A STUDY OF CLUSTER ANALYSIS TECHNIQUES AND THEIR APPLICATIONS

SAAD MOWAFAC ASIM HAFIDH

Submitted for the

Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF ASTON IN BIRMINGHAM

SEPTEMBER 1981

A STUDY OF CLUSTER ANALYSIS TECHNIQUES AND THEIR APPLICATIONS

Saad Mowafac Asim Hafidh

Submitted to the
University of Aston in Birmingham
for the degree of
Doctor of Philosophy, 1981

Summary

This thesis seeks to describe the development of an inexpensive and efficient clustering technique for multivariate data analysis. The technique starts from a multivariate data matrix and ends with graphical representation of the data and pattern recognition discriminant function. The technique also results in distances frequency distribution that might be useful in detecting clustering in the data or for the estimation of parameters useful in the discrimination between the different populations in the data. The technique can also be used in feature selection. The technique is essentially for the discovery of data structure by revealing the component parts of the data.

The thesis offers three distinct contributions for cluster analysis and pattern recognition techniques. The first contribution is the introduction of transformation function in the technique of nonlinear mapping. The second contribution is the use of distances frequency distribution instead of distances time-sequence in nonlinear mapping. The third contribution is the formulation of a new generalised and normalised error function together with its optimal step size formula for gradient method minimisation.

The thesis consists of five chapters. The first chapter is the introduction. The second chapter describes multidimensional scaling as an origin of nonlinear mapping technique. The third chapter describes the first developing step in the technique of nonlinear mapping that is the introduction of "transformation function". The fourth chapter describes the second developing step of the nonlinear mapping technique. This is the use of distances frequency distribution instead of distances time-sequence. The chapter also includes the new generalised and normalised error function formulation. Finally, the fifth chapter, the conculsion, evaluates all developments and proposes a new program for cluster analysis and pattern recognition by integrating all the new features.

Key Words

NONLINEAR MAPPING, MULTIDIMENSIONAL SCALING, CLUSTER ANALYSIS

PATTERN RECOGNITION, MULTIVARIATE ANALYSIS

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. A. J. Harget, for his continuous concern and help throughout this study. I would like to thank Mr. K. J. Bowcock, Head of Computer Centre, for his kindness and encouragement. I would also like to thank all operators who helped me in my work, and Miss D. Cashman and Mrs. H. M. Turner for typing the manuscript.

CONTENTS

LIST OF FIGURES AND TABLES

" Either we may neglect a part of the multiple features which are found in the concrete thing (by what is called analysis) and select only one of them; or, neglecting their variety, we may concentrate the multiple characters into one."

Hegel

Hegel's Logic
Page 166,
Oxford University Press, 1975

CHAPTER ONE

INTRODUCTION

INTRODUCTION

The object of this work was to develop a computationally efficient and inexpensive multivariate data analysis technique for users in the different scientific fields.

The technique seeks to analyse multivariate data which is in the form of a matrix of m rows and n columns. Each row corresponds to a sample given by n numerical values. In order to have a meaningful result there are some conditions to be met. First the data should contain sufficient information content to be uncovered by the technique. Second, although the technique can analyse multivariate data without knowing the class-membership of the patterns, it is important to know them in order to evaluate the result of using the technique. Third and according to some studies, Gray (1976), the number of samples to the number of measurements ratio should be greater or equal to 3. However a smaller ratio does not necessarily mean that the technique is not useful to apply.

In analysing multivariate data the technique seeks to isolate the component parts of the data. If the data is composed of a concentration of patterns or points then the technique should reflect this. The analysis of the data is carried out in an objective manner. That is the technique is intended not to enhance clustering in the data although it can.

The technique results in the discrimination between the clusters in the data if clustering does exist. In order to discriminate between the clusters the technique seeks to find a hyperplane or hyperplanes in the n-dimensional patterns space between the clusters.

The position of the patterns with respect to the hyperplane is an indication to their cluster membership. Furthermore cluster membership can be quantitatively measured. This result is achieved by mapping the patterns or points in their n-dimensional space to a lower dimensionality space and especially to one-dimensional space. This is why in order to discriminate between patterns and to reveal clustering in data we resort to map the data from the higher to the lower dimension space.

Multivariate data originate from many scientific experiments. In chememistry the nonlinear mapping of Sammon (1969) was used in general applications, Koskinen (1975), Kowalski et al. (1975) and Kowalski et al. (1972). The same technique was used in the study of pharmacological activity of some organic compounds, Chu (1974) and Ting et al. (1973). Also and as a chemical application in archaeology, Kowalski (1974) and Boulle et al. (1979).

As it has been mentioned earlier the object of our work was the development of an inexpensive multivariate data analysis technique. This is specially the case with storage requirements. In addition the technique offers exceptional advantage by supplying the user with a simple linear function that can be used every time a knew unknown pattern is to be recognised.

This thesis consists of five chapters. In the second chapter we describe Shepard's multidimensional scaling technique and its development by Kruskal. In the same chapter we describe Sammon's nonlinear mapping technique and its developments.

Shepard's multidimensional technique was meant "for the discovery and representation of structures underlying matrices of similarity data", Shepard (1974). On the other hand Sammon's nonlinear mapping technique was meant "to detect and identify 'structure' which may be present in a list of N L-dimensional vectors", Sammon (1969). Multidimensional scaling is the origin of nonlinear mapping. In fact it has been demonstrated that nonlinear mapping is a special case of multidimensional scaling, Kruskal (1971). One essential difference between multidimensional scaling and nonlinear mapping is that the first analyses proximity (similarity) data and the second analyses multivariate data.

In the third chapter we describe the first step in developing the linear mapping method. The method has been developed by introducing what we call "transformation function". The function transforms the patterns in their higher n_1-dimensional space to the lower n_2-dimensional space. Geometrically the function is a hyperplane in the n_1-dimensional space. The transformation function mapping technique seeks to isolate the clusters in the patterns n_1-dimensional space by one or more hyperplanes. In the case of two-dimensional mapping two transformation functions have to be employed. The chapter also describes two important outcomes resulting from the introduction of transformation function. The first outcome is a computationally more efficient method and second the use of transformation function as a discriminant function in recognising patterns. Three data sets have been described in chapter three together with their results.

In the fourth chapter the second developing step is considered. The second step is the use of distances frequency distribution instead of distances time-sequence. The main target of this step was to minimise the cost of data storage. The result of this was a radical reduction in memory requirement. The chapter then discusses the results of two applications. In this chapter we also formulate a new error function that is generalised and normalised. The error function is also invariate against similarity transformations. The formulation is based on simple difference of squares. In addition the chapter formulates mathematical models for interpoints distances frequency distribution. The model is meant for the study of distances frequency distributions. It is useful to note here that the program described in chapter four employs transformation function together with distances frequency distribution.

Finally, the fifth chapter evaluates the theoretical and the practical results from transformation function and distances frequency distribution mapping. The chapter also includes the description of some structural properties and function of a new program for multivariate analysis technique. The program uses the generalised error function and an optimal step size gradient minimisation method. The technique is expected to carry out automatic pattern classification as well as providing graphical representation for data structure.

CHAPTER TWO

MULTIDIMENSIONAL SCALING

AND

NONLINEAR MAPPING

2.1 INTRODUCTION

In this chapter we will consider the two methods of Shepard (1962 a) and Sammon (1969) that analyse similarity and multivariate data respectively. Also the important developments on Sammon's method will be closely considered.

Both methods attempt to provide a graphical representation of a set of objects (or points) so as to give information about the relationship between them or their grouping. The graphical representation is in one, two or three-dimensional Euclidean space.

Although the two methods are similar they do differ in, first, the method of multidimensional scaling starts from the similarities between a set of objects.

In contrast the non-linear mapping method starts from a set of n_1-dimensional space points given in the form of multivariate data.

The two methods employ two different criteria that judge the progress and when to terminate the process of scaling or mapping. In multidimensional scaling the criterion is the measure of departure from monotonicity by the similarity-distance relationship. In non-linear mapping the criterion is the degree of difference between the distances in the n_1 and n_2-dimensional spaces.

2.2 SHEPARD

The multidimensional scaling method of Shepard (1962 a) seeks to obtain a spatial representation for a number of objects under consideration and their relationships. Normally the relationships are given in the form of similarities and serve as input. The similarities are extracted from the objects by comparing every two of them. The similarities can suitably be represented by a matrix form called the similarity matrix. Normally the similarities occupy the upper or lower part of the matrix. Each element s_{ij} of the similarity matrix represents the similarity between the i-th and j-th objects. The multidimensional scaling technique takes the matrix of similarity as input, and yields a configuration of points as output, in other words multidimensional scaling transforms a similarity matrix into distances between spatially represented points, Kruskal (1977).

The other aspect of multidimensional scaling technique is the monotic functional relationship between distance and similarity. Multidimensional scaling assumes that two or more similar objects have close proximity in the n-dimensional pattern space. Mathematically, the relationship between similarity and distance is assumed monotonic. The technique does not require that the form of the monotonic function is known. Furthermore, the technique can graphically extract the form of the monotonic function from the similarity matrix data. There are a number of familiar monotonic functions that arise from the different applications, Shepard (1962 b). The condition of monotonicity was further coupled with the condition of convexity, Shepard (1974).

Ideally, the similarity-distance functional relationship forms a perfectly monotonic relation. In such a case the spatial representation of the objects and their distances are a perfect reflection of the similarities between the objects. Such a configuration is regardedas opimal and it has zero departure from monotonicity. In contrast optimal spatial configurations always have their respective distance-dissimilarity functions depart from monotonicity. The obtaining of an optimal spatial configuration means that the departure from monotonicity is minimal.

Shepard multidimensional scaling employs a governing criterion. The criterion measures the degree of departure from monotonicity by the similarity-distance relationship formed by the data. The criterion used by Shepard (1962 a) takes the following form:

$$\text{criterion} = \Sigma(s_{ij} - s(d_{ij}))^2/(m(m-1)/2) \qquad (2.1)$$

where m is the number of objects in the dataset and s_{ij} is the similarity between i-th and j-th objects. The term $s(d_{ij})$ stands for the similarity of rank $R'(d_{ij})$, Shepard (1962 a).

The computation of the spatial configuration, where similarities between the objects are represented by distances starts from a "guess" configuration and the above criterion is employed to measure the degree of departure from monotonicity. The process of approaching the optimal spatial configuration is done iteratively and it stops when the value of the criterion reaches a tolerated value. The output from this process is a set of m points (usually one or two dimensional points). In order to minimise the criterion quantitative value, a steepest-descent method is used having the criterion as the objective function and the coordinates of the spatial configuration points as the independent variables.

2.2.1 RESULTS OF SHEPARD'S METHOD

Shepards method was tested on two types of data, namely, artificially generated data and naturally occurring data. Using given monotonic similarity versus distance relation to generate artificial data, it was possible for Shepard's method to recover the "intrinsic" dimensionality of the data and its monotonic function. Using natural data on the other hand, it was possible to extract the "intrinsic" dimensionality of the data and the form of the monotonic similarity versus distance function. One example of natural data was the one describing facial expressions, Shepard (1962 b). This data was mapped to a two-dimensional space, where the relationship between different expressions were revealed and the form of the monotonic similarity versus distance function was recovered. Another set of natural data consisted of fourteen colours of different hue, Shepard (1962 b). The method attempted to obtain a spatial representation of the **inter-relationships between the colours.** exponential similarity versus distance relation was obtained. The fourteen colours formed a C-shape spatial configuration suggesting one "intrinsic" dimension, Shepard (1962 b) suggested two-dimensions.

2.3 KRUSKAL'S REFINEMENT OF SHEPARD'S METHOD

Kruskal (1964 a) refined the termination criterion of Shepard and replaced it by another criterion called "stress". Kruskal offered a more rigorous quantitative measure for the departure from monotonicity.

If d_{ij} are a monotone sequence of numbers. D_{ij} is the dissimilarity between the corresponding i-th and j-th objects under consideration. Then Kruskal defines the criterion function, "stress", that measures the "goodness of fit" between the distances of the points in the resultant spatial configuration on one hand and the corresponding dissimilarities between the objects on the other hand.

The first step in constructing Kruskal's "stress" function is to define the "raw stress", Kruskal (1964):

$$\text{raw stress} = s^* = \Sigma(D_{ij} - d_{ij})^2 \tag{2.2}$$

The raw stress is invariant to transformations such as translation, rotation and reflection of the spatial configuration of the points. However, it is not invariant to any uniform stretching and shrinking of the configuration. If the "raw stress" is divided by the scaling expression:

$$T^* = \Sigma D_{ij}^2 \tag{2.3}$$

then the following expression becomes invariant against shrinking and stretching transformations:

$$\frac{S^*}{T^*} = \frac{\Sigma(D_{ij} - d_{ij})^2}{\Sigma D_{ij}^2} \tag{2.4}$$

and finally Kruskal (1964 a) defines the "stress" s by:

$$s = \left(\frac{\Sigma(D_{ij} - d_{ij})^2}{\Sigma D_{ij}^2}\right)^{\frac{1}{2}} \tag{2.5}$$

The method seeks to obtain a spatial configuration of points that have the set of their distances minimises the "stress" function above. The method employs the same minimisation technique used by Shepard (1962 a).

2.4 COMMENTS ON MULTIDIMENSIONAL SCALING

It seems that multidimensional scaling to one-dimension is the solution for the problem of dimensionality posed by Shepard (1974).

It was originally thought, Kruskal (1964), that dimensionality versus stress relation exhibits an "elbow" at a dimensionality between 1 and m - 1, where m is the number of objects in the data set. This dimensionality is assumed to be the "intrinsic" dimensionality.

However, there is evidence that the "intrinsic" dimensionality and the "elbow" indicating it do not really exist. Firstly, the Monte Carlo experiments, Stenson (1968), Stenson and Knoll (1969) and Klahr (1969), on the relation between stress and dimensionality in multidimensional scaling have shown that this relation does not exhibit the "elbow" reported by Kruskal (1964). Secondly, Shepard (1974) noticed that many two-dimensional spatial configurations were "disguised" by shapes like the C and S.

2.5 SAMMON

The non-linear mapping technique of Sammon (1969) maps a set of points in a metric n_1-dimensional space into another set of points in a lower metric n_2-dimensional space. The two sets are on a one-to-one correspondence, that is each point of either set is paired with exactly one point of the other set. In effect the technique transforms multivariate data into a spatially represented data. Such a transformation results in reducing the dimensionality of the multivariate data.

The essential matter about non-linear mapping is to transform the multivariate data into a spatial configuration normally in two-dimensional space, in order "to detect and identify 'structure'" which may be present", in the data, Sammon (1969). Generally the coordinates of each point in the n_1-dimensional space are non-linearly related to the corresponding point in the n_2-dimensional space. The relation is implicit.

Mapping in Sammon's technique is done such that the distances between the points in the n_2-space are as similar as possible to those corresponding to them in the n_1-dimensional space.

Sammon's technique employes a mathematical criterion that measures the difference between the distances in the n_2-space and the corresponding distances in the n_1-dimensional space. The criterion is in the form of an error that sums all the individual errors between the distances in the two n_2 and n_1 spaces. A zero error means that every distance in the n_2-space is equal to its corresponding distance in the n_1-space. The error is a function of the distances in the n_2-space, that is a function of the coordinates of the points in the n_2-dimensional space. The technique proceeds to preserve the structure of points in the n_1-space while mapping them into the n_2-space. This is through the changing of the coordinates in the n_2-space so as to obtain an optimal reflection of the structure of points in the n_1-space. Thus the coordinates are the independent variables of the error function. The number of such variables is $m \times n_2$, where m is the number of points.

2.5.1 SAMMON'S ERROR FUNCTION

In Sammon's method the error function took the form:

$$E = \sum_{i<j} \frac{(D_{ij} - d_{ij})^2}{D_{ij}} / \sum_{i<j} D_{ij} \qquad (2.6)$$

where D_{ij} is the distance between the i-th and j-th points in the n_1-dimensional space; d_{ij} is the distance between the i-th and j-th points in the n_2 dimensional space and i and j are such that $(i = 1, m - 1)$, $(j = i + 1, m)$ and $j>i$.

The denominator of 2.6 is included to normalise the error. The error, therefore, takes the value of unity when all d_{ij}'s are zero. This is the case when, for instance, the starting configuration in the n_2-space is merely m points superimposed each on the other.

The error function is invariant to transformations such as rotation of axes, reflection in the axes and translation of axes. Generally the value of the error increases as the number of points increases and as the difference between the dimensionality of the higher and lower spaces increases.

2.5.2 DISTANCES IN THE n_1 AND n_2-DIMENSIONAL SPACES

Generally, distances in the n_1-dimensional space are evaluated according to the Minkowski distance function:

$$D_{ij} = (\Sigma|q_{ik} - q_{jk}|^r)^{1/r} \qquad (2.7)$$

Sammon (1969) used the Euclidean distance function, that is $r = 2$, but he also accepted the possibility of using other distance measures. White (1972) reported the use ofadistance measure with $r = 1$.

Distances in the n_1 and n_2-dimensional spaces must be greater than zero. This implies that there must be no repeated points in the data so as to ensure finite value error function.

The distribution of distance values in the n_1 and n_2 spaces assumes different forms depending on the number of clusters and the distribution of points in the clusters. In the case of two clusters that are sufficiently separate, two sets of distances are identifiable. The distances between the points of the same cluster and the distances between the points belonging to different clusters. The first set of distances consists of the smaller distances and the second set consists of the larger distances. It is possible to set Sammon's error function so as to enhance the discrimination between the clusters by preserving the smaller distances on the expense of the larger distances, Kowalski et al (1973).

2.5.3 COMPUTATIONAL ASPECT

The structure of the non-linear mapping program consists mainly of two parts, the error function procedure together with its first and second derivatives and the steepest-descent minimisation procedure.

The program starts by reading the m x n, multivariate data matrix and the m x n_2 matrix for the solution estimate. Then all the $m(m - 1)/2$ distances between the m points in the n_1-dimensional space are calculated and stored. Next the steepest-descent minimisation procedure generates a sequence of coordinates each set of them forms a configuration of m points in the n_2-dimensional space. The sequence of generated coordinates starts from the previously given estimate. Each time a set of coordinates is generated the error function procedure is called and it calculates all the $m(m - 1)/2$ distance between the m points in the n_2-dimensional space. The error function compares the distances in both spaces and measures the difference which is the error. The steepest-descent iteration stops when one of the following three criteria is met:

1. a minimum value for the error function is obtained;
2. a predetermined number of iterations is reached,

or 3. the human observer is satisfied with the resultant n_2-dimensional space configuration of points, Chien (1976).

A. STORAGE

Before minimization begins, Sammon's algorithm calculates and stores the distances between the points in the n_1-dimensional space. Distances storage forms the major memory requirement for the program. The storage size is directly related to the number of distances and approximately directly related to the square number of points in the data set:

$$s_{dis} = c.m(m - 1)/2 \tag{2.8}$$

where c is a constant depending on the computer, s_{dis} is the storage, m is the number of points in the data set and $m(m - 1)/2$ is the number of distances.

In order to avoid storing the distances White (1972) used a Minkowski distance with $r = 1$. This distance is computationally more efficient in storage and timing. The distance, however, is not particularly better than the Euclidean distance as far as clustering is concerned.

B. TIMING

Sammon's program is relatively expensive as far as execution time is concerned. In an experiment, Gelsema et al (1980) found that a one hundred points data set of six dimensions required 120 seconds per iteration. The execution time T is given by the empirical formula:

$$T \simeq a\, L.m(m - 1)/2 \qquad (2.9)$$

where a is a constant depending on the computer, L is the number of iterations and m is the number of points. It can be seen from 2.9 that the execution time is approximately proportional to the square of the number of points in the data set.

C. STARTING CONFIGURATION

As we have mentioned earlier, the error function minimisation requires an initial estimate for the coordinates of the points in the n_2-dimensional space. The two n_1 and n_2-spaces must not be confused with the search space which has $m \times n_2$ dimensionality. It is possible to start the search from any point in the search space particularly when there are few points in the data set.

Sammon (1969) used three ways to start the minimisation. The first was to randomly generate $m \times n_2$ values for the coordinates of the initial configuration. This method is only practical when there is a small number of points, otherwise the search can be very costly. The second way is called the "maximum variance coordinate plane" where the variance coordinates are taken as the $m \times n_2$ coordinates of the starting point. The third way is the method of the eigenvector projection, Kowalski (1974). In this method the feature-by-feature covariance matrix c is diagonalised

$$cy = by \qquad (2.10)$$

where y is the eigenvector and b is the eigenvalue of the matrix c. The covariance matrix is such that:

$$c_{ij} = \sum_{k=1}^{m} (q_{ik} - \bar{q}_i)(q_{jk} - \bar{q}_j) \qquad (2.11)$$

where q_{ik} is the i-th **object** of the **k-th feature**, $\bar{q}_i$ is the mean of all the i-th feature coordinates, q_{ij} is the element of the covariance matrix c and m is the number of points. If y_1 and y_2 are the largest eigenvalues of matrix c and the corresponding eigenvectors are $\bar{Y}$ and $\bar{Y}_2$, then it is possible to obtain the coordinates of the m points through:

$$U_i = \sum_{k=1}^{n_1} \bar{Y}_{1k}\, q_{ki} \qquad (i = 1, m) \qquad (2.12)$$

$$V_i = \sum_{k=1}^{n_1} \bar{Y}_{2k}\, q_{ki} \qquad (i = 1, m) \qquad (2.13)$$

where U_i and V_i $(i = 1, m)$ are the estimates of the starting point coordinates. The number of the largest eigenvalues taken is equal to n_2. The estimate obtained using the eigenvector method **gives the best initial estimate for the starting values for the non-linear mapping method. The** method, however, becomes computationaly expensive with **a** large number of dimensions in the n_1-space.

2.5.4 RESULTS OF SAMMON'S METHOD

To test and evaluate his method, Sammon used two types of data, namely, artificially generated data and natural data.

Some of the artificial data were,

1. 9-dimensional space points distributed on a line,
2. 4-dimensional space points distributed on the vertices of a simplex and
3. 3-dimensional space points distributed on a helix.

All this data was mapped to two-dimensional space. The results of these mappings reflected the geometry of the figures as if they were projected to the two-dimensional space.

The natural data consisted of, first, the classical Iris flower data which was first used by Fisher (1936). The data composed of three classes of Iris flowers where each flower is described by four measurements relating to four biological features in the flower. The sample size is of 150 points. The data separates into three clusters, where two of them are more similar. Second, a set of data based on "document retrieval by content" using 188 17-dimensional points, each point represents a document. The data was mapped into two-dimensional space. Overlapping occured between some of the clusters.

2.6 NIEMANN AND WEISS DEVELOPMENT

Niemann et al. (1979) made two developments to Sammon's non-linear mapping technique. The first was the replacement of the "magic factor" in the steepest-descent minimisation by an optimal step size and the square of the Euclidean distance instead of the distance itself.

2.6.1 OPTIMAL STEP SIZE

Error function minimisation in the search space is carried out either by an empirically predetermined and constant step size value, Sammon (1969) or by a variable optimal step size that minimises the error function in each step towards the solution.

Under certain conditions, Niemann et al. (1979) the optimal step size is important in, first, convergence is to the solution is assured and second the error function decreases monotonically. Consequently the optimal path in the search space from the estimate point to the solution point is obtained . However, the step size analytical formulation is not simple and owing to the lack of a general error function form the derivation of its formula has to be repeated every time the error function is changed.

The step size analytical formulation starts by defining a step size function g which represents the functional dependence of the error function on the step size b, Niemann et al (1979). The final form of this function is a polynomial of degree four:

$$g_{L+1}(b) = k_4b^4 + k_3b^3 + k_2b^2 + k_1 b + k_0 \qquad (2.14)$$

The polynomial is then differentiated and the resultant polynomial of degree three is set to zero:

$$g_{L+1}(b) = 4k_4b^3 + 3k_3b^2 + 2k_2b + k_1 = 0 \qquad (2.15)$$

The above equation is then solved for its roots. The root that gives g its minimum value is taken to be the value of step size in the next L + 1 iteration. The k's are coefficients and their formulae are given by Niemann et al (1979).

Two possible search-direction techniques can be employed with the optimal step size, namely, the steepest-descent and the coordinate-descent. The latter has been found to have computational advantage over the former for it is simpler to compute k_i, $(i = 1, 4)$.

Non-linear mapping with optimal step size has the algorithm, Niemann (1981) as follows:

Step 1: a starting configuration for m coordinates in the lower dimensional space is randomly generated or by some other means, eg, the eigenvector projection;

Step 2: The coefficients k_i $(i = 1, 4)$ are computed;

Step 3: The root that gives a minimal error value is taken as an optimal step size value;

Step 4: If convergence is achieved then stop, else go to step 2.

2.6.2 NIEMANN'S ERROR FUNCTION

Niemann et al (1979) used the following error function:

$$E = \sum s_{jk}^{-(p+2)} \cdot \sum s_{jk}^{p} (s_{jk} - s'_{jk}) \qquad (2.16)$$

where s_{jk} and s'_{jk} are the square Euclidean distances in the n_1 and n_2-dimensional spaces respectively, and p is an integer that can be set such that local distances are preserved. Sammon's error function is a special case of the above function when p is set to minus one.

In contrast to Sammon's error function, Niemann used the square of the Euclidean distance, which is far more efficient than the distance itself.

2.6.3 REMAPPING

Niemann et al, 1979, used a set of 200 handwritten characters to test the optimal step size program. Each character was represented by a binary vector with 320 components. Originally the handwritten characters were scanned with 40 x 30 raster points. The sample of the handwritten characters was of 200 points and the following strategy was followed: the first mapping to two-dimensional space resulted in two groups of classes. The first (group 1) contained the classes of the numerals 1, 4, 7 and 9 and the second group (group 2) contained the numerals 0, 2, 3, 5, 6 and 8. All points in group 1 were linearly separable from the points of group 2. So the points in group 1 were isolated and mapped into another two-dimensional space to form yet another two linearly separable groups: 1.1 and 1.2. From the two-dimensional mapping it was clear that group 1.1 contained the two numerals 1 and 4. Group 1.2 on the other hand contained the two numerals 9 and 7. Group 2 was also mapped to form two new groups. The process was continued until eventually all classes of numerals had been separated. The method of remapping has shown that it is not necessary to separate all clusters in the first mapping since successive remapping may eventually separate all clusters.

2.7 THE FRAME METHOD

Chang et al (1973) considered the difficulty of dealing with a large amount of data. Their method divided a set of m points into two groups. The first group of points were mapped according to the method of Sammon, and the result was called a "frame". The second group was then mapped to the same plane containing the points of the first group taking into account the distances between points in the first and second group **by considering only the distances between the points in the first and second group and by neglecting the inter-point distances in the second group of points. Depending upon the partition of the** data set, substantial memory savings can be achieved.

The method was applied to two sets of data. The first set consisted of a collection of 40 handwritten characters. The set consisted of the Iris data. In both cases, separation of the clusters was achieved.

2.8 CONCLUSION ON NON-LINEAR MAPPING

Different aspects of Sammon's and related methods have been considered. Although the method is powerful in analysing and detecting data structures, it suffers from three important drawbacks:

First : the technique cannot efficiently classify new unknown patterns;

Second: the technique is not able to provide useful information about the relative effects of the different features on classification;

Third : the technique requires a vast amount of computer storage.

In contrast to the discriminant function in pattern recognition Sammon's technique does not provide for fast pattern recognition. In pattern recognition it is possible to obtain a linear discriminant function to classify patterns with little computation cost. In order to classify unknown patterns, Sammon's technique requires the inclusion of the unknown pattern in the original data set and remapping the data set.

The non-linear mapping technique is not able to give an indication of the importance of each feature in classification. The method, in other words, does not include feature selection or extraction. Pattern recognition and cluster analysis methods use feature selection to extract the most important features only, thereby, improving the computational efficiency by reducing the amount of processed data.

Finally, it has been reported by Sammon (1969) that a data set of 250 patterns required a computer storage of 128 k. Furthermore, the increase is approximately proportional to the square of the number of points. Thus Sammon's method is found to be computationaly expensive by White (1972), Chang et al (1973), Schaeter (1978) and Pykett (1978).

Thus Sammon's non-linear mapping method contains a number of deficiencies. In the following chapters we describe techniques which overcome the deficiencies described above.

CHAPTER THREE

TRANSFORMATION FUNCTION

3.1 INTRODUCTION

In this chapter we wish to introduce a mapping technique which incorporates a feature we have called the "Transformation Function". The theory of Transformation Function is given in the first part of the chapter, and its application is described in the second part of the chapter, where experiments on natural and artificial data are given together with an assessment of the results. The chapter concludes within an overall assessment of the mapping technique.

3.2 THEORY

The transformation function X_i is defined as:

$$X_i = X_i(q_1, \ldots, q_{n1}) \ (i = 1, n_2) \tag{3.1}$$

where $q_{ij}(j = 1, n_1)$ are the independent variables of the function $X_i(i = 1, n_2)$. The terms n_1 and n_2 are the sizes of the higher and lower dimensional spaces, and are such that $n_1 > n_2$. Alternatively, $q_j(j = 1, n_1)$ are the coordinates of the point P_{n1} in the higher space (the n1-space) of the data set, and $X_i(i = 1, n_2)$ are the coordinates of the corresponding point P_{n2} in the lower space (the n2-space). If there are m points in the n1-space, that is we have a $(m \times n_1)$ matrix of data then the n2 transformation functions can furnish us with an output $(m \times n_2)$ matrix data.

The set of n2 transformation functions above, defines a transformation or mapping which establishes a correspondance between points in the n1 and n2 spaces. The transformation is defined in terms of a one-to-one correspondance between the two sets of points in the n1 and n2-spaces. This is based on the assumption that $X_i (i = 1, n_2)$ are continuously differentiable.

The transformation functions can take different forms, e.g., a polynomial where the linear form is of special interest. The transformation functions can include cross products as independent variables multiplied in a certain manner to form the cross product terms. It is assumed that all transformation functions are of the same order.

In contrast to the non-linear mapping method of Sammon, 1969, transformation functions offer a direct functional link between the two coordinate systems in the n1 and n2 dimensional spaces. In schematic form, Sammon's algorithm can be represented as:

$$\begin{tikzcd}
q_j (j = 1, n1) \arrow[r, "\text{distance}", "\text{transform}"'] & D_k (K = 1, m(m - 1)/2) \arrow[d] \\
 & E(D, d) \\
X_i (i = 1, n2) \arrow[r, "\text{distance}", "\text{transform}"'] & d_k (K = 1, m(m - 1)/2) \arrow[u]
\end{tikzcd}$$

where m is the number of points, E is the error function employed by Sammon, 1969, D_k is the k-th distance in the n1-space, d_k is the k-th distance in the n2-space and the term $m(m - 1)/2$ is the number of distances between the m points.

The coordinates of the points P_{n1} and P_{n2} in the n1 and n2-spaces are $q(j = 1, n_1)$ and $Xi(i = 1, n2)$ respectively. The distance transform ie, the distance measure is generally defined as:

$$D_{Pq} = \left(\sum_{j=1}^{n1} |q_{Pj} - q_{qj}|^r\right)^{1/r} \tag{3.2}$$

Where D_{Pq} is the distance in the n1-space between points P and q, and r is a non-negative integer usually taken as equal to 2. The distance d_{Pq} between the points P and q in the n2-space is evaluated through the Euclidean distance measure.

With the introduction of the transformation functions in Sammon's mapping scheme we have the following new scheme:

$$\begin{tikzcd}
q_j(j = 1, n1) \arrow[r, "\text{distance}", "\text{transform}"'] \arrow[d, "\text{Transformation Functions}"'] & D_k(K = 1, m(m-1)/2) \arrow[d] \\
 & E(D,d) \\
X_i(i = 1, n2) \arrow[r, "\text{distance}", "\text{transform}"'] & d_k(K = 1, m(m-1)/2) \arrow[u]
\end{tikzcd}$$

3.3 MATHEMATICAL ASPECTS

As we have mentioned earlier, the transformation function can take the linear form in the polynomial expression. This form has the advantage of being very simple to compute. The linear form is mathematically easy to handle. In addition the use of the linear form means that we are pursuing a linear mapping. In this chapter we restrict all of our interest to linear transformation functions.

We start by giving the mathematical expression of the linear transformation functions.

$$X_i = \Sigma \quad a_{ij}\, q_j (i = 1, n2)\ (j = 1, n1) \tag{3.3}$$

the a's are the coefficients of the transformation functions. Alternatively and in the matrix form the functions can be written:

$X^T = AQ^T$ (3.4)

where A is the matrix of the coefficients and has the size (n2 x n1). Both matrices X and Q are row matrices, having the sizes $(1 \times n_2)$ and $(1 \times n_1)$ respectively. If mapping is to be made to a two-dimensional space, then n_2 is set to the value 2. Normally the n_2-dimensional space is chosen such that it is possible to give a visual representation to the data, that is the n_2-space is one, two or three-dimensional. Mapping in general has as its goal the visual representation of data.

3.3.1 TRANSFORMATION FUNCTIONS AS PLANES IN THE N1-DIMENSIONAL SPACE

In a general sense, transformation functions can be viewed as hyperplanes (or plane in the 2-dimensional space), a situation which can be achieved if we set function 3-3 equals to a constant C:

$X_i = \Sigma a_{ij} q_j = C \; (i = 1, n2) \; (j = 1, n1)$ (3.5)

In the case where C = 0, equation 3-5 becomes the equation of the hyperplane passing through the origin of the n1-space. With respect to the hyperplane 3-5 the whole n1-dimensional space is divided into two parts: the region where the inquality

$\Sigma a_{ij} q_j - C > 0$ (3.6)

holds and the region where we have:

$\Sigma a_{ij} q_j - C \leqslant 0$ (3.7)

These regions are called half spaces.

Each point P in the n1-space has a location relative to the hyperplane. We can determine whether this point is on the positive or negative side of the plane, or it lies in the plane. The distance between any point P in the nl-space and the hyperplane X_i is given by 3-8:

$$\text{distance} = \frac{\Sigma a_{ij}\, q_j - C}{(\Sigma a_{ij}2)^{\frac{1}{2}}} \qquad (3.8)$$

The hyperplane can take two positions relative to the positions of the clusters in the nl-space. The hyperplane either separates the clusters or they lie on one side of it. In the first case, the sign of the distance inidicates to what cluster P belongs.

In the second case there can be one sign for the distance value, and the resultant distances of all the m points in the data set have to be normalised so as to have one group of them (the first cluster) giving positive distances from the hyperplane, and the other group (the second cluster) giving negative distances. The distances are in fact the coordinates resulting from using one transformation function, ie, one hyperplane. If mapping is to be made to a 2-dimensional space, then we have to use another hyperplane, ie, another transformation function. Here each transformation function supplies one set of coordinates (or distances from hyperplane) to form, and in the case of mapping to 2-dimensional space, the xy-plane plot. In other words the first hyperplane furnishes us with m distances forming, say, the x-axis coordinates and the second hyperplane gives the y-axis coordinates, **see figure 3.7.**

In addition, each hyperplane offers a different angle of "viewing" the clusters. If the clusters exhibit different distributions of points in the different dimensions, then the use of more than one transformation function may reflect the different distributions of points in different directions. Naturally, not every hyperplane "views" the clusters in the optimal way. It is possible that, using two transformation functions, the hyperplane corresponding to the first function "views" the clusters from an optimal direction and the second in contrast, offers no useful contribution to the separability of the clusters. We speak of "optimal direction" meaning the direction of "viewing" which gives maximum separability of the clusters in the n2-space, and secondly, a distribution of points in the n2-space that reflects the distribution in the n1-space. Furthermore, we speak of optimal hyperplanes, because there can be more than one set of them.

We have put a constant term C in equation 3-5. The effect of this constant is nil as far as mapping is concerned. We have included the constant for the sake of generality. To clarify the effect of the constant C on the resultant coordinates of the n2-space we imagine that every coordinate has been increased by C. The result is only a shift of the total structure of points in the n2-space which does not have any effect on the distances between the points.

In addition, the denominator in 3-8 being a constant can be taken as equal to unity without affecting the coordinates in the n2-space because of the linear nature of this effect.

3.3.2 TRANSFORMATION FUNCTIONS AS VECTORS

Each transformation function has been viewed as a hyperplane. They can also be considered as vectors perpendicular to their corresponding hyperplanes. One implication of this is that the transformation functions are orthogonal to each other, that is the angle between any two corresponding vectors or, alternatively, between the corresponding two hyperplanes, is equal to $\pi/2$. In this case the dot product is equal to zero. Naturally if the angle between the two vectors is zero then they are identical and only one vector can be used.

Two vectors/transformation functions can be made orthogonal to each other by the adjustment of the coefficients so that the dot product of the two vectors is equal to zero. If A_1, and A_2 are two orthogonal vectors in the n1-space then:

$$A_1 . A_2 = 0 \tag{3.9}$$

that is $$\sum_{j=1}^{n1} a_{1j}\, a_{2j} = 0 \tag{3.10}$$

$$\sum_{j=1}^{n1-1} a_{1j}\, a_{2j} + a_{1n1}\, a_{2n1} = 0 \tag{3.11}$$

then
$$a_{2n1} = -\frac{\sum_{j=1}^{n1} a_{1j}\, a_{2j}}{a_{1n1}} \tag{3.12}$$

in this manner and in mapping 2-dimensional space, the number of coefficients is reduced by one to become $2n_1-1$.

In vector form, the distance between the hyperplane A and the point P having the coordinate vector Q is given by:

$$A.Q/|A| \qquad (3.13)$$

where the denominator is the length or the Euclidean norm of vector A. In(3-13), the factor $A/|A|$ is the unit vector of A. In order to make an orthogonal system of transformation functions an orthonormal one, we divide each vector (transformation function) by its Euclidean norm.

Another implication of the vector representation of the transformation functions is that the vector is in fact pointing towards the same direction of the axis that lines the centres of the two clusters. This happens when the corresponding hyperplane takes an optimal direction and gives maximum separability of clusters. In addition and in special cases we can have the vector pointing to the centres of the two clusters, that is the vector passes through the two clusters.

3.3.3 REDUCING THE NUMBER OF TRANSFORMATION FUNCTIONS

In certain cases the resultant mapping of points in the n_1 to the n_2-space are two identifiable clusters that are reasonably separated. In such a case the use of more than n_2 transformation functions would not seem to be justified. Further, it can be shown that in the case of a two-class structure, all n_2 transformation functions are redundant except one which is responsible for the cluster separation. Normally the transformation functions are transformed so as to have only one left. The transformation here is linear. One advantage of transformation functions is their readiness to be reduced in number thereby reducing the dimensionality to a still lower level.

The reduction in the number of transformation functions can only be carriedout after mapping and when mapping shows a reasonable degree of cluster separation. Reducing the number of transformation functions eliminates the need to re-map the data to still lower level. Normally reduction is carried from 3 or 2-dimensional space to the one-dimensional space.

There are two ways by which dimensionality reduction is achieved, the first is reduction from the n_1-space directly to the one-dimensional space, and the second way is to reduce the n_2-space to the one-dimensional space. Both ways must be carriedout after mapping the n_1-space to the n_2-space. The first way is carried through the following function:

$$G_{n1}(Q) = \frac{2(A(Q - Q_o)).(A(Q_{1C} - Q_{2C}))}{|(A(Q_{1C} - Q_{2C}))|} \qquad (3.14)$$

where G_{n1} is a one-dimensional space transformation function, Q_{1C}, Q_{2C} are the centroids of cluster 1 and 2 respectively, Q_o is the half distance between Q_{1C} and Q_{2C}, Q is any point in the n_1-dimensional space and A is the matrix of the n_2 transformation functions with a size of $(n_2 \times n_1)$.

If reduction is to be made from the centroids of the two clusters in the n_2-space then the following function must be used:

$$G_{n2}(X) = \frac{2((X - X_o).(X_{1C} - X_{2C}))}{|(X_{1C} - X_{2C})|} \qquad (3.15)$$

where $|(X_{1C} - X_{2C})|$ is the distance between the points X_{1C} and X_{2C}, X is any point in the n_2-space, X_o is the half distance between X_{1C} and X_{2C} and G_{n2} is a one-dimensional space transformation function that relates the n2-space with the 1-space.

The first function G_{n1} requires the use of transformation function matrix A and the second function G_{n2} does not require A because it deals with the already transformed coordinates Q of the n_1-space to the coordinates X of the n_2-space.

3.3.4 TRANSFORMATION FUNCTION AND DISCRIMINANT FUNCTION

As we have shown, it is possible to, either, map the n_1-space to a one-dimensional n_2-space, or to reduce a 3 or 2-dimensional space to a one-dimensional transformation function. And with the help of transformations like rotation, translation or scaling, it is possible to produce a one-dimensional transformation function that ideally has the following properties:

1. it gives a positive value to any point from one of two categories of points in the nl-space, and it gives a negative value to any point from the second category;
2. the points in the first and second categories are normally distributed and G(Q) of their respective means are +1 and -1 respectively. Mathematically if G is the discriminant/transformation function then:

1. $G(Q) = \{ \begin{matrix} \text{positive, if } Q \text{ is a member of category } 1 \\ \text{negative, if } Q \text{ is a member of category } 2 \end{matrix}$

2. $G(Q) = \{ \begin{matrix} +1\text{, if } Q \text{ is the centroid of category } 1 \\ -1\text{, if } Q \text{ is the centroid of category } 2 \end{matrix}$

In contrast to the linear discriminant function in pattern recognition, G(Q) does not require that each member of the data set be of known classification. The linear discriminant function G(Q), which is a linear transformation function at the same time, is of the unsupervised kind.

3.4 COMPUTATIONAL ASPECTS

The program of transformation functions consists of five parts:

1. the main program;
2. the function subroutine;
3. the minimization subroutine;
4. the scaling subroutine, and
5. the plotting subroutine.

The minimization subroutine is either incorporated with the program or is called from a library as is the case here. The scaling and plotting subroutines are called from a subroutine called FINAL.

3.4.1 THE MAIN PROGRAM

The main program is composed of the input part, the normalization part, the generation of the nl-space distances together with the elimination of repeated points and finally the part which calls the minimization subroutine and the "final" subroutine. The main program communicates with the FUNCT 1 subroutine through COMMON variables.

A. THE INPUT PART

The program reads three types of information, the first type consists of the following items:

N, is an INTEGER which represents the size of the higher space. N must be greater than one.

NPARAM is an INTEGER used in linear transformation functions with normally NPARAM = 2*N. It represents the number of coefficients employed by the transformation functions.

M, is an INTEGER which represents the number of points in the data set. M is kept constant throughout the program execution, it only changes when one of the points is found to be repeated and is deleted.

IT, is an INTEGER variable which is initially set to zero and incremented by 1 on each iteration. On exit it gives the number of function evaluations. Each iteration requires N function evaluations.

KIND, is an INTEGER, which contains the number of classes in the total set. In case where the number of classes is not known KIND is set to 1.

MM, is an INTEGER array which contains the number of points in each class. In the case where the number of classes is not known, then the first element of MM is given the value of M and the rest of the array elementsare given the value of zero.

L, is an INTEGER array, which contains the symbols attached to each class. When the number of classes is not known then only one symbol is used.

TIME, is a real variable which contains the time limit allowed for program execution. This is useful when it is expected that convergence will require a much longer time than can be requested on the computer.

XTRAN and YTRAN, are real variables which contain the value of the X and Y axes respectively in millimetres as plotted on the graph paper.

The second reading stage is the one that inputs the (m x n1) matrix (m is the number of points and n1 is the size of the higher space). The data matrix is the array Q which is read in a row-by-row manner. The reading is formatted in a way that is appropriate to the particular data in use.

The third and final reading stage is the one that inputs the initial estimate for the transformation functions.

B THE NORMALIZATION PART

The normalization is the second part of the main program. It loads the first column of the Q array, then calls the scaling subroutine to normalize the column. Finally the normalized column is re-loaded in its original place in array Q after being normalized. The normalization here is to the domain $|0, 1|$.

C GENERATION OF THE DISTANCES IN THE N1-SPACE

Distances of the nl-space are generated in a tertiary nested DO loops. It is required to generate $m(m - 1)/2$ distances (m is the number of points). The generation employs the following distance measure, **which is of a purely empirical basis:**

$$D_{ij} = (\Sigma(q_{ik} - q_{jk})^2)^{1/nl} \tag{3.16}$$

where D_{ij} is the distance between the i-th and the j-th points in the nl-space. And q_{ik} is the coordinate of the i-th point and k-th dimension. The first loop (the i-th loop) starts from $i = 1$ to $i = m - 1$ (wherem is the number of points), the j-th loop starts from $j = i + 1$ to $j = m$, and the k-th loop starts from $k = 1$ to $k = nl$. Each time a distance is evaluated there is a corresponding index to lable it. The relation between i, j and the index INDEX is:

$\text{index} = (I - 1)(2M - I)/2 + j - I$

In this part of the main program, a procedure for eliminating repeated points is also included. The repeated points are the points that have a distance of zero value. When such a distance is detected the program eliminates the second copy of the point overwriting it with a new point taken from the last element of the Q array. The size of the array is then decremented by one. This process is repeated until there are no repeated points left in the data set. In order not to destroy the identity of each point, a number is assigned to each one of them.

At the end of the main program, two CALL statements are used to call the minimization subroutine and the FINAL subroutine.

3.4.2 THE FUNCTION SUBROUTINE

This subroutine is in two parts. The first one is responsible for the evaluation of all (m x 2) coordinates of the two dimensional space ($n_2 = 2$). The second part is the error function evaluation. In the first part two nested DO loops are employed. The i-th loop starts from $i = 1$ to $i = m$ (m is the number of points) and the j-th loop starts from $j = 1$ to $j = n_1$. The j-th loop is a summation mechanism. There are two such summations, the first summation evaluates the x-coordinates and the second the y-coordinates.

The other part of the function subroutine is the one responsible for evaluating the error function. This part is of two nested loops. The outer loop has i taking values from $i = 1$ to $i = m - 1$ and the inner loop has j taking values from $j = i + 1$ to $j = m$. This means that statements in the inner loop are used for calculating the m(m - 1)/2 distances of the n2-dimensional space and comparing them with the already computed and stored distances of the n1-dimensional space. The distances of the n2-dimensional space are computed using the (m x n2) coordinates generated in the first part of the function subroutine. The two distances are related mathematically in the following way:

$$E = \Sigma \, (D_{ij}^2 - d_{ij}^2)^2/D_{ij}^2 \qquad (i<j) \tag{3.17}$$

where D_{ij} is the distance in the n1-space between the i-th and j-th points, and d_{ij} is the distance in the n2-space between the i-th and j-th points. For computational efficiency, the above function is written in such a way that as d_{ij} is calculated it is subtracted directly from D_{ij} thus removing the need to store the value and access it later.

$$\sum_{i<j} (D_{ij}^2 - (X_i - X_j)^2 - (y_i - y_j)^2)/D_{ij}^2 \tag{3.18}$$

This is done purely to increase computational efficiency.

Also for computational efficiency the distances in the n1-space are accessed according to an index in exactly the same way as they have been stored.

In this part of the function subroutine, programming optimizations play the most important role in decrementing execution time. This is because of the above mentioned two nested DO loops having a frequency of m(m - 1)/2 per function calling. In addition this frequency is proportional to the square of the number of points.

Furthermore because of the frequency with which the inner loop is executed any inefficiencies here will seriously affect the overall computational efficiency of the program. One important step to prevent this inefficiency is to eliminate the square root operation. This has been done with Sammon's technique by Niemann, et al, 1979.

Theoretically, the following two error functions are identical in that they both lead to the same solution but G_{ij} is more efficient for not requiring the use of the square root operation:

$$F_{ij} = \sum_{i<j} (D_{ij} - d_{ij})^2/D_{ij} \tag{3.19}$$

$$G_{ij} = \sum_{i<j} (D_{ij}^2 - d_{ij}^2)^2/D_{ij}^2 \tag{3.20}$$

notice if $D_{ij} = d_{ij}$ for all i less than j, then both expressions give the same minimal value of zero.

Before the function subroutine reaches the end, two subroutines are called. The first subroutine is called to find how much time has elapsed since the start of the execution. This subroutine called from a special system library. If the time that has elapsed is less than a predefined limit then control is RETURNED to the minimization subroutine to continue the process of minimizing the error funciton. If the time elapsed is greater than the given limit, then the second subroutine is called, that is, the FINAL subroutine.

3.4.3 THE MINIMIZATION TECHNIQUES

The prime target of the techniques is to minimize the error function which is composed of a number of independent variables. In contrast to Sammon's technique, the number of independent variables is $n_1 \times n_2$, where n_1 is the size of the higher space and n_2 is the size of the lower space, usually $n_2 = 2$. The error function has the property of being positive all over the intervals of variation in the $n_1 \times n_2$-dimensional space which is also called the search space, (this space must not be confused with the n1 and n2 dimensional spaces, ie, the higher and lower mapping spaces). Usually, and because of the complex nature of the error function, the search space consists of many minima, a global minimum and many local minima. The reason behind this complexity is probably due to the many ways in which it is possible to map a structure of points in the n1-space to the n2-space. It is the goal of the minimization technique to attain the global minimum, that is to achieve the lowest value for the error function. The global minimum is difficult to reach. Furthermore, is complicated by the fact that "virtually all numerical methods for unconstrained minimization are designed to obtain estimates of local minimizers...." over the search space, Wolfe, pp 22 1978. The search for a minimum starts from some chosen point in the space. One way of starting the minimization for example is to choose a point at random. In practice it may require more than one trial in order to reach the global minimum. For reasons of symmetry, the choice of the n1 x n2-search space as a starting point is appropriate for transformation functions. This means that all n1 x n2 coefficients of the transformation functions are chosen to be zero's at the start of the minimization process. The other implication is that, all coordinates in the n2-space are initially set equal to zero, ie, all m points in this space are superimposed on each other before iteration starts. With the progress of the mapping process, the m points start to move to the different directions reflecting in a gradual manner the structure in the n1-space and forming the would be n2-space mapping. When such a start is used ie, from the origin of the n1 x n2-search space, it is possible then to compare the different mapping results obtained from different applications.

An alternative way for furnishing estimates for the coefficients of the transformation functions, is to apply the eigenvector projection technique. The eigenvectors of the two highest (if n2 = 2) eigenvalues are taken, the two vectors supply us with 2*n1 values of the estimates for the transformation functions coefficients. It is useful to note that the eigenvector projection is a kind of linear transformation function. The use of the eigenvector projection technique in providing estimates for the coefficients of the transformation functions could be very useful in directing the minimization process to a global minimum and possibly also minimizing the execution time.

The other important aspect of minimization here is the minimization of the error function without using its first or second derivatives. Apart from simplicity and convenience, the use of a minimization subroutine without derivatives eliminates the necessity of changing the form of the derivative subroutine each time the transformation function is changed or the error function form is changed. On the other hand the use of minimization with derivatives is computationally more efficient.

Three different techniques of minimization have been considered to minimize the error function of n1 x n2 variables. The techniques used were the Quasi-Newton method of Gill et al, 1976, the direct search minimization as described by Rosenbrock, 1960 and the simplex method of Nelder et al, 1965. The three techniques do not require first or second derivatives.

A THE QUASI-NEWTON METHOD

The method is available as a routine in the NAG (Numerical Algorithm Group) library (NAGFLIB: 1414/0: Mk5: Mar 76). The purpose of this method is to minimize a function $F(\underline{X})$ of N independent variables $\underline{X} = (X_1, X_2, \ldots, X_N)^T$. The method is easy to implement and parameters need to be defined. The method was originally intended for users having little knowledge of the behaviour of the function to be minimized. The subroutine is called from the NAG library by a statement of the form:

```
CALL EØ4CEF(N, X, F, W, LW, IFAIL)
```

The subroutine is based on the FORTRAN version UCNDQ1 by Gill et al, 1972. The user must have a subroutine called FUNCT1 to compute the function $F(\underline{X})$ at any point $\underline{X}$, and also the user must supply an initial estimate of the minimum. Then from the given estimate the subroutine generates a sequence of points intended to converge to a minimum of $F(\underline{X})$. The points are generated by using estimates of the gradient and curvature of the objective function. The subroutine attempts to verify that the final point is a minimum. The subroutine EØ4CEF employs a number of parameters to communicate with the function subroutine FUNCT1. The rest of the parameters required by the method are set automatically. It is normally the case that the minimization process is more efficient if the user can choose the parameters to suit the function being minimized. One example of such a parameter is the step size. Below is the list of parameters employed by the EØ4CEF subroutine together with their description.

N, is an INTEGER, which on entry specifies the number of variables, it remains unchanged on exit. N must be greater than 0.

X, is a real array of DIMENSION greater than or equal to N. On entry X takes the estimate of the starting point in the search space. On exit, it contains the value of X corresponding to the final in F.

F, is a real variable, on exit it contains the lowest function value found by the minimization subroutine.

W, is a real array of DIMENSION at least (LW). W is used as working space.

LW, is an INTEGER variable, on entry it specifies a value greater than or equal to 10*N + N*(N - 1)/2, or LW = 11 if N = 1. IFAIL, is an INTEGER variable which must be assigned a value of zero before entry to the procedure. Unless the routine detects an error, IFAIL contains zero on exit. In the case of an error IFAIL takes a value between 1 and 8 depending upon the type of error.

The FUNCT1 subroutine takes the form:

```
SUBROUTINE FUNCT1 (N, XC, FC)
INTEGER N
real XC(N), FC
```

N, is an INTEGER variable which contains the number of variables. Its value must not be changed in FUNCT1.

XC, is a real array of DIMENSION, N. It contains the value of the current point. Its value must not be changed in FUNCT1.

FC, is a real variable. On exit, FC contains the value of the function at the current point XC.

B ROSENBROCK FUNCTION MINIMIZATION

This method minimizes a function of N independent variables for an unconstrained optimization. The routine uses the method for direct search minimization due to Rosenbrock, 1960. The minimum of a function is attained by cyclic searches in parallel to each of the N orthogonal unit vectors, the coordinate directions. Each stage of the iteration process consists of N searches. For the next stage, a new set of orthogonal unit vectors is generated, such that the first vector of this set lies along the direction of greatest advance for the previous stage. The Gram-Schmidt orthogonalization procedure is used to calculate the new unit vectors. The subroutine statement takes the form:

```
SUBROUTINE ROMIN (N, X, F, STEP, MONITOR)
```

where N is the number of independent variables of the function to be minimized, X(N) is an estimate of the solution. The subroutine requires another subroutine FUNCT (N, X, F) for the calculation of the value of F at any point X. STEP is an initial step length for all searches of the first stage. MONITOR is a subroutine for printing intermediate results. The algorithm was coded in FORTRAN and taken from Machura et al, 1973.

C THE SIMPLEX MINIMIZATION METHOD

This method is an iterative one. The estimate supplied by the user is the first vertex of the N + 1 simplex. The subroutine generates the rest of the N vertices. The largest value vertex is reflected in the centre of gravity of the remaining vertices and the function value at this new point compared with the remaining function values. The outcome of this test decides whether the new point is accepted or rejected. A further expansion move may be made, or a contraction may be carried out. When no further progress can be made the sides of the simplex are reduced in length and the method is repeated.

3.4.4 THE SCALING SUBROUTINE

The subroutine is used for two purposes. The first is to normalize the coordinates of the n1-dimensional space to the domain $|0, 1|$. The second purpose is to scale the n2-dimensional space such that it can be plotted when the plotting subroutine is called.

The subroutine takes a one-dimensional array of coordinates, their number and the scaling factor. The subroutine finds the minimum and maximum values in the above mentioned array, and scales the coordinates in a linear manner having the maximum value coordinate equal to zero. If U_i is the i-th element of the array U, and V_i is the i-th element of the array after normalization, then:

$$V_i = S(U_i - U_{min})/(U_{max} - U_{min}) \tag{3.21}$$

where S is the scaling factor, in normalization, S is set to 1. In the case of scaling the values of the coordinates to be plotted, S is set to the length of the U and/or y-axes in millimetres. U_{min} and U_{max} are the minimum and maximum values in the U array. The subroutine finds the two values by a simple sorting method. The subroutine can scale or normalize one array at a time. The subroutine is therefore called n_1 times for normalization and n_2 times for scaling.

As it has been mentioned, the transformation performed on the coordinates by the scaling subroutine is of a linear form, and is given by:

$$V_i = aU_i + b \tag{3.22}$$

where

$$a = (V_{max} - V_{min})/(U_{max} - U_{min}) \tag{3.23}$$

$$b = V_{min} - aU_{min} \tag{3.24}$$

3.4.5 THE PLOTTING SUBROUTINE

This subroutine is employed to reproduce the n2-space (n2 = 2 only) in a graphical form. The plotting is merely an Xy-plane with two axes X and y subdivided into intervals and a number of symbolized points scattered in the plane. The plotted axes are not important as far as separability is concerned. They simply provide a plane of reference for viewing the clusters.

The plotting subroutine is called after calling the scaling subroutine. The 2-dimensional space coordinates have to be scaled to a suitable level convenient for the plotting subroutine.

The plotting subroutine consists of two parts. The first part positions, scales and then draws the axes. The second part plots the points. The subroutine can either plot a single symbol when the points classification is unknown or different symbols corresponding to the different classes when classification is known beforehand. The PLOT subroutine employs the following list of parameters:

XPLOT, YPLOT are real arrays containing M coordinates of the 2-space after being scaled by the scaling subroutine.

KIND, is an INTEGER defines the number of classes of points. If the number is unknown, KIND is set to 1.

MM, is an INTEGER array containing the number of points in each class of points. If the number of classes is unknown, NN(1) is given the value of M.

L, is an INTEGER array containing the codes of the symbols representing the different classes of points.

XTRAN, YTRAN, are real variables define the length in millimetres of the X and y axes respectively.

3.4.6 THE FINAL SUBROUTINE

The subroutine is the last part of the transformation function program. This subroutine can be called from two places in the program. It is either called from the main program when the minimization process has converged before the

the time limit, or it is called from the FUNCT1 subroutine just before the time limit.

The subroutine firstly prints the number of function evaluations, the error function value, the $2n_1$ coefficients of the transformation functions and the coordinates of the n_2-space before and after scaling them. Secondly, the subroutine calls the plotting subroutine to plot the points of the 2-dimensional space.

3.5 THE RESULTS

The results of three data sets are reported here. The sets are:

1. the Iris Data taken from Chen, 1973;
2. the data set prepared from the experiments on Adenorcarcinoma 755 (CA 755) taken from Goldin et al, 1968, and
3. artificially generated data taken from Jurs et al, 1975.

The same program was used throughout the three experiments mentioned above. All parameters were kept the same, except, the number of points and the dimension size of each set, which are of course data-dependent.

3.5.1 IRIS DATA

This data has been used by many researchers to test statistical techniques. The data is composed of 150 samples describing each of three species of Iris flowers. Four measurements were made on each flower. From each species: Setosa, Versicolor and Virginica, the sepal length and width, and petal length and width were taken as four features from each flower.

A INPUT DATA OF IRIS DATA

Number of points is 149

Number of Dimensions is 4

(149 x 4) matrix of the Iris Data

B OUTPUT DATA OF IRIS DATA

Number of function evaluations is 519.

Final Error Function value is 0.127.

Coefficients of the transformation functions matrix are:

$$\left\| \begin{matrix} -0.885 & -0.972 & 0.766 & -1.05 \\ 0.14 & 0.644 & -1.099 & -0.109 \end{matrix} \right\|$$

Execution time is 938 seconds Core is 50 K.

The Error Function used was:

$$\frac{1}{\sum_{i<j} D_{ij}^2} \sum_{i<j} \frac{(D_{ij}^2 - d_{ij}^2)^2}{D_{ij}^2} \qquad (3.25)$$

where D_{ij} is the distance in the n1-dimensional space between points i and j and d_{ij} is the corresponding distance in the n2-space. The distance D_{ij} takes the form:

$$D_{ij} = \left(\sum_{k=1}^{n} |q_{ik} - q_{jk}|^2\right)^{1/n_1} \qquad (3.26)$$

where $n_1 = 4$.

C DISCUSSION

A two dimensional display of the Iris data using the transformation function program is shown in Figure 3.1. The result is very similar to that obtained by Sammon, 1969. In the display, the top cluster is the Iris Setosa flower then comes Iris Versicolor and at the bottom comes Iris Virginica. The latter two clusters are obviously closer to each other. The display also shows that the y-axis transformation function is the discriminating one. the X-axis transformation function, however, shows very little contribution to the discrimination between the classes of points. Consequently, the y-transformation function, can be taken alone as a discriminant function. In addition it seems that we can only discriminate between the first class of flowers, that is the Iris Setosa, on classes of Iris Versicolor and Virginica on the other hand, and we cannot discriminate between the second and third classes of Iris flowers.

3.5.2 ADENOCARCINOMA 755 DATA

The program of the transformation functions has been applied to biologically active drugs, the Adenocarcinoma 755 data, Goldin et al, 1968. The mapping is from $n_1 = 19$ to $n_2 = 2$-space. The final form of the data is a 251 x 19 size matrix.

The data has been based on a list of molecular formulae of biologically active drugs. The formulae were taken from the previously tested drugs by the National Cancer Institute for activity in the solid tumor Adenocarinoma 755 (CA 755) screening system, Goldin et al, 1968. In the test, drugs were administered to small animals with solid tumors, and tumor growth was measured. A parameter called tumor weight inhibition (TWI) defined as a percentage was calculated. A high percentage value indicates that the drug belongs to the carcinogenic group, (a threshold of 70% was used by Kowalski, 1974).

Two hundred and fifty one molecular formulae were taken to form the basis for the data set. From each molecular formulae a set of nineteen features was formed. The features were almost all the features used by Kowalski, 1974. Originally, fifty structural features were extracted from each molecular formula by Kowalski. Fourteen features were eliminated because of their scarcity.

A <u>INPUT DATA OF ADENOCARINOM 755 DATA</u>

Number of points is 251

Number of dimensions is 19

(251 x 19) matrix of the Adenocarinom Data.

B <u>OUTPUT DATA OF ADENOCARINOMA 755 DATA</u>

Number of function evaluations is 2044

Error function value is 0.291

Coefficients of the transformation function matrix transpose are:

-0.401	0.249
0.447	0.007
-0.415	0.629
0.001	0.073
1.228	1.439
-0.514	-1.196
0.217	-0.264
0.541	0.924
0.425	0.495
0.962	-0.235

```
-0.124      0.078
 0.164     -0.111
-0.556     -0.372
 0.889     -0.233
-0.381      0.332
 0.294     -0.42
-0.007     -0.061
-0.212     -0.317
-0.726     -0.705
```

Execution time is 113 minutes

Core 100Kk

The Error Function used: same as in the Iris.

C DISCUSSION

The transformation program was then used to process the new form of the data. The program eliminated 52 identical points from the data set leaving 199 points to be mapped. The details of the run are shown below. The graphical outputs is also shown in Figures 3-2, 3.

In the graphical output, Figure 3-2, three clusters can be identified. The top (cluster 1) cluster contains a dominance of the biolocially inactive compounds, (total membership of cluster 1 is 105 compounds), the bottom cluster (cluster 2) contains a mixture of biologically active and inactive compounds (the ratio is 39/27), and the circular cluster (cluster 3) on the left of the map contains the biologically active compounds, in this cluster one of the 41 compounds has been incorrectly classified.

The category membership of every point in the above data set used was known, and this enabled the success of the technique to be determined. This information was of course not inputed to the program, that is our technique is of the unsupervised form as opposed to Kowalski's (1974) supervised method. The membership of each point was determined by its TWI (tumor weight inhibition) value. When the percentage value of the TWI was greater than or equal to 70%, then the membership was taken as positive. For values less than 50% the membership was taken as negative, values between 49 and 70% were ignored.

It can be shown that two groups of points can be identified in cluster 1, the top right part of cluster 1 is shown magnified in Figure 3.3. The first group contains 95 points and the second contains 10 points. In the first group, 13 errors were counted giving the group a classification error of 14%. However in the second group, only one point out of the 10 points was correctly classified. These 10 points constitute one class of drugs, the Halopurine nonsugar analogs, Goldin et al, 1968. The reason for these incorrect classifications lies in the choice of nineteen features which failed to discriminate this class of compounds from the remainder. In addition, as cluster 1 graphically shows two parts, the first (top right) contains a mixture of the correctly classified together with the incorrectly classified, the second (bottom left) is totally composed of correctly classified compounds. Those compounds are the least carginogenic compounds and they are all having a hydroxyl or amino group attached to C-6 position in the purine nucleus.

In cluster two, there are 66 points, and again two groups can be identified. The first contains 27 points and the second 39 points. In the first group misclassification was 81%. On investigating this group, it was found that the members of this group were drawn from the Uracil class of compounds which has been tested by the National Cancer Institute. In the nineteen features that were chosen to describe the molecules, only one feature was allocated to differentiate between the purine and the pyrimidine derivatives. In the second group of cluster 2, with 39 points, 87% classification was obtained.

In cluster 3, the cluster is shown magnified in Figure 3.4 there are 41 points. On considering the classification, five points out of the 41 were found to be misclassified. Further investigation revealed taht four of the five misclassified points differed from the rest of the 41 points in having the carbon-sulphur-carbon bonding located in the sugar part of the molecule. The rest of the 41 points contain the sulphur atom connected to carbon-6 in the purine part of the molecule. Only one molecule out of the 37 strongly carcinogenic compounds was misclassified. It had been noted by Kowalski, 1974, that the carbon-sulphur bond is a most important feature with regard to carcinogenic activity. Our results show more specifically, that the sulphur atom must be connected to a carbon atom and both linked to carbon-6 in the purine molecule for carcinogenic activity.

In Figure 3.5 three symbols have been used to mark the following classes of molecules:

1. an asterisk to each molecule having sulphur-carbon functional group at the carbon-6 in the purine molecule;
2. a circle to each molecule having sulphur-hydrogen functional group at the carbon-6 position in the purine molecule, and
3. a triangle to the rest of the molecules.

The result is the same mapping as in Figure 2 only the symbols are different, but it shows that the compounds being classified tend to cluster on the basis of being purine-SH or purine-SR on one hand and the remainder on the other side.

Finally, the application of the transformation functions program on the Adenocarcinom 755 data show a very high degree of stability. Through the running of the program some points in the 2-dimensional map showed strange locations relative to the clusters. The validity of the points were checked and errors were found. On correcting the errors the points joined the clusters and neither the transformation functions coefficient nor the coordinates of the whole set of point showed considerable change.

3.5.3 ARTIFICALLY GENERATED DATA

This data was taken from Jurs et al, 1975. The data was generated in such a way that it contains two linearly separable classes of points. Every class is of 50 points, and the data has a dimensionality of five.

Mapping of the data by the transformation function program confirmed the existence of two linearly separable classes of points, Figure 3.6.

Jurs et al, 1975, used the same data but augmenting it with an extra data column to define the membership of each pattern. In our application no information was given to the program about the class of membership of each point.

3.6 CONCLUSION

This study has shown that a direct and linear relationship between the n1-space and n2-space can be practically established. It has also shown that it is possible to obtain a form of discriminant function out of transformation functions. The introduction of transformation functions leads to increased computational efficiency because of the reduction in the number of independent variables of the error function from m x n2 to n1 x n2, where m is the number of points and n1, n2 being the dimensionality of the higher and lower space respectively.

Although it is possible to use non-linear transformation functions between the higher and lower spaces, the linear form proved to be efficient. In addition, the use of direct and linear transformation functions acquired the advantage of discriminant functions known in pattern recognition for their use in classifying new sample points after training a similar set of data. A one-dimensional transformation function is, in a certain sense, a form of discriminant function. Furthermore, the transformation function does not require that a known classified data set is used; since it is an example of an unsupervised classification method.

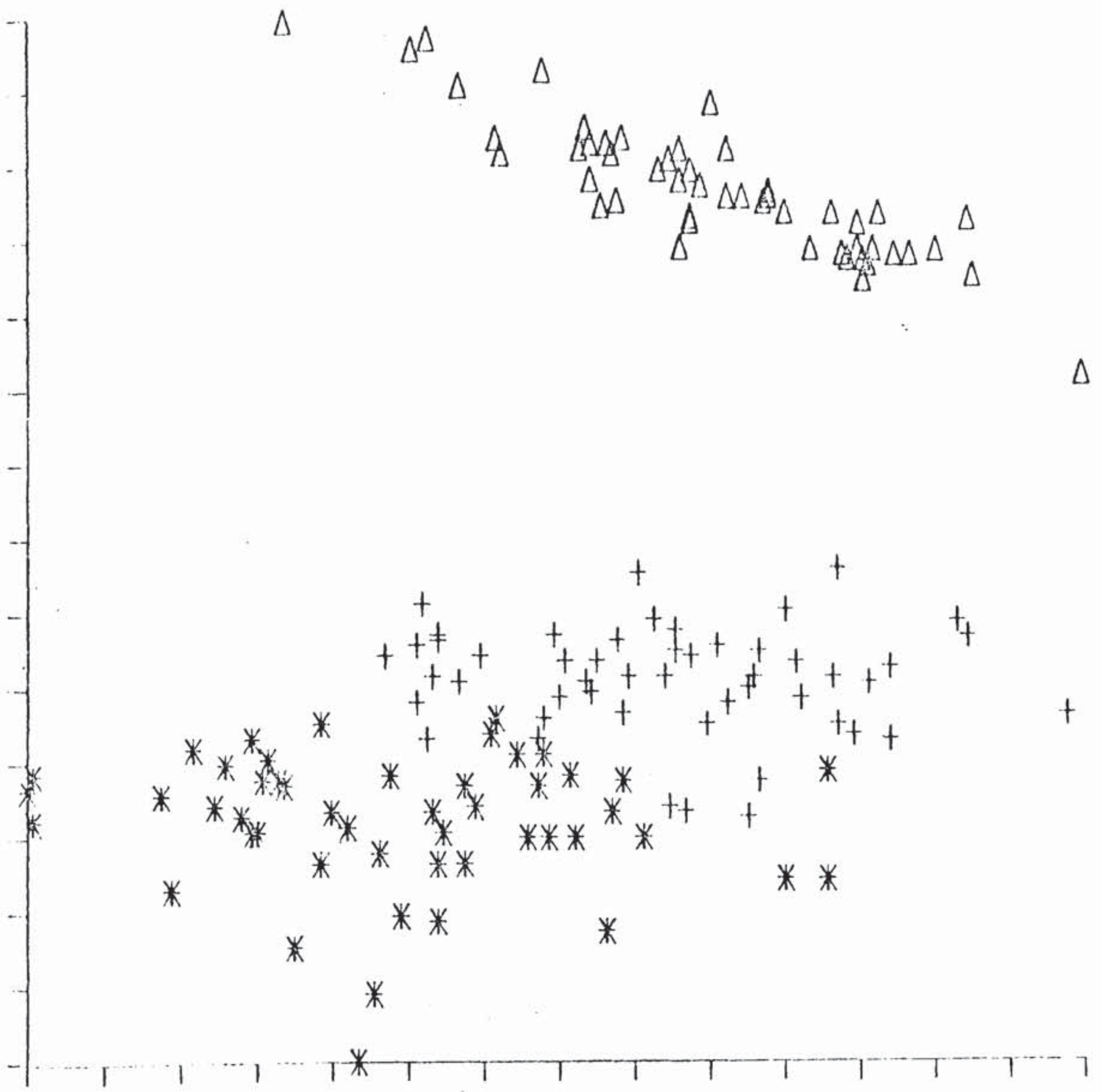

FIGURE 3.1 Iris data 2-dimensional map

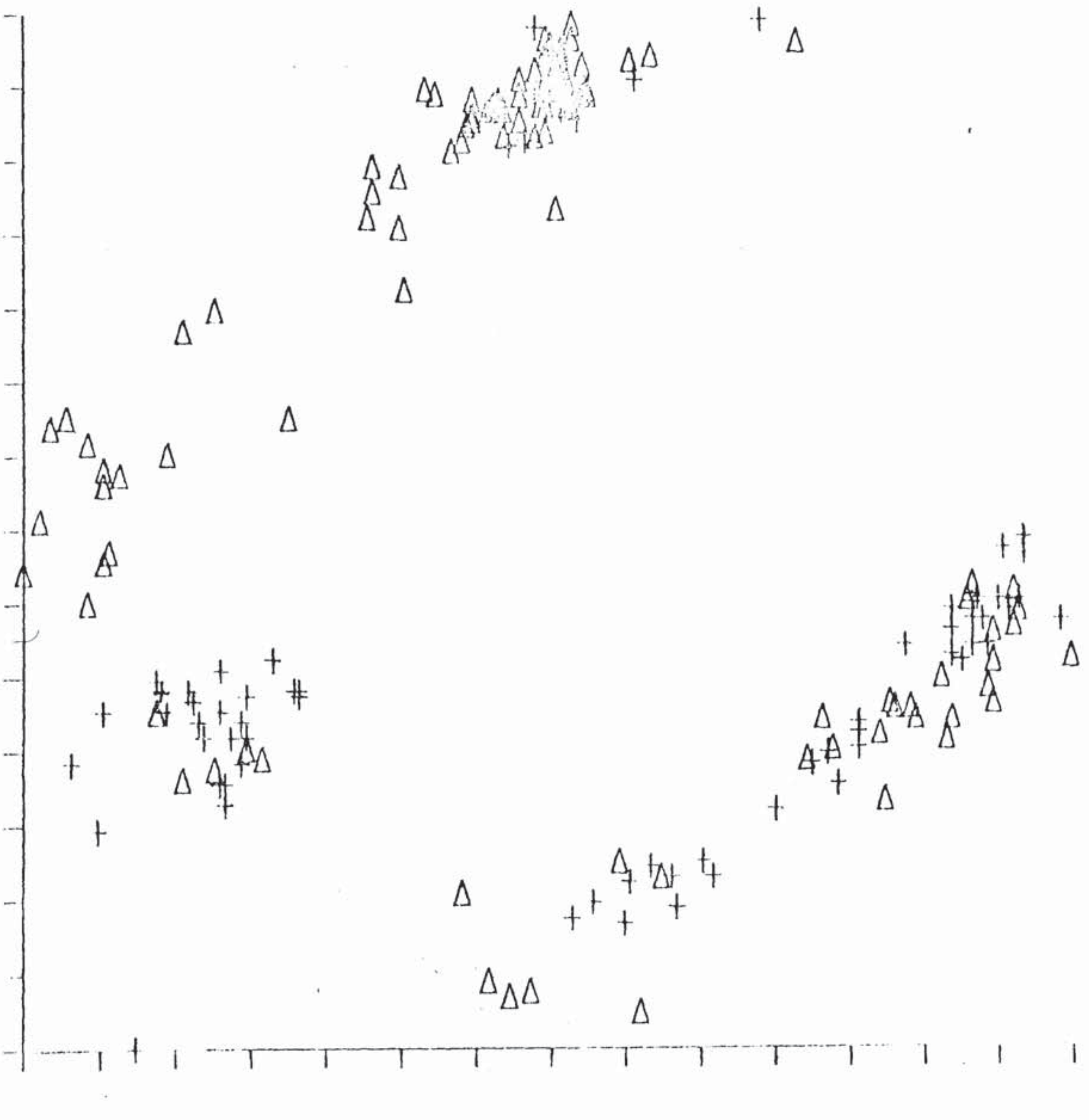

FIGURE 3.2   CA755 data 2-dimensional map

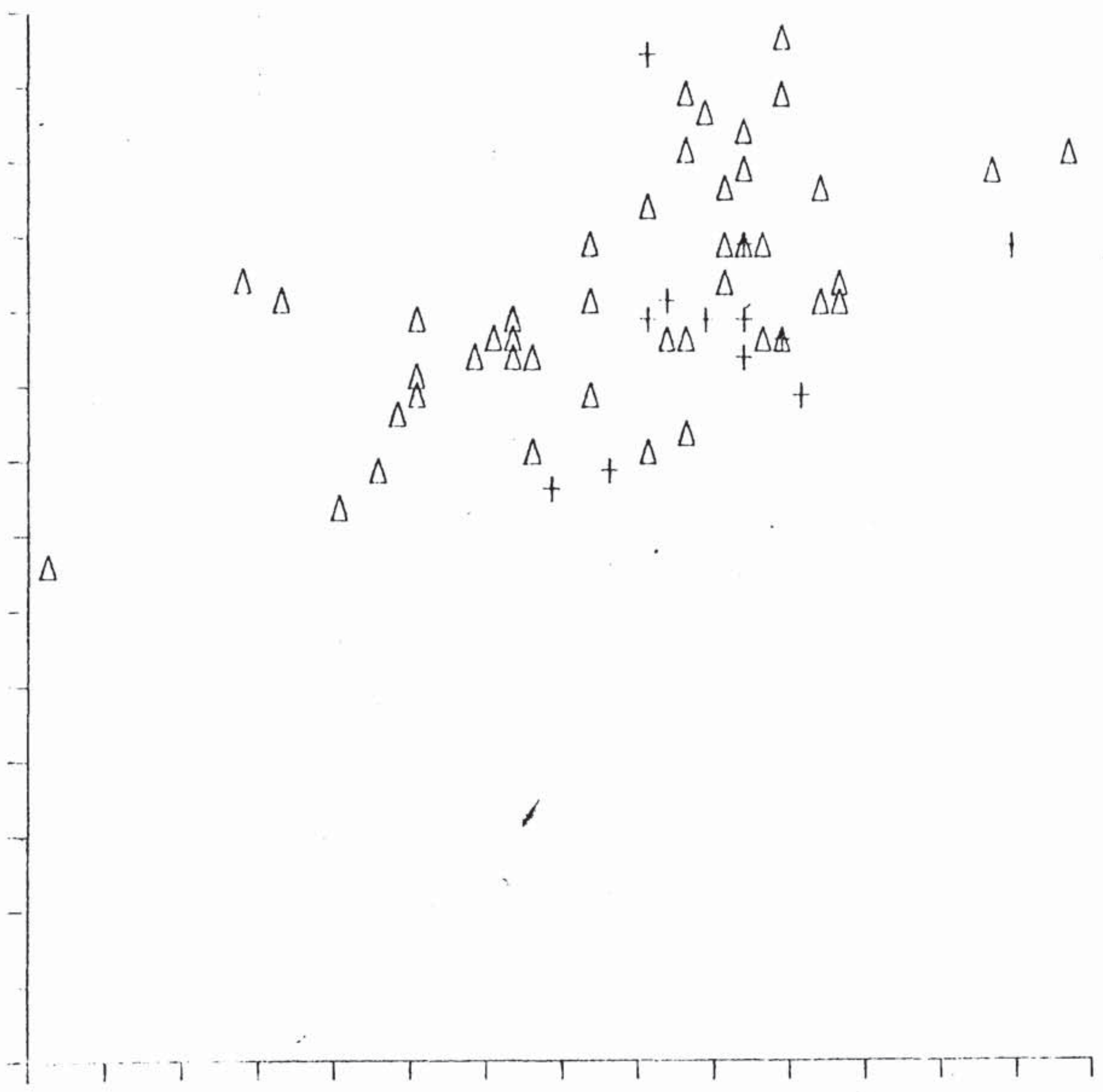

FIGURE 3.3 Magnified map of cluster 1 in Figure 3.2

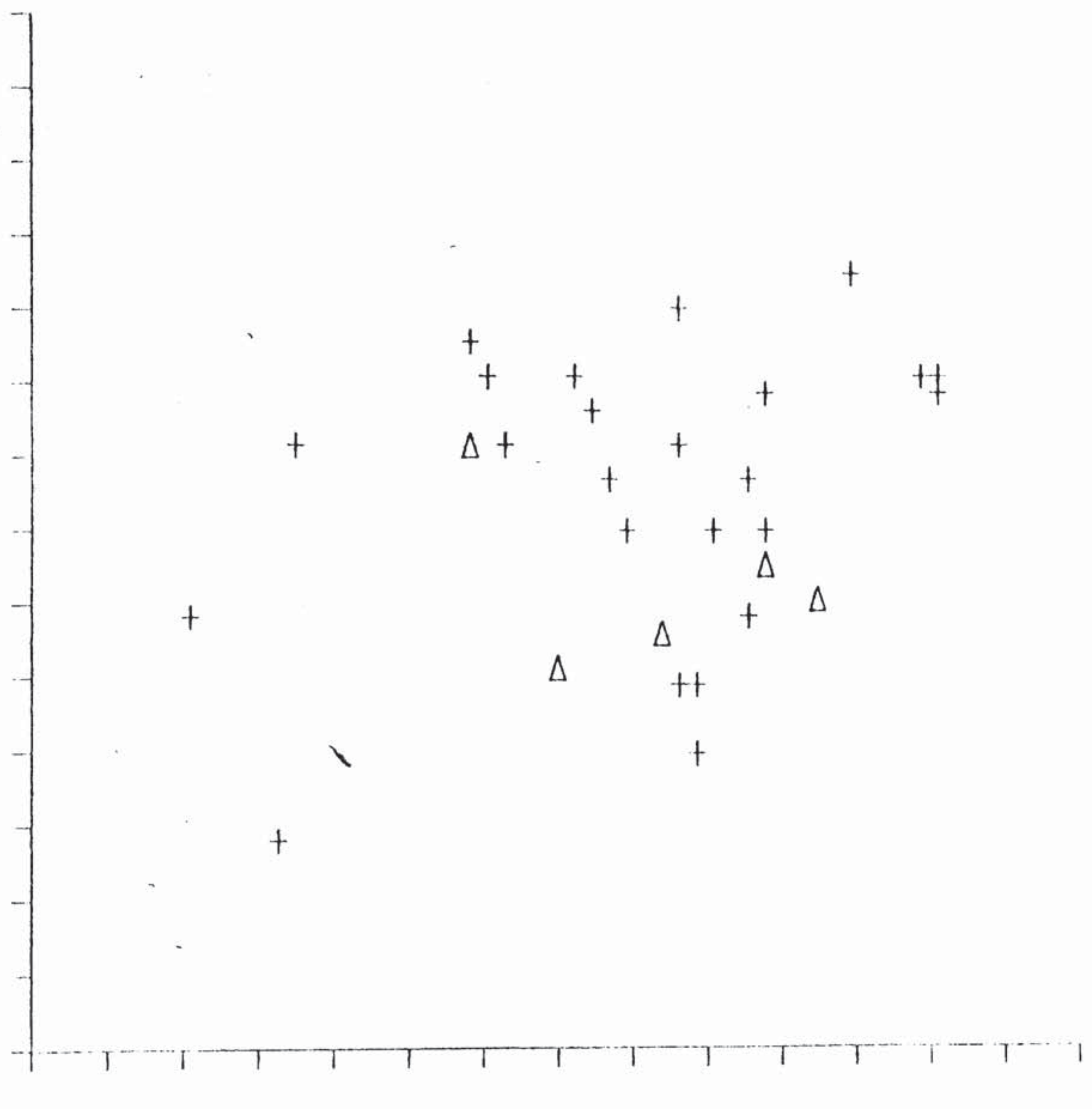

FIGURE 3.4 Magnified map of cluster 3 in Figure 3.2

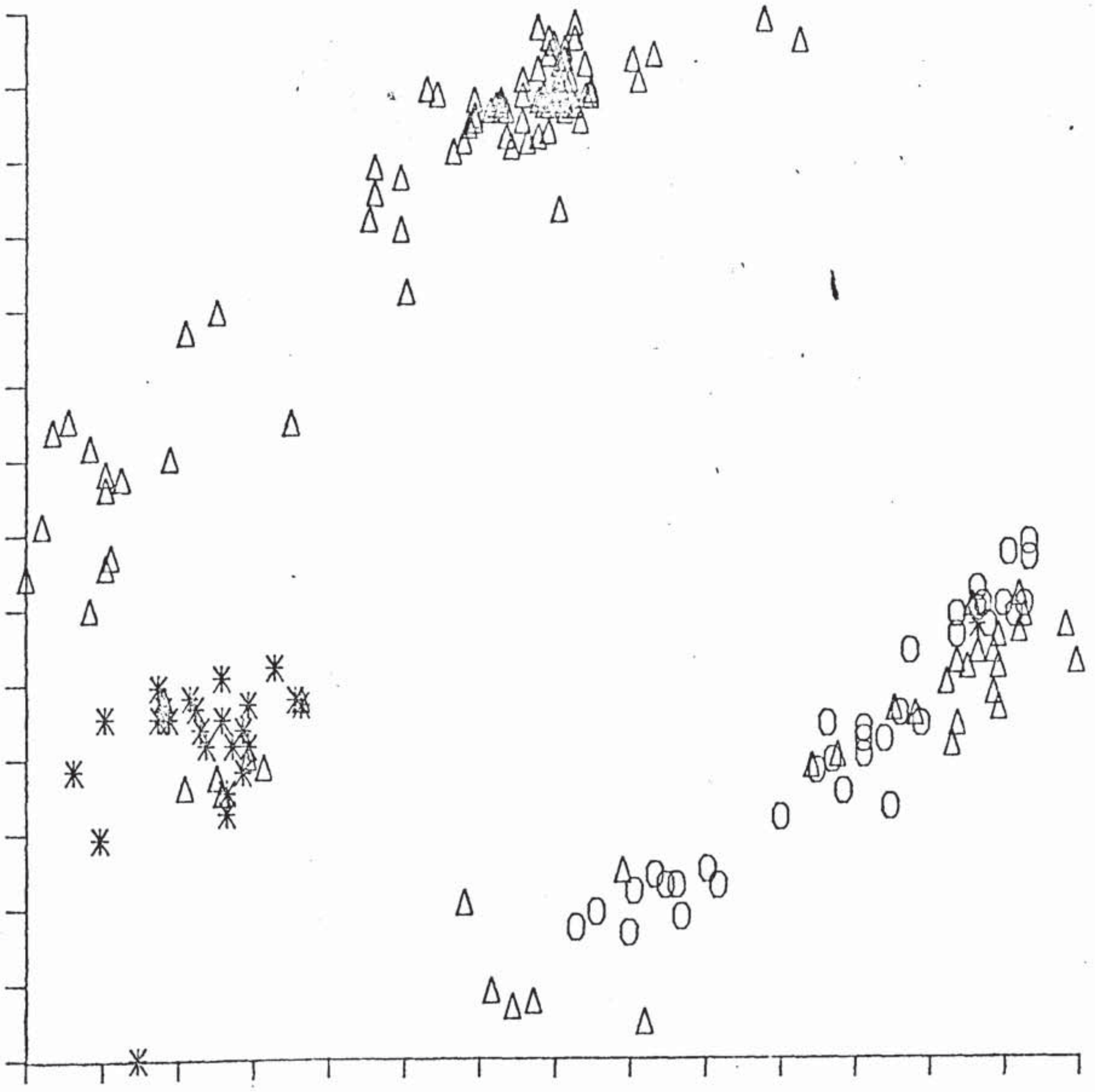

FIGURE 3.5 CA755 data 2-dimensional map classified as SH, S and otherwise compounds.

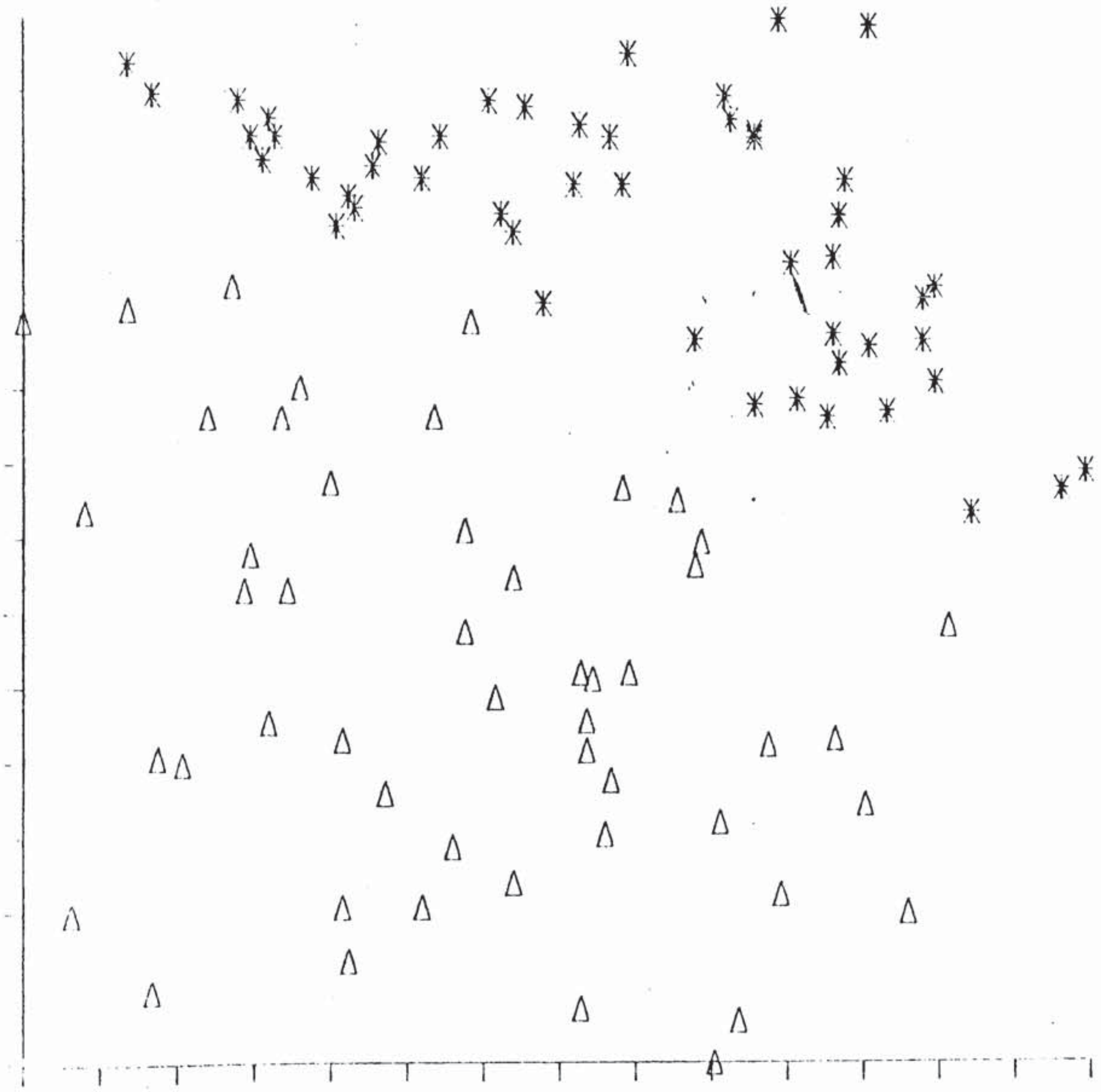

FIGURE 3.6 Artificially generated 2-dimensional map

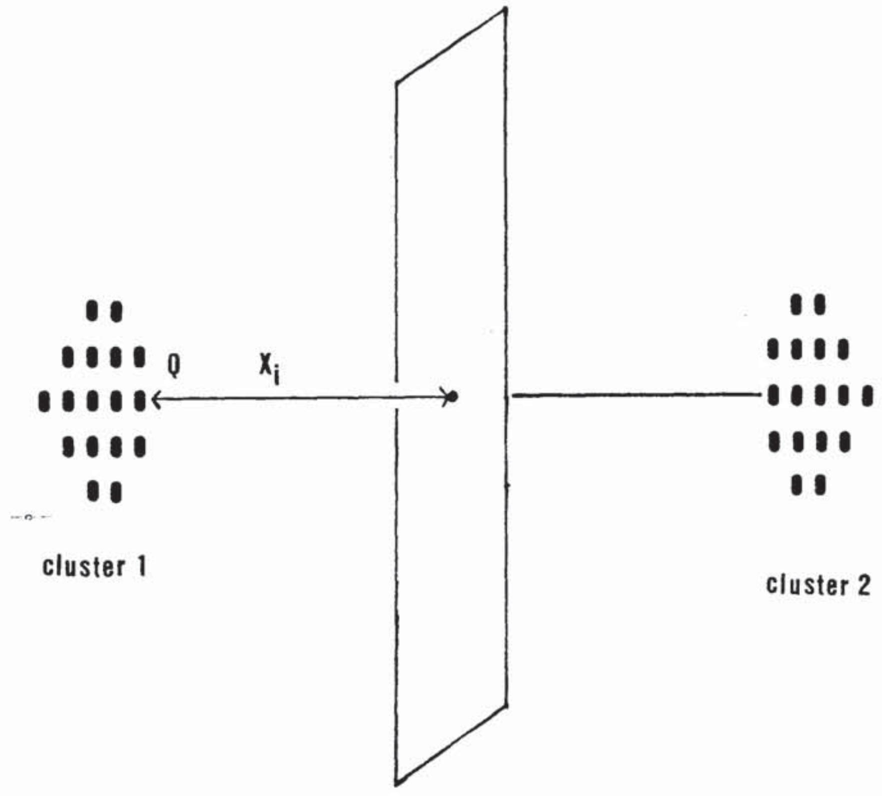

FIGURE 3.7 Hyperplane (transformation function) separation of clusters

CHAPTER FOUR

DISTANCES FREQUENCY DISTRIBUTION

4.1 INTRODUCTION

In this chapter we introduce and consider the theoretical and practical aspects of the distances frequency distribution. The chapter also discusses the results of two applications and concludes with an appraisal of the method. Carl Michael and Sneath (TAXMAP program).

4.2 THEORY

In Sammon's and similar methods, the interpoint distances in the n_1 and n_2 dimensional spaces are expressed in the form of finite time-sequence of $m(m-1)/2$ terms, where m is the number of points in the data set. The time-sequence is a function whose domain is the set of integers $\{1,2,3,\ldots,m(m-1)/2\}$. If d denotes the function, then d_k denotes the kth term. The kth distance between the ith and the jth points is such that

$$k= (i-m)(2m-i)/2-i+j \qquad (4.1)$$

The distances time-sequence is consecutively formed by calculating the distances one by one.

Alternatively, the proposed distances frequency distribution is defined as follows

$$F = \{f_\ell = n(d_\ell) \mid \ell = \text{int}((d_i-d_{min})/w)+1\} \qquad (4.2)$$

where w is the class interval width such that $w=(d_{max}-d_{min})/p$ $n(d_\ell)$ is the number of distances , p is the number of class-intervals, i is such that

$$1\leqslant i\leqslant m(m-1)/2 \qquad (4.3)$$

and d_{min}, d_{max} are the greatest lower bound and least upper bound of distances values respectively.

Replacing the distances time-sequence, the frequency distribution results in first, a considerable reduction in the amount of stored information, and second the frequency distribution curve may be useful in detecting the existence of clusters by revealing the inherent statistical trend in the data. The frequency distribution curve has its ordinate axis as the frequency and its abscissa as the distance value.

The distances time sequence is a special case of distances frequency distribution. This is when the number of class-intervals is equal to the number of distances and that each class-frequency value is one.

4.2.1 NUMBER OF CLASS INTERVALS

In distances frequency distribution, the number of class-intervals must be predetermined, and statistically its value is usually taken between 5 and 20 depending on the data, Spiegel (1972). The lower and upper bounds of the number are 1 and $m(m-1)/2$ respectively, where m is the number of points in the data set. One factor that restricts the number of class-intervals is the number of significant digits in the distances values. If the number of class-intervals is incremented then "gaps" start to appear in the distances frequency distribution. The "gaps" are those class-intervals that have zero frequency. The "gaps" disappear if the number of class-intervals is decremented. The criterion for determining the number is achievement of minimal degree of fluctuations in the frequency distribution curve.

4.2.2 TRANSFORMING THE FREQUENCY DISTRIBUTION

Two transformation categories are related to distances frequency distribution. The first category consists of: rotation of the axes, reflection in the axes and translation of the axes. The second category consists of distances shrinking and stretching.

The first category has no effect on distances values and consequently it affects the co-ordinates only without affecting the frequency distribution. However, the second category affects the frequency distribution though the change is linear and is as if all distances are multiplied by a constant.

4.2.3 NORMALISATION OF CO-ORDINATES

The normalisation of the m co-ordinate values in each n_1 dimension is such that (1) the co-ordinates are transformed into the closed interval (-1,1), or (2) the co-ordinates mean and standard deviation are zero and one respectively. Both normalisation forms are linear transformations.

In the first normalisation form, the co-ordinate g_i of the jth dimension is normalised to g_i' as follows:

$$g_i' = c_1 g_i + c_o \qquad (i=1,m) \qquad (4.4)$$

$$c_1 = 2/(g_{max} - g_{min}) \qquad (4.5)$$

$$c_o = -1-ag_{min}) \qquad (4.6)$$

where g_{min} and g_{max} are the lower and upper bounds of the g co-ordinate set of values.

The second form of normalisation is given by:

$$g_i' = (g_i - \bar{g}_j)/s_j \qquad (i=1,m) \tag{4.7}$$

where $\bar{g}_j$ and s_j are the mean and standard deviation of $g_i(i=1,m)$ respectively.

Computationally the first form of normalisation is less expensive but it may result in reducing the number of significant digits in the co-ordinates values. This is particularly the case when the distribution of points exhibits a value measure of skewness. However, this form of normalisation is useful in providing a bounded frequency distribution as follows:

$$\begin{aligned} d_{max} &= (\Sigma|a-b|^r)^{1/r} \\ &= n^{1/r}\,|a-b| \end{aligned} \tag{4.8}$$

where d_{max} is the maximum distance value, r is a positive integer, a and b are the lower and upper bounds of the n_1-dimensional space normalised co-ordinates respectively. Dividing each element in the distance set by d_{max}, the upper bound of the distances set becomes unity:

$$\ell = \text{int}((d_i - d_{min})/w) + 1 \qquad (i=1,m(m-1)/2) \tag{4.9}$$

$$w = (d_{max} - d_{min})/p \tag{4.10}$$

$d_{min} = 0$

$d_{max} = 1$

therefore

$$\ell = \text{int}(pd_i)+1 \tag{4.11}$$

Multiplying each co-ordinate value by the number of intervals p, the last formula is further reduced to

$$\ell = \mathrm{int}(d_i)+1. \tag{4.12}$$

If the second form of normalisation, called standardisation of variable, is applied to two normally distributed clusters having the properties

$$v_1 \simeq v_2, \tag{4.13}$$
$$v_1 \ll |u_1-u_2| \tag{4.14}$$

and

$$v_2 \ll |u_1-u_2|, \tag{4.15}$$

where v_i and u_i are the variance and mean of the ith cluster. Then the two clusters have their means as

$$u_1 \simeq -1 \tag{4.16}$$

and

$$u_2 \simeq 1. \tag{4.17}$$

That is the two clusters and in all dimensions have the same respective mean and variance values.

4.2.4 DISTANCES FREQUENCY DISTRIBUTION AND CLUSTERS

The frequency distribution of distances is related to the existence and number of clusters in the multivariate data. In order to analyse this relation we consider two simple cases.

The first case is the existence in the multivariate data of one normally distributed cluster:

$$NP_i(x) = m_i \exp[-(x-u_i)^2/2v_i] \tag{4.18}$$

in this cluster there is one set of distances only, that is the inter-cluster set of distances.

The second case is the existence of two normally distributed clusters is given by:

$$Np_1(x) + Np_2(x) \tag{4.19}$$

where m_i, v_i and u_i are the number of points, variance and the mean of the ith cluster, $(i=1,2)$ respectively.

4.2.5 ANALYTICAL MODEL FOR THE DISTANCES FREQUENCY DISTRIBUTION

In order to use the distances frequency distribution and to understand its properties an analytical model for the distances frequency function is presented here. The formulation of the function is one-dimensional and it is generated from two normally distributed clusters. We start from the following frequency function given by

$$f(x) = b_1\exp(-x^2/2v_1)+b_2\exp(-(x-a)^2/2v_2) \tag{4.20}$$

where

$$b_1 = m_1(2\pi v_1)^{-1/2} \tag{4.21}$$

$$b_2 = m_2(2\pi v_2)^{-1/2} \tag{4.22}$$

v_1 and v_2 are the variances of the first and second cluster respectively. m_1 and m_2 are the sizes of the first and second clusters respectively. a is the distance between the means of the first and second clusters. The means of the first and second clusters are assumed to be zero and (a) respectively. The distances frequency function is defined as:

$$g(d) = \int_{-\infty}^{\infty} f(x).f(x+d)\, dx \tag{4.23}$$

The integration gives

$$\begin{aligned} g(d) = {} & m_1^2(2\pi w_1)^{-1/2}\exp|-d^2/2w_1| \\ & + m_2^2(2\pi w_2)^{-1/2} \exp|-d^2/2w_2| \\ & + m_1m_2(2\pi w)^{-1/2} \exp|-(d+a)^2/2w| \\ & + m_1m_2(2\pi w)^{-1/2} \exp|-(d-a)^2/2w| \end{aligned} \qquad (4.24)$$

where

$$w_1 = 2v_1 \qquad (4.25)$$

$$w_2 = 2v_2 \qquad (4.26)$$

and

$$w = \frac{1}{2}(w_1+w_2) = v_1 + v_2 \qquad (4.27)$$

The distances frequency function has been tested with the following parameter values:

$a = 6$

$w_1 = w_2 = 2$

$m_1 = m_2 = 81$

Table 4.1, shows both the function and actual frequency values.

d	Function	Actual
1	2886	2880
2	1396	1328
3	589	480
4	749	688
5	1449	1440
6	1891	1896
7	1441	1440
8	681	696
9	199	160
10	34	16
11	4	0
12	0	0

TABLE 4.1

Ideally, the distances frequency function exhibits two maxima. The first maximum is the sum of two normal functions and the second maximum is merely a normal function. The first maximum results from the shorter intra-cluster distances in the two clusters and the second maximum results from the longer inter-cluster distances.

In the distances frequency function the variance of the second maximum is equal to the sum of the variances of the first and second cluster. This is when the two clusters differ in their variances. In the case of equal variances the distances frequency function becomes such that its two maxima have their variances equal and double the variances of the clusters. Consequently distances frequency distributions exhibit lower power of resolution than the clusters themselves. This means that if the distances freuquency distribution is of resolved peak then stronger cluster separability is expected.

The distance frequency can be useful in estimating the variance and the population of the two clusters in the data set. Also the distance between the two clusters centroids is available from the distances frequency distribution. The estimation of the parameters is particularly simple when the variances of the two clusters are equal. In this case the clusters populations m_1 and m_2 are calculated from the following equations:

$$m = m_1 + m_2 \tag{4.28}$$

$$A = m_1 m_2 \tag{4.29}$$

where A is the area under the maximum of the inter-cluster distances. The variance of the two clusters are calculated from the standard deviation of the same maximum and which is directly available. The constant (a) which represents the distance between the centroids of the two clusters is the distance between the two maxima in the distances frequency distribution curve.

The estimation of parameters can be useful in the determination of the point in the one-dimensional map which optimally discriminates the two normal populations. The point of optimal discrimination is such that

$$f'(x) = 0 \qquad 0<x<a \tag{4.30}$$

This is based on empirical tests. The above equation and after rearrangement results in the following:

$$x = (v/a)\ell n[(m_1/m_2)/(a/s-1)] + a/2 \tag{4.31}$$

This expression has been tested and found to be efficient for the method of direct iteration to solve for x. The best starting value for the iteration is a/2.

4.2.6 DISTANCE MEASURE AND FREQUENCY DISTRIBUTION

The form of the frequency distribution is dependent on the type of the distance measure employed in generating the frequency distribution. Given the general distance measure formula

$$D_{ij} = \left(\sum_{k=1}^{n_1} |g_{ik} - g_{jk}|^r\right)^{1/r} \qquad r\geqslant 1 \tag{4.32}$$

the frequency distribution varies according to r. Depending on the data, there exists a value of r such that the frequency distribution exhibits maximum degree of cluster separability reflected by the width and separation of the peaks in the distances frequency distribution curve.

4.2.7 FEATURE SELECTION AND FREQUENCY DISTRIBUTION

Beside the frequency distribution generated using the distance

$$D_{ij} = (\sum_{k=1}^{r} (g_{ik} - g_{jk})^r)^{1/r}$$

we can also generate n_1 extra frequency distributions from the co-ordinate values of one dimension at a time. The kth dimension frequency distribution results from the distance

$$d_{ij} = |g_i - g_j|. \qquad (4.33)$$

Each frequency distribution exhibits a different degree of cluster separability. This is because of the random measurement errors. Consequently, it is possible to select those dimensions that maximally contribute to cluster separability.

4.2.8 FREQUENCY DISTRIBUTION AND THE ERROR FUNCTION

The frequency distribution error function measures the difference between the fixed n_1 space frequency distribution and the variable n_2 space frequency distribution. It follows that the error is a function of the n_2 space distances frequency distribution which itself depends on the co-ordinates of the points in the n_2 dimensional space.

The frequency distribution error function is invariant against transformations such as rotation of axes, reflection of the axes and translation of axes in the n_1 and n_2 dimensional spaces. However, the error function is not invariant against shrinking and stretching transformations where the n_1 and n_2 space distances change linearly by a constant. There can be more than one form of function. The simplest form is defined as

$$E_1 = \sum_{k=1}^{p} |f_{n1k} - f_{n2k}| \tag{4.34}$$

where f_{n1k} and f_{n2k} are the kth distance frequency in the n_1 and n_2 space respectively and p is the number of class-intervals. The other error function form is defined as

$$E_2 = \sum_{k=1}^{p} (f_{n1k} - f_{n2k})^2. \tag{4.35}$$

The frequencies f_{n1k} and f_{n2k} are defined as

$$f_{n1k} = n(d_k) \tag{4.36}$$

and

$$f_{n2k} = n(D_k) \tag{4.37}$$

where $n(d_k)$ and $n(D_k)$ are the number of d_k and D_k distances such that

$$k = \text{int}\ (d_i) + 1 \tag{4.38}$$

and

$$k = \text{int}\ (D_i) + 1 \tag{4.39}$$

respectively.

The use of non-normalized frequency distribution error function such as 4.34 and 4.35 makes it difficult to measure the rate of success of mapping.

4.3 COMPUTATIONAL ASPECTS

The distances frequency distribution program is similar in many of its parts to the transformation functions program described in the previous chapter. Both programs use linear transformation functions and the main difference is the first program uses distances time-sequence while the second uses distances frequency distribution. The frequency distribution can be regarded as a modification to the first program.

The frequency distribution program consists of the following segments,

(1) the source program,

(2) the minimization subroutine,

(3) the error function subroutine,

(4) the scaling subroutine, and

(5) the n_1-space frequency distribution generation subroutine.

4.3.1 THE SOURCE PROGRAM.

The source program consists mainly of the following parts: the input, the normalization of the multivariate data, the initiation and calling of the minimisation subroutine, the output and the 2-space points plotting. The input part reads the number and dimensionality of the multivariate data and the number of class-intervals. The second input set is the mxn_1 multivariate data matrix and the final input is a set of m integers used as plotting symbols. The normalisation part is similar to the one described in chapter three and it is to a closed interval of (-1.1). The initiation and calling of the minimisation subroutine is described later in this chapter.

The output of the program takes the following form : number of iterations, error function value, the coefficients estimates for the transformation functions, the n_2 space frequency distribution values and the n_2-space m co-ordinates values. The plotting procedure is incorporated in the source program. The procedure is similar to that described in the third chapter except that it is more efficient in handling the plotting symbols.

4.3.2 THE MINIMISATION SUBROUTINE

The minimisation subroutine by Gill et al. (1976) employed in the distances frequency distribution program was of the kind that requires no function derivatives and the independent variables are of fixed upper and lower bounds.

4.3.3 THE ERROR FUNCTION SUBROUTINE

The component parts of the error function subroutine are: the n_2-space co-ordinates generation by the transformation functions, the n_2-space frequency distribution generation and the error function final evaluation.

The generation of the n_2-space points co-ordinates is identical to that in the third chapter.

Distances frequency distribution of the n_2-dimensional space are generated by calculating all $m(m-1)/2$ distances from the mxn_2 co-ordinate values. Frequency distribution generation is accomplished by two nesting loops. The outer loop is the i loop and the inner loop is the j loop. The arguments of the i and j loops are such that:

i from 1 to m-1

j from I+1 to m

The execution frequency of the instructions in the inner loop is m(m-1)/2 per iteration and the kth execution is given by

$k = (i-1)(2m-i)/2-i+j$

The inner loop consists of two instructions, the first is the calculation of the kth n_2-space distance and the second is the classification of that distance in the frequency distribution. The calculation of the kth n_2-space distance is done by the one-dimensional Euclidean distance

$$d_{ij} = \mathrm{abs}(x_i - x_j) \tag{4.40}$$

or the two dimensional Euclidean distance:

$$d_{ij} = |(x_i - x_j)^2 + (y_i - y_j)^2|^{1/2} \tag{4.41}$$

The second instruction

$\ell = \mathrm{int}(d_{ij}) + 1$

is for classifying the distance d_{ij_1} where the ℓth class-interval H2(ℓ) is incremented by 1 for each d_{ij} such that $\ell = \mathrm{int}(d_{ij}) + 1$:

$$H2(\ell) = H2(\ell) + 1 \tag{4.42}$$

or alternatively

$$H2(\ell) = H2(\ell) + d_{ij}. \tag{4.43}$$

The execution outcome of the outer and inner loops is the n_2-space frequency distribution. Each element of the n_2-space frequency distribution is given by

$$f_\ell = n(d_\ell) \tag{4.44}$$

for all i and j such that $\ell = \text{int}(d_{ij})+1$, or in the alternative second case

$$f_\ell = n(d_\ell).d_\ell \tag{4.45}$$

where $n(d_\ell)$ symbolises the number of distances d_ℓ.

The final step in the error function subroutine is the evaluation of the error. Here the two n_1 and n_2 space frequency distributions are compared:

$$E_1 = \sum_{k=1}^{p} \text{abs}(f_{n1k} - f_{n2k})$$

or

$$E_2 = \sum_{k=1}^{p} (f_{n1k} - f_{n2k})^2$$

E_1 can be normalised and made bounded in the interval (0,1) by dividing it by $2(m(m-1)/2)$.

4.3.4 THE SCALING SUBROUTINE

This subroutine is employed to normalise the multivariate data such that the m co-ordinate values of each n_1 dimension are in the interval (-1,1). The normalisation is a linear transformation and as follows:

$$g' = ag + b \tag{4.46}$$

where

$$a = 2/(g_{max} - g_{min}) \tag{4.47}$$

$$b = -1 - ag_{min} \tag{4.48}$$

and g, g' are the original and the normalised co-ordinates respectively.

4.3.5 THE HISTOGRAM SUBROUTINE

This subroutine is used for generating the fixed n_1-space frequency distribution.

The subroutine receives the mxn_1 multivariate data matrix and the number of class-intervals from the source program. The subroutine then results in the distances frequency distribution. The subroutine calculates all $m(m-1)/2$ distances and classifies each one into its class-interval.

The FORTRAN list of the program is given at the end of the thesis under "Program List 2".

4.4 RESULTS

Two data sets have been used to test the distances frequency distribution program. The first set was the Iris flower data and the second is the Adenocarcinom data set. Both sets are described in chapter three.

The two-dimensional space mapping of the Iris data is shown in Figure 4.1. In Figure 4.2 the two-dimensional map of the CA755 data is shown.

As it is apparent from Figure 4.1 the Iris data result is superior to that of the Adenocarcinuma data. Furthermore, the Iris data result seems even better than that obtained by the distances time-sequence program.

One experiment on the Iris data produced what turned out to be a one-dimensional space solution. Figure 4.3. As is the case with the two-dimensional space solution, the one-dimensional mapping exhibits the complete separation of the first Iris flowers class from the second and third classes of flowers. In the solution each coefficient of the first transformation function is equal to corresponding one in the second transformation function. Essentially the solution is identical to the one obtained by the projection of the points on the two-dimensional space mapping on the line passing through the three clusters.

4.4.1 THE PROBLEM OF LOCAL MINIMUM

Throughout testing the frequency distribution program it became evident that the problem of local minimum was more pressing than that in the distances time-sequence program. The problem was acute when the search started such that

$$a_{ij} = 0 \qquad (i=1,n_2)(j=1,n_1) \tag{4.49}$$

It was obvious that beside the global minima, which resulted in solutions of maximum cluster separability, there existed trivial and non-trivial local minima solutions. However in the trivial solution the graphical output was of no apparent meaning. However, the non-trivial solution was the one-dimensional result. In this case the graphical output consists of m points clustered on the y=x line in the xy plane. The coefficients of the transformation functions in the trivial case were either all equal to the same value or they alternate in value. In the non-trivial case each coefficient value in one transformation function was equal to the corresponding one in the second transformation function. This suggests that only one transformation function has contributed in the mapping. It seems to us that one cause behind the existence of local minima is the error function form. The error functions used by Sammon (1969) and in both distances time-sequence and frequency distribution programs have caused the problem of local minima in the search space. One solution to this problem is to increase the number of global minima so as to increase the probability of finding one.

In Sammon's error function and in both time-sequence and frequency distribution programs the minimum is global when

$$D_i = d_i \quad (i=1,m(m-1)/2) \tag{4.50}$$

If instead we make the condition for global minimum when

$$D_i = k\, d_i \quad (i=1,m(m-1)/2) \tag{4.51}$$

where k is a positive quantity. Consequently, the number of global minima is increased and every time the above relation is satisfied a global solution is achieved. This kind of error function can be regarded as invariant to distances shrinking and stretching transformations.

4.4.2 INVARIANT ERROR FUNCTION

In order to formulate the shrinking and stretching invariant error function we proceed from the following expression:

$$E' = \Sigma\,(D_i - d_i)^2 \tag{4.52}$$

and modify it into:

$$E_1 = \Sigma(D_i - k_1 d_i)^2 \tag{4.53}$$

where k_1 is a positive quantity resulting either in shrinking or stretching the configuration d_i $(i=1,m(m-1)/2)$. Now we wish to find k_1 such that E_1 is minimal. Hence,

$$\frac{dE}{dk_1} = -2\Sigma|(D_i - k_1 d_i)d_i| = 0 \tag{4.54}$$

and we have

$$k_1 = \frac{\Sigma d_i D_i}{\Sigma d_i^2} \tag{4.55}$$

The same can be applied on

$$E_2 = \Sigma(k_2 D_i - d_i)^2 \tag{4.56}$$

$$\frac{dE}{dk_2} = 2\Sigma|(k_2 D_i - d_i)D_i| = 0 \tag{4.57}$$

and we have

$$k_2 = \frac{\Sigma d_i D_i}{\Sigma D_i^2} \tag{4.58}$$

Substituting k_1 and k_2 in E_1 and E_2 respectively we get

$$E_1 = \Sigma D_i^2 - \frac{\Sigma^2 d_i D_i}{\Sigma d_i^2} \tag{4.59}$$

and

$$E_2 = \Sigma d_i^2 - \frac{\Sigma^2 d_i D_i}{\Sigma D_i^2} \tag{4.60}$$

Dividing E_1 by ΣD_i^2 or E_2 by Σd_i^2 we have the normalised error function in the interval $(0,1)$:

$$E = 1 - \frac{\Sigma^2 d_i D_i}{\Sigma d_i^2 \; \Sigma D_i^2} \tag{4.61}$$

that is

$$E = 1 - k_1 k_2 \qquad (4.62)$$

and E is minimum when

$$D_i = kd_i$$

For the frequency distribution error function we have

$$E = 1 - \frac{\sum_{i=1}^{p} {}^{2}(n_i d_i)(N_i D_i)}{\sum_{i=1}^{p} (n_i d_i)^2 \quad (N_i D_i)^2} \qquad (4.63)$$

where n_i and N_i are the ith class frequencies of the n_1 and n_2 space frequency distributions respectively.

The invariant and normalised error function has two main properties

(1) E is minimal if $D_i = kd_i$, and

(2) E is bounded in the interval (0,1).

4.4.3 OPTIMAL STEP SIZE GRADIENT METHOD

Another practical problem that has faced both time-sequence and frequency distribution programs has been the execution time. Almost all the execution time is for the error function minimisation. With the optimal step size the path towards the minimum is the shortest. Here we follow Niemann (1979) in constructing a formula for optimal step size. The formula is based on the invariant normalised error function of the previous section for both time-sequence and frequency distribution programs. The formula is also for one-dimensional space mapping.

We start from the (I+1)th iteration error function which is given by;

$$E^{(I+1)} = 1 - \frac{\Sigma^2 D_{ij}\, d_{ij}^{(I+1)}}{\Sigma D_{ij}^2\ \Sigma (d_{ij}^{(I+1)})^2} \qquad i<j \tag{4.64}$$

if we make the substitution

$$d_{ij}^{(I+1)} = d_{ij}^{(I)} - r^{(I)}\, B_{ij} \tag{4.65}$$

where

$$B_{ij} = F_A |Q_i - Q_j| \tag{4.66}$$

F_A being the partial derivative vector and $|Q_i-Q_j|$ being the one-dimensional distance in the n_1-space then the result is the (I+1)th iteration function in terms of $d^{(I)}$ instead of $d^{(I+1)}$.

Differentiating $E^{(I+1)}$ with respect to r, setting the derivative to zero and then solving for r we have

$$r = \frac{\Sigma D_{IJ} D_{ij} \Sigma d_{ij} B_{ij} - \Sigma d_{ij}^2\ \Sigma D_{ij} B_{ij}}{\Sigma\ d_{ij} D_{ij}\ \ \Sigma B_{ij}^2\ - \Sigma d_{ij} B_{ij} \Sigma D_{ij} B_{ij}} \tag{4.67}$$

For distances frequency distribution the expression becomes

$$r = \frac{\overset{p}{\Sigma}\ (n_i d_i)(N_i D_i)\Sigma(n_i d_i)B_i - \overset{p}{\Sigma}(n_i d_i)^2\ \overset{p}{\Sigma}(N_i D_i)B_i}{\overset{p}{\Sigma}\ (n_i d_i)(N_i D_i)\overset{p}{\Sigma} B_i^2 - \overset{p}{\Sigma}(n_i d_i)B_i \overset{p}{\Sigma}(N_i D_i)B_i} \tag{4.68}$$

where n_i and N_i are the ith n_1 and n_2 space class-frequencies respectively.

For the above formulae there is one value of r only.

4.5 CONCLUSION

The target of this chapter was to demonstrate the feasibility of using distances frequency distribution instead of distances time-sequence in mapping multivariate data. Although the method has basically worked problems still remain: these will now be considered.

The first problem is the frequent convergence to local minimum, which suggests the existence of more than one solution beside the global one and that noneof these solutions are as good as the global one. It was thought that the form of the error function is responsible for the appearance of local minima. A generalised error function that is invariate against shrinking and stretching transformations and is normalised in the domain (0,1) was formulated as an answer to the problem of local minima.

The two most important computational aspects considered in this chapter were the reduction in data storage and the construction of an optimal step size gradient method.

With the introduction of distances frequency distribution the n_1-space distances storage requirement became insignificant. In addition, while the storage is approximately proportional to the number of points, the distances frequency distribution storage requirement is almost independent from the number of points in the data set.

The second important computational aspect is the mathematical and computational feasibility of constructing a gradient method that employs optimal step size. The formula is single valued, that is in every minimisation step there is a unique step size value that minimises the error function. It is expected that the optimal step size will result in reducing the execution time by following an optimal path towards the minimum.

Finally, the distances frequency distribution offers a mean for estimating the parameters of the one-dimensional space clusters assuming their normal distribution. This is useful in the determination of the point at which maximum cluster separation exists.

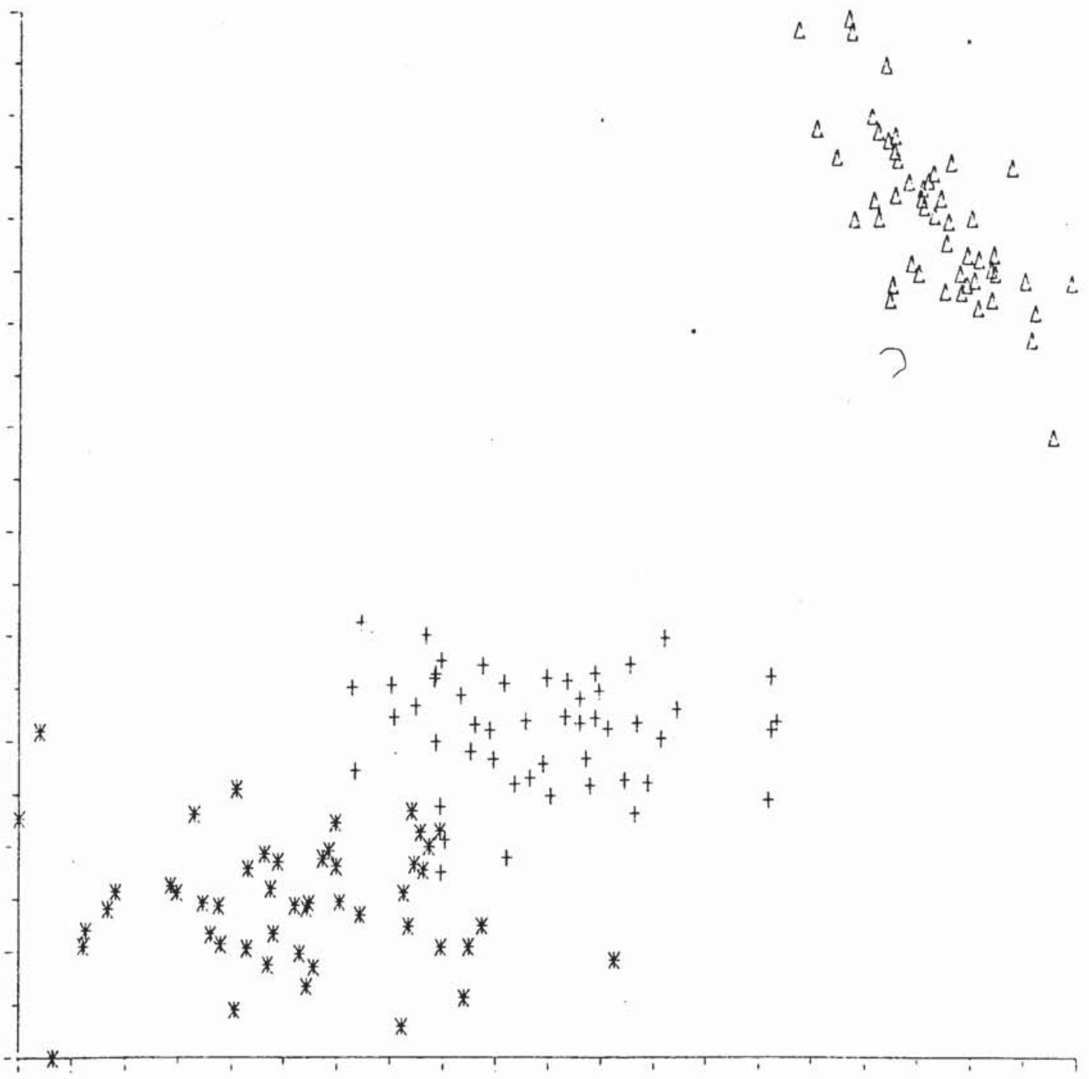

FIGURE 4.1 Iris data 2-dimensional transformation function and frequency distribution mapping.

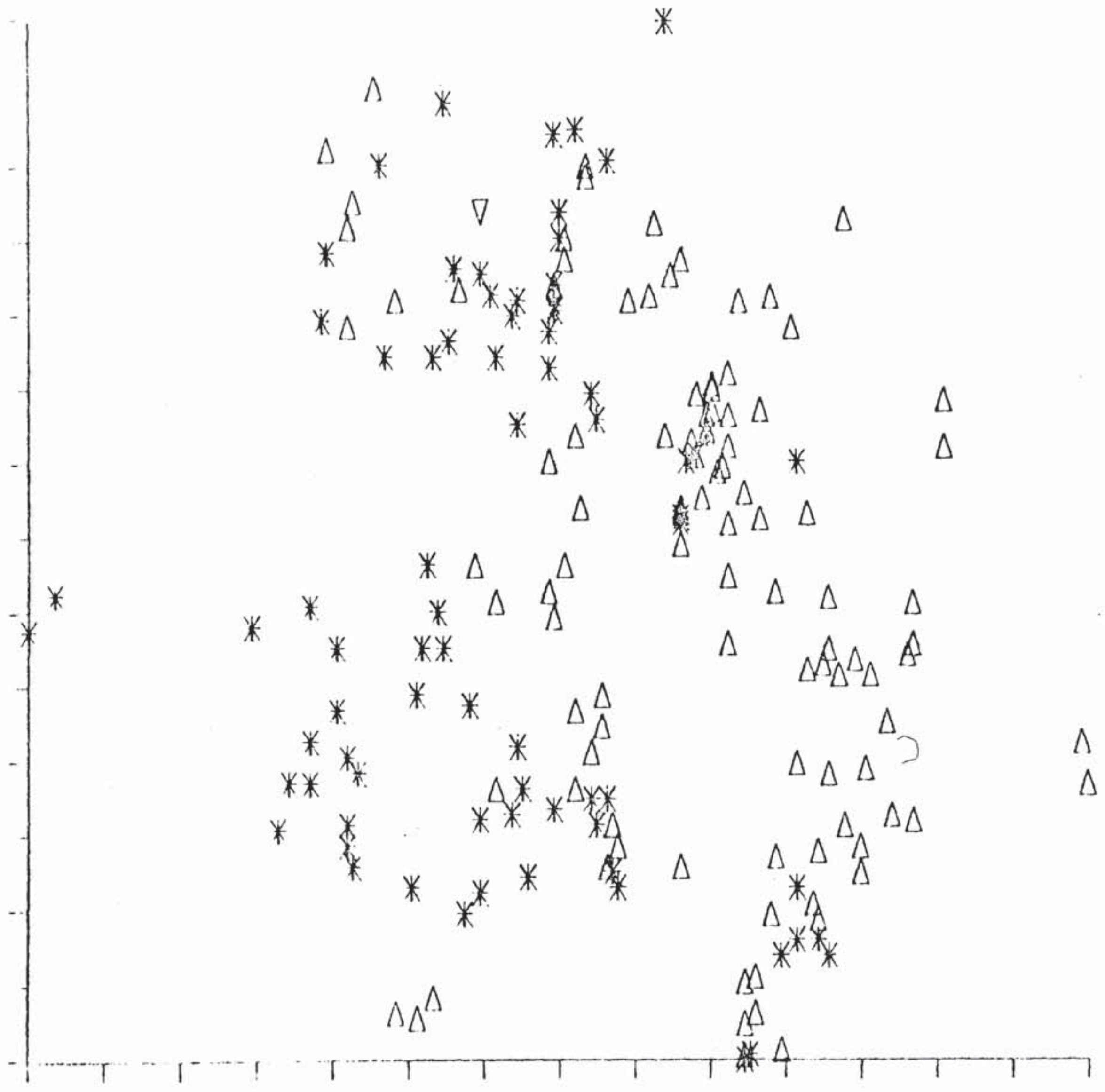

FIGURE 4.2 CA755 data 2-dimensional transformation function and frequency distribution mapping.

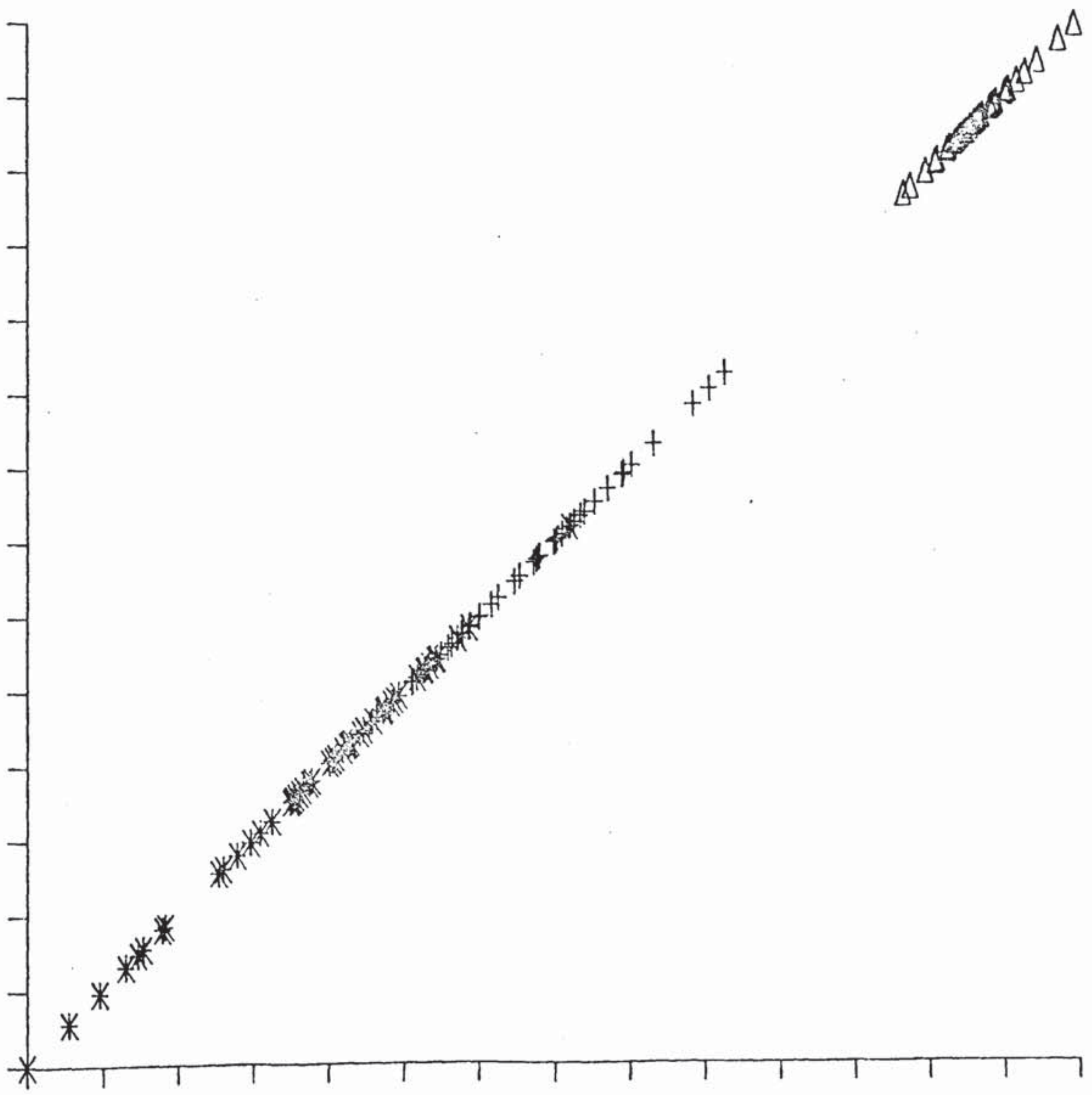

FIGURE 4.3 Iris data 1-dimensional transformation function and frequency distribution mapping.

CHAPTER FIVE

CONCLUSION

CONCLUSION

As it has been commented earlier in Chapter Two, two-dimensional scaling data can be represented as one-dimensional, e.g. taking C- and S-like structures, Shepard (1974). Niemann et al. (1979) successive two-dimensional mapping of a set of characters demonstrated that the clusters were linearly separable allowing the use of one-dimensional mapping. This has been strengthened by evidence from transformation function mapping where again linearly separable clusters do appear. On the other hand, transformation function mapping strengthens the possibility of using one-dimensional mapping instead of two- or three-dimensional mapping. The acceptance of one-dimensional mapping results in first, the bridging of mapping and pattern recognition, second, the possibility of having an automatic cluster separation procedure and third, far less expensive computational procedures in mapping especially in having lesser number of independent variables and simpler one-dimensional Euclidean distance which excludes the expensive square root calculation.

Based on the theoretical and practical results from multi-dimensional scaling, nonlinear mapping and transformation function mapping, we shall describe the aims of a new program together with its important structural properties. The aims of the proposed program, which are meant to be computationally inexpensive and easy to use in minicomputers, are: First, as an efficient pattern recognition procedure and the automatic and exhaustive isolation of clusters. The program is also expected to be useful in feature selection and detecting clustering tendency.

One of the most important results of transformation function is its possible use as a parametric discriminant function similar to that in pattern recognition. For this aim first the data set must have two pattern classes of sufficient separability and the patterns must be known. Second the patterns set is mapped to one-dimensional space and a global solution must be found to ensure maximum pattern recognition.

The resultant transformation function is then standardised such that the patterns of the first class are scattered around minus one and the patterns of the second class are scattered around plus one on the one dimensional axis. In the ideal situation the mapping result is merely two normally distributed patterns of small value standard deviations and minus one and plus one means respectively. The resultant discriminant function is negative when the pattern is from class one and positive when the pattern is from class two. Also the class membership of the patterns is quantitatively measured by the discriminant function value.

The advantage of this discriminant function is in its ability to offer, besides recognising patterns, graphical representation of the patterns space confirming its ability or inability in recognising the patterns.

The second aim of the proposed program is to have an automatic and exhaustive cluster separation procedure. This means that if we start with a number of clusters in the data set then the program should be able to isolate all the clusters in a sequential manner. The automatic cluster separation procedure is done by following Niemann (1979) method of remapping. In order to automate the method we must have the two clusters or groups of clusters in the one-dimensional space centred on the -1 and +1 values respectively and to have a 'clear' area separating them and centred on the origin of the one-dimensional space axis. This is expected to be achieved by the standardisation of co-ordinates before and possibly after each mapping. This might allow the program to isolate all points on the positive side of the one-dimensional axis regarding them as one cluster or group of clusters and mapping is repeated. The same is then done to the points of the negative side of the one-dimensional axis. The procedure stops when the 'clear' area shows signs indicating that what has been left is one cluster.

The other prospect of the proposed program is the direct use of the transformation function coefficients for selecting features. This may be coupled by normalising the coefficients values in such a way so that the most important feature takes the maximum absolute value. The importance of a feature is measured by its misclassification rate and its cluster separability.

The distances frequency distribution in the projected program is expected to be useful in detecting clustering tendency, Dubes et al. (1980). On the other hand, it is intended to exploit the distances frequency distribution in a direct way so as to determine the point in the one-dimensional space that splits the two clusters, paving the way for the automatic clusters separation. It is hoped that an easier procedure will be found which replaces the one described in chapter four.

The new invariant normalised error function will be incorporated in the projected program. This is expected to solve the problem of local minimum and to offer global convergence which is expected to result in a maximum cluster separability solution. Being normalised the new error function offers a common objective measure of error which results from the standardised closed range of variation which is independent from the data used.

The new error function does not have preference to shorter or longer distances. The error function is theoretically sound in being founded on simple difference of squares expression similar to the well known least square. The most important feature of the new error function is its invariancy against shrinking and stretching transformations.

Finally, the program will employ a gradient minimisation method with optimal step size. This is thought to decrement execution time and strengthens the possibility of global convergence. The optimal step size formula is for the invariant normalised error function.

PROGRAM LISTING 1

TRANSFORMATION FUNCTION
2-DIMENSIONAL MAPPING

```
      TRACE 0
      MASTER MAPPING

      DIMENSION W(108),V(8),XNORMAL(149)

      COMMON/B1/M,MLESS1,SUM2,DN2(11026),TIME,KIND
      COMMON/B1/CS(8),L(8),XTRAN,YTRAN,IT,N
      COMMON/B1/Q(149,4),X(149),Y(149),NU(149)

      READ(3,10) N,NPARAM,M,IT,KIND,
     *           (CS(I),L(I),I=1,KIND),
     *           TIME,XTRAN,YTRAN
   10 FORMAT(11I0,3F0.0)

      READ(3,20) ((Q(I,J),J=1,N),I=1,M)
   20 FORMAT(4F0.0)

      DO 60 I=1,N
         DO 40 J=1,M
            XNORMAL(J)=Q(J,I)
   40    CONTINUE

         CALL SCALE(M,XNORMAL,1.0)

         DO 50 J=1,M
            Q(J,I)=XNORMAL(J)
   50    CONTINUE
   60 CONTINUE

```

```
      POWER=2.0/N
      SUM2=0.0E+0
      MLESS1=M-1
      INDEX=0
      NN=M
      DO 61 I=1,M
         NU(I)=I
   61 CONTINUE
      DO 130 I=1,MLESS1
         IPLUS1=I+1
         DO 120 J=IPLUS1,M
            INDEX=INDEX+1
   70       SUM=0.0E+0
            DO 80 K=1,N
               DELTA=Q(I,K)-Q(J,K)
               SUM=SUM+DELTA*DELTA
   80       CONTINUE
      D=SUM**POWER
      IF ( D .NE. 0.0 ) GOTO 110
      DO 90 KK=1,N
         Q(J,KK)=Q(M,KK)
   90 CONTINUE
      NU(J) = NU(NN)
      NN = NN -1
      M=M-1
      MLESS1=M-1
      WRITE(6,100) I,J,NN,NU(I),NU(J),NU(NN)
  100 FORMAT(1H ,6I11)
      GOTO 70
  110 DN2(INDEX)=D
      SUM2=SUM2+D
  120 CONTINUE
  130 CONTINUE

      IFAIL=0
      LW=10*NPARAM+NPARAM*(NPARAM-1)/2
      CALL E04CEF(NPARAM,V,F,W,LW,IFAIL)
      CALL FINAL(IT,F,V,NPARAM,X,Y,M,
                XTRAN,YTRAN,KIND,N,LETTER,NU)
      STOP
      END
```

```
C*********************************************************************
      TRACE 0
      SUBROUTINE FUNCT1(NPARAM,V,F)

      DIMENSION V(1)

      COMMON/B1/M,MLESS1,SUM2,IN2(11026),TIME,KIND
      COMMON/B1/CS(3),L(8),XTRAN,YTRAN,IT,N
      COMMON/B1/Q(149,4),X(149),Y(149),NU(149)

      IT=IT+1

      DO 20 I=1,M
         SUMX=0.
         SUMY=0.
         DO 10 J=1,N
            QIJ=Q(I,J)
            SUMX=SUMX + V(J)  *QIJ
            SUMY=SUMY + V(J+N)*QIJ
   10    CONTINUE
         X(I)=SUMX
         Y(I)=SUMY
   20 CONTINUE
      SUM1=0.0
      INDEX=0

      DO 40 I=1,MLESS1
         IPLUS1=I+1
         XI=X(I)
         YI=Y(I)
         DO 30 J=IPLUS1,M
            INDEX=INDEX+1
            DN=DN2(INDEX)
            DELTAX=XI-X(J)
            DELTAY=YI-Y(I)
            DNLD2=DN-DELTAX*DELTAX-DELTAY*DELTA
            SUM1=SUM1+DNLD2*DNLD2/DN
 30      CONTINUE
 40   CONTINUE
      F=SUM1/SUM2

      CALL UAMILLTIME(A)

      IF(A.LT.TIME) RETURN

      CALL FINAL(IT,F,V,NPARAM,X,Y,M,
     *              XTRAN,YTRAN,KIND,N,LETTER,NU)
      STOP
      RETURN
      END
```

```
  C*****************************************************
  TRACE 0
  SUROUTINE SCALE (M,X,XTRAN)

  DIMENSION X(149)

  XMINIMUM=X(1)
  XMAXIMUM=XMINIMUM

  DO 2 I=1,M
     XI=X(I)
     IF(XMINIMUM.GT.XI) XMINIMUM=XI
     IF(XMAXIMUM.LT.XI) XMAXIMUM=XI
2 CONTINUE

  DO 3 I=1,M
     X(I)=XTRAN*(X(I)-XMINIMUM)/(XMAXIMUM-XMINIMUM)
3 CONTINUE

  RETURN
  END
```

```
      C********************************************************
      TRACE 0
      SUBROUTINE PLOT(XPLOT,YPLOT,KIND,N,LETTER,XTRAN,YTRAN

      DIMENSION XPLOT(149),YPLOT(149),CS(5),L(5)
      DIMENSION UPLOT(149),VPLOT(149)

      CALL OPENGINOGP
      CALL SHIFT2(50.0,50.0)
      INCX=XTRAN/10
      INCY=YTRAN/10
      CALL AXIPOS(1,0.0,0.0,XTRAN,1)
      CALL AXIPOS(1,0.0,0.0,YTARN,2)
      CALL AXISCA(1,INCX,1.0,XTRAN,1)
      CALL AXISCA(1,INCY,1.0,YTRAN,2)
      CALL AXIDRA(1,0,1)
      CALL AXIDRA(-1,0,2)
      NSUM=0
      DO 1 I=1,KIND
         CSI=CS(I)
         LET=L(I)
         DO 2 J=1,NI
            NSUMJ=NSUM+J
            UPLOT(J)=XPLOT(NSUMJ)
            VPLOT(J)=YPLOT(NSUMJ)
    2    CONTINUE
         CALL SYMTO2(UPLOT,VPLOT,NI,LET)
         NSUM=NSUM+NI
    1 CONTINUE
      CALL DEVEND
      RETURN
      END
      C*************************************************
      TRACE 0
      SUBROUTINE FINAL(IT,F,V,NPARAM,X,Y,M,
     *                   XTRAN,YTRAN,N,LETTER,NU)

      DIMENSION V(8),X(149),Y(149),CS([illegible]),L([illegible]),NU(149)

      WRITE(6,1) IT,F
    1 FORMAT(1H ,'F(',I5,')=',E20.11)

      DO 3 I=1,NPARAM
         WRITE(6,2) I,V(I)
    2    FORMAT(1H ,' A(',I2,')=',E20.11)
    3 CONTINUE

      WRITE(2,4) (NU(I),X(I),Y(I),I=1,M)

      CALL SCALE(M,X,XTRAN)
      CALL SCALE(M,Y,YTRAN)
      WRITE(2,4) (NU(I),X(I),Y(I),I=1,M)
    4 FORMAT(1H ,I10,2F12.2)

      CALL PLOT(X,Y,KIND,N,LETTER,XTRAN,YTRAN)

      RETURN
      END
      FINISH
      ****
```

PROGRAM LISTING 2

TRANSFORMATION FUNCTION

AND

FREQUENCY DISTRIBUTION .

2-DIMENSIONAL MAPPING

```
      TRACE 0
      MASTER MAPPING

C     MINIMIZATION BY "QUASI-NEWTON" METHOD

      DIMENSION W(124),IW(10),BL(8),BU(8),V(8),XNORMAL(149)

      COMMON/B1/M,INTERVAL,N,IT,NUMBER(149)
      COMMON/B1/DATA(149,4),X(149),Y(149),HISTOD2(200),
     *          HISTODN(200)
C-------------------------------------------------------------
      READ (3,100) N,M,INTERVAL
      WRITE(6,110) N,M,INTERVAL

      READ (3,120) (  (DATA(I,J),J=1,N),I=1,M)
      WRITE(6,130) (I,(DATA(I,J),J=1,N),I=1,M)

      READ (3,140) (NUMBER(I),I=1,M)
      WRITE(6,150) (NUMBER(I),I=1,M)
```

```
C----------------------------------------------------------------

      DO 30 I=1,N
         DO 10 J=1,M
            XNORMAL(J)=DATA(J,I)
   10    CONTINUE

         CALL SCALE(M,XNORMAL,A,B,-1.0,1.0)

         DO 20 J=1,M
            DATA(J,I) = A*XNORMAL(J) + B
   20    CONTINUE
   30 CONTINUE

      DO 50 J=1,M
         SUM = 0.0
         DO 40 I=1,N
            SUM = SUM + DATA(J,I)
   40    CONTINUE
         SD(J) = SUM
   50 CONTINUE

      WRITE (6,130) (I,(DATA(I,J),J=1,N),SD(I),I=1,M)

      CALL HISTOGRAM (M,N,DATA,HISTODN,INTERVAL)

```

```
C-----------------------------------------------------------------
      CALL UAMILLTIME(T1)
      WRITE(6,190) T1

      NX2 = N * 2

      DO 60 I=1,NX2
         BL(I) = -5.0
         BU(I) =  5.0
   60 CONTINUE
      WRITE(6,160) (BL(I),BU(I),I=1,NX2)

      DO 70 I=1,NX2
         V(I) = 0.0
   70 CONTINUE
      WRITE(6,170) (V(I),V(N+I),I=1,N)

      IT     = 0
      IFAIL  = 1
      IBOUND = 0
      LW     = 12 * 2*N + 2*N * ( 2*N -1 )/2
      LIW    = 2*N + 2
      CALL E04JAF(2*N,IBOUND,BL,BU,V,FUNCTION,IW,LIW,W,LW)

      WRITE(6,250) IFAIL

   80 CALL UAMILLTIME (T2)
      WRITE(6,200) T2
C-----------------------------------------------------------------

      WRITE(6,210) IT,FUNCTION

      WRITE(6,220) (I,V(I),V(I+N),I=1,N)

      INTER = 2 * INTERVAL
      WRITE(6,230) (HISTOD2(I),I=1,INTER)

      CALL SCALE (M,X,AX,BX,0.0,140.0)
      CALL SCALE (M,Y,AY,BY,0.0,140.0)

```

```
C-----------------------------------------------------------------
      CALL OPENGINOGP
      CALL SHIFT2(50.0,50.0)
      CALL AXIPOS(1,0.0,0.0,140.0,1)
      CALL AXIPOS(1,0.0,0.0,140.0,2)
      CALL AXISCA(1,14,1.0,140.0,1)
      CALL AXISCA(1,14,1.0,140.0,2)
      CALL AXIDRA(1,0,1)
      CALL AXIDRA(-1,0,2)
      DO 90 I=1,M
         XX = AX*X(I) + BX
         YY = AY*Y(I) + BY
         WRITE(6,240) I,XX,YY,X(I),Y(I)
         CALL MOVTO2(XX,YY)
         CALL SYMBOL(NUMBER(I))
   90 CONTINUE
      CALL DEVEND
```

```
C-----------------------------------------------------------------

  100 FORMAT(3I0)
  110 FORMAT(3I10)
  120 FORMAT(4F0.0)
  130 FORMAT(I10,4F5.1)
  140 FORMAT(10I0)
  150 FORMAT(10I2)
  160 FORMAT(2F10.2)
  170 FORMAT((2E20.11)/)
  180 FORMAT(I10,4F12.2,F12.2)
  190 FORMAT(' TIME BEFORE MINIMIZATION IS ',F8.3)
  200 FORMAT(' TIME AFTER  MINIMIZATION IS ',F8.3)
  210 FORMAT(' FUNCTION(',I5,')=',E20.11)
  220 FORMAT(4(/,I6,2E20.11))
  230 FORMAT(10F7.0)
  240 FORMAT(I10,4F10.2)
  250 FORMAT(' IFAIL = ',I2)
      STOP
      END
```

```
C*****************************************************************
      TRACE 0
      SUBROUTINE FUNCT1(NX2,V,FUNCTION)

      DIMENSION V(8)

      COMMON/B1/M,INTERVAL,N,IT,NUMBER(149)
      COMMON/B1/DATA(149,4),X(149),Y(149),HISTOD2(200),
     *           HISTODN(200)
      IT=IT+1
      INTER = 2 * INTERVAL

      FACTOR = FLOAT(INTERVAL)/2.0/SQRT(2.0)/FLOAT(N)
      DO 20 I=1,M
         SUMX=0.
         SUMY=0.
         DO 10 J=1,N
            DATAIJ=DATA(I,J)
            SUMX=SUMX + V(J)        *DATAIJ
            SUMY=SUMY + V(J+N)*DATAIJ
   10    CONTINUE
         X(I)=FACTOR*SUMX
         Y(I)=FACTOR*SUMY
   20 CONTINUE

      DO 110 I=1,INTER
         HISTOD2(I) = 0.0
  110 CONTINUE

```

```
      MLESS1 = M - 1
      DO 40 I=1,MLESS1
         IPLUS1=I+1
         XI = X(I)
         YI = Y(I)
         DO 30 J=IPLUS1,M
            L = INT(SQRT((XI-X(J))**2 + (YI-Y(J))**2)) + 1
            HISTOD2(L)=HISTOD2(L)+1.0
 30      CONTINUE
 40   CONTINUE

      SUM1 = 0.0
      DO 50 I=1,INTER
         SUM1 = SUM1 +  ABS( HISTODN(I) - HISTOD2(I) )
 50   CONTINUE

      PI = 1.0
      NX2LESS1 = NX2 - 1
      DO 70 I=1,NX2LESS1
         IPLUS1 = I + 1
         DO 60 J=IPLUS1,NX2
            PI = PI * (V(I) - V(J))
 60      CONTINUE
 70   CONTINUE

      FUNCTION = SUM1/FLOAT(M*(M-1)/2)/2.
      CALL UAMILLTIME(TIME)
      IF ( TIME .LT. 540.0 ) RETURN
      WRITE(6,99) (V(I),I=1,NX2)
 99   FORMAT(E20.11)
```

```
      CALL SCALE (M,X,AX,BX,0.0,140.0)
      CALL SCALE (M,Y,AY,BY,0.0,140.0)
      CALL OPENGINOGF
      CALL SHIFT2(50.0,50.0)
      CALL AXIPOS(1,0.0,0.0,140.0,1)
      CALL AXIPOS(1,0.0,0.0,140.0,2)
      CALL AXISCA(1,14,1.0,140.0,1)
      CALL AXISCA(1,14,1.0,140.0,2)
      CALL AXIDRA(1,0,1)
      CALL AXIDRA(-1,0,2)
      DO 90 I=1,M
         XX = AX*X(I) + BX
         YY = AY*Y(I) + BY
         WRITE(6,240) I,XX,YY,X(I),Y(I)
  240    FORMAT(I10,2F10.0,2F10.2)
         CALL MOVTO2(XX,YY)
         CALL SYMBOL(NUMBER(I))
   90 CONTINUE
      CALL DEVEND


      RETURN
      END
      SUBROUTINE SCALE(M,X,A,B,XMIN2,XMAX2)

      DIMENSION X(149)

      DMIN1 = X(1)
      DMAX1 = X(1)

      DO 1 I=1,M
         IF(XMIN1.GT.X(I)) XMIN1=X(I)
         IF(XMAX1.LT.X(I)) XMAX1=X(I)
    1 CONTINUE

      A = ( XMAX2 - XMIN2 ) / ( XMAX1 - XMIN1 )
      B = XMIN2 - A*XMIN1

      RETURN
      END
```

```
C*************************************************************
      TRACE 0
      SUBROUTINE HISTOGRAM (M,N,DATA,HISTODN,INTERVAL)

      DIMENSION DATA(149,4),HISTODN(200)

      MLESS1=M-1

      DO 4 I=1,MLESS1
         IPLUS1=I+1
         DO 3 J=IPLUS1,M
            SUM=0.0
            DO 1 K=1,N
               SUM=SUM+((DATA(I,K)-DATA(J,K))/2.0)**2
    1       CONTINUE
            L=INT(FLOAT(INTERVAL)*SQRT(SUM/FLOAT(N)))+1

            HISTODN(L)=HISTODN(L)+1.0
    3    CONTINUE
    4 CONTINUE

      INTER = 4 * INTERVAL
      WRITE(6,2) (HISTODN(I),I=1,INTER)
    2 FORMAT(10F7.0)

      RETURN
      END
      FINISH
****
```

PROGRAM LISTING 3

TRANSFORMATION FUNCTION

AND

FREQUENCY DISTRIBUTION

1-DIMENSIONAL MAPPING

```
      TRACE 0
      MASTER MAPPING

C     MINIMIZATION BY "QUASI-NEWTON" METHOD

      DIMENSION W(54),IW(6),BL(4),BU(4),V(4),XNORMAL(149),SD(149)

      COMMON/B1/ M,IN,IT,DATA(149,4),X(149),F2(200),FN(200)

      READ (3,10) N,M,IN
      WRITE(6,11) N,M,IN

      DO 55 I = 1,M
         READ (3,20)    (DATA(I,J),J = 1,N)
         WRITE(6,21) I,(DATA(I,J),J = 1,N)
   55 CONTINUE

      DO 44 I = 1,N
         V(I) =  0.0
         BL(I) = -5.0
         BU(I) =  5.0
         WRITE(6,45) V(I),BL(I),BU(I)
   44 CONTINUE

```

```
      DO 60 I = 1,N
         DO 40 J = 1,M
            XNORMAL(J) = DATA(J,I)
   40    CONTINUE

         CALL SCALE(M,XNORMAL,1.0,0.0)

         DO 50 J = 1,M
            DATA(J,I) = XNORMAL(J)
   50    CONTINUE
   60 CONTINUE

      DO 22 J = 1,M
         SUM = 0.0
         DO 33 I = 1,N
            SUM = SUM + DATA(J,I)
   33    CONTINUE
         SD(J) = SUM
   22 CONTINUE

      WRITE (6,111) (I,(DATA(I,J),J = 1,N),SD(I),I = 1,M)

      CALL HISTOGRAM (M,N,DATA,FN,IN)
      CALL UAMILLTIME(T1)
      WRITE(6,140) T1

      IT     = 0
      IFAIL  = 1
      IBOUND = 0
      LW     = 12*N + N*(N-1)/2
      LIW    = N + 2
      CALL E04JAF (N,IBOUND,BL,BU,V,FUNCTION,IW,LIW,W,LW,IFAIL)

  120 CALL UAMILLTIME (T2)
      WRITE(6,90) T2

      WRITE(6,66) IT,FUNCTION

      WRITE(6,77) (I,V(I),I = 1,N)

      INTER = 3 * IN
      WRITE(6,101) (F2(I),I = 1,INTER)

      WRITE(6,99) (X(I),I = 1,M)
   10 FORMAT(3I0)
   11 FORMAT(3I10)
   20 FORMAT(4F0.0)
   21 FORMAT(I10,4F5.1)
   45 FORMAT(3F10.2)
  111 FORMAT(I10,4F12.2,F12.2)
  140 FORMAT(' TIME BEFORE MINIMIZATION IS ',F8.3)
   90 FORMAT(' TIME AFTER  MINIMIZATION IS ',F8.3)
   66 FORMAT(' FUNCTION(',I5,') = ',E20.11)
   77 FORMAT(4(/,I6,E20.11))
  101 FORMAT(10F7.0)
   99 FORMAT(F10.0)
      STOP
      END
```

```
C***********************************************************************
      TRACE 0
      SUBROUTINE FUNCT1(N,V,FUNCTION)

      DIMENSION V(4)

      COMMON/B1/ M,IN,IT,DATA(149,4),X(149),F2(200),FN(200)

      IT = IT + 1
      INTER = 4 * IN

      FACTOR = FLOAT(IN)/2.0/FLOAT(N)
      DO 20 I = 1,M
         SUMX = 0.
         DO 10 J = 1,N
            SUMX = SUMX + V(J)*DATA(I,J)
   10    CONTINUE
         X(I) = FACTOR*SUMX
   20 CONTINUE

      DO 110 I = 1,INTER
         F2(I) = 0.0
  110 CONTINUE

      MLESS1 = M - 1
      DO 40 I = 1,MLESS1
         IPLUS1 = I+1
         DO 30 J = IPLUS1,M
            L = INT( ABS(X(I)-X(J)) ) + 1
            F2(L) = F2(L) + 1.0
   30   CONTINUE
   40 CONTINUE

```

```
      SUM1 = 0.0
      DO 50 I = 1,INTER
         SUM1 = SUM1 + ABS(FN(I) - F2(I))
   50 CONTINUE

      PI = 1.0
      NLESS1 = N - 1
      DO 70 I = 1,NLESS1
         IPLUS1 = I + 1
         DO 80 J = IPLUS1,N
            PI = PI * (V(I) - V(J))
   80    CONTINUE
   70 CONTINUE

      FUNCTION = SUM1/FLOAT(M*(M-1)/2)/2.

      WRITE(6,60) IT,FUNCTION
   60 FORMAT(' FUNCTION(',I5,') = ',E20.11)

      WRITE(6,61) (V(I),I = 1,N)
   61 FORMAT(4E20.11)

      RETURN
      END
```

```
C*********************************************************************
      TRACE 0
      SUBROUTINE SCALE (M,X,XTRAN,C)

      DIMENSION X(149)

      XMINIMUM = X(1)
      XMAXIMUM = XMINIMUM

      DO 2 I = 1,M
         XI = X(I)
         IF(XMINIMUM.GT.XI) XMINIMUM = XI
         IF(XMAXIMUM.LT.XI) XMAXIMUM = XI
    2 CONTINUE

      XMEAN = ( XMAXIMUM + XMINIMUM ) / 2.0
      DO 3 I = 1,M
         X(I) = XTRAN*(X(I)-XMEAN)/(XMAXIMUM-XMEAN) + C
    3 CONTINUE

      A = XTRAN/(XMAXIMUM-XMEAN)
      B = -XTRAN*XMEAN/(XMAXIMUM-XMEAN)

      WRITE(6,4) XMINIMUM,XMAXIMUM,A,B
    4 FORMAT(' Q MINIMUM = ',E20.11/
     *       ' Q MAXIMUM = ',E20.11/
     *       '           A = ',E20.11/
     *       '           B = ',E20.11)

      RETURN
      END
```

```
C*************************************************************
      TRACE 0
      SUBROUTINE HISTOGRAM (M,N,DATA,FN,IN)

      DIMENSION DATA(149,4),FN(200)

      MLESS1 = M-1
      DO 4 I = 1,MLESS1
         IPLUS1 = I+1
         DO 3 J = IPLUS1,M
            SUM = 0.0
            DO 1 K = 1,N
               SUM = SUM+((DATA(I,K)-DATA(J,K))/2.0)**2
    1       CONTINUE
            L = INT(FLOAT(IN)*SQRT(SUM/FLOAT(N)))+1
            FN(L) = FN(L)+1.0
    3    CONTINUE
    4 CONTINUE

      INTER = 3 * IN
      WRITE(6,2) (FN(I),I = 1,INTER)
    2 FORMAT(10F7.0)

      RETURN
      END
      FINISH
****
```

REFERENCES

1. Boulle, G. J. and Peisach, M.
"Trace element analysis of archaeological materials and the use of pattern recognition methods to establish identity".
J. Radioanal Chem. 50, 209 (1979)

2. Chang, C. L. and Lee, R. C. T.
"A heuristic relaxation method for nonlinear mapping in cluster analysis",
IEEE Trans. Systems Man and Cybernetics, SMC-3, No.2, March(1973), pp. 197-200.

3. Chen, Chi-Hau.
"Statistical pattern recognition",
Hayden Book Company, Washington, D.C., (1973)

4. Chien, Y. T.
"Interactive pattern recognition techniques and systems",
Computer, May (1976), pp. 11-52.

5. Chu, K. C.
"Application of artificial intelligence to chemistry",
Analytical Chemistry, 46, 1181, (1974).

6. Dubes, R. and Jain, A. K.
"Clustering methodologies in exploratory data analysis",
Advances in Computers, edited by Marshall C. Yovits, Vol. 19,(1980)

7. Fisher, R. A.
"The use of multiple measurements in taxonomic problems",
Ann. Eugen., Vol. 7, pp. 179-188, (1936)

8. Gelsema, E. S. and Eden, G.
"Mapping algorithms in ispahan",
Pattern Recognition, Vol. 12, pp. 127-136, (1980)

9. Gill, P. E. and Murray, W.
"Minimisation subject to bounds on the variables",
National Physical Laboratory report NAC 72, (1976)
and
Quasi-Newton methods for unconstrained optimization, JIMA, 1972 Vol 9

10. Goldin, A., Wood, H. B. Jr., and Engle, R. R.
Cancer Chemother. Rep., 1(1$\frac{1}{8}$, 1, (1968).

11. Gray, N. A. B.,
"Constrains on 'Learning Machine' classification method",
Analytical Chemistry, 48, 2269 (1976).

12. James/James,
"Mathematics Dictionary",
4th Edition, Van Nostrand Reinhold, (1976)

13. Jurs, P. C. and Isenhour, T. L.
"Chemical applications of pattern recognition",
John Wiley and Sons, (1975)

14. Klahr, D.
"A Monte Carlo investigation of the statistical significance of Kruskal's non-metric scaling procedure".
Psychometrika, 34, pp. 319-330, (1969)

15. Koskinen, J. R. and Kowalski, B. R.
"Interactive pattern recognition in the chemical laboratory",
J. Chem. Information and Computer Sciences, Vol. 15, No. 2, (1975)

16. Kowalski, B. R.
"Pattern recognition techniques for predicting performance",
Chemtech, pp. 300, May (1974)

17. Kowalski, B. R. and Bender, C. F.
"Pattern recognition. A powerful approach to interpreting chemical data",
J. Amer.Chem. Soc., 94, S632 (1972)

18. Kowalski, B. R. and Bender, C. F.
"Solving chemical problems with pattern recognition",
Naturwissenschaftern, 62, 10-14, (1975).

19. Kowalski, B. R. and Bender, C. F.
"Pattern recognition II. Linear and nonlinear methods for displaying chemical data",
J. Amer. Chem. Soc., 99, 686, (1973)

20. Kruskal, J. B.
"Comments on 'A nonlinear mapping for data structure analysis'",
IEEE Trans. Comp. 1614 (1971).

21. Kruskal, J. B.
"Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis",
Psychometrika, Vol. 29, No. 1, pp. 1-27, March, (1964)

22. Kruskal, J. B.
"The relationship between multidmensional scaling and clustering", in Classification and Clustering edited by J. Van Ryzin, pp. 17-44, (1977)

23. Kruskal, J. B.
"Nonmetric multidimensional scaling: a numerical method",
Psychometrika, Vol. 29, No. 2, pp. 115-129, June (1964)

24. Nelder, J. A. and Mead, R.
"A simplex method for function minimization"
The Computer Journal, 7, pp 308-313, (1969)

25. Niemann, H. and Weiss, J.
"A fast-converging algortihm for nonlinear mapping of high dimensional data to a plane",
IEEE Trans. Comp., Vol. C-28, No. 2, Feb (1979)

26. Orhaug, T. and Severinson-Eklundh, K.
"Clustering by nonlinear mapping",
National Defence Research Institute, S-140, Stockholm 80, Sweden.

27. Pykett, C. E.
"Improving the efficiency of Sammon's nonlinear mapping by using clustering archetypes",
Electronic Letters, 14, 779 (1978)

28. Rosenbrock, H. H.
"An automatic method for finding the greatest or least value of a function",
The Computer Journal, 3, pp. 179-184, (1960)

29. Sammon, J. W., Jr.,
"A nonlinear mapping for data structure analysis",
IEEE Trans.on Computers, C-18, pp. 401-409, May (1969)

30. Schachter, B.,
"A nonlinear mapping algorithm for large data sets",
Computer Graphics and Image Processing, 8, pp. 271-276 (1978)

31. Shepard, R. N.
"The analysis of proximities: multidimensional scaling with an unknown distance function. I",
Psychometrika, Vol. 27, No. 2, June (1962a)

32. Shepard, R. N.
"The analysis of proximities: multidimensional scaling with an unknown distance function. II",
Psychnometrika, Vol. 27, No. 3, September (1962b)

33. Shepard,R. N.
"Representation of structures in similarities: problems and prospects",
Psychometrika, Vol. 39, pp. 373-421, (1974)

34. Speigel, M. R.
"Statistics",
Schaum's outline series, McGraw-Hill Book Company, pp. 28, (1972)

35. Stenson, H. H. and Knoll, R. L.
"Goodness of fit for random ranking in Kruskal's nonmetric scaling procedure",
Psycho. Bulletin, 71, pp. 122-126, (1969)

36. Stenson, H. H.
"The psychological dimensions of similarity among random shapes",
Perception and Psyschphysics, Vo. 3 (3B), pp. 201, (1968)

37. Ting, Kai-Li, H., Lee, R. C. T., Milne, F. G. W., Shapiro, M., and Guarino, A. M.
"Applications of artificial intelligence: relationships between mass spectra and pharmacological activity of drugs",
Science, 180, 417, (1973)

38. Ullman, J. R.
"Pattern recognition techniques",
Butterworths, London (1973)

39. White, I.
"Comments on a nonlinear mapping for data structure analysis",
IEEE Trans. Systems, Man and Cybernetics, SMC-3, No. 2, pp. 197-200, February, (1972)

40. Wolfe, M. A.
"Numerical methods for unconstrained optimization",
Van Nostrand Reinhold, pp. 22, (1978)