest neighbors in the mth cluster. Then the user has to specify another parameter, namely, p. Lu and Fu (1978) have used this clustering algorithm to cluster patterns represented by sentences or strings in an application of syntactic pattern recognition to character recognition.

Jarvis and Patrick (1973) defined a proximity measure as the number of matches in near-neighbor lists for two patterns. Their clustering algorithm can be summarized as follows: Place patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ into the same cluster if $\mathbf{x}_i$ and $\mathbf{x}_j$ share at least k_t near neighbors and $\mathbf{x}_i$ and $\mathbf{x}_j$ are k-near neighbors of each other. This algorithm is noniterative and is computationally attractive since near neighbors can be computed efficiently (Kamgar-Parsi and Kanal, 1985). However, the user has to specify the size of the neighborhood, k, and the similarity threshold, k_t. Jarvis and Patrick (1973) do not provide any guidelines for choosing these parameters but suggest finding the "best" value interactively. Note that large values of k bias the algorithm toward globular structures, whereas small values of k favor chained or elongated structures (Jarvis, 1978). A hierarchy can also be generated by varying the value of k_t. Jarvis and Patrick claim that since Zahn's MST method is based on the first near neighbor, it is a first-order method and seeks linear and elongated clusters at the expense of globular structures.

The notion of proximity based on shared nearest neighbors has been modified by Gowda and Krishna (1978) to measure the "mutual nearness" of two patterns. If $\mathbf{x}_j$ is the pth near neighbor of $\mathbf{x}_i$ and $\mathbf{x}_i$ is the qth near neighbor of $\mathbf{x}_j$, then the *mutual neighborhood value* (MNV) between $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as $(p + q)$. The smaller MNV, the more similar the patterns. This represents a stronger notion of similarity than the number of shared neighbors of Jarvis and Patrick. Gowda and Krishna's clustering algorithm is described below.

MUTUAL NEIGHBORHOOD CLUSTERING ALGORITHM

Step 1. Determine the k near neighbors of every pattern.

Step 2. Compute the MNV for every pair of patterns. If patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ are not mutual neighbors for a given value of k, set MNV $(\mathbf{x}_j, \mathbf{x}_j)$ to an arbitrarily large number.

Step 3. Identify all the pairs of patterns with MNV of 2. Merge each such pair into a cluster, starting with the pair having the smallest distance.

Repeat step 3 for MNV thresholds of $3, 4, \ldots, 2k$ to generate a hierarchy.

The parameter k that controls the neighborhood depth is crucial to the performance of the algorithm. Small values of k give several "strong" clusters and large values of k give fewer "weak" clusters. In fact, k can always be chosen sufficiently large to make the algorithm return a single cluster. Gowda and Krishna (1978) demonstrate that the algorithm is able to identify nonspherical clusters, linearly nonseparable clusters, clusters with unequal populations, and clusters with low-density bridges when k is 5 in two dimensions. However, no heuristic is provided to select an appropriate value of k for arbitrary data sets.

3.3.8 Fuzzy Clustering

The clustering algorithms described so far assign each pattern to one and only one cluster. In other words, the patterns are partitioned into disjoint sets; patterns in one cluster are supposed to be more similar to each other than to patterns in different clusters. If the clusters are compact and well separated, as demonstrated in Figure 3.34(a), there is no ambiguity or uncertainty associated with assigning each pattern to one cluster. We can easily see that there are two clusters in Figure 3.34(a) with well-defined boundaries. But what happens if the

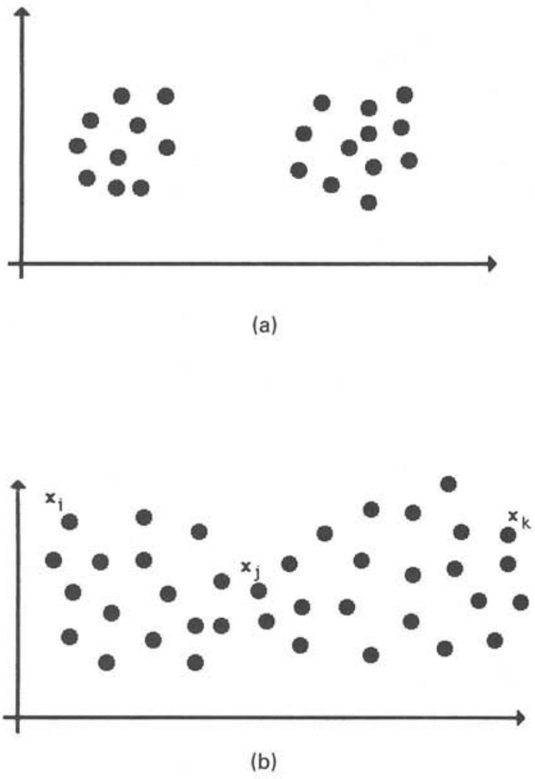


Figure 3.34 Examples of cluster structures: (a) well-separated clusters; (b) touching or overlapping clusters.

clusters are touching or overlapping? Figure 3.34(b) illustrates a case in which cluster boundaries are not sharp and the assignment of patterns to clusters is difficult. Although it is clear that patterns $\mathbf{x}_i$ and $\mathbf{x}_k$ should be put in different clusters, pattern $\mathbf{x}_j$ could be put into either the cluster containing $\mathbf{x}_i$ or the cluster containing $\mathbf{x}_k$. Such clusters are said to have "fuzzy" boundaries.

The fuzzy set theory developed by Zadeh (1965) permits an object to belong to a cluster with a grade of membership. The degree of membership takes a value in the interval [0, 1]. For ordinary clusters, called "crisp" clusters, the membership grade for pattern $\mathbf{x}_i$ in a particular cluster is 1 if the pattern belongs to the cluster and 0 if it does not. With fuzzy clusters, pattern $\mathbf{x}_i$ has a grade of membership, $f_q(\mathbf{x}_i) \geq 0$, or degree of belonging to the qth cluster, where $\sum_q f_q(\mathbf{x}_i) = 1$. The larger $f_q(\mathbf{x}_i)$, the more confidence exists that $\mathbf{x}_i$ belongs to cluster q. If $f_j(\mathbf{x}_i)$ is 1, pattern $\mathbf{x}_i$ belongs to cluster j with absolute certainty. The interpretations of values such as 0.3 is less clear. Membership grades are subjective in nature and are based on definitions rather than measurements (Zadeh, 1984). For example, pattern $\mathbf{x}_j$ in Figure 3.34(b) could belong to one cluster with membership value 0.45 and to the other cluster with membership value 0.55.

The grade of membership is not the same as the probability that the pattern belongs to the cluster even though grades of membership and probabilities both take values in the range [0, 1]. Under a probabilistic framework, pattern $\mathbf{x}_j$ belongs to one and only one cluster, depending on the outcome of a random experiment. In fuzzy set theory, pattern $\mathbf{x}_j$ can belong to two clusters simultaneously. The membership grades determine the degree to which two cluster labels are applicable.

Another interpretation of the degree of membership is that it measures the compatibility of a pattern or an object with the description of a fuzzy set. Sometimes this property is useful in interpreting the results of a clustering algorithm. Consider the problem of clustering a variety of computers ranging from microcomputers to mainframes based on such attributes as memory size, CPU speed, and processor type. The objective might be to partition computers into two clusters labeled "personal" and "multiuser system." To which cluster should a computer based on a Motorola 68020 processor be assigned, which is used in a variety of personal computers and workstations?

Proponents argue that fuzzy clustering is more appropriate than ordinary clustering for capturing human concepts such as "small," "big," "high," and "low." Fuzzy sets are likely to find increasing use in applications involving imprecise and incomplete information, commonsense reasoning, and complex concepts (Zadeh, 1984). Skeptics of fuzzy clustering do not doubt its mathematical correctness, but are not convinced that it offers any advantages over the classical and better understood clustering methods. Many more papers have been written on the theoretical foundations of fuzzy sets and fuzzy logic than on its practical applications.

Clustering has always been a popular domain for fuzzy sets. Early work by Bellman et al. (1966), Ruspini (1969), Gitman and Levine (1970), Bezdek (1974), and Dunn (1974) have culminated in two books on fuzzy clustering (Bezdek, 1981; Backer, 1978). Virtually all the clustering algorithms based on fuzzy set theory are partitional in nature, but a few generate hierarchies. Some of these fuzzy algorithms are straightforward modifications of the square-error type of partitional algorithms discussed in Section 3.3.1. Indeed, Bezdek (1976) proposed a fuzzy ISODATA clustering algorithm for which convergence theorems are available.

The crucial step in a fuzzy clustering algorithm is the definition of the membership function. Backer (1978) shows how to construct a membership function based on similarity decomposition. Let the set of patterns $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be initially partitioned into clusters $\{C_1, \ldots, C_K\}$ and let n_i be the number of patterns in C_i. Let $\delta(\mathbf{x}, C_i)$ denote the similarity between pattern $\mathbf{x}$ and cluster C_i. The larger this value, the closer are the pattern and the cluster. The cluster membership function $f_{C_i}(\mathbf{x})$ for pattern $\mathbf{x}$ induced by cluster C_i is given by

$$f_{C_i}(\mathbf{x}) = P_i\delta(\mathbf{x}, C_i) \Big/ \sum_{k=1}^{K} P_k\delta(\mathbf{x}, C_k)$$

where $P_k = n_k/n$ is the relative size of cluster C_k. This membership function is nonnegative and sums to 1 for every pattern.

$$f_{C_k}(\mathbf{x}) \geq 0 \qquad \text{and} \qquad \sum_{k=1}^{K} f_{C_k}(\mathbf{x}) = 1$$

The similarity or affinity function, $\delta(\mathbf{x}, C_k)$, can be based on the distance concept, the neighborhood concept, or the probabilistic concept (Backer, 1978). It measures the relationship between a pattern and a cluster as a whole or between a pattern and one or more representatives of that cluster. The choice of this function depends on the data. Backer and Jain (1981) define an affinity function based on the mean vectors of the clusters. The membership function is

$$f_{C_k}(\mathbf{x}) = \frac{1 - (1/\beta)d(\mathbf{x}, \mathbf{m}^{(k)})}{K - (1/\beta)\sum_j d(\mathbf{x}, \mathbf{m}^{(j)})}$$

where $d(\mathbf{x}, \mathbf{m}^{(k)})$ denotes the Euclidean distance between the pattern vector $\mathbf{x}$ and the centroid $\mathbf{m}^{(k)}$ of cluster C_k. The parameter β controls the neighborhood size and affects the values of cluster belongingness. Only general guidelines are available for choosing β. The performance of a fuzzy clustering algorithm depends critically on the definition of the membership function.

Fuzzy partitional clustering algorithms generate partitions that minimize induced fuzziness following the same steps as square-error clustering algorithms. The induced fuzziness takes its minimum value if we can obtain a partition for which $f_{C_k}(\mathbf{x}) \in \{0, 1\}$ or, equivalently, when the partition is nonfuzzy. Therefore, a criterion function needs to be defined to characterize the induced fuzziness of a partition. Backer (1978) defines a number of fuzzy partitioning criterion functions. For example, a criterion based on the average pairwise fuzzy set separability is given by

$$\Phi_f = 1 - \frac{2}{K-1} \sum_{k=1}^{K-1} \sum_{j=k+1}^{K} I(f_{C_k} \cap f_{C_j})$$

where K is the number of clusters and $I(f_{C_k} \cap f_{C_j})$ is the intersection of two fuzzy sets (clusters) defined as follows:

$$I(f_{C_k} \cap f_{C_j}) = \frac{1}{n} \sum_{\mathbf{x} \in \mathscr{X}} \min\,[f_{C_k}(\mathbf{x}), f_{C_j}(\mathbf{x})]$$

The minimum value of Φ_f is 0, which represents maximum fuzziness, and the maximum value of Φ_f is 1, which corresponds to a nonfuzzy partition. The problem of fuzzy clustering is to find that partition which maximizes Φ_f. The basic steps in a fuzzy partitional clustering algorithm are given below.

FUZZY PARTITIONAL CLUSTERING ALGORITHM

Step 1. Select an initial partition $\{C_k\}_{k=1}^{K}$.
Repeat steps 2 to 4 until the cluster memberships stabilize.
Step 2. Compute the membership functions $\{f_{C_k}(\mathbf{x}_j)\}$.
Step 3. Compute the criterion function Φ_f.
Step 4. Reclassify patterns to improve Φ_f.

The output of a fuzzy algorithm not only includes a partition but also additional information in the form of membership values. However, the new information provided by the membership values must be interpreted by the data analyst. In summary, fuzzy clustering is an interesting concept that includes most partitional clustering algorithms as special cases. However, its superiority to ordinary clustering has yet to be demonstrated in applications.

3.4 CLUSTERING SOFTWARE

Someone interested in applying clustering techniques has plenty of software to choose from. As we reported earlier, there is no shortage of clustering algorithms, and most of them have been implemented to run on a variety of computers. One of the main reasons for the large number of "different" clustering programs available is that most researchers and research groups put all their trust in the algorithm they have developed even though it is very similar to existing algorithms. In addition, documented and tested clustering packages were not available until recently, so users had to write their own software. The paper by Johnston et al. (1979) illustrates how the "law of serendipity" can lead to a "new" clustering algorithm. During the coding of the mode-seeking algorithm by Koontz et al. (1976), Johnston et al., by mistake, modified one of the expression for the relative density of a pair of patterns. This led to a new clustering algorithm which provided better clustering for "uniform, touching" clusters than the original!

A clustering technique imposes a particular structure on the given data. Since the purpose of cluster analysis is to reveal structure or ''pattern'' in large quantities of numerical data, one should not put too much faith in interpreting a single clustering. An agreement in the clusterings generated by two or more different clustering methods increases one's confidence in accepting the imposed structure as the true structure present in the data. We again emphasize the difference between a clustering method and a clustering algorithm. An algorithm is a particular way of implementing a method. Thus the square-error clustering method can be implemented in several ways, as illustrated by programs such as ISODATA, CLUSTER, FORGY, and K-MEANS. These algorithms might not generate the same clusterings when applied to a data set. So it is important that a user of cluster analysis should be aware of the true differences between the various clustering programs.

Blashfield et al. (1982) provide excellent coverage of the availability of cluster analysis software. They even suggest that ''there may be as many different programs in existence to perform cluster analysis as there are users.'' They list three major categories of cluster analysis software which are summarized briefly below.

1. *Collection of subroutines and algorithms*. These are usually programs developed by a research group which can be obtained without any charge. Some of these collections have appeared in the books by Anderberg (1973) and Hartigan (1975). There is generally no guarantee on the quality of such software. A few subroutines are available through a commercial organization called IMSL (International Mathematical and Statistical Library of scientific subroutines) which are well maintained. Users have to write their own driver programs to execute IMSL routines.
2. *General statistical packages that contain clustering methods*. Several of the well-known statistical packages, such as the BMDP series, SAS, and OSIRIS, have a limited collection of clustering programs. These packages are widely available at university computer centers, are well maintained, and come with good documentation. New versions of these packages now run on personal computers also.
3. *Cluster analysis packages*. The best known and most versatile clustering package available is a collection of programs written in Fortran called CLUSTAN. CLUSTAN contains a comprehensive choice of clustering techniques. This package can be obtained from: Computer Center, University College, London, 19 Gordon Street, London, WC1H 0AH. Other clustering packages are NT-SYS, CLUS, TAXON, BC-TRY, S, and CLAB. A package called ICICLE performs hierarchical clustering and displays the results as Icicle plots. It can be obtained from: Computer Information Services Group, Room 2F128, AT&T Bell Laboratories, Murray Hill, NJ 07974. These packages offer more flexibility than a general statistical package. Some of the options available to the user include data screening, data transformation, a wide choice of similarity measures, and cluster diagnostics.

In summary, there is no "complete" cluster analysis package available in the market. The software described above implements only the most popular techniques. Perhaps this is appropriate in light of the large number of clustering methods reported in the literature. The packages available tend to favor hierarchical techniques over the partitional techniques because of the predominance of clustering users in social, behavioral, and biological sciences. Since there is no "best" clustering program, serious users of clustering must be prepared to implement algorithms which are more suited to their data than those available commercially. Software for pattern recognition algorithms, such as ISPAHAN (Gelsema, 1980), and multidimensional scaling, such as MDSCAL, will also be useful to a user of cluster analysis.

3.5 CLUSTERING METHODOLOGY

This chapter has explained the details of clustering algorithms. Chapter 2 presented ways of inspecting and representing data. Chapter 4 reviews a number of techniques for validating the results of clustering algorithms. How does it all go together? This section proposes a generic methodology for focusing the tools developed in this book into a procedure for analyzing data. The needs and special circumstances encountered in individual problems may make parts of the methodology unnecessary and other parts impossible. However, it is worthwhile to take a step back from the details and look at the big picture as is done in Section 3.5.1. Comparative analyses of clustering methods that summarize experimental facts about cluster analysis and aid in choosing strategies and tactics are reviewed in Section 3.5.2.

This methodology is more of a proposal than an accepted standard. Case studies that follow this methodology are difficult to find because each application has its special needs. We recommend this methodology for the serious practitioner.

3.5.1 Exploratory Data Analysis

Figure 3.35 relates the major steps to be considered when undertaking an exploratory data analysis whose central component is a cluster analysis. We see

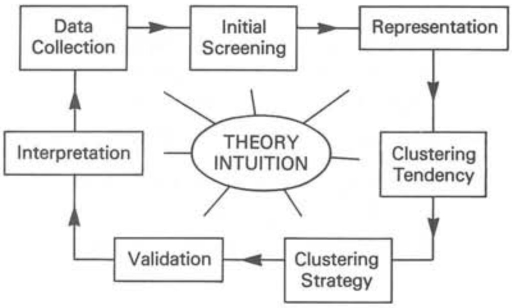


Figure 3.35 Clustering methodology.

the process as an endless loop in which new insights are obtained and new ideas generated each time through the loop. The end result could be the design of an experiment that uses standard statistical tools to come to decisions about the phenomenon being studied. One might derive enough information about the phenomenon from an exploratory data analysis itself to draw informal conclusions. We do not intend to discuss the appropriate use of information but hasten to add that an expert in the field of application is invaluable in guiding the data analysis and providing insights. Exploratory data analysis remains a tool for discovery and the techniques presented in this book should serve that end. However, the fact that cluster analysis is exploratory in nature does not mean that only ad hoc procedures can be adopted. Comments on the individual boxes in Figure 3.35 are given below.

1. *Data collection*. The careful recording of data in accordance with the standards in the area of application is the first important step in the analysis. Previous work in the subject matter, the resources available, and the patience of the investigator must be considered. The amount and type of data will strongly influence the strategies available for analyzing the data, so a few iterations through the loop in Figure 3.35 might be necessary before a meaningful and compact procedure for data collection can be established. The material in Sections 2.1 and 2.2 should be applicable to this phase.

2. *Initial screening*. Raw data usually need some massaging before they are ready for formal analysis. We have the normalizations in Section 2.3 in mind, but the data can be looked at in a rough manner in several ways by Chernoff faces, Andrew's plots, and other visual aids (Section 2.5.1). For example, all the values of a particular feature might be the same, so that feature can be eliminated. The feature variances quickly isolate such features. The screening method should suit the data and the investigator should have an understanding of and confidence in the method chosen.

3. *Representation*. The problem here is to put the data into a form suitable for further analysis. This includes choosing a proximity index (Section 2.2), projecting the data to a suitable feature space (Sections 2.4 and 2.5), examining the intrinsic dimensionality (Section 2.6), and performing a multidimensional scaling (Section 2.7). The suitability of any of these procedures depends on the problem at hand. The end result should either be a pattern matrix or a proximity matrix. The representation chosen will depend on the data, the application area, the experience of the investigator, and the availability of computer software.

4. *Clustering tendency*. Are the data random or does some justification exist for clustering? This step is often ignored, but we feel that it is important. The information gained from this step can not only prevent the inappropriate application of clustering algorithms, but can also provide information on the fundamental nature of the data. Section 4.6 reviews the procedures for this step. The issues raised are somewhat subtle and require one to determine just what is known about the data, such as the sampling window and the definition of randomness suited to the data. If the data cannot be shown to have the tendency to cluster, one is well advised to pursue analysis techniques other than cluster analysis.

5. *Clustering strategy*. Sections 3.1 to 3.3 lay out the factors involved in a cluster analysis and in choosing a clustering strategy. A major question is the choice between hierarchical (Section 3.2) and partitional (Section 3.3) procedures. Within each type of clustering, thought must be given to several details, such as matching the algorithm to the data, the presentation of results, and the choice of parameters. The amount of data, as measured by the number of patterns and the number of features, is a major factor. One can also choose to cluster both the patterns and the features. Section 3.5.2 summarizes information about the comparative performance of clustering algorithms that might help in choosing an algorithm. Computer software (Section 3.4) in which the investigator has confidence must be available.

6. *Validation*. The prudent and enlightened validation of clustering results is the essential step that changes a qualitative analysis into hard evidence. The external indices of cluster validity discussed in Chapter 4 compare the results of cluster analysis to what the investigator would like to see. Internal indices assess the merit of the clustering results on an objective basis. Both types of indices should be considered. Validation often involves Monte Carlo analysis and statistical testing. It demands more computer resources, time, and care than collecting the data and clustering it. This is the price for generating reliable results.

Validation can mean more than the application of the validity indices in Chapter 4. One might choose to study the stability of the analysis by perturbing the data slightly and repeating the analysis. The data can be ''shaken'' (Van Ness, 1983; Gnanadesikan et al., 1977; Strauss, 1973) by adding random noise or by removing some patterns or by removing some features. Smith and Dubes (1980) propose two indices for studying the stability of single- and complete-link hierarchies. Stability is one basis for comparing clustering methods.

7. *Interpretation*. How does one integrate the results of cluster analysis with previous studies and draw conclusions about the data? How does one get ideas? We can offer no concrete suggestions, except to look at the applications in Chapter 5 and other uses of cluster analysis to see what has been done. The more exploratory data analysis is used, the more confident one becomes in its use.

3.5.2 Comparative Analysis

Several choices must be made in selecting a clustering strategy (Dubes and Jain, 1976). Setting aside the question of data representation, the key choices are the clustering method and the validity index. Few theoretical guidelines are available to choose among methods and indices. In this section we review some work on comparative analysis of clustering methods. No generally accepted methodology for comparative analysis exists, so we report a sampling of the approaches taken in the literature.

Basis for comparison. Milligan (1981a) reviews most of the literature on comparative analysis of clustering algorithms prior to 1981 and provides details on several early studies. Most of this work tried to understand the relative accuracy of various clustering algorithms in extracting the "right" number of clusters from artificially generated data. Anderberg (1973) provides a comprehensive discussion of the factors involved in comparing clustering methods. Dubes and Jain (1976) apply some of these suggestions. It is difficult to summarize the findings of all this work. No two studies used the same methodology or included all the same clustering methods. In fact, it is not clear that the same implementation of a clustering method was used by two studies claiming to be using a particular method. Some studies introduced outliers and noise of various kinds. Each study has its own way of generating artificial data.

A theoretical comparison of clustering algorithms is not feasible because clustering algorithms are almost impossible to model mathematically in such a way that the models can be compared. Jardine and Sibson (1971) attempted to list the essential characteristics of useful clustering methods and concluded that the single-link method was the only one that satisfied all the mathematical criteria. However, many Monte Carlo studies performed over the years report that the single-link method almost always performs poorly. Various attempts have been made to create an axiomatic basis for clustering analyses, thus implying a means for rating clustering methods. Wright (1973) provides one such approach, but the result is a standard *K*-means algorithm. The axioms automatically bias the result toward a certain procedure.

One way to distinguish among clustering methods is by examining the clustering criteria themselves. Gower (1967) concluded one such study by acknowledging the difficulty in defining "cluster" and stressing the relation between clustering method and the type of cluster expected. Marriott (1982) used the effect of adding a single pattern to the data as a way of comparing clustering algorithms. Golden and Meehl (1980) found that the average-link, complete-link, and Ward's clustering method outperformed the single-link, centroid, and median methods on a particular data set. Part of the lore of cluster analysis is that the single-link method produces "straggly" clusters (Hartigan, 1985), while the complete-link and Ward's method look for hyperspherical, compact clusters. The basis for these observations are discussed in Section 3.2.

Another strategem for comparing clustering methods is to compile a list of "admissibility criteria," such as those suggested by Fisher and Van Ness (1971) or Rubin (1967). The idea is to propose a desirable property, *P*, and call a clustering method *P*-admissible if it has this property. Such properties reflect the manner in which clusters are formed, the structure of data, and the sensitivity of clustering techniques to changes in the data such as cluster size. Dubes and Jain (1976) give these properties for several clustering methods. Admissibility provides some insight into a clustering method but does not provide a fair comparison of the performance of clustering algorithms. The remainder of this section considers only Monte Carlo studies.

Monte Carlo studies. Specific conclusions are difficult to distill from all the Monte Carlo studies because the results are limited by the scope of each study. No general agreement exists on names of clustering methods, which further frustrates the process of comparing results. Some examples of broad conclusions drawn from the literature are listed below. All these examples must be qualified by the factors involved in the various studies.

Bayne et al. (1980) suggested preprocessing data prior to cluster analysis to delete unnecessary features and small clusters. Hierarchical methods such as Ward's method and complete-link method were preferable to median, group average, and centroid methods. The single-link method was particularly unsuited to Gaussian data. Blashfield (1976) found that single-link clustering tended to form chains rather than compact clusters, but surprisingly, performed well with spherical clusters and that Ward's method provided accurate results. Cunningham and Ogilvie (1972) concluded that the UPGMA method of hierarchical clustering performed as well as single-link clustering when data were perturbed and stressed the strong interaction between the type of input data and clustering method. Edelbrock (1979) noted that algorithms using correlation as a proximity measure were uniformly more accurate than those using Euclidean distance. Edelbrock and McLaughlin (1980) determined that no single algorithm is best for all applications; clustering methods and proximity measures should be chosen rationally. Gross (1972), Kuiper and Fisher (1975), and Mojena (1975) emphasized the accuracy of Ward's method. Milligan and Isaac (1980) found that both the UPGMA and the complete-link hierarchical clustering methods outperformed Ward's method and that single-link clustering performed worst of the four.

The most comprehensive comparative analysis available in the literature was reported in related papers by Milligan (1980), Milligan et al. (1983), and Milligan and Cooper (1985). The methodology for this study is summarized below. The key elements are the manner in which artificial data are generated and the choice of criterion.

1. Fifteen clustering algorithms, including all the popular hierarchical and partitional ones, were compared to determine if they recovered the "right" number of clusters.
2. Ten groups of artificial data sets were constructed, each with 108 pattern matrices. One group was "error-free" and the other nine had various error conditions imposed, such as outliers, errors in distance, noise features, and normalizations. The 108 error-free pattern matrices consisted of three replication of each of 36 cells. A cell was defined by three factors: the number of clusters (two, three, four, or five), dimensionality (4, 6, or 8) and three

types of cluster sizes. All pattern matrices contained 50 patterns. Individual features were generated from truncated normal distributions.

3. The Rand statistic (Section 4.4.1) was the external criterion and Hubert's Γ index (Section 4.1.2) was the internal index. The external index measured the degree to which the correct partition was recovered. The correct solution is known for all the data sets and dendrograms were cut at the level corresponding to the correct number of clusters.
4. All 1080 data sets were clustered by each of the 15 clustering algorithms. Random data consisting of 108 random pattern matrices formed the baselines for evaluating the values of the Rand and Γ indices.

Milligan (1980) provides tables showing averages of the two indices over all 108 data sets corresponding to each error condition and performs a careful statistical analysis of the results. The effects of alternative starting configurations on the performance of the *K*-means algorithm were also examined. The main results of this study are summarized next.

The *K*-means algorithm gave better results than hierarchical methods only when the starting partition was close to the final solution. The single-link method was only mildly affected by outliers but was strongly affected by errors in distance at levels having virtually no effect on other hierarchical algorithms. No one group of hierarchical algorithms was consistently superior to any other group. The choice of clustering algorithm was found to be more important than the choice of proximity measure. Researchers are cautioned to select variables carefully.

Milligan and Shilling (1985) used the same data set with four external measures of validity (Rand, corrected Rand, Jaccard, and Fowlkes and Mallows, Section 4.4.1) but limited the comparison to four hierarchical clustering methods (single-link, complete-link, UPGMA, and Ward's). Ward's method performed best for clusters of equal size and UPGMA performed best for clusters of unequal size. Recovery increased with the number of dimensions, as would be expected since the number of patterns was fixed at 50. The paper drew other conclusions about the validity indices which are not considered here.

These two papers carried out some of the five recommendations Milligan made in his 1981 paper reviewing Monte Carlo tests of cluster analysis. The number of factors and levels in each factor can skyrocket in such comparisons, but there is no other way to compare clustering methods. The method of data generation suggested is as carefully done as any reported in the literature. However, it involves a number of ad hoc decisions about ranges and levels that are unsettling. Most comparative analyses that begin with pattern matrices generate data from Gaussian distributions because the separation of clusters can be assured (Blashfield, 1976).

Fowlkes and Mallows (1983a,b) proposed another methodology for comparing two hierarchical clusterings; their index is defined in Section 4.4.1. The basic cluster structure was defined by a mixture of Gaussian distributions (Section 3.3.4) in two dimensions. Choosing the variances and correlation coefficient defined the ''true'' structure. The data were ''shaken'' by adding random (normally distributed) deviates to the feature values. They suggested plotting the Fowlkes and Mallows index as a function of the number of clusters to compare two clustering methods, which is a valuable suggestion. Unfortunately, they tried to interpret their external index as an internal index and the results cannot reasonably be interpreted. This fact was noted in discussions of the paper.

Baker (1974) compared single- and complete-link clustering on their sensitivity to perturbations in the structure of the data. The complete-link method was found to outperform the single-link method under a chained type of structure, to which the single-link method is supposed to be sensitive. Hubert (1974b) extended this work to the effects of sample size. Hartigan (1985) supplies evidence for the superiority of single-link over complete-link by stating that complete-link is the worst of all standard methods for finding clusters defined as regions of high density because of its asymptotic behavior.

The comparative analysis of clustering methods presents a continuing problem for research. Hartigan (1985) provides a succinct summary of this research: ''Different classifications [clusterings] are right for different purposes, so we cannot say any one classification is best.'' The sampling of studies and conclusions summarized in this section suggest that one must carefully define the problem domain before a meaningful comparison of clustering methods is possible. New clustering methods are published regularly and are often justified on the basis of their performance on a few carefully chosen sets of artificial data or on a single application. We agree with Everitt (1979) who states that ''it appears unlikely that the relations between different methods and data types will be un-tangled solely by formal analysis and argument.'' We reiterate his call for more investigations using a range of data types. Milligan's (1980) methodology might serve as a model for such investigations.

3.6 SUMMARY

A large collection of clustering algorithms is available to analyze experimental data in a variety of scientific disciplines. New clustering programs continue to appear in the scientific literature. However, most of these algorithms are based on the following two popular clustering techniques: iterative square-error partitional clustering and agglomerative hierarchical clustering. Hierarchical techniques organize the data in a nested sequence of groups. This nesting is displayed in the form of a dendrogram. To obtain clusters from the tree, a threshold on the similarity value (among the patterns) needs to be specified so that the dendrogram can be cut to obtain a partition or clustering. Various agglomerative hierarchical algorithms differ in their definition of the similarity measure between a pattern and a cluster, and between two clusters.

Square-error partitional algorithms attempt to obtain that partition which minimizes within-cluster scatter or maximizes between-cluster scatter. To guarantee that a global optimum solution has been obtained, one has to examine all possible partitions, which is not computationally feasible. So various heuristics are used to reduce the search, but then there is no guarantee of optimality. The type of heuristics used differs from one algorithm to the other. Hierarchical techniques are popular in biological, social, and behavioral sciences, whereas partitional techniques are used more frequently in engineering applications.

Several software packages containing clustering algorithms are available. There is no ''best'' clustering algorithm, so a user is advised to try several clustering algorithms on a given data set. How should one select a clustering method? Several comparative studies have been reported in the literature which can serve as useful guidelines. Cluster analysis is just one of the tools for exploratory data analysis. Thus issues of data collection, data representation, assessing the results of clustering, and interpreting the clusters found are as important as the choice of clustering strategy.

4

Cluster Validity

Cluster validation refers to procedures that evaluate the results of cluster analysis in a quantitative and objective fashion. Hierarchies, clusterings, and clusters are sometimes justified by ad hoc methods based on the application area. After all, exploratory data analysis uses whatever tools are handy to get the job done. This chapter examines statistically based indices for judging the merits of clustering structures in a quantitative manner.

In the first section we explain some tools that are employed in several phases of cluster validity. In particular, the language of testing hypotheses and procedures for performing a Monte Carlo analysis are reviewed. Section 4.2 lays out the factors that must be considered when choosing an index of cluster validity. The next three sections review the three classes of cluster validity problems: the validation of hierarchies, partitions, and individual clusters. Section 4.6 looks at the problem of testing for randomness. The methodology in Section 3.5 links the ideas of this chapter to the algorithms in Chapter 3.

4.1 BACKGROUND

This chapter is based on the premise that the problems of cluster validity are inherently statistical. A clustering structure is ''valid'' if it is ''unusual'' in some sense. We choose to express unusualness in a statistical framework and require that probabilities have an objective interpretation to whomever is validating clustering structures. The first section presents the fundamentals of statistical testing of hypothesis to define terms used throughout the chapter. In other sections we introduce the Goodman–Kruskal gamma statistic and Hubert–Mantel statistics from which tests for cluster validity can be formed. In the last two sections we discuss two computer simulation tools: Monte Carlo analysis and bootstrapping.

4.1.1 Testing Hypotheses

It is easy to propose indices of cluster validity. It is very difficult to fix thresholds on such indices that define when the index is large or small enough to be "unusual." Statistical methods provide a framework for rationally deciding how large is "large" and how small is "small." Several textbooks in statistics cover this material, such as Conover (1971) and Wilks (1963).

A statistic T is a function of the data that is supposed to contain useful information. Statistic T could be the square error of a clustering or the level at which a partition forms in a hierarchy or a compactness measure for a cluster. In mathematical terms, T is a random variable and its distribution describes the relative frequency with which values of T occur under some hypothesis. A distribution requires that a sample space, or baseline population, exists. The choice of population and assumptions made about the population embody ideas of randomness and structure. A hypothesis is a statement about the relative frequency of events in the sample space that expresses one's concept of phrases such as "the data are random" or "the data are clustered." A hypothesis is tested by observing a value of T and deciding whether the observation is unusual, based on a distribution for T.

Randomness hypotheses. A null hypothesis in cluster validity is a statement of "no structure," or randomness, that asserts the frequency with which members of a baseline population occur. The three most common null hypotheses in cluster validity work are the random graph hypothesis, the random label hypothesis, and the random position hypothesis. The subscript "0" refers to a null hypothesis. A major problem in cluster validity is establishing the distributions of statistics under null hypotheses.

Random graph hypothesis:

H_0: All $n \times n$ rank order proximity matrices are equally likely.

Random label hypothesis:

H_0: All permutations of the labels on n objects are equally likely.

Random position hypothesis:

H_0: All sets of n locations in some region of a d-dimensional space are equally likely.

The selection of a null hypothesis depends on the type of data and the aspect of the data being tested. Several examples are presented in this chapter. Some details of the three null hypotheses are explained below.

The baseline, or reference, population for the *random graph hypothesis* is the set of all ordinal proximity matrices on n objects. An $n \times n$ ordinal proximity matrix, also called a rank-order proximity matrix, has $n(n - 1)/2$ entries, which can be taken to be the integers from 1 to $n(n - 1)/2$. Ties in proximity do not occur in this reference population. The random graph hypothesis states that all rank-order proximity matrices are equally likely (Ling, 1973a), so each distinct ordinal proximity matrix is assigned probability $1/[n(n - 1)/2]!$. Applications are discussed in Section 4.5.

The baseline population for the *random label hypothesis* is the set of $n!$ permutations of the labels on the n objects under study. The labels are usually the integers $(1, 2, \ldots, n)$. Each permutation has probability $1/n!$. This hypothesis establishes baseline distributions for statistics which match an observed proximity matrix to a structure imposed a priori, as discussed in Section 4.1.2, and can be applied with all data types.

The *random position hypothesis* requires that all sets of n points in some region of a d-dimensional space be equally likely and is appropriate for ratio data. For example, the region could be a hypercube having side 1 or a hypersphere of fixed radius. Another way of expressing the random position hypothesis is to require that each of the n points be inserted randomly into the region. With a hypercube, this implies that the coordinates of each point have uniform distributions over the edges of the hypercube. Yet another expression of the random position hypothesis is that the points are samples of a Poisson process, conditioned on the number of points, as discussed in Section 4.6. The idea behind the random position hypothesis is to specify a set of points with no structure in such a way that the meaning is clear and the distribution of statistics of interest can be either derived or estimated. Being able to generate samples of points on a computer under a hypothesis is of great value in Monte Carlo analysis.

The first major difference between the random graph and the random position hypotheses is that the former is for ordinal proximities and the latter is for proximities on a ratio scale. A second major difference involves dimensionality. When n points are restricted to a few dimensions and Euclidean distance is the measure of proximity, some rank orderings of distances cannot occur. For example, if points a, b, and c are placed on the real line in that order, the distance from a to c cannot be smaller than the distance from a to b. The triangle inequality prevents all orderings of distance from being realized. In fact, n points must be placed in a space of $(n - 2)$ dimensions to ensure that all rank orderings of the $n(n - 1)/2$ distances can be realized. Thus the random graph hypothesis is not a reasonable null population when the data occur as patterns, or points in d dimensions.

Definition of a test. Suppose that a statistic T and a null hypothesis H_0 have been agreed upon. Suppose further that the distribution of T is known under this null hypothesis. (We will see that finding this distribution is a very difficult task.) How to test whether hypothesis H_0 is an appropriate description for the data at hand? In cluster validity we want to determine if the fit of a hierarchy or a partition to the given data is unusually good. To be unusual, the fit must at least be better than the fit of a hierarchy or partition to a random data set. A mechanism for testing whether an observed statistic is unusual is now discussed.

Let $P(B \mid H_0)$ be the probability of event B under the null hypothesis. Event B could by "$T \geq t$" or "$T \leq t$," where t is a fixed number called a threshold. One must know whether a small value or a large value of T corresponds to good structure. For example, the smaller the square error, the better the clustering, but the ratio of between-cluster scatter to within-cluster scatter should be large for a good clustering. We now establish a test of the null hypothesis.

Let α be a small number, such as 0.05 or 0.01, called the *size* or *level* of a test. Given the distribution of T under H_0, and assuming that a large value of T indicates that H_0 should be rejected, we can place a threshold, t_α, on T by solving the following equation:

$$P(T \geq t_\alpha \mid H_0) = \alpha$$

Suppose that the value of T measured in an experiment is t^*. To answer the question: "Should H_0 be rejected?", which can be rephrased as "Is t^* large enough to call the fit unusual?", apply the following rule.

$$\text{If } t^* \geq t_\alpha, \text{ reject } H_0 \text{ at level } \alpha.$$

Thus α is the probability of deciding against H_0 when H_0 is actually true.

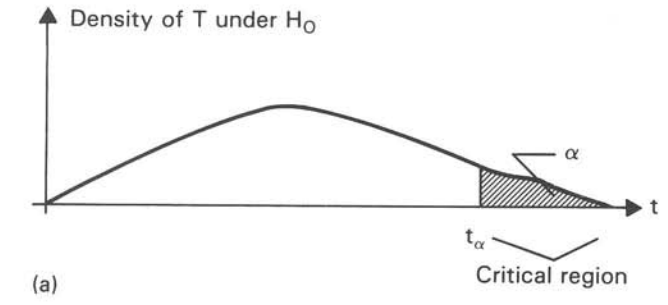
This test produces one of two answers; either T is large or it is not, so either the clustering structure is valid at level α or it is not. The *critical region* of the test is the set of values of the statistic that lead to rejection of H_0, or $\{t : t \geq t_\alpha\}$. This test requires that the null distribution of T be known.

Another way of assessing validity is to solve the following equation for α^*; recall that t^* is the value of T observed in an experiment.

$$P(T \geq t^*) = \alpha^*$$

Since H_0 would be rejected at level α^*, the value α^* is called the *critical level* for the test. The smaller α^*, the better evidence we have for deciding against H_0. We seek statistics having smooth distributions so that small changes in threshold do not have dramatic effects on decision making. The testing situation is pictured in Figure 4.1. When small values of T imply nonrandom structure, the test is to reject the null hypothesis when T is smaller than a threshold. In general, one can define any sort of critical region, but in cluster validity, statistics are chosen so that either large values or small values reflect structure.

Power of a test. The test of H_0 tells only half the story. What is missing is a hypothesis of structure, or an alternative hypothesis, H_1, to which H_0 can be compared. Alternative hypotheses establish specific structures, such as "the clusters are samples from a Gaussian mixture" or "the data contain two clusters." If the



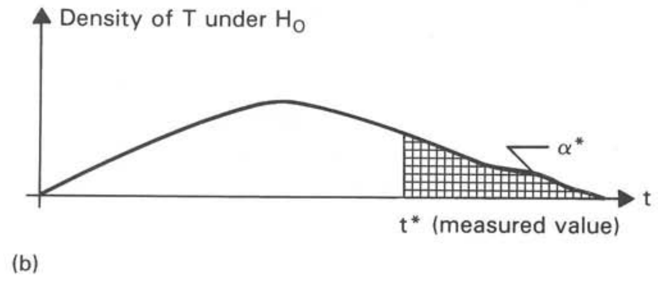


Figure 4.1 Hypothesis testing and critical level: (a) critical region for testing H$_O$; (b) critical level of significance α^*.

critical region of a test is $\{t : t \geq t_\alpha\}$, the *power* of the test is the probability of reaching the correct decision when H_1 is true, or

$$\text{power} = P(T \geq t_\alpha \mid H_1)$$

We seek statistics T that lead to tests having high power.

The concepts of size and power can be extended to critical regions other than the simple ones discussed so far. However, we assume that statistics have smooth distributions under H_0 and H_1, so critical regions $\{t : t \geq t_\alpha\}$ or $\{t : t \leq t_\alpha\}$ are ordinarily used. This assumption is justified by the fact that the distributions of T under the two hypotheses are not normally available. We do not intend to discuss decision theory, by which optimal critical regions can be selected and statistics can be compared. See Duda and Hart (1973) or Devijver and Kittler (1982) for decision-making strategies from a pattern recognition viewpoint.

In summary, our approach to cluster validity involves several steps. A null hypothesis expressing our idea of no structure must be defined with the data type in mind. A statistic, or index, sensitive to the presence of structure in the data must then be selected and the distribution of the statistic under the null hypothesis must be established. A threshold can then be found that defines how large is "large" for the statistic selected. The threshold establishes a formal test of hypothesis. The power of the test evaluates the ability of the statistic to recognize the presence of a structure specified in an alternative hypothesis. These steps will be illustrated throughout this chapter.

4.1.2 Hubert's Γ Statistics

One way of validating a clustering structure is to compare it to an a priori structure, which is assigned without regard to the measurements. For example, the cluster numbers assigned to objects by a clustering algorithm can be compared to category labels, assigned independent of the clustering. The statistic that has come to be known as Hubert's Γ has been shown to be effective in assessing fit between data and a priori structures. Hubert and Schultz (1976) provide a comprehensive explanation of the statistic and display an impressive list of applications. The idea behind this statistic is attributed to Mantel (1967), who was trying to identify subtle space-time clustering in the outbreaks of disease. Barton and David (1962) used statistics based on permutations that are similar to Mantel statistics when assessing the randomness of points in a plane.

The abstract problem to which Hubert's Γ is applicable can be stated as follows. Let $\mathcal{X} = [X(i, j)]$ and $\mathcal{Y} = [Y(i, j)]$ be two $n \times n$ proximity matrices on the same n objects. The matrices must contain data having no built-in or implied relationships. For example, $X(i, j)$ could denote the observed proximity between objects i and j and $Y(i, j)$ could be defined as

$$Y(i, j) = \begin{cases} 0 & \text{if objects } i \text{ and } j \text{ have the same category label} \\ 1 & \text{if not} \end{cases}$$

Other applications have $X(i, j)$ as the geographical separation between objects i and j, while $Y(i, j)$ is the separation of these objects in time, as when the objects are outbreaks of a disease. Still another application has $X(i, j)$ as the observed proximity between objects i and j and $Y(i, j)$ as the reconstructed proximity from some theoretical model.

The Hubert Γ statistic is, simply, the point serial correlation between the two matrices. It can be expressed in raw form as follows when the two matrices are symmetric:

$$\Gamma = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X(i, j)Y(i, j)$$

In normalized form, Γ is the sample correlation coefficient between the entries of the two matrices. If m_x and m_y denote the sample means and s_x and s_y denote the sample standard deviations of the entries of matrices $\mathcal{X}$ and $\mathcal{Y}$ (Section 2.3), the normalized Γ statistic is

$$\Gamma = \left\{(1/M) \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} [X(i, j) - m_x][Y(i, j) - m_y]\right\} / s_x s_y$$

where $M = n(n-1)/2$ is the number of entries in the double sum and the moments are given by

$$m_x = (1/M)\,\Sigma\,\Sigma\,X(i, j) \qquad m_y = (1/M)\,\Sigma\,\Sigma\,Y(i, j)$$

$$s_x^2 = (1/M)\,\Sigma\,\Sigma\,X^2(i, j) - m_x^2 \qquad s_y^2 = (1/M)\,\Sigma\,\Sigma\,Y^2(i, j) - m_y^2$$

All sums are over the set $\{(i, j) : 1 \le i \le n-1,\ i+1 \le j \le n\}$. When the matrices $\mathscr{X}$ and $\mathscr{Y}$ are not symmetric, all sums extend over the n^2 entries in the two matrices. We limit our treatment to symmetric matrices.

The Γ statistic measures the degree of linear correspondence between the entries of $\mathscr{X}$ and $\mathscr{Y}$. Unusually large absolute values of Γ suggest that the two matrices agree with each other. The normalized Γ is always between -1 and 1, while the range of the raw Γ depends on the ranges of values in the matrices and on the number of entries. The most common application of Γ tests whether the association between $\mathscr{X}$ and $\mathscr{Y}$ is unusually large under the random label hypothesis. That is, could the row and column numbers on one of the matrices have been inserted at random? The random label hypothesis can be stated as follows for this purpose.

H_0: All permutations of the row (and column) labels of $[Y(i, j)]$ are equally likely.

The permutation referenced in H_0 is a reordering of the object labels $\{1, 2, \ldots, n\}$. The reordering is applied to both the rows and the columns of $\mathscr{Y}$, so the entries of $\mathscr{Y}$ are rearranged. However, the entries of $\mathscr{Y}$ are not themselves rearranged randomly as in the random graph hypothesis, which specifies that all permutations of the entries of $\mathscr{Y}$ are equally likely. The intelligent application of tests based on Γ demand that the null hypothesis be understood in each case. Testing the random label hypothesis means asking whether the observed Γ could have been reasonably obtained by chance labeling.

One often tests whether the value of Γ is unusually large, but the absolute value is also used in some cases. No matter how the test of hypothesis is formulated, the distribution of Γ under H_0 must be known, estimated, or approximated to fix a threshold and establish how large is "large." For very small values of n, this distribution can be found by evaluating Γ for all $n!$ permutations of the row and column numbers of $\mathscr{Y}$ and accumulating them in a histogram.

Example 4.1

Matrices $\mathscr{X}$ and $\mathscr{Y}$ are given below.

$$\mathscr{X} = \begin{array}{c|cccc} & x_1 & x_2 & x_3 & x_4 \\ \hline x_1 & 0 & 1.2 & 0.6 & 0.2 \\ x_2 & - & 0 & 0.3 & 0.4 \\ x_3 & - & - & 0 & 0.1 \\ x_4 & - & - & - & 0 \end{array} \qquad \mathscr{Y} = \begin{array}{c|cccc} & x_1 & x_2 & x_3 & x_4 \\ \hline x_1 & 0 & 1 & 0 & 1 \\ x_2 & - & 0 & 1 & 0 \\ x_3 & - & - & 0 & 1 \\ x_4 & - & - & - & 0 \end{array}$$

Matrix $\mathscr{X}$ is a dissimilarity matrix for four objects and matrix $\mathscr{Y}$ is derived by placing objects x_1 and x_3 in one category and objects x_2 and x_4 in a second category, then assigning proximity 0 to pairs of objects in the same category and proximity 1 to object pairs in different categories. There are 24 permutations of the sequence numbers {1, 2, 3, 4} on the objects. The distribution of Γ under H_0 is found by computing

$$\sum_{i=1}^{3} \sum_{j=i+1}^{4} X(i, j)Y[g(i), g(j)]$$

for all 24 permutations $\{g(1), g(2), g(3), g(4)\}$ of {1, 2, 3, 4}. For example, the $\mathcal{Y}$ matrix for the permutation $\{g(1), g(2), g(3), g(4)\} = \{2, 4, 1, 3\}$ is given below.

$$[Y(g(i), g(j))] = \begin{array}{c|cccc} & x_2 & x_4 & x_1 & x_3 \\ \hline x_2 & 0 & 0 & 1 & 1 \\ x_4 & - & 0 & 1 & 1 \\ x_1 & - & - & 0 & 0 \\ x_3 & - & - & - & 0 \end{array}$$

The Γ value for this permutation is the sum of the four numbers in the upper right of the $\mathcal{X}$ matrix, or $\Gamma(g) = 1.5$. Eight other permutations lead to 1.5. In fact, only the three values shown in the following frequency table can occur.

Value of Γ	1.5	1.8	2.3
Frequency	8	8	8

The observed value of Γ, computed from $\mathcal{X}$ and the original $\mathcal{Y}$ matrix, is 1.8. Thus a value as large as the observed Γ is not at all unusual and we might conclude that the rows and columns of $\mathcal{Y}$ are randomly labeled.

Computational realities prevent the application of the procedure in Example 4.1 to practical problems. With 8 objects, $8! = 40{,}320$ values of Γ must be computed, and with 12 objects, over 470 million values must be found. It is clear that the distribution of Γ under H_0 must be approximated or estimated. The two standard approaches to this problem are Monte Carlo analysis and moment estimation.

Let $\{g(1), g(2), \ldots, g(n)\}$ denote a permutation of the integers $\{1, 2, \ldots, n\}$. For example, {6, 3, 1, 4, 2, 5} is a permutation of {1, 2, 3, 4, 5, 6}. The random variable whose distribution is required under H_0 can be written as

$$\Gamma(g) = \left\{(1/M) \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} [X(i, j) - m_x][Y[g(i), g(j)] - m_y]\right\} / s_x s_y$$

We could have applied permutation g to the entries of $\mathcal{X}$ instead of $\mathcal{Y}$. The sample means and standard deviations are invariant to permutations g.

A Monte Carlo analysis (Section 4.1.4) randomly selects a number of permutations g, and rank orders the resulting sequence $\{\Gamma(g)\}$ to estimate the distribution of Γ under H_0. This distribution can be visualized by a histogram, as in Example 4.2. The methodology of Section 4.1.4 is applied by deciding that the observed Γ is unusually large if it is larger than, say, 95% or 99% of the values in $\{\Gamma(g)\}$.

Example 4.2 ✓

This example demonstrates the estimation of the distribution of Γ under the random label hypothesis for two pattern matrices. The 80X data set, introduced in Chapter 2, contains 15 patterns in an eight-dimensional space from each of three categories for a total of 45 patterns. Suppose that we want to test whether the category labels are matched unusually well to the locations of the patterns in the eight-dimensional feature space. Matrix $\mathcal{X}$ is taken to be the matrix of Euclidean distances between the patterns and matrix $\mathcal{Y}$ contains a 1 in position (i, j) if the ith and jth patterns are in different categories and a 0 if they are in the same category. If the first 15 patterns are from category 8, the next 15 are from category O, and the last 15 are from category X, then $\mathcal{Y}$ can be blocked into nine 15 × 15 submatrices as indicated below, where *1* denotes a 15 × 15 matrix of 1's and *0* denotes a 15 × 15 matrix of 0's.

$$\mathcal{Y} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Only the 990 upper diagonal entries are used in the computation. A simple counting argument shows that

$$m_y = \frac{3 \times 15^2}{990} = \frac{675}{990} = 0.682 \qquad \text{and} \qquad s_y = \left[\frac{675}{990} - \left(\frac{675}{990}\right)^2\right]^{1/2} = 0.466$$

The mean and standard deviation of the Euclidean distances between patterns were found to be

$$m_x = 9.07 \qquad \text{and} \qquad s_x = 1.88$$

The gamma statistic can be written as follows for this example.

$$\Gamma(g) = \left\{(1/990)\sum_{i=1}^{44}\sum_{j=i+1}^{45} X[i,j]Y[g(i), g(j)] - (0.682)(9.07)\right\}/(0.466)(1.88)$$

Figure 4.2 shows a histogram of $\{\Gamma(g)\}$ for 100 permutations g of the 45 pattern labels. The 100 values of $\Gamma(g)$ were grouped into 40 bins and Figure 4.2 shows the counts in each bin. The observed value of Γ is 0.33087 and is the blip on the right of the histogram. It is clear that the observed value is significantly higher than all the values in the histogram, so we reject the random label hypothesis. That is, the evidence suggests a nonrandom assignment of category labels.

The same procedure was repeated for a set of 45 patterns chosen randomly inside an eight-dimensional hypercube having sides of length 12. The first 15 of the random patterns were assigned to the first category, the next 15 were arbitrarily assigned to the second category, and the last 15 to a third category to match the category numbers in the 80X data. Of course, the category labels have no meaning whatever. The histogram of $\Gamma(g)$ values for 100 permutations is shown in Figure 4.3. The observed value of Γ is −0.0014 and falls in the middle of the histogram. At least 42 of the 100 permutations produced larger values of Γ than the observed value, so we would reject H_0 only at level 0.42. Thus we strongly suspect that the category labels have no meaning.

A second way of assessing whether Γ is unusually large that avoids creating all $n!$ permutations is to compute the mean and variance of $\Gamma(g)$ under the random label hypothesis and assume that the underlying distribution is normal. Figures

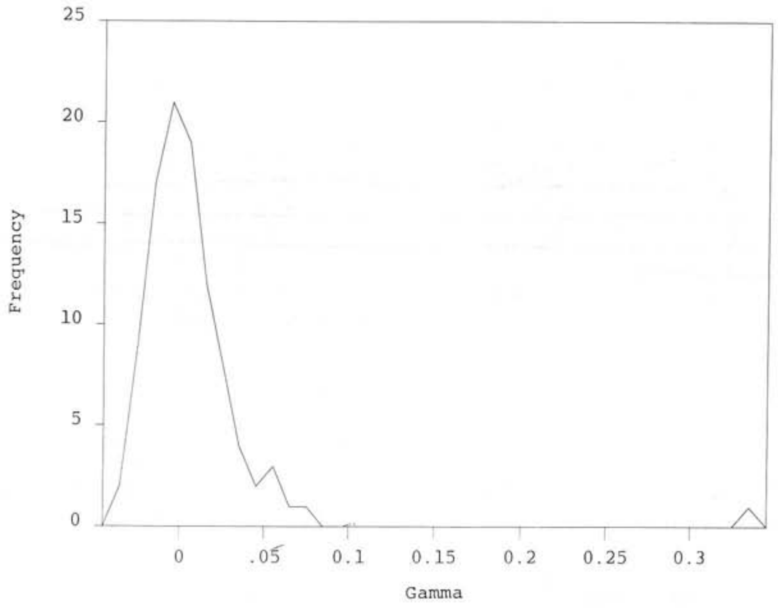


Figure 4.2 Histogram of Γ under the random label hypothesis for the 80X data with category labels.

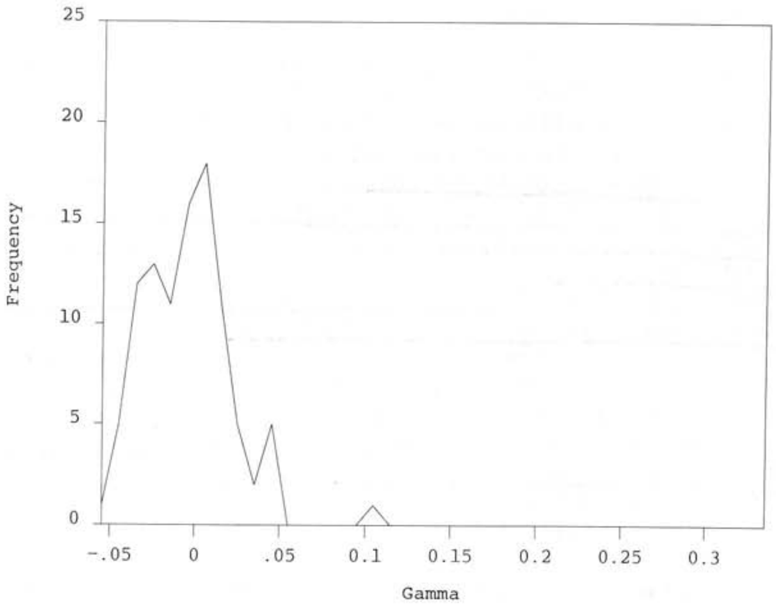


Figure 4.3 Histogram of Γ under the random label hypothesis for 45 random patterns in 8 dimensions with random category labels.

4.2 and 4.3 suggest that the assumption of normality is reasonable. The normalized statistic is

$$\Gamma' = \frac{\Gamma - E_0(\Gamma)}{\mathrm{SD}_0(\Gamma)}$$

where the expected value, E_0, and the standard deviation, SD_0, are computed under the random label hypothesis. Hubert and Schultz (1976) and Mantel (1967) provide explicit formulas for the two moments required and support the claim that Γ' is asymptotically normal. The formulas are quite complicated and demand a significant amount of computation, perhaps comparable to that required in a Monte Carlo analysis. Expressions for the moments will be provided in a few special cases later in this chapter.

Siemiatycki (1978) cites studies indicating that the asymptotic normality of Γ is not always an appropriate assumption and suggests computing the first four moments of the raw Γ and approximating the distribution under H_0 by Pearson curves. The computation of all these moments is not a simple task.

The Γ statistic has a wide range of applications. Hubert and Subkoviak (1979) use Γ to test whether a given proximity matrix supports an a priori labeling of the objects, as in Example 4.2. Hubert and Golledge (1981) examine five problems involving matching of matrices via correlation coefficients. Hubert (1979) devises a statistic based on Γ to assess the concordance among elements of K independent proximity matrices. Dietz (1983) compares two proximity matrices computed from different statistics with two versions of the Γ statistic and studies the performance of other nonparametric tests of association. Sokal (1979) tests the significance and nature of departure from randomness in patterns of geographic variation by Mantel statistics. Several other papers authored by Hubert that attack problems with Γ are listed in the bibliography. Specific applications in cluster validity are treated later in this chapter.

4.1.3 Goodman–Kruskal γ Statistic

The Goodman–Kruskal (1954) γ statistic measures rank correlation between two ordinal sequences of numbers. These numbers may be obtained from proximity matrices, as with Hubert's Γ statistic. The Γ statistic is more effective when at least one of the proximity matrices is on a quantitative scale. Here both sequences of numbers are assumed to be on an ordinal scale. For example, Hubert (1974b) suggests using γ to measure correspondence between a rank order proximity matrix and the cophenetic matrix. However, Milligan and Cooper (1985) apply γ with ratio data, as discussed in Section 4.4.

The idea behind this statistic can be seen from an abstract problem. Let $A = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_m\}$ be two ordinal sequences of m numbers. Ties can exist in one or both of the sequences. The γ statistic measures the degree of association between the two sequences in terms of the number of "concordant" pairs, denoted S_+, and the number of "discordant" pairs, denoted S_-. A "pair" is a set of two doublets, one doublet from A and one doublet from B, taken from corresponding positions in the two sequences. A doublet is a pair of numbers from one of the sequences. For example, (a_2, a_4) is a doublet and (b_1, b_2) is another doublet.

The pair of doublets $\{(a_i, a_j), (b_i, b_j)\}$ is *concordant* if either of the following conditions is satisfied.

$$(1)\ a_i < a_j \text{ and } b_i < b_j \qquad \text{or} \qquad (2)\ a_i > a_j \text{ and } b_i > b_j$$

The pair is *discordant* if either of the following is satisfied.

$$(3)\ a_i < a_j \text{ and } b_i > b_j \qquad \text{or} \qquad (4)\ a_i > a_j \text{ and } b_i < b_j$$

The pair of doublets is neither concordant nor discordant if $a_i = a_j$ or if $b_i = b_j$. For example, the pair of doublets $\{(3, 15), (8, 12)\}$ is concordant because $3 < 15$ and $8 < 12$. The pair $\{(3, 5), (8, 7)\}$ is discordant because $8 > 7$ whereas $3 < 5$. The pair $\{(3, 3), (5, 7)\}$ is neither concordant nor discordant because of the tie in the first doublet. The γ statistic is computed from the total numbers of concordant and discordant pairs as follows:

$$\gamma = \frac{S_+ - S_-}{S_+ + S_-}$$

Example 4.3

The six-position sequences of ranks $A = \{a_1, \ldots, a_6\}$ and $B = \{b_1, \ldots, b_6\}$ are given below. Note the tie in the first sequence.

	i					
	1	2	3	4	5	6
a_i	3	5	2	2	4	6
b_i	2	3	1	6	4	5

All 15 possible doublets defined by these sequences are listed in Table 4.1 along with their states of concordancy. State "+" means "concordant," state "−" means "discordant," and "*" means "neither concordant nor discordant."

Adding up the plus and minus signs in Table 4.1 shows that

$$S_+ = 9 \qquad S_- = 5 \qquad \gamma = 4/14$$

The computation can be simplified by rewriting the two sequences so that one of them is ordered. For example, the two sequences above can be written in the order of the first sequence as follows:

	i					
	4	3	1	5	2	6
a_i	2	2	3	4	5	6
b_i	6	1	2	4	3	5

With the sequences written in this form, one need only scan the $\{b_i\}$ sequence from left to right.

TABLE 4.1 Computation of γ Statistic

Doublet (i, j)	Ranks A	Ranks B	State	Doublet (i, j)	Ranks A	Ranks B	State
(1, 2)	(3, 5)	(2, 3)	+	(2, 6)	(5, 6)	(3, 5)	+
(1, 3)	(3, 2)	(2, 1)	+	(3, 4)	(2, 2)	(1, 6)	*
(1, 4)	(3, 2)	(2, 6)	−	(3, 5)	(2, 4)	(1, 4)	+
(1, 5)	(3, 4)	(2, 4)	+	(3, 6)	(2, 6)	(1, 5)	+
(1, 6)	(3, 6)	(2, 5)	+	(4, 5)	(2, 4)	(6, 4)	−
(2, 3)	(5, 2)	(3, 1)	+	(4, 6)	(2, 6)	(6, 5)	−
(2, 4)	(5, 2)	(3, 6)	−	(5, 6)	(4, 6)	(4, 5)	+
(2, 5)	(5, 4)	(3, 4)	−				

The γ statistic takes on values between -1 and $+1$ and is invariant to monotone transformations of either sequence. Values of γ near 1 suggest that one sequence increases when the other one does. Values near -1 suggest a strong correlation between the sequences, but one decreases when the other increases. Values near 0 indicate no relation between the patterns of increase and decrease in the sequences, in some unstated sense of ''relation.'' The actual distribution of γ depends on the null hypothesis of interest, the sample size, and the number of ties.

4.1.4 Monte Carlo Analysis

Monte Carlo analysis is a method for estimating parameters and probabilities by computer sampling when the quantities are difficult or impossible to calculate directly. The distributions of various indices of cluster validity depend on many problem-specific parameters and can be estimated only by Monte Carlo sampling. Hammersley and Handscomb (1965) provide a readable and compact introduction to the topic. Shreider (1964) is another general reference. To illustrate the general idea of Monte Carlo sampling, suppose that the integral

$$Q = \int_0^1 f(x)\, dx$$

is to be estimated, where $f(x)$ is known to be between 0 and 1 when x is between 0 and 1. Suppose that $f(x)$ can be computed for any x but that the integral is intractable. Evaluating the integral is equivalent to estimating the area under the function, as shown by the shaded region in Figure 4.4. Two approaches to estimating this area are called the *crude Monte Carlo* estimate and *binomial sampling*. The general paradigm for Monte Carlo estimation by binomial sampling, also called ''hit or miss'' sampling, is given below.

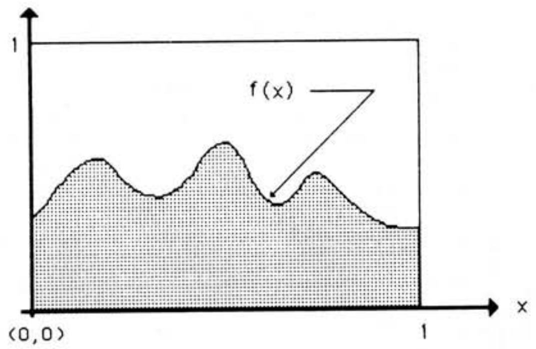


Figure 4.4 Evaluating an integral with Monte Carlo analysis.

ALGORITHM FOR MONTE CARLO ESTIMATION BY BINOMIAL SAMPLING

Step 1. Arrange to perform a computer sampling experiment in which the occurrence of an event is observed each time the experiment is performed and the event is related to the quantity being estimated.

Step 2. Repeat the experiment a large number of times, counting the number of experiments which end in "success," meaning that the event happens.

Step 3. Estimate the quantity of interest by the relative number of successes.

For the integration example, the sampling experiment is to select two statistically independent $U(0, 1)$ samples, which are numbers in the interval [0, 1] chosen so that all numbers are equally likely. These numbers locate a point in the unit square in Figure 4.4. A "success" occurs when the point falls in the shaded region. The relative number of successes estimates the value of the integral.

If X_i is a random variable representing the outcome of the ith experiment that is 1 when the event happens (success) and 0 when it does not, Monte Carlo sampling produces values of the binomial random variable Y.

$$Y = \sum_{i=1}^{m} X_i$$

Here m is the number of Monte Carlo trials and $\{X_1, \ldots, X_m\}$ are independent and identically distributed (i.i.d.) because the Monte Carlo trials are assumed to be conducted independently. Properties of the binomial distribution show that the mean of a binomial distribution is an unbiased estimator of the quantity, Q, being sampled. The expectations and probabilities are with respect to the sampling distribution, which is derived from the assumption of i.i.d. uniform samples. The mean and variance of Y are

$$E(Y) = Q \qquad \text{var}\,(Y) = \frac{Q(1 - Q)}{m}$$

Confidence bounds can be placed on the estimate Y (Conover, 1971). A 95% confidence interval, also called a 95% interval estimator, is the interval

$$[Y - c_m, Y + c_m]$$

where c_m is selected so that the probability of Q being within the 95% confidence interval is 0.95. Tables of c_m as a function of m are given in Conover (1971) for several values of m.

The second approach, the crude Monte Carlo estimate, is more difficult to set up than binomial sampling but has certain theoretical advantages. This approach is now demonstrated on the integration problem above. Let X be a $U(0, 1)$ random variable and define random variable $Y = f(X)$. Probability theory shows that

$$E(Y) = \int_0^1 f(x)\, dx$$

so the problem is to estimate the mean of Y. If $\{x_1, \ldots, x_m\}$ are samples of random variables distributed as $U(0, 1)$, the usual estimate of the mean is

$$\hat{Q} = (1/m) \sum_{i=1}^{m} f(x_i)$$

Estimator $\hat{Q}$ is an unbiased estimator of Q since its expected value is Q. Accuracy is measured by the *standard error*, which is the standard deviation of $\hat{Q}$, or $\sigma/\sqrt{m}$, where σ is the standard deviation of $f(X)$ and can be estimated by s.

$$s = \left[\frac{1}{m-1} \sum_{i=1}^{m} (f(x_i) - \hat{Q})^2\right]^{1/2}$$

The results of the Monte Carlo study would be reported by saying that Q is in the interval

$$[\hat{Q} - (s/\sqrt{m}), \hat{Q} + (s/\sqrt{m})]$$

Confidence intervals are one way of monitoring the accuracy of Monte Carlo estimates. The Chebychev inequality can be employed to construct a bound on the error $|\hat{Q} - Q|$ in terms of the variance of $\hat{Q}$. The normal approximation to the binomial distribution, through the central limit theorem, is a third possibility. The central limit theorem shows that $\hat{Q}$ has an asymptotic normal distribution with mean Q. Its standard deviation can be estimated by s. Thus we know that if m is large enough,

$$\text{Prob}\,\{|\hat{Q} - Q| \le 3s\} = 0.997$$

The *relative error* is defined as the actual error divided by the true value, or $|\hat{Q} - Q|/Q$. An asymptotic analysis shows that

$$\text{Prob}\left\{\frac{|\hat{Q} - Q|}{Q} \le \frac{3}{Q}\left[\frac{Q(1-Q)}{m}\right]^{1/2}\right\} = 0.997$$

so the bound on the relative error is $3[(1 - Q)/Qm]^{1/2}$. For small Q this is approximately $3/(Qm)^{1/2}$, so we cannot expect extremely accurate results when Q is small unless m is very large.

Monte Carlo analysis can approximate an unknown distribution if an experimental sampling procedure can be programmed on a computer that simulates the process being studied. This occurs when estimating the null, or baseline, distribution of an index for cluster validity. For example, the scatter ratio, or Wilks's lambda statistic S (Section 2.4.3), measures clustering but its distribution, even under randomness assumptions, is unknown because it depends on several problem parameters. A Monte Carlo analysis defines an experiment, such as sampling a random population in a hypercube in d dimensions, which embodies the notion of randomness. One can form several pattern matrices that match the given matrix in number of patterns, features, and spread of features. Each pattern matrix is clustered and the scatter ratio is recorded to develop an empirical distribution which serves as a baseline distribution.

Suppose that s_1 is the value of the index, such as the scatter ratio or Hubert's Γ statistic, measured on the data being clustered, and let $s_2, \ldots, s_m$ denote $m - 1$ values obtained from Monte Carlo sampling under some null hypothesis. How to determine if s_1 is "significantly" large, or small? Suppose that large values of the statistic are desired for "good" clustering. In the case of scatter ratio, small values would be desirable, but the methodology remains the same. To test whether s_1 is significantly large, select integer k so that $k/m = \alpha$, where α is the level of significance, such as 0.05 or 0.01. If s_1 is among the k largest of the m values $\{s_i\}$, the index is significantly large and the null hypothesis can be rejected at level α. The probability of incorrectly rejecting the null hypothesis is exactly α. One interesting application of this methodology is given by Besag and Diggle (1977), who apply it to the problem of recognizing nonrandom structure in spatial patterns of points. Foutz (1980) proposed a randomized version of this test that can increase power.

The Monte Carlo analysis blurs the critical region of the test, in comparison to when the null distribution is known. The threshold for rejecting the null hypothesis can be viewed as a range of values in the Monte Carlo method, instead of the single value that is obtained when the null distribution is known. Increasing m would reduce this blurring, but at a heavy computational cost. Let q be the actual probability that a sample from the true distribution of the s-value is greater than s_1. The probability that the Monte Carlo test would reject the null hypothesis is the probability that no more than $k - 1$ of the samples from this distribution exceed s_1, or

$$P(k, p) = \sum_{r=0}^{k-1} \binom{m-1}{r} p^{m-r-1} q^r$$

where $p = 1 - q$. Small values of this probability are desirable. Since $m = k/\alpha$, this probability is a function of k and p. Marriott (1979) examined $P(k, p)$ for k

from 1 to 10 and α of 0.05 and 0.01. If k is too small, say 1 or 2, there is a substantial probability of reaching a seriously wrong conclusion. For example, if $\alpha = 0.05$, $p = 0.99$, and $k = 1$, $P(k, p) = 0.826$, so the null hypothesis has a high probability of being rejected when it is true. Increasing k, and thus increasing m, decreases $P(k, p)$ but with a diminishing return for increased computing. Marriott agreed with earlier work that a k of about 5 is a reasonable compromise. For α of 0.05, this result suggests that 100 Monte Carlo trials are required, while for α of 0.01, it suggests at least 500 trials. If α is fixed, say at 0.05, then $k = \alpha m$, so $m = 20k$ for $\alpha = 0.05$.

The most difficult computational question in Monte Carlo analysis is arranging to sample from an arbitrary distribution. Random number generators are usually available for generating "pseudorandom" samples from $U(0, 1)$ distributions (Knuth, 1969; Fishman and Moore, 1982; Whitney, 1984) and techniques are available for translating these uniform samples into samples from other distributions (Dubes, 1968; Hammersley and Handscomb, 1965). A significant literature exists on algorithms for computer sampling. For example, Johnson et al. (1980) propose a new family of probability distributions appropriate for Monte Carlo studies. Deak (1980) examines computationally efficient ways of sampling the normal distribution, Atkinson (1979a,b) provides ways of sampling Poisson random variables, and Shore (1982) approximates quantities related to the normal distribution. Examples of Monte Carlo analysis are given throughout this chapter.

4.1.5 Bootstrapping

A Monte Carlo analysis requires that a computer experiment be set up to simulate the generation of data under some hypothesis, such as randomness. Bootstrap techniques are for situations when such an experiment cannot easily be arranged and all one has is the set of data itself. The general idea is to resample the data, with replacement, to create a set of "fake" data, which is used as if it were a replication of a Monte Carlo experiment.

At first glance, bootstrapping may appear to be a classy way of cheating. However, bootstrapping has been successfully applied to many problems in estimation and statistical inference since it was first proposed by Efron (1979, 1981, 1982, 1983). Diaconis and Efron (1983) provide a particularly readable introduction. Bootstrap techniques have not yet been applied to the problem of validating clustering structures, so no experience exists in this application. A sense of the bootstrapping approach taken can be gained from the following estimation problem.

Suppose that a parameter, θ, of the distribution of random variable X is to be estimated. A single sample $\mathscr{X} = \{x_1, x_2, \ldots, x_n\}$ is a set of i.i.d. random variables, all distributed as X, and represents the data in hand. If little is known about the distribution of X, and no further samples are available, even by Monte Carlo means, how can properties of estimators for θ be determined? The bootstrap technique is described as follows.

ALGORITHM FOR BOOTSTRAP PARAMETER ESTIMATION

Step 1. Define a distribution with cumulative distributive function $\hat{F}$ by assigning probability mass $(1/n)$ to each of the n data points in $\mathscr{X}$. This is the nonparametric maximum likelihood estimator of the distribution function for X.

Step 2. Draw a bootstrap sample $\mathscr{X}^* = \{x_1^*, x_2^*, \ldots, x_n^*\}$ from $\hat{F}$ by sampling $\mathscr{X}$ with replacement. Thus $\mathscr{X}^*$ contains samples of i.i.d. random variables distributed as $\hat{F}$ and some x_i may be repeated.

Step 3. Compute $\theta^* = \theta(\mathscr{X}^*, \hat{F})$, an estimate of θ from the bootstrap sample.

Step 4. Repeat steps 2 and 3 m times to obtain bootstrap replications θ_1^*, $\theta_2^*, \ldots, \theta_m^*$ and calculate properties of interest.

For example, the expected value of θ can be estimated by

$$(1/m) \sum_{k=1}^{m} \theta_k^*$$

The choice of m is not critical, but Efron (1983) suggests values between 100 and 200. The distribution of the bootstrap estimate converges to the true distribution of the estimator for several statistics (Bickel and Freedman, 1981; Singh, 1981) as n increases without bound. The bootstrap confidence interval for an estimator has been shown to have almost the same width as the usual confidence interval. Bootstrap techniques substitute computational power for theoretical analysis.

Other applications of bootstrapping include the estimation of bias in decision rules (Efron, 1983; Chernick et al., 1985; Jain et al., 1987), the significance of principal components (Diaconis and Efron, 1983), and the fit of regression lines (Efron and Gong, 1983). Bootstrapping has also been suggested in the problem of estimating density functions for spatial point processes (Silverman, 1984). The importance of bootstrapping in cluster validity has yet to be determined, but some work is appearing that uses bootstrapping (Moreau and Jain, 1986).

4.2 INDICES OF CLUSTER VALIDITY

An index for cluster validity measures the adequacy of a structure recovered through cluster analysis in terms that can be interpreted objectively, which we take to be probabilities. The adequacy of a clustering structure refers to the sense in which the clustering structure provides true information about the data, or the ability of the recovered structure to reflect the intrinsic character of the data. The three types of structures to which we refer are listed below.

Hierarchies are the nested sequences of partitions obtained by applying hierarchical clustering methods and model the global structure of the data (Section

3.2). Hierarchies can also be proposed independent of data, as from a theoretical model.

Partitions result from partitional clustering algorithms and impose equivalence relations on the data (Section 3.3). Partitions are also obtained from category information.

Clusters are individual subsets of patterns that can be defined by any means whatever, including cluster analysis, category information, and pure hunches.

The validity of a clustering structure can be expressed in terms of the three types of criteria named below.

External criteria measure performance by matching a clustering structure to a priori information. For example, an external criterion measures the degree of correspondence between cluster numbers, obtained from a clustering algorithm, and category labels, assigned a priori. An external criterion can also measure the degree to which data confirm a priori ideas without a formal cluster analysis being performed.

Internal criteria assess the fit between the structure and the data, using only the data themselves. For example, an internal criterion would measure the degree to which a partition, obtained from a clustering algorithm, is justified by the given proximity matrix.

Relative criteria decide which of two structures is better in some sense, such as being more stable or more appropriate for the data. For example, a relative criterion would measure quantitatively whether a single-link or a complete-link hierarchy fits the data better.

A criterion expresses the strategy by which a clustering structure is to be validated, while an index is a statistic in terms of which validity is to be tested. Tests of hypothesis use indices to determine if a recovered structure is appropriate for the data. In the case of external and internal criteria, this comes down to testing whether the value of the index is either unusually large or unusually small. This, in turn, requires that a baseline population be established, as discussed in Section 4.1.1. The same index can be used in an internal and an external criterion, although the baseline distributions of the index will be different for the two criteria. Various statistical tools are employed to define actual tests of cluster validity. Some of these tools, such as Monte Carlo analysis and bootstrapping, can be used across the board. Other statistics, such as Hubert's Γ and the Goodman–Kruskal γ, can be modified to work in individual situations. The main issues in using an index of cluster validity are summarized below.

Definition of index. The index should make good intuitive sense, should have a basis in theory, and should be readily computable.

Baseline distribution. A baseline distribution is a null distribution derived from a population containing no "structure." A reference population is defined or implied by the baseline distribution.

Test for nonrandom structure. An index of cluster validity is applied by comparing it to a threshold that establishes a given level of statistical significance. The threshold is defined from the baseline distribution, which is seldom known from theory. Monte Carlo analysis and asymptotic approximations are often required.

Test for specific structure. The statistical power of an index for cluster validity can be established by examining its ability to recover a known structure. The choice of structure depends on the particular application.

Other issues turn up when specifics are considered. The data type (qualitative or quantitative) is a particularly important consideration in choosing an index of validity. Each type of recovered structure is considered in a separate section.

Example 4.4 ✓

This example demonstrates the difference between using the Hubert Γ statistic in an internal and in an external criterion. Example 4.2 uses Γ as an external index because it assesses the degree to which category labels are justified. The category label 8, 0, or X is a characteristic of the object itself and has nothing to do with the manner in which the pattern is measured. The baseline distribution of Γ, illustrated in Figures 4.2, was generated in Example 4.2 by specifying the $\mathcal{Y}$ matrix from category information, then computing Γ for several permutations of the rows and columns of $\mathcal{Y}$.

Hubert's Γ is used as an internal criterion when the classification of the patterns is obtained from a cluster analysis of the data. The Γ value for the 80X data is obtained by cutting the complete-link dendrogram for the 80X data to obtain three clusters and defining the $\mathcal{Y}$ matrix as follows.

$$Y(i,j) = \begin{cases} 0 & \text{if patterns } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are in the same cluster} \\ 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are in different clusters} \end{cases}$$

The cluster-by-category table for the three-cluster complete-link clustering, including row and column sums, is given in Table 4.2.

TABLE 4.2 Cluster by Category Table for 80X Data from Three-Cluster Complete-Link Classification

		Category			
		8	O	X	
Cluster	1	13	15	4	32
	2	2	0	0	2
	3	0	0	11	11
		15	15	15	

Can the baseline distribution for Γ be approximated as in Example 4.2? Is the random label hypothesis reasonable when Γ is to be used as in an internal criterion? Suppose that we answer both questions in the affirmative and evaluate $\Gamma(g)$ for 100 permutations of the 45 row and column numbers of the $\mathcal{Y}$ matrix. The results are shown in Figure 4.5. The observed value of Γ is 0.567, which is substantially above all values $\{\Gamma(g)\}$. Can we conclude that the three-cluster complete-link clustering fits the data unusually well?

Before answering the last question, consider the effect of applying the same procedure to the random data set described in Example 4.2. Figure 4.3 shows the baseline distribution for 45 random patterns in eight dimensions when category labels are assigned randomly. We decided that the randomly applied category labels were, indeed, applied at random in Example 4.2. When we replace the random category labels by cluster labels obtained by applying the complete-link clustering method and cutting the dendrogram to obtain three clusters, Table 4.3 and the histogram of Γ in Figure 4.6 are obtained.

The observed Γ is 0.291 for the random data, which is twice as large as the largest $\{\Gamma(g)\}$ in Figure 4.6. Does this mean that the cluster labels fit the data unusually well? Certainly not. The data were generated randomly, so no valid classifications should exist. It may happen, by pure chance, that a three-cluster clustering fits well. However, repeating this procedure for any number of clusters produces similar results, so one suspects that something other than pure chance is at work. Thus the random label hypothesis is not appropriate with an internal index and Figure 4.5 cannot be used to decide that the three-cluster complete-link solution fits the data well.

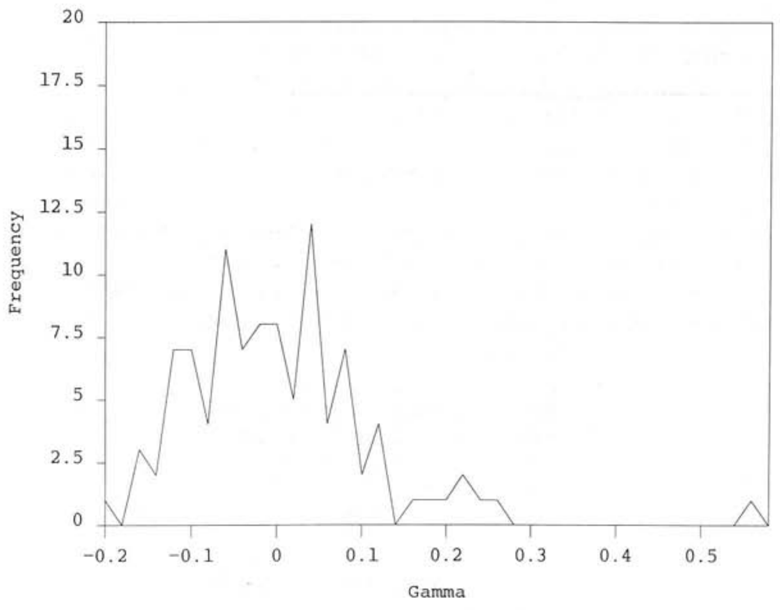


Figure 4.5 Histogram of Γ for 80X data under random label hypothesis from three-cluster complete-link classification.

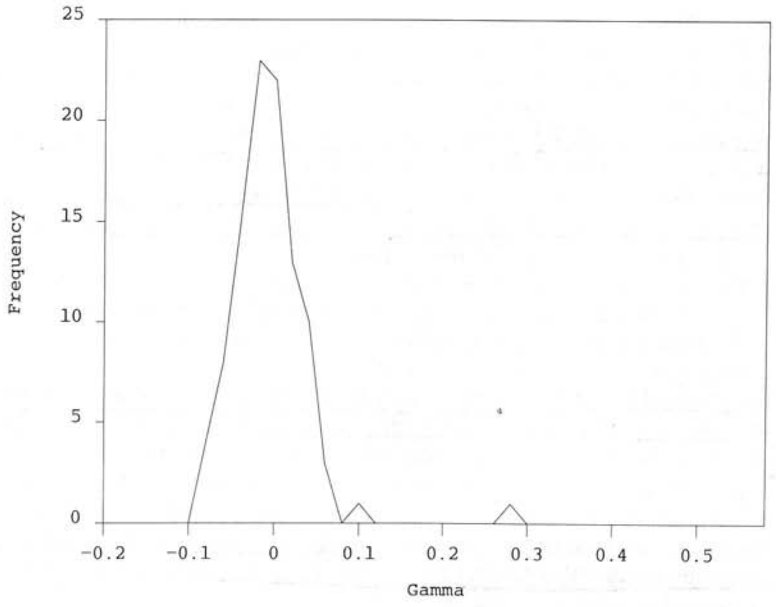


Figure 4.6 Histogram of Γ for random data under random label hypothesis when classification is obtained from cluster analysis.

Figure 4.6 illustrates that the degree to which clusters derived from cluster analysis match the clusters in the data cannot be verified under the random label hypothesis. Of course, the cluster labels fit better than random labels. The cluster analysis tries hard to do a good job, even with random data. Needed is a null population and a null hypothesis that is appropriate to the problem at hand. The choice of a baseline population and the manner of generating the baseline distribution are the primary differences between an internal and an external index.

We now suggest a procedure for generating a baseline distribution appropriate for an internal criterion to be used with the 80X data. The general idea is to create a large number of random matrices under the random position hypothesis, determine the three-cluster complete-link solution for each, and compute the corresponding Γ values.

TABLE 4.3 Cluster by Category Table for random Data from Three-Cluster Complete-Link Classification

		Category			
		1	12	13	
Cluster	1	10	6	6	22
	2	1	3	5	9
	3	4	6	4	14
		15	15	15	

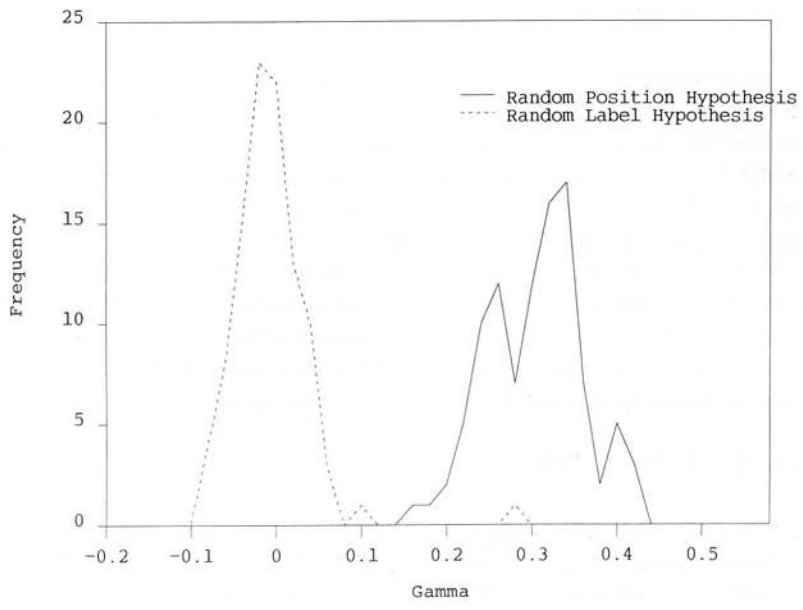


Figure 4.7 Histogram of Γ for 45 random patterns under random position and random label hypotheses from three-cluster complete-link classification.

ALGORITHM FOR BASELINE DISTRIBUTION OF INTERNAL INDEX

Step 1. Generate a set of 45 random vectors in an eight-dimensional hypercube and form the matrix $\mathscr{X}$ of interpoint distances.

Step 2. Cluster the vectors by the complete-link method and cut the dendrogram to form three clusters.

Step 3. Form the $\mathscr{Y}$ matrix from the three-cluster solution as explained earlier and compute Γ between $\mathscr{Y}$ and $\mathscr{X}$.

Step 4. Repeat steps 1 to 3 100 times and record the Γ values in a histogram.

The results of applying this algorithm are shown in Figure 4.7. The histogram of Γ values has shifted up by about 0.33 when compared to the histogram under the random label hypothesis. An internal criterion can now be established. The 80X data produced an internal index value of 0.567, while random data produced one of 0.291 when evaluated for the three-cluster complete-link solution. The scale of Figure 4.7 shows that 0.567 is significantly large but 0.291 is not unusually large; it is in the middle of the histogram. Thus the three-cluster complete-link solution does fit the 80X data unusually well but does not fit random data unusually well.

4.3 VALIDITY OF HIERARCHICAL STRUCTURES

How well does a hierarchical structure, such as that imposed by a hierarchical clustering method (Section 3.2), fit a given pattern matrix? This section reviews some measures of global fit for hierarchies.

4.3.1 External Indices

The problem here is to determine if a hierarchy computed for a given data set matches an expected hierarchy. The expected hierarchy can be expressed as a type of dissimilarity matrix in which entry (i, j) is k if objects i and j are expected to first be in the same cluster at level k of the hierarchy. Such a matrix can be compared to the cophenetic matrix determined from a hierarchical clustering by Hubert's Γ statistic or the Goodman–Kruskal γ statistic, depending on the data type. As discussed in Section 4.1.2, the null distributions for these statistics depend on a number of factors, especially the type of reference population, the number of objects, and the type of hierarchical clustering used. This particular problem in validating clustering structures has not received a great deal of attention because an expected hierarchy is not often available.

4.3.2 Internal Indices

Does a hierarchy obtained from a hierarchical clustering method fit the data from which it was derived unusually well? Should one be confident in the results of a hierarchical clustering, or not? The index used for answering these questions depends on the data scale. The cophenetic correlation coefficient (CPCC) has been proposed for quantitative data, and measures of rank correlation have been used with quantitative data.

In the notation of Chapter 3, $d(i, j)$ is the given proximity between objects i and j and $d_C(i, j)$ is the cophenetic proximity, or the level in the dendrogram for a particular clustering method at which objects i and j are first placed in the same cluster. Both proximities are assumed to be on ratio or interval scales. The CPCC is the product-moment correlation coefficient between the entries of these two $n \times n$ matrices and is defined below. Only the $M = n(n - 1)/2$ entries above the main diagonals are used because the two matrices are symmetric.

$$\text{CPCC} = \frac{(1/M)\,\Sigma\, d(i,j)d_C(i,j) - (m_D m_C)}{[(1/M)\,\Sigma\, d^2(i,j) - m_D]^{1/2}\,[(1/M)\,\Sigma\, d_C^2(i,j) - m_C]^{1/2}}$$

where $m_D = (1/M)\,\Sigma\, d(i, j)$, $m_C = (1/M)\,\Sigma\, d_C(i, j)$, and all sums are over the set

$$\{(i,j) : 1 \le i < j \le n\}$$

The value of CPCC is between -1 and 1; the closer CPCC to 1, the better the match and the better the hierarchy fits the data. But how close is "close"? The cophenetic matrix is an ultrametric matrix, so a perfect fit demands that all triples from the given proximity matrix satisfy the ultrametric inequality. This is seldom the case in practice since the proximity matrix would need to contain large numbers of ties, as discussed in Section 3.2.3. The CPCC depends on all problem parameters and on the clustering method employed. Rohlf and Fisher (1968) studied the CPCC by Monte Carlo analysis when patterns were randomly chosen from uniform and Gaussian distributions and were clustered by the UPGMA method. The average CPCC tended to decrease with n, the number of patterns, and to be independent of the number of features. However, very few Monte Carlo trials were used. This study also showed that CPCC is more sensitive to the choice of proximity measure than to the underlying distribution of the patterns. Rohlf and Fisher exhibited approximate 5% thresholds for testing the null hypothesis of a single Gaussian cluster against the alternative of a system of nested clusters using CPCC as a test statistic. Rohlf (1970) found that the UPGMA method seemed to produce consistently high values of CPCC and warned that "even a CPCC near 0.9 does not guarantee that the (dendrogram) serves as a sufficiently good summary of the phenetic relationship."

The CPCC has been widely used, especially in numerical taxonomy. For example, Farris (1969) proposed a clustering method that optimized CPCC but warned against using CPCC in this capacity. The major difficulty with CPCC in validating hierarchies is that its distribution depends on so many factors that one is forced into a Monte Carlo analysis to establish a baseline distribution for each particular problem. When the patterns occur as points in a d-dimensional space, one might choose the n patterns randomly, cluster them by a particular hierarchical method, record the CPCC, and repeat the process a large number of times. This Monte Carlo method would develop a baseline distribution for CPCC. A CPCC that is large on the scale of such a baseline distribution indicates that the hierarchy fits the data better than hierarchies fit random data.

A rank correlation can measure the match between proximity and cophenetic matrix when the proximities are on an ordinal scale. Hubert (1974b) proposed the Goodman–Kruskal γ statistic (Section 4.1.3) for this purpose. Cunningham and Ogilvie (1972) suggest another measure of rank correlation called Kendall's τ statistic for comparing rank matrices.

$$\tau = \frac{S_+ - S_-}{n(n-1)/2}$$

where S_+ and S_- are the numbers of concordant and discordant pairs of objects, respectively, as defined in Section 4.1.3. Thus τ and γ differ only in the denominator.

As with CPCC, one sequence of numbers used to compute γ consists of the $n(n-1)/2$ proximities given and the other consists of the cophenetic proximities obtained from a clustering method. The γ statistic has been used more for comparing clustering methods than for objectively validating the global fit of a hierarchy (Baker, 1974; Hubert, 1974b). Only when γ is 1 can one conclude that a dendrogram truly fits the proximity matrix. The baseline distribution of γ, like the distribution of CPCC, depends on all the problem parameters. The random graph hypothesis can be used with ordinal proximities in a Monte Carlo analysis. Hubert (1974b) created tables of percentage points for γ under the random graph hypothesis with single-link and complete-link clustering for n between 4 and 16 from the following algorithm.

ALGORITHM FOR BASELINE DISTRIBUTION OF γ UNDER RANDOM GRAPH HYPOTHESIS

Step 1. For fixed n (number of objects) form a dissimilarity matrix under the random graph hypothesis; that is, fill in the $n(n - 1)/2$ entries with a randomly chosen permutation of the integers from 1 to $n(n - 1)/2$.

Step 2. Cluster the n objects by a clustering method, such as the single-link or complete-link method.

Step 3. Form the cophenetic matrices for the dendrogram resulting from the clustering method.

Step 4. Compute γ between the dissimilarity and cophenetic matrices.

Repeat steps 1 to 4 on a Monte Carlo basis to create a baseline distribution for γ specific to the clustering method and value of n. Hubert (1974b) used 1,000 trials for each value of n.

This algorithm for generating a baseline distribution is clumsy because it must be repeated for each possible value of n. Hubert conjectured that the statistic

$$n\gamma - a \ln (n)$$

has an approximate standardized normal, or $N(0, 1)$, distribution with $a = 1.8$ for complete-link clustering and $a = 1.1$ for single-link clustering. If the conjecture is correct, Monte Carlo analysis is not needed and a single distribution can be used for all n.

Example 4.5

This example demonstrates the use of γ in measuring the degree to which two hierarchies fit the ordinal proximity matrix given below on five objects.

$$\begin{array}{c c} & \begin{array}{cccc} 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 4 & 6 & 1 & 5 \\ — & 8 & 9 & 3 \\ — & — & 7 & 10 \\ — & — & — & 2 \end{bmatrix} \end{array}$$

Figure 4.8 shows the single-link and complete-link threshold dendrograms for this dissimilarity matrix. Two γ statistics can be found for the two dendrograms showing the rank correlation between the given proximity matrix and the cophenetic matrices. Note that γ is invariant to transformations on proximity that do not change the order of the proximities so that either a threshold dendrogram, as in Figure 4.8, or a proximity dendrogram will give the same γ value. Table 4.4 shows the proximity ranks and the partition ranks for the two dendrograms in a form that makes it easy to find γ. Recall that the "partition" rank for object pair (i, j) is the first level in a dendrogram at which objects i and j are in the same cluster.

Table 3 in Hubert (1974b) shows that the 70th percentile for the single-link baseline distribution is 0.76 and the 50th percentile for the complete-link baseline distribution is

$$\begin{array}{c|cccc} & 2 & 3 & 4 & 5 \\ \hline 1 & 4 & 6 & 1 & 5 \\ 2 & & 8 & 9 & 3 \\ 3 & & & 7 & 10 \\ 4 & & & & 2 \end{array}$$

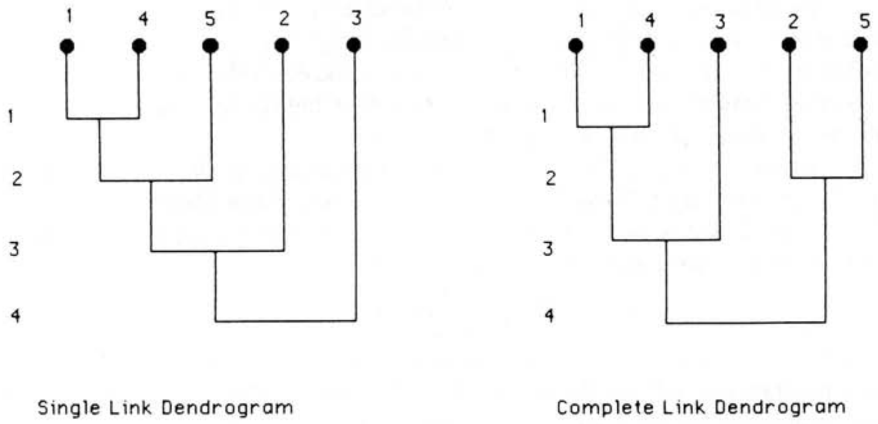


Figure 4.8 Single-link and complete-link dendrograms for the ordinal proximity matrix in Example 4.5.

TABLE 4.4 Proximity and Partition Ranks for Computation of γ in Example 4.5

		Partition Ranks	
Object Pair	Proximity	SL	CL
(1, 4)	1	1	1
(4, 5)	2	2	4
(2, 5)	3	3	2
(1, 2)	4	3	4
(1, 5)	5	2	4
(1, 3)	6	4	3
(3, 4)	7	4	3
(2, 3)	8	4	4
(2, 4)	9	3	4
(3, 5)	10	4	4
	S_+	30	22
	S_-	5	7
	γ	0.714	0.517

0.72 when n is 5. If we are testing the fit of the single-link and complete-link hierarchies to the given proximity matrix, we could conclude that the single-link method provides a better fit than the complete-link method but that neither method fits the data unusually well. The proximity matrix itself may be judged to have little hierarchical structure based on these results.

4.3.3 Relative Indices

In addition to the problem of validating a specific hierarchical clustering, one is often faced with the problem of deciding which of two hierarchical clustering solutions fits the given data better. This is separate from the problem of whether either of them fits the data unusually well. The indices are the same as those discussed above, but they are applied differently.

Baker (1974) provided a prototype for comparative studies on ordinal data. He began with three ''basal'' taxonomies, or dendrograms shown in Figure 4.9 for $n = 16$. Baker perturbed the ranks, R_{old}, obtained from the cophenetic matrices, with noise and generated the following proximities.

$$R_{\text{new}} = R_{\text{old}}(1 + KU)$$

where U is a sample of a uniform random variable over the range $(-1, 1)$ and K is a constant that reflects the magnitude of the noise perturbation. Rank ordering the resulting proximities produces the perturbed, ordinal proximity matrix.

One would expect the single-link method to pick out proximity matrices obtained by perturbing the chained taxonomy in Figure 4.9(a), and the complete-link method to be sensitive to the binary taxonomy in Figure 4.9(b). The arbitrary taxonomy [Figure 4.9(c)] should provide an intermediate basis for comparison. A number of proximity matrices were generated by the perturbation procedure. Each was clustered by single- and complete-link methods and a γ was computed between each pair of proximity and corresponding cophenetic matrices.

The procedure was repeated 100 times for each basal taxonomy and for three values of K (0.9, 1.0, 1.1). Thus three basal taxonomies and three noise levels were examined. The mean γ for the complete-link method was better than that for the single-link method in all situations, with the lone exception of a chained basal taxonomy at low noise (K of 0.9). The standard deviations of γ did not exhibit a consistent pattern. The fact that the complete-link method provided a larger mean γ even under chaining is counterintuitive. Hubert (1974b) extended Baker's study and provided more evidence for the superiority of the complete-link method over the single-link method in practical situations with ordinal proximities.

Fowlkes and Mallows (1983a) suggested comparing two hierarchical clusterings by computing an external criterion for each partition of the patterns that appears in the hierarchy and plotting this criterion as a function of the number of clusters. Unfortunately, they tried to apply their procedure as if it were an internal criterion. Such a procedure is interesting, but we are left with the problem of establishing

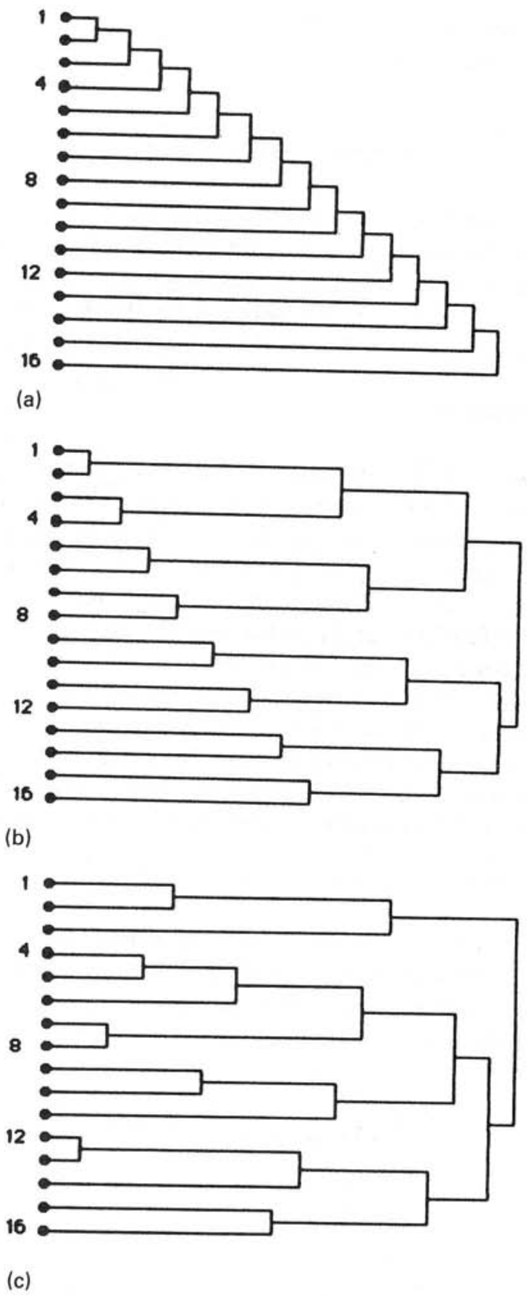


Figure 4.9 Basal taxonomies: (a) chained taxonomy; (b) binary taxonomy; (c) arbitrary taxonomy.

the behavior of such a plot under a baseline distribution. This, in turn, depends on several problem parameters.

4.4 VALIDITY OF PARTITIONAL STRUCTURES

The validation of a partition is the most commonly encountered of all the validation problems. This problem includes questions such as: "How many clusters are in the data?," "Does the partition match the categories?," "Where should the dendrogram be cut?," and "Which of two partitions fits the data better?" This section examines several indices for attacking these questions in a quantitative manner.

4.4.1 External Indices

An external index of partitional adequacy assesses the degree to which two partitions of n objects agree. One partition comes from a clustering solution, such as from a partitional clustering algorithm or from cutting the dendrogram generated by a hierarchical clustering algorithm. The second partition is assigned a priori, independent of the data and the first partition, as from category labels. Hubert and Arabie (1985) carefully define several indices for comparing two partitions. When the partitions are "independent," in the sense that one depends on the data values and the other does not, the distributions of some of these indices can be established from theory. All such indices are derived from the contingency table constructed from the two partitions in Table 4.5. The two partitions of the n objects are denoted $\mathcal{U}$ and $\mathcal{V}$. The partitions are identified with the clustering algorithm and a priori information when a specific index is defined.

$$\mathcal{U} = \{u_1, u_2, . . . , u_R\} \qquad \text{and} \qquad \mathcal{V} = \{v_1, v_2, . . . v_C\}$$

Entry n_{ij} in Table 4.5 is the number of objects that are both in group u_i and in group v_j. The term $n_{i.}$ is the row sum for the ith row, or the number of objects in group u_i and $n_{.j}$ is the number of objects in group v_j.

External indices of partitional adequacy can be expressed in terms of this

TABLE 4.5 Contingency Table for Two Partitions

	v_1	v_2	$\cdots$	v_C	
u_1	n_{11}	n_{12}	$\cdots$	n_{1C}	$n_{1.}$
u_2	n_{21}	n_{22}	$\cdots$	n_{2C}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
u_R	n_{R1}	n_{R2}	$\cdots$	n_{RC}	$n_{R.}$
	$n_{.1}$	$n_{.2}$		$n_{.C}$	$n_{..} = n$

contingency table. They can also be expressed in terms of the following indicator functions to show their similarity to measures of proximity for binary data (Section 2.2). The n objects are denoted $\{x_1, x_2, \ldots, x_n\}$

$$I_{\mathcal{U}}(i, j) = \begin{cases} 1 & \text{if } x_i \in u_r \text{ and } x_j \in u_r, \text{ some } 1 \le r \le R \\ 0 & \text{otherwise} \end{cases}$$

$$I_{\mathcal{V}}(i, j) = \begin{cases} 1 & \text{if } x_i \in v_s \text{ and } x_j \in v_s, \text{ some } 1 \le s \le C \\ 0 & \text{otherwise} \end{cases}$$

If partition $\mathcal{U}$ is obtained from a clustering solution and partition $\mathcal{V}$ is obtained from category labels, then $I_{\mathcal{U}}(i, j)$ is 1 when objects x_i and x_j are in the same cluster and 0 if they are in different clusters. Similarly, $I_{\mathcal{V}}(r, s)$ is 1 if the two objects have the same category label and 0 if not. The contingency table for the two indicator functions given in Table 4.6 lists the numbers of ways in which the pairs of patterns are treated by the two partitions.

TABLE 4.6 Contingency Table for Indicator Functions

		$I_{\mathcal{V}}$		
		1	0	
$I_{\mathcal{U}}$	1	a	b	m_1
	0	c	d	$M - m_1$
		m_2	$M - m_2$	

Thus a is the number of pairs of objects that are in the same group in both partitions, d the number of pairs that are in different groups in both partitions, and b the number of pairs of objects that are in the same group in $\mathcal{U}$ but in different groups in $\mathcal{V}$. The number of pairs of objects that are in the same group in $\mathcal{U}$ is denoted $m_1 = a + b$, while $m_2 = a + c$ is the number of pairs in the same group in $\mathcal{V}$. The total number of pairs of objects is

$$M = a + b + c + d = \frac{n(n-1)}{2}$$

The entries in Tables 4.5 and 4.6 are related as shown below. These equations can be obtained by straightforward combinatorial considerations. To simplify the notation, we use Σ_i to denote $\Sigma_{i=1}^{R}$ and Σ_j to denote $\Sigma_{j=1}^{C}$.

$$a = \sum_i \sum_j \binom{n_{ij}}{2} = (1/2) \sum_i \sum_j n_{ij}^2 - (n/2)$$

$$b = \sum_j \binom{n_{.j}}{2} - \sum_i \sum_j \binom{n_{ij}}{2} = (1/2) \sum_j n_{.j}^2 - (1/2) \sum_i \sum_j n_{ij}^2$$

$$c = \sum_i \binom{n_{i.}}{2} - \sum_i \sum_j \binom{n_{ij}}{2} = (1/2) \sum_j n_{i.}^2 - (1/2) \sum_i \sum_j n_{ij}^2$$

$$d = \binom{n}{2} - a - b - c = \frac{n(n+1)}{2} - \frac{1}{2}\left[\sum_i n_{i.}^2 + \sum_j n_{.j}^2\right]$$

If the marginal sums in Table 4.5 are fixed, which means that the sizes of the groups in the two partitions are fixed, the four entries in Table 4.6 are all linear functions of

$$Z = \sum_i \sum_j n_{ij}^2$$

so they are linear functions of one another. Table 4.7 lists some common external indices for comparing two partitions.

Note that the Rand statistic, the Fowlkes and Mallows statistic, and the Γ statistic are all linear functions of Z and thus are linear functions of one another when the row and column sums are fixed in Table 4.5. The Jaccard statistic is not a linear function of Z. Large values of the three statistics imply close agreement between the two partitions. The Γ statistic is a correlation, so its value is between -1 and 1. The Rand and Jaccard statistics have values between 0 and 1. The maximum value of 1 may not be achievable when the two partitions have different numbers of clusters.

The Γ statistic in Table 4.7 is a special case of the Hubert Γ statistic introduced in Section 4.1.2. The notation of Section 4.1.2 can be identified here as follows:

$$X(i,j) = I_{\mathcal{U}}(i,j)$$

$$Y(i,j) = I_{\mathcal{V}}(i,j)$$

$$m_x = (1/M)\,\Sigma\,\Sigma\,X(i,j) = m_1/M \qquad m_y = m_2/M$$

$$s_x^2 = (1/M)\,\Sigma\,\Sigma\,X^2(i,j) - m_x^2 = m_1/M - (m_1/M)^2 \qquad s_y^2 = m_2/M - (m_2/M)^2$$

$$(1/M)\,\Sigma\,\Sigma\,X(i,j)Y(i,j) = a/M$$

$$\Gamma = \frac{[a/M - (m_1/M)(m_2/M)]}{\sqrt{[(m_1/M - (m_1/M)^2][m_2/M - (m_2/M)^2]}} = \frac{Ma - m_1 m_2}{\sqrt{m_1 m_2 (M - m_1)(M - m_2)}}$$

The key step in applying these external indices of partitional adequacy is the formation of a baseline, or reference distribution. A clustering can then be

TABLE 4.7 External Indices of Partitional Adequacy

Name	Formula
Rand (1971)	$(a+d)/\binom{n}{2} = 1 + \left[Z - (1/2)\left(\sum_i n_{i.}^2 + \sum_j n_{.j}^2\right)\right] \Big/ \binom{n}{2}$
Jaccard	$a/(a+b+c) = (Z-n)/\left(\sum_i n_{i.}^2 + \sum_j n_{.j}^2 - Z - n\right)$
Fowlkes and Mallows (1983a)	$a/(m_1 m_2)^{1/2} = (1/2)(Z-n) \Big/ \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right]^{1/2}$
Γ statistic	$(Ma - m_1 m_2)/[m_1 m_2 (M-m_1)(M-m_2)]^{1/2}$

termed "valid" if it has an unusually high value, as measured with respect to the baseline distribution. The baseline distribution can also be used to correct an index for randomness. Hubert and Arabie (1985) suggest a baseline distribution in which the row and column sums are fixed in Table 4.5 but the partitions are selected at random, so the hypergeometric distribution can be applied to determine the expected values of quantities such as

$$E\left[\sum_i \sum_j \binom{n_{ij}}{2}\right] = (1/2)E[Z - n] = \sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2} \Big/ \binom{n}{2}$$

The expected values of the Rand, Fowlkes and Mallows, and Γ statistics can be obtained from this result. Correcting a statistic for chance means to normalize the statistic so that its value is 0 when the partitions are selected by chance and 1 when a perfect match is achieved. If S is a statistic, the form of the corrected statistic is

$$S' = \frac{S - E(S)}{\text{Max} - E(S)}$$

where Max is the maximum value of the statistic. Hubert and Arabie (1985) assume that the maximum value of the Rand statistic is 1 and propose the following "corrected" Rand statistic. Other corrections have been suggested in the literature but some have been based on incorrect mean values.

$$R' = \frac{\left[(a + d)\Big/\binom{n}{2}\right] - E\left[(a + d)\Big/\binom{n}{2}\right]}{1 - E\left[(a + d)\Big/\binom{n}{2}\right]}$$

$$= \frac{\sum_i \sum_j \binom{n_{ij}}{2} - \left[1\Big/\binom{n}{2}\right] \sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}}{(1/2)\left[\sum_j \binom{n_{.j}}{2} + \sum_i \binom{n_{i.}}{2}\right] - \left[1\Big/\binom{n}{2}\right] \sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}}$$

Corrections for other indices can be found in a similar manner. The Jaccard coefficient is not a linear function of Z, so its mean does not have a simple form. Fowlkes and Mallows (1983a) give the mean and variance of their statistic under the baseline distribution. Although the mean values of these indices are known, the distributions themselves are not known. Thus no theoretical thresholds exist for deciding when the indices are unusually large.

The null distribution of Γ is usually taken to be the distribution under the random label hypothesis. The cluster sizes are fixed. Hubert and Schultz (1976) provide the mean and variance of Γ under this baseline distribution. A threshold for "unusually large" can be approximated by assuming that the baseline distribution is normal and using the estimates of the mean and variance, but precise thresholds can only be found by Monte Carlo methods.

Milligan et al. (1983) compared the Rand, a corrected Rand, the Jaccard, and the Fowlkes and Mallows statistics as external indices of partitional adequacy. They suggest selecting either the Rand or corrected Rand and either the Fowlkes and Mallows or the Jaccard statistics. Their correction to the Rand statistic is not the same as that given above.

The external indices defined here can also be used to judge the relative merits of two partitions in recovering structure from the data and to test whether the data themselves confirm a prior hypothesis, independent of any clustering. For example, one can ask whether the proximity matrix itself, as opposed to a partition of the objects, corresponds to a set of category labels. The data are not clustered. Hubert and Subkoviak (1979) provide an example.

Example 4.6

It is often necessary to generate data having a specified number of clusters, as when two indices for estimating the number of clusters are being compared in a Monte Carlo study. The algorithm in Appendix H generates clustered data by choosing a cluster center at random and dispersing patterns around each cluster center according to a Gaussian distribution with a specified spread parameter. Provisions are made to ignore potential cluster centers that overlap existing clusters. This example, drawn from Dubes (1986), demonstrates the use of external indices of partitional adequacy in verifying that data generated by the algorithm in Appendix H are, indeed, clustered. This is not a trivial exercise. Large spread parameters could generate data which, for all practical purposes, are random with no well-defined clusters.

This example is concerned with data sets containing 100 patterns generated in five-dimensional hypercubes. Random data refer to sets of patterns chosen independently and uniformly from five-dimensional hypercubes. A spread parameter of 0.1 was used for clustered data and either two or eight clusters were required with at least 40 and 10 patterns in each cluster, respectively. The baseline distributions of the Jaccard and the corrected Rand indices were developed as follows. Table 4.8 summarizes the results.

ALGORITHM FOR BASELINE DISTRIBUTION OF EXTERNAL INDEX

Step 1. Generate a set of 100 patterns in five dimensions. Category labels for clustered data are taken as true cluster labels. Category labels for random data are assigned randomly to match the category sizes used with clustered data.

Step 2. Cluster the pattern matrix by the single-link and complete-link methods and cut the dendrograms at the "correct" levels of two and eight clusters to obtain cluster labels.

Step 3. Compute the Jaccard and the corrected Rand indices between the partition obtained from category information and that from cluster analysis for each clustering method.

Repeat 100 times for each situation.

Table 4.8 demonstrates clearly the need for correcting an index for randomness. The mean Jaccard index becomes almost 0.5 for purely random data, while the corrected Rand index maintains a mean close to 0 under all experimental conditions. The mean values of both indices are much larger under clustering than under randomness. Considering

TABLE 4.8 Comparison of External Indices for Partitions

	Jaccard		Corrected Rand	
k^a	Mean	Std. Dev.	Mean	Std. Dev.
		Clustered data		
SL				
2	0.934	0.169	0.870	0.336
8	0.597	0.172	0.683	0.169
CL				
2	0.989	0.043	0.988	0.050
8	0.859	0.116	0.908	0.082
		Random data		
SL				
2	0.496	0.007	0.00027	0.005
8	0.118	0.004	0.00053	0.004
CL				
2	0.354	0.029	−0.00069	0.017
8	0.068	0.008	−0.00102	0.015

Note: [a] k is the true number of clusters for clustered data and the number of pseudo clusters for random data.

the relatively small sample standard deviations, it is clear that the clustered data do, indeed, contain clusters. However, this conclusion can be drawn more readily with the corrected Rand coefficient than with the Jaccard coefficient because one would need to create a baseline distribution for the Jaccard index in each case. It appears that the corrected Rand coefficient maintains a zero value with a small standard deviation under randomness, independent of experimental conditions such as the number of clusters and the clustering method. Note that the complete-link method produces consistently larger index values than the single-link method.

4.4.2 Internal Indices

Internal indices of partitional adequacy measure the degree to which a clustering obtained from a clustering algorithm is justified in light only of the pattern or proximity matrix. In other words, they measure the fit between the partition imposed by a clustering algorithm and the data themselves. Category labels and other a priori information are not used in internal indices. Dubes and Jain (1979), Milligan (1981), and Milligan and Cooper (1985) provide definitions and evaluations of several internal indices.

The partitional adequacy problem includes the problem of estimating the "true" number of clusters, which has been called the fundamental problem of cluster validity (Duda and Hart, 1973; Everitt, 1979). The question "How many clusters in the data?" can be asked in at least three meaningful ways. One might ask: "Do the data contain the number of clusters I expect?" An external index (Section 4.4.1) is appropriate for answering this question when an a priori partition is available. A second posing of the question is: ''Is it unusual to achieve an external index as small (or as large) as the one I have just computed?'' For example, the partition may have come from one of the square-error clustering algorithms in Chapter 3, and one would like to know if the value of square-error obtained is ''significantly'' small, or small enough so that the clustering is ''valid.'' Two major difficulties arise in attempting to quantify the meanings of ''significant'' and ''valid'' that are serious enough to make one doubt that this question can ever be answered satisfactorily. The difficulties are the definition of baseline distribution and the dependence of indices on problem parameters and are discussed in this section. A third way of asking the question is: ''Which of two clusterings is better?'' Section 4.4.3 treats this question.

Consider first the definition of a baseline distribution. A Monte Carlo analysis will develop a baseline distribution of square-error, or of any other internal index, if a suitable model for the data is available. Choosing such a model is not a trivial task. One could simply generate data randomly over some sampling window, such as a hypercube or hypersphere, but what sampling window is appropriate? A more serious difficulty is inherent in the fact that data with any number of clusters will probably yield a smaller square-error than random data. Thus, imposing a clustering solution with four clusters on data having six true clusters will result in a smaller square-error than will imposing a four-cluster solution on random data. Comparing the square-error for the four-cluster solution to the baseline distribution could then lead one to the erroneous conclusion that the four-cluster solution is valid. A way out of this conundrum is to employ internal indices as relative indices, as discussed in Section 4.4.3.

The second difficulty with internal indices is their dependence on problem parameters, such as the numbers of patterns, features, and clusters, and the spread of the data. Square-error, for example, naturally decreases as the number of clusters increases and increases as the number of patterns and the number of features increase. We exhibit this dependence in the analysis of 16 artificial data sets which were generated to demonstrate the point. Each consists of four Gaussian, symmetric, reasonably separated clusters of approximately the same size in a unit hypercube (see Appendix H). Dimensionalities of 2, 4, 6, and 8 and sample sizes of 50, 100, 200, and 400 for each dimensionality were chosen to establish the 16 data sets. Four random data sets in six dimensions with the four sample sizes given were also generated. In an attempt to reduce the effects of sample size, we used the average error, or the square root of the square-error divided by the sample size, as a criterion. The 20 data sets (16 clustered and 4 random) were analyzed by the CLUSTER program (Section 3.3.3) and solutions containing from one to eight clusters were obtained for each of the 20 data sets. Figure 4.10 shows eigenvector projections of three clustered data sets in six dimensions for sample sizes of 50(c), 100(d), and 200(a) and of the random data set in six dimensions (b).

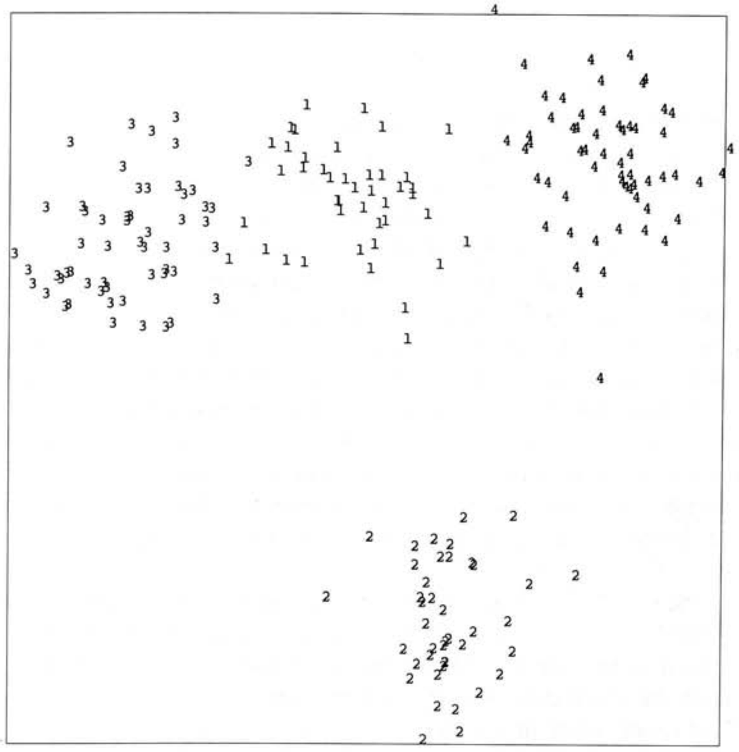
Figure 4.11 demonstrates the effects of dimensionality and number of clusters on average error when sample size is fixed. The curves of average error for the eight-dimensional data sets show definite knees at four clusters, the "correct" number. Only with 400 patterns does a sharp knee occur at four clusters in four and six dimensions. The two-dimensional data sets do not exhibit knees at the correct number of clusters with any of the data sets. Spurious knees can be seen in a few cases, especially the one at three clusters for six-dimensional data with 200 patterns. The random data sets exhibit no knees whatever. The same data are plotted in Figure 4.12 for each of the dimensionalities. It is clear that only the eight-dimensional data exhibit the correct number of clusters at all sample sizes. Intuitively, the eight-dimensional data contain more information about clustering than comparable data sets in fewer dimensions and eight dimensions provide enough space for the clusters to form. We do not hazard a guess as to why this effect occurs at eight dimensions rather than at a lower number of dimensions. This simple experiment suggests that the effects of dimensionality and sample size are severe, even for idealized data that are well matched to the clustering algorithm.

If one is willing to risk the difficulties discussed above, a baseline distribution can be established by Monte Carlo means and one can perform a test of hypothesis, as discussed in Section 4.1. This requires generating $m - 1$ sets of random data that match the given data in number of patterns, number of features, and overall spread. If small values of the index denote good clusterings, as with square-error, the null hypothesis of randomness is rejected at significance level r/m if the index observed is among the r smallest values achieved in the Monte Carlo trials. This approach is discussed in Section 4.2. The use of random data is crucial. For example, McClain and Rao (1974) overlooked this point in their test of hypothesis, which resulted in a misleading test (Milligan, 1981).

Another way of determining whether an index is unusually small is to know the theoretical distribution of the index under some reasonable hypothesis of randomness. Although actual distributions are known only in trivial cases, Bock (1985) has derived asymptotic distributions of some indices. It is never clear that an asymptotic distribution is appropriate in a particular situation. One must be extremely cautious in applying derived and asymptotic distributions to evaluate internal indices. For example, the distributions of several statistics which clustering algorithms optimize, such as ratios of within to between scatter, have been derived (Wilks, 1963; Cooley and Lohnes, 1971). But these distributions require that the labels on the patterns be category labels, which means that the labels are assigned without regard to the data themselves. Theoretical distributions of within-cluster and between-cluster measures are useful in analysis of variance, but not in cluster analysis.

Although comparing an internal index to a baseline distribution is not advisable when validating a partition, this procedure is appropriate when trying to determine if a given set of data is better described with one or two clusters. This can be useful in deciding whether or not to split a cluster. Engelman and Hartigan (1969) obtained the distribution of the between-cluster scatter to the within-cluster scatter (Section 3.3.1) under a null hypothesis of a single Gaussian distribution in one dimension. The alternative hypothesis contained two clusters. This distribution

(a)



(b)

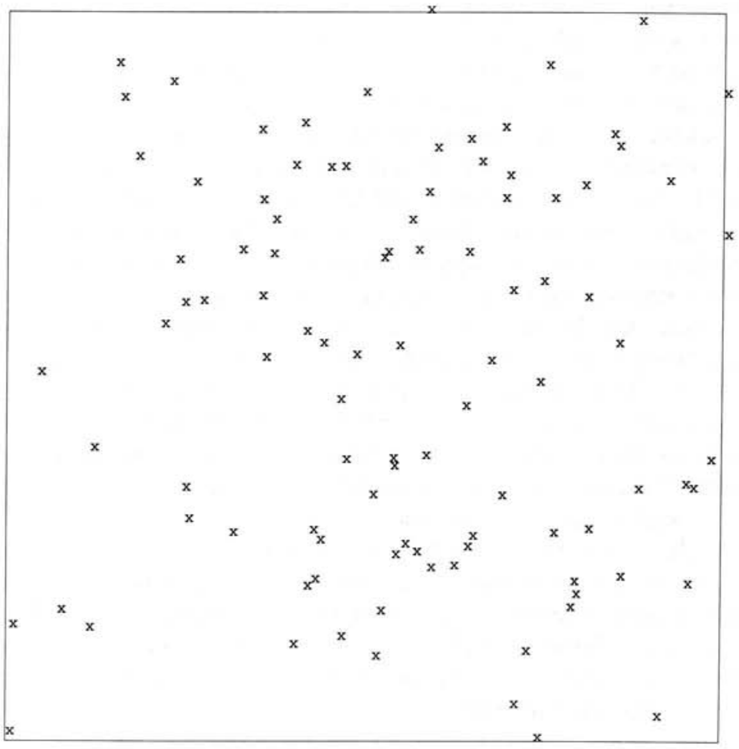
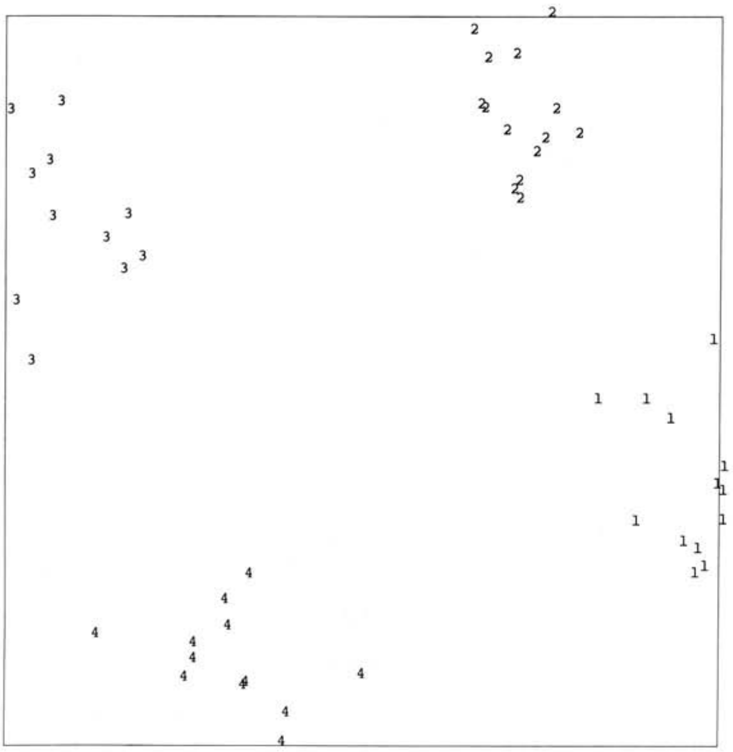


Figure 4.10 Eigenvector projections of six-dimensional data sets.

(c)



(d)

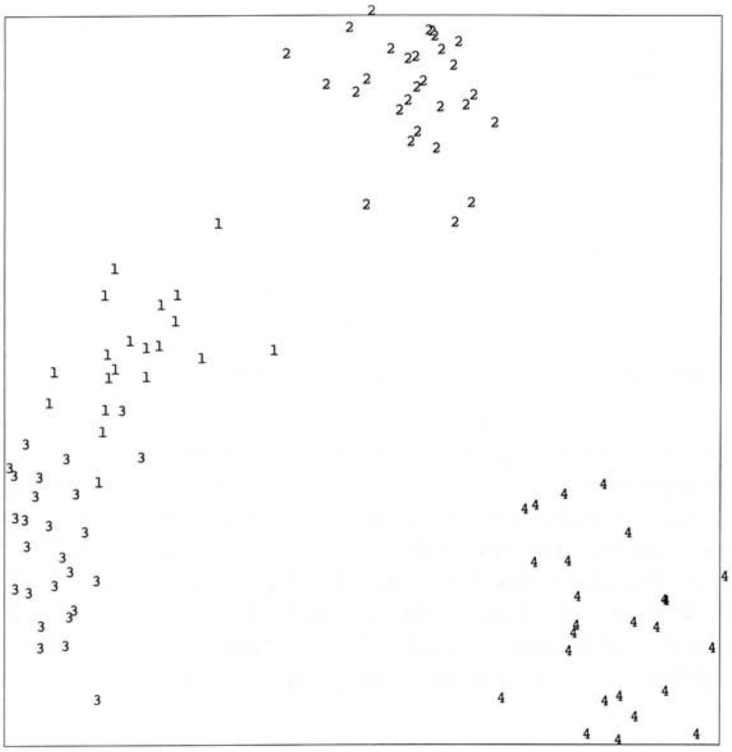
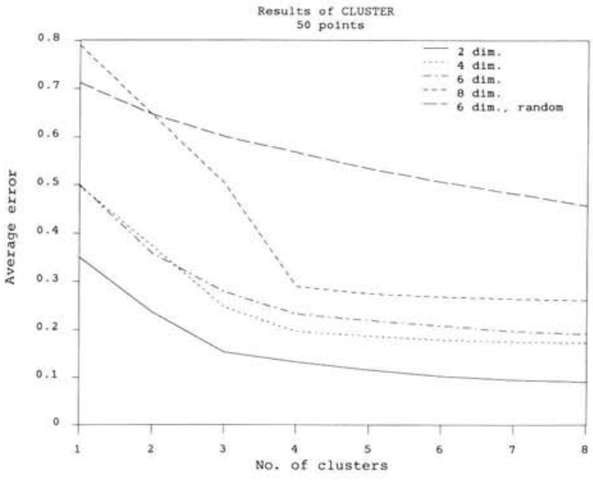
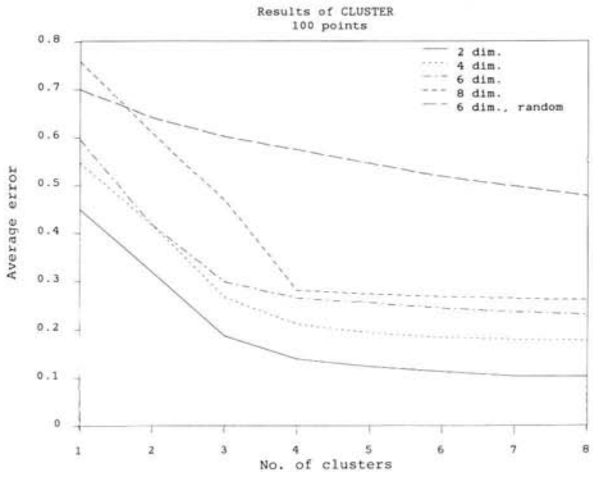


Figure 4.10 *(continued)*



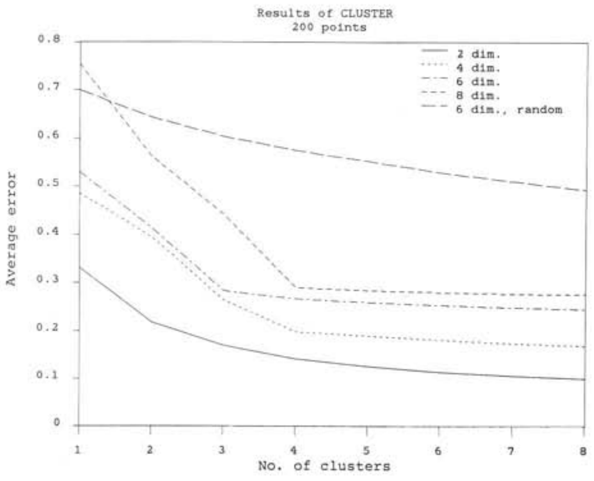
(a)



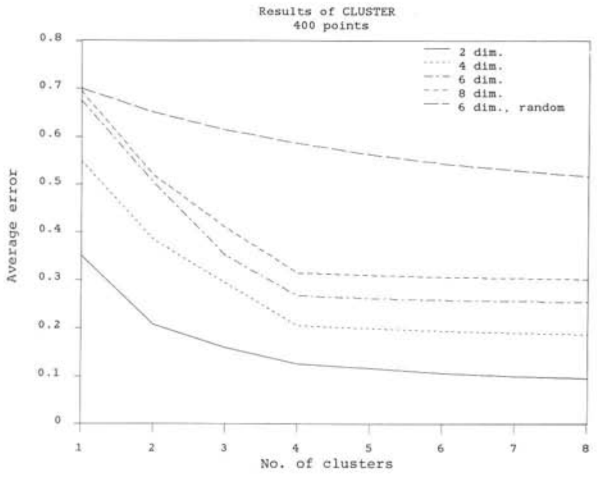
(b)

Figure 4.11 Average error as a function of number of clusters for each sample size.

can be realistically obtained only in one dimension, where only $(n - 1)$ clusterings need be evaluated for each Monte Carlo trial. The huge number of possible clusterings (Section 3.3) makes developing the null distribution impractical in more than one dimension. Hartigan (1975) shows that the log of the scatter ratio has an asymptotic Gaussian distribution under the null hypothesis. Bock (1985) has proposed some extensions of this ''gap'' test to higher dimensions, but their utility has yet to be demonstrated. Sneath (1977) proposed another gap test based on scatter ratios of data projected to the line joining the centers of two clusters.



(c)

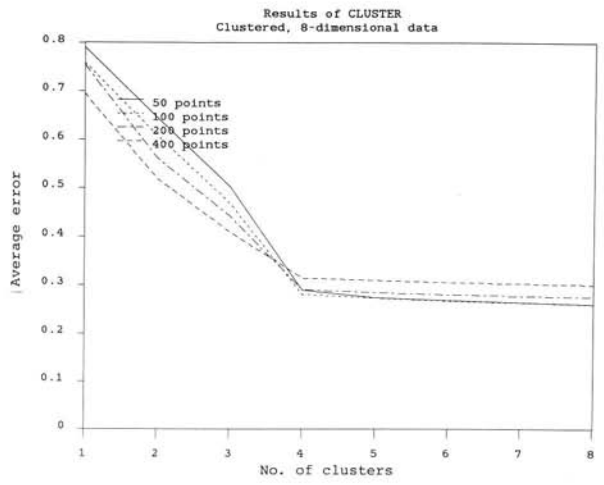


(d)

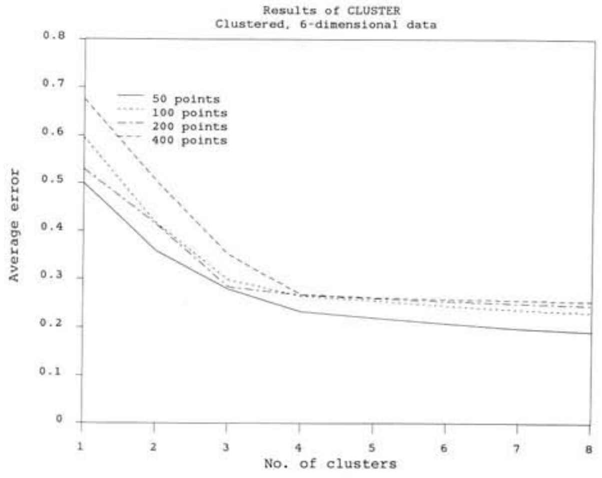
Figure 4.11 *(continued)*

4.4.3 Relative Indices

Which of a set of clusterings best matches the data? A relative index of partitional adequacy should answer this question in a quantitative way. All of the internal indices could serve as relative indices. A relative index is distinguished by the manner in which it is applied. For example, one might develop a procedure for recognizing a "significant" knee in curves, such as those in Figures 4.11



(a)

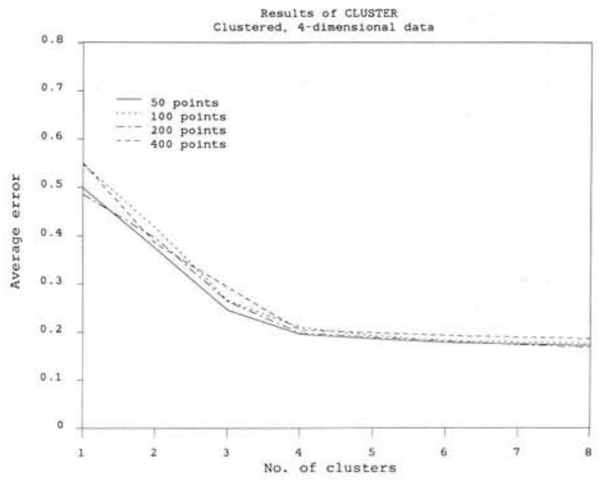


(b)

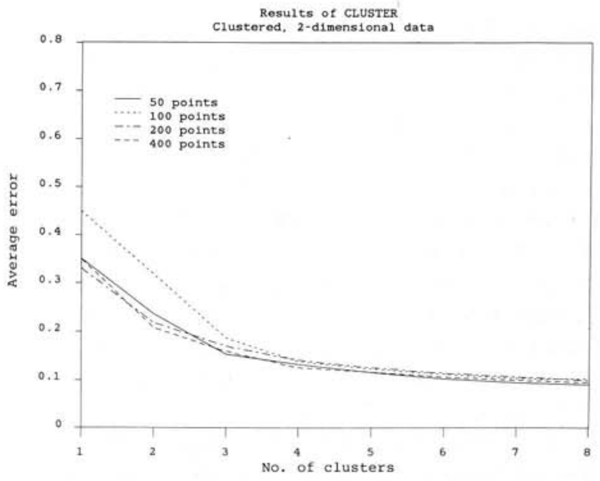
Figure 4.12 Average error as a function of sample size for each dimensionality.

and 4.12. The sequence of clusterings can be obtained from repeated applications of a nonhierarchical clustering algorithm, such as CLUSTER, but are usually obtained from cutting a dendrogram at successive levels. A procedure for choosing the best level for cutting a dendrogram is called a stopping rule.

A relative index is easier to apply than an internal index because a sequence of values is obtained, one for each clustering in the sequence. One looks for some unusual aspect of the sequence, such as a maximum, minimum, or significant



(c)



(d)

Figure 4.12 *(continued)*

knee. Milligan and Cooper (1985) provide a comparative study of 30 internal indices used in stopping rules. Several of the indices are based on Hubert's Γ (Section 4.1.2). Two specific indices are explained below to demonstrate the application of relative indices. We are not promoting either index. Few results are available that permit one confidently to choose a relative criterion.

The Davies–Bouldin (1979) index was originally proposed as a way of deciding when to stop clustering data. The index is plotted against the number of clusters and clustering is stopped when the index is minimized. It was claimed that the index does not depend on either the number of clusters or the clustering method. Given a partition of the n objects into K clusters, one first defines the following measure of within-to-between cluster spread for all pairs of clusters (j, k) in the notation of Section 3.3.1.

$$R_{j,k} = \frac{e_j + e_k}{m_{j,k}}$$

where e_j is the average error for the jth cluster and $m_{j,k}$ is the Euclidean distance between the centers of the jth and kth clusters. The index for the kth cluster is

$$R_k = \max_{j \neq k}\{R_{j,k}\}$$

and the Davies–Bouldin index for the K-cluster clustering is

$$\mathrm{DB}(K) = (1/K) \sum_{k=1}^{K} R_k \qquad \text{for } K > 1$$

The smaller DB(K), the better the clustering. This index is identically zero for the trivial clustering that places each object in an individual cluster and should be computed only when each cluster contains a reasonable number of objects. It is not defined for the case when all objects are in the same cluster. One plots DB(K) against K for successive values of K, as when a dendrogram is cut at successive levels, and chooses that partition that minimizes the index. The case when the data contain exactly two clusters should produce a curve whose only minimum occurs at $K = 2$, the lowest possible value of K. A heuristic procedure must be adoped to distinguish this from the case of no clusters, in which DB(K) may exhibit the same behavior. Figure 4.13 illustrates the behavior of DB(K) for well-clustered data (σ of 0.01) and poorly clustered data (σ of 0.1), both containing four clusters generated by the algorithm in Appendix H, and also for random data. All data sets contained 100 objects, generated as points in a five-dimensional hypercube. The data were clustered by the complete-link method. Figure 4.13 demonstrates the wild behavior of DB(K) for random data. The index is supposed to decrease monotonically as K decreases until the "correct" number of clusters is achieved for well-clustered data.

A second index is a modification of Hubert's Γ (Section 4.1.2) defined as the point serial correlation coefficient between the proximity matrix, assumed to be the Euclidean distance between n patterns in a d-dimensional pattern space, and a "model" matrix. The model matrix in Section 4.1.2 is an $n \times n$ matrix with 0 in position (i, j) if objects i and j are in the same cluster and 1 if not. Here the Euclidean distance between the centers of the clusters to which the objects belong are used instead of 0 and 1. The intuitive idea is that cluster centers estimate the "true" positions of the clusters in the pattern space and that deviations from the centers are due to errors and distortions.

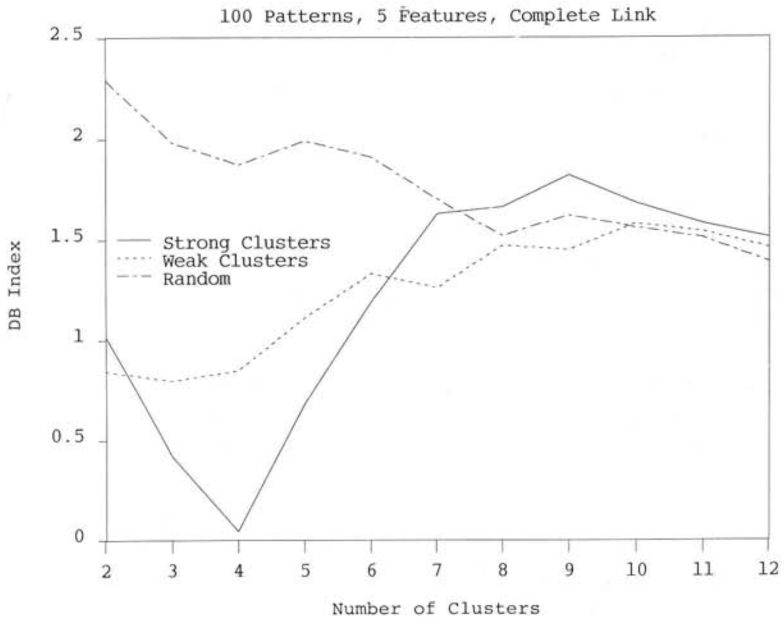


Figure 4.13 Davies–Bouldin relative index for strongly and weakly clustered data containing four clusters and for random data.

The new statistic MH (modified Hubert Γ) can be defined in the notation of Section 4.1.2 as follows. A clustering is defined by the label function:

$$L(i) = k \qquad \text{if object } i \text{ is in the } k\text{th cluster}$$

If $m_{j,k}$ is the Euclidean distance between cluster centers j and k, then

$$Y(i, j) = m_{L(i),L(j)}$$

The index MH(K) is computed by the formula for Γ in Section 4.1.2. This index decreases monotonically as K decreases, so one must seek a significant knee in a plot of MH(K) against K. This index will be 1 for the trivial clustering in which each object is in an individual cluster and is not defined for the clustering in which all objects are in the same cluster. Figure 4.14 provides examples of this index for cases similar to those in Figure 4.13. Again, the data contain four true clusters. The main difference between well-clustered and weakly clustered data is in the level of the curve for a large number of clusters. The knee at four clusters is evident in the strongly clustered data but shifts to three clusters in the weakly clustered data. Separating data with two clusters from random data, or data with one cluster, can be accomplished by observing the slope of the curve. The plot of MH(K) as K increases tends to be flat when the data contain clusters but tends to have a positive slope for random data.

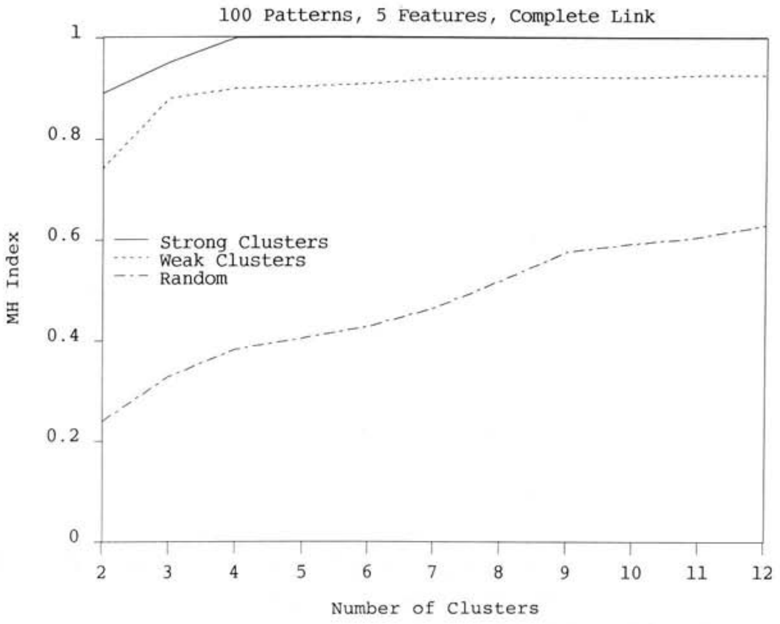


Figure 4.14 MH relative index for strongly and weakly clustered data containing four clusters and for random data.

A preliminary Monte Carlo study suggests that MH is a more reliable way to identify the true number of clusters than DB. However, MH requires some judgment as to the definition of a significant knee, whereas DB requires only that a minimum be identified. The Hubert Γ ranked third best in Milligan (1981) and seventh best in Milligan and Cooper (1985), both out of 30 indices. The MH index was not part of either study. The DB index ranked tenth best out of 30 in Milligan and Cooper (1985).

These indices provide only two examples and demonstrate how a relative index of partitional adequacy can be applied. These indices depend on a number of problem-specific parameters and may be applicable only when the data cluster into hyperspherical clusters. Not much is known about the performance of relative indices.

4.5 VALIDITY OF INDIVIDUAL CLUSTERS

The two main properties of a cluster are *compactness* and *isolation*. Compactness measures the internal cohesion among the objects in the cluster whereas isolation measures separation between the cluster and other patterns. A valid cluster is unusually compact and unusually isolated. Clusters that are isolated but not compact or compact but not isolated are valid in a restricted sense. All indices of individual cluster validity measure compactness and isolation. The trick is to define a reference population and baseline distribution in terms of which "unusual" has meaning, that make intuitive sense, and which can be established from theory.

4.5.1 External Indices

An external index of cluster validity is appropriate when the cluster labels are assigned without regard to the proximity matrix. When the data are on an ordinal scale, the problem of validating individual clusters can be solved from probability theory. This section is restricted to the case of ordinal data and follows Bailey and Dubes (1982).

The reference population is implied by the random graph hypothesis (Section 4.1.1), which states that all ordinal proximity matrices are equally likely. Given n objects, the proximity matrix contains $n(n - 1)/2$ entries, so there are $[n(n - 1)/2]!$ different proximity matrices. Selecting a proximity matrix under the random graph hypothesis is equivalent to randomly placing edges between distinct nodes on an n-vertex graph where all unfilled positions are equally likely at each step. The order in which the edges are inserted establishes the ordinal proximities.

The compactness and isolation indices of a cluster are defined as the number of edges internal to the cluster and the number of edges linking the cluster to other clusters at each level of proximity. When the cluster is specified a priori, with no reference to the proximities, the distributions of compactness and isolation at a particular level of proximity can be obtained from the hypergeometric distribution so that an exact measure of cluster validity can be formed. The details of this idea are now explained.

An (n, N) random threshold graph is a graph formed by inserting N edges at random into an n-node graph, where each node represents one object. A set of objects is represented by the integers $\{1, 2, \ldots, n\}$, and the a priori cluster of interest is defined by the subset of integers A; $d(i, j)$ denotes the ordinal proximity between objects i and j, expressed as an integer in the range

$$1 \le d(i, j) \le \frac{n(n-1)}{2}$$

Given a set of n objects and ordinal dissimilarities, the edges in an (n, N) random threshold graph are classified into sets of "inner," "outer," or "linking" edges as follows for cluster A.

Inner edges: $D_{\text{in}}(A) = \{d(i, j) : i \in A, j \in A, i < j\}$
Outer edges: $D_{\text{out}}(A) = \{d(i, j) : i \notin A, j \notin A, i < j\}$
Linking edges: $D_{\text{link}}(A) = \{d(i, j) : i \in A, j \notin A, i < j\}$

The *compactness* of cluster A at level N, $e_A(N)$, is the number of internal, or inner, edges in an (n, N) random threshold graph.

$$e_A(N) = |\{d(i,j) : d(i,j) \le N\} \cap D_{\text{in}}(A)\}|$$

The *isolation* of cluster A at level N, $b_A(N)$, is the number of edges linking A with the rest of the graph.

$$b_A(N) = |\{d(i,j) : d(i,j) \le N\} \cap D_{\text{link}}(A)\}|$$

Letting $k = |A|$, the number of objects in cluster A, a perfect cluster at level N is one for which

$$b_A(N) = 0 \qquad \text{and} \qquad e_A(N) = \frac{k(k-1)}{2}$$

A valid cluster should have $e_A(N)$ unusually large and $b_A(N)$ unusually small. How large and small are "unusually" large and small? Here is where the baseline distribution comes into play. Let B be a random variable denoting the number of linking edges and E be a random variable denoting the number of internal edges in an (n, N) random threshold graph. The hypergeometric distribution (Appendix B) provides the distributions for these random variables under the random graph hypothesis as long as A is chosen without regard to the proximity matrix. For example, letting "defectives" be linking edges in a random threshold graph containing n nodes and N edges, the probability of observing exactly b edges linking nodes in A to the rest of the graph in an (n, N) random threshold graph is given below. The notation "$n : 2$" means "n things taken two at a time," or $n(n-1)/2$.

$$P(B = b \mid H_0) = \frac{\binom{k(n-k)}{b}\binom{n:2-k(n-k)}{N-b}}{\binom{n:2}{N}}$$

where $\max\{0, N - (n:2) + k(n-k)\} \le b \le \min\{N, k(n-k)\}$.

In the language of Appendix B, this is the probability of observing b defectives in a sample of size N when the population of $n : 2$ edges contains $k(n-k)$ defectives. The probability of exactly e internal edges is also obtained from the hypergeometric distribution by labeling edges internal to cluster A as "defectives." The probability of observing e defectives in a sample of size N when the population of $n : 2$ edges contains $k : 2$ defectives is given next.

$$P(E = e \mid H_0) = \frac{\binom{k:2}{e}\binom{n:2-k:2}{N-e}}{\binom{n:2}{N}}$$

where $\max\{0, N - (n:2) + (k:2)\} \le e \le \min\{N, k:2\}$.

If b_A is the observed number of edges linking objects in a priori cluster A to the rest of the objects at dissimilarity level N, the *isolation index* for A is defined as

$$I(A) = P(B \leq b_A \mid H_0) = \sum_{b=0}^{b_A} P(B = b \mid H_0)$$

Denoting the number of edges internal to a priori cluster A at level N as e_A, the *compactness index* for A is

$$C(A) = P(E \geq e_A \mid H_0) = \sum_{e=e_A}^{k:2} P(E = e \mid H_0)$$

Both indices should be small for a ''good'' cluster at level N. A *cluster profile* is a plot showing $I(A)$ and $C(A)$ as functions of N. A valid cluster has both indices small for acceptably long spans of N values. Both indices will be 1 when all $n(n - 1)/2$ edges have been inserted. The definition of ''acceptably long'' must be established in each application. The responsibility lies with the user of the indices.

Example 4.7

This example demonstrates the use of the compactness index as an external index of validity for an a priori cluster. The ordinal dissimilarity matrix for 10 objects is given below and the cluster

$$A = \{2, 3, 6, 8\}$$

is to be examined for compactness. We emphasize that this cluster was chosen without regard to the dissimilarity matrix.

	2	3	4	5	6	7	8	9	10
1	29	27	16	10	34	5	15	1	30
2	—	8	33	28	14	32	22	37	10
3	—	—	45	19	20	12	23	44	41
4	—	—	—	9	35	17	7	2	4
5	—	—	—	—	36	11	6	18	31
6	—	—	—	—	—	38	21	39	26
7	—	—	—	—	—	—	25	42	3
8	—	—	—	—	—	—	—	43	24
9	—	—	—	—	—	—	—	—	13

It is difficult to judge the compactness of a priori cluster A from the proximity matrix itself. Drawing threshold graphs is not very informative. Figure 4.15 shows the threshold graph containing the first 20 edges. We can observe that three edges are internal to the cluster, five edges link the cluster to the remaining nodes, and 12 edges are external. Is it unusual to have as many as three internal edges and five linking edges in a 20-edge threshold graph on 10 objects under the random graph hypothesis? The answer to this question determines whether a priori cluster A is valid.

Cluster profiles plot $C(A)$ and $I(A)$ against N, the number of edges in the threshold graph, to assess the compactness and isolation of the a priori cluster A. For instance, exactly two internal edges occur when N is between 14 and 19. Thus the numerical formula for $C(A)$ can be written as follows when N is between 14 and 19:

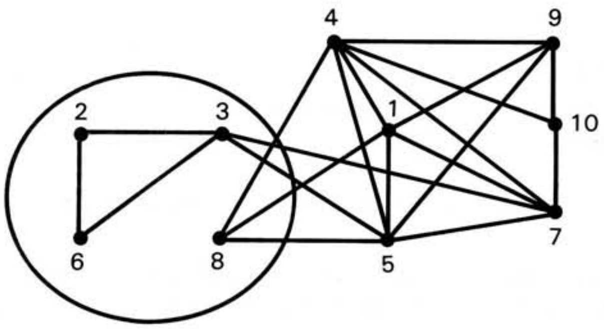


Figure 4.15 Threshold graph on 10 objects with 20 edges.

$$C(A) = 1 - \frac{\binom{39}{N}}{\binom{45}{N}} - \frac{6\binom{39}{N-1}}{\binom{45}{N}}$$

A compact cluster has small $C(A)$ and an isolated cluster has small $I(A)$ for a range of N values. A valid cluster is both compact and isolated. The cluster profiles in Figure 4.16 plot $C(A)$ and $I(A)$ for all values of N.

The edges internal to cluster $\{2, 3, 6, 8\}$ are inserted at ranks 8, 14, 20, 21, 22, and 23. The response of $C(A)$ to the occurrences of these edges is clear in Figure 4.16. Figure 4.16 exhibits some evidence of compactness between rank 23 and 28 because $C(A)$ is less than 0.05 between these ranks. However, it exhibits much more evidence of isolation than of compactness. For how wide a range of N must $I(A)$ be less than 0.05 for one to call A a truly isolated cluster? The answer depends on the experience and philosophy of the experimenter. Remember that we require probabilities to have intrinsic meaning for our measures of cluster validity to be meaningful. It appears that cluster A is more isolated than compact. The evidence for compactness is weak, so we would not call A a valid cluster.

4.5.2 Internal Indices

An external index validates an a priori cluster by contrasting the isolation and compactness of the cluster to the compactness and isolation of a randomly chosen cluster. This validation scheme is not fair when a cluster is chosen after inspecting the data with a clustering algorithm. Almost any cluster identified by a clustering algorithm is unusually isolated and compact when compared to a randomly chosen cluster. How to define a reference population and baseline distribution that provide fair tests of cluster validity? We discuss two possibilities: the CM-reachable method (CM for ''clustering method'') and the best-case method. The CM-reachable method uses Monte Carlo analysis and the best-case method employs bounds on probabilities.

Consider an (n, N) threshold graph selected under the random graph hypothesis

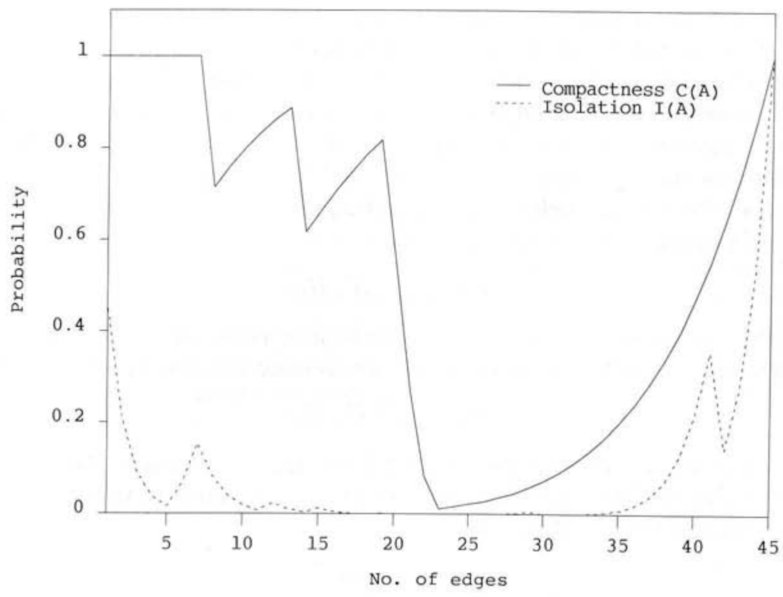


Figure 4.16 Cluster profiles for a priori cluster {2, 3, 6, 8}.

and suppose that a population of k-node clusters is to be defined. Under the random graph hypothesis, all $n : k$ clusters in this population are equally likely. The CM-reachable method limits the population of clusters to those clusters that can be "reached," or achieved, by a particular clustering method when applied to proximity matrices created under the random graph hypothesis. A sampling distribution can then be worked out over the limited population by Monte Carlo means, taking into account the number of objects, the level, N, at which the cluster is formed, the size of the cluster, and the clustering method being employed. Baker and Hubert (1976) develop a test of cluster validity in this way for complete-link clusters. We emphasize that the number of parameters implies that a Monte Carlo analysis be carried out for each particular case.

The procedure suggested above is specific to a particular clustering method, a particular number of objects, a particular cluster size, and a particular proximity level. Each cluster must be evaluated by the laborious Monte Carlo method. The "best-case" method provides a second way of validating a cluster that is independent of the clustering method. The idea is to establish conservative bounds on the probabilities required in computing compactness and isolation indices. First select an (n, N) graph under the random graph hypothesis and reserve the k-node subsets for which the isolation index is "best," or minimum, among all k-node subsets. All the best k-node subsets are assigned equal probability mass and all other k-node subsets are assigned probability mass zero. The population of all such k-node subsets that can be achieved from random graphs is denoted B^*. Similarly, population E^* is the set of k-node subsets of n nodes having the best, or largest, compactness index in N-edge graphs selected under the random graph hypothesis. We use the number of linking edges to measure isolation and the number of internal edges to measure compactness. The bases for comparison are the isolation and compactness measures for subsets in B^* and E^*. A measure of isolation for cluster A with isolation index b_A is the probability that A is at least as isolated as the most isolated cluster in a random graph.

$$P(B \leq b_A \mid B^*, H_0)$$

Similarly, the compactness of cluster A with compactness index e_A is the probability of obtaining a cluster in a random graph that is more isolated than A.

$$P(E \geq e_A \mid E^*, H_0)$$

Exact expressions for these probabilities are not available, but in lieu of Monte Carlo analysis, the following bounds can be used to define indices of cluster validity for k-node clusters.

$$P(B \leq b \mid B^*, H_0) \leq \binom{n}{k} P(B \leq b \mid H_0)$$

$$P(E \geq e \mid E^*, H_0) \leq \binom{n}{k} P(E \geq e \mid H_0)$$

The proof of the first bound follows from two facts. Elementary probability theory shows that

$$P(B \leq b \mid B^*, H_0) = \frac{P(B \leq b \cap B^* \mid H_0)}{P(B^* \mid H_0)} \leq \frac{P(B \leq b \mid H_0)}{P(B^* \mid H_0)}$$

Since every threshold graph has at least one most isolated subset and there are $\binom{n}{k}$ k-node subsets,

$$P(B^* \mid H_0) \geq 1/\tbinom{n}{k}$$

The second bound can be established in the same way. The probabilities on the right sides of these bounds are exactly those derived with internal indices.

The best-case isolation index, $I_1(A)$, and the best-case compactness index, $C_1(A)$, for the k-node cluster A with isolation b_A (number of linking edges) and compactness e_A (number of internal edges) are given below. Both are functions of dissimilarity level, N.

$$I_1(A) = \binom{n}{k} \sum_{b=0}^{b_A} \frac{\binom{k(n-k)}{b} \binom{n:2 - k(n-k)}{N-b}}{\binom{n:2}{N}}$$

$$C_1(A) = \binom{n}{k} \sum_{e=e_A}^{k:2} \frac{\binom{k:2}{e}\binom{n:2-k:2}{N-e}}{\binom{n:2}{N}}$$

Plots of $I_1(A)$ and $C_1(A)$ as functions of N, the number of edges in the random graph, define cluster profiles for cluster A. If $I_1(A)$ and $C_1(A)$ are "small" for a reasonably large range of values of N, cluster A is called valid. Since we are dealing with bounds, the fact that these indices are not small does not mean the cluster is not valid. These indices are conservative, so good clusters might be missed. Two other indices are defined below that permit one to judge whether a cluster is compact among all clusters as isolated as the given cluster and whether a cluster is unusually isolated among all clusters having the same degree of compactness as the given cluster.

$$I_2(A) = P(B \le b_A \mid E = e_A, H_0)$$

$$C_2(A) = P(E \ge e_A \mid B = b_A, H_0)$$

Exact formulas for these indices can be derived from the hypergeometric distribution by naming either internal edges or linking edges as "defectives."

$$I_2(A) = \sum_{b=0}^{b_A} \frac{\binom{k(n-k)}{b_A}\binom{(n-k):2}{N-e_A-b_A}}{\binom{k(n-k)+(n-k):2}{N-b_A}}$$

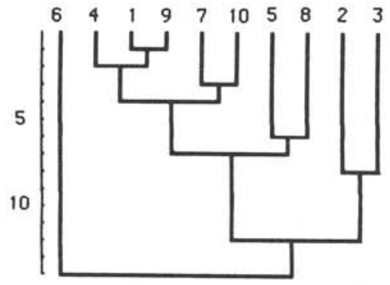
$$C_2(A) = \sum_{e=e_A}^{k:2} \frac{\binom{k:2}{e}\binom{(n-k):2}{N-b_A-e}}{\binom{k:2+(n-k):2}{N-b_A}}$$

A *probability profile* plots the four indices as functions of N. If criteria $I_1(A)$ and $C_1(A)$ are both below a suitable level, say 0.05, for an appropriate span of ranks, cluster A is called valid. A small $C_2(A)$ for a range of N indicates that A is unusually compact among clusters as isolated as A for that range of N, while $I_2(A)$ measures the isolation of A relative to clusters as compact as A. The relative indices are weaker than the best-case indices. Judging a cluster "valid" from a probability profile does not always imply that all objects in the cluster have the same degree of isolation from other nodes in the graph or the same degree of adhesion to other nodes in the cluster. For example, a cluster formed as the union of two very compact clusters might itself be judged unusually compact because of the individual clusters, not because of any homogeneity among all nodes in the cluster. Note that these indices validate isolated clusters and do not treat the case of mutual isolation or compactness among a set of clusters. Bailey and Dubes (1982) provide several examples.

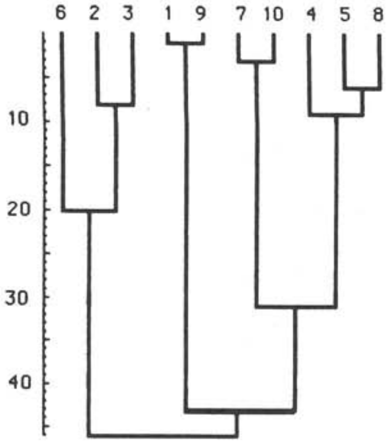
Example 4.8

Single-link and complete-link proximity dendrograms for the 10 by 10 ordinal dissimilarity matrix in Example 4.7 are drawn in Figure 4.17. These dendrograms should suggest candidates for "valid" clusters. Only internal criteria of validity can be used with clusters derived from dendrograms. A "good" cluster is both compact and isolated, which means that it forms early in the dendrogram and is not absorbed into another cluster until late in the dendrogram. Complete-link cluster {2, 3, 6} in Figure 4.17(b) appears to be a good candidate for a valid cluster and will be examined in this example.

The probability profiles for complete-link cluster A = {2, 3, 6} are shown in Figure 4.18. We emphasize that these profiles treat the numbers of internal and linking edges as external indices. The upper bound $C_1(A)$ is identically 1, which shows that this cluster is not unusually compact. On a relative basis, $C_2(A)$ shows that A is also not unusually compact for a cluster as isolated as A because $C_2(A)$ never falls below our agreed threshold of 0.05. Cluster A is somewhat isolated because $I_1(A)$ is below 0.05 for ranks 14 through 22 and is unusually isolated for a cluster this compact because $I_2(A)$ is below 0.05 for ranks 5 through 35. We conclude that A shows evidence of isolation but not of compactness and is not a valid cluster.



(a)



(b)

Figure 4.17 Proximity dendrograms for dissimilarity matrix of Example 4.7: (a) single-link proximity dendrogram; (b) complete-link proximity dendrogram.

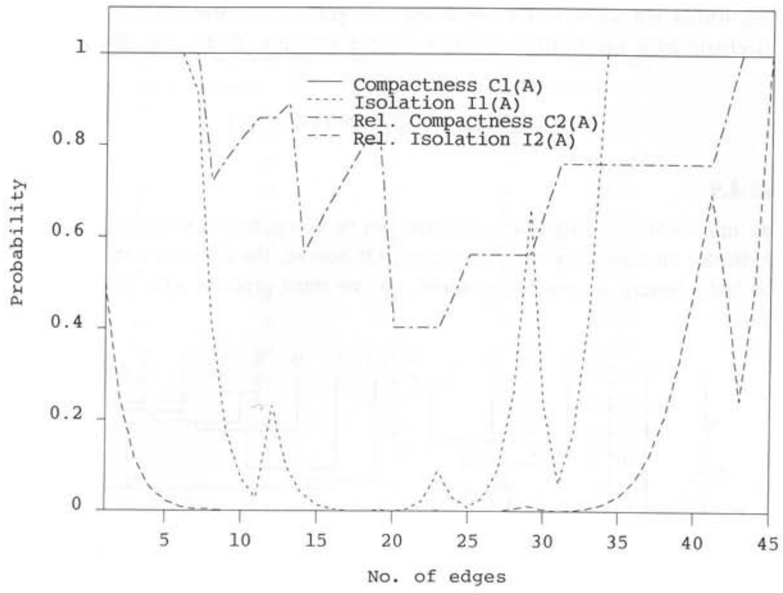


Figure 4.18 Probability profiles for complete link cluster {2, 3, 6}.

This type of analysis for complete-link cluster {1, 9} in Figure 4.17(b) shows that cluster {1, 9} is neither compact nor isolated; both $C_1(\{1, 9\})$ and $I_1(\{1, 9\})$ are identically 1 for all ranks. The same is true for single-link cluster {1, 4, 9}.

Ling (1972, 1973a) has proposed an index of cluster validity for qualitative data as the probability that the lifetime of a cluster exceeds the observed lifetime under the random graph hypothesis. The lifetime, $L(A)$, of cluster A is

$$L(A) = c_2 - c_1$$

where c_2 is the rank at which A is absorbed into another cluster and c_1 is the rank at which A is first defined. Valid clusters should ''live'' longer than randomly chosen clusters. Ling provides the following expression for the probability that a randomly chosen single-link cluster of size k has lifetime s, under the random graph hypothesis; L is a random variable representing lifetime and n is the number of objects being clustered.

$$P[L = s \mid H_0] = \frac{k(n-k)\binom{n:2 - c_1 - k(n-k)}{s-1}}{s\binom{n:2 - c_1}{s}}$$

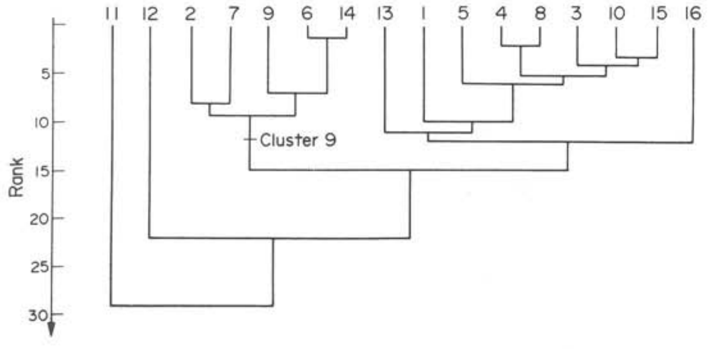
This expression can be derived by realizing that a cluster has lifetime s only if the s edges inserted into the random graph after level c_1 are either internal to or external to the cluster. The cluster is absorbed with the next linking edge.

The Ling index for cluster A containing k objects is defined to be the probability that a lifetime of a randomly selected cluster exceeds $L(A)$, the observed lifetime of A.

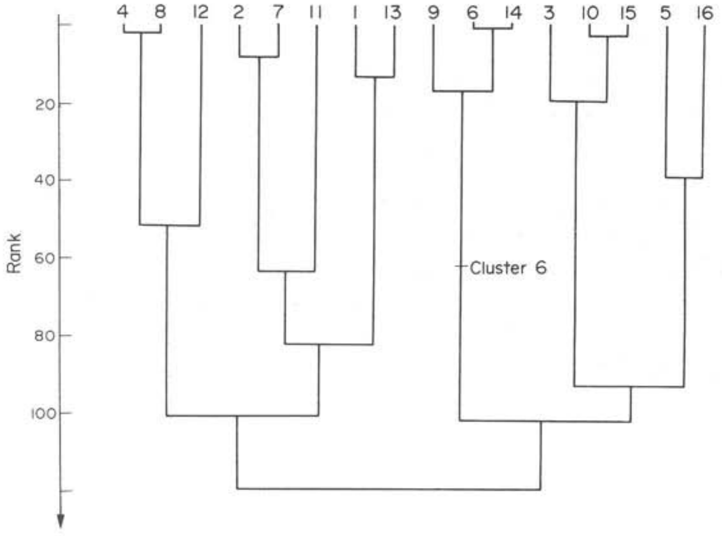
$$\text{Ling index} = P[L > L(A) \mid H_0]$$

Example 4.9

The Ling indices for all single-link clusters can be computed to search for a small index, which indicates an unusually isolated cluster. Of course, the lifetime random variables for all single-link clusters are not independent, so one must proceed with caution. Bailey and



(a)



(b)

Figure 4.19 Dendrograms for random proximity matrix. (Source: Reproduced with permission from Bailey and Dubes.)

TABLE 4.9 Ling Indices for Single-Link Clusters

Cluster	Birth Rank	Lifetime	Size	Ling Index
1	1	6	2	0.196
2	2	3	2	0.441
3	3	1	2	0.761
4	4	1	3	0.664
5	5	1	5	0.522
6	6	4	6	0.054
7	7	2	3	0.428
8	8	1	2	0.750
9	9	6	5	0.023
10	10	1	7	0.427
11	11	1	8	0.413
12	12	3	9	0.076
13	15	7	14	0.111
14	22	7	15	0.303
15	29	—	16	—

(Source: Reproduced with permission from Bailey and Dubes.)

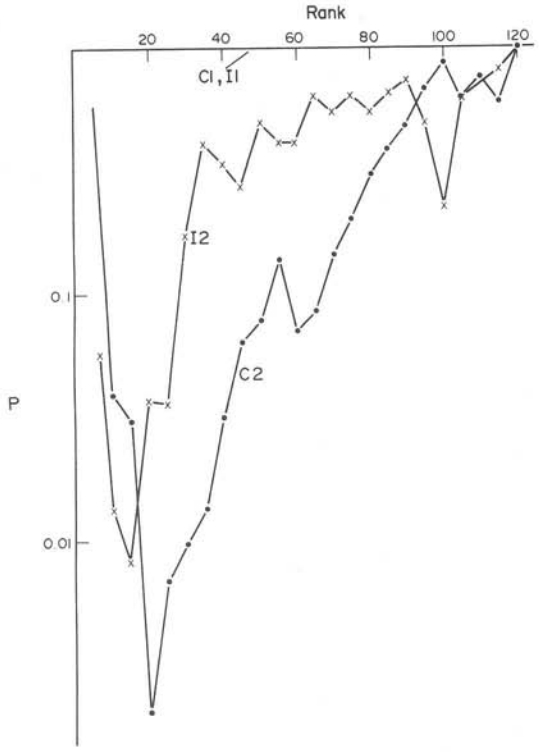


Figure 4.20 Profiles for best single-link cluster. (Source: Reproduced with permission from Bailey and Dubes.)

TABLE 4.10 Matula Indices for Complete-Link Clusters

Cluster	Birth Rank	Lifetime	Size	Matula Index
1	1	16	2	0.502
2	2	50	2	0.670
3	3	16	2	0.755
4	8	55	2	0.896
5	13	69	2	0.936
6	17	95	3	0.488
7	19	74	3	0.550
8	39	54	2	0.983
9	52	59	3	0.900
10	63	19	3	0.931
11	82	29	5	0.639
12	93	19	5	0.768
13	111	9	8	0.575
14	112	8	8	0.615
15	120	—	16	—

(Source: Reproduced with permission from Bailey and Dubes.)

Dubes (1982) demonstrated this process with an ordinal dissimilarity matrix of size 16, generated under the random graph hypothesis. Figure 4.19(a) shows the single-link dendrogram and Table 4.9 provides the Ling indices for all single-link clusters.

Cluster 9 is the most unusual single-link cluster according to the Ling index. One might conclude that none of the clusters are valid because the minimum Ling index is as large as 0.023. The validity of cluster 9 can also be assessed with the probability profiles in Figure 4.20. The conservative indices C_1 and I_1 are always zero, correctly indicating that the cluster is not valid. The relative criteria C_2 and I_2 suggest that cluster 9 is somewhat unusual. This example demonstrates that it is easy to erroneously judge the validity of a cluster.

Example 4.10

Matula (1977) proposed a lower bound on the asymptotic distribution for the size of the largest complete subgraph in a random threshold graph that can be used to assess the compactness of a complete-link cluster. A small value indicates an unusual cluster. Table 4.10 identifies all complete-link clusters in the 16-object random data set reported above. The complete-link dendrogram is shown in Figure 4.19(b). None of the clusters have unusually small values. Cluster 6 has the smallest value and the probability profiles for cluster 6 are shown in Figure 4.21. The two conservative indices do not deviate from 0, while the relative index C_2 suggests that cluster 6 is somewhat compact.

4.5.3 Relative Indices

The problem of deciding the relative merits of two clusters has not received much attention and is not as important as the validation of individual clusters. Comparing the probability profiles of the two clusters is one approach. For example, if the data were perturbed in some way to test the stability of the cluster, the

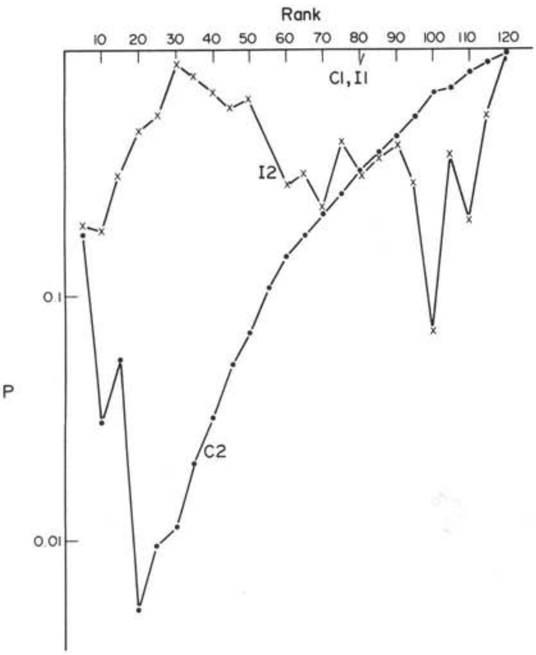


Figure 4.21 Profiles for complete-link cluster 6. (Source: Reproduced with permission from Bailey and Dubes.)

probability profiles before and after the perturbation can be compared to assess stability.

4.6 CLUSTERING TENDENCY

The term "clustering tendency" refers to the problem of deciding whether data exhibit a predisposition to cluster into natural groups without identifying the groups themselves. Clustering algorithms will create clusters whether the data are naturally clustered or purely random. We have seen that the problem of validating clusters after the fact is very difficult. Therefore, a preliminary assessment of clustering tendency should be an important part of clustering methodology (Section 3.5). It would establish whether clusters exist and would prevent the inappropriate application of clustering algorithms. A test for clustering tendency would also guard against the embarrassment of applying elaborate clustering techniques and cluster validity methodology to data in which the clusters can only be artifacts of the clustering algorithm.

The problem of testing for clustering tendency can also be phrased as the problem of testing for spatial randomness. The procedures discussed in this section have been used to test for "complete spatial randomness" in geology, biology, forestry, and geography (Diggle, 1983; Ripley, 1981). Almost all these applications are to problems in two dimensions. Our treatment of the clustering tendency problem characterizes data as samples of spatial point processes. A related problem is to model data by fitting a spatial point process to a sample of patterns. For example, Shapiro et al. (1985) fit a spatial point process to data taken from the blue cones of the retina. Ogata and Tanemura (1984) fit a process to data from molecular dynamics. Diggle (1983) fits models to data describing stands of trees and locations of towns. It may eventually be possible to use such an analysis as an alternative to cluster analysis. For example, one might fit a spatial point process to data by estimating parameters and interpreting the parameters of the model instead of interpreting the results of cluster analysis. Not enough is known about choosing a class of models, about estimating parameters, and about testing for goodness of fit to call such an approach practical at present. Almost all available theory treats models in two dimensions and depends on the ability to see the data when fitting a model. Knowledge of the sampling window is also necessary.

A test for clustering tendency is stated in terms of an internal criterion. No category or other a priori information is brought into the analysis. The definition of randomness and the type of clustering tendency test depend on the form of the available data. The first four sections treat the situation when the patterns are points in a *d*-dimensional space and ''randomness'' means ''spatial randomness,'' or the random position hypothesis (Section 4.1.1). Section 4.6.5 discusses randomness when the data occur as an ordinal proximity matrix.

4.6.1 Spatial Point Processes and Spatial Randomness

Consider a pattern matrix with n rows (patterns) and d columns (features) in which proximity is measured by distance. A test for clustering tendency (Panayirci and Dubes, 1983; Smith and Jain, 1984; Dubes and Zeng, 1987) examines the spatial arrangement of the patterns and draws one of the following conclusions.

- The patterns are arranged randomly.
- The patterns are aggregated, or clustered, or exhibit mutual attraction.
- The patterns are regularly spaced, or exhibit mutual repulsion.

Our point of view is that random and regularly spaced patterns should not be submitted to clustering algorithms; only data exhibiting the tendency to aggregate should be examined with clustering algorithms. Figure 4.22 illustrates these three types of data in a unit square. This section examines the definition of spatial randomness and the role of the sampling window.

A test for clustering tendency will mean a statistical test of the random position hypothesis, H_0 (Section 4.1.1). This hypothesis is equivalent to what Diggle (1983) calls complete spatial randomness. Thus we are testing a hypothesis of no structure, or that data come from a continuous uniform distribution over some set S in a d-dimensional space, called the *sampling window*. Smith and Jain (1984) define the sampling window mathematically as the compact convex support set for the underlying distribution. In applications, some convenient geometric definition is used, such as a d-dimensional cube or a d-dimensional sphere. As we will see later, the sampling window plays a crucial role in establishing a test of randomness.

The random position hypothesis can also be stated in terms of a d-dimensional spatial Poisson process (Ripley, 1981; Diggle, 1983) restricted to sampling window S. A *spatial point process* is an arrangement of patterns or points scattered about a Euclidean space according to some probability model. The *intensity*, λ, of a process is the expected number of points per unit volume. We deal with processes that are *stationary*, so the characteristics of the process do not vary with position in the sampling window, and *isotropic*, so the process has no directionality.

A *Poisson process* is a particular type of spatial point process that scatters patterns about in such a way that a random variable X denoting the number of patterns in a region having volume V has a Poisson distribution with parameter λV.

$$P(X = k \mid H_0) = \frac{(\lambda V)^k}{k!} e^{-\lambda V} \qquad \text{if } k = 0, 1, 2, \ldots$$

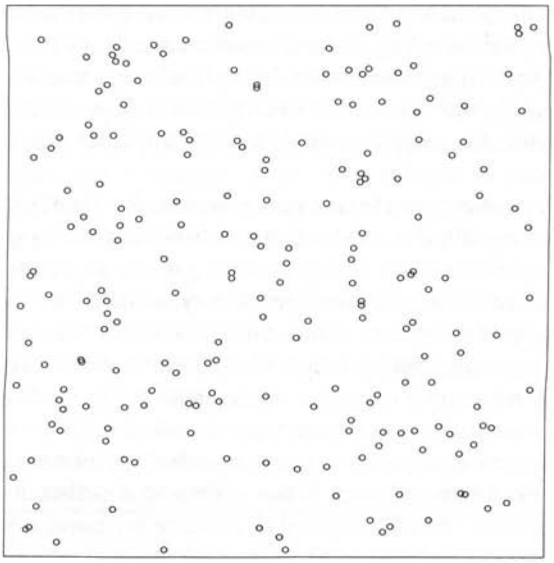
In addition, the position of any one pattern is entirely independent of the positions of other patterns but the patterns do not fall on top of one another. Diggle (1983) and Cox and Isham (1980) provide details. Poisson processes are good models of randomness, but the number of patterns in sampling window S is a random variable. *Binomial sampling* means selecting a fixed number of points in a given region uniformly; in other words, the process is Poisson, conditioned on the number of points.

As discussed in Section 4.1.1, a test statistic T and a threshold t are required to test for randomness. A typical one-sided test of H_0 against an alternative of aggregation has the form

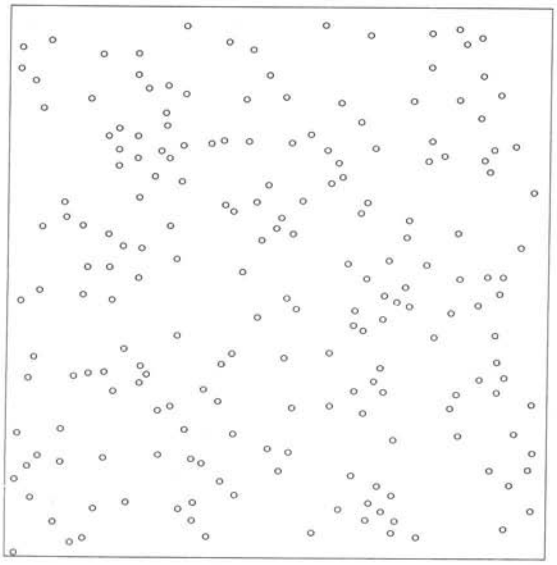
$$\text{Accept } H_0 \text{ if } T \geqslant t$$

where threshold t is selected to achieve a specified probability, α, of rejecting H_0 when it is true. The two main difficulties in testing for clustering tendency are the definition of sampling window and the dimensionality of the feature space. A brief survey of approaches to the clustering tendency problem is provided in Section 4.6.3.

The concept of sampling window is fundamental to the notion of randomness. Consider two-dimensional data generated uniformly inside a unit-diameter circle. If the sampling window is taken to be this unit-diameter circle, we expect good tests of randomness to view these data as random. What happens if the sampling window is enlarged to a unit-radius circle? The data are no longer uniformly distributed over this larger circle. Thus the same data can appear as random or

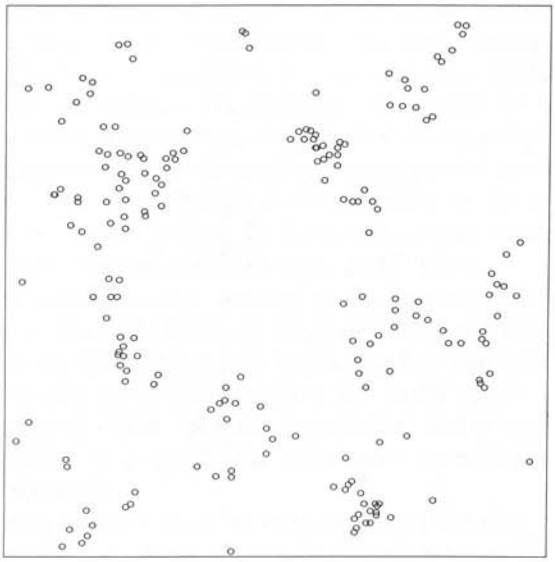


(a)

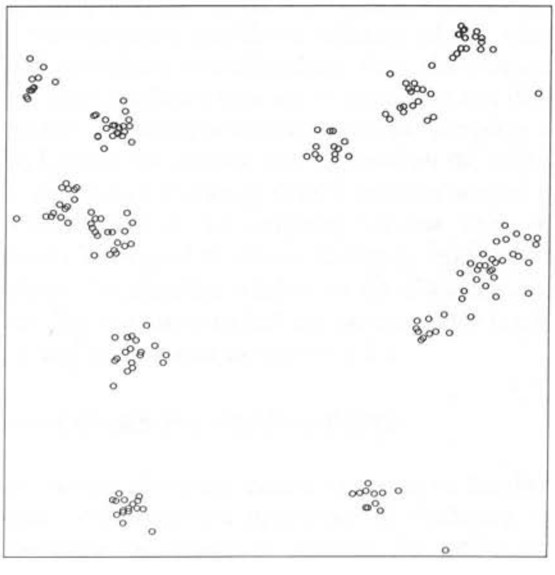


(b)

Figure 4.22 Random, clustered and regular data: (a) Poisson process (200 patterns); (b) hardcore process (200 patterns, radius = 0.02); (c) cluster process (200 patterns, $\mu = 4.0$, $\sigma = 0.025$); (d) cluster process (200 patterns, $\mu = 16$, $\sigma = 0.025$).

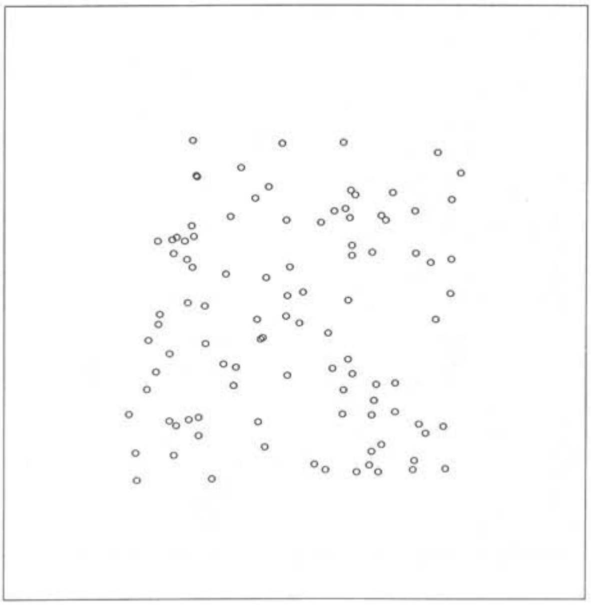


(c)

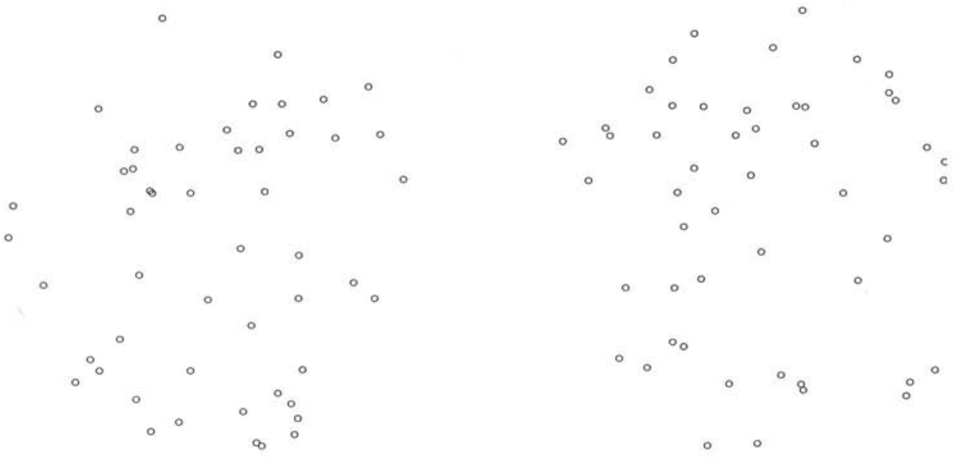


(d)

Figure 4.22 *(continued)*



(a)



(b)

Figure 4.23 Random data in inappropriate sampling windows: (a) data uniform over small subsquare (100 patterns); (b) data uniform over two disjoint circles (50 patterns per cluster).

nonrandom, depending on the sampling window assumed. Figure 4.23 shows random data in sampling windows that make the data appear nonrandom.

A second difficulty in assuming an arbitrary sampling window without regard to the data is that the statistical characteristics of the data are different near the edges of the sampling window than in the center of the sampling window. For example, the distribution of the distance between a pattern and its closest pattern depends on how close the pattern is to the boundary of the sampling window. This problem is of special concern in geography, where several techniques for edge correction have been proposed (Griffith, 1983, 1985).

Knowledge about the shape and size of the sampling window is required in virtually all tests of clustering tendency which have been proposed in the literature. One way of avoiding the problem is to begin with a d-dimensional rectangle as a sampling window and repeat it throughout the entire space, like a patchwork quilt. This technique is called ''periodic boundaries'' and is equivalent to viewing the sampling window as a torus. One must know something about the data to select the basic rectangle.

The sampling window must be estimated from the data in many practical situations, as explained by Smith (1982). If the sampling window is restricted to be a hyper-rectangle aligned with the coordinate axes, then a minimum variance, unbiased estimator of the sampling window exists under H_0. When the sampling window is an arbitrary convex set in two dimensions, the convex hull of the uniform data is the maximum likelihood estimate of the window (Ripley and Rasson, 1977). This introduces two difficulties. First, the complexity of algorithms for finding convex hulls precludes their use in more than five dimensions. Second, the distribution of the test statistics derived when the sampling window is known must be modified because the convex hull depends on the data. Zeng and Dubes (1984, 1985a,b), and Dubes and Zeng (1987) examine several tactics for dealing with insufficient knowledge of the sampling window when statistics based on interpattern distances were used to assess clustering tendency. The one showing most promise defines the sampling window as the d-dimensional sphere centered at the mean of the data that encloses half the patterns. This is a type of ''sampling frame'' technique and is discussed in section 4.6.4.

4.6.2 Spatial Clustering and Regularity

The form of a test for clustering tendency depends on the alternative hypothesis under consideration. An alternative hypothesis of clustering, or aggregation, is needed when comparing the powers of statistics for testing clustering tendency and for determining whether new statistics work better than existing ones. The Neyman–Scott (1972) process has been used for these purposes. This process is most easily defined by its generation mechanism. The parameters are the mean cluster size, μ, and the spread, σ. The sampling window and the number of patterns desired, n, are assumed to be given.

ALGORITHM FOR NEYMAN–SCOTT PROCESS

Step 1. Select a cluster center at random (uniformly) over S.

Step 2. Select N_1, the number of patterns in the current cluster, as a sample of a Poisson random variable with mean μ.

Step 3. Assign the d coordinates of each of the N_1 patterns relative to the cluster center by selecting d samples from an $N(0, \sigma^2)$ distribution. If a pattern falls outside S, ignore it and select another point.

Repeat steps 1 to 3 until n patterns have been generated.

The Neyman–Scott process generates a random number of Gaussian clusters of random size at random positions. The clusters may overlap because all cluster centers are selected randomly but all clusters have the same spread. Appendix H describes a mechanism for generating clustered data that is based on the Neyman–Scott process but which controls the overlap among clusters.

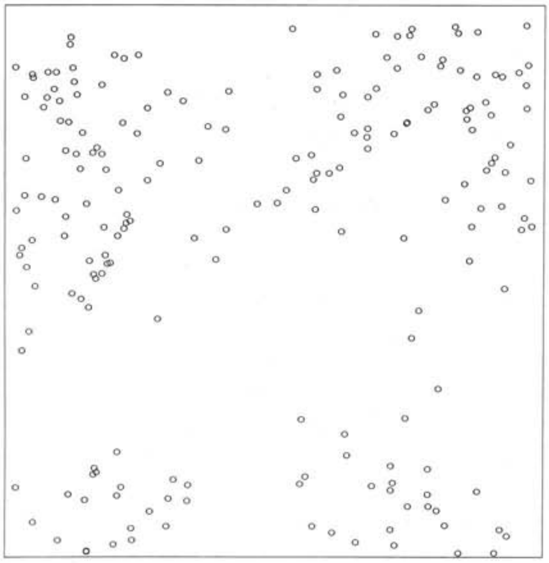
When periodic boundaries are assumed, clusters can overlap the boundaries of S. Some algorithms include the cluster centers as patterns. The covariance structure here is as simple as possible to minimize the number of parameters. This implementation generates exactly the number of patterns requested. Some implementations obtain the number of patterns from a sample of a Poisson random variable with parameter λV, where V is the volume of the sampling window. Our algorithm assumes that $n = \lambda V$. Each cluster is then filled in as indicated in steps 2 and 3. Ripley (1977, 1981) suggests other models for clustering in two dimensions. Figure 4.24 shows a few examples of Neyman–Scott process realizations in a unit hypercube in two dimensions, using a torus topology.

Alternative hypotheses of regularity are not as important as those of clustering in the current context. Lattice regularity, in which a pattern is placed at each vertex of a lattice defined over S, is the simplest model. The SSI (simple sequential inhibition) process is a more reasonable model of regularity for our purposes. Imagine each pattern to be surrounded by a small sphere and suppose that the patterns are packed into the sampling window randomly but in such a way that the spheres do not intersect. These spheres surrounding the patterns are called ''hard'' spheres. Some models of regularity allow ''soft'' spheres, or limited intersection among the spheres. The packing density, ρ, is the ratio of the space used up by the hard spheres to the volume, V, of the sampling window.

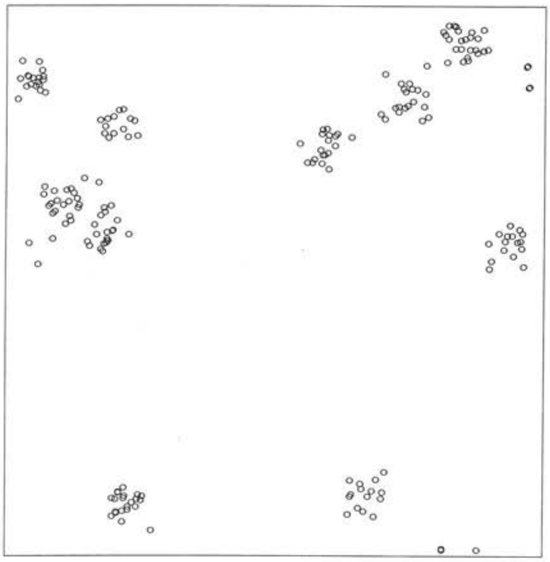
$$\rho = \frac{LA[(r/2)^d]}{V}$$

The total volume consumed by the hard spheres is determined from L, the expected number of patterns per unit volume, and r, the radius of the hard sphere surrounding each pattern; A is the volume of a unit-radius sphere in d dimensions.

$$A = \frac{\pi^{d/2}}{\Gamma[(d/2) + 1]}$$



(a)



(b)

Figure 4.24 Realizations of Neyman-Scott cluster processes with periodic boundaries: (a) cluster process (200 patterns, $\mu = 20$, $\sigma = 0.75$); (b) cluster process (200 patterns, $\mu = 20$, $\sigma = 0.025$).

The most direct way of generating a sample from this process is to place patterns into the sampling window one at a time at random locations, each surrounded by a hard sphere of radius r, ensuring that no two hard spheres overlap. Patterns falling near enough the edge of the sampling window to cause the hard sphere to overlap the edge can either be discarded or kept, but the equation for packing density above will not be valid if hard spheres are allowed to extend outside the sampling window. With periodic boundaries, part of the hard sphere for a pattern at one edge of the sampling window can appear at the opposite side of the sampling window. The process terminates when the required number of patterns have been inserted or when S is ''full,'' as indicated by the failure to find a place for a pattern after a few thousand trials. Ripley (1977) discusses other ways of generating SSI processes on a computer. The maximum packing density in two dimensions has been shown to be about 0.9069 and is achieved by arranging circles in a triangular lattice. The maximum packing density is not known for more than two dimensions (Sloane, 1984). Kamel et al. (1979) derive near-neighbor distributions for some hard-core models. Figure 4.25 provides an example of an SSI process realization in a square under a torus topology.

The alternative hypotheses of clustering and regularity discussed here were selected with Monte Carlo simulation in mind. Models for use in theoretical analysis

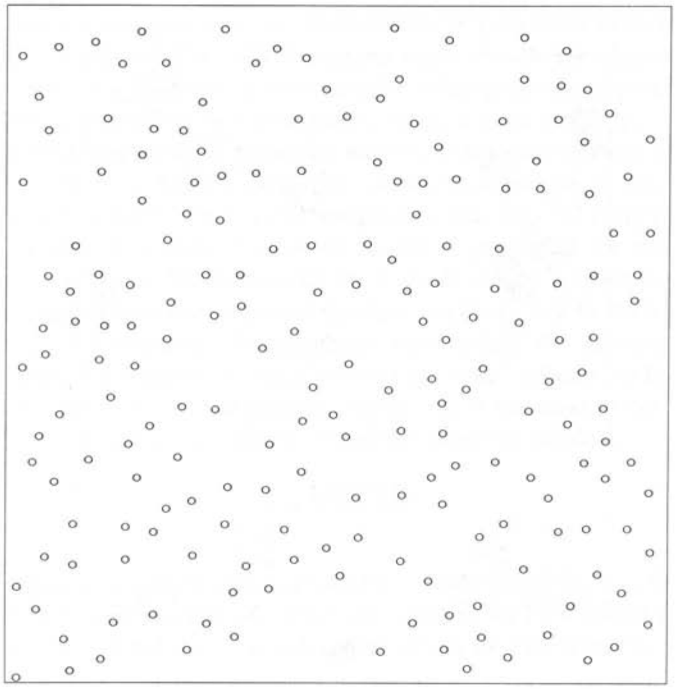


Figure 4.25 Realization of a simple sequential inhibition process with periodic boundaries.

would require that the likelihood function of the process be known. Cox and Isham (1980), Ogata and Tanemura (1981, 1984), Diggle (1983), and others have proposed models such as the Gibbs process as alternatives. The likelihood function for the Gibbs process has the form

$$f(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{e^{-\Sigma\phi[d(i,j)]}}{Z}$$

where ϕ is a "potential" function, $d(i, j)$ is the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$, the sum is over all pairs of patterns, and Z is the "partition" function, or the integral of the numerator with respect to each of the n pattern positions over the sampling window.

Potential functions can be specified to provide models of aggregation and models of regularity. For example, Ogata and Tanemura (1984) define potential functions that create hard-core and soft-core regularity models. Two factors make the Gibbs model very difficult to use in clustering tendency. First, the partition function is notoriously difficult to determine analytically, even in very simple situations. A number of approximations have been proposed over the years, especially in the literature of statistical mechanics. Second, sampling a Gibbs process on a computer is not trivial and requires an iterative algorithm, such as the Metropolis (Metropolis et al., 1953) algorithm. The difficulty with this approach is that one never knows when to stop the iteration. The inability to reliably generate a sample from a Gibbs process on a computer along with the intractability of the partition function are the two main drawbacks to using the Gibbs process as a model in an alternative hypothesis.

4.6.3 Tests for Spatial Randomness

Most of the tests for randomness reported in the literature have been derived for applications in astronomy (Osmer, 1982), ecology (Hopkins, 1954), forestry (Hines and Hines, 1979), and geography (Ripley, 1981), where the data are two-dimensional and the sampling window is known. Some of these tests can be extended to the d-dimensional data encountered in clustering tendency and are surveyed in this section.

Tests for randomness can be characterized according to the type of information used to make a decision. The five categories of tests reviewed briefly below are based on scan statistics, quadrat analysis, second moment estimates, interpoint distances, and graph structures. Section 4.6.4 treats tests based on near-neighbor information in somewhat more detail. We assume that the sampling window is known. All of these tests are subject to the difficulties with sampling windows described above.

Scan tests. *Scan tests* are based on the number of patterns in the most populous subregion of the sampling window. An unusually large count would indicate the presence of clustering. The theory for these tests can be developed only in one dimension (Naus, 1966a, 1982). One two-dimensional test has been proposed (Conover et al., 1979). The span of the subregion must be fixed a priori and is a critical parameter. Extensions to d-dimensions would require searching an entire d-dimensional sampling window for the region with the most patterns, which imposes a severe computational burden.

Quadrat analysis. *Quadrat analysis* partitions a rectangular sampling window into rectangles of equal size, called quadrats, and counts the number of points falling in each quadrat (Grieg-Smith, 1964; Pielou, 1969; Mead, 1974; Rogers, 1974). This set of counts will follow a Poisson distribution under randomness. A chi-square test is typically performed to test a hypothesis of randomness. Quadrats of different sizes can be used to detect the spatial arrangement in the data at more than one scale (Mead, 1974). The number of quadrats becomes enormous even in a moderate number of dimensions, and most of them are empty, so this method cannot easily be extended to more than two dimensions.

Second moment structure. *Second moment structure* refers to the covariance function of the spatial point process. A test based on this structure estimates the second moment of a spatial point process from the given data. Ripley (1977) shows that the second moment structure of a process may be reduced to a function $K(t)$, which has come to be known as ''Ripley's $K(t)$ function,'' having the following properties; λ is the intensity of the process.

1. The quantity $\lambda^2 K(t)$ is the expected number of ordered pairs of distinct points less than distance t apart when the first point is in a given set of unit area.
2. The quantity $\lambda K(t)$ is the expected number of additional points within a distance t of an arbitrary point in the process.

For example, the $K(t)$ function for a Poisson process over the entire d-dimensional feature space is

$$K(t) = At^d$$

where A is the volume of a sphere having unit radius, given in Section 4.6.2. Diggle (1983) demonstrated how a plot of $\hat{K}(t)$ versus t, where $\hat{K}(t)$ is the estimated $K(t)$ function, can be compared to the theoretical $K(t)$ function for a Poisson process to test the random position hypothesis, H_0, in two dimensions. The general idea is to estimate upper and lower envelopes by Monte Carlo simulation under H_0 to establish a confidence band around the theoretical $K(t)$ function, as demonstrated in Figure 4.26 for a Neyman–Scott process. If $\hat{K}(t)$ lies within this band for a span of t values, H_0 is accepted. This test has been applied only in two dimensions. The estimation of $K(t)$, and of other such summary functions, poses a number of theoretical and practical problems. Other tests (Liebetrau, 1977) have been proposed in the same spirit for two-dimensional data in rectangular sampling windows.

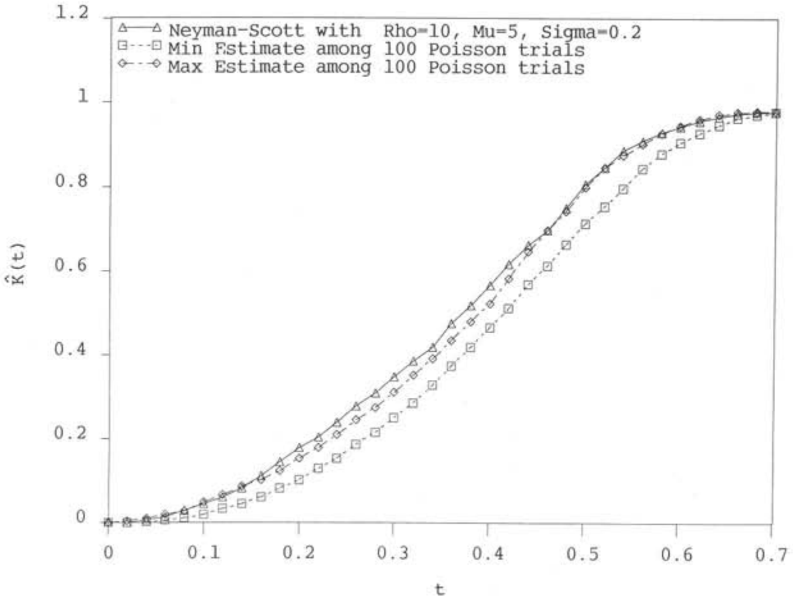


Figure 4.26 Example of $K(t)$ function for Neyman–Scott process.

Interpoint distances. *Interpoint distances* reflect structural relationship among the patterns. One obvious test for randomness compares the observed distribution of interpattern distances to the theoretical distribution under the random position hypothesis, H_0. The distribution of two points randomly chosen from a d-dimensional hypersphere has been established by Hammersley (1950) and Alagar (1976), among others. If an i.i.d. (independent and identically distributed) sample of interpattern distances were available, a standard Kolmogorov–Smirnov statistic or a chi-square statistic could lead to a test of H_0. The set of all $n(n - 1)/2$ interpattern distances is not an i.i.d. sample and the distribution of the sum of these distances is not known under H_0, so some scheme for sampling the distances must be adopted. Ignoring the dependence leads to spurious rejections of the random position hypothesis.

Cross (1980) studied tests for randomness based on a random sample of 50 interpattern distances from among the 4950 distances between 100 patterns and assumed independence. A two-tailed K-S test was applied under various experimental conditions. These tests were found to be very sensitive to scaling. For example, data randomly generated in hypercubes of dimension 2, 5, and 10 rather than in hyperspheres were consistently labeled nonrandom. When random data in a hypersphere were normalized to have zero mean and unit covariance matrix, the random position hypothesis was again rejected repeatedly; the data appeared to be Gaussian, not random. Cross (1980) concluded that the exact sampling window must be known when using this scheme for subsampling distances to avoid spurious rejections of H_0.

Another way of using interpoint distances is to observe only the "small" interpoint distances. Intuitively, a clustered process has an abundance of small distances, a regular process has few small distances, and a random process is somewhere between these extremes. Some reasonable tests based on this idea have been proposed in two dimensions (Ripley and Silverman, 1978; Silverman and Brown, 1978). One problem is to define a suitable threshold for "small." Saunders and Funk (1977) extended a characterization of clustering (Strauss, 1975; Kelly and Ripley, 1976) and derived the distribution of the number of small interpoint distances under certain restrictions. These tests have not been extended to d dimensions.

Structural graphs. *Structural graphs*, especially the minimum spanning tree (MST), Delaunay tessellation and the relative neighborhood graph (see Appendix G and Section 3.3.6) capture more "global" structure than the methods based on small interpoint distances and nearest-neighbor distances discussed in Section 4.6.4. Defining a test statistic based on the distribution of edge lengths in these graphs is a difficult problem. Hoffman and Jain (1983) found that the distribution of edge lengths in the MST under the random position hypothesis, H_0, depends on the number of patterns, number of dimensions, and the sampling window.

Smith and Jain (1984) proposed an MST-based test of H_0 which avoids direct computation of an edge-length distribution. The idea of this test comes from multivariate extension of the Wald–Wolfowitz run test as proposed by Friedman and Rafsky (1979), which tests whether two sets of high-dimensional patterns arise from the same population. Smith and Jain (1984) modified this test for checking randomness as follows.

ALGORITHM FOR MST-BASED TEST OF CLUSTERING TENDENCY

Step 1. Determine the convex region containing the n patterns $\{\mathbf{x}_i\}$ being tested for clustering tendency.

Step 2. Generate m points $\{\mathbf{y}_j\}$ uniformly over a region that approximates the convex region found in step 1.

Step 3. Pool $\{\mathbf{x}_i\}$ and $\{\mathbf{y}_j\}$ and find the MST of the $m + n$ points.

Step 4. Determine T, the number of x-y joins in the MST.

Step 5. Reject H_0 in favor of a clustered alternative if T is "small." Reject H_0 in favor of a regular alternative if T is "large."

An "x-y join" links a pattern in $\{\mathbf{x}_i\}$ to a generated point in $\{\mathbf{y}_j\}$ by an edge in the MST of the pooled sample $\{\mathbf{x}_i, \mathbf{y}_j\}$. We usually set m to n, so the same number of points as patterns are used to compute T. Clustered data should show a higher number of x-x and y-y joins, and thus a lower number of x-y joins, than random data. The number of x-y joins under regularity should be larger than under randomness. Friedman and Rafsky provide the following expressions for the expected value of T, the number of x-y joins, under H_0 and for the variance of T, conditioned on C, the number of pairs of MST edges incident to the same node.

$$E(T \mid H_0) = \frac{2mn}{m + n}$$

$$\operatorname{var}(T \mid C, H_0) = \frac{2mn}{L(L-1)}\left[\frac{2mn - L}{L} + \frac{C - L + 2}{(L-2)(L-3)}[L(L-1) - 4mn + 2]\right]$$

where $L = m + n$, the total number of nodes in the MST. The test statistic is the following normalized version of T that has an approximate normal distribution under H_0:

$$T' = \frac{T - E(T \mid H_0)}{\sqrt{\operatorname{var}(T \mid C, H_0)}}$$

To test H_0 against clustering, reject H_0 when

$$T' < Z(\alpha)$$

where $Z(\alpha)$ is the αth quantile of the standard normal distribution. For example, $Z(0.05) = -1.65$ and $Z(0.01) = -2.33$. Smith and Jain (1984) obtained the size and power of this MST-based test by Monte Carlo simulation. The test gave significantly higher power against clustered alternatives than the Hopkins and Cox–Lewis tests presented in Section 4.6.4. Unfortunately, tests based on the MST have little power against regular alternatives.

Example 4.11

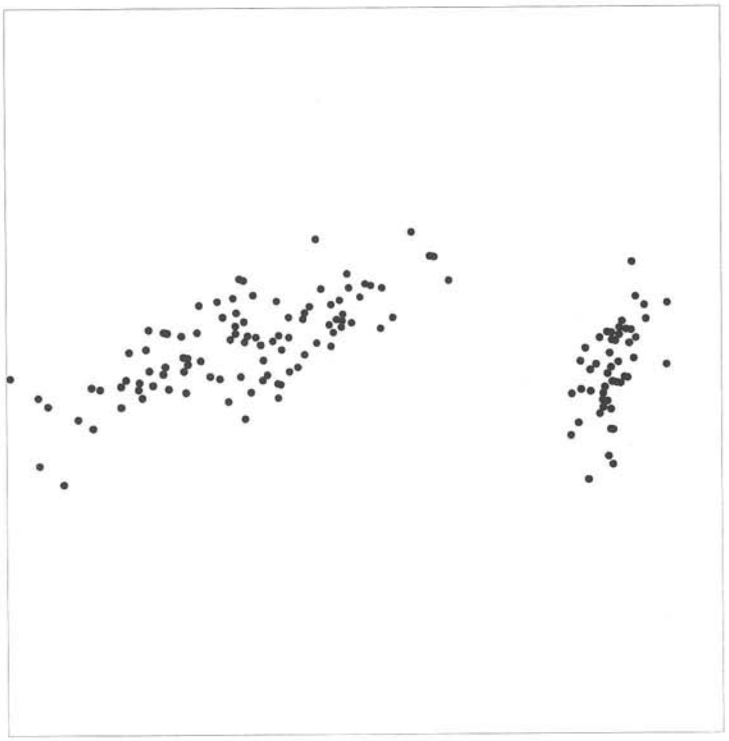
This example demonstrates the Friedman–Rafsky test on a data set called the "iris data" that is famous in the lore of cluster analysis. The data were popularized by Fisher (1936) and consist of 150 patterns, each representing a flower, with four measurements taken on each flower (petal length and width and sepal length and width). The 150 patterns contain 50 samples from each of three varieties (setosa, versicolor, and virginica). The MST-based test was applied to the two-dimensional eigenvector projection of the four-dimensional data both to ease the problem of finding the convex hull of the data and to enable the actual MST to be drawn.

Figure 4.27(a) shows the eigenvector projection of the 150 patterns. The data are clearly clustered. Figure 4.27(b) shows the pooled sample of 150 patterns and 150 points generated randomly inside the convex hull of the 150 patterns. The MST of the pooled sample is shown in Figure 4.27(c). The statistic T' is -6.13, so the critical level of the test described above is less than 0.00001 and we reject the random position hypothesis.

4.6.4 Tests Based on Nearest-Neighbor Distances

Applications of tests for randomness in forestry and ecology require extensive fieldwork. The "patterns" are locations of trees or plants or eagle nests or towns. Mapping out the exact locations of all objects on a coordinate system is difficult

(a)



(b)

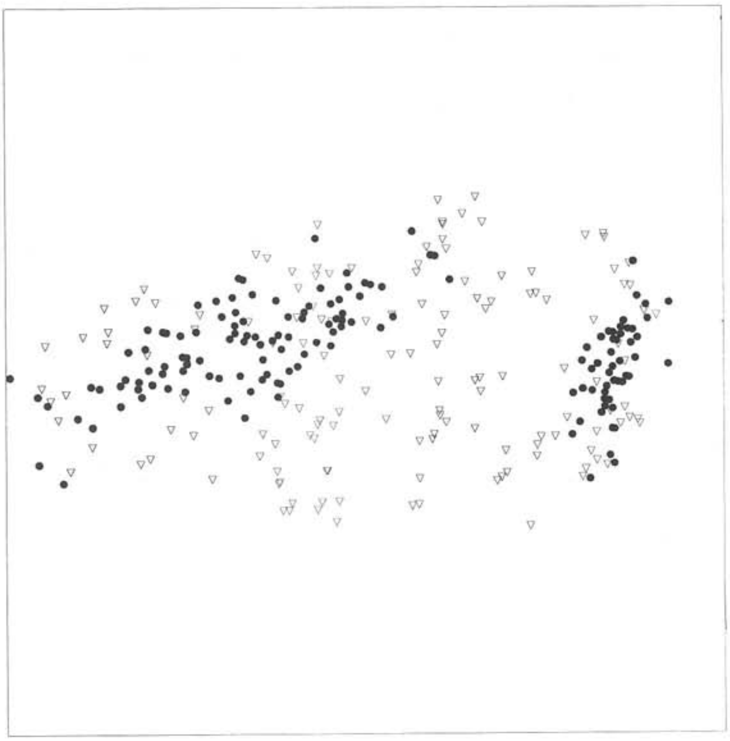
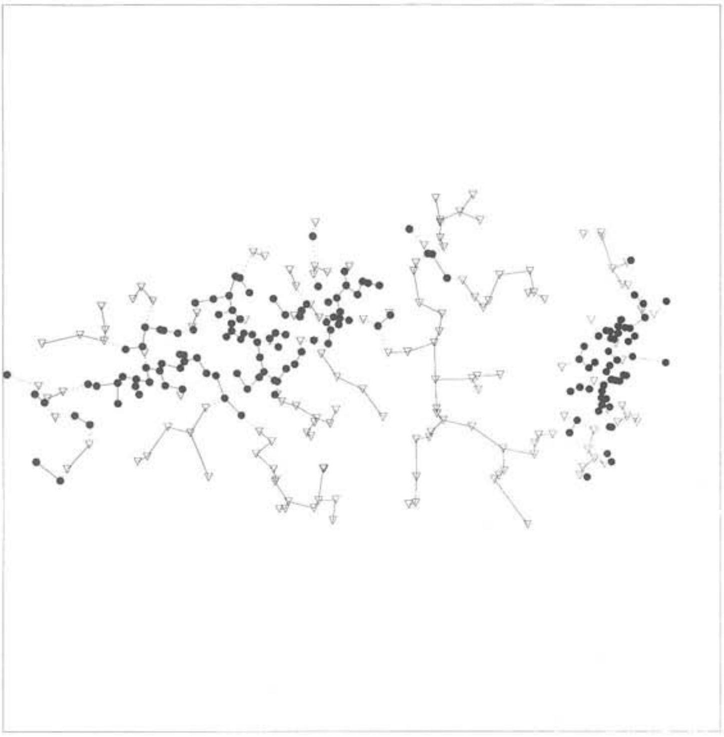


Figure 4.27 Example of MST-based statistic; (a) eigenvector projection of Iris data; (b) pooled sample containing patterns and points generated inside the convex hull of the patterns; (c) minimum spanning tree of the pooled sample.



(c)

Figure 4.27 (*continued*)

in such cases, but measuring the distance from each object to the closest object is relatively easy. Thus a number of tests for randomness have been based on nearest-neighbor distances and are called *distance methods*. Most of these tests use *sampling origins*, or points introduced artificially into the sampling window such as the points $\{\mathbf{y}_j\}$ in the MST-based test described above. These tests have been extended to d dimensions (Cross, 1980; Panayirci and Dubes, 1983) and are described in this section. Generally speaking, these methods provide quick and easy tests of H_0 but may not be the most powerful tests.

Pattern-to-pattern nearest-neighbor distances. Consider first tests of the random position hypothesis, H_0, based only on the distances from patterns to their nearest neighbors. Can H_0 be tested by comparing the NN (nearest neighbor) distribution for the given data to that under H_0? No analytical form for the NN distribution under H_0 exists because of the dependence among NN distances. Clark and Evans (1954) ignored the dependence and used an incorrect distribution for the sum of the NN distances under H_0.

Another possibility is to compare the histogram of NN distances for the patterns being examined to the NN distribution for a Poisson process, which is known from theory. Let U_k be the distance from any point in the feature space to the kth closest pattern. The random variable $\lambda V U_k^d$ can be shown to have a gamma distribution with parameter k. A Kolmogorov–Smirnov test can be used to compare the theoretical and observed distributions for some k. Cross (1980) noted that such a test would have little power. For example, consider a cluster process in which the points in each cluster are randomly distributed in a disk. The distribution of NN distances in the cluster process would match that in a Poisson process; only the intensity would be different. Other processes can be devised whose NN distributions match that for a Poisson process but which are not random. The sampling window cannot be assumed to be infinite in practical problems, so the gamma distribution is only an approximation that holds for points close to the center of the sampling window. We conclude that tests based entirely on NN distributions are impractical without Monte Carlo analysis.

Sparse sampling. The difficulties cited above have led to the use of sampling origins and sparse sampling. The objective here is to establish a test of the random position hypothesis, H_0, that is simple and intuitive, whose size (and threshold) can be derived from theory, which has good power against clustered alternatives, and which does not respond to variations in factors like sampling window estimates. Such a test does not require a Monte Carlo analysis to establish the threshold for the test. Tests based on the Hopkins (1954) and Cox-Lewis (1976) statistics are presented below which have some, but not all, of these properties. This approach has been summarized by Diggle (1983) in two dimensions and extended to d dimensions by Cross (1980), Cross and Jain (1982), and Panayirci and Dubes (1983).

Let $\{\mathbf{y}_j\}$ be m sampling origins placed at random in the d-dimensional sampling window and let $\{\mathbf{x}_i\}$ be the n patterns, $m << n$. Let U_j be the minimum distance from $\mathbf{y}_j$ to points in $\{\mathbf{x}_i\}$, $j = 1, 2, \ldots, m$. Suppose that m of the patterns $\{\mathbf{x}_i\}$ are randomly chosen for marking and let $\{W_j\}$ be the distances from the marked patterns to their nearest patterns as shown in Figure 4.28. The Hopkins statistic in d dimensions is

$$H = \frac{\Sigma\, U_j^d}{\Sigma\, U_j^d + \Sigma\, W_j^d}$$

All sums are from 1 to m. This statistic compares the nearest-neighbor distribution of randomly selected locations to that for the randomly selected patterns. On the average, distances from patterns to nearest patterns are smaller than distances from sampling origins to nearest patterns when the patterns are clustered because the sampling origins are selected uniformly. Thus values of H close to 1 suggest aggregation. Similarly, values close to zero suggest repulsion, or regular spacing. Values of 1/2 are expected under H_0 because near-neighbor distributions are the same from patterns and from sampling origins when the patterns are generated by a Poisson process.

When the patterns are generated by a Poisson process with intensity λ over the entire d-dimensional feature space, which is another statement of the random position hypothesis, and the near-neighbor distances are statistically independent, H has a beta distribution with parameters (m, m), independent of λ and d. The beta distribution follows from the fact that each near-neighbor distance has an exponential distribution and the sum of independent exponential distributions has a gamma distribution. Thus each of the two sums in H has a gamma distribution, if all near-neighbor distances are independent. Probability theory shows that when independ 1t gamma distributions are arranged as in H, the ratio has a beta distribution. The density function for this distribution is given below.

$$f_H(z \mid H_0) = \frac{z^{m-1}(1-z)^{m-1}}{B(m, m)} \qquad \text{if } 0 \leq z \leq 1$$

where

$$B(m, k) = \int_0^1 u^{m-1}(1-u)^{k-1}\, du = \frac{\Gamma(m)\Gamma(k)}{\Gamma(m+k)}$$

The fact that near-neighbor distributions are not the same at the edges of the sampling window as they are in the center distorts the distribution of H. The number of sampling origins, m, must be kept much smaller than the number of patterns, n, so as not to void the assumption of independence among near-neighbor distances. Zeng and Dubes (1985b) have extended H by measuring distance to the kth, rather than the first, nearest neighbor, but the results were not dramatically better than for first nearest-neighbor distances.

The Cox-Lewis (1976) statistic measures second-order information and is more powerful in detecting regularity than the Hopkins test under some circumstances. It is also more tedious to compute and less intuitive than the Hopkins statistic. The Cox-Lewis statistic is based on the ratio, R_i, of two distances. The first is U_i, the distance from the ith sampling origin to its nearest pattern, say $\mathbf{x}_i'$. The second distance is from $\mathbf{x}_i'$ to its nearest pattern. Panayirci and Dubes (1983)

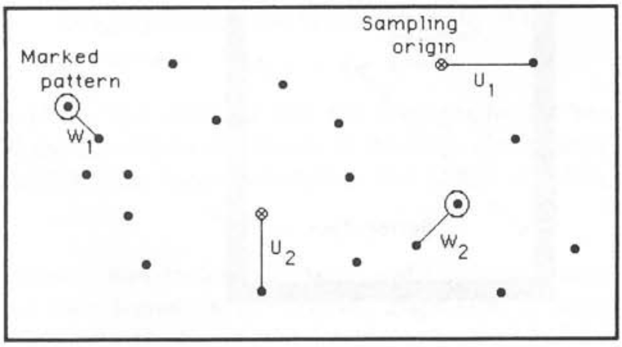


Figure 4.28 Distances used in Hopkins Statistic.

extended this to d dimensions. The ratio is normalized to have a uniform distribution over the unit interval under H_0. We test H_0 by testing whether a set of independent random variables $\{R_i\}$ are sampled from a uniform distribution. The average

$$R = \frac{1}{m} \sum_{i=1}^{m} R_i$$

has an approximate normal distribution with mean value 0.5 and variance $12m$ under randomness. Large positive values of R, measured on the scale of a standardized normal distribution, suggest regularity and large negative values suggest aggregation, or clustering. Values around zero correspond to randomness.

In fieldwork, as when the patterns are positions of trees or plants, the Cox-Lewis statistic is easier to measure than the Hopkins statistic and was proposed to simplify the gathering of data. In clustering tendency, the pattern matrix is stored in a computer, so all near-neighbor distances are readily available. We expect the Hopkins statistic to outperform the Cox-Lewis statistic. Our studies have confirmed this in almost all cases, with the exception of some cases of regular data.

Knowledge of the sampling window is very important in the successful application of distance methods. Patterns near the edges of the sampling window have different near-neighbor distributions than patterns in the center of the sampling window. Griffith (1983, 1985) reviews some remedies for these edge effects in a two-dimensional geographic context. One is to define a sampling frame, which is a region congruent to but inside the sampling window. The region between the sampling frame and the sampling window is called a buffer zone. Sampling origins are positioned inside the sampling frame, but NN computations are made to patterns inside the entire sampling window, including the buffer zone. The idea is to ensure that near-neighbor distances from origins and patterns inside the sampling frame are independent of location in the sampling frame. Figure 4.29 demonstrates a sampling frame.

Zeng and Dubes (1984) demonstrated that the probability mass for the Cox-

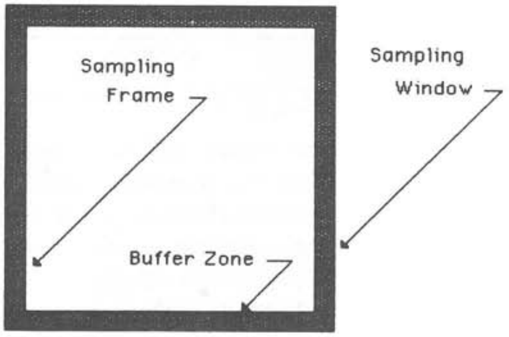


Figure 4.29 Example of sampling frame.

Lewis statistic R under the random position hypothesis flattened appreciably using a sampling frame when $d > 6$. The probability mass of R also shifted to the right from what was expected from theory. Some evidence suggested that the effect was caused more by dependence among distances between sampling origins and nearest patterns, as they are crowded together in high dimensions, than by edge effects. Dubes and Zeng (1987) demonstrated that these effects were evident for hypercubical and hyperspherical sampling windows and that the Hopkins statistic dominated the Cox-Lewis statistic in all tests of randomness against clustering. Dubes and Zeng (1987) examined a percentile frame, which is a hypersphere centered at the sample mean and covering 50% of the data. This sampling frame seems to be robust to the actual sampling window.

4.6.5 Ordinal Data

When the data occur as an ordinal proximity matrix, the random graph hypothesis serves as a null hypothesis, H_0, so all ordinal proximity matrices are taken to be equally likely. Various properties of graphs can lead to tests for randomness. No general model for clustering or regularity with proximity data have been adopted in the literature. Baker's (1974) basal taxonomies in Figure 4.9, perturbed by noise, can serve as an alternative hypothesis of structure for ordinal proximity data. Fillenbaum and Rapoport (1971) and Rapoport and Fillenbaum (1972) propose three statistics for use in tests of randomness.

Connectivity. Let V be the minimum number of edges needed to connect a graph on n nodes, one per object. Fillenbaum and Rapoport applied an asymptotic approximation to

$$P_{n,v} = P(V \leq v \mid H_0)$$

based on a result of Erdos and Renyi (1960). Schultz and Hubert (1973) showed that the asymptotic approximation was not accurate for small sample sizes. Ling (1975) and Ling and Killough (1976) derived an exact equation for $P_{n,v}$. Let $G(n, v)$ be the graph obtained by inserting v edges in the order dictated by the ordinal proximity matrix being examined. If v^* is the value observed for V in $G(n, v)$, the random graph hypothesis is rejected at size α if

$$P_{n,v*} > 1 - \alpha$$

The intuitive idea is that clustered data will have graphs that become connected only after all the edges between objects in the same cluster have been inserted. Large values of V thus suggest clustering. See Dubes and Jain (1979) for an example.

Node degrees and cycles. Rapoport and Fillenbaum suggested using the distribution of node degrees in the observed graph $G(n, v)$ with n and v fixed. The distribution of the number of edges incident to any node is known under H_0.

The number of edges incident to randomly selected nodes in $G(n, v)$ can thus be compared to what is expected under H_0. Choosing a value of v presents a practical difficulty. The number of cycles of order k in $G(n, v)$ can also be used in place of the node degree.

Number of components. A component of $G(n, v)$ can be interpreted as a maximal single-link cluster in $G(n, v)$. The number of components in $G(n, v)$ is thus the number of clusters in a single-link dendrogram at level v. Ling and Killough (1976) provide tables for $E(n, v)$, the expected number of components in an n-node graph having v edges under H_0. If the number of components, or single-link clusters, in $G(n, v)$ is significantly higher or lower than $E(n, v)$, one has evidence of nonrandomness.

4.7 SUMMARY

Assessing the validity of a clustering structure is a statistical problem. By clustering structure we mean either a hierarchy, a partition, or an individual cluster. A statistic is chosen, a null hypothesis of randomness is adopted, and a baseline distribution, or a distribution for the statistic under the null hypothesis, is derived. A threshold for the statistic is then selected from the baseline distribution to assure that the probability of rejecting the null hypothesis when it is true is not greater than a predefined level, α. The random graph and random permutation hypotheses have been widely adopted for ordinal data, while the random position hypothesis, corresponding to a Poisson process, has been applied when the objects are described by pattern matrices. One must pay close attention to the details and be sure that all assumptions are appropriate if a theoretical null distribution for the statistic is to be used. Otherwise, this distribution must be established with Monte Carlo methods.

The distinction between external and internal indices of cluster validity has often been blurred and confused in the literature. An external index evaluates a clustering structure in terms of prior information, such as category labels which have been assigned without reference to the pattern or proximity matrix. An internal index, on the other hand, uses only the proximity matrix itself and information from the cluster analysis. A relative criterion compares two statistics, clustering methods, or characteristics.

The clustering tendency problem has not received a great deal of attention but is certainly an important problem. One wants to believe that data are clustered and is naturally biased toward believing the results of a cluster analysis. Testing the data for randomness before actually clustering the data should help remove this bias. A number of tests were cataloged and literature references were provided.

The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.

5

Applications

Cluster analysis deals with automating a natural and commonly utilized human activity of forming classes or groups of similar objects, irrespective of their origin. Thus the objects to be clustered could be patients in a hospital, different brands of a consumer product, students at a university, different species of plants, books in a library, or pixels in a digital image. Cluster analysis has found applications in such diverse disciplines as biology, psychology, archaeology, geology, marketing, information retrieval, and remote sensing. Some interesting applications of clustering include clustering of orientations of fractures in a porphyry copper deposit (Shanley and Mahtab, 1976), comparing the performance of investment portfolios (Cohen et al., 1977), clustering of job analytic data (Zimmerman et al., 1982), developing relatively homogeneous groups of industrial companies (Chen et al., 1974), clustering of sculptures found in Indian temples of the seventh and eighth centuries A.D. (Siromoney et al., 1985), clustering of collinear line segments in digital images (Scher et al., 1982), cluster analysis for studying race mixture in human populations (Rao, 1977), and clustering for automatic indexing and document classification (Garland, 1983; Salton et al., 1975).

The use of clustering in computer science and engineering applications has been relatively recent. Cluster analysis plays an important role in solving many problems in pattern recognition and image processing. Cluster analysis is used for feature selection (Jain and Dubes, 1978), in numerous applications involving unsupervised learning [where it is difficult to assign a reliable category label to the training patterns as in Sanderson and Wong (1980)], in grammatical inference (Lu and Fu, 1978; Wang, 1984), speech and speaker recognition (Rabiner and

Wilpon, 1979), work-load characterization (Agrawala et al., 1976), image segmentation (Hoffman and Jain, 1987), and image registration (Stockman, 1980). Some of these applications are summarized in the chapter by Dubes and Jain (1980). A large number of papers have been written on the clustering of multispectral data arising in remote sensing. The recently proposed technique of conceptual clustering (Michalski and Stepp, 1983) is touted as being capable of aiding machine learning and knowledge representation in artificial intelligence (Cheng and Fu, 1984). Because of our backgrounds, and because these applications have not been adequately described elsewhere, we choose to discuss several examples from image processing. Our discussion will focus mainly on the problem description and the results, but not on the details of methodology. Applying clustering algorithms to a specific problem domain and interpreting the results require a thorough familiarity with the domain. Thus we first review some fundamentals of digital images and processing techniques.

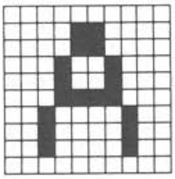
5.1 IMAGE PROCESSING

Digital image processing, or simply image processing, turns images or pictures taken from a camera into usable data with the help of digital computers. The input images can be obtained by a variety of sensors depending on the environment. For example, multispectral scanners on Landsat satellites provide images (in several electromagnetic bands) of the earth's surface, x-ray sources give images of human bodies, and laser scanners provide "range" images in industrial environments. Visible images have numerous applications, including optical character recognition. A typical image processing application consists of three stages:

1. Image acquisition and digitization
2. Processing and quantitatively determining features in an image
3. Making decisions based on the available features

The phrases "picture processing," "machine vision," "computer vision," "scene analysis," and "image understanding" also refer to this activity of processing and interpreting images using digital computers. Several textbooks cover the fundamental issues in image processing (Rosenfeld and Kak, 1976; Gonzalez and Wintz, 1977; Ballard and Brown, 1982).

Two main reasons for processing images with a computer are (1) to improve image quality and facilitate human interpretation (we all remember seeing the processed images of Saturn and Uranus on network television), and (2) to automatically classify various "objects" present in the image. For example, in remote sensing, trained photointerpreters delineate homogeneous regions in satellite images and classify them into various land-use categories such as forest, water, urban area, and agricultural fields. The inherent noise and the variability present in



(a)

```
0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 1 0 0 1 0 0 0
0 0 0 1 0 0 1 0 0 0
0 0 0 1 1 1 1 0 0 0
0 0 1 0 0 0 0 1 0 0
0 0 1 0 0 0 0 1 0 0
0 0 1 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0
```

(b)

Figure 5.1 (a) Binary image of character A on a 10 x 10 grid; (b) representation of binary image as a matrix of 0's and 1's.

these images and the large number of such images to be classified make image processing techniques very suitable for this application (Hord, 1982). In many instances the machine classification of remotely sensed images is more accurate than manual photointerpretation.

The image of a scene that is to be interpreted or analyzed is acquired by sensors. This ''analog'' or continuous image is digitized and the light intensities in the image are quantized to a finite number of gray levels which are entered into a digital computer. A digital image is represented as a $N \times N$ matrix of pixels or picture elements, $[f(i, j)]$, $i, j = 1, 2, \ldots, N$. The intensity value $f(i, j)$ takes one of G different gray values. For example, Figure 5.1 shows a 10×10 binary image of the character A and its representation as a binary image, or matrix of 0's and 1's. A value of 1 in this matrix indicates that the corresponding pixel is a part of the object. The term ''spatial resolution'' refers to the size of the pixel and ranges from tens of meters in remote-sensing applications to micrometers in biomedical cell analysis. The digitized and quantized image is enhanced to reduce noise degradation. Digital high-pass filters sharpen an image and low-pass filters eliminate high-frequency noise. Gray-level histogram modification procedures enhance an image. At the next stage, pixels with similar properties are grouped to form regions or segments in an image. Regions are described by properties like shape, area, texture, and color. Relationships among the various regions (such as adjacency, left, right, and surround) are then used to recognize the objects and interpret the scene. These stages are not necessarily independent and quite often a feedback mechanism is employed to improve the results.

5.2 IMAGE SEGMENTATION BY CLUSTERING

Image segmentation is a very critical component of an image processing system because errors at this stage influence feature extraction, classification, and interpretation. Image segmentation is also closely related to the clustering problem, so we will discuss this topic in some detail. The problem of image segmentation can be stated as follows: Partition a given image into regions or segments such that pixels belonging to a region are more similar to each other than pixels belonging to different regions. We also require that these regions be connected so a region consists of contiguous or neighboring pixels. It is generally assumed that the interior of the region has uniform intensities. What criterion of similarity should be used in grouping pixels? How many segments are in the image? These questions emphasize and illustrate the similarity between an image segmentation problem and a typical clustering problem.

A large number of image segmentation techniques are available in the literature (Fu and Mui, 1981). These techniques are based on one of the following three approaches: (1) thresholding or clustering, (2) boundary detection, and (3) region growing. We concentrate on the clustering approaches. The choice of a segmentation technique is data dependent. Image segmentation has the same relationship to image classification that unsupervised learning has to supervised learning in pattern recognition. Image segmentation is a more difficult problem than image classification. First, the number of classes is specified in advance in classification problems, whereas in segmentation problems the number of classes is unknown. A segmentation algorithm must include some means for determining the appropriate or the actual number of classes from the data. Second, the objects being classified in classification problems are subimages. In segmentation problems, the objects to be classified are individual pixels.

The simplest case of image segmentation either has a "dark" object or region on a "light" background, as in the image of a printed text on a white paper, or a light object on a dark background. The pixels belonging to the dark object can be isolated from the background pixels by a simple thresholding operation on the intensity values. Pixels with intensity or gray values greater than a threshold t belong to the object, whereas the remaining pixels belong to the background. How should t be selected? A common technique detects peaks and valleys in the gray-level histogram. A gray-level histogram of an image is a plot of the number of pixels versus the gray level. Other global as well as local methods for segmentation are also available (Weszka, 1978). Note the similarity of this procedure to one-dimensional partitional clustering using density estimation or mode seeking (Section 3.3.5); each pixel is a pattern and the gray value is the only feature used to group the pixels.

The concept of one-dimensional gray-level histogram can be extended to several dimensions if we view each pixel as being represented by a d-dimensional feature vector. The number of features needed depends on the complexity of the image. The components of the feature vector may be gray values through different filters, texture measures, and gradient magnitudes and directions. For example, if a color image of the scene is available, the intensity at each pixel can be decomposed into three primary colors (red, green, and blue), and each pixel can be viewed as a point in this three-dimensional feature space (Ohlander et al., 1979). A clustering method can group the pixels in the feature space into clusters. These clusters are then mapped to the spatial domain to display the segmented image. The size of an image in a typical industrial application is 256 pixels by 256 pixels (denoted as 256×256), so the number of pixels to be clustered is quite large. To reduce the computational burden only pixels in, say, every fourth row and column are clustered. The remaining pixels are then assigned to the nearest cluster center.

Because of the size of the pattern matrix and the need for a single partition, partitional clustering algorithms are more popular for image segmentation than hierarchical algorithms. Jain et al. (1980) have used hierarchical clustering to segment muscle cell images.

The following sections illustrate the use of clustering in the segmentation of textured images, range images, and multispectral images. Instead of presenting a review of the vast literature available on segmentation using clustering, we discuss three studies in detail.

5.3 SEGMENTATION OF TEXTURED IMAGES

Texture is a property of the spatial distribution of gray levels or the overall pattern of gray-level changes in an image. When an image does not portray any particular object or form, only certain aspects of the overall pattern of gray-level changes in the image are perceived (Julesz, 1975). Quite often, a region in an image can be distinguished from the others based on the differences in their textures. For example, different land-use categories (orchard, residential, water, forest) in remote-sensing applications have different textures. Note that two adjacent textured regions in an image may not be separated by an edge or a sharp transition in gray levels. Thus textural information is utilized in both image segmentation and classification tasks.

How do we define and measure textural information? Textural qualities are typically expressed by adjectives such as coarse, streaked, sharp, irregular, fine, cellular, rippled, and directional. Several statistical, structural, and modeling approaches to texture have been proposed in the literature which attempt to quantify these textural adjectives. Some common approaches are based on image autocorrelation, power spectrum, co-occurrence matrices, primitive placement rules, and Markov random fields (Ballard and Brown, 1982). Here we will discuss the textural features derived from an implementation of the channel-filtering model of the human visual system (Coggins, 1982; Coggins and Jain, 1985).

Neurological and psychophysical experiments support the hypothesis that the analysis of a stimulus by the visual system might involve a set of quasi-independent mechanisms, called channels, which could be conveniently modeled as filters. Each channel responds to gray-level changes over regions of different size or at different orientations, so each filtered image contains limited spectral information from the original image. The shapes, sizes, locations, and number of channels are based on psychophysical and neurological data and the size of the image to be analyzed. Gray levels in each filtered image can be interpreted as representing the spectral energy arising from small areas in the original image. This interpretation motivates the definition of a texture energy feature which measures the spread of the gray-level frequency histogram of the filtered images. Coggins (1982) has used this approach for texture classification and segmentation.

Spatial filtering transforms the texture segmentation problem into a decision problem in a feature space; corresponding to each pixel there is a feature vector in which the number of features equals the number of channels used. We now have to assess the structure of these n patterns in d-dimensional space, where n is the number of pixels in the image and d is the number of texture features. This assessment can be made by applying a partitional clustering algorithm. Several other studies (Mitchell and Carlton, 1978; Schachter et al., 1978; Coleman and Andrews, 1979; Davis and Mitiche, 1982) have used clustering in texture segmentation problems with texture features different from those used above. Note that this pattern matrix representation of segmentation problem is independent of which texture features are used.

Coggins (1982) used the CLUSTER (see Section 3.3.3) algorithm to organize the pixels. CLUSTER was selected because it does not require the user to specify parameters. CLUSTER partitions the data with the number of clusters going from two to a user-specified bound (eight in this study). Each partitioning of the data corresponds to a segmentation of the given image. CLUSTER provides several statistics which can be used to qualitatively assess the validity of a clustering (Section 3.3.3). One statistic, S_k, which measures the "validity" of cluster k and takes into account the compactness and the isolation of the cluster, is defined below:

$$S_k = \frac{\min\limits_{\substack{l \\ l \neq k}} \sum\limits_{j=1}^{d} (m_j^{(k)} - m_j^{(l)})^2}{\left\{ \dfrac{1}{n_k} \sum\limits_{i=1}^{n_k} \sum\limits_{j=1}^{d} (x_{ij}^{(k)} - m_j^{(k)})^2 \right\}^{1/2}}$$

where n_k = number of patterns in cluster k,
$m_j^{(k)}$ = cluster center for cluster k, along feature j
d = the number of features,
$x_{ij}^{(k)}$ = value of the jth feature for the ith pattern belonging to cluster k.

Large values of S_k suggest compact, well-isolated clusters. An acceptable clustering is defined as one in which the value of S_k exceeds a threshold for all clusters. Coggins (1982) determined an empirical threshold of 1.70 for S_k by tuning the threshold in preliminary experiments to yield approximately the same number of clusters as perceived by human subjects. Determining the null distribution of S_k is an open research problem (see Chapter 4). The threshold value specifies that for each cluster the minimum distance to another cluster center be greater than 1.70 times the average within-cluster distance. The clusterings that are accepted by this criterion are ranked by the average value of S_k over all clusters weighted by the number of points in the clusters. Clusterings with high average S_k values are preferred.

Coggins's segmentation experiments used 128×128 images. Segmenting a 128×128 image by clustering requires the algorithm to process 16,384 patterns. To reduce computational requirements, Coggins clustered only 64 pixels spaced 16 rows and 16 columns apart. This sampling assumes that the textured regions are not irregularly shaped or very small in area. Coggins used eight channels, the four orientation channels and four spatial frequency channels with center frequencies of 4, 8, 16, and 32 cycles per image width to compute texture features. This results in a 64×8 pattern matrix for clustering. The resulting cluster centers are used to define a minimum distance classifier (Duda and Hart, 1973) to classify the remaining pixels. The classification results are displayed as a segmented image in which gray levels denote the cluster labels assigned to each pixel.

Now we show clustering results for a texture segmentation problem. The 128×128 binary image shown in Figure 5.2 contains two artificially generated textures. The left half of the image contains a regular texture in which the dots are separated by three pixels from their horizontal and vertical neighbors. The other texture is a random dot pattern in which the probability that each pixel is black is independent of the other pixels and is approximately equal to 1/9. Segmented images containing two, three, and four clusters are shown in Figure 5.3. Pixels belonging to the same cluster have been assigned the same gray level. Remember that the pixels have been clustered in an eight-dimensional space. The actual gray levels in the segmented images have no significance other than to distinguish the regions. The two-cluster segmentation shown in Figure 5.3(a) correctly identifies the two textured regions present; over 98% of the pixels in the image are correctly labeled. Segmented images with more than two clusters [Figure 5.3(b) and (c)] subdivide the random texture region. The regular texture region is not split. Further, none of the segments lie across the boundary between the textures. These results indicate that the texture features used in the clustering are able to identify regions of uniform textures.

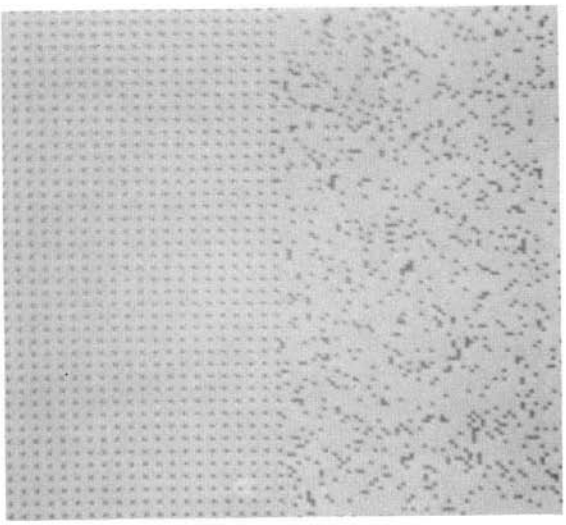
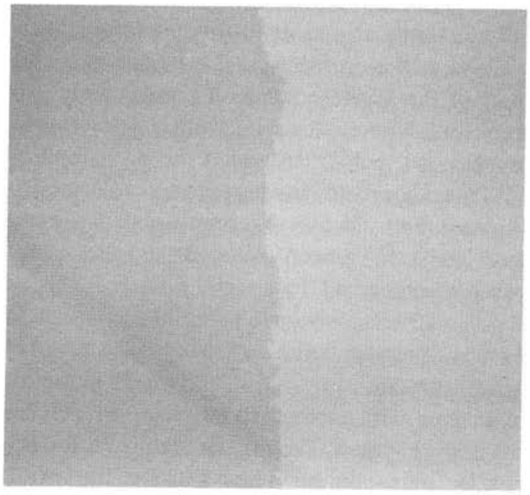
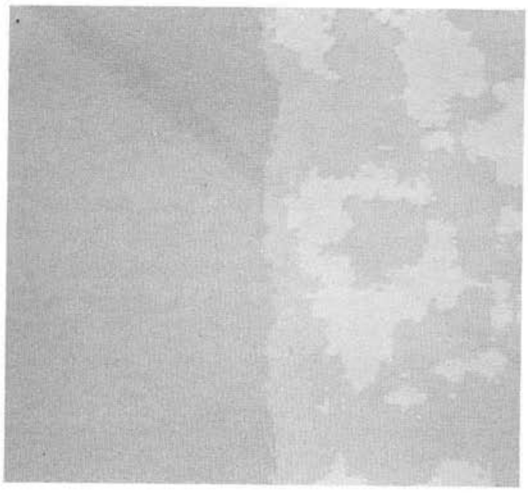


Figure 5.2 Dot texture composite. (Source: Reproduced with permission from Coggins and Jain, 1985.)

Which of the segmentation results is correct? The validity test defined earlier is summarized in Table 5.1, which shows that only the two-cluster solution is accepted. Note the high value of S_k for cluster 1, which corresponds to the regular texture. Since the cluster corresponding to the regular texture is always compact and well isolated, the regular texture is never subdivided in the segmented images.



(a)



(b)

Figure 5.3 Segmentation of dot texture of Figure 5.2: (a) two-cluster solution; (b) three-cluster solution; (c) four-cluster solution. (Source: Reproduced with permission from Coggins and Jain, 1985.)

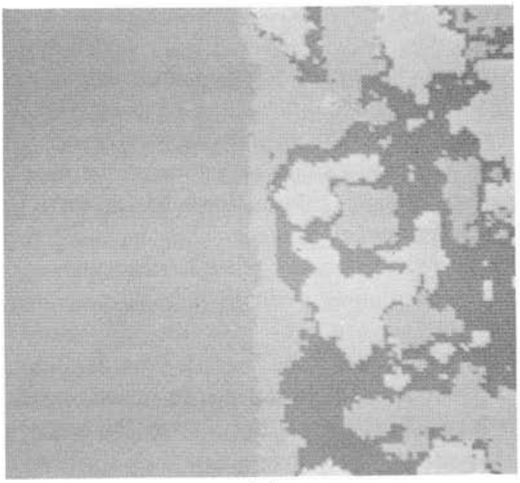

(c)

Figure 5.3 *(continued)*

Coggins (1982), Coggins and Jain (1985), and Jain (1985) report on several other texture segmentation experiments using this approach. These results demonstrate that clustering methods are useful for identifying textured regions in digital images.

TABLE 5.1 Evaluation of Clusterings on Dot Textures

Two-cluster solution		
k	n_k	S_k
1	32	6.93
2	32	2.31
Average value of $S_k = 4.62$[a]		
Three-cluster solution		
k	n_k	S_k
1	32	6.17
2	19	1.41
3	13	1.24
Average value of $S_k = 3.76$		
Four-cluster solution		
k	n_k	S_k
1	32	6.08
2	20	1.48
3	8	1.15
4	4	2.20
Average value of $S_k = 3.78$		

[a] Accepted segmentation. (Source: Reproduced with permission from Coggins and Jain, 1985.)

5.4 SEGMENTATION OF RANGE IMAGES

One of the important and challenging problems in image processing and computer vision is the recognition of three-dimensional objects. Three-dimensional object recognition is inherently more difficult than two-dimensional recognition because we usually have access to only a single two-dimensional view of the three-dimensional object. This loss of depth information leads to ambiguous interpretations. For example, the recognition of printed characters on a page or flat machine parts placed on a conveyor belt is, in general, easier than the identification and location of obstacles present along a robot's path as it navigates around a factory floor.

Several types of constraints have been utilized in computer vision research to determine the depth from a single reflectance image, where the pixel value is proportional to the intensity of light reflected from the corresponding location on object's surface. Monocular cues such as shape from shading, shape from texture, and shape from motion are intuitively appealing but difficult to implement (Barrow and Tenenbaum, 1978). For example, the basis for shape from shading is that the reflectance equation provides one constraint on the surface orientation and another constraint is provided by the heuristic requirements that the surfaces of the scene are smooth. The density gradient of texture primitives provides slant, slope, and range information about the surface in the image. Light stripping is a relatively simple monocular method where the coordinates of object surface in three dimensions are determined by calculating the intersection of the "line of sight" with a plane sheet of projected light. A popular binocular method obtains depth or range information of a point using triangulation given its image point in each of two views. The difficulty with this stereo-based technique is in deriving the correspondence between image points in the two images. Several of these techniques are reviewed in the book by Ballard and Brown (1982).

Laser rangefinders provide depth values directly; we call the output of such sensors range images. For example, time-of-flight rangefinders measure the elapsed time between emitting a laser beam and detecting its reflected energy from an object surface. The gray values in a range image correspond to depth directly rather than to reflected light. Lighter gray values in a region of a range image indicate that it is closer to the sensor or the viewer. Laser rangefinders are gaining popularity in the computer vision community. Ideally, one would like to use both the range image and the traditional reflectance images for scene interpretation.

How do we utilize range images to analyze a given scene? One approach that appears promising can be summarized as follows (Ittner and Jain, 1985; Hoffman and Jain, 1987). We can easily derive the orientation (surface normal) at each point of the visible surfaces of an object from the depth values available in the range images. The three-dimensional coordinates and surface normal information are used to segment the range image into "surface patches." A partitional clustering algorithm is used for segmentation. The next stage classifies these patches as planar, convex, or concave based on a statistical test for trend, curvature values, and eigenvalue analysis. Compatible patches can be merged to produce reasonable faces of the object. The final stage matches objects in the scene to their stored models. Since the objects in this approach are described in terms of the surface patches, the stored models are also in terms of the surface representation.

Here we report only on the application of clustering to the segmentation problem. The results on classification of surface patches and modeling are reported elsewhere (Hoffman and Jain, 1987). As we discussed earlier, clustering has long been used to segment intensity images and seems a reasonable method to segment range data. The objective is to use the clustering so that points on different faces of the object are put in different clusters.

5.4.1 Features for Clustering

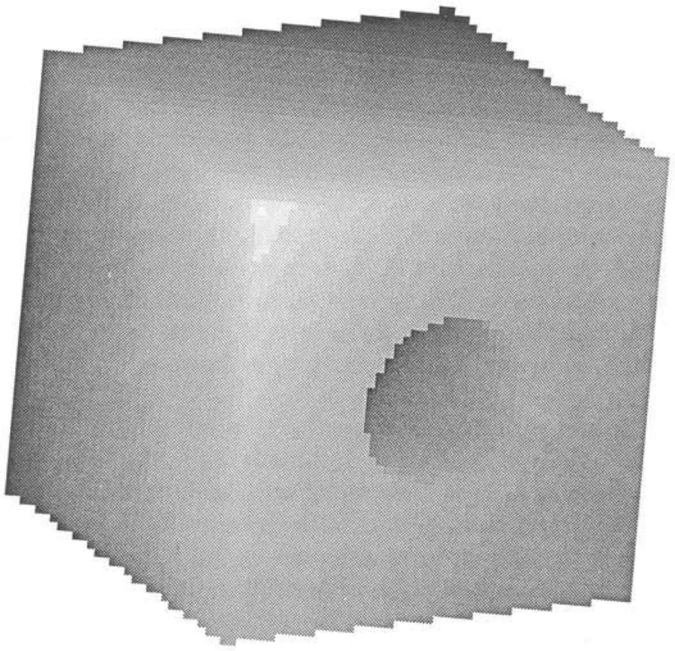
An important issue in any clustering application is that of deciding what features should be used to establish similarity between pixels. Some candidate features at a pixel are the spatial coordinates x, y, the range or depth value, z, the surface normal, the coefficients for the best-fitting quadric surface, and curvature measures. In our clustering experiments we used the coordinate features x and y, the depth value z, and the estimated surface normal vector resulting in six-dimensional pattern vectors. The unit normal vector represents the orientation of a surface at a pixel and is estimated by finding the best-fitting plane (by linear least-squares method) over a small neighborhood of the pixel.

Each one of the six features plays a distinct and important role in segmentation. The x and y features help to provide connected segments. The depth value z is important in the detection of "jump" edges, and the normal vector is needed to detect "crease" edges since the vector (x, y, z) does not experience abrupt changes over such a boundary. Jump edges in range images are formed where depth values are discontinuous. Crease edges correspond to points over which surface normals are discontinuous. Since we do not wish inequalities of scaling to give undue precedence to some feature(s) over the others, we preprocess the data so that each feature has unit variance.

5.4.2 Clustering Results

We have applied a number of different clustering techniques to segment range images for comparative evaluation, including MST-based approaches which cut inconsistent edges (Section 3.3.6), the mutual nearest-neighbor clustering algorithm (Section 3.3.7), single-link and complete-link hierarchical clustering methods (Section 3.2), and several square-error clustering algorithms. We have had the most success with the CLUSTER algorithm (Section 3.3.3).

Figure 5.4 shows the segmentation of a synthetic range image. For ease in display, the maximum number of clusters in the CLUSTER program has been set to 12. Each cluster has a unique gray level. Due to the memory and time requirements of CLUSTER only 1000 pixels of the 128×128 range images


(a)

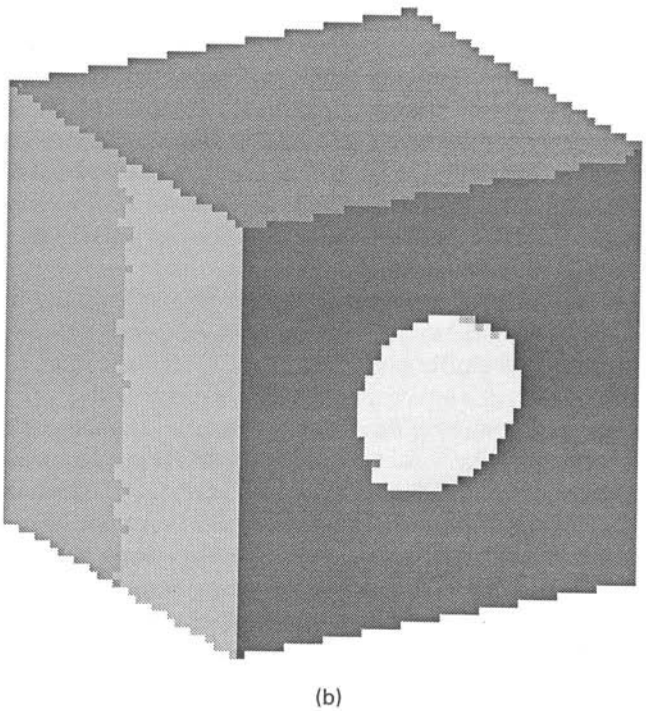

(b)

Figure 5.4 (a) Synthetic range image; (b) segmentation of range image into five clusters.

were clustered. The remaining pixels were assigned to the cluster with the closest center.

The results of these experiments on segmentation are reasonable. The clustering approach segmented the distinct object faces quite well. In some cases clusters are not connected in the image. This happens when the normal vectors of two faces on an object in an image are very similar. We found that it is important to direct the clustering algorithm to generate more clusters than the number of faces actually present in the object. For example, if there are three distinct faces in the object, a clustering with 6 to 10 clusters usually provided the best segmentation. Individual object faces were broken up during segmentation, but in general, each of the clusters contained points of a single face. The merging procedure described in Hoffman and Jain (1987) rectifies this problem.

5.5 SEGMENTATION OF MULTISPECTRAL IMAGES

Clustering techniques have been very popular in remote-sensing applications (Narendra and Goldberg, 1977; Goldberg and Shlien, 1978; Bryant, 1979; Ince, 1981; Wharton, 1983; Gowda, 1984). Multispectral scanners on LANDSAT satellites sense the reflected energy from the earth's surface in several different bands, or wavelengths of the electromagnetic spectrum (Hord, 1982). A pixel in these multispectral images represents the smallest area on earth's surface that can be identified or distinguished from the neighboring areas. The pixel size and the number of bands varies depending on the scanner; LANDSAT 4 has seven bands with a pixel size (resolution) of 28.5 meters $\times$ 28.5 meters. The wavelengths of these bands ranges from 0.45 to 2.35 μm. The wavelength interval associated with each band is tuned to a particular ground cover category. For example, the green band in the visible portion of the electromagnetic spectrum (0.52 to 0.6 μm) is useful for identifying areas of shallow water, such as shoals, and reefs, whereas the red band (0.63 to 0.69 μm) emphasizes urban areas (Hord, 1982).

LANDSAT images have been used for land-use planning, identifying locations of mineral deposits, studying weather pattern, forest inventory, and estimating crop yield. Pattern recognition and clustering techniques are particularly useful in remote sensing to classify or group pixels. In many applications of remote sensing, the a priori labels (ground truth) on the pixels are not available, and therefore clustering techniques are needed to group pixels based on the observed reflectance in various bands of the multispectral scanner. This is referred to as clustering of pixels in the spectral domain. Thus all the pixels in the image that represent water bodies should have similar spectral characteristics which will be different from pixels representing, say, urban areas. Spatial information about the pixels is also utilized in the clustering. For example, neighboring pixels in a two-dimensional image are more likely to belong to the same cluster than pixels that are farther apart. Similarly, contextual clues such as "a pixel surrounded by urban areas should not be labeled as forest" are also incorporated either in the grouping algorithm or in the postprocessing stage.

We present an example to demonstrate that clustering algorithms can identify clusters in multispectral images which correspond to land-use categories. The data for this example are taken from a seven-band LANDSAT image, acquired on October 18, 1982, of Fredrick Township, Crawford County, located in mid-Michigan. Band 6, or the thermal band, was not used, so the clustering was done in six dimensions. A 256 $\times$ 256 image was extracted for clustering. Figure 5.5 shows the band 3 image of this area. Based on the Geographical Information System (GIS) maintained by the Department of Natural Resources, State of Michigan, there are four major land-use categories present in this image: urban area and bareland, water, coniferous forest (needlelike, dense trees), and deciduous forest and shrub (green vegetation cover). This GIS information was obtained from a manual air photo interpretation done in 1978. These four major land-use categories can be detected in Figure 5.5, although they are better seen in pseudocolor images. One of the goals of remote sensing is to update the GIS with the help of LANDSAT images.

A square-error clustering program was used to segment this LANDSAT

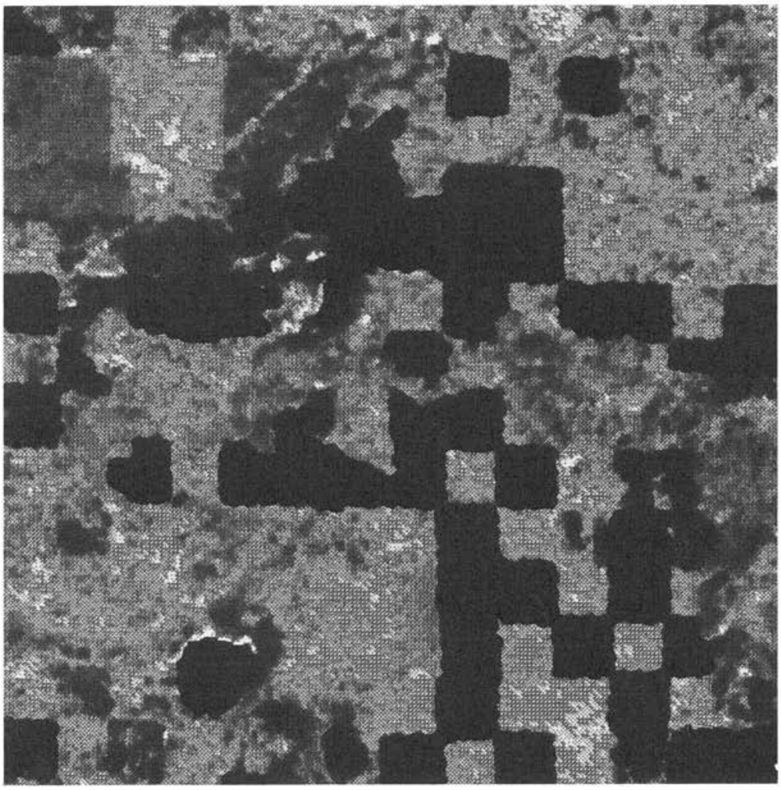


Figure 5.5 256 x 256 LANDSAT band 3 image of Fredrick Township, Michigan.

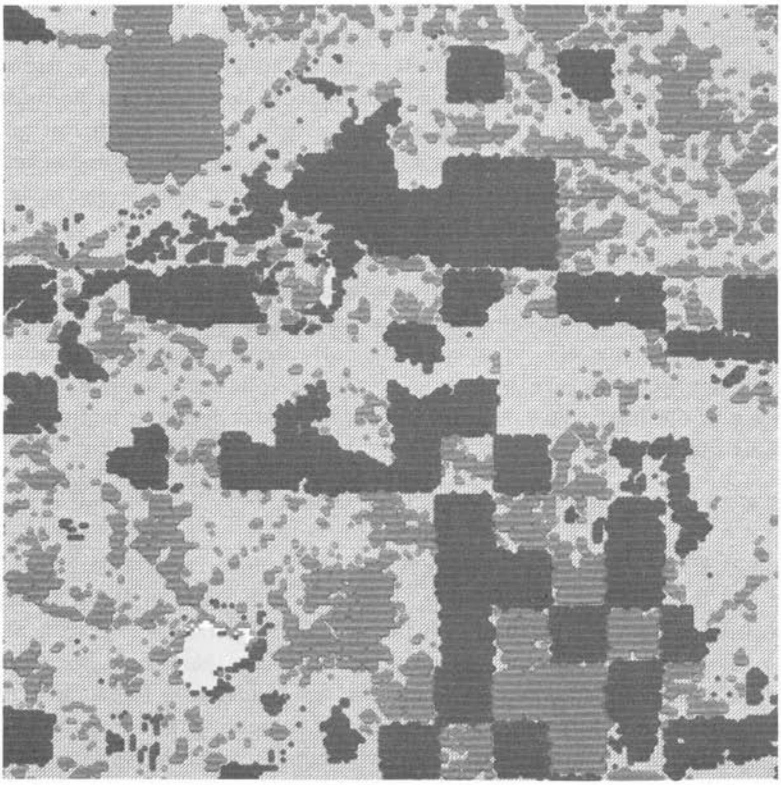


Figure 5.6 Clustering of multispectral LANDSAT image shown in Figure 5.5.

image. This clustering program is part of the image analysis software package on an ERDAS (Earth Resources Data Analysis System) microcomputer system. The four-cluster solution is shown in Figure 5.6. Four different gray levels are used to show the four clusters of pixels. Clusters match quite well with the land-use categories obtained from the GIS data base. Note that the clustering was done in six dimensions and no spatial information was used. Square-error clustering techniques have been very popular in remote-sensing applications. NASA Goddard Space Flight Center has implemented the ISODATA clustering algorithm on its Massively Parallel Processor (MPP) to be used for remote-sensing applications (see Section 3.3.2).

5.6 IMAGE REGISTRATION

Image registration deals with the problem of matching two images of the same scene taken under different conditions. In other words, given a reference or a prototype image (often called a template) of an object, locate the position and orientation of this object in a larger image. The general image registration problem involves geometrical (rotation, translation, and scale) and sensor (infrared, radar, and visible band) variations. A transformation function must be determined so that the coordinates of a point in one of the images can be found from the coordinates of the same point in the other image. Image registration is a difficult problem, but it has numerous important applications, such as detecting ecological and environmental changes based on comparing several images of the same area, combining information derived from different sensors, and object detection. Here we will concentrate on the problem of object detection.

The simplest approach to object detection is template matching or correlation (Li and Dubes, 1985). Given a template or a prototype of an object, which is really a small image $w(x, y)$ of size $M \times M$, the problem is to locate it in a given image $f(x, y)$ of size $N \times N$, $N > M$. We place the template at different locations in the image and measure the similarity between the template and the image as follows:

$$R(m, n) = \sum_{x} \sum_{y} f(x,y)\; w(x - m, y - n)$$

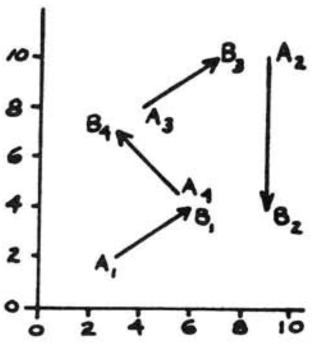
Note that $f(x, y)$ and $w(x, y)$ represent the gray values at location (x, y) in the image and the template, respectively. The maximum value of $R(m, n)$ indicates the position where $w(x, y)$ best matches $f(x, y)$. Often, the similarity measure $R(m, n)$ is normalized by average image and template intensities. Sequential and hierarchical search techniques are employed to reduce the computation time of matching (Ballard and Brown, 1982). Instead of computing the correlation between the template and the image based on gray values, matching can be performed using image and model (object) structures such as edge features and invariant moments.

We describe a technique for object detection via image registration which is a hybrid of template matching and structural analysis (Stockman, 1980). Template matching is tolerant to noise but expensive to compute. Structural techniques recognize objects as a synthesis of parts and are relatively efficient to compute, but are sensitive to imperfect or noisy data. The object detection technique of Stockman (1980) estimates the global transformation needed to map the features derived from the given image to similar features in object model. Following are the two main steps of this matching procedure.

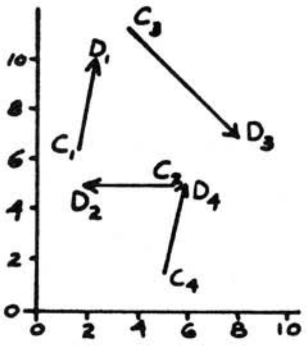
1. Let (f_i, f_m) be a pair of image and model features. Compute the transformation parameter vector **a** such that the transformation $T_{\mathbf{a}}$ maps f_i to f_m. Represent this vector **a** in the parameter space. We are assuming that all features of the same type correspond. For example, straight lines in the image correspond to straight lines in the object model.
2. All possible transformations between the image and the model are now represented as clusters in the **a**-space. Find the best transformation by identifying the densest cluster.

Note that each point in **a**-space represents evidence of a local match. Clusters in **a**-space represent possible global matches.

Stockman uses three types of features for registration: straight-edge elements, points of sharp curvature, and points of angular intersection. Model edges are constructed by human beings and image edges whose lengths are assumed to be accurate are obtained by computer. Only rotation and translation will be allowed, so $\mathbf{a} = (\theta, \Delta x, \Delta y)$. Figure 5.7 shows this procedure for a simple example (Stockman, 1980). First an edge AB in the image is rotated to get an edge $A'B'$ which is parallel to a model edge CD. The translational part $(\Delta x, \Delta y)$ of **a** is constrained because the lengths of corresponding edge structures are forced to agree. Out of a total possible $4 \times 4 = 16$ edge elements pairings in Figure 5.7, six are eliminated



(a)



(b)

i	j	A_i	B_i	C_j	D_j	Θ	ΔX	ΔY	
1	1	3.0,2.0	6.0,4.0	1.7,6.4	2.3,10.0	0.82	1.1	2.8	
1	2	3.0,2.0	6.0,4.0	5.3,5.0	1.8, 5.0	2.55	8.9	5.0	
1	4	3.0,2.0	6.0,4.0	5.1,1.5	5.8, 5.0	0.79	4.4	-2.0	*
2	3	9.0,10.0	9.0,4.0	3.7,11.2	8.0, 7.0	0.79	4.4	-2.2	*
3	1	4.0,8.0	7.0,10.0	1.7,6.4	2.3,10.0	0.82	4.8	-2.0	*
3	2	4.0,8.0	7.0,10.0	5.3,5.0	1.8,5.0	2.55	13.1	9.4	
3	4	4.0,8.0	7.0,10.0	5.1,1.5	5.8,5.0	0.79	7.9	-7.0	
4	1	5.5,4.5	3.0,7.0	1.7,6.4	2.3,10.0	5.33	-5.2	8.3	
4	2	5.5,4.5	3.0,7.0	5.3,5.0	1.8,5.0	0.79	4.6	-2.1	*
4	4	5.5,4.5	3.0,7.0	5.1,1.5	5.8,5.0	5.30	-1.7	3.6	

(c)

Figure 5.7 Examples of global registration via clustering of local evidence. Edges in (a) need to be rotated $\pi/4$ and then translated $(4.5, -2.0)$ to register with edges of (b) 10 points are in the cluster space (c) forming a cluster at ($\theta = 0.79$, $\Delta x = 4.5$, $\Delta y = -2.0$). (Source: Reproduced with permission from Stockman, 1980.)

due to length inconsistency. The three-dimensional parameter space **a** has 10 points and there is a "strong" cluster of size 4 which gives the correct match. In real-world matching problems the parameter space will have hundreds of points. This necessitates the use of a clustering algorithm.

This registration procedure has been successfully applied to several industrial and satellite images (Stockman, 1980). The main computational burden lies in image feature extraction. The clustering procedure used is fairly simple and looks for peaks in the quantized parameter space obtained by inserting each of the (θ, Δx, Δy) values into bins. A similar technique, called the Hough transform, is popular in image processing to detect lines or other shapes in a given image (Ballard and Brown, 1982). Stockman has recently extended this matching scheme to determine the "pose" of a three-dimensional object from multiple views (Stockman and Esteva, 1985). Clustering is now accepted as a powerful technique for object recognition in computer vision.

5.7 SUMMARY

Clustering techniques were first developed in biology and zoology to group similar animals and plants to construct taxonomies. The need to organize vast amounts of data into "meaningful" groups, clusters, categories, partitions, or classes in several scientific disciplines has made clustering a valuable tool in data analysis. A variety of entities or objects have been clustered, including mental diseases, land-use patterns, rock samples, fingerprints, training methods, stars, consumers, prose, and images. In several of these applications it is not extremely important to identify the exact number of clusters or the correct membership of each pattern into a cluster. Often it is enough to group the objects in a reliable and parsimonious manner so that the underlying physical, biological, or evolutionary process can be understood or learned. Learning is a central issue in artificial intelligence (AI), and it is not surprising that AI researchers have recently adopted clustering as a form of "learning from observation," which is another term for unsupervised learning mentioned in Appendix A.

This does not mean that clustering techniques can automatically be applied to data without human intervention. The methodology for applying clustering methods is discussed in Section 3.5. The choice of features or measurements, similarity measure, and grouping technique requires familiarity with the subject area in which data arise. Most important, the clusters are best interpreted by an expert in the subject area. Various validity tests presented in Chapter 4 are merely tools at the disposal of an expert. Naive users of clustering can often generate incorrect interpretation or description of data.

A

Pattern Recognition

Pattern recognition refers to the classification or description of objects or patterns. The patterns themselves can range from characters in an image of printed text to biological waveforms, such as EKG or EEG waveforms. The recognition problems are to automatically identify the individual characters and to label the waveforms according to category, such as normal or abnormal. The design of such recognition systems requires that a set of training patterns, which are patterns with extrinsic pattern-class labels, be available. The main philosophical difference between pattern recognition and cluster analysis is the role of pattern class labels. These labels are crucial to the formation of decision rules in pattern recognition, but are only used to verify the results of cluster analysis. In other words, pattern recognition requires extrinsic information, while cluster analysis uses only the data themselves. Commercial pattern recognition systems are available for optical character recognition (OCR), speech recognition, speaker identification, fingerprint recognition, and automated cytology.

There are essentially two basic paradigms to classify a pattern into one of K different categories or pattern classes. The first is a *geometric* or *statistical* approach. A pattern is represented in terms of d features or measurements taken on a single object. If the choice of features is good, pattern vectors belonging to different categories will occupy different regions of this pattern space. Given training patterns from each pattern class, the objective is to establish decision boundaries in the feature space that separate patterns belonging to different classes.

In *statistical pattern recognition,* the features are assumed to have a probability density function conditioned on the pattern class. Thus a pattern vector $\mathbf{x}$ belonging to class ω_j is viewed as an observation drawn randomly from the class-conditional density $p(\mathbf{x} \mid \omega_j)$, $j = 1, \ldots, K$. Well-known concepts from statistical decision theory (Ferguson, 1967) and discriminant analysis (Lachenbruch and Goldstein, 1979) are utilized to establish decision boundaries between the pattern classes. If the class-conditional densities are known, the optimal decision rule is based on Bayes' decision theory (Duda and Hart, 1973). However, the class-conditional densities are never known in practice so various strategies are utilized to design a classifier based on the nature of information available about the class-conditional densities. If the forms of the class-conditional densities are known, we have a parametric decision problem. Otherwise, we must either estimate the density function or use some nonparametric decision rule, like the k-nearest-neighbor rule. Figure A.1 describes a hierarchy of problems in statistical pattern recognition.

The density functions are estimated from training samples. So another dichotomy in the theory of statistical pattern recognition is between *supervised learning* (labeled training samples) and *unsupervised learning* (unlabeled training samples). The label on each training pattern represents the category to which that pattern belongs. Other dichotomies that appear in statistical pattern recognition are shown in the tree structure of Figure A.1. The classification problems get more difficult as one traverses the tree from top to bottom and left to right. Cluster analysis essentially deals with the unsupervised learning mode, when the number of pattern classes is unknown, and it tries to find natural groupings in the data.

Many factors need to be considered in the design of a statistical pattern recognition system. The performance of the recognition system critically depends

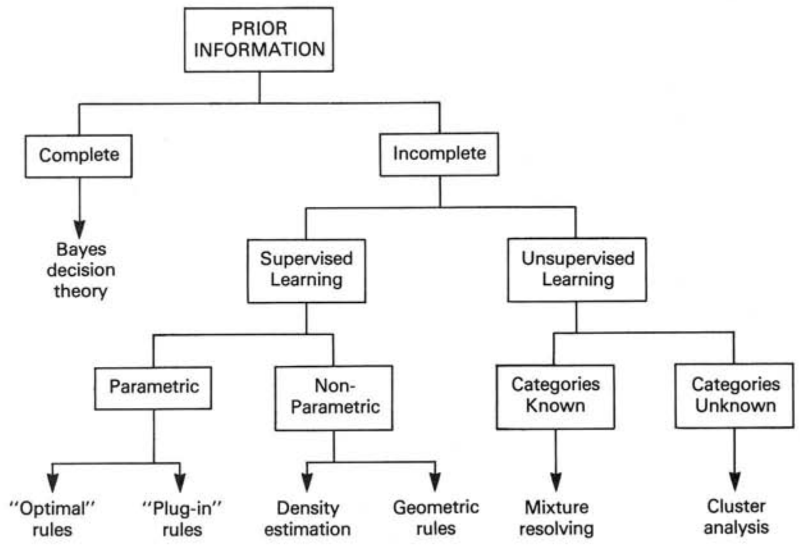


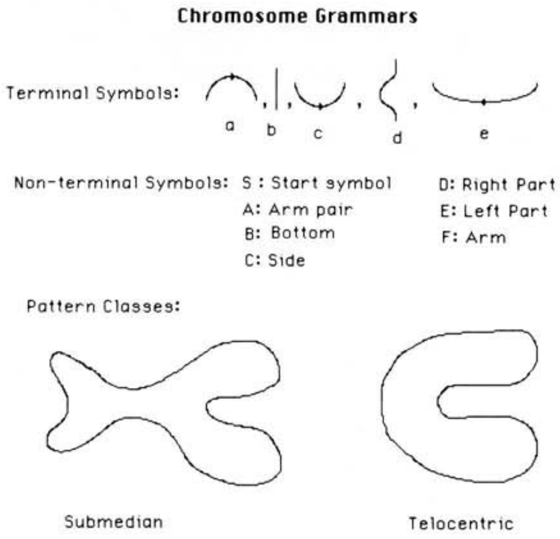
Figure A.1 Breakdown of problems in statistical pattern recognition.

on the choices made (Devijver and Kittler, 1982; Duda and Hart, 1973). Some of the issues encountered are listed below.

1. *One-shot versus hierarchical (tree) classifier.* In the one-shot classification scheme, the distinction between all classes is made in one stage. This may not be the most appropriate scheme, especially if the number of pattern classes is large. An alternative classification structure is a binary tree, where the most obvious discriminations are done first, postponing the more subtle distinctions to a later stage. This increases the processing speed since the average number of features per node is much smaller than the total number of features (Mui and Fu, 1980).
2. *Parametric versus nonparametric classification.* Most of the popular parametric techniques for classifying patterns are optimal if the class-conditional densities happen to be Gaussian. Even though statistics based on Gaussian distributions are robust, it is wise to consider nonparametric techniques, which are not always based directly on the pattern class distributions. Examples are near-neighbor classifiers and polynomial decision surfaces (Duda and Hart, 1973).
3. *Dimensionality and sample-size relationship.* The well-known phenomenon of the curse of dimensionality cautions a system designer to use a limited number of features for a given sample size. It is recommended that the number of training samples per class be at least 5 to 10 times the number of features (Jain and Chandrasekaran, 1982).
4. *Feature selection.* For computational reasons and cost considerations, one often faces the problem of deciding which of the available features are good for classification (Jain and Dubes, 1978). It has been shown that the problem of determining an optimal subset of features of size m from the d available features requires an exhaustive search over all possible subsets of size m. Branch-and-bound techniques have been proposed to avoid exhaustive search (Narendra and Fukunaga, 1977).
5. *Error estimation.* The available samples must be partitioned into training and test sets. The classifier is designed using the training set and evaluated on the samples belonging to the test set. Depending on the number of samples available, the error rate is estimated using either the hold-out method or the leave-one-out method. How should the available data be partitioned to optimize performance on future patterns? Devijver and Kittler (1982) and Toussaint (1974) investigate several aspects of this problem.

The second main paradigm for pattern recognition is called the *structural* or *syntactic* approach (Fu, 1974, 1982; Gonzalez and Thomason, 1978). In many recognition problems involving complex patterns, the number of features required to establish a reasonable decision boundary is very large. It is more appropriate to view such patterns as being composed of simple subpatterns. A subpattern itself could be built from simpler parts with grammatical techniques. Primitives are the simplest subpatterns to recognize and the given complex pattern is represented in terms of the interrelationships among these primitives. Figure A.2 gives a simple example to illustrate this approach. The boundaries of two types of chromosomes (submedian and telocentric) can be constructed from the five primitive curves or terminals shown in Figure A.2. Production rules determine the order of placement or concatenation of the primitives to form the pattern. Each pattern class has a unique set of production rules. The nonterminals describe partial or intermediate descriptions of the patterns in terms of the primitives.

The main advantage of the structural approach over the statistical approach is that, in addition to classification, it provides a description of the pattern. It specifies how the given pattern can be constructed from primitives. This paradigm has been used in situations where the patterns have a definite structure which can



Productions:

$S \longrightarrow AA$ (Submedian)
$S \longrightarrow BA$ (Telocentric)
$A \longrightarrow CA \mid AC \mid FD \mid EF$
$E \longrightarrow Fc$
$D \longrightarrow cF$
$B \longrightarrow bB \mid Bb \mid e$
$C \longrightarrow bC \mid Cb \mid b \mid d$
$F \longrightarrow bF \mid Fb \mid a$

Figure A.2 Example of syntactic pattern recognition.

be captured in terms of a set of rules, as with EKG waveforms, textured images, and shape analysis of contours.

Syntactic pattern recognition draws an analogy between the structure of patterns and the syntax of a language. The patterns belonging to a category are viewed as sentences belonging to a language. The sentences are generated according to a grammar and the primitives are viewed as the alphabet of the language. For example, one can define a grammar for abnormal EKG waveforms and one for normal EKG waveforms in terms of the P-Q-R-S complex of simple wave shapes. A given waveform is classified as normal or abnormal based on which one of the two grammars can correctly parse the pattern (Stockman et al., 1976). A large set of complex patterns can be described by using a small number of primitives and grammatical rules. Again, the grammar for each pattern class must be inferred from training samples.

Syntactic pattern recognition has a great deal of intuitive appeal. However, implementation of this approach leads to many difficulties, such as segmenting noisy patterns to detect the primitives, and inferring grammars. Although the statistical approach also has problems in classifying noisy and complex patterns, it is well understood and relies on firmly established elements of statistical decision theory (Jain, 1987). Perhaps this is why most commercial recognition systems utilize the statistical approach.

B

Distributions

This appendix provides background about two distributions that appear regularly in cluster analysis: the Gaussian, or normal, distribution, and the hypergeometric distribution.

B.1 THE GAUSSIAN DISTRIBUTION

The crucial link between multivariate statistics and normalization is in the mathematical model. The wide applicability of Gaussian or normal models in statistics and in cluster analysis is due to two reasons. First, the mathematical properties of Gaussian distributions are well understood. Quite often these models are justified by invoking the central limit theorem. Second, many applications summarize the knowledge about the data in the form of a mean vector and a covariance matrix. If the density function of the data is expected to be unimodal, the Gaussian density function is a useful and simplifying assumption. We begin with d random variables in a column vector representing the d features, or measurements, taken on each object.

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_d \end{bmatrix}$$

If the random variables have a d-dimensional Gaussian (normal) distribution, their joint density function is

$$f(x_1, x_2, \ldots, x_d) = [(2\pi)^d |\Sigma|]^{-1/2} \exp\,[-(1/2)(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})]$$

where Σ is the $d \times d$ *covariance matrix*, $|\Sigma|$ is its determinant, $\boldsymbol{\mu}$ is the d-vector of expected values, and $\mathbf{x}$ is a (column) d-vector containing variables $\{x_1, x_2, \ldots, x_d\}$. The mean vector, $\boldsymbol{\mu}$, and symmetric covariance matrix, Σ, are sufficient to define a Gaussian distribution. Thus a multivariate Gaussian density is specified by $d + d(d + 1)/2$ parameters. These parameters must be estimated by sample covariances and sample means to fit a Gaussian distribution to a set of patterns.

Care must be taken to distinguish sample moments, such as the sample means, sample variances, and sample covariance matrix defined in Section 2.3, from population moments, which are parameters of mathematical models describing the data. To be specific, let E denote the expectation operator with respect to an underlying probability model. The population mean for random variable X_i is $\mu_i = E(X_i)$. The (population) covariance between X_i and X_j is

$$\sigma_{ij} = E[(X_i - \mu_i)(X_j - \mu_j)] \qquad \text{where } i \neq j$$

Similarly, the (i, i), or diagonal, element of Σ is

$$\sigma_{ii}^2 = \sigma_i^2 = E[(X_i - \mu_i)^2] \qquad \text{for } i = 1, \ldots, d$$

which is the (population) variance for X_i. The (population) covariance can be written as

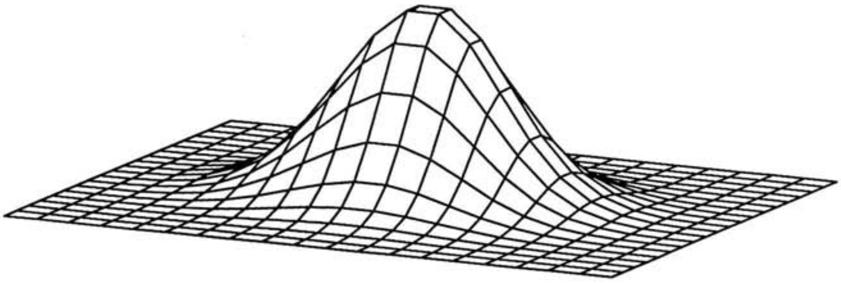
$$\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$$

where ρ_{ij} is the (population) *correlation coefficient* between X_i and X_j. Random variables X_i and X_j are both linearly and statistically independent if $\sigma_{ij} = 0$. Note that $\sigma_{ij} = \sigma_{ji}$ and $\rho_{ij} = \rho_{ji}$. Features are pictured as random variables and each pattern is a "realization" of these random variables. The population moments are estimated by the sample moments. Evaluating the quality of estimates is a statistical problem.

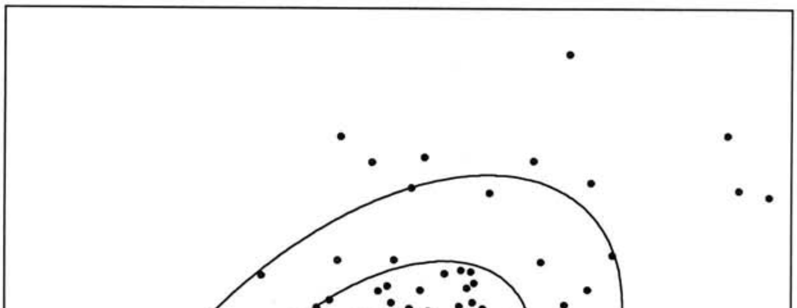
If statistical theory based on this model is to be employed in the interpretation of the results, the goodness of the Gaussian fit to a set of patterns must be investigated. A number of tests for determining the multivariate normality of a given data set are available (Lesaffre, 1983; Smith and Jain, 1985) but the task of testing for multivariate normality is very difficult. Ball (1965) pointed out several traps for the careless user of Gaussian models.

One of the concepts that is embedded in the Gaussian model but applies to all kinds of data is the ellipsoid of concentration (Cramer, 1945). Consider the simple case when $d = 2$ and $\boldsymbol{\mu} = \mathbf{0}$, the zero vector. The population covariance matrix can be written as follows.

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$



(a)



(b)

Figure B.1 Two pictures of a two-dimensional Gaussian density: (a) Gaussian density function; (b) Ellipses of constant density.

The joint density function for two Gaussian random variables is

$$f(x_1, x_2) = [4\pi^2\sigma_1^2\sigma_2^2(1 - \rho^2)]^{-1/2} \exp{[-q(x_1, x_2)]}$$

The quadratic form in the exponent is

$$q(x_1, x_2) = [2(1 - \rho^2)]^{-1}\left[\left(\frac{x_1}{\sigma_1}\right)^2 - \frac{2\rho x_1 x_2}{\sigma_1\sigma_2} + \left(\frac{x_2}{\sigma_2}\right)^2\right]$$

The equation $q(x_1, x_2) = K$, where K is a constant, forms an ellipse of constant density in the (x_1, x_2) plane. Varying K produces a family of concentric ellipses. The ellipse defined by $K = 2$ is called the *ellipse of concentration*. A uniform distribution defined over this ellipse will have the same first- and second-order moments as the Gaussian distribution.

Figure B.1 shows two different representations of a bivariate Gaussian density. The first is the three-dimensional picture of the density function. The second is a series of ellipses. Samples drawn from a multivariate normal population tend to fall in a single cloud or cluster.

The quadratic form $(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})$ is the squared Mahalanobis distance from $\mathbf{x}$ to $\boldsymbol{\mu}$ that is discussed in Section 2.2. Setting the Mahalanobis distance to a constant produces an ellipsoid in d dimensions. The principle axes of the hyperellipsoid are given by the eigenvectors of the covariance matrix. Eigenvalues determine the lengths of segments of the axes inside the hyperellipsoid. For the special case when $\rho = 0$, the axes of the ellipsoid are the coordinate axes. When the distribution of the d random variables is singular (this will be the case if one or more of the variables is a linear combination of other variables), its probability mass lies in a subspace of the d-dimensional pattern space. The rank of Σ equals the dimensionality of the subspace.

B.2 THE HYPERGEOMETRIC DISTRIBUTION

The hypergeometric distribution is an appropriate model for the isolation and compactness indices used to evaluate individual clusters in Section 4.5. It is also used in Section 2.3 to establish a measure of proximity between two binary vectors. This section defines the hypergeometric distribution in generic terms.

Suppose that we have a population of M objects and D of them are "defective." This terminology comes from quality control applications where one is sampling a lot of manufactured parts to estimate the number of substandard parts. We will interpret "defective" as a special object designation. For example, if the objects are edges inserted into a graph at random, the defectives could be the edges linking a subset of nodes to the rest of the graph.

Suppose that r objects are selected without replacement from the population of M objects containing D defectives and the order of the sample is immaterial.

If the sample of size r is taken randomly, all $\binom{M}{r}$ samples are equally likely, where $\binom{M}{r}$ is the binomial coefficient or

$$\binom{M}{r} = \frac{M!}{r!\,(M-r)!}$$

Let random variable X denote the number of defectives in the sample of size r. The D defectives can be sampled in $\binom{D}{x}$ ways and the $M - D$ nondefectives can be sampled in $\binom{M-D}{r-x}$ ways, when the sample contains exactly x defectives. The hypergeometric distribution provides the probability that exactly x objects in the sample are defective when the sampling is random.

$$\text{Prob}\,(X = x) = \frac{\binom{D}{x}\binom{M-D}{r-x}}{\binom{M}{r}}$$

The variable x in this probability lies in the following range.

$$\max\{0, r - M + D\} \le x \le \min\{D, r\}$$

For example, suppose that a population of five objects, labeled 1, 2, 3, 4, and 5, has three defectives, taken to be objects 1, 2, and 3. If two objects are sampled, the 10 possible results of sampling without replacement when the order of the sample is ignored are shown in Figure B.2. The number of defectives in the sample can be 0, 1, or 2. The probabilities can be determined either from the formula above or by counting the samples in Figure B.2 and are

$$\text{Prob}\,(X = x) = \begin{cases} 1/10 & \text{if } x = 0 \\ 6/10 & \text{if } x = 1 \\ 3/10 & \text{if } x = 2 \end{cases}$$

Computing hypergeometric probabilities is not a trivial task except when the values of M, D, and r are small, as in this example. For example, the individual factorials making up the binomial coefficients cannot be computed individually and combined because of numerical accuracy and overflow. The recursive formula given below is one approach. If the values of D, M, and r are fixed, the probability of exactly x defectives can be written as

$$\text{Prob}\,(X = x + 1) = \text{Prob}\,(X = x)\,\frac{(D - x)(r - x)}{(x + 1)(M - D - r + x + 1)}$$

The value of the random variable X must always lie in the range given above.

The cumulative distribution function

$$H(a) = \text{Prob}\,(X \le a)$$

is particularly tedious to compute, but is needed in the proximity measure defined in Section 2.2.3. Ling and Pratt (1984) provide an extensive empirical study of three approximations based on the binomial distribution and 12 approximations

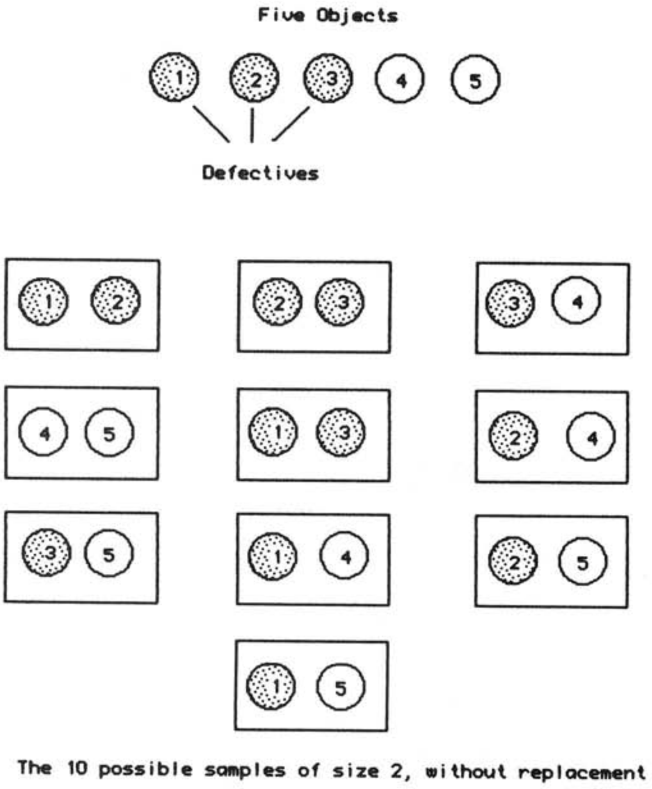


Figure B.2 Sampling without replacement from a population of five objects containing three defectives.

based on the Gaussian distribution. The Peizer approximation was found to be superior to other normal approximations. Binomial approximations were not recommended because the tails of the binomial distribution are almost as difficult to compute as the tails of the hypergeometric distribution being approximated. Li (1984) provides a two-level pipeline design for the recursive computation of $H(a)$ that can be implemented in hardware.

C

Linear Algebra

This appendix briefly reviews some facts from linear algebra used in Chapter 2. Several standard texts can be consulted for details, such as Green and Carroll (1976). If $\mathcal{R}$ is a real, symmetric, positive-definite matrix having rank d, such as the sample covariance matrix defined in Eq. (2.3), then the determinant equation

$$|\mathcal{R} - \lambda\mathcal{I}| = 0$$

where $\mathcal{I}$ is a unit matrix of order d and λ is a scalar, has d positive, real solutions for λ, counting a solution of multiplicity k as k solutions. These solutions, or eigenvalues of $\mathcal{R}$, are labeled $\lambda_1, \lambda_2, \ldots, \lambda_d$ in arbitrary order. An eigenvector corresponding to λ_i is a d-vector (column matrix) $\mathbf{c}_i$ which satisfies

$$(\mathcal{R} - \lambda_i\mathcal{I})\mathbf{c}_i = \mathbf{0}$$

$$\mathbf{c}_i^{\mathrm{T}}\mathbf{c}_j = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

This implies that

$$\mathbf{c}_i^{\mathrm{T}}(\mathcal{R} - \lambda_i\mathcal{I})\mathbf{c}_i = 0, \qquad i = 1, 2, \ldots, d$$

from which the following standard expansion is obtained:

$$\mathcal{C}_R\mathcal{R}\mathcal{C}_R^{\mathrm{T}} = \Lambda_R$$

Here $\Lambda_R = \operatorname{diag}(\lambda_1, \ldots, \lambda_d)$ is a diagonal matrix of order d. The rows of $\mathcal{C}_R$ form a set of eigenvectors of $\mathcal{R}$.

$$\mathcal{C}_R^{\mathrm{T}} = [\mathbf{c}_1 \quad \mathbf{c}_2 \quad \cdots \quad \mathbf{c}_d]$$

Since $\mathcal{C}_R$ is an orthogonal matrix, its inverse is its transpose, so both $\mathcal{R}$ and the inverse of $\mathcal{R}$ can be expressed in terms of the eigenvectors and eigenvalues.

$$\begin{aligned} \mathcal{C}_R^{-1} &= \mathcal{C}_R^{\mathrm{T}} \\ \mathcal{R} &= \mathcal{C}_R^{\mathrm{T}} \Lambda_R \mathcal{C}_R \\ \mathcal{R}^{-1} &= \mathcal{C}_R^{\mathrm{T}} \Lambda_R^{-1} \mathcal{C}_R \end{aligned} \tag{C.1}$$

The trace of $\mathcal{R}$, written tr $(\mathcal{R})$, and the determinant of $\mathcal{R}$, written $|\mathcal{R}|$, can be expressed as follows:

$$\begin{aligned} \mathrm{tr}\,(\mathcal{R}) &= \sum_{i=1}^{d} \mathrm{r}_{ii} = \sum_{i=1}^{d} \lambda_i \\ |\mathcal{R}| &= \prod_{i=1}^{d} \lambda_i \end{aligned} \tag{C.2}$$

If the linear transformation defined by $\mathcal{C}_R$ is applied to each pattern in the normalized pattern matrix $\mathcal{A}$, the $n \times d$ matrix $\mathcal{B}$ in Eq. (C.3) is obtained.

$$\mathcal{B} = \mathcal{A}\mathcal{C}_R^{\mathrm{T}} \tag{C.3}$$

where

$$\mathcal{B}^{\mathrm{T}} = [\mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_n] \qquad \text{and} \qquad \mathbf{y}_i = \mathcal{C}_R \mathbf{x}_i$$

The rows of $\mathcal{C}_R$ are eigenvectors of $\mathcal{R}$.

The transformation in Eq. (C.3) simply rotates the axes of the pattern space. Since $y_{ij} = \mathbf{c}_j^T \mathbf{x}_i$, the scalar y_{ij} is the portion of $\mathbf{x}_i$ in the direction of $\mathbf{c}_j$. Thus the new coordinate system "lines up" with the eigenvectors of $\mathcal{R}$. If the patterns in $\mathcal{A}$ are pictured as an ellipsoidal swarm of points in the pattern space, the axes of the rotated pattern space can be selected parallel to the axes of the ellipsoid. The rigid rotation does not change Euclidean distances between patterns. This interpretation of Eq. (C.3) follows from the fact that $\mathcal{R}$, computed from $\mathcal{A}$ by Eq. (2.3), is a covariance matrix, so $(1/n)\mathcal{B}^{\mathrm{T}}\mathcal{B}$ is the covariance matrix in the rotated coordinates. Applying Eqs. (C.3) and (C.1) proves Eq. (C.4).

$$(1/n)\mathcal{B}^{\mathrm{T}}\mathcal{B} = \mathcal{C}_R \mathcal{R} \mathcal{C}_R^{\mathrm{T}} = \Lambda_R \tag{C.4}$$

Thus the eigenvalues are sample variances in the rotated space and indicate the amount of spread along each axis. The features are uncorrelated in the rotated space because Λ_R is a diagonal matrix. The rows of $\mathcal{C}_R$ can be interchanged in d! ways, so d! different rotations can be defined having the diagonal property in Eq. (C.4), assuming distinct eigenvalues. The invariance of length and distance is shown below.

$$\mathcal{C}_R^{\mathrm{T}} \mathcal{C}_R = \mathcal{I}$$

so that

$$\mathbf{y}_i^{\mathrm{T}}\mathbf{y}_i = \mathbf{x}_i^{\mathrm{T}}\mathscr{C}_R^{\mathrm{T}}\mathscr{C}_R\mathbf{x}_i = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_i$$

and

$$(\mathbf{y}_i - \mathbf{y}_j)^{\mathrm{T}}(\mathbf{y}_i - \mathbf{y}_j) = (\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)$$

The transformation in Eq. (C.3) replaces the original features with new, uncorrelated features that are linear functions of the original features. This transformation is also the first step in a linear projection to a lower-dimensional space that minimizes square-error (see Section 2.4).

Example C.1

Two pattern matrices containing raw data are given below as $\mathscr{A}_1^*$ and $\mathscr{A}_2^*$. This example demonstrates normalization and the dangers of summarizing data with a correlation matrix. The patterns are shown in Figure C.1.

$$\mathscr{A}_1^* = \begin{bmatrix} 2\sqrt{2} & 2 & 0 & -2 & -2\sqrt{2} & -2 & 0 & 2 \\ 0 & 2 & 2\sqrt{2} & 2 & 0 & -2 & -2\sqrt{2} & -2 \end{bmatrix}^{\mathrm{T}}$$

$$\mathscr{A}_2^* = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 2 & -1 & -2 \\ -2 & -1 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$$

The sample means for both features in $\mathscr{A}_1^*$ are zero and the sample variances are 4. For example, for the first feature of $\mathscr{A}_1^*$,

$$m_1 = (1/8)\,(2\sqrt{2} + 2 + 0 - 2 - 2\sqrt{2} - 2 + 0 + 2) = 0$$

$$s_1^2 = (1/8)\,(8 + 4 + 0 + 4 + 8 + 4 + 0 + 4) = 4$$

Applying Eq. (2.2) simply scales both features and produces the following normalized pattern matrices:

$$\mathscr{A}_1 = \begin{bmatrix} \sqrt{2} & 1 & 0 & -1 & -\sqrt{2} & -1 & 0 & 1 \\ 0 & 1 & \sqrt{2} & 1 & 0 & -1 & -\sqrt{2} & -1 \end{bmatrix}^{\mathrm{T}}$$

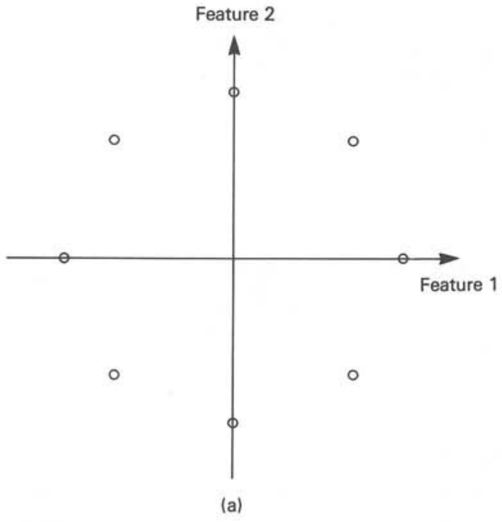
$$\mathscr{A}_2 = \sqrt{0.9}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 2 & -1 & -2 \\ -2 & -1 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$$

The correlation matrices, Eq. (2.3), are

$$\mathscr{R}_1 = (1/8)\mathscr{A}_1^{\mathrm{T}}\mathscr{A}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathscr{R}_2 = (1/9)\mathscr{A}_2^{\mathrm{T}}\mathscr{A}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Since both correlation matrices are the same, one might expect the two sets of patterns to resemble one another. Figure C.1 shows that the two sets of patterns are not at all alike, except that both exhibit the same type of circular symmetry. Thus first and second moments do not necessarily characterize data. We normally picture the patterns as a hyperellipsoidal swarm of patterns, but as Ball (1965) has pointed out, such a description is not always applicable and can be misleading.



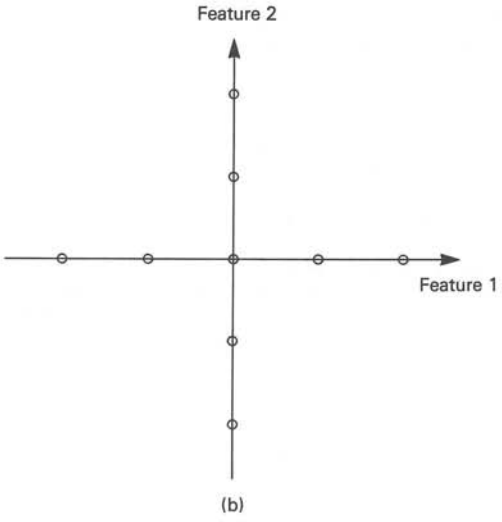


Figure C.1 Two sets of two-dimensional patterns.

Example C.2

This example demonstrates the effect of Eq. (C.3), which rotates the swarm of patterns so as to uncorrelate the features. We begin with the following matrix of eight patterns:

$$\mathscr{A}^* = \begin{bmatrix} 1 & 2 & 3 & 2 & 1 & 2 & 3 & 4 \\ 1 & 1 & 2 & 2 & 2 & 3 & 3 & 4 \end{bmatrix}^{\mathrm{T}}$$

The sample means are $m_1 = m_2 = 9/4$. Applying the normalization of Eq. (2.1) produces the following pattern matrix. The patterns are shown in Figure C.2.

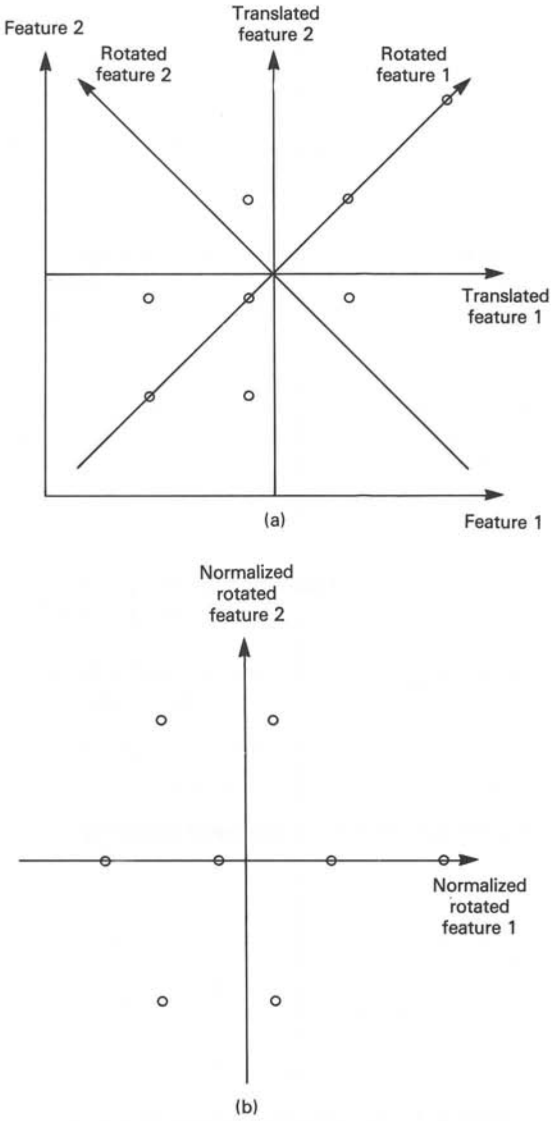


Figure C.2 Original and rotated patterns in two dimensions.

$$\mathscr{A} = (1/4)\begin{bmatrix} -5 & -1 & 3 & -1 & -5 & -1 & 3 & 7 \\ -5 & -5 & -1 & -1 & -1 & 3 & 3 & 7 \end{bmatrix}^{\mathrm{T}}$$

The covariance matrix is

$$\mathscr{R} = (1/8)\mathscr{A}^{\mathrm{T}}\mathscr{A} = (1/16)\begin{bmatrix} 15 & 11 \\ 11 & 15 \end{bmatrix}$$

The eigenvalues of $\mathcal{R}$ are values of λ satisfying

$$\begin{vmatrix} (15/16) - \lambda & 11/16 \\ 11/16 & (15/16) - \lambda \end{vmatrix}$$

or

$$32\lambda^2 - 60\lambda + 13 = 0$$

The two solutions, which are eigenvalues of $\mathcal{R}$, are $\lambda_1 = 13/8$ and $\lambda_2 = 1/4$. There are four sets of eigenvectors corresponding to these eigenvalues that differ in the directions of the axes. We choose directions represented by the following transformation [Eq. (C.3)]:

$$\begin{bmatrix} y_{i1} \\ y_{i2} \end{bmatrix} = (1/\sqrt{2}) \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix} \qquad \text{for } i = 1, \ldots, 8$$

$$\mathcal{B} = \mathcal{A}\mathcal{C}_R^{\mathrm{T}} = (1/2\sqrt{2}) \begin{bmatrix} -5 & -3 & 1 & -1 & -3 & 1 & 3 & 7 \\ 0 & -2 & -2 & 0 & 2 & 2 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$$

Figure C.2(a) shows that this equation defines a rigid rotation of the translated pattern space. Note that the spread of the patterns in the direction of the first rotated feature is much greater than that for the second rotated feature. Recall that $\lambda_1 = 13/8$ and $\lambda_2 = 1/4$ are sample variances along the new axes. Applying Eq. (2.2) to the *rotated* patterns changes the ellipsoidal swarm of patterns to the spherical swarm in Figure C.2(b). The sample means, after translation, are $m_1 = 0$, and $m_2 = 0$, while the sample variances, after rotation, are $s_1^2 = 13/8$ and $s_2^2 = 1/4$. The normalization is

$$y'_{ij} = \frac{y_{ij} - m_j}{s_j}$$

where $\mathcal{B} = [y_{ij}]$. The normalized patterns shown in Figure C.2(b) are

$$\mathcal{B}' = [y'_{ij}] = \begin{bmatrix} -1.39 & -0.83 & 0.28 & -0.28 & -0.83 & 0.28 & 0.83 & 1.94 \\ 0 & -1.41 & -1.41 & 0 & 1.41 & 1.41 & 0 & 0 \end{bmatrix}$$

D

Scatter Matrices

Scatter matrices play an important role in discriminant analysis and in cluster analysis and are defined in this appendix. The objects under consideration are described by d-dimensional patterns and are assumed to have been separated into K clusters or into K categories. The (unnormalized) patterns in the kth group, n_k in number, are denoted by the (column) vectors

$$[\mathbf{x}_1^{*(k)}, \ldots, \mathbf{x}_{n_k}^{*(k)}]^{\mathrm{T}}$$

where

$$\mathbf{x}_j^{*(k)} = [x_{j1}^{*(k)} \quad \cdots \quad x_{jd}^{*(k)}]^{\mathrm{T}}$$

The mean of the ith feature for the kth group is

$$m_i^{(k)} = (1/n_k) \sum_{j=1}^{n_k} x_{ji}^{*(k)}$$

The vector of feature means for the kth group is $\mathbf{m}^{(k)}$:

$$\mathbf{m}^{(k)} = [m_1^{(k)} \quad m_2^{(k)} \quad \cdots \quad m_d^{(k)}]^{\mathrm{T}}$$

The pooled mean, $\mathbf{m}$, is the grand mean vector for all patterns.

$$\mathbf{m} = (1/n) \sum_{k=1}^{K} n_k \mathbf{m}^{(k)} \qquad \text{where } n = \sum_{k=1}^{K} n_k$$

The normalization of Eq. (2.1) can be applied by subtracting the grand mean from all patterns.

$$\mathbf{x}_j^{(k)} = \mathbf{x}_j^{*(k)} - \mathbf{m}$$

Then the scatter matrix, $\mathcal{S}$, for the pooled sample is defined as

$$\mathcal{S} = \sum_{k=1}^{K} \sum_{j=1}^{n_k} (\mathbf{x}_j^{(k)})(\mathbf{x}_j^{(k)})^{\mathrm{T}}$$

The scatter matrix for the kth group is defined as

$$\mathcal{S}^{(k)} = \sum_{j=1}^{n_k} (\mathbf{x}_j^{*(k)} - \mathbf{m}^{(k)})(\mathbf{x}_j^{*(k)} - \mathbf{m}^{(k)})^{\mathrm{T}}$$

The within-group scatter matrix, $\mathcal{S}_W$, is defined as the sum of the group scatter matrices.

$$\mathcal{S}_W = \sum_{k=1}^{K} \mathcal{S}^{(k)}$$

Finally, the between-group scatter matrix, $\mathcal{S}_B$, is defined as the scatter matrix for the group means.

$$\mathcal{S}_B = \sum_{k=1}^{K} \sum_{j=1}^{n_k} (\mathbf{m}^{(k)} - \mathbf{m})(\mathbf{m}^{(k)} - \mathbf{m})^{\mathrm{T}} = \sum_{k=1}^{K} n_k \mathbf{m}^{(k)} \mathbf{m}^{(k)\mathrm{T}} - n\mathbf{m}\mathbf{m}^{\mathrm{T}}$$

The form of the first equation for $\mathcal{S}_B$ is similar to the equation for $\mathcal{S}$ above.

The three scatter matrices are related in a straightforward manner, as can be seen by writing

$$\mathbf{x}_j^{(k)} = (\mathbf{m}^{(k)} - \mathbf{m}) + (\mathbf{x}_j^{*(k)} - \mathbf{m}^{(k)})$$

Forming the product $\mathbf{x}_j^{(k)}\,\mathbf{x}_j^{(k)\mathrm{T}}$, summing over k and j, and realizing that the cross-products on the right sum to zero, show that

$$\mathcal{S} = \mathcal{S}_B + \mathcal{S}_W$$

Thus the total scatter in the data is divided into the between-group scatter and within-group scatter. This relationship is useful for defining projection algorithms (Section 2.4.3) and clustering algorithms (Section 3.3.1).

E

Factor Analysis

The goals of factor analysis (Harman, 1967) are to create a model for a set of objects containing fewer and more pertinent features than did the original representation. A good model in factor analysis reproduces the correlation matrix well, especially the variances. Some "underlying" or "fundamental" factors are being sought from which the correlation matrix for the observed data can be reproduced. A great deal of effort has been devoted to minimizing the number of factors involved. The factor analytic solution discussed here is called the *principal component* solution and is closely related to the eigenvector projection described in Section 2.4.1.

The normalization of Eq. (2.2) is assumed and the factor analysis model belongs to the class of models having the following form:

$$\mathbf{x}_i^\circ = \mathcal{H}\mathbf{F}_i \qquad \text{for } i = 1, \ldots, n \tag{E.1}$$

Here $\mathbf{x}_i^\circ$ is the representation provided by this model for the d-dimensional pattern, $\mathbf{x}_i$, representing the ith object; $\mathbf{F}_i$ is the m-vector of *factors* which defines $\mathbf{x}_i^\circ$; $\mathcal{H} = [h_{ij}]$ is a $d \times m$ matrix of *factor loadings* which establishes the factor analysis. The factors themselves cannot be observed. The problem is to estimate the elements of $\mathcal{H}$.

To simplify the problem of defining $\mathcal{H}$, the sample variances of all factors are assumed to be unity, the sample means are assumed to be zero, and the sample covariances between factors are assumed to be zero. These assumptions do not concern random variables representing the features but are imposed on the sample moments themselves. Specifically, writing

$$\mathbf{F}_i = [F_{i1} \quad F_{i2} \quad \cdots \quad F_{im}]^{\mathrm{T}}$$

as the factor vector corresponding to pattern $\mathbf{x}_i$, the assumptions are

$$(1/n) \sum_{i=1}^{n} F_{ij} = 0 \qquad \text{for } j = 1, \ldots, m$$

$$(1/n) \sum_{i=1}^{n} F_{ij} F_{ik} = \begin{cases} 0 & \text{if } j \neq k \\ 1 & \text{if } j = k \end{cases}$$

The sample moments provided by the model are given below for j from 1 to d.

$$m_j^\circ = (1/n) \sum_{i=1}^{n} x_{ij}^\circ = 0$$

$$(s_j^\circ)^2 = (1/n) \sum_{i=1}^{n} (x_{ij}^\circ)^2 = \sum_{k=1}^{m} h_{jk}^2$$

$$r_{jp}^\circ = (1/n) \sum_{i=1}^{n} x_{ij}^\circ x_{ip}^\circ = \sum_{k=1}^{m} h_{jk} h_{pk}$$

All solutions to the problem of factor analysis try to select the $\mathcal{H}$ matrix in such a way that these moments match the moments computed in the (normalized) pattern space. These goals are

$$(s_j^\circ)^2 = 1, \qquad j = 1, \ldots, d$$

$$\bar{r}_{jp} = r_{jp} - r_{jp}^\circ = 0 \qquad \text{for } j, p = 1, \ldots, d \quad \text{where } j \neq p$$

The scalars $\{\bar{r}_{jp}\}$ are called *residual correlations* and measure the goodness of the factors. The variance $(s_j^\circ)^2$ is also called the *communality* of the jth feature, and the sum over all m factors is called the *total communality* of the model. The contribution of the kth factor to the total communality can then be identified as

$$V_k = \sum_{j=1}^{d} h_{jk}^2$$

The strategy of the "component analysis" or "principal factor" approach to selecting the $\mathcal{H}$ matrix can be outlined as follows. First, try to select the matrix $\mathcal{H}$ to maximize V_1, the contribution of the first factor to the total communality, under the constraint that all residual correlations are 0. This leads to the first column of $\mathcal{H}$. Then select the remaining elements of $\mathcal{H}$ to maximize V_2, subject to the same constraint. The second column of $\mathcal{H}$ is obtained in this step. Repeating this procedure results in the $d \times m$ matrix $\mathcal{H}_F$ which is to be used in Eq. (E.1).

$$\mathcal{H}_F = \mathcal{H}_m^{\mathrm{T}} \Lambda_m^{1/2}$$

Here $\Lambda_m^{1/2} = \operatorname{diag}(\lambda_1^{1/2}, \lambda_2^{1/2}, \ldots, \lambda_m^{1/2})$ is a diagonal matrix and the $m \times d$ matrix $\mathcal{H}_m$ is the matrix of eigenvectors in Eq. (2.4).

The number, m, of factors can be no larger than the rank of $\mathcal{R}$. Since

$V_k = \lambda_k$ for $k = 1, \ldots, m$ when $\mathcal{H} = \mathcal{H}_F$, the contribution of each factor to the total communality can be obtained directly from the eigenvalues of $\mathcal{R}$. The cumulative sum of eigenvalues, when compared to d, provides a running account of the degree to which the total communality is being reproduced.

A factor analysis resolves each pattern into factors that may or may not have any interpretation in the original problem. The values of the factors themselves for each pattern cannot be computed because $\mathbf{x}_i \neq \mathbf{x}_i^\circ$ and $m < d$; even if $\mathbf{x}_i = \mathbf{x}_i^\circ$ were assumed, $\mathcal{H}$ in Eq. (E.1) is not a square matrix and does not have an inverse. Thus the factor analytic point of view is quite different from the linear projection point of view presented in Section 2.4. This difference is evident when the positions of $\mathcal{H}$ in Eqs. (2.4) and (E.1) are compared. However, if the rank of $\mathcal{R}$ is d, then the $d \times d$ matrix

$$\mathcal{H}_F = [\mathbf{c}_1 \quad \mathbf{c}_2 \quad \cdots \quad \mathbf{c}_d]\Lambda_R^{1/2}$$

has inverse

$$\mathcal{H}_F^{-1} = \Lambda_R^{-1/2}\mathcal{C}_R^{\mathrm{T}}$$

Thus the ith factor vector can be written as

$$\mathbf{F}_i = \mathcal{H}_F^{-1}\mathbf{x}_i$$

Since the ordering $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ was assumed, the factors decrease in importance from top to bottom in the vector of factors. Choosing the first m factors only provides a projection of the original patterns into an m-dimensional space. Since

$$\mathbf{F}_i^{\mathrm{T}}\mathbf{F}_i = \mathbf{x}_i^{\mathrm{T}}\mathcal{C}_R\Lambda_R^{-1}\mathcal{C}_R^{\mathrm{T}}\mathbf{x}_i = \mathbf{x}_i^{\mathrm{T}}\mathcal{R}^{-1}\mathbf{x}_i$$

lengths of patterns and distances between patterns are different in the two spaces. However, the correlation matrix in the new space is the unit matrix.

$$(1/n)\sum_{i=1}^{n} \mathbf{F}_i\mathbf{F}_i^{\mathrm{T}} = \mathcal{H}_F^{-1}\mathcal{R}(\mathcal{H}_F^{-1})^{\mathrm{T}} = \mathcal{I} \tag{E.2}$$

Comparing Eq. (E.2) to Eq. (2.5) exhibits the effect of including $\Lambda_R^{1/2}$ in the transformation. Another item to note is that when $d = m$, Eq. (E.2) can be written as

$$\mathcal{R} = \mathcal{H}_F(\mathcal{H}_F)^{\mathrm{T}} \tag{E.3}$$

The *factor loadings* in $\mathcal{H}_F$ are correlations between the given features and the factors. The factors can sometimes be named by determining which features influence the factors most strongly, the ideal being bipolar factor loadings, or entries of $\mathcal{H}_F$ which are either 0 or 1.

The correlation matrix can be expressed in the form of Eq. (E.3) with nonsingular matrices other than $\mathcal{H}_F$. In particular, a matrix having all zeros below the main diagonal can be defined that satisfies Eq. (E.3) and such a matrix defines another transformation with the property of Eq. (E.2).

If the residual correlations are to be made zero and the correlation matrix $\mathcal{R}$ is employed, the number of factors must be d and no reduction in the number of features can be achieved. However, the rank of $\mathcal{R}$ can be reduced by replacing the 1's on the main diagonal of $\mathcal{R}$ with estimates of communalities. Reducing the rank of $\mathcal{R}$ reduces the number of factors necessary to make all residual correlations zero. A great deal of effort in factor analysis is directed at estimating communalities to use on the diagonals in $\mathcal{R}$.

F

Multivariate Analysis of Variance

The basic problem in multivariate analysis of variance (MANOVA) is to decide whether means of group populations differ significantly. Groups are formed from information incidental to the data, such as category labels. The group covariance matrices are assumed to be the same. MANOVA is a well-known statistical technique which is used in the design of experiments, where, for example, the response or outcome of K treatments on agricultural plots is studied. The test statistic for testing the null hypothesis that all group means are the same is based on $\Lambda = |\mathcal{S}_B|/|\mathcal{S}_W|$, where $\mathcal{S}_B$ is the between-group scatter matrix and $\mathcal{S}_W$ is the within-group scatter matrix defined in Appendix D. Various approximations to the distribution of Λ are possible, such as the Rao approximation, in which a test statistic having an F-distribution is formed. The assumption of equal covariance matrices can be checked with the Box test (Cooley and Lohnes, 1971).

In the context of discriminant analysis, the following hypothesis-testing problem can be studied. The objects are represented by d-dimensional patterns. Suppose that the t most significant, or largest, eigenvalues of $\mathcal{S}_W^{-1}\mathcal{S}_B$ are accepted. How well can the K groups be discriminated when the objects are projected to a space defined by the remaining $m - t$ eigenvectors, where $m = \min(K - 1, d)$?

The null hypothesis is that there is no significant difference among groups. However, the groups must be specified before the data are observed. For example, if the groups correspond to different treatments of a disease, the patients must be labeled according to treatment. This is an important point to remember when attempting to validate the results of clustering algorithms using MANOVA. When the group labels are cluster labels imposed after the patterns have been separated into clusters, the following distributions are *not* valid. The test statistics is

$$T_t = -(n - \frac{d + K}{2} - 1) \log (\Lambda_t)$$

where

$$\Lambda_t = \prod_{j=t+1}^{m} (1 + \zeta_j)^{-1}$$

It can be shown that under the null hypothesis, T_t has an approximate chi-squared distribution with $(d - t)(K - t - 1)$ degrees of freedom. Since a small value of Λ_t indicates good separation according to group and $0 < \Lambda_t < 1$, large values of T_t reflect significant discrimination. The form of the test is:

Reject the null hypothesis if $T_t > \alpha$

where α is the level of significance (Chapter 4).

Even though the distribution of T_t is unknown when the groups are formed by cluster analysis, the linear projection defined by discriminant analysis based on clusters provides an excellent two-dimensional projection of the patterns. When $K > 3$, the two-dimensional projection defined by the two largest eigenvalues of $\mathcal{S}_W^{-1}\mathcal{S}_B$, or ζ_1 and ζ_2, will provide the smallest increase in scatter ratio among all other two-dimensional projections, as can be seen from Eq. (2.7). This projection is the first two rows of Eq. (2.6).

G

Graph Theory

This brief appendix lists some definitions from graph theory that are used in clustering. Most books on discrete mathematics have sections on graph theory (Korfhage, 1974).

G.1 DEFINITIONS

A graph is a mathematical structure that has a multitude of applications in cluster analysis. This appendix covers undirected, finite graphs. We begin with a finite set of vertices, or nodes, $V = \{v_i\}$ which usually represent the objects being clustered. A set of edges $E = \{e_j\}$ records the interactions between pairs of vertices. The vertices and edges are related by a function f that maps edges into unordered pairs of vertices. That is, f assigns each edge in E to an element of the product set $V \times V$. A *graph* G is the triple $G = <V, E, f>$.

The graph G defined above is an undirected graph because the pairs of vertices are unordered. In other words, this type of graph is a symmetric, nonreflexive, binary relation on the set of vertices. Vertices are written on paper as dots and edges are denoted by lines connecting the dots, as in Figure G.1. Directed graphs have arrows on the edges indicating an ordering of the vertices. We assume that there are no "self-loops," so the two vertices to which f assigns an edge are distinct. Multiple edges are also disallowed, so f assigns each edge to a distinct pair of vertices. The word "graph" will thus refer to an undirected graph with no self-loops or multiple edges.

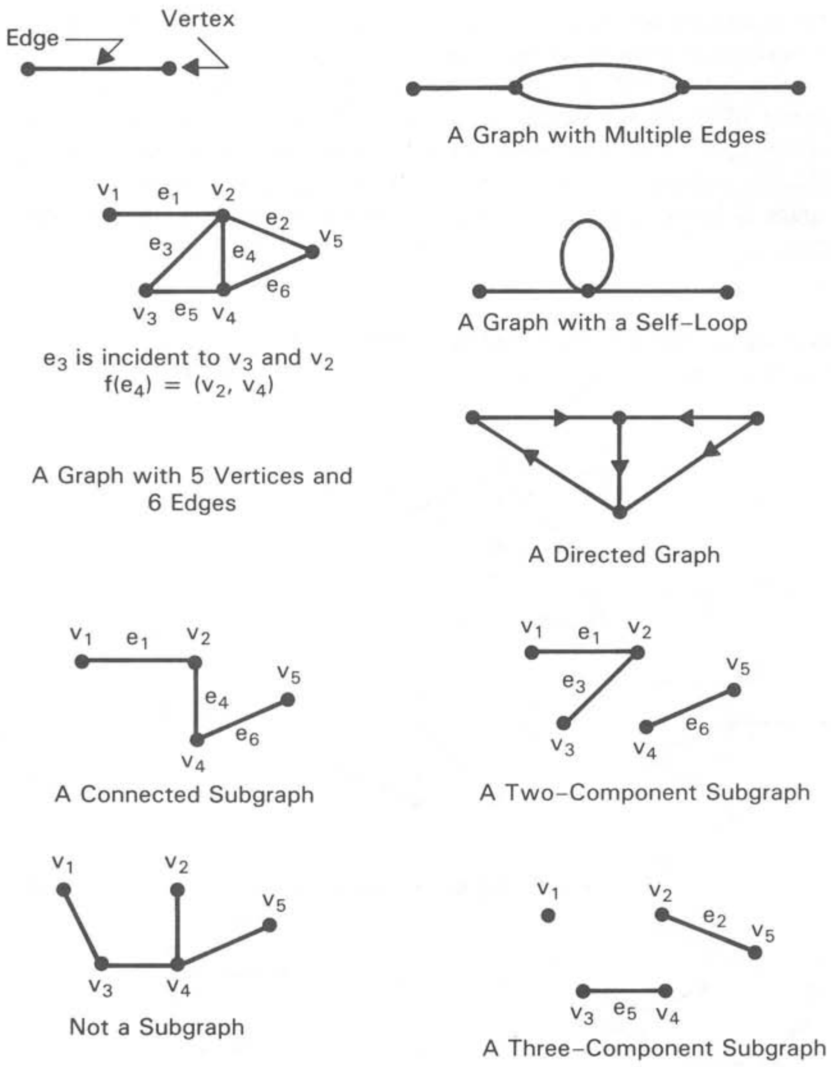


Figure G.1 Examples of graph definitions.

Some applications require that labels, or weights, be assigned to the vertices and/or to the edges. For example, the vertices could represent points in the plane and an edge could link two vertices if the points are closer than some threshold distance. The nodes could be labeled according to object and the edges could be labeled by the distance between patterns representing the objects.

If edge e is mapped into vertices (v_1, v_2) by $f(e) = (v_1, v_2)$, edge e is said to be *incident to* vertices v_1 and v_2. In general, some vertex pairs will not be assigned an edge and some vertices will not have any incident edges. Definitions of several terms from graph theory are listed below.

Graph $H = <V', E', f'>$ is a *subgraph* of graph $G = <V, E, f>$ if V' is a subset of V, E' is a subset of E, and f' is the function f with its domain restricted to E'. Thus a subgraph is a subset of the nodes and edges of the original graph that contains no edges or vertices that are not in the original graph. A *path* in graph G between vertices v_1 and v_n is an alternating sequence of vertices and edges:

$$v_1 e_1 v_2 e_2 \quad \cdots \quad v_{n-1} e_{n-1} v_n$$

containing no repeated edges and no repeated vertices and for which edge e_i is incident to vertices v_i and v_{i+1} for $i = 1, \ldots, n$. Examples are in Figure G.2.

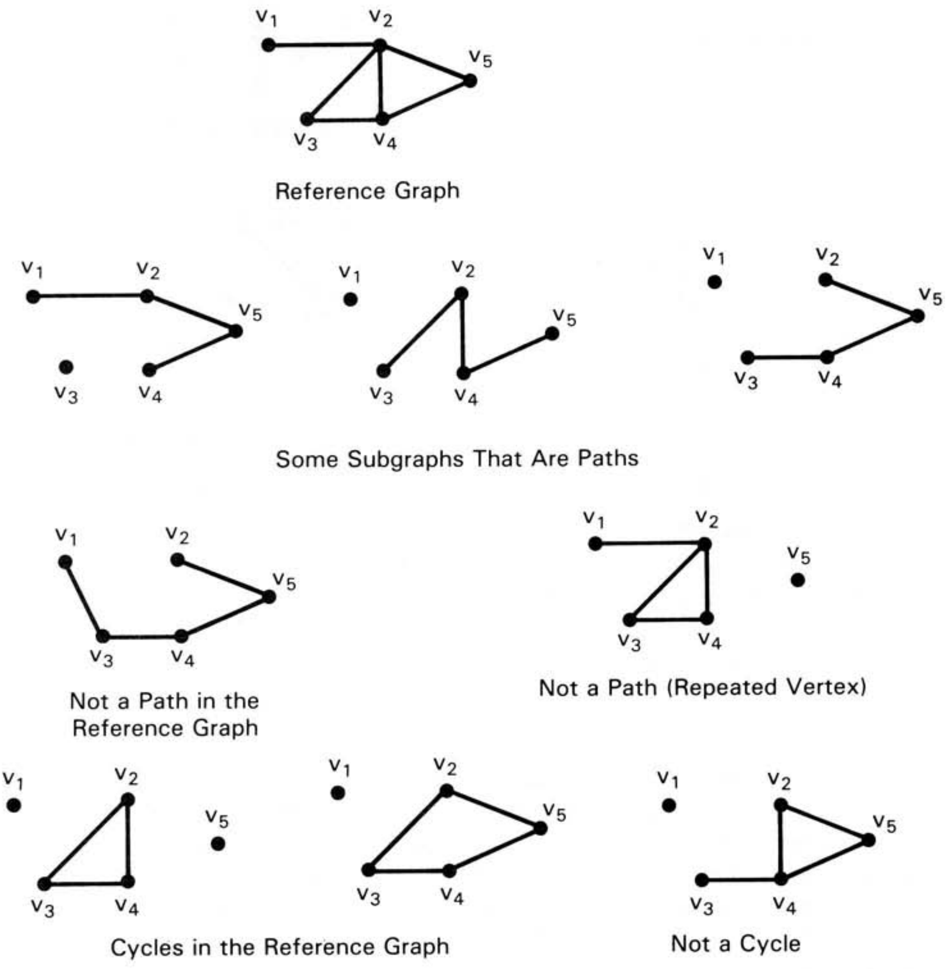

v1
v2
v3
v4
v5
Reference Graph
Some Subgraphs That Are Paths
Not a Path in the Reference Graph
Not a Path (Repeated Vertex)
Cycles in the Reference Graph
Not a Cycle

Figure G.2 Properties of graphs.

A graph is *connected* if a path exists between any two vertices in the graph. A *component* is a maximal piece of a connected graph. The word "maximal" means "as many nodes as possible," so a component is not a proper subgraph of another connected graph. A graph G is *complete* if an edge is assigned to every possible pair of nodes. A complete graph on n nodes contains exactly $n(n - 1)/2$ edges. A *maximal complete subgraph* of G is a complete subgraph of G that is not a proper subgraph of any other complete subgraph of G. That is, H is a maximal complete subgraph of G if an edge is incident to every pair of

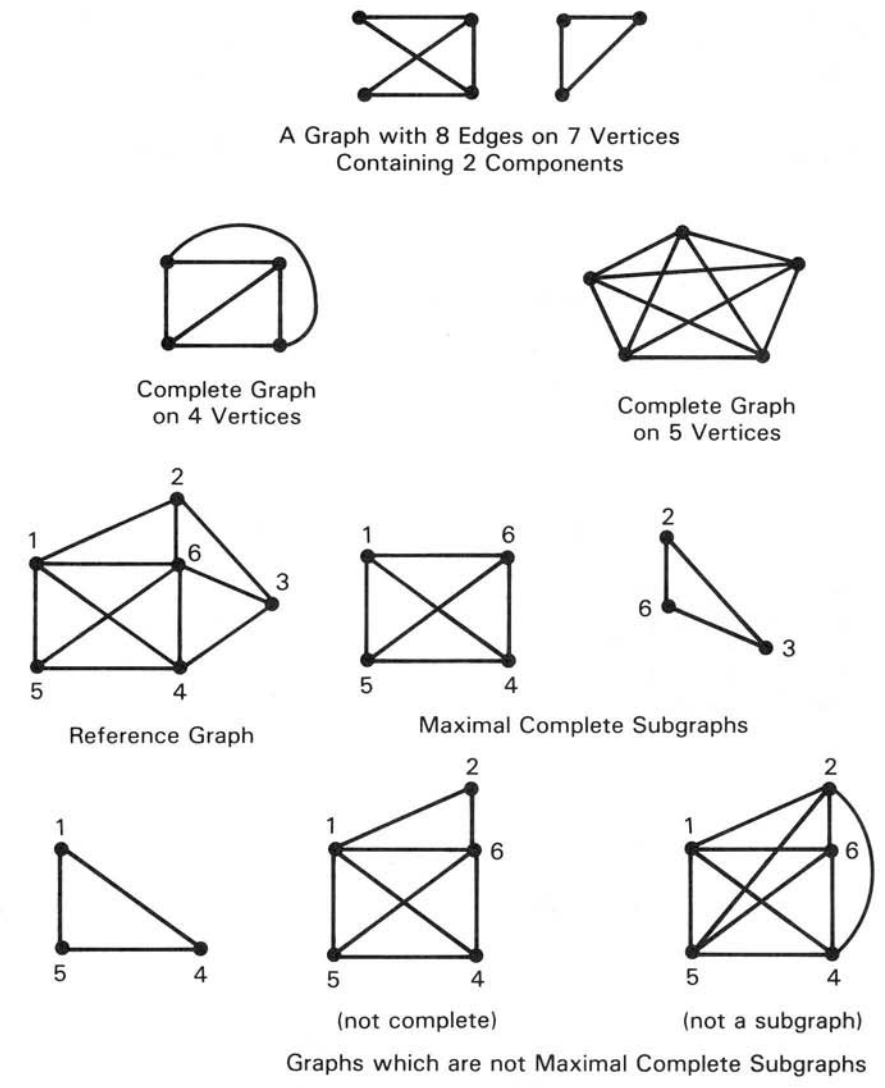

A Graph with 8 Edges on 7 Vertices
Containing 2 Components
Complete Graph on 4 Vertices
Complete Graph on 5 Vertices
1
2
3
4
5
6
Reference Graph
Maximal Complete Subgraphs
(not complete)
(not a subgraph)
Graphs which are not Maximal Complete Subgraphs

Figure G.3 More examples of graphs.

vertices in H but no additional vertices of G can be added to H without destroying the completeness. The name *clique* is used for a maximal complete subgraph. Figure G.3 provides examples.

G.2 TREES

A *cycle* is the same as a path except that vertices v_1 and v_n are the same vertex. A *tree* is a connected graph with no cycles. If a subgraph has m vertices, it is

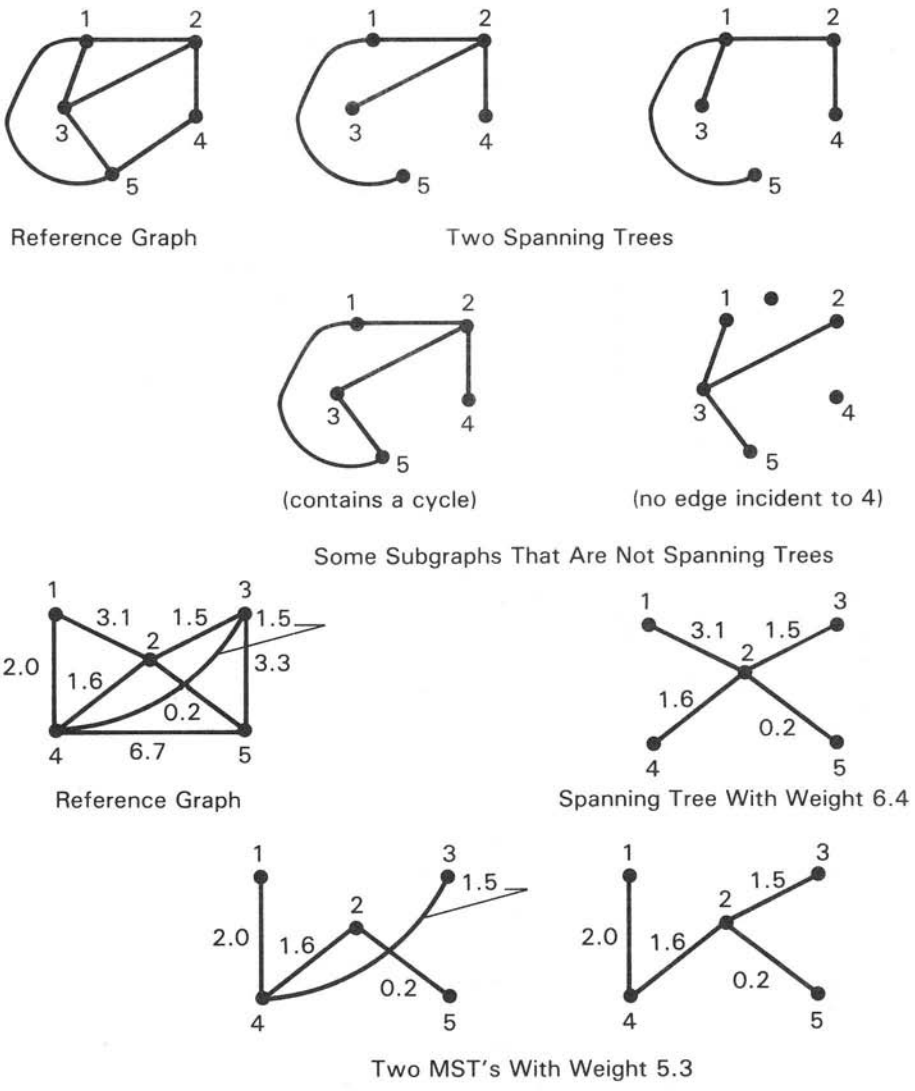


Figure G.4 Spanning tree examples.

easy to prove that a tree containing these vertices has exactly $m - 1$ edges. A *spanning tree* is a tree containing all vertices of the graph. When the edges in a graph are weighted by dissimilarities on an interval or ratio scale, the *weight of a tree* is the sum of the edge weights in the tree. A *minimum spanning tree* (MST) of G is a tree having minimal weight among all other spanning trees of G. As demonstrated in Figure G.4, a graph can have more than one MST. MSTs of complete graphs are especially important in cluster analysis.

A number of algorithms have been proposed for identifying an MST. Prim's (1957) algorithm is generally taken to be the best computationally, but the "naive" algorithm is easier to apply with pencil and paper. That is, rank order the edge

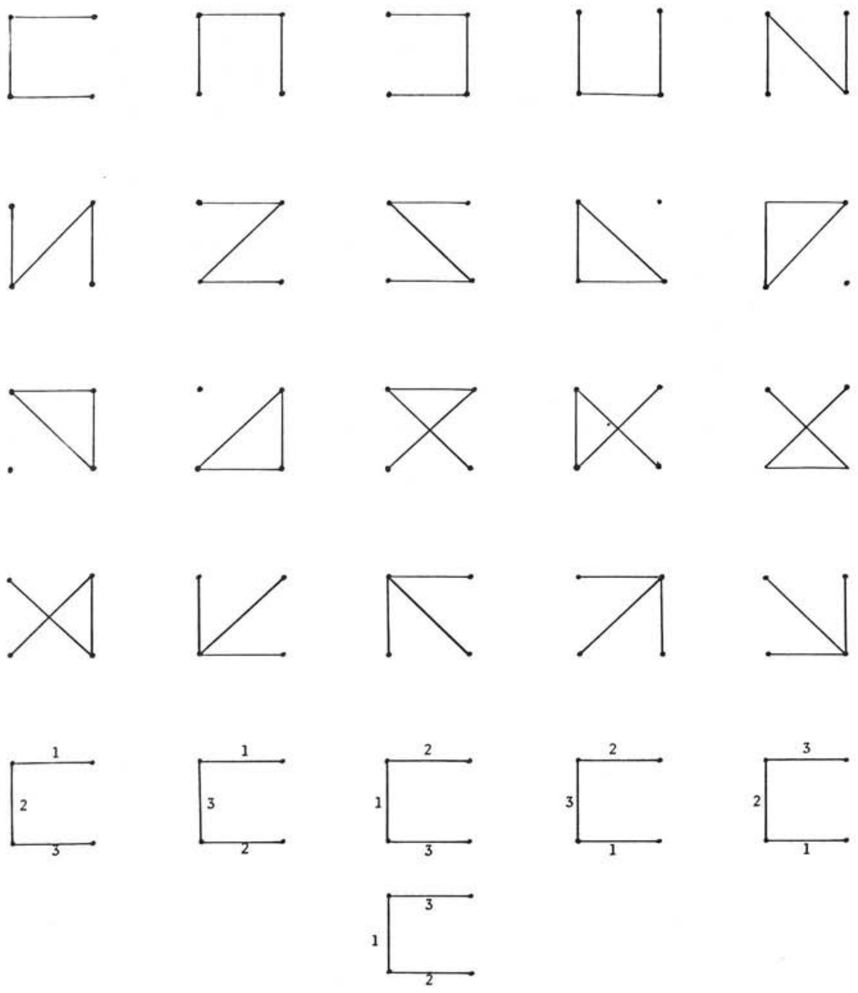


Figure G.5 Random threshold and rank graphs.

weights and insert edges in order of dissimilarity, shortest first, until a tree is formed, being careful not to form any cycles.

G.3 RANDOM GRAPHS

An (n, N) *threshold graph* is defined as a graph on n labeled nodes containing N unlabeled edges. The adjective ''random'' is sometimes used to indicate that the edges have been inserted at random. A *rank graph* is a threshold graph in which the edges are labeled. There are $\binom{n:2}{N}$ distinct (n, N) threshold graphs and $\binom{n:2}{N}N!$ is the number of (n, N) rank graphs.

The notation $\binom{m}{k}$ indicates the number of ways in which k objects can be sampled from a population of m objects when sampling is done without replacement and ordering of the sample is ignored. The special notation $n : 2$ indicates n objects sampled two at a time so that

$$n : 2 = \frac{n(n-1)}{2}$$

Figure G.5 shows all (4, 3) threshold graphs and all rank graphs for a particular random threshold graph.

An (n, N) random threshold graph can be generated by inserting N edges in a graph with n labeled nodes in such a way that as each edge is inserted, all unfilled positions are equally likely. The edges themselves are not labeled. Multiple edges between two nodes are not allowed and an edge cannot have the same node at both ends, so self-loops are not allowed.

H

Algorithm for Generating Clustered Data

The algorithm presented in this appendix generates samples from a spatial point process that is a modification of the Neyman–Scott (1972) process in which spherically shaped Gaussian clusters are located randomly in the sampling window. The algorithm ensures that the clusters do not overlap more than a specified amount, provides for a minimum number of points per cluster, and permits the exact number of clusters to be specified. The parameters of the algorithm are:

- n: number of patterns, or points, to be generated
- d: dimensionality of space
- σ: cluster spread
- $n_{\min}$: minimum number of points per cluster
- I_o: overlap index
- c: number of clusters ($n_{\min} c \leq n$)

The objective is to create a set of n patterns in a d-dimensional sampling window, taken here to be the unit hypercube, arranged into c clusters with at least $n_{\min}$ patterns per cluster. Points in individual clusters are independent samples from d-dimensional Gaussian distributions centered at a randomly chosen cluster center and having covariance matrix:

$$\sigma^2 \mathcal{I}, \qquad \text{where } \mathcal{I} \text{ is a } d \times d \text{ unit matrix}$$

The overlap between clusters i and j, $O(i, j)$, is defined as the Bayes error in the following hypothesis-testing problem that classifies a d-vector, $\mathbf{X}$, according to

$$H_i\text{: } \mathbf{X} \text{ is distributed as } N(\boldsymbol{\mu}_i, \sigma^2 \mathcal{I})$$

$$H_j\text{: } \mathbf{X} \text{ is distributed as } N(\boldsymbol{\mu}_j, \sigma^2 \mathcal{I})$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ are the true centers of two clusters. Prior probabilities for the two hypotheses are defined in terms of the relative number of patterns in the two clusters. The Bayes rule for this problem is well known (Duda and Hart, 1973). Letting n_i denote the number of patterns in cluster i and letting N_i and N_j denote the numbers of patterns from clusters i and j misclassified by the Bayes rule, the overlap between the two clusters is defined as

$$O(i, j) = \frac{N_i + N_j}{n_i + n_j}$$

The overlap varies from 0 to 1, with 0 meaning well-separated clusters and 1 meaning coincident clusters. The overlap parameter I_o specifies the maximum allowable overlap between any pair of clusters and is not allowed to exceed I_o. The algorithm itself is outlined below.

ALGORITHM FOR GENERATING CLUSTERED DATA

Step 1. Establish cluster sizes $\{n_1, n_2, \ldots, n_c\}$ for which

$$\sum_{k=1}^{c} n_k = n \quad \text{and} \quad n_k \geq n_{\min} \qquad \text{for all } k$$

This is accomplished by setting n_k to $n_{\min}$ for all k, then selecting clusters at random and incrementing their size by 1 until the sums of the cluster sizes is n.

Repeat steps 2 to 5 for i from 1 to c.

Step 2. Generate cluster center $\boldsymbol{\mu}_i$ at random in the sampling window (unit hypercube).

Step 3. Scatter n_i patterns around $\boldsymbol{\mu}_i$ according to a $N(\boldsymbol{\mu}_i, \sigma^2 \mathcal{I})$ distribution. Reject patterns falling outside the sampling window. Continue until n_i patterns have been generated inside the sampling window.

Step 4. If any of the overlaps $O(i, i - 1), O(i, i - 2), \ldots, O(i, 1)$ exceeds I_o, repeat steps 2 and 3.

Step 5. If 50 repetitions do not succeed in generating a new cluster center, increase I_o to the smallest overlap encountered in the 50 repetitions and repeat steps 2 to 4.

BIBLIOGRAPHY

AGRAWALA, A. K., MOHR, J. M., and BRYANT, R. M. (1976). "An approach to the workload characterization problem." *Computer 9* (June), 18–32.

AHUJA, N. (1982). "Dot pattern processing using voronoi neighborhoods." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 4*, 336–343.

ALAGAR, V. S. (1976). "The distribution of the distance between random points." *Journal of Applied Probability 13*, 558–566.

ANDERBERG, M. R. (1973). *Cluster Analysis for Applications*. Academic Press, Inc., New York.

ANDREWS, D. F. (1972). "Plots of high-dimensional data." *Biometrics 28*, 125–136.

ATKINSON, A. C. (1979a). "The computer generation of Poisson random variables." *Applied Statistics 28*, 29–35.

ATKINSON, A. C. (1979b). "Recent developments in the computer generation of Poisson random variables." *Applied Statistics 28*, 260–263.

BACKER, E. (1978). *Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets*. Delft University Press, Delft, The Netherlands.

BACKER, E., and JAIN, A. K. (1981). "A clustering performance measure based on fuzzy set decomposition." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 3*, 66–75.

BAILEY, T. A., and DUBES, R. C. (1982). "Cluster validity profiles." *Pattern Recognition 15*, 61–83.

BAKER, F. B. (1974). "Stability of two hierarchical grouping techniques—Case 1. Sensitivity to data errors." *Journal of the American Statistical Association 69*, 440–445.

BAKER, F. B., and HUBERT, L. J. (1976). "A graph-theoretic approach to goodness-of-fit in complete-link hierarchical clustering." *Journal of the American Statistical Association 71*, 870–878.

BALL, G. H. (1965). "Data analysis in the Social Sciences: what about the details?" *Proceedings of the AFIPS Fall Joint Computer Conference*, pp. 533–560.

BALL, G. H., and HALL, D. J. (1964). "Some fundamental concepts and synthesis procedures for pattern recognition preprocessors." *International Conference on Microwaves, Circuit Theory, and Information Theory*, September, Tokyo.

BALLARD, D. H., and BROWN, C. M. (1982). *Computer Vision*. Prentice-Hall, Inc., Englewood Cliffs, N.J.

BARROW, H. G., and TENENBAUM, J. H. (1978). "Recovering intrinsic scene characteristics from images." In *Computer Vision Systems* (A. Hanson and E. Riseman, eds.), Academic Press, Inc., New York, pp. 3–26.

BARTELS, P. H., BAHR, G. F., CALHOUN, D. W., and WIED, G. L. (1970). "Cell recognition by neighborhood grouping techniques in TICAS." *Acta Cytologica 14*, 313–324.

BARTLETT, M. S. (1975). *The Statistical Analysis of Spatial Pattern*. Chapman & Hall Ltd., London.

BARTON, D. E., and DAVID, F. N. (1962). "Randomization bases for multivariate tests in the bivariate case—randomness of points in a plane." *Bulletin of the International Statistical Institute 39*, 455–467.

BAYNE, C. K., BEAUCHAMP, J. J., BEGOVICH, C. L., and KANE, V. E. (1980). "Monte Carlo comparisons of selected clustering procedures." *Pattern Recognition 12*, 51–62.

BELLMAN, R. E., KALABA, R., and ZADEH, L. A. (1966). "Abstraction and pattern classification." *Journal of Mathematical Analysis and Applications 13*, 1–7.

BENNETT, R. S. (1969). "The Intrinsic dimensionality of signal collections." *IEEE Transactions on Information Theory IT15*, 517–525.

BENTLER, P. M., and WEEKS, D. G. (1978). "Restricted multidimensional scaling models." *Journal of Mathematical Psychology 17*, 138–151.

BESAG, J. E., and DIGGLE, P. J. (1977). "Simple Monte Carlo tests for spatial patterns." *Applied Statistics 26*, 327–333.

BEZDEK, J. C. (1974). "Cluster validity with fuzzy sets." *Journal of Cybernetics 3*, 58–73.

BEZDEK, J. C. (1976). "A physical interpretation of fuzzy ISODATA." *IEEE Transactions on Systems, Man and Cybernetics SMC6*, 387–389.

BEZDEK, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York.

BICKEL, P. J., and FREEDMAN, D. A. (1981). "Some asymptotic theory for the bootstrap." *Annals of Statistics 9*, 1196–1217.

BINDER, D. A. (1978). "Bayesian cluster analysis." *Biometrika 65*, 31–38.

BISWAS, G., JAIN, A. K., and DUBES, R. (1981). "Evaluation of projection algorithms." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAM I3*, 701–708.

BLASHFIELD, R. K. (1976). "Mixture model tests of cluster analysis: Accuracy of four agglomerative hierarchical methods." *Psychological Bulletin 83*, 377–388.

BLASHFIELD, R. K., ALDENDERFER, M. S., and MOREY, L. C. (1982). "Cluster analysis software." In *Handbook of Statistics*, Vol. 2 (P. R. Krishnaiah and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 245–266.

BOCK, H. H. (1985). "On some significance tests in cluster analysis." *Journal of Classification 2*, 77–108.

BROCKETT, P. L., HAALAND, P. D., and LEVINE, A. (1981). "Information theoretic analysis of questionnaire data." *IEEE Transactions on Information Theory IT 27*, 438–445.

BRYANT, J. (1979). "On the clustering of multidimensional pictorial data." *Pattern Recognition 11*, 115–125.

CALVERT, T. W. (1970). "Nonorthogonal projections for feature extraction in pattern recognition." *IEEE Transactions on Computers C19*, 447–452.

CARMICHAEL, J. W., GEORGE, J. A., and JULIUS, R. S. (1968). "Finding natural clusters." *Systematic Zoology 17*, 144–150.

CARROLL, J. D., and WISH, M. (1971). "Measuring preference and perception with multidimensional models." *Bell Laboratories Record*, pp. 147–154.

CHANG, W. C. (1983). "On using principal components before separating a mixture of two multivariate normal distributions." *Applied Statistics 32*, 267–275.

CHANG, C. L., and LEE, R. C. T. (1973). "A heuristic relaxation method for nonlinear mapping in cluster analysis." *IEEE Transactions on Systems, Man and Cybernetics SMC 3*, 197–200.

CHEN, C. K., and ANDREWS, H. C. (1974). "Nonlinear intrinsic dimensionality computations." *IEEE Transactions on Computers C23*, 178–184.

CHEN, H. J., GNANADESIKAN, R., and KETTENRING, J. R. (1974). "Statistical methods for grouping corporations." *Sankhyā (The Indian Journal of Statistics) B34*, Part 1, 1–28.

CHENG, Y., and FU, K. S. (1984). "Conceptual clustering in knowledge organization," *Proceedings First IEEE Conference on Artificial Intelligence Applications*, Denver, December 1984, pp. 274–279.

CHERNICK, M. R., MURTHY, V. K., and NEALY, C. D. (1985). "Application of bootstrap and other resampling techniques: evaluation of classifier performance." *Pattern Recognition Letters 3*, 167–178.

CHERNOFF, H. (1973). "Using faces to represent points in *k*-dimensional space graphically." *Journal of the American Statistical Association 68*, 361–368.

CHERNOFF, H. (1978). "Graphical representations as a discipline." *Report AD/A-056–633*, U.S. Department of Commerce.

CHHIKARA, R. S., and REGISTER, D. T. (1979). "A numerical classification method for partitioning of a large multidimensional mixed data set." *Technometrics 21*, 531–537.

CHIEN, Y. T. (1978). *Interactive Pattern Recognition*. Marcel Dekker, Inc., New York.

CLARK, P., and EVANS, F. (1954). "Distance to nearest neighbor as a measure of spatial relationships in populations." *Ecology 35*, 445–453.

CLIFFORD, H. T., and STEPHENSON, W. (1975). *An Introduction to Numerical Classification*. Academic Press, Inc., New York.

COGGINS, J. M. (1982). "A framework for texture analysis based on spatial filtering." Ph.D. thesis, Department of Computer Science, Michigan State University, East Lansing.

COGGINS, J. M., and JAIN, A. K. (1985). "A spatial filtering approach to texture analysis." *Pattern Recognition Letters 3*, 195–203.

COHEN, A., GNANADESIKAN, R., KETTENRING, J. R., and LANDWEHR, J. M. (1977). "Methodological developments in some applications of clustering." In *Applications of Statistics* (P. R. Krishnaiah, ed.), North-Holland Publishing Company, Amsterdam, pp. 141–162.

COLE, A. J., ed. (1969). *Numerical Taxonomy*. Academic Press, Inc., New York.

COLEMAN, G. B., and ANDREWS, H. C. (1979). "Image segmentation by clustering." *Proceedings of the IEEE 67*, 773–785.

CONOVER, W. J. (1971). *Practical Nonparametric Statistics*. John Wiley & Sons, Inc., New York.

CONOVER, W. J., BEMENT, T. R., and IMAN, R. L. (1979). "On a method for detecting clusters of possible uranium deposits." *Technometrics 21*, 277–282.

COOLEY, W. W., AND LOHNES, P. R. (1971). *Multivariate Data Analysis*. John Wiley & Sons, Inc., New York.

CORMACK R. M. (1971). "A review of classification." *Journal of the Royal Statistical Society A134*, 321–367.

COX, D. R., and ISHAM, V. (1980). *Point Processes*. Chapman & Hall Ltd., London.

COX, T. F., and LEWIS, T. (1976). "A conditional distance ratio method for analyzing spatial patterns." *Biometrika 63*, 483–491.

CRAMER, H. (1945). *Mathematical Models of Statistics*. Princeton University Press, Princeton, N.J.

CROSS, G. R. (1980). "Some approaches to measuring clustering tendency." *Technical Report TR-80–03*, Department of Computer Science, Michigan State University, East Lansing.

CROSS, G. R., and JAIN, A. K. (1982). "Measurement of clustering tendency." *Proceedings IFAC Symposium on Digital Control*, New Delhi, pp. 24–29.

CUNNINGHAM, J. P., and SHEPARD, R. N. (1974). "Monotone mapping of similarities into a general metric space." *Journal of Mathematical Psychology 11*, 335–363.

CUNNINGHAM, K. M., and OGILVIE, J. C. (1972). "Evaluation of hierarchical grouping techniques: a preliminary study." *Computer Journal 15*, 209–213.

DAVIES, D. L., and BOULDIN, D. W. (1979). "A cluster separation measure." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 1*, 224–227.

DAVIS, L., and MITICHE, A. (1982). "MITES: a model-driven iterative texture segmentation procedure." *Computer Graphics and Image Processing 19*, 95–110.

DAY, N. E. (1969). "Estimating the components of a mixture of two normal distributions." *Biometrika 56*, 463–474.

DAY, W. H. E., and EDELSBRUNNER, H. (1984). "Efficient algorithms for agglomerative hierarchical clustering methods." *Journal of Classification 1*, 7–24.

DEAK, I. (1980). "Fast procedures for generating stationary normal vectors." *J. of Statistical Computation and Simulation 10*, 225–242.

DEVIJVER, P. A., and KITTLER, J. (1982). *Pattern Recognition: A Statistical Approach*. Prentice-Hall International, Hemel Hemstead, Hertfordshire, England.

DEWDNEY, A. K. (1986). "Computer Recreations." *Scientific American*, January, 16–25.

DIACONIS, P., and EFRON, B. (1983). "Computer-intensive methods in statistics." *Scientific American*, May, 116–130.

DIDAY, E., and SIMON, J. C. (1976). "Cluster analysis." In *Digital Pattern Recognition* (K. S. Fu, ed.), Springer-Verlag, Berlin, pp. 47–94.

DIETZ, E. J. (1983). "Permutation tests for association between 2 distance matrices." *Systematic Zoology 32*, 21–26.

DIGGLE, P. J. (1979). "On parameter estimation and goodness-of-fit testing for spatial point patterns." *Biometrics 35*, 87–101.

DIGGLE, P. J. (1983). *Statistical Analysis of Spatial Point Patterns*. Academic Press, Inc., New York.

DIXON, J. K. (1979). "Pattern recognition with partly missing data." *IEEE Transactions on Systems, Man and Cybernetics SMC 9*, 617–621.

DUBES, R. (1968). *Theory of Applied Probability*. Prentice-Hall, Inc., Englewood Cliffs, N.J.

DUBES, R. (1986). "Experiments in estimating the number of clusters." *Technical Report MSU-ENGR-86-019*, Department of Computer Science, Michigan State University, East Lansing.

DUBES, R., and JAIN, A. K. (1976). "Clustering techniques: the user's dilemma." *Pattern Recognition 8*, 247–260.

DUBES, R., and JAIN, A. K. (1979). "Validity studies in clustering methodologies." *Pattern Recognition 11*, 235–254.

DUBES, R., and JAIN, A. K. (1980). "Clustering methodologies in exploratory data analysis." In *Advances in Computers*, Vol. 19 (M. C. Yovits, ed.), Academic Press, Inc., New York, pp. 113–215.

DUBES, R., and ZENG, G. (1987). "A test for spatial homogeneity in cluster analysis." *Journal of Classification* 4, 33–56.

DUDA, R. O., and HART, P. E. (1973). *Pattern Classification and Scene Analysis*. John Wiley & Sons, Inc., New York.

DUNN, J. C. (1974). "Well-separated clusters and optimal fuzzy partitions." *Journal of Cybernetics 4*, 95–104.

DURAN, B. S., and ODELL, P. L. (1974). *Cluster Analysis: A Survey*. Springer-Verlag, Berlin.

EDELBROCK, C. (1979). "Mixture model tests of hierarchical clustering algorithms—problem of classifying everybody." *Multivariate Behavioral Research 14*, 367–384.

EDELBROCK, C., and MCLAUGHLIN, B. (1980). "Hierarchical cluster analysis of intraclass correlations: a mixture model study." *Multivariate Behavioral Research 15*, 299–318.

EDWARDS, A. W. F., and CAVALLI-SFORZA, L. L. (1965). "A method for cluster analysis." *Biometrics 21*, 362–375.

EFRON, B. (1979). "Bootstrap methods: another look at the jackknife." *Applied Statistics 7*, 1–26.

EFRON, B. (1981). "Nonparametric standard errors and confidence intervals." *Canadian Journal of Statistics 9*, 139–172.

EFRON, B. (1982). "The bootstrap and other resampling plans." *SIAM CBMS-NSF Monograph*.

EFRON, B. (1983). "Estimating the error rate of the prediction rule: improvement on cross-validation." *Journal of the American Statistical Association 78*, 316–331.

EFRON, B., and GONG, G. (1983). "A leisurely look at the bootstrap, the jackknife, and cross-validation." *The American Statistician 37*, 36–48.

EIGEN, D. J., FROMM, F. R., and NORTHOUSE, R. A. (1974). "Cluster analysis based on dimensional information with applications to feature selection and classification." *IEEE Transactions on Systems, Man and Cybernetics SMC 4*, 284–294.

Engelman, L., and Hartigan, J. A. (1969). "Percentage points of a test for clusters." *Journal of the American Statistical Association 64*, 1647–1648.

Enslein, K., Ralston, A., and Wilf, H. S. (1977). *Statistical Methods for Digital Computers*. John Wiley & Sons, Inc., New York.

Erdos, P., and Renyi, A. (1960). "On the evolution of random graphs." *Publication of the Mathematical Institute of the Hungarian Academy of Sciences 5*, 17–61.

Everitt, B. S. (1974). *Cluster Analysis*. John Wiley & Sons, Inc., New York.

Everitt, B. S. (1978). *Graphical Techniques for Multivariate Data*. Elsevier North-Holland, Inc., New York.

Everitt, B. S. (1979). "Unresolved problems in cluster analysis." *Biometrics 35*, 169–181.

Everitt, B. S., and Nicholls, P. (1975). "Visual techniques for representing multivariate data." *The Statistician 24*, 37–49.

Farris, J. S. (1969). "On the cophenetic correlation coefficient." *Systematic Zoology 18*, 279–285.

Fehlauer, J., and Eisenstein, B. A. (1978). "Structural editing by a point density function." *IEEE Transactions on Systems, Man and Cybernetics SMC 8*, 362–370.

Ferguson, T. S. (1967). *Mathematical Statistics—A Decision Theoretic Approach*. Academic Press, Inc., New York.

Fillenbaum, S., and Rapoport, A. (1971). *Structures in the Subjective Lexicon*. Academic Press, Inc., New York.

Fisher, R. A. (1936). "The use of multiple measurements in taxonomic problems." *Annals of Eugenics 3*, 179–188.

Fisher, L., and Van Ness, J. W. (1971). "Admissible clustering procedures." *Biometrika 58*, 91–104.

Fishman, G. S., and Moore, L. R. (1982). "A statistical evaluation of multiplicative congruential random number generators with modulus 2^{31}-1." *Journal of the American Statistical Association 77*, 129–136.

Forgy, E. (1965). "Cluster analysis of multivariate data: efficiency versus interpretability of classifications." *Biometrics 21*, 768. (Abstract)

Fortier, J. J., and Solomon, H. (1966). "Clustering procedures." In *Multivariate Analysis* (P. R. Krishnaiah, ed.), Academic Press, Inc., New York, pp. 493–506.

Foutz, R. V. (1980). "A method for constructing exact tests from test statistics that have unknown null distributions." *Journal of Statistical Computation and Simulation 10*, 187–193.

Fowlkes, E. B., and Mallows, C. L. (1983a). "A method for comparing two hierarchical clusterings." *Journal of the American Statistical Association 78*, 553–569.

Fowlkes, E. B., and Mallows, C. L. (1983b). "Rejoinder." *Journal of the American Statistical Association 78*, 584.

Friedman, J. H., and Rafsky, L. C. (1979). "Multivariate generalizations of the Wald-Wolfowitz and Smirnov 2-sample tests." *Ann. Statistics 7*, 697–717.

Friedman, J. H., and Rafsky, L. C. (1981). "Graphics for the multivariate two-sample problem." *Journal of the American Statistical Association 76*, 277–287.

Friedman, H. P., and Rubin, J. (1967). "On some invariant criteria for grouping data." *Journal of the American Statistical Association 62*, 1159–1178.

Friedman, J. H., and Tukey, J. W. (1974). "A projection pursuit algorithm for exploratory data analysis." *IEEE Transactions on Computers C 23*, 881–890.

Frieze, A. M. (1980). "Probability analysis of some Euclidean clustering problems." *Discrete Applied Mathematics 2*, 295–309.

Fu, K. S. (1974). *Syntactic Methods in Pattern Recognition*. Academic Press, Inc., New York.

Fu, K. S. (1982). *Syntactic Pattern Recognition and Applications*. Prentice-Hall, Englewood Cliffs, N.J.

Fu, K. S., and Mui, J. K. (1981). "A survey on image segmentation." *Pattern Recognition 13*, 3–16.

Fukunaga, K. (1972). *Introduction to Statistical Pattern Recognition*. Academic Press, Inc., New York.

Fukunaga, K., and Ando, M. (1977). "The optimum nonlinear features for a scatter criterion in discriminant analysis." *IEEE Transactions on Information Theory IT 23*, 453–459.

Fukunaga, K., and Narendra, P. M. (1975). "A branch and bound algorithm for computing *k*-nearest neighbors." *IEEE Transactions on Computers C 24*, 750–753.

Fukunaga, K., and Olsen, D. R. (1971). "An algorithm for finding intrinsic dimensionality of data." *IEEE Transactions on Computers C 20*, 176–183.

Fukunaga, K., and Short, R. D. (1978). "Generalized clustering for problem localization." *IEEE Transactions on Computers C 27*, 176–181.

Garland, K. (1983). "An experiment in automatic hierarchical document classification." *Information Processing and Management 19*, 113–120.

Gelsema, E. S. (1980). "ISPAHAN: an interactive system for pattern analysis-structure and capabilities." In *Pattern Recognition in Practice* (E. S. Gelsema and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 481–491.

Gitman, I., and Levine, M. D. (1970). "An algorithm for detecting unimodal fuzzy sets and its application as a clustering technique." *IEEE Transactions on Computers C 19*, 583–593.

Gnanadesikan, R., Kettenring, J. R., and Landwehr, J. M. (1977). "Interpreting and assessing the results of cluster analysis." *Bulletin of the International Statistical Institute 47*, 451–463.

Goldberg, M., and Shlien, S. (1978). "A clustering scheme for multispectral images." *IEEE Transactions on Systems, Man and Cybernetics SMC 8*, 86–92.

Golden, R. R., and Meehl, P. E. (1980). "Detection of biological sex—an empirical test of cluster methods." *Multivariate Behavioral Research 15*, 475–496.

Gonzalez, R. C., and Thomason, M. G. (1978). *Syntactic Pattern Recognition*. Addison-Wesley Publishing Company, Inc., Reading, Mass.

Gonzalez, R. C., and Wintz, P. (1977). *Digital Image Processing*. Addison-Wesley Publishing Company, Inc., Reading, Mass.

Good, I. J. (1977). "The botryology of botryology." In *Classification and Clustering* (J. Van Ryzin, ed.), Academic Press, Inc., New York, pp. 73–94.

Goodall, D. W. (1966). "A new similarity index based on probability." *Biometrics 22*, 882–907.

Goodman, L. A., and Kruskal, W. H. (1954). "Measures of association for cross-classifications." *Journal of the American Statistical Association 49*, 732–764.

Gordon, A. D. (1981). *Classification*. Chapman & Hall Ltd., London.

Gordon, A. D., and Henderson, J. T. (1977). "Algorithm for Euclidean sum of squares classification." *Biometrics 33*, 355–362.

Gowda, K. C. (1984). "A feature reduction and unsupervised classification algorithm for multispectral data." *Pattern Recognition 17*, 667–676.

Gowda, K. C., and Krishna, G. (1978). "Agglomerative clustering using the concept of mutual nearest neighborhood." *Pattern Recognition 10*, 105–112.

Gower, J. C. (1967). "A comparison of some methods of cluster analysis." *Biometrics 23*, 623–637.

Gower, J. C. (1971). "A general coefficient of similarity and some of its properties." *Biometrics 27*, 857–872.

Gower, J. C. (1977). "The analysis of asymmetry and orthogonality." In *Recent Developments in Statistics* (J. R. Barra, F. Brodeau, G. Romier, and B. van Cutsem, eds.), North-Holland Publishing Company, Amsterdam, pp. 109–123.

Gower, J. C., and Legendre, P. (1986). "Metric and Euclidean properties of dissimilarity coefficients." *Journal of Classification 3*, 5–48.

Gower, J. C., and Ross, G. J. S. (1969). "Minimum spanning trees and single-linkage cluster analysis." *Applied Statistics 18*, 54–64.

Gray, J. B., and Ling, R. F. (1984). "*K*-clustering as a detection tool for influential subsets in regression." *Technometrics 26*, 305–318.

Green, P. E., and Carmone, F. J. (1970). *Multidimensional Scaling and Related Techniques in Marketing Analysis*. Allyn and Bacon, Inc., Boston.

Green, P. E., and Carroll, J. D. (1976). *Mathematical Tools for Applied Multivariate Analysis*. Academic Press, Inc., New York.

Greig-Smith, P. (1964). *Quantitative Plant Ecology*, 2nd ed. Butterworth & Company (Publishers) Ltd., London.

Griffith, D. A. (1983). "The boundary value problem in spatial statistical analysis." *Journal of Regional Science 23*, 377–387.

Griffith, D. A. (1985). "An evaluation of correction techniques for boundary effects in spatial statistical analysis—contemporary methods." *Geographical Analysis 17*, 81–88.

Gross, A. L. (1972). "A Monte Carlo study of the accuracy of a hierarchical grouping procedure." *Multivariate Behavioral Research 7*, 379–389.

Hall, A. V. (1967). "Methods for demonstrating resemblance in taxonomy and ecology." *Nature 214*, 830–831.

Hall, A. V. (1969). "Group forming and discrimination with homogeneity functions." In *Numerical Taxonomy* (A. J. Cole, ed.), Academic Press, Inc., New York, pp. 53–67.

Hall, D. J., Duda, R. O., Huffman, D. A., and Wolf, D. E. (1973). "Development of new pattern recognition methods." *Report AD-772614*, Aerospace Research Laboratories, Wright Patterson AFB, Ohio.

Hamdan, M. A., and Tsokos, C. P. (1971). "An information measure of association in contingency tables." *Information and Control 19*, 174–179.

Hammersley, J. M. (1950). "The distribution of distance in a hypersphere." *Annals of Mathematical Statistics 21*, 447–452.

Hammersley, J. M., and Handscomb, D. C. (1965). *Monte Carlo Methods*. Methuen & Company Ltd., London.

Hansen, P., and DeLattre, M. (1978). "Complete-link cluster-analysis by graph coloring." *Journal of the American Statistical Association 73*, 397–403.

Harman, H. H. (1967). *Modern Factor Analysis*. The University of Chicago Press, Chicago.

Hartigan, J. A. (1975). *Clustering Algorithms*. John Wiley & Sons, Inc., New York.

Hartigan, J. A. (1985). "Statistical theory in clustering." *Journal of Classification 2*, 63–76.

Hawkins, D. M., Muller, M. W., and Ten Krooden, J. A. (1982). "Cluster analysis." In *Topics in Applied Multivariate Analysis*, Cambridge University Press, Cambridge, pp. 303–356.

Hines, W. C. S., and Hines, R. J. O. (1979). "The Eberhardt statistic and the detection of nonrandomness of spatial point distributions." *Biometrika 66*, 73–79.

Hoffman, R., and Jain, A. K. (1983). "A test of randomness based on the minimal spanning tree." *Pattern Recognition Letters 1*, 175–180.

Hoffman, R., and Jain, A. K. (1987). "Segmentation and classification of range images." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 9*, 608–620.

Hopkins, B. (1954). "A new method of determining the type of distribution of plant individuals." *Annals of Botany 18*, 213–226.

Hord, R. M. (1982). *Digital Image Processing of Remotely Sensed Data*. Academic Press, Inc., New York.

Howarth, R. J. (1973). "Preliminary assessment of a nonlinear mapping algorithm in a geological context." *Mathematical Geology 5*, 39–57.

Howe, S. E. (1978). "Estimating regions and clustering spatial data: analysis and implementation of methods using the voronoi diagram." Ph.D. thesis, Brown University, Providence, R. I.

Hubalek, Z. (1982). "Coefficients of association and similarity based on binary (presence, absence) data—an evaluation." *Biological Review 57*, 669–689.

Huber, P. J. (1985). "Projection pursuit." *Annals of Statistics 13*, 435–475.

Hubert, L. J. (1973). "Min and max hierarchical clustering using asymmetric similarity measures." *Psychometrika 38*, 63–72.

Hubert, L. J. (1974a). "Some applications of graph theory to clustering." *Psychometrika 39*, 283–309.

Hubert, L. J. (1974b). "Approximate evaluation techniques for the single-link and complete-link hierarchical clustering procedures." *Journal of the American Statistical Association 69*, 698–704.

Hubert, L. J. (1979). "Matching models in the analysis of cross-classifications." *Psychometrika 44*, 21–41.

HUBERT, L. J., and ARABIE, P. (1985). "Comparing partitions." *Journal of Classification 2*, 193–218.

HUBERT, L. J., and BUSK, P. (1976). "Normative location theory: placement in continuous space." *Journal of Mathematical Psychology 14*, 187–210.

HUBERT, L. J., and GOLLEDGE, R. G. (1981). "A heuristic method for the comparison of related structures." *Journal of Mathematical Psychology 23*, 214–226.

HUBERT, L. J., and SCHULTZ, J. (1976). "Quadratic assignment as a general data-analysis strategy." *British Journal of Mathematical and Statistical Psychology 29*, 190–241.

HUBERT, L. J., and SUBKOVIAK, M. J. (1979). "Confirmatory inference and geometric models." *Psychological Bulletin 86*, 361–370.

INCE, F. (1981). "The application of the coalescence clustering algorithm to remotely sensed multispectral data." *Pattern Recognition 14*, 121–130.

ISHAM, V. (1981). "An introduction to spatial point processes and Markov random fields." *International Statistical Review 49*, 21–43.

ITTNER, D., and JAIN, A. K. (1985). "3D surface discrimination from local curvature measures." *Proceedings IEEE CVPR Conference*, June, pp. 119–123.

JAIN, A. K. (1987). "Advances in Statistical Pattern Recognition." *Pattern Recognition Theory and Applications* (P. A. Devijver, ed.), Springer Verlag, New York pp. 1–19.

JAIN, A. K. (1985). "Experiments in texture analysis using spatial filtering." *Proceedings IEEE Workshop on Languages for Automation*, Palme De Mallorca, June, pp. 66–70.

JAIN, A. K. (1986). "Cluster analysis." In *Handbook of Pattern Recognition and Image Processing* (T. Y. Young and K. S. Fu, eds.), Academic Press, Inc., New York, pp. 35–57.

JAIN, A. K., and CHANDRASEKARAN, B. (1982). "Dimensionality and sample size considerations in pattern recognition practice." In *Handbook of Statistics*, Vol. 2 (P. R. Krishnaiah and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 835–855.

JAIN, A. K., and DUBES, R. (1978). "Feature definition in pattern recognition with small sample size." *Pattern Recognition 10*, 85–97

JAIN, A. K., SMITH, S. P., and BACKER, E. (1980). "Segmentation of muscle-cell pictures." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 2*, 232–242.

JAIN, A. K., DUBES, R., and CHEN, C. C. (1987). "Bootstrapping techniques for error estimation." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 9*, 628–633.

JARDINE, N., and SIBSON, R. (1971). *Mathematical Taxonomy*. John Wiley & Sons, Inc., New York.

JARVIS, R. A. (1978). "Shared near neighbor maximal spanning trees for cluster analysis." *Proceedings Fourth International Conference on Pattern Recognition*, Kyoto, Japan, pp. 308–313.

JARVIS, R. A., and PATRICK, E. A. (1973). "Clustering using a similarity measure based on shared near neighbors." *IEEE Transactions on Computers C 22*, 1025–1034.

JENSEN, R. E. (1969). "A dynamic programming algorithm for cluster analysis." *Operations Research 17*, 1034–1057.

JOHNSON, S. C. (1967). "Hierarchical clustering schemes." *Psychometrika 32*, 241–254.

JOHNSON, M. E., TIETJEN, G. C., and BACKMAN, R. J. (1980). "A new family of probability distributions with applications to Monte Carlo studies.'' *Journal of the American Statistical Association 75*, 276–279.

Johnston, B., Bailey, T., and Dubes, R. (1979). ''A variation on a nonparametric clustering method.'' *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 1*, 400–408.

Julesz, B. (1975). ''Experiments in the visual perception of texture.'' *Scientific American 232*, 34–43.

Kakusho, O., and Mizoguchi, R. (1983). ''A new algorithm for non-linear mapping with applications to dimension and cluster analyses.'' *Pattern Recognition 16*, 109–117.

Kamel, M. T., Tory, E. M., and Jodrey, W. S. (1979). ''Distribution of *k*th nearest neighbors and its application to cluster settling in dispersions of equal spheres.'' *Powder Technology 24*, 19–34.

Kamenskii, V. S. (1977). ''Methods and models of nonmetric multidimensional scaling.'' *Automation and Remote Control 38*, 1212–1243.

Kamgar-Parsi, B. and Kanal, L. N. (1985). ''An improved branch and bound algorithm for computing *k*-nearest neighbors.'' *Pattern Recognition Letters 3*, 7–12.

Kaminuma, T., Takekawa, T., and Watanabe, S. (1969). ''Reduction of clustering problems to pattern recognition.'' *Pattern Recognition 1*, 195–205.

Katz, J. O., and Rohlf, F. J. (1973). ''Function-point cluster analysis.'' *Systematic Zoology 22*, 295–301.

Kelly, F. P., and Ripley, B. D. (1976). ''A note on Strauss's model for clustering.'' *Biometrika 63*, 357–360.

Kemeny, J. G., and Snell, J. L. (1962). *Mathematical Models in the Social Sciences*. Ginn and Company, Lexington, Mass.

Kempthorne, O. (1952). *Design and Analysis of Experiments*. John Wiley & Sons, Inc., New York.

Kendall, M. G. (1966). ''Discrimination and classification.'' In *Multivariate Analysis* (P. R. Krishnaiah, ed.), Academic Press, Inc., New York, pp. 165–185.

Kendall, M. G. (1973). ''The basic problems of cluster analysis.'' In *Discriminant Analysis and Applications* (T. Cacoullos, ed.), Academic Press, Inc., New York, pp. 179–191.

Kinderman, A. J., and Ramage, J. G. (1976). ''Computer generation of normal random variables.'' *Journal of the American Statistical Association 71*, 893–896.

King, B. (1967). ''Step-wise clustering procedures.'' *Journal of the American Statistical Association 69*, 86–101.

Kirkpatrick, S. (1984). ''Optimization by simulated annealing: quantitative studies.'' *Journal of Statistical Physics 34*, 975–986.

Kittler, J. (1976). ''A locally sensitive method for cluster analysis.'' *Pattern Recognition 8*, 22–33.

Kittler, J. (1978). ''Classification of incomplete pattern vectors using modified discriminant functions,'' *IEEE Transactions on Computers C 27*, 367–374.

Klahr, D. (1969). ''A Monte Carlo investigation of the statistical significance of Kruskal's nonmetric scaling procedure.'' *Psychometrika 34*, 319–330.

Kleiner, B., and Hartigan, J. A. (1981). ''Representing points in many dimensions by trees and castles.'' *Journal of the American Statistical Association 76*, 260–269.

KNUTH, D. E. (1969). *The Art of Computer Programming*, Vol. 2. Addison-Wesley Publishing Company, Inc., Reading, Mass.

KOONTZ, W. L. G., and FUKUNAGA, K. (1972). "A nonlinear feature extraction algorithm using distance transformation." *IEEE Transactions on Computers C 21*, 56–63.

KOONTZ, W. L. G., NARENDRA, P. M., and FUKUNAGA, K. (1975). "A branch and bound clustering algorithm." *IEEE Transactions on Computers C 23*, 908–914.

KOONTZ, W. L. G., NARENDRA, P. M., and FUKUNAGA, K. (1976). "A graph-theoretic approach to nonparametric cluster analysis." *IEEE Transactions on Computers C 25*, 936–944.

KORFHAGE, R. R. (1974). *Discrete Computational Structures*. Academic Press, Inc., New York.

KRISHNAIAH, P. R., and KANAL, L. N., eds. (1982). *Handbook of Statistics*, Vol. 2. North-Holland Publishing Company, Amsterdam.

KRUSKAL, J. B. (1964a). "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis." *Psychometrika 29*, 1–27.

KRUSKAL, J. B. (1964b). "Nonmetric multidimensional scaling: a numerical method." *Psychometrika 29*, 115–129.

KRUSKAL, J. B. (1971). "Comments on 'A nonlinear mapping for data structure analysis.'" *IEEE Transactions on Computers C 20*, 1614.

KRUSKAL, J. B. (1977a). "Multidimensional scaling and other methods for discovering structure." In *Statistical Methods for digital Computers* (K. Enslein, A. Ralston, and H. S. Wilf, eds.), John Wiley & Sons, Inc., pp. 296–339.

KRUSKAL, J. B. (1977b). "The relationship between multidimensional scaling and clustering." In *Classification and Clustering* (J. Van Ryzin, ed.), Academic Press, Inc., New York, pp. 7–44.

KRUSKAL, J. B., and CARROLL, J. D. (1969). "Geometrical models and badness-of-fit functions." In *Multivariate Analysis II* (P. R. Krishnaiah, ed.), Academic Press, Inc., New York, pp. 639–671.

KRUSKAL, J. B., and HART, R. E. (1966). "A geometric intrepretation of diagnostic data from a digital machine: based on a study of the Morris, Illinois Electronic Control Office." *Bell System Technical Journal 45*, 1299–1338.

KRZANOWSKI, W. J. (1979). "Some exact percentage points of a statistic useful in analysis of variance and principal component analysis." *Technometrics 21*, 261–263.

KUIPER, F. K., and FISHER, L. A. (1975). "Monte Carlo comparison of six clustering procedures." *Biometrics 31*, 777–783.

LACHENBRUCH, P. A., and GOLDSTEIN, M. (1979). "Discriminant analysis." *Biometrics 35*, 69–85.

LANCE, G. N., and WILLIAMS, W. T. (1967). "A general theory of classificatory sorting strategies: II. Clustering systems." *Computer Journal 10*, 271–277.

LEE, K. L. (1979). "Multivariate tests for clusters." *Journal of the American Statistical Association 74*, 708–714.

LEE, R. C. T. (1981). "Clustering analysis and its applications." In *Advances in Information System Science* (J. T. Tou, ed.), Plenum Press, New York, pp. 80–150.

LEE, D. T., and PREPARATA, F. P. (1984). "Computational geometry—a survey." *IEEE Transactions on Computers C 33*, 1072–1101.

Lee, R. C. T., Slagle, J. R., and Blum, H. (1977). "A triangulation method for the sequential mapping of points from N-space to two-space." *IEEE Transactions on Computers C 26*, 288–292.

Lefkovitch, L. P. (1980). "Conditional clustering." *Biometrics 36*, 43–58.

Lesaffre, E. (1983). "Normality tests and transformations." *Pattern Recognition Letters 1*, 187–199.

Levine, D. M. (1977a). "Multivariate analysis of the visual information processing of numbers." *Journal of Multivariate Behavioral Research 12*, 347–355.

Levine, D. M. (1977b). "Nonmetric multidimensional scaling and hierarchical clustering—procedure for investigations of perception of sports." *Research Quarterly 48*, 341–348.

Levine, D. M. (1978). "Monte-Carlo study of Kruskal's variance based on measure of stress." *Psychometrika 43*, 307–315.

Li, X. (1984). A Probabilistic Association Measure for Pattern Recognition. Ph.D. Thesis, Department of Computer Science, Michigan State University, East Lansing.

Li, X., and Dubes, R. C. (1984). "The selection of significant dichotomous features." *Proceedings Seventh International Conference on Pattern Recognition*, Montreal, 260–264.

Li, X., and Dubes, R. C. (1985). "The first stage in two-stage template matching." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 7*, 700–707.

Liebetrau, A. M. (1977). "Tests of randomness in two dimensions." *Communications in Statistics—Theory and Methods A6*(14), 1367–1383.

Lindman, H., and Caelli, T. (1978). "Constant curvature Riemannian scaling." *Journal of Mathematical Psychology 17*, 89–109.

Ling, R. F. (1972). "On the theory and construction of k-clusters." *Computer Journal 15*, 326–332.

Ling, R. F. (1973a). "Probability theory of cluster analysis." *Journal of the American Statistical Association 68*, 159–164.

Ling, R. F. (1973b). "The expected number of components in random linear graphs." *Annals of Probability 1*, 876–881.

Ling, R. F. (1975). "An exact probability distribution on the connectivity of random graphs." *Journal of Mathematical Psychology 12*, 90–98.

Ling, R. F., and Killough, G. S. (1976). "Probability tables for cluster analysis based on a theory of random graphs." *Journal of the American Statistical Association 71*, 293–300.

Ling, R. F., and Pratt, J. W. (1984). "The accuracy of Peizer approximations to the hypergeometric distribution, with comparison to some other approximations." *Journal American Statistical Association 79*, 49–60.

Lingoes, J. C. (1971). "Some boundary conditions for a monotone analysis of symmetric matrices." *Psychometrika 36*, 195–203.

Lorr, M. (1983). *Cluster Analysis for Social Scientists*. Jossey-Bass, Inc., Publishers, San Francisco.

Lu, S. Y., and Fu, K. S. (1978). "A sentence-to-sentence clustering procedure for pattern analysis." *IEEE Transactions on Systems, Man and Cybernetics SMC 8*, 381–389.

Lumelsky, V. (1982). "A combined algorithm for weighting the variables and clustering in the clustering problem." *Pattern Recognition 15*, 53–60.

MacCallum, R. C. (1974). "Relations between factor-analysis and multidimensional scaling." *Psychological Bulletin 81*, 505–516.

MacCallum, R. C., and Cornelius, E. T. (1977). "A Monte Carlo investigation of recovery of structure by ALSCAL." *Psychometrika 42*, 401–428.

Magnuski, H. S. (1975). "Minimal spanning tree clustering method." *Communications of ACM 18*, 119.

Mandelbrot, B. B. (1977). *The Fractal Geometry of Nature*. W. H. Freeman and Company, Publishers, New York.

Mantel, N. (1967). "The detection of disease clustering and a generalized regression approach." *Cancer Research 27*, 209–220.

Mantock, J. M., and Fukunaga, K. (1980). "A two-dimensional display for multiclass multivariate data." In *Pattern Recognition in Practice* (E. S. Gelsema and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 361–368.

Marriott, F. H. C. (1979). "Barnard's Monte Carlo tests: how many simulations?" *Applied Statistics 28*, 75–77.

Marriott, F. H. C. (1982). "Optimization methods of cluster-analysis." *Biometrika 69*, 417–421.

Matula, D. W. (1977). "Graph theoretic techniques for cluster analysis algorithms." In *Classification and Clustering* (J. Van Ryzin, ed.), Academic Press, Inc., New York, pp. 95–129.

Matula, D. W., and Sokal, R. R. (1980). "Properties of Gabriel graphs relevant to geographic variation research and the clustering of points in the plane." *Geographical Analysis 12*, 205–222.

McClain, J. O., and Rao, V. R. (1974). "Tradeoffs and conflicts in evaluation of health service alternatives: methodology for analysis." *Health Services Research*, 35–52.

McLachlan, G. J. (1982). "The classification and mixture maximum likelihood approaches to cluster analysis." In *Handbook of Statistics*, Vol. 2 (P. R. Krishnaiah and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 199–208.

McNeill, D. R. (1977). *Interactive Data Analysis*. John Wiley & Sons, Inc., New York.

McQueen, J. B. (1967). "Some methods of classification and analysis of multivariate observations." *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297.

Mead, R. (1974). "A test for spatial pattern at several scales using data from a grid of contiguous quadrats." *Biometrics 30*, 295–308.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). "Equation of state calculations by fast computing machines." *Journal of Chemical Physics 21*, 1087–1092.

Michalski, R. S., and Stepp, R. E., III (1983). "Automated construction of classifications: conceptual clustering versus numerical taxonomy." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 5*, 396–410.

Milligan, G. W. (1979). "Ultrametric hierarchical clustering algorithms." *Psychometrika 44*, 343–346.

Milligan, G. W. (1980). "An examination of the effect of six types of error perturbation on fifteen clustering algorithms." *Psychometrika 45*, 325–342.

Milligan, G. W. (1981). "A Monte-Carlo study of 30 internal criterion measures for cluster-analysis." *Psychometrika 46*, 187–195.

Milligan, G. W. (1985). "An algorithm for generating artificial test clusters." *Psychometrika 50*, 123–127.

Milligan, G. W., and Cooper, M. C. (1985). "An examination of procedures for determining the number of clusters in a data set." *Psychometrika 50*, 159–179.

Milligan, G. W., and Isaac, P. D. (1980). "The validation of four ultrametric clustering algorithms." *Pattern Recognition 12*, 41–51.

Milligan, G. W., and Schilling, D. A. (1985). "Asymptotic and finite-sample characteristics of four external criterion measures." *Multivariate Behavioral Research 20*, 97–109.

Milligan, G. W., Soon, S. C., and Sokol, L. M. (1983). "The effect of cluster size, dimensionality, and the number of clusters on recovery of true cluster structure." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 5*, 40–47.

Mitchell, O. R., and Carlton, S. C. (1978). "Image segmentation using a local extrema texture measure." *Pattern Recognition 10*, 205–210.

Mizoguchi, R., and Shimura, M. (1980). "A nonparametric algorithm for detecting clusters using hierarchical structure." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 2*, 292–300.

Mojena, R. (1975). "Hierarchical grouping methods and stopping rules: An evaluation." *Computer J. 20*, 359–363.

Moreau, J. V., and Jain, A. K. (1986). How many clusters?" *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, Miami Beach, pp. 634–636.

Mucciardi, A. N., and Gose, E. E. (1972). "An automatic clustering algorithm and its properties in high-dimensional spaces." *IEEE Transactions on Systems, Man and Cybernetics SMC 2*, 247–254.

Mui, J. K., and Fu, K. S. (1980). "Automated classification of nucleated blood cells using a binary tree classifier." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 2*, 429–443.

Narendra, P. M., and Fukunaga, K. (1977). "A branch and bound algorithm for feature subset selection." *IEEE Transactions on Computers C 26*, 917–922.

Narendra, P. M., and Goldberg, M. (1977). "A non-parametric clustering scheme for LANDSAT." *Pattern Recognition 9*, 207–215.

Naus, J. I. (1966a). "A power comparison of two tests on non-random clustering." *Technometrics 8*, 493–517.

Naus, J. I. (1966b). "Some probabilities, expectations, and variances for the size of largest clusters and smallest intervals." *Journal of the American Statistical Association 61*, 1191–1199.

Naus, J. I. (1982). "Approximations for distributions of scan statistics." *Journal of the American Statistical Association 77*, 177–183.

Neyman, J., and Scott, E. L. (1972). "Processes of clustering and applications." In *Stochastic Point Processes: Statistical Analysis, Theory, and Applications* (P. A. W. Lewis, ed.), John Wiley & Sons, Inc., New York, pp. 646–681.

Ni, L. M., and Jain, A. K. (1985). "A VLSI systolic architecture for pattern clustering." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 7*, 80–89.

NEIMANN, H., and WEISS, J. (1979). "A fast converging algorithm for nonlinear mapping of high-dimensional data to a plane." *IEEE Transactions on Computers C 28*, 142–147.

ODELL, P. L., and BASU, J. P. (1976). "Concerning several methods for estimating crop acreages using remote sensing data." *Communications in Statistics—Theory and Methods A5*(12), 1091–1114.

OGATA, Y., and TANEMURA, M. (1981). "Estimation of interaction potentials of spatial point patterns through the maximum likelihood procedure." *Annals of the Institute of Statistical Mathematics 33*, 315–338.

OGATA, Y., and TANEMURA, M. (1984). "Likelihood analysis of spatial point patterns." *Journal of the Royal Statistical Society 46*, 496–518.

OHLANDER, R., PRICE, K., and REDDY, D. R. (1979). "Picture segmentation using a recursive region splitting method." *Computer Graphics and Image Processing 8*, 3–12.

OKADA, T., and TOMITA, S. (1985). "An optimal orthonormal system for discriminant analysis." *Pattern Recognition 18*, 139–144.

OSMER, P. S. (1982). "Quasars as probes of the distant and early universe." *Scientific American*, February, 126–138.

PAGE, R. L. (1974). "Minimal spanning tree clustering method." *Communications of the ACM 17*, 321–323.

PANAYIRCI, E., and DUBES, R. C. (1983). "A test for multidimensional clustering tendency." *Pattern Recognition 16*, 433–444.

PEAY, E. R. (1975). "Nonmetric grouping—clusters and cliques." *Psychometrika* 40, 297–313.

PETTIS, K., BAILEY, T., JAIN, A. K., and DUBES, R. (1979). "An intrinsic dimensionality estimator from near-neighbor information." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 1*, 25–37.

PIELOU, E. C. (1969). *An Introduction to Mathematical Ecology*. John Wiley & Sons, Inc., New York.

POLLARD, D. (1981). "Strong consistency of *k*-means clustering." *Annals of Statistics 9*, 135–140.

PRIM, R. C. (1957). "Shortest connection networks and some generalizations." *Bell System Technical Journal 36*, 1389–1401.

PYKETT, C. E. (1978). "Improving the efficiency of Sammon's nonlinear mapping by using clustering archetypes." *Electronics Letters 14*, 799–800.

RABINER, L. R., and WILPON, J. G. (1979). "Considerations in applying clustering techniques to speaker-independent word recognition." *Journal of the American Statistical Association 66*, 663–673.

RAMMAL, R., TOULOUSE, G., and VIRASOVO, M. A. (1986). "Ultrametricity for physicists." *Review of Modern Physics 58*, 765–788.

RAMSAY, J. O. (1977). "Maximum likelihood estimation in multidimensional scaling." *Psychometrika 42*, 241–266.

RAMSAY, J. O. (1978). "Confidence regions for multidimensional scaling analysis." *Psychometrika 43*, 145–160.

RAND, W. M. (1971). "Objective criteria for the evaluation of clustering methods." *Journal of the American Statistical Association 66*, 846–850.

RAO, M. R. (1971). "Cluster analysis and mathematical programming." *Journal of the American Statistical Association 66*, 622–626.

RAO, C. R. (1977). "Cluster analysis applied to a study of race mixture in human populations." In *Classification and Clustering* (J. Van Ryzin, ed.), Academic Press, Inc., New York, pp. 175–197.

RAPOPORT, A., and FILLENBAUM, S. (1972). "An experimental study of semantic structures." In *Multidimensional Scaling*, Vol. 2, *Applications* (A. K. Romney, R. N. Shepard, and S. B. Nerlove, eds.), pp. 93–131. Seminar Press, Inc., New York.

RIPLEY, B. D. (1977). "Modelling spatial patterns." *Journal of the Royal Statistical Society B39*, 172–212.

RIPLEY, B. D. (1979a). "Tests of randomness for spatial point patterns." *Journal of the Royal Statistical Society B41*, 368–374.

RIPLEY, B. D. (1979b). "Simulating spatial patterns: Dependent samples from a multivariate density." *Applied Statistics 28*, 109–112.

RIPLEY, B. D. (1981). *Spatial Statistics*. John Wiley & Sons, Inc., New York.

RIPLEY, B. D., and RASSON, J. P. (1977). "Finding the edge of a Poisson forest." *Journal of Applied Probability 14*, 483–491.

RIPLEY, B. D., and SILVERMAN, B. W. (1978). "Quick tests for spatial interaction." *Biometrika 65*, 641–642.

ROGERS, A. (1974). *Statistical Analysis of Spatial Dispersion*. Pion, London.

ROHLF, F. J. (1970). "Adaptive hierarchical clustering schemes." *Systematic Zoology 19*, 58–82.

ROHLF, F. J. (1973). "Hierarchical clustering using minimal spanning tree." *Computer Journal 16*, 93–95.

ROHLF, F. J. and FISHER, D. R. (1968). "Tests for hierarchical structure in random data sets." *Systematic Zoology 17*, 407–412.

ROMNEY, A. K., SHEPARD, R. N., and NERLOVE, S. B., eds. (1972). *Multidimensional Scaling*, Vol. 2, *Applications*. Seminar Press, Inc., New York.

ROSENFELD, A., and KAK, A. C. (1976). *Digital Picture Processing*. Academic Press, Inc., New York.

RUBIN, J. (1967). "Optimal classification into groups: an approach for solving the taxonomy problem." *Journal of Theoretical Biology 15*, 103–144.

RUSPINI, E. H. (1969). "A new approach to clustering." *Information and Control 15*, 22–32.

SALTON, G., WONG, A., and YANG, C. S. (1975). "A vector space model for automatic indexing." *Communications of the ACM 18*, 613–620.

SAMMON, J. W. (1969). "A nonlinear mapping for data structure analysis." *IEEE Transactions on Computers C 18*, 401–409.

SANDERSON, A. C., and WONG, A. K. C. (1980). "Pattern trajectory analysis of nonstationary multivariate data." *IEEE Transactions on Systems, Man and Cybernetics SMC 10*, 384–392.

SAUNDERS, R., and FUNK, G. M. (1977). "Poisson limits for a clustering model of Strauss." *Journal of Applied Probability 14*, 776–784.

SCHACHTER, B. (1978). "A nonlinear mapping algorithm for large databases." *Computer Graphics and Image Processing 7*, 271–278.

SCHACHTER, B. J., DAVIS, L. S., and ROSENFELD, A. (1978). "Some experiments in image segmentation by clustering of local feature values." *Pattern Recognition 11*, 19–28.

SCHER, A., SHNEIR, M., and ROSENFELD, A. (1982). "Clustering of collinear line segments." *Pattern Recognition 15*, 85–91.

SCHIKHOF, W. H. (1984). *An Introduction to p-adic Analysis. Studies in Advanced Mathematics*, Vol. 4. Cambridge University Press, Cambridge.

SCHULTZ, J. R., and HUBERT, L. J. (1973). "Data-analysis and connectivity of random graphs." *Journal of Mathematical Psychology 10*, 421–428.

SCHWARTZMANN, D. H., and VIDAL, J. J. (1975). "An algorithm for determining the topological dimensionality of point clusters." *IEEE Transactions on Computers C 24*, 1175–1182.

SCLOVE, S. L. (1977). "Population mixture-models and clustering algorithms." *Communications in Statistics—Theory and Methods A6*, 417–434.

SCOTT, A. J., and SYMONS, M. J. (1971). "Clustering methods based on likelihood ratio criteria." *Biometrics 27*, 387–397.

SEBESTYEN, G. S. (1962). *Decision-Making Processes in Pattern Recognition*. Macmillan Publishing Company, Inc., New York.

SEBESTYEN, G. S., and EDIE, J. (1966). "An algorithm for non-parametric pattern recognition." *IEEE Transactions on Electronic Computers EC 15*, 908–915.

SELIM, S. Z., and ISMAIL, M. A. (1984). "*K*-Means-type algorithms: a generalized convergence theorem and characterization of local optimality." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 6*, 81–87.

SHAFFER, E., DUBES, R., and JAIN, A. K. (1979). "Single-link characteristics of a mode-seeking algorithm." *Pattern Recognition 11*, 65–73.

SHANLEY, R. J., and MAHTAB, M. A. (1976). "Delineation and analysis of clusters in orientation data." *Mathematical Geology 8*, 9–23.

SHAPIRO, M. B., SCHEIN, S. J., and DE MONASTERIO, F. M. (1985). "Regularity and structure of the spatial pattern of blue cones of macaque retina." *Journal of the American Statistical Association 80*, 803–814.

SHEPARD, R. N. (1962). "The analysis of proximities: multidimensional scaling with an unknown distance function: 1." *Psychometrika 27*, 125–140.

SHEPARD, R. N. (1974). "Representation of structure in similarity data—problems and prospects." *Psychometrika 39*, 373–421.

SHEPARD, R. N. (1980). "Multidimensional scaling, tree-fitting, and clustering." *Science 210*, 390–398.

SHEPARD, R. N., and ARABIE, P. (1979). "Additive clustering: representation of similarities as combinations of discrete overlapping properties." *Psychological Review 86*, 87–123.

SHEPARD, R. N., and CARROLL, J. D. (1966). "Parametric representation of nonlinear data structures." In *Multivariate Analysis* (P. R. Krishnaiah, ed.), Academic Press, Inc., New York, pp. 561–592.

SHEPARD, R. N., ROMNEY, A. K., and NERLOVE, S. B. (1972). *Multidimensional Scaling*, Vol. 1, *Theory*. Seminar Press, Inc., New York.

SHORE, H. (1982). "Simple approximations for the inverse cumulative function, the density function, and the loss integral of the normal distribution." *Applied Statistics 31*, 108–114.

Shreider, Y. A. (1964). *Method of Statistical Testing: Monte Carlo Method.* Elsevier North-Holland, Inc., Amsterdam.

Sibson, R. (1971). "Some observations on a paper by Lance and Williams." *Computer Journal 14*, 156–157.

Sibson, R. (1980). "The Dirichlet tessellation as an aid in data analysis." *Scandanavian Journal of Statistics 7*, 14–20.

Siemiatycki, J. (1978). "Mantel's space-time clustering statistics: computing higher moments and a comparison of various data transforms." *Journal of Statistical Computation and Simulation 7*, 13–31.

Silverman, B. W. (1984). "Discussion of the paper by Dr. Diggle and Dr. Gratton." *Journal of the Royal Statistical Society 46*, 212–214.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis.* Chapman & Hall Ltd., London.

Silverman, B., and Brown, T. (1978). "Short distances, flat triangles and Poisson limits." *Journal of Applied Probability 15*, 815–825.

Singh, K. (1981). "An asymptotic accuracy of Efron's bootstrap." *Annals of Statistics 9*, 1187–1195.

Siromoney, G., Govindaraju, S. and Bagavandas, M. (1985). "Temple carvings of southern India." *Perspectives in Computing 5*, 34–43.

Sloane, N. J. A. (1984). "The packing of spheres." *Scientific American 250*, 116–125.

Smith, S. P. (1982). "Structure of multidimensional patterns." Ph.D. thesis, Department of Computer Science, Michigan State University, East Lansing.

Smith, S. P., and Dubes, R. C. (1980). "The stability of a hierarchical clustering." *Pattern Recognition 12*, 177–187.

Smith, S. P., and Jain, A. K. (1984). "Testing for uniformity in multidimensional data." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 6*, 73–81.

Smith, S. P., and Jain, A. K. (1985). "An experiment on using the Friedman–Rafsky test to determine the multivariate normality of a data set." *Proceedings of IEEE Computer Vision and Pattern Recognition Conference*, San Francisco, pp. 423–425.

Sneath, P. H. A. (1977). "Method for testing distinctness of clusters—test of disjunction of 2 clusters in Euclidean space as measured by their overlap." *Mathematical Geology 9*, 123–143.

Sneath, P. H. A., and Sokal, R. R. (1973). *Numerical Taxonomy.* W. H. Freeman and Company, Publishers, San Francisco.

Sokal, R. R. (1979). "Testing statistical significance of geographic-variation patterns." *Systematic Zoology 28*, 227–232.

Srivastava, J. N. (1973). "An information function approach to dimensionality analysis and curved manifold clustering." In *Multivariate Analysis III* (P. R. Krishnaiah, ed.), Academic Press, Inc., New York, pp. 369–382.

Stenson, H. H., and Knoll, R. L. (1969). "Goodness of fit for random rankings in Kruskal's nonmetric scaling procedure." *Psychological Bulletin 71*, 122–126.

Stepp, R. E., III, and Michalski, R. S. (1986). "Conceptual clustering: Inventing goal-oriented classifications of structured objects." In *Machine Learning*, Vol. 2 (R. S. Michalski, J. G. Carbonnell, and T. M. Mitchell, eds.), Morgan Kaufman Publishers, Los Altos, California, pp. 471–498.

STOCKMAN, G. (1980). ''Object detection via image registration.'' In *Pattern Recognition in Practice* (E. S. Gelsema and L. N. Kanal, eds.), Elsevier North-Holland, Inc., New York, pp. 75–86.

STOCKMAN, G., and ESTEVA, J. C. (1985). ''3D object pose from clustering with multiple views.'' *Pattern Recognition Letters 3*, 279–286.

STOCKMAN, G., KANAL, L., and KYLE, M. C. (1976). ''Structural pattern recognition of carotid pulse waves using a general waveform parsing system.'' *Communications of the ACM 19*, 688–695.

STRAUSS, J. S. (1973). ''Classification by cluster analysis.'' In Report of the International Pilot Study of Schizophrenia, Vol. 1. World Health Organization, Geneva, pp. 336–359.

STRAUSS, D. J. (1975). ''Model for clustering.'' *Biometrika 62*, 467–475.

SYMONS, M. J. (1981). ''Clustering criteria and multivariate normal mixtures.'' *Biometrics 37*, 35–43.

THOMPSON, H. K., JR., and WOODBURY, M. A. (1970). ''Clinical data representation in multidimensional space.'' *Computers and Biomedical Research 3*, 58–73.

TILTON, J. C., and STRONG, J. P. (1984). ''Analyzing remotely sensed data on the massively parallel processor.'' *Proceedings Seventh International Conference on Pattern Recognition*, Montreal, pp. 398–400.

TITTERINGTON, D. M., SMITH, A. F. M., and MAKOV, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions*. John Wiley & Sons, Inc., New York.

TORGERSON, W. S. (1958). *Theory and Methods of Scaling*. John Wiley & Sons, Inc., New York.

TORN, A. I. (1977). ''Cluster analysis using seed points and density-determined hyperspheres as an aid to global optimization.'' *IEEE Transactions on Systems, Man and Cybernetics SMC 7*, 610–616.

TOU, J. T., and HEYDORN, R. P. (1967). ''Some approaches to optimum feature extraction.'' In *Computer and Information Sciences II* (J. T. Tou, ed.), Academic Press, Inc., New York, pp. 57–89.

TOUSSAINT, G. T. (1974). ''Bibliography on estimation of misclassification.'' *IEEE Transactions on Information Theory IT 20*, 472–479.

TOUSSAINT, G. T. (1975). ''Subjective clustering and bibliography of books on pattern recognition.'' *Information Sciences 8*, 251–257.

TOUSSAINT, G. T. (1980). ''The relative neighborhood graph of a finite planar set.'' *Pattern Recognition 12*, 261–268.

TRUNK, G. V. (1976). ''Statistical estimation of the intrinsic dimensionality of a noisy signal collection.'' *IEEE Transactions on Computers C 25*, 165–171.

TRYON, R. C., and BAILEY, D. E. (1970). *Cluster Analysis*. McGraw-Hill, Inc., New York.

TUCERYAN, M. (1986). ''Extraction of perceptual structure in dot patterns.'' Ph.D. thesis, Department of Computer Science, University of Illinois, Urbana-Champaign.

TVERSKY, A. (1977). ''Features of similarity.'' *Psychological Review 84*, 327–352.

TZENG, O. C. S., and LANDIS, D. (1978). ''Three-mode multidimensional-scaling with points of view solutions.'' *Multivariate Behavioral Research 13*, 181–213.

URQUHART, R. (1982). "Graph theoretical clustering based on limited neighborhood sets." *Pattern Recognition 15*, 173–187.

VAN NESS, J. W. (1983). "Comment on paper by Fowlkes and Mallows." *Journal of the American Statistical Association 78*, 576–579.

VAN RYZIN, J., ed. (1977). *Classification and Clustering*. Academic Press, Inc., New York.

VINOD, H. D. (1969). "Integer programming and theory of grouping." *Journal of The American Statistical Association 64*, 506–519.

WANG, P. S. P. (1984). "An application of array grammar to clustering analysis for syntactic patterns." *Pattern Recognition 17*, 441–451.

WANG, D. K., URQUHART, R. B., and MACLEOD, E. S. (1983). "The equal-angle spanning tree mapping: a sequential method for projecting from *h*-space to 2-space." *Pattern Recognition Letters 2*, 69–73.

WARD, J. H., JR. (1963). "Hierarchical grouping to optimize an objective function." *Journal of the American Statistical Association 58*, 236–244.

WESZKA, J. (1978). "A survey of threshold selection techniques." *Computer Graphics and Image Processing 7*, 259–265.

WHARTON, S. W. (1983). "A generalized histogram clustering scheme for multidimensional image data." *Pattern Recognition 16*, 193–199.

WHITNEY, C. A. (1984). "Generating and testing pseudorandom numbers." *Byte 9*, 128.

WILKS, S. S. (1963). *Mathematical Statistics*. John Wiley & Sons, Inc., New York.

WILLIAMS, W. T., and LANCE, G. N. (1977). "Hierarchical classificatory methods." In *Statistical Methods for Digital Computers* (K. Enslein, A. Ralston, and H. S. Wilf, eds.), John Wiley & Sons, Inc., New York, pp. 269–295.

WILLIAMS, W. T., CLIFFORD, H. T., and LANCE, G. N. (1971). "Group-size dependence: a rationale for choice between numerical classifications." *Computer Journal 14*, 157–162.

WISHART, D. (1969). "Mode analysis: a generalization of nearest neighbor which reduces chaining effects." In *Numerical Taxonomy* (A. J. Cole, ed.), Academic Press, Inc., New York, pp. 282–308.

WISMATH, S. K., SOONG, H. P., and AKL, S. G. (1981). "Feature selection by interactive clustering." *Pattern Recognition 14*, 75–80.

WOLFE, J. H. (1970). "Pattern clustering by multivariate mixture analysis." *Multivariate Behavioral Research 5*, 329–350.

WONG, A. K. C., and LIU. T. S. (1977). "A decision-directed algorithm for discrete data." *IEEE Transactions on Computers C 26*, 75–82.

WONG, M. A., and LANE, T. (1983). "A *k*th nearest neighbor clustering procedure." *Journal of the Royal Statistical Society B45*, 362–368.

WONG, A. K. C., and WANG, D. C. C. (1979). "DECA: a discrete-valued data clustering algorithm." *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI 1*, 342–349.

WRIGHT, W. E. (1973). "A formalization of cluster analysis." *Pattern Recognition 5*, 273–282.

WYSE, N., DUBES, R., and JAIN, A. K. (1980). "A critical evaluation of intrinsic dimensionality algorithms.'' In *Pattern Recognition in Practice* (E. S. Gelsema and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 415–425.

YOUNG, F. W. (1970). ''Nonmetric multidimensional scaling: recovery of metric information.'' *Psychometrika 35*, 455–473.

YOUNG, F. W., TAKANE, Y., and LEWYCKYJ, R. (1980). ''ALSCAL—a multidimensional scaling package with several individual-differences options.'' *American Statistician 34*, 117–118.

ZADEH, L. A. (1965). ''Fuzzy sets.'' *Information and Control 8*, 338–353.

ZADEH, L. A. (1984). ''Making computers think like people.'' *IEEE Spectrum 21*, 26–32.

ZAGORUIKO, N. G., and YOLKINA, V. N. (1982). ''Inference and data tables with missing values.'' In *Handbook of Statistics*, Vol. 2 (P. R. Krishnaiah and L. N. Kanal, eds.), North-Holland Publishing Company, Amsterdam, pp. 493–500.

ZAHN, C. T. (1971). ''Graph-theoretical methods for detecting and describing Gestalt clusters.'' *IEEE Transactions on Computers C 20*, 68–86.

ZENG, G., and DUBES, R. (1984). ''The Sampling window in tests for randomness and regularity.'' *Proceedings 1984 Conference on Intelligent Systems*, Oakland University, Rochester.

ZENG, G., and DUBES, R. C. (1985a). ''A comparison of tests for randomness.'' *Pattern Recognition 18*, 191–198.

ZENG, G., and DUBES, R. C. (1985b). ''A test for spatial randomness based on *k-NN* distances.'' *Pattern Recognition Letters 3*, 85–91.

ZIMMERMAN, R., JACOBS, R., and FARR, J. (1982). ''A comparison of the accuracy of four methods of clustering jobs.'' *Applied Psychological Measurement 6*, 353–366.

Author Index

General Index