28
12/7/79

25 cy to NTIS

UCC-ND

NUCLEAR
DIVISION

UNION
CARBIDE

ORNL/CSD/TM-65

A Probabilistic Method for Grouping Data

C. L. Begovich

MASTER

OPERATED BY
UNION CARBIDE CORPORATION
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
NTIS price codes—Printed Copy: A09; Microfiche A01

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for any third party's use or the results of such use of any information, apparatus, product or process disclosed in this report, or represents that its use by such third party would not infringe privately owned rights.

ORNL/CSD/TM-65

Contract No. W-7405 eng 26

Computer Sciences Division

A PROBABILISTIC METHOD FOR GROUPING DATA

C. L. Begovich

Sponsor: V. E. Kane
Originator: C. L. Begovich

Date Published - November 1979

NOTICE This document contains information of a preliminary nature. *It is subject to revision or correction and therefore does not represent a* final report.

MASTER

UNION CARBIDE CORPORATION, NUCLEAR DIVISION
operating the
Oak Ridge Gaseous Diffusion Plant Oak Ridge National Laboratory
Oak Ridge Y-12 Plant Paducah Gaseous Diffusion Plant
for the
DEPARTMENT OF ENERGY

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. *Reference herein to any specific* commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its *endorsement, recommendation, or favoring* by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not *necessarily state or reflect those of the* United States Government or any agency thereof.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

FIGURE PAGE

LIST OF SYMBOLS

$A_\alpha(j,j')$	Bernoulli random variable which represents the occurrence of sample j and j' in the same cluster for simulation α
$A(j,j')$	frequency of occurrence of sample j and j' in the same cluster for m iterations
$a(j,j')$	realization of the random variable $A(j,j')$
B	between-group dispersion matrix, Eq. (5)
c	number of clusters
d_{ij}	distance between sample i and sample j, Eq. (1)
d_{IJ}	distance between groups I and J
$\underline{E}_j$	error vector associated with observation j
$E(X)$	expected value of X
g	number of groups in the population
$g(c)$	number of groups predicted by PMG when data are clustered into c groups
$\hat{g}$	estimate of number of groups
G_k	group k
$\hat{G}_k$	estimate of G_k
m	number of Monte Carlo iterations
N	number of generated pseudorandom numbers
n	number of observations in data set
p	number of variables or measurements of each sample
q_i	number of random numbers occurring in an interval
s	run of length s

s_i	scale of measurement i
T	total dispersion matrix, Eq. (3)
t	test statistic
W	within-group dispersion matrix, Eq. (4)
w_i	weight of measurement i
$\underline{X}_j$	observation vector j of length p
X_{ij}	j-th measurement on sample i
$\underline{Y}_j$	perturbed observation j of length p
z_i	i-th generated random variable
λ	significance level for the binomial test test statistic
θ	probability that 2 samples are in the same cluster
Σ	covariance matrix
μ	mean

ACKNOWLEDGMENTS

The writer wishes to acknowledge the support of the Computer Sciences Division at the Oak Ridge National Laboratory, operated by Union Carbide Corporation under contract number W-7405 eng 26 with the U. S. Department of Energy.

The support, encouragement, and helpful suggestions of V. E. Kane of the Computer Sciences Division are also sincerely appreciated. Thanks are also due to N. M. Larson and R. C. Gonzalez for their review and suggestions. This work was presented as a thesis for the Master of Science Degree at the University of Tennessee, Knoxville.

ABSTRACT

A probabilistic method for grouping data has been developed to incorporate measurement error into standard cluster analysis procedures. In the analysis, the data are perturbed using Monte Carlo techniques to simulate the experimental error, and the resultant data sets are clustered. By varying the number of clusters, a procedure is given to estimate the unknown number of groups. This technique and other standard procedures for determining the number of groups are described and compared for three different examples. The probabilistic method is shown to have advantages for determining the number of groups and the probabilities for a sample's membership in the hypothesized groups.

CHAPTER 1

INTRODUCTION

1.1 Summary

Cluster analysis is a mathematical method to subdivide data into meaningful groups. Two main purposes of cluster analysis are to determine the number of groups in a data set and to place each data sample or observation into the proper group. Many numerical techniques to perform cluster analysis have been developed over the past ten years [Everitt, 1974].

The purpose of this paper is to suggest a new method for determining the number of groups present in a data set and to compare it with current methods. Estimation of the number of groups is dependent on the definition of group or cluster; some investigation into this definition is necessary. Everitt [1974] reviews various previously proposed definitions and concludes that many are vague. They also use the terms "similarity," "alike," etc., which are not well defined. He suggests that perhaps one single definition cannot be all encompassing.

The definition of groups considered in this paper relies on the clustering of data. A group is a set of samples which are consistently clustered together, even when perturbed by experimental or measurement errors. A definition of consistent clusters and a more rigorous definition of groups is in Section 3.1. The unique features of this definition are not only the dependence on clustering but also the emphasis on experimental error. Both properties are useful for applying the definition to data.

The above definition of groups is the result of a Probabilistic Method for Grouping Data (PMG) developed in this paper. The basic steps included in the method are perturbing the observed data by hypothesized experimental error, clustering the resultant data sets, and summarizing the results. Using three different examples, it is demonstrated that this method results in a practical and meaningful definition of groups.

The PMG method has several advantages. The number of groups is estimated, and instead of distinct clusters being defined, the probability of membership of a data point in a cluster is computed. The method can be used with any clustering technique and does improve the results of that technique in the examples considered. In addition, experimental error is incorporated to avoid results dependent on a single set of measurements.

General notation is described in the remainder of this chapter. Chapter 2 describes clustering methods and current methods for estimating the number of groups. The PMG method is described in Chapter 3 with application of the described methods in Chapter 4.

1.2 Basic Definitions and Notation

Let $\underline{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ denote the vector of p measurements on the j-th sample where $j = 1, 2, \ldots, n$. The quantity $\underline{x}_j$ is _observation_ j or _sample_ j. The purpose of cluster analysis is to divide the data into g groups with n_i representing the number of samples in the i-th group. The basic steps in clustering include scaling and weighting of the data, selecting a distance measure, executing a cluster analysis algorithm, and interpreting the results. These steps are defined below.

Scaling of the data is necessary for some cluster analysis algorithms in order to use variables which have different scales of variation. An observation $\underline{x}_j$ is _scaled_ if

$$x_{ji} = s_i x_{ji} \quad ,$$

where s_i is a scale associated with the i-th measurement. Scaling can remove the different influences of the variable due to varying units and ranges. Usual methods of scaling include dividing each variable by its range or by its standard deviation:

$$s_i = 1/(\max_j x_{ji} - \min_j x_{ji})$$

$$s_i = 1/ (\sum_j [x_{ji} - \bar{x}_i]^2/(n-1)) \text{ where } \bar{x}_i = \sum_j x_{ji}/n \quad .$$

The variables can also be _weighted_ to stress certain variables or sets of variables using any _a priori_ information about the variables:

$x_{ji} = w_i x_{ji}$ where

w_i is the weight for variable i. The usual method of weighting is to select the weights which sum to one.

A distance measure quantifies the "likeness" or "nearness" of two samples. Let d_{kj} be the <u>distance between samples</u> k and j, then $d_{kj} < d_{kj'}$ implies that samples k and j are closer or more alike than samples k and j'. Two commonly used measures are the square of the Euclidean distance

$$d_{kj} = \sum_{i=1}^{p} (X_{ki} - X_{ji})^2 \quad , \qquad (1)$$

and the Mahalanobis distance:

$$d_{kj} = (\underline{X}_k - \underline{X}_j)' S^{-1} (\underline{X}_k - \underline{X}_j) \quad , \qquad (2)$$

where S is the covariance matrix of a sample population. In some clustering analysis techniques, it is necessary to extend the definition of distance between samples to distances between groups. The <u>distance between groups</u> I and J, which will be denoted d_{IJ}, is usually a function of the distance between samples in the groups.

A <u>clustering algorithm</u> separates observations into groups. The selection of the technique is dependent upon time, money, and computer core availability as well as upon theoretical considerations. Some comparisons of different techniques and their applicability to data are in the literature (e.g., see Rand [1971], Slagle et al. [1974], and Gower [1967]). A comparison of techniques based upon their ability to separate bivariate normal populations has also been done to aid in selection of techniques [Bayne et al., 1978].

Estimates for the number of groups and for the group memberships are the results of applying cluster analysis techniques. The analyst should be aware of all physical features of the data in order to interpret the groups realistically. These physical features include the error in the data, any secondary information on similarities of observations, and the purposes for clustering the data. Using a variety of clustering techniques can provide the user with different aspects of the data and their groupings [Kittler, 1976].

The PMG procedure is an attempt at answering two questions of cluster analysis in this interpretive step: "What are the number of groups present in the data set?" and "What is the probability that each sample is in a group?". The number of groups and the observations that define those groups are estimated by incorporating random error. The probability of each observation's group membership is calculated, allowing for any one observation to have a probability of being in more than one group.

CHAPTER 2

BACKGROUND

2.1 History

Clustering observations into different groups is a very intuitive thought process. For example, a small child learns the different animal groups: dogs, cats, cows, horses, etc. People are separated into groups on the basis of sex, age, origin, income level ; the list is endless. Although clustering observations using different measurements is a very natural process, the actual study of methods of clustering analysis has only been widespread since the availability of electronic computers.

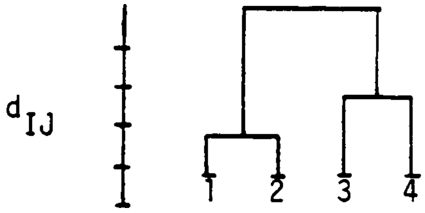
Cluster analysis is considered a technique in the broader field of pattern recognition. The study of pattern recognition includes not only finding the groups in the data, but also defining the groups so that any new observations can be automatically placed into one of the existing groups. Pattern recognition methods which define the groups and classify new objects are implemented in programs such as RECOG-ORNL [Begovich and Larson, 1976] and ARTHUR [Duewer et al., 1975].

Cluster analysis techniques have been developed and applied to scientific fields ranging from artificial intelligence (A) to zoology (Z). Cluster analysis research has been done especially in the areas of biology, psychology, and statistics. Sokal and Sneath [1963] have one of the first books dedicated to analyzing cluster analysis techniques. Everitt [1974] and Anderberg [1973] both have written useful general descriptive cluster analysis references. Other comprehensive references include Fukunaga [1972], Hartigan [1975], Cormack [1971], Lance and Williams [1967, 1968], Nagy [1968], and Dorofeyuk [1971]. The Pattern Recognition Society and the Classification Society, two associations which have developed during the last ten years, are concerned with cluster analysis.

2.2 Clustering Algorithms

Many different cluster analysis algorithms are available. A general classification of these techniques is the division into hierarchical and nonhierarchical methods. Hierarchical methods proceed in a step-wise fashion to combine the data from n single-member clusters to one cluster (agglomerative) or vice versa (divisive). Nonhierarchical techniques are usually optimization algorithms, where an initial set of clusters is updated until a set criterion is optimized.

Agglomerative hierarchical methods, which are the only hierarchical methods described here, consider each sample as a separate cluster at the first step. A distance measure is calculated for each pair of (single-member) clusters and the two samples with the smallest distance are combined to form a new cluster. The distances between the newly formed cluster and the other samples are calculated, and the two closest clusters are combined for the next step. This combination of clusters proceeds until all samples are in one cluster. A tree-like structure called a dendrogram graphically displays the results of the analysis; for example, a dendrogram of four samples is shown below where d_{IJ} is the distance between clusters.



Different hierarchical methods are derived by deciding how to define the distance between two clusters. The most direct method is known as single linkage, in which the distance between two clusters is taken to be the distance between their two closest members. Complete linkage is a slight variation; the distance between two clusters is defined as the largest distance between two of their members. Arithmetic functions of the distances between group members are incorporated in centroid, median, group and weighted average hierarchical clustering methods. Measures of the error sum of squares and the minimum increase in the variance are known as Ward's and variance methods. Descriptions of the various techniques are in Cormack [1971] and Larson et al. [1977]. All of the techniques described above are implemented in a FORTRAN program DENDRO [Larson et al., 1977].

The other class of clustering techniques which will be considered here uses an optimization criterion to find the groups in the data. An initial partition is formed, usually by selecting a set of initial cluster centers and dividing the data points among these centers. Some criterion is selected to test the group memberships. Samples are reallocated to try to improve the criterion until there is no further change.

Optimization procedures differ in the optimization criterion used. One multivariate analysis method deals with dispersion matrices. If the total dispersion matrix,

$$T = \sum_{j=1}^{n} \underline{X}_j' \underline{X}_j \quad , \qquad (3)$$

then T = W + B, where

$$W = \sum_{k=1}^{g} \sum_{i=1}^{n_k} (\underline{X}_i - \underline{C}_k)'(\underline{X}_i - \underline{C}_k) \quad , \qquad (4)$$

$$B = \sum_{k=1}^{g} n_k \underline{C}_k' \underline{C}_k \quad , \qquad (5)$$

with

$$\underline{C}_k = \sum_{j=1}^{n_k} \underline{X}_j / n_k \quad .$$

The matrix W is known as the within-group dispersion matrix and B is the between-group dispersion matrix. Four different clustering criteria are derived from these equations [Friedman and Rubin, 1971]:

1) Minimizing the trace of W (maximizing trace B), equivalent to minimizing the total within group sum of squares.

2) Minimizing the determinant of W, equivalent to minimizing Wilk's lambda statistic.

3) Maximizing the largest root of $B - \lambda W = 0$, referred to as the largest root test.

4) Maximizing the trace of $W^{-1}B$, known as the Hotelling's trace criteria.

McRae [1972] has implemented these optimization methods in a FORTRAN program, MICKA. The clustering in MICKA is performed in two steps. The first step uses a k-means procedure developed by MacQueen [Anderberg, 1973]. The second step uses one of the above criterion to test a sample's group membership.

An alternative optimization technique is NORMIX [Wolfe, 1971]. Here the data are assumed to be a mixture of multivariate normal populations. Thus, the optimal division is separation of the data in order to maximize the likelihood function. The iterative equations consist of

$$n_i/n = 1/n \sum_{k=1}^{n} \hat{P}(G_i | \underline{X}_k) \quad ,$$

$$\underline{C}_i = 1/n_i \sum_{k=1}^{n} \hat{P}(G_i | \underline{X}_k) \underline{X}_k \quad , \qquad (6)$$

$$\Sigma_i = 1/n_i \sum_{k=1}^{n} (\underline{X}_k - \underline{C}_i)(\underline{X}_k - \underline{C}_i)' \hat{P}(G_i | \underline{X}_k) \quad ,$$

where

n_i/n = the mixing proportion for cluster i,

$\underline{C}_i$ = the mean of cluster i,

Σ_i = the covariance matrix of cluster i,

$\hat{P}(G_i | \underline{X}_k)$ = estimated probability that sample k is in cluster i.

From an initial configuration, NORMIX uses a simplified "Aitken" iterative scheme until convergence. Initial clustering is done using Ward's method with Mahalanobis distance; alternatively, the user may input an initial set of clusters.

A third optimization method for clustering multivariate data is the heuristic interactive program ISODATA [Ball, 1965]. The set of input parameters, which a user can change at each step, includes an initial guess at the number of clusters, a smallest allowable cluster size, a cluster splitting parameter, and a cluster lumping parameter. Documentation for this procedure is given in Ball [1965].

Tou and Gonzalez [1974] also give a description of the program with examples.

The four specific methods described above and implemented in DENDRO, MICKA, NORMIX, and ISODATA were chosen to be included in this study because of their applicability to determining the number of groups. These diverse methods are also in widespread use. A summary of these methods is presented in Table 1. Procedures used to estimate the number of groups are in the next section.

2.3 Estimating the Number of Groups

Hierarchical clustering techniques group the observations from n to one groups. Optimization techniques require at least an initial guess for the number of groups. An estimate of the number of groups should be a result of these techniques, however.

The dendrogram described in Section 2.2 is useful for indicating the total structure of data. The number of clusters actually present in the data is left to the user's judgment. If there is a large separation of distances between two or more clusters shown in the dendrogram, then the clusters are distinct and well defined. However, a dendrogram on real data rarely displays large differences, giving little inference to the number of groups in the data.

Some of the optimization methods change the number of clusters while iterating; the resultant number of clusters as well as the cluster separation is optimized. The

Table 1. Summary of cluster analysis techniques described in this study

Technique	Clustering method	Program used
Hierarchical	Single linkage	DENDRO
(Agglomerative)	Complete linkage	
	Group average	
	Weighted average	
	Centroid	
	Median	
	Ward's method	
	Variance	
Optimization	Trace W	MICKA
	Det W	
	Trace $W^{-1}B$	
	Root of $W^{-1}B$	
	Maximum Likelihood	NORMIX
	Group separation	ISODATA

majority of methods, however, optimize only on the group separations. In this case, the optimization criterion for different number of clusters can be compared to determine the number of clusters present in the data.

The clustering criteria available in MICKA are used only to optimize on the group membership and not the number of groups. A test for determining the number of groups is to plot the criteria versus the number of groups; a sharp change in the value of the criteria, followed by a small percentage change, can be used to indicate the correct number of groups. This procedure for estimating the number of groups has been found to be unsatisfactory [Everitt, 1974].

NORMIX does not change the number of clusters within an optimization; however, a number of different guesses for the number of clusters can be tried within one run. Wolfe [1971] has determined a significance test for rejecting the null hypothesis that fewer clusters, r, exist rather than more clusters, r', using the maximum likelihood estimates, L_r and $L_{r'}$:

$$-2(1/n)(n-1-p-r'/2)\log(L_r/L_{r'}) \quad , \tag{7}$$

which is a χ^2 distribution with 2p(r-r') degrees of freedom.

The ISODATA procedure iterates on the number of clusters present in the data as well as the members of the clusters. ISODATA is a very elaborate procedure and requires user interaction [Anderberg, 1973]. The parameters which determine if lumping or splitting of the groups is to cccur are difficult to determine. Dubes [1976] concludes that the ISODATA procedure is very sensitive to the input parameters and requires several runs to get reasonable results.

Some mathematical indicators have been used to test the number of clusters present in the data independent of the method used to determine the clusters. Everitt [1974] describes three methods. The first, attributed to Beale, is an F statistic

$$F(r,r') = \frac{S_r - S_{r'}}{S_{r'}} \Big/ \left[\frac{n-n_r}{n-n_{r'}}\left(\frac{n_r}{n_{r'}}\right)^{\frac{2}{p}} - 1\right] , \qquad (8)$$

where S_r denotes the trace W for r groups, which tests the significance of r' over r groups with p(r-r') and p(n-r') degrees of freedom. The second method, suggested by Calinski and Harabasz, is based on the variation of the ratio Kg

$$Kg = \frac{\text{trace } B}{g-1} \Big/ \frac{\text{trace } W}{n-g} , \qquad (9)$$

where B and W are defined as in Eqs. (4) and (5). The distribution of Kg determines the number of groups present: if Kg reaches a maximum for j, then there are j groups present; if Kg increases monotonically, there are no groups; and if Kg decreases monotonically, the samples have a hierarchical structure. The third indicator results from an investigation of the determinant of W criterion by Marriot [1971]. He suggests that g^2(det W) should be at a minimum when g groups are present in the data.

Nonparametric mode-seeking [Fukunaga and Hostetler, 1975], valley-seeking [Koontz and Fukunaga, 1972], and graph theoretical algorithms [Koontz et al., 1976] are also methods for estimating the number of groups. These algorithms are iterative procedures which estimate the number of clusters by searching for regions of dense or sparse concentrations of observations. Graph theoretical techniques can be used to extend these methods to find irregular shaped clusters [Koontz et al., 1976].

CHAPTER 3

A PROBABILISTIC METHOD FOR GROUPING DATA

The incentive for a new method to determine the number of groups is motivated by the desire to improve upon the current techniques and to incorporate experimental error into a clustering procedure. Data used in cluster analysis consists of observations; each measurement has experimental error associated with it. The experimental error is likely to affect the clustering results [Nagy, 1968]. Preliminary investigation of a data set can be used to find outliers and extreme experimental errors [i.e., Kane et al., 1977]; however, in typical clustering applications the dependence of the groups upon experimental error is unclear.

The Probabilistic Method for Grouping Data (PMG) uses Monte Carlo simulations to perturb the data within a specified range as the first step. The combination of Monte Carlo techniques and cluster analysis has been used previously to test the dependence of variables [Borucki et al., 1975], to test the significance of a technique [Ling, 1971], and to compare cluster analysis techniques [Rand, 1971 and Bayne et al., 1978]. The PMG procedure uses Monte Carlo techniques to approximate the experimental error. An earlier study by Kane and Larson [1976] also investigated the use of Monte Carlo perturbations in cluster analysis.

The perturbed data sets, created with Monte Carlo techniques, are each clustered as the next step in the PMG algorithm. Any set of samples which groups together for a large percentage of the perturbed data sets is defined to be a group. The probability of each sample being in a group is given in terms of the number of times that a sample occurred with that defined group.

The number of groups estimated by this method, in some sense, is the maximum number of groups that an analyst should consider. Any larger number of groups is dependent on the experimental error. Any smaller number of groups, however, might make more sense in terms of the physical situation. For example, apples, oranges, potatoes, and peas are four groups of foods, but they can be combined into only two groups of fruits and vegetables.

The application of the PMG procedure is especially useful for data in which measurement errors are known, such as chemical analyses and biological tests. The procedure can also be used to find the significance of clusters or variables used in the clustering by defining the experimental error as significance bounds for the variables.

The PMG method is implemented in a set of FORTRAN programs documented in Appendix A. General formulas of the method are described in Section 3.1 and the details of their implementation are stated in Section 3.3. Comparisons of this method with the techniques described in Section 2.3 are in Chapter 4.

3.1 Theory of the PMG Procedure

The notation in the development of the PMG procedure uses a capital letter to denote a random quantity, with a lower case letter denoting the realization of the corresponding upper case variable. Estimation of any variable will be denoted by a "$\wedge$" superscript, i.e. $\hat{g}$ is the estimate of the number of groups in the data, g. The groups are represented by sets G_k with n_k samples.

A perturbed sample, $\underline{Y}_{j\alpha}$, is formed by combining the original sample measurement, $\underline{X}_j$, and a random vector, $\underline{E}_{j\alpha}$

$$\underline{Y}_{j\alpha} = \underline{X}_j + \underline{E}_{j\alpha} \tag{10}$$

where $\underline{E}_{j\alpha}$ is a random vector denoting the experimental error associated with $\underline{X}_j$ for the α-th simulation. The collection of n samples are perturbed to form data set α, for $\alpha = 1, 2, \ldots, m$. An example of the above error analysis is given by Kane and Larson [1976].

The justification of the PMG method is clustering of the perturbed data into c groups, for increasing values of c. The data are first grouped into the smallest number of groups which could possibly exist in the data. The value of c = 2 is used if no additional information about the smallest possible number is known. The number of groups that appear consistently through the clustering of the data is estimated and denoted by $g(c)$. The value of c is increased by one and the clustering of the perturbed data sets is performed. Again the number of groups, $g(c)$ is defined. The method continues until $g(c) = g(c+1) = g(c+2)$. The estimate of the number of groups is then $g = g(c)$.

The motivation for this technique is derived from considering the results of clustering g distinct groups into c groups where $c < g$ or $c > g$. If $c < g$, then each group clustered will be either one of the g groups or a collection of two or more of the groups. None of the g groups will be subdivided to form the c clusters, considering that the intergroup distances are larger than the intragroup distances. For example, if the data have four groups, A, B, C, and D, and c is set to two, one of the groups formed by the clustering will be either A, B, C, D or a combination of groups AB, AC, AD, ABC, ABD, BCD. If $c = g$, the data will cluster into the g groups. If $c > g$, the groups in the data will be forced to subdivide in order to form c clusters. When clustering the perturbed data sets, the subdivision of the groups is due to fluctuations in the error, and the number of groups which appear consistently in the clustering will be the g groups, so $g(c) = g$.

The method for finding the groups which consistently appear when clustering the m different data sets for one value of c is given below. For each data set clustered, a Bernoulli random variable, $A_\alpha(j,j')$, is used to associate the sample pairs j and j' that appear in the same cluster

$$A_\alpha(j,j') = \begin{cases} 1 \text{ if } \underline{Y}_{j\alpha} \text{ and } \underline{Y}_{j'\alpha} \text{ are in the same cluster} \\ 0 \text{ if } \underline{Y}_{j\alpha} \text{ and } \underline{Y}_{j'\alpha} \text{ are in different clusters.} \end{cases} \qquad (11)$$

The value of $A_\alpha(j,j')$ summarizes the clustering of one perturbed data set. The quantity $A_\alpha(j,j')$ is a Bernoulli random variable with $\Pr(A_\alpha(j,j') = 1) = \Pr(\underline{Y}_{j\alpha}$ and $\underline{Y}_{j'\alpha}$ are in the same cluster$) = \theta$.

The collection of $A_\alpha(j,j')$ for all sample pairs j and j' forms a symmetric matrix. A_α can be summed over all m iterations to form a frequency matrix of the number of times each sample pair occurred in the same cluster. The j,j' member of A, A(j,j'), can be considered a measure of distance between sample j and j' since $A(j,j') > A(i,j')$ implies that sample j was in a cluster with j' a greater number of times than i was in a cluster with j', i.e., samples j and j' have a higher probability of being in the same cluster than samples i and j'. The individual elements of the frequency matrix are also binomially distributed variables resulting from the sum of m Bernoulli trials.

A distance measure between sample $\underline{X}_j$ and $\underline{X}_{j'}$ can be defined as

$$d_{jj'} = 1 - A(j,j')/m \quad . \qquad (12)$$

The measure $d_{jj'}$, a frequency measure, assumes the value 0 if sample j and sample j' always occurred in the same cluster (i.e., $A(j,j') = m$) and the value 1 if sample j and sample j' never occurred in the same cluster (i.e., $A(j,j') = 0$). This distance measure can be input to a clustering algorithm to form a dendrogram or to cluster into a specified number of groups. Examples of dendrograms formed from $d_{jj'}$ are in Chapter 4.

Simply clustering the frequency measure does not improve upon an estimate for the number of groups since the number of groups and the group membership are still only interpretable using the available techniques of the methods used to cluster $d_{jj'}$. However, it is possible to evaluate the number of groups and the probability of group membership by using the binomial distribution properties of A(j,j').

From the binomial properties of A(j,j')

$$E[A(j,j')] = m\theta \quad . \tag{13}$$

For A(j,j') close to m, there is a high probability that $\underline{X}_j$ and $\underline{X}_{j'}$ are in the same group. For a given θ ,

$$\lambda = \Pr[A(j,j') \geq a_0] = \sum_{v=a_0}^{m} \binom{m}{v}\theta^v(1-\theta)^{m-v} \tag{14}$$

from the binomial formula. Therefore, to find the groups $\hat{G}_k$ and estimate g(c), select a particular λ and probability level θ , and calculate the resulting a_0. Then search all sample pairs and find all j and j' where $a(j,j') \geq a_0$. Combine the pairs to form $\hat{G}_k$, where

$$\hat{G}_k = \left\{\underline{x}_j, \text{ for some } j = 1, 2, \ldots, n_k\right\} , \qquad (15)$$

such that $\underline{x}_j \in \hat{G}_k$ implies that there exists $\underline{x}_{j'} \in \hat{G}_k$ where $a(j,j') \geq a_0$. The value of $g(c)$ is set to the number of $\hat{G}_k$ formed.

The set $\hat{G}_k$ is cne of the clusters which consistently appear throughout the clustering of the m data sets into c groups, since each $\underline{X}_j$ in $\hat{G}_k$ appeared with at least one other member of $\hat{G}_k$ a_0 times. Note that $\underline{X}_j, \underline{X}_{j'} \in \hat{G}_k$ does not imply $a(j,j') \geq a_0$, but instead $a(j,i) \geq a_0$ and $a(j',i') \geq a_0$, for some $\underline{X}_i$ and $\underline{X}_{i'}$ in $\hat{G}_k$. This definition of $\hat{G}_k$ allows chaining; that is, if $a(j,j') \geq a_0$ and $a(i,j') \geq a_0$, samples $\underline{X}_i$, $\underline{X}_j$, and $\underline{X}_{j'}$ are a member of $\hat{G}_k$ even if $a(i,j) < a_0$.

A probability measure can be defined to calculate the probability that sample j is in group G_k. Let the data be partitioned into g groups $\hat{G}_1, \hat{G}_2, \ldots, \hat{G}_g$ as given above. Consider $P(\underline{X}_j \in G_k)$ for a fixed sample j, where $P(\underline{X}_j \in G_k)$ is the probability that $\underline{X}_j$ and $\underline{X}_{j'}$ are members of the same group for all $\underline{X}_{j'}$ in group k. The maximum likelihood estimate of the probability that $\underline{X}_j$ and $\underline{X}_{j'}$ are in the same cluster is $a(j,j')/m$. Using the approximation of the groups $\hat{G}_k$, the $P(\underline{X}_j \in \hat{G}_k)$ is equal to the estimate that $\underline{X}_j$ and $\underline{X}_{j'}$ are in the same group times the probability of $\underline{X}_{j'}$, given $\hat{G}_k$, summed over all members of $\hat{G}_k$, or

$$P(\underline{X}_j \in \hat{G}_k) = \sum_{\underline{X}_{j'} \in \hat{G}_k} a(j,j')/m \ (1/n_k) \quad . \qquad (16)$$

Note that the summation of $P(\underline{X}_j \epsilon \hat{G}_k)$ over all k is not necessarily equal to one since a(j,j')/m is an estimate of the probability that sample j and j' are in the same cluster and since the $\hat{G}_k$ are only estimates of the groups G_k. With normalization the probability that sample j is in any of the groups is one

$$P(\underline{X}_j \epsilon \hat{G}_k) = \sum_{\underline{X}_{j'} \epsilon \hat{G}_k} a(j,j')/n_k \Big/ \sum_{k=1}^{g} \sum_{\underline{X}_{j'} \epsilon \hat{G}_k} a(j,j')/n_k \quad . \tag{17}$$

The above probability measure can be shown to satisfy the three axioms of probability:

$$1)\ \sum_{k=1}^{g} \left[\sum_{\underline{X}_{j'} \epsilon \hat{G}_k} a(j,j')/n_k \Big/ \sum_{k=1}^{g} \sum_{\underline{X}_{j'} \epsilon \hat{G}_k} a(j,j')/n_k \right] = 1 \quad ,$$

$$2)\ 0 \leq P(\underline{X}_j \epsilon \hat{G}_k) \leq 1 \quad , \text{ and} \tag{18}$$

$$3)\ P(\underline{X}_j \epsilon \hat{G}_i) + P(\underline{X}_j \epsilon \hat{G}_k) = P(\underline{X}_j \epsilon \hat{G}_i \text{ or } \underline{X}_j \epsilon \hat{G}_k)$$
$$\text{for } \hat{G}_i \cap \hat{G}_k = \phi \, .$$

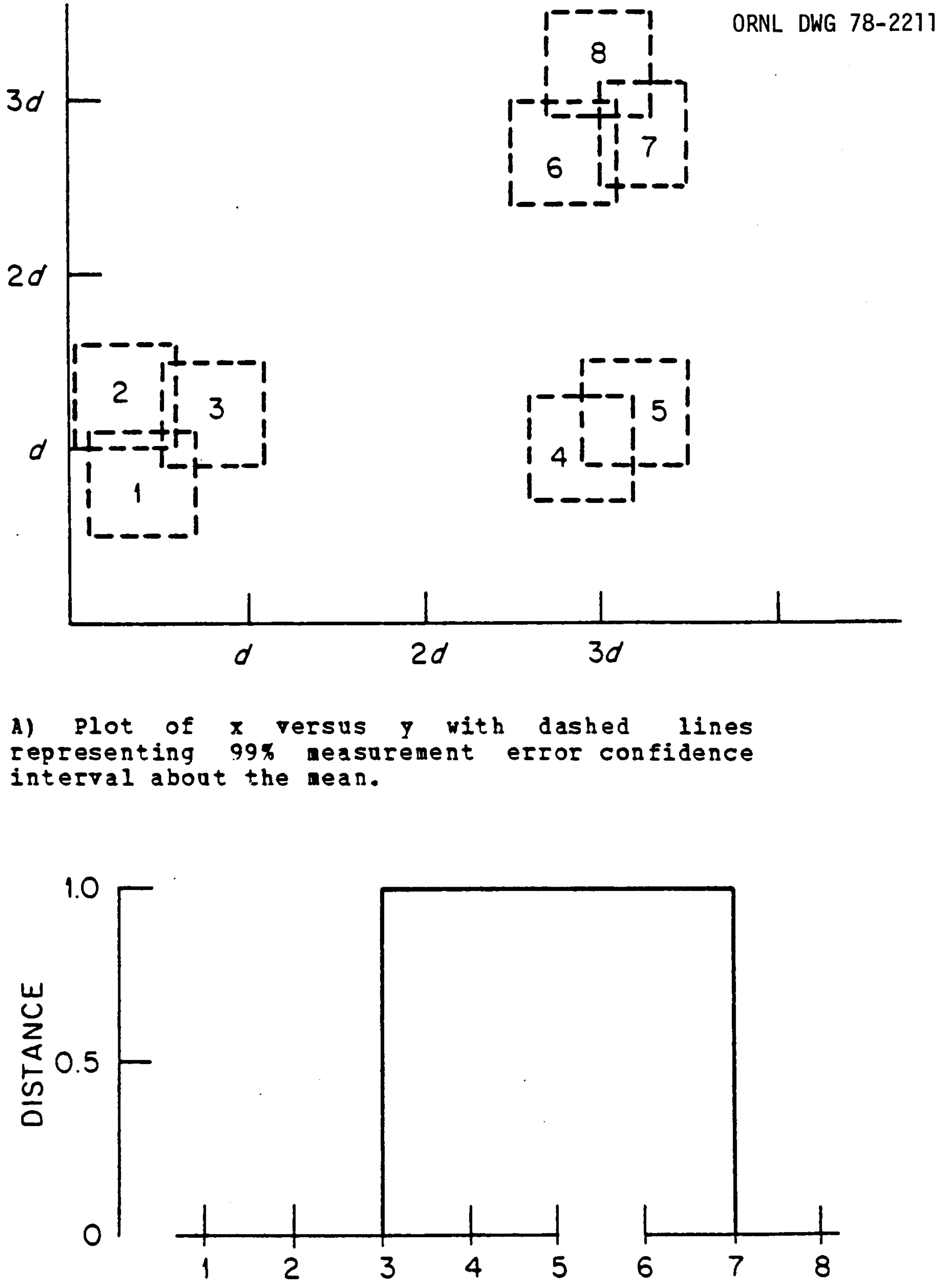
If $\underline{X}_j \epsilon \hat{G}_k$ for some k, the calculation of $P(\underline{X}_j \epsilon \hat{G}_k)$ includes a(j,j) in the summation over all samples in the group. Alternatively, sample j could be dropped as a member of any group estimate when calculating $P(\underline{X}_j \epsilon \hat{G}_k)$. Any difference in the estimate of the probability is small since $\underline{X}_j \epsilon G_k$ implies that a(j,j')/m is close to one. Therefore, including a(j,j)/m = 1 in the average has little effect. Equation (17) was used in the examples.

The number of groups and the group membership have been estimated for one value of c. The prediction results from the definition of groups: A group is a set of observations where $\underline{X}_j \epsilon G_k$ implies that there exists a $\underline{X}_{j'} \epsilon G_k$ such that the probability that samples j and j' are in the same cluster is greater than or equal to θ. The number of groups is estimated using this definition by determining the groups $\hat{G}_k$ from Monte Carlo clustering of the data into c clusters. When an increase in c does not affect the groups formed, the groups are stable and additional splitting is dependent on experimental error.

3.2 A Simple Example

Consider the eight samples as shown in Fig. 1A). A step-by-step analysis of this data set using the PMG procedure is described in this section. The data are defined as a set of points in two-dimensional space with a normal error distribution with mean 0 and standard deviation 0.1d. The dashed boxes in the figure represent the 99% confidence limit that each data point is within those boundaries.

The PMG is applied by perturbing the data within the error bounds m times. The m data sets are clustered into c groups; first consider c = 2. The optimum separation into two clusters is the combination of samples 1-5 in one group and samples 6-8 in a second group. Since even with error considerations, any pairwise distance between samples 1-5 is



A) Plot of x versus y with dashed lines representing 99% measurement error confidence interval about the mean.

B) Dendrogram of frequency measure for c = 2.

Figure 1. Example of eight two-dimensional observations.

less than 2d and the pairwise distances between samples 1-5 and 6-8 are greater than 2d, this clustering will be consistent for all m data sets. Therefore,

$$A = \begin{pmatrix} M_{5x5} & 0_{5x3} \\ 0_{3x5} & M_{3x3} \end{pmatrix} ,$$

where M_{ixj} and 0_{ixj} are i by j matrices with all elements equal to m and 0, respectively. For any $0 \leq a_o \leq m$,

$$\hat{G}_1 = \{\underline{x}_1, \underline{x}_2, \underline{x}_3, \underline{x}_4, \underline{x}_5\}, \text{ and}$$

$$\hat{G}_2 = \{\underline{x}_6, \underline{x}_7, \underline{x}_8\} \text{ with } g(c) = 2.$$

In addition,

$$P(\underline{X}_j \epsilon \hat{G}_1) = \begin{cases} 1 & j = 1,2,3,4,\text{or } 5 \\ 0 & j = 6,7,\text{or } 8 \end{cases} ,$$

$$P(\underline{X}_j \epsilon \hat{G}_2) = \begin{cases} 1 & j = 6,7,\text{or } 8 \\ 0 & j = 1,2,3,4,\text{or } 5 \end{cases} ,$$

A dendrogram of $d_{jj'}$ is shown in Fig. 1B).

Note that even though there are three distinct groups in the data in Fig. 1, g(2) = 2. Obviously, this is because the data were forced to separate into only two groups. However, if c is chosen to be three, the three groups 1-3, 4-5, and 6-8 will be clustered together to form

$$A = \begin{pmatrix} M_{3x3} & 0_{3x2} & 0_{3x3} \\ 0_{2x3} & M_{2x2} & 0_{2x3} \\ 0_{3x3} & 0_{3x2} & M_{3x3} \end{pmatrix} ,$$

and

$$\hat{G}_1 = \{\underline{X}_1, \underline{X}_2, \underline{X}_3\} ,$$

$$\hat{G}_2 = \{\underline{X}_4, \underline{X}_5\} ,$$

$$\hat{G}_3 = \{\underline{X}_6, \underline{X}_7, \underline{X}_8\} ,$$

with g(3) = 3 and

$$P(\underline{X}_j \epsilon \hat{G}_1) = \begin{cases} 1 & j = 1,2,\text{or } 3 \\ 0 & j = 4,5,6,7,\text{or } 8 \end{cases} ,$$

$$P(\underline{X}_j \epsilon \hat{G}_2) = \begin{cases} 1 & j = 4 \text{ or } 5 \\ 0 & j = 1,2,3,6,7,\text{or } 8 \end{cases} ,$$

$$P(\underline{X}_j \epsilon \hat{G}_3) = \begin{cases} 1 & j = 6,7,\text{or } 8 \\ 0 & j = 1,2,3,4,\text{or } 5 \end{cases} .$$

Now let c = 4. Since there are only three groups present in the data, a division of one of the groups will be necessary to form four clusters. The clusters formed will now depend on the individual sample error fluctuations. For example, if a perturbation of sample 8 is in the positive direction, and 6 and 7 are both perturbed negatively, the optimum group division would be 1-3, 4-5, 6-7, and 8. Because the clustering is affected by the error perturbations, the division of the three distinct groups is random and $g(4) = 3$. The frequency matrix would be equivalent to the matrix for $c = 3$, except the M submatrices would have elements of $m - \epsilon_i$ and the O submatrices would now have elements less than or equal to ϵ_i where ϵ_i is the number of times sample i was forced to separate from its group. For $\epsilon_i \leq a_0 \leq m - \epsilon_i$, the $\hat{G}$ are defined as:

$$\hat{G}_1 = \{\underline{x}_1, \underline{x}_2, \underline{x}_3\} \quad ,$$

$$\hat{G}_2 = \{\underline{x}_4, \underline{x}_5\} \quad ,$$

$$\hat{G}_3 = \{\underline{x}_6, \underline{x}_7, \underline{x}_8\} \quad .$$

A similar analysis to the above can be used to show that $g(5) = 3$ also. The number of groups $g = 3$ is estimated since $g(3) = g(4) = g(5)$. Application of the procedure on test data is in Chapter 4.

3.3 The PMG Algorithm

The basic steps in the PMG algorithm are displayed in Fig. 2. This section briefly describes the steps used in implementing the PMG procedure. Flowcharts of the main routines appear in Appendix A.

Each sample $\underline{X}_j$ is perturbed by $\underline{E}_{j\alpha}$ to form $\underline{Y}_{j\alpha}$ as in Eq. (10). To determine $\underline{E}_{j\alpha}$, a distribution for the experimental error is estimated or assumed using knowledge of the measurement error of the cluster analysis variables. Three common families of distribution for the error are multivariate normal or Gaussian distribution, multivariate

ORNL DWG 78-6924

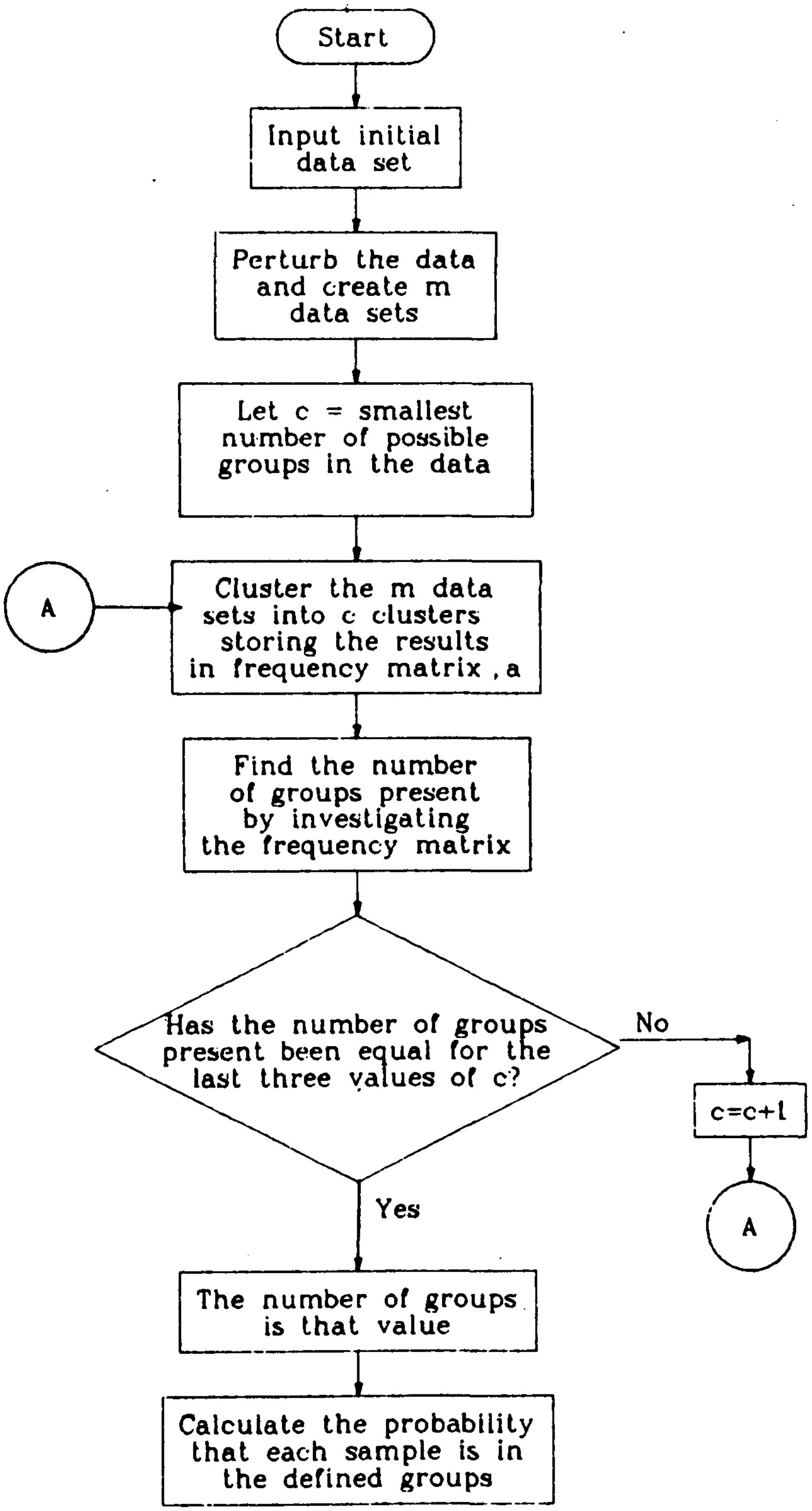

Start
Input initial data set
Perturb the data and create m data sets
Let c = smallest number of possible groups in the data
A
Cluster the m data sets into c clusters storing the results in frequency matrix , a
Find the number of groups present by investigating the frequency matrix
Has the number of groups present been equal for the last three values of c?
No
c=c+1
A
Yes
The number of groups is that value
Calculate the probability that each sample is in the defined groups

Figure 2. Basic steps involved in the PMG procedure.

truncated normal distribution, and multivariate uniform distribution. The normal distribution is characterized by a mean vector and covariance matrix. The truncated normal distribution also requires an upper and lower bound, and the uniform distribution necessitates only a range. All three distributions are implemented in the PMG procedure.

The algorithm used for either of the normal distributions is

$$Y_{ij} = X_{ij} + \xi_i , \tag{19}$$

or

$$Y_{ij} = X_{ij} + v_i X_{ij} \xi_i \quad , \tag{20}$$

where ξ_i is a random variable with mean μ and covariance matrix Σ , and v_i is the coefficient of variation of i-th variable. The transformation matrix used to generate normally distributed numbers is described by Bryan and Tebbe [1970]. The upper and lower limits of ξ_i are set if the truncated normal distribution is used. The uniformly distributed error is computed by generating uniform random numbers from a specified range.

Simulating the three distributions requires a uniform random number generator. Before choosing a method to generate the uniform randon numbers, a set of locally available generators was tested. Testing any random number generator before using it in a Monte Carlo simulation is important to prevent bias of the results [Halton, 1970]. A FORTRAN program written to test the generator, and the results of the testing, are described in Appendix B. Based on these results, a congruential uniform random number generator, URAND [McRae, 1970], was chosen; and a previously tested algorithm KR [Kinderman and Ramage, 1976] was used to transform the uniform numbers to a normally distributed set.

Each data set, Y_α, is clustered into c clusters as the next step in the PMG procedure. Any algorithm can be selected to use in the clustering. Before clustering, the Y_α can be transformed (using logarithm, square root, etc.) and standardized (e.g., divide by range or standard deviation).

The PMG procedure described here incorporates a hierarchical clustering program, DENDRO [Larson et al., 1976] because of its diversity in clustering algorithms, ease of determining the clustering at a specified range of groups, and speed of execution. The PMG procedure also includes options for transforming the data. The standardization options available in DENDRO are available in this procedure.

The computer code which is executed as the first step in the PMG procedure is called PMGPER. This routine includes the algorithm for generating the m data sets, preparing each one for clustering, and then clustering them and storing the results in the matrix A. This matrix is output for use in the next step of the PMG procedure.

Each A(j,j') can be transformed into the frequency measure between samples j and j' using Eq. (12). The program used to cluster the samples using this measure is also an altered form of DENDRO [Larson et al., 1976] called PMGCLS. All of the hierarchical methods described in Chapter 2 are available for clustering; however, since $d_{jj'}$ is a correlation-like measure, the methods which average or sum over the matrix are not realistic.

The frequency measure is already a distance measure, so no standardization or weighting is necessary. The arc-sin normalization can be performed on $d_{jj'}$ to normalize the distribution [Brownlee, 1960]:

$$d^{*}_{jj'} = \sin^{-1}(d_{jj'}) \quad . \tag{21}$$

Clustering by using $d^{*}_{jj'}$ instead of $d_{jj'}$ provided little difference in the examples investigated. Both dendrograms can be output by PMGCLS.

The PMG procedure is implemented to find $\hat{G}_k$, k= 1, 2, . . . , g(c) and to estimate $P(\underline{X}_j \epsilon \hat{G}_k)$. First, a search is performed to find all sample pairs j and j' such that $A(j,j') \geq a_0$. Any set of samples where $A(j,j') \geq a_0$, $A(i,j) \geq a_0$, and $A(i,j') \geq a_0$ are combined. After all these sets are formed, any two sets which have the same sample are combined to form the $\hat{G}_k$ in Eq. (15). The probability that each sample belongs to $\hat{G}_k$ is computed from Eq. (17) and the basic algorithm is essentially complete for one value of c. An interactive program PMGEST implements this part of the procedure.

Additional consideration is necessary to find outliers or one member groups occurring in the data. Since the frequency matrix stores only the occurrence of sample pairs, outliers will not appear as a group. However, if a sample $\underline{X}_j$ is a single member cluster at least a_0 times in the clustering, it should be defined as a single member $\hat{G}$. Therefore, sample $\underline{X}_j$ is considered a single member cluster if $A(j,j') < m - a_0$ for all j'.

Dendrograms of the frequency measure and determinations of g(c) are output for increasing value of c. When g(c) = g(c+1) = g(c+2) then the number of groups is estimated as g(c). The probabilities of sample membership are then those defined by $P(\underline{X}_j \epsilon \hat{G}_k)$ when $\hat{G}_k$ are the g(c) groups estimated from Eq. (15).

The procedure recommended to find g(c) is to compute a_0 for θ = .9, .85, .7, .75 for significance levels of 10%, 1%, and .1%. The binomial formula, Eq. (14), is used to determine a_0 or the normal approximation is useful for large values of m [Lingren, 1976]:

$$\lambda = \Pr[A(j,j') \geq a_0] = 1 - \Phi\left(\frac{a_0 + 1/2 - m\theta}{\sqrt{m\theta(1-\theta)}}\right) \quad . \qquad (22)$$

where Φ is the cumulative normal distribution. If for any combination of the significance levels and one value of θ,

$g(c) = g(c+1) = g(c+2)$, the number of groups estimated is $g(c)$.

The number of times each sample is forced to split from its group will increase as c increases above g. In addition, it is possible that no value of c will be found where $g(c) = g(c+1) = g(c+2)$ (see Section 4.2). This indicates that either the error perturbations are as large as any cluster separation (no groups exist) or a hierarchical tree structure might better represent the data. An analysis for varying values of a_0 can still aid the user in determining group separations and group definitions at different probability levels θ.

A plot of the probabilities for each sample is useful for analyzing the data. The very distinct clusters are evident, as are samples which are almost equally likely to be members of two or more groups. Examples of these plots appear in Chapter 4.

The probabilities defined by the PMG procedure might vary from the groups suggested by clustering the frequency measure. Any differences are due to the algorithms used to calculate the distance between two clusters in clustering the frequency measure. In addition, not all $a(j,j')$ pairs are used in the determination of sample membership using the PMG procedure. Sample j and/or sample j' must be a member of a $\hat{G}_k$ for $a(j,j')$ to be used (Eq. (17)). The probabilities of group membership determined by the PMG procedure should be and have been (see Section 4.2) better estimates.

CHAPTER 4

COMPARISON OF GROUPING METHODS

The PMG procedure is compared to the methods described in Section 2.3. The first data set discussed consists of 5 distinct groups each having 10 samples. The second example uses 10 data sets generated from 2 normally distributed populations. Finally, the methods are applied to geochemical data [Kane and Larson, 1976] as an example of the application to real data.

4.1 Five Well-Separated Groups

The first example of 50 samples (Fig. 3) was selected to illustrate a set of obviously distinct two-dimensional groups. If grouping procedures do not estimate five groups for this data, it is not likely that they will perform well cn more complex data.

The dendrograms for each hierarchical method are shown in Fig. 4. Clearly, the five groups are indicated by each method. A large separation between the groups is shown by the longer lines adjoining each of the five separate groups.

A plot of $\hat{g}$ versus the optimization value using W and B and separating the data into $\hat{g}$ groups is displayed in Fig. 5 for the four methods in MICKA. Both the trace W and the determinant W criteria show the smallest percentage change in values from $\hat{g}$ = 5 to $\hat{g}$ = 6 and 7. Neither of the criterion, however, reached an absolute minimum. The

ORNL DWG 78-1432

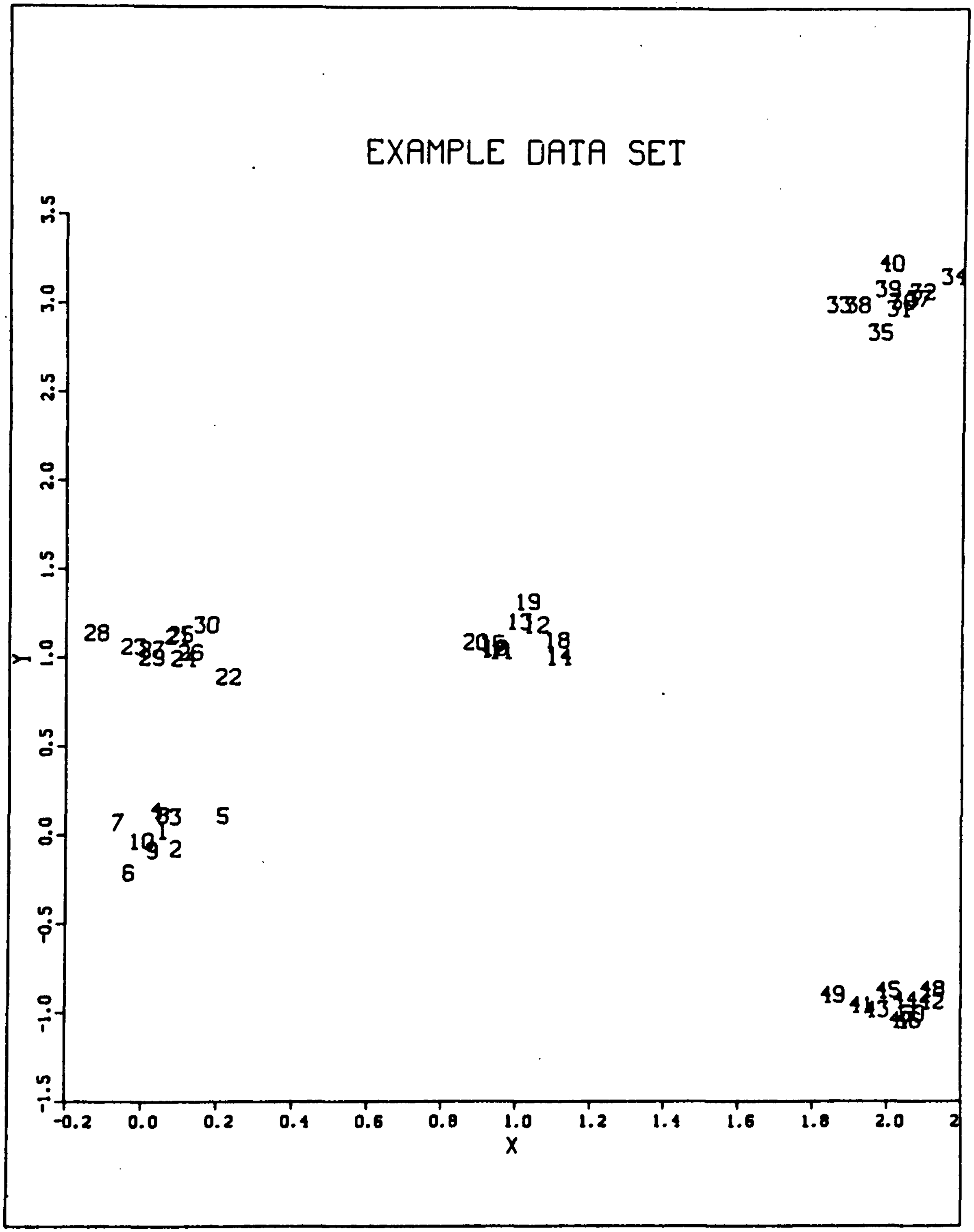
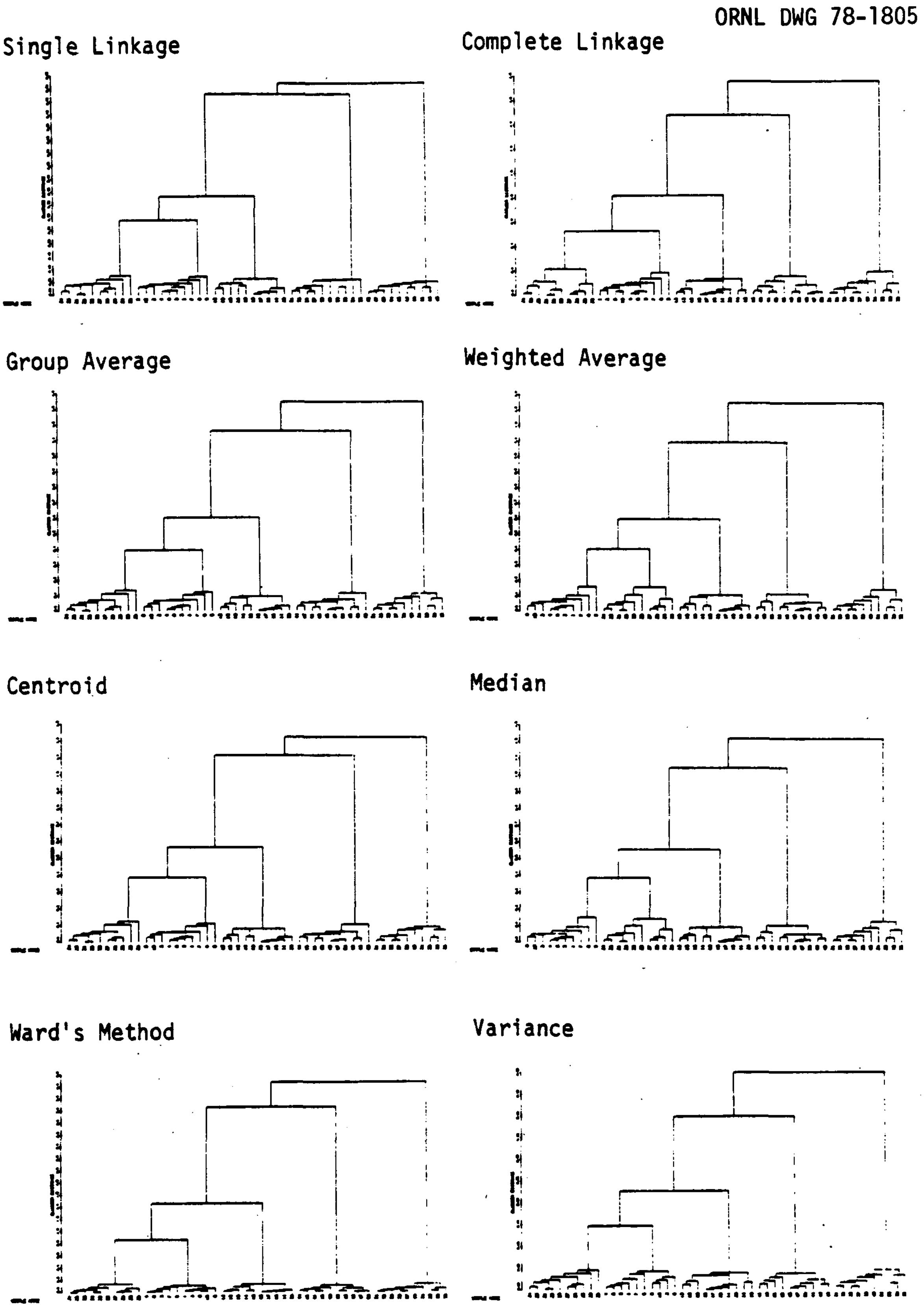


Figure 3. Example 1: 50 samples which form 5 groups.



ORNL DWG 78-1805

Single Linkage

Complete Linkage

Group Average

Weighted Average

Centroid

Median

Ward's Method

Variance

Figure 4. Dendrograms for 10 hierarchical methods for Example 1.

ORNL DWG 78-2213

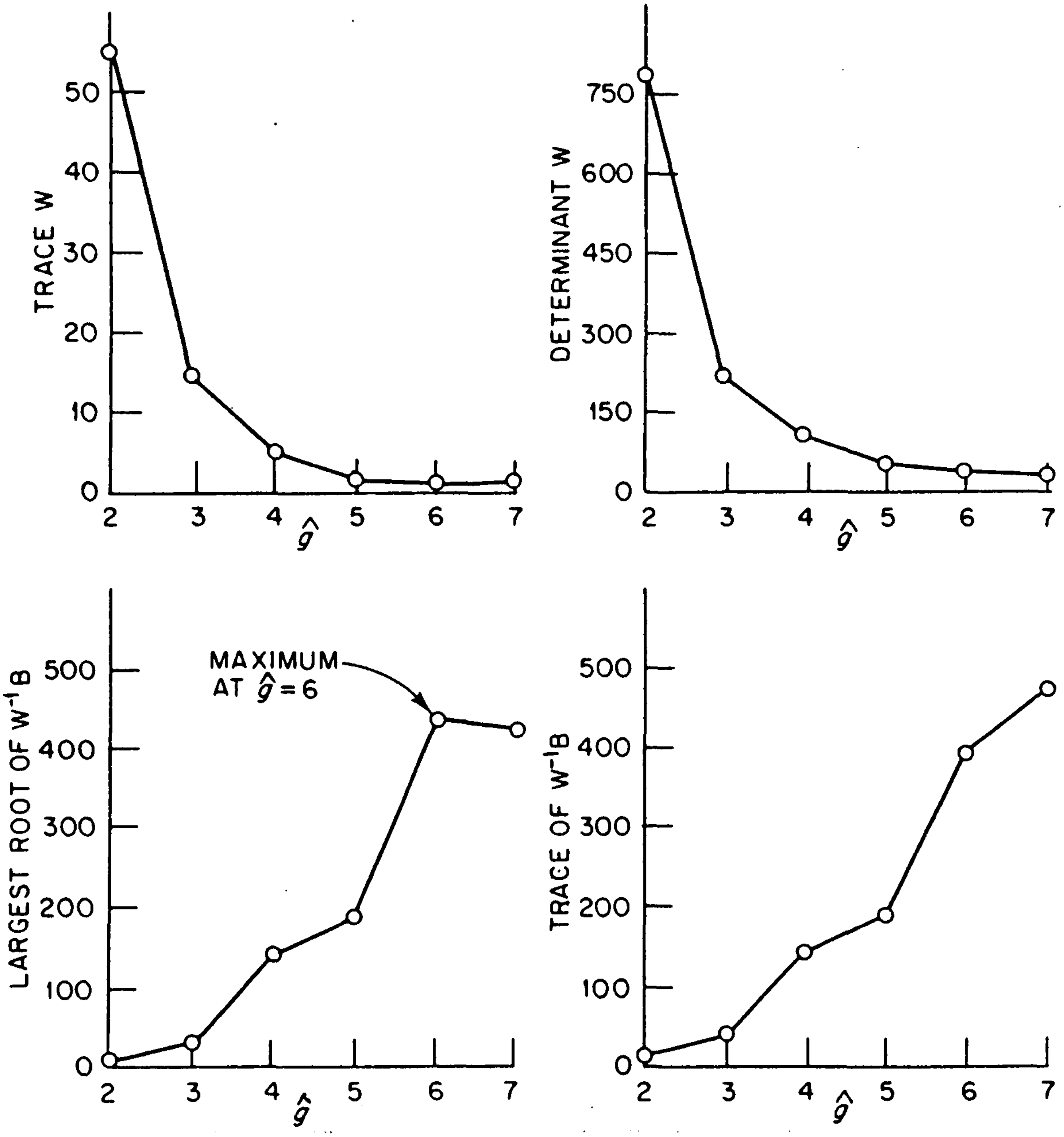


Figure 5. Plot of $\hat{g}$ versus optimization values using dispersion matrices criteria for Example 1.

largest root of $W^{-1}B$ has a maximum value at $\hat{g} = 6$. The trace of $W^{-1}B$ does not show a significant maximum in the range investigated. Trace W and determinant W can be used to estimate $\hat{g} = 5$, but the other two criterion give unsatisfactory results.

The maximum likelihood estimator, NORMIX, performed well on this test. A plot of the log likelihood is in Fig. 6. A maximum was reached for $\hat{g} = 5$. The probability for the null hypothesis of $\hat{g}$ versus $\hat{g}+1$ groups was less than 0.01 for $\hat{g} = 1$ to 4. However, for $\hat{g} = 5$ the probability of the null hypothesis was 86%, in which case H: $\hat{g} = 5$ is not rejected. NORMIX predicts the correct values of $\hat{g} = 5$.

For the ISODATA program, the number of groups output by the method was the same as the input number desired. Fig. 7 shows the trace of W for the groups determined by ISODATA. This figure is similar to the plot produced by MICKA's trace W criterion. The plot can be used to estimate $\hat{g} = 5$.

Results of the mathematical indicators are shown in Fig. 8. Using Beale's F statistic, the probability that $\hat{g} = 5$ and not 4 is 1.0; whereas, the probability that $\hat{g} = 6$ and not 5 is not significant at 0.45. The Calinski and Harabasz Kg is at a maximum for $\hat{g} = 5$, correctly suggesting five groups. Marriot's $g^2(\det W)$ is at a maximum for $\hat{g} = 6$; it is the only one of these indicators that does not perform satisfactorily.

ORNL DWG 78-2210

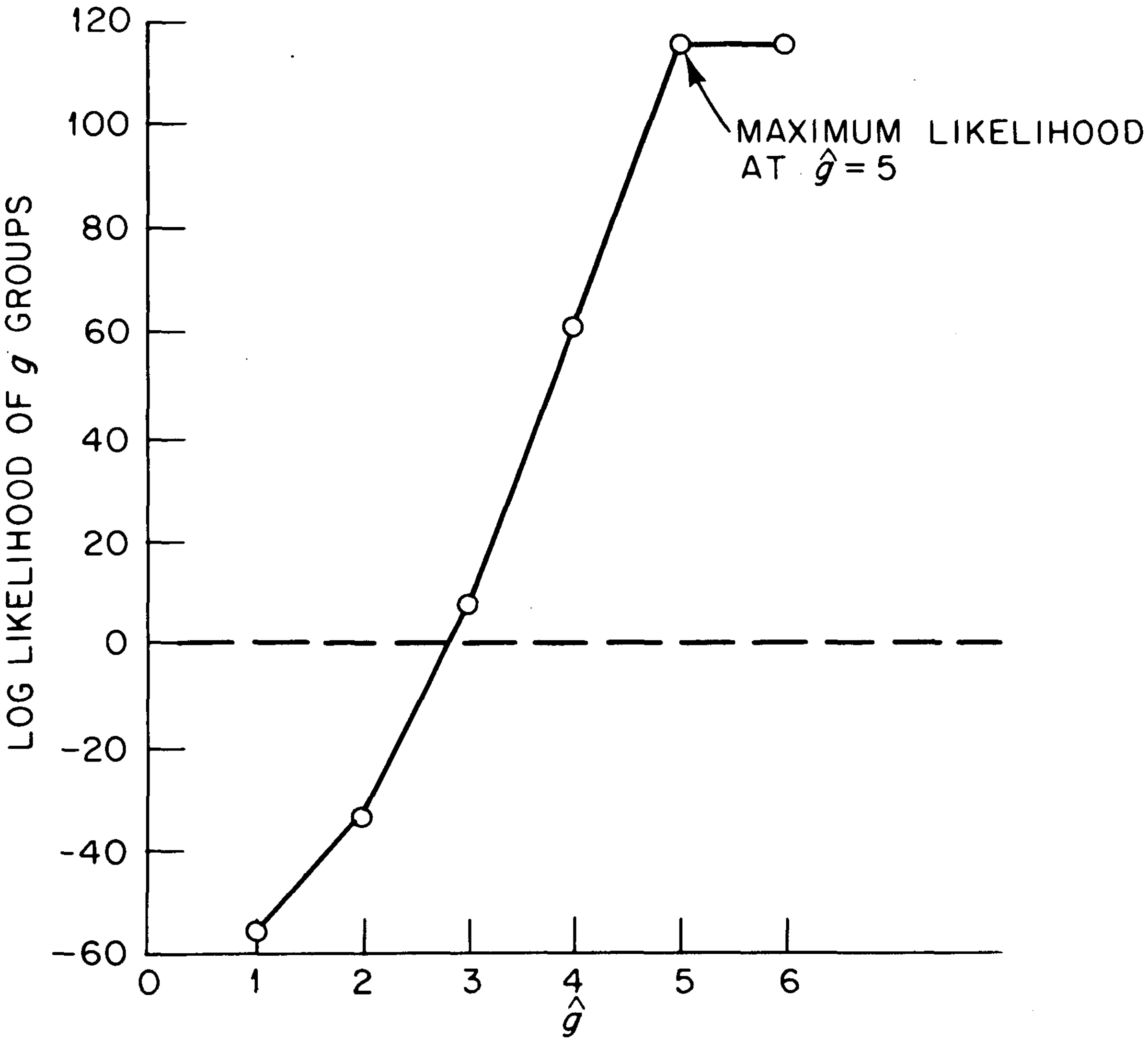


Figure 6. Maximum likelihood estimate for increasing values of $\hat{g}$ for Example 1.

ORNL DWG 78-2209

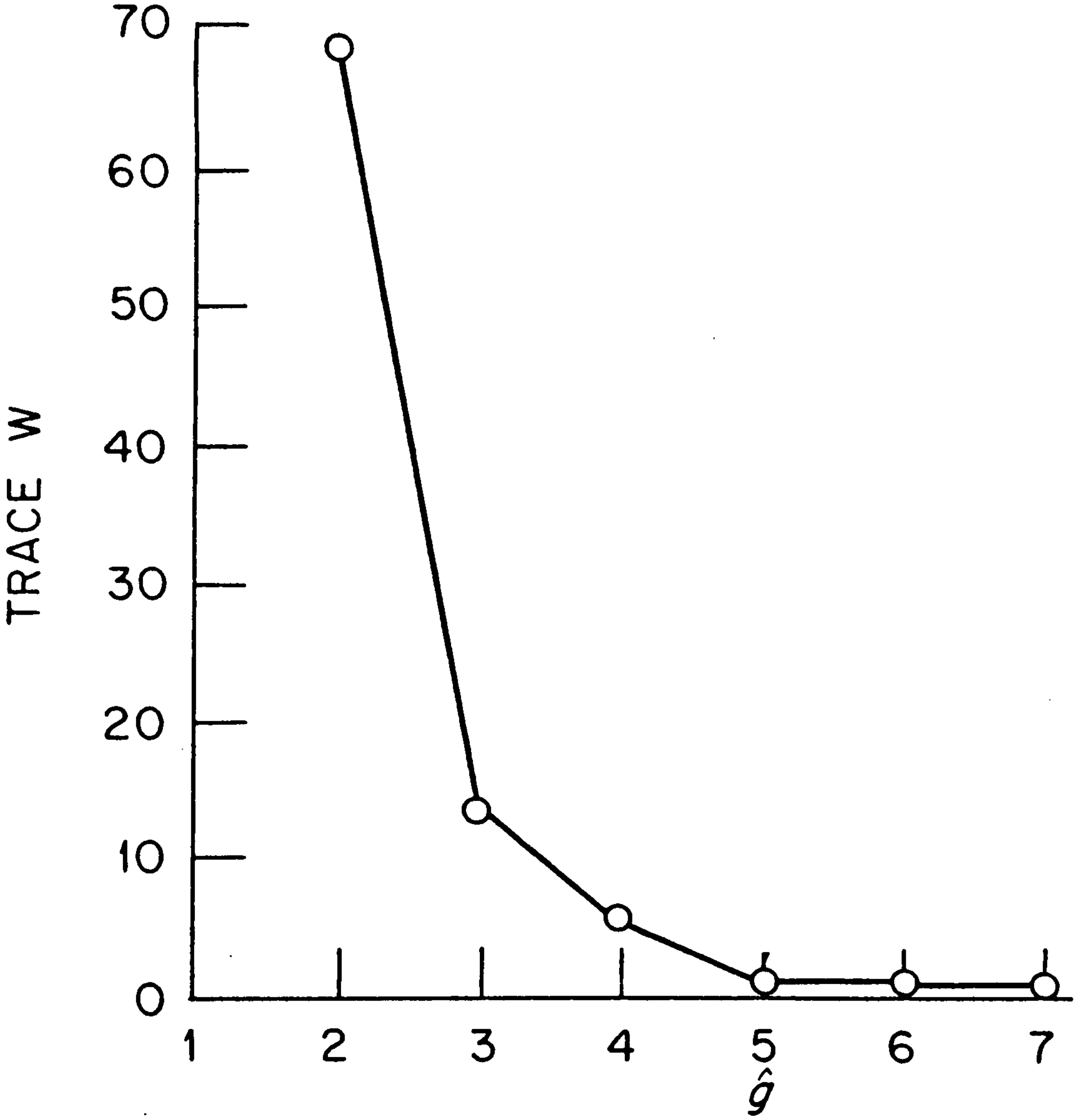


Figure 7. Plot of $\hat{g}$ versus trace W using ISODATA program.

ORNL DWG 78-2212

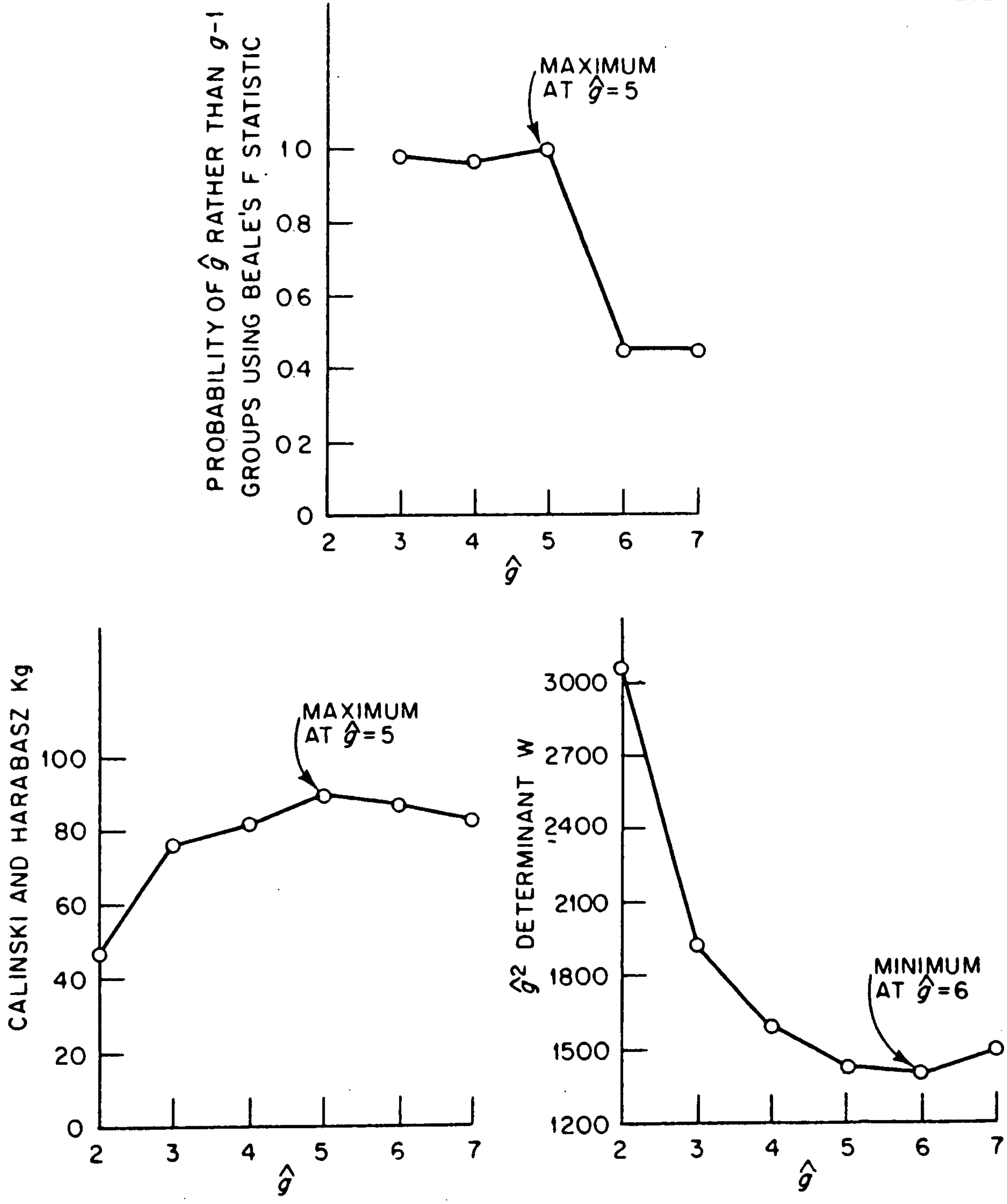


Figure 8. Results of Beale's F statistic, Calinski and Harabasz's Kg and Marriot's g^2(det W) for Example 1.

The PMG procedure was applied to this data set with the experimental error assumed to be normally distributed with mean 0 and standard deviation of 0.1. Twenty simulations were run using the minimum increase in variance hierarchical procedure as the clustering technique. The values of g(c) for c = 2 to 7 are in Table 2. The g(c) listed in the table is the value of g(c) for the probability of θ = .9 for significance levels 10%, 1%, and .1%. In this table g(c) = g(c+1) = g(c+2) for c = 5, and the method correctly predicts five groups. The probability of group membership at g(c) = 5 are shown in Fig. 9; the groups are delineated correctly. The dendrogram of the frequency measure for c = 5 is in Fig. 10. The dendrogram also indicates five groups.

For the first trivial example, the dendrograms for all methods, NORMIX, Beale's F statistic, the Calinski and Harabasz Kg, and the PMG method all clearly identify the five distinct groups. The trace W and determinant W criteria of MICKA marginally indicate the five groups, as does the ISODATA program. The trace of $W^{-1}B$ does not indicate any number of groups, and both the root of $W^{-1}B$ and the g^2(det W) criterion predict the wrong value of $\hat{g}$ = 6.

4.2 Normally Distributed Data

Ten different data sets were generated from a normal distribution using means of (0,0) and (2,2) and a covariance matrix of 0.9I and normally distributed error with mean (0,0) and covariance matrix 0.1I. The sample

Table 2. Results of the PMG procedure for Example 1

Number of clusters, c	Values of g(c) for θ = .9 λ =10.%	λ=1.0%	λ=0.1%
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
6	5	5	5
7	5	5	5

ORNL DWG 78-6708

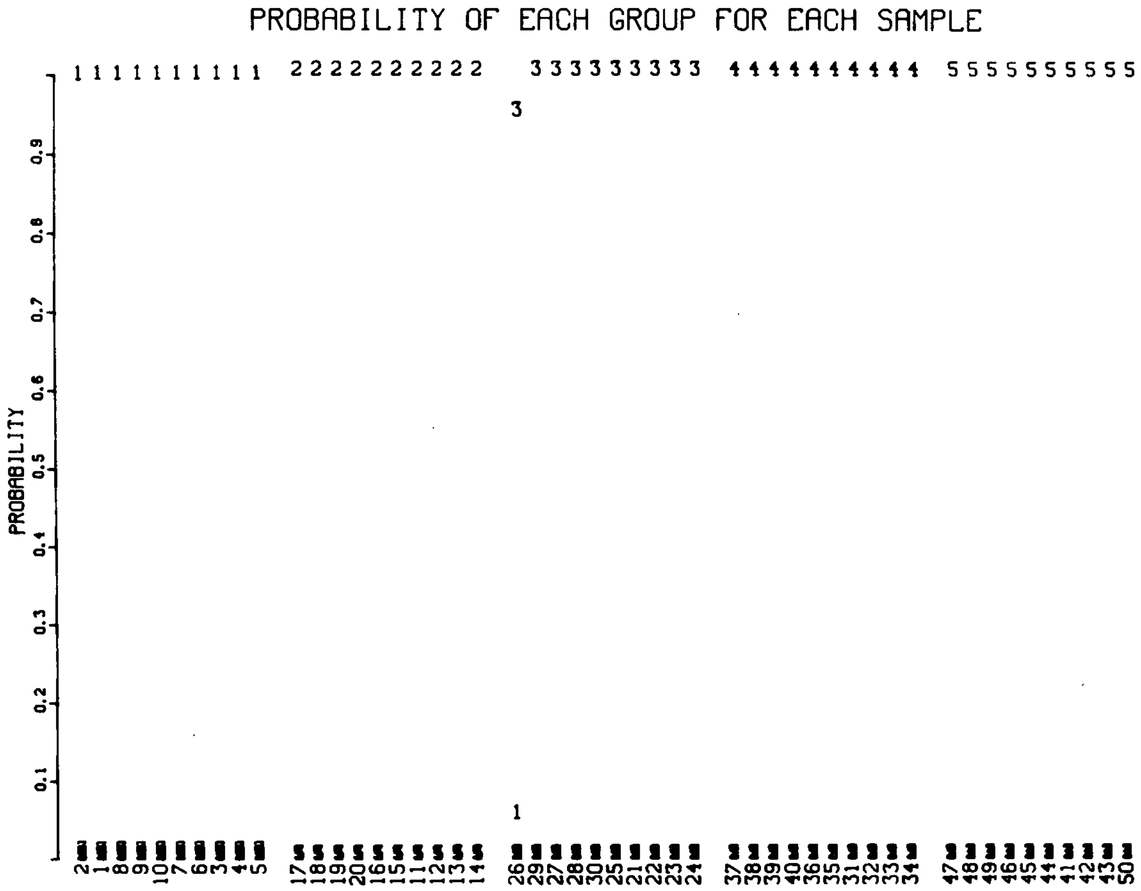


Figure 9. Probabilities of group membership for each sample using the PMG procedure for Example 1.

ORNL DWG 78-1433

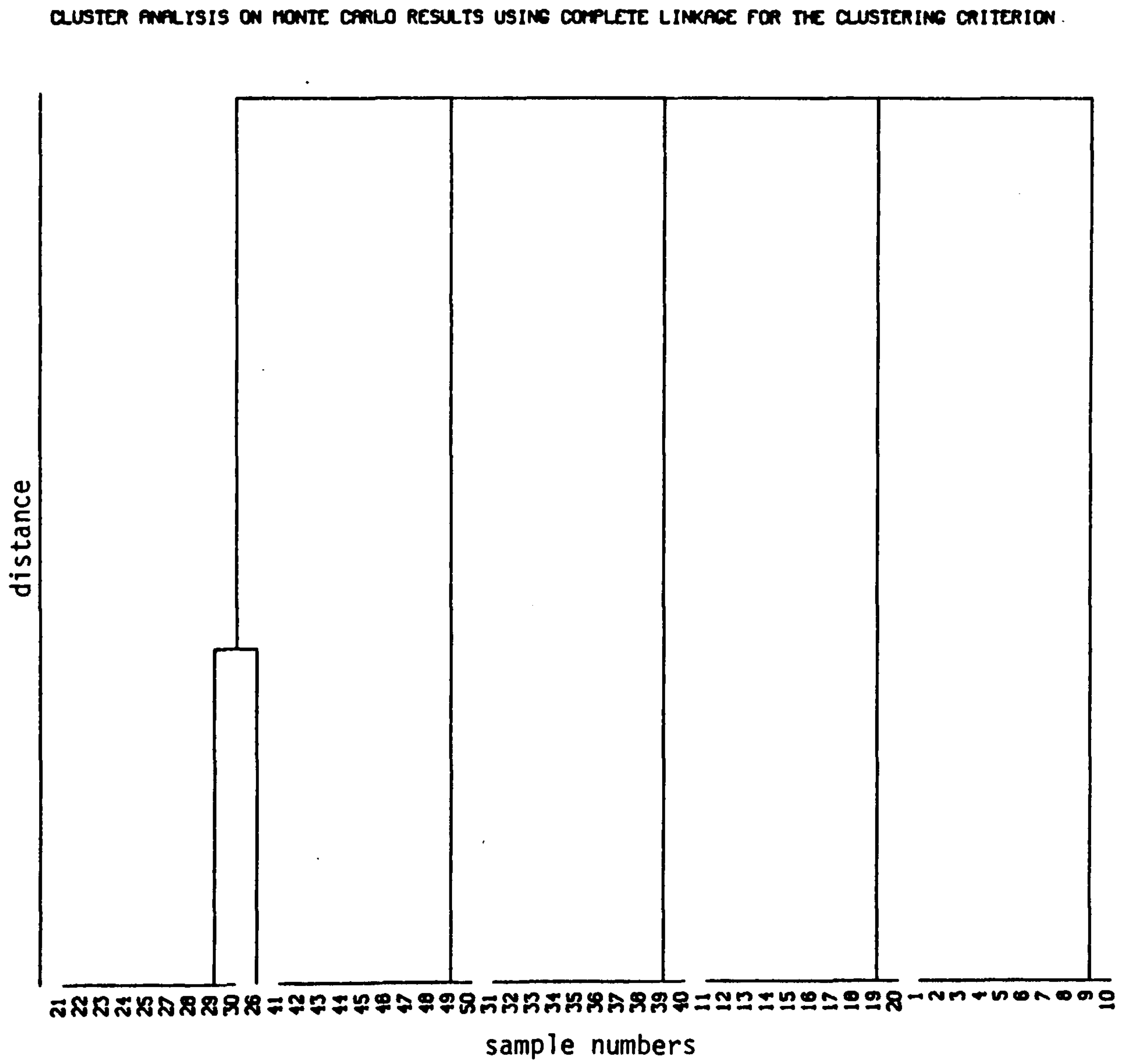


Figure 10. Dendrogram of frequency measure for Example 1. Complete linkage is the clustering criteria.

mean, standard deviation, and Shapiro-Wilk statistic [Shapiro and Wilk, 1965] for each data set, Y, is given in Table 3. The data form overlapping groups; the theoretical misclassification is 7.9% [Bayne et al., 1978]. Ten data sets were selected to allow a more general test of the methods.

The purpose of this example is not only to compare group estimation but also to compare misclassification of the PMG procedure. All of the procedures were applied to Y. Only the PMG method incorporates the consideration of the error (E).

Dendrograms for the 10 data sets using the variance method as the clustering criterion appear in Fig. 11. Data sets 1, 2, 3, and 5 are subjectively separated into three groups. The rest of the sets appear to consist of two groups. The variance criterion was chosen above the others because of its theoretical appeal and its classification ability [Bayne et al., 1978].

All four optimization methods using dispersion matrices were tested on the 10 sets; results are in Fig. 12. The values of each optimization criterion do not show any indication of the number of groups in any of the 10 data sets. The first two criteria continue to decrease for g = 2, 3, 4; whereas, the last two increase monotonically. None of the optimization methods even marginally indicates the existence of two groups.

Table 3. Summary statistics of the 10 data sets used in Example 2

		Mean		Standard deviation		Shapiro-Wilk statistic	
Data		Variable		Variable		Variable	
Set	Class	1	2	1	2	1	2
1	1	0.12	0.10	1.28	1.20	0.61	0.09
	2	2.02	2.00	1.02	0.81	0.56	0.21
2	1	0.36	-0.12	1.20	1.02	0.58	0.29
	2	2.15	2.14	1.00	1.07	0.92	0.27
3	1	0.13	0.06	1.03	1.04	0.35	0.65
	2	2.11	2.07	1.10	0.89	0.31	0.67
4	1	-0.34	0.00	0.75	1.06	0.31	0.67
	2	1.80	2.54	0.78	0.85	0.08	0.39
5	1	-0.16	0.04	1.00	1.07	0.22	0.68
	2	1.77	2.14	0.90	0.90	0.21	0.46
6	1	-0.41	0.22	1.05	0.95	0.26	0.65
	2	2.01	2.17	0.94	0.95	0.42	0.20
7	1	-0.01	0.04	1.04	0.69	0.34	0.72
	2	2.04	1.91	0.82	1.17	0.62	0.85
8	1	-0.13	0.04	0.84	0.82	0.34	0.91
	2	2.07	2.16	0.86	1.06	0.70	0.57
9	1	-0.41	0.11	1.08	1.25	0.44	0.52
	2	2.00	1.25	1.09	0.90	0.24	0.42
10	1	0.06	-0.05	0.77	0.97	0.56	0.62
	2	1.93	1.88	0.95	0.90	0.99	0.75

ORNL DWG 78-1807

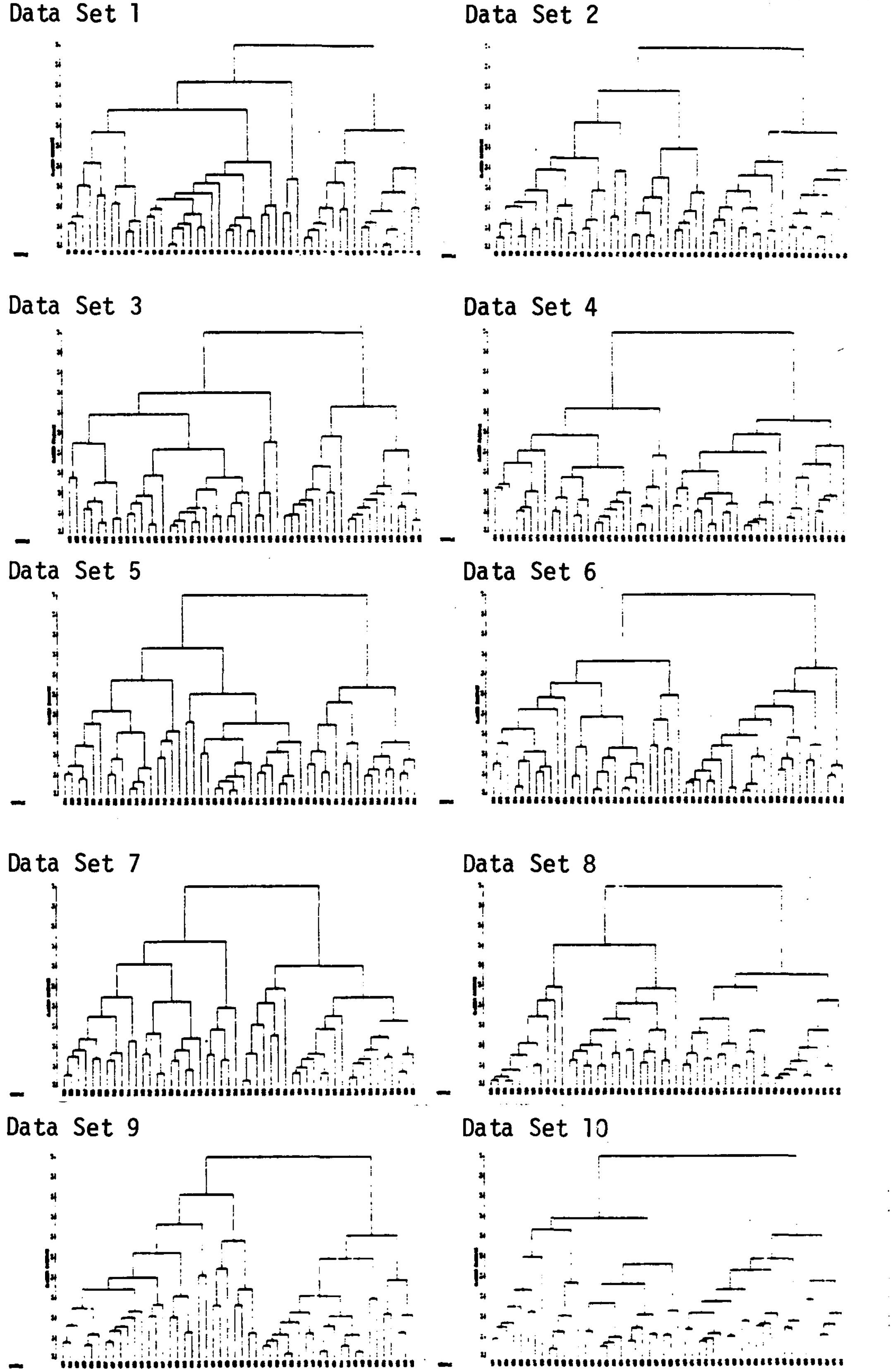


Figure 11. Dendrograms using variance criterion for Example 2.

ORNL DWG 78-2217

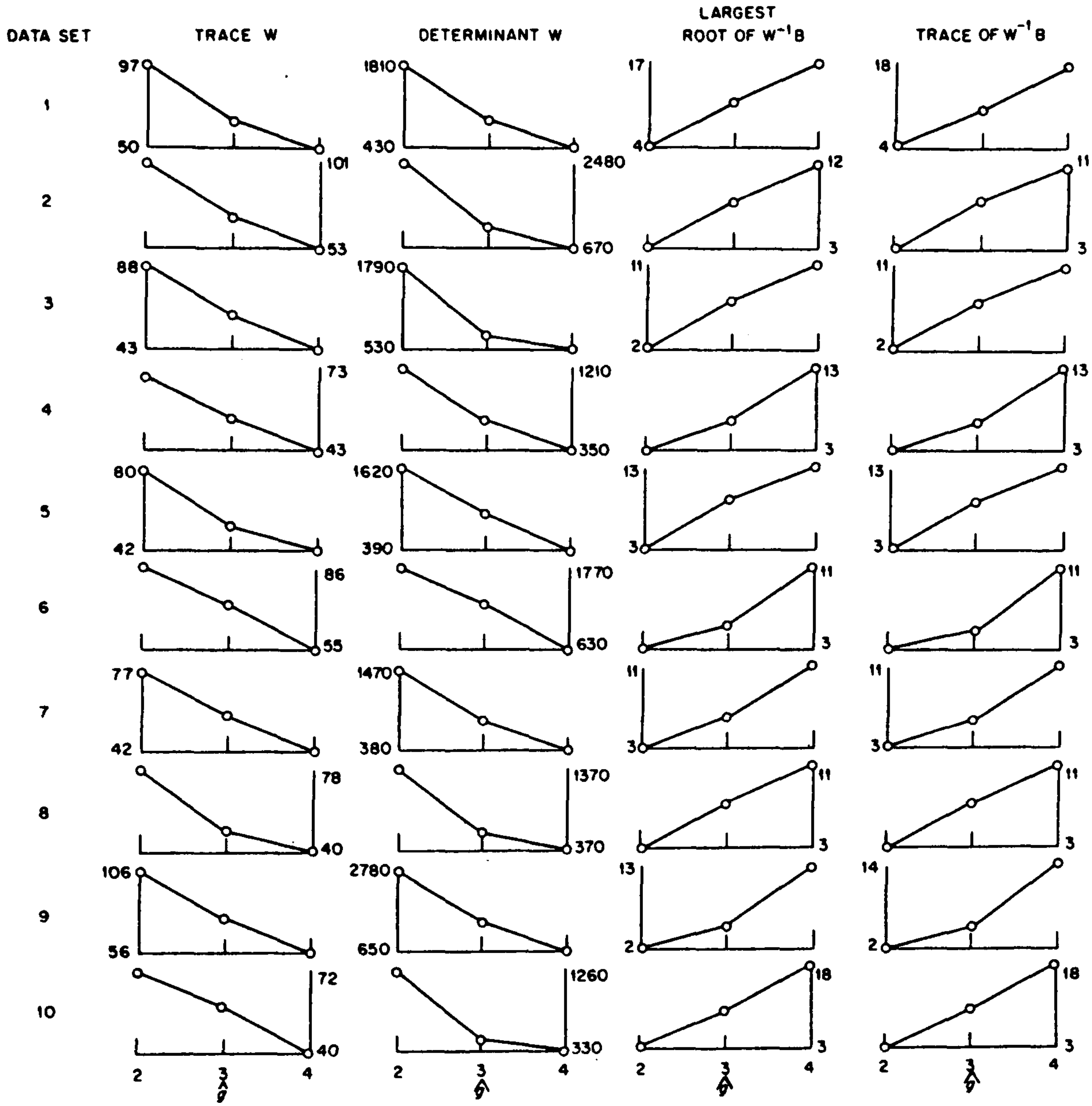


Figure 12. Plot of ĝ versus optimization values using dispersion matrices criterion for Example 2.

The maximum log likelihood for g = 2 and g = 3 ranged between -80 and -64 with the maximum always occurring at $\hat{g}$ = 3. The probabilities of the null hypothesis of $H_0:\hat{g} = 3$ versus the alternative $H:\hat{g} = 2$ groups is in Table 4. The hypothesis of two groups would normally not be rejected for all except data sets 1, 7, and 9. Therefore, the estimate for $\hat{g}$ is 70% correct for this example.

The ISODATA program was run on each of these sets. The number of groups desired was input as two, but in each case, the number of resultant groups output by ISODATA was four. The method always began with approximately the correct two groups and then split each of them into two groups. Varying parameters cause some differences, but no indication of two groups was suggested.

The results using the mathematical indicators are in Fig. 13; the predicted value of groups, $\hat{g}$, is indicated in each case. Beale's F statistic has the highest probability of two groups in 9 of the 10 data sets; in data set 1 there is a larger probability for $\hat{g}$ = 3. For many of the data sets, the probabilities that $\hat{g}$ = 2, $\hat{g}$ = 3, or $\hat{g}$ = 4 are not different by more than 0.2. The Calinski and Harabasz Kg miscalculates the number of groups twice; all except data sets 1 and 9 are found to have two groups. The g^2(det W) criterion is at a minimum for $\hat{g}$ = 1 in 6 of the 10 cases and predicts $\hat{g}$ = 2 twice, $\hat{g}$ = 3 and $\hat{g}$ = 4 once.

Table 4. Probabilities of the null hypothesis $\hat{g} = 3$ versus $\hat{g} = 2$ using the maximum likelihood estimator for Example 2

Data Set	Probability of null hypothesis	Likelihood estimate at $\hat{g}=2$	Likelihood estimate at $\hat{g}=3$
1	0.97	-75.4	-69.6
2	0.10	-80.7	-79.4
3	0.03	-65.0	-64.7
4	0.33	-67.6	-66.3
5	0.00	-70.6	-70.6
6	0.00	-74.0	-74.0
7	0.73	-68.1	-65.3
8	0.37	-67.8	-66.4
9	0.52	-80.4	-78.5
10	0.10	-64.5	-63.9

ORNL DWG 78-2218

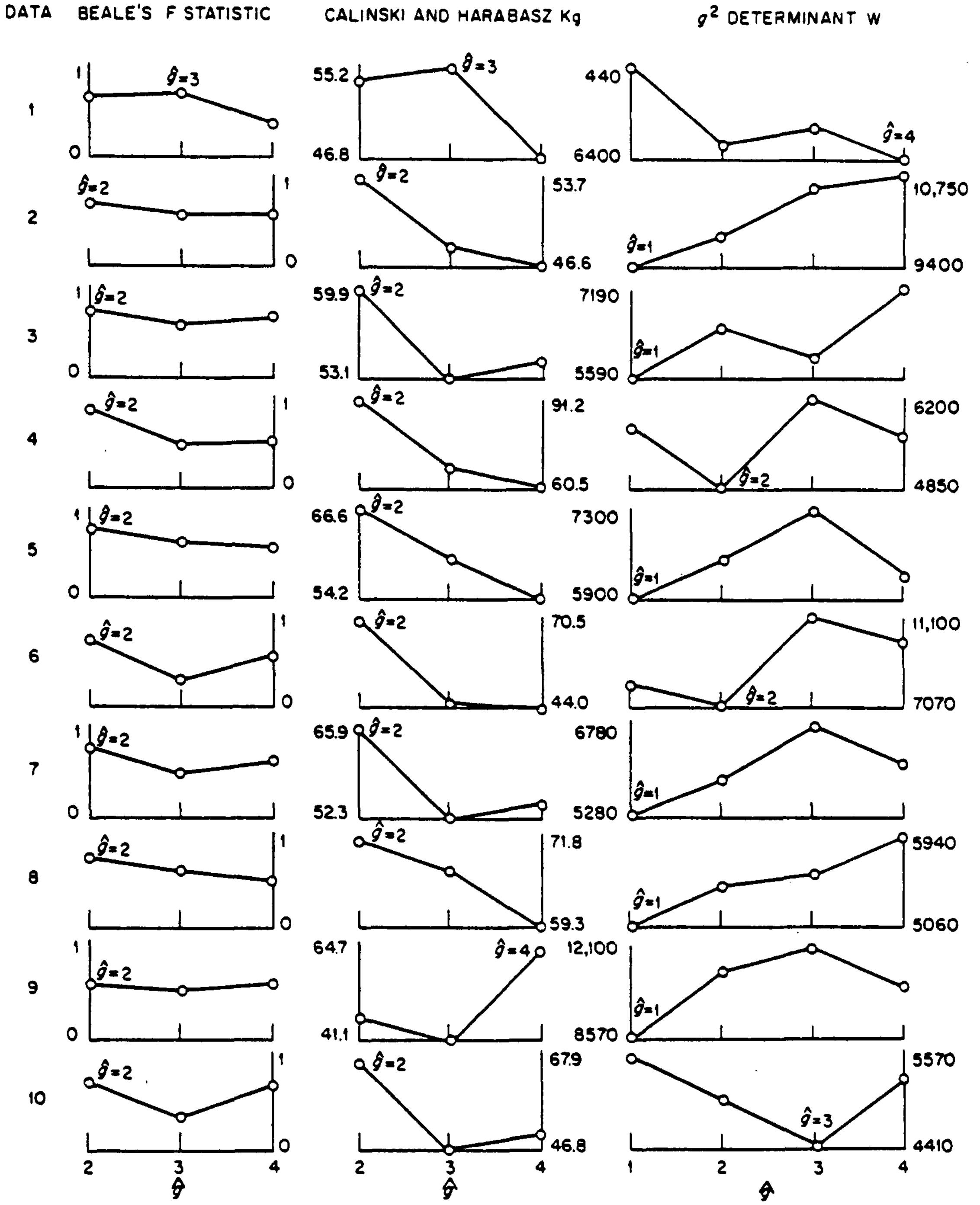


Figure 13. Results of Beale's F statistic, Calinski and Harabasz's Kg, and Marriot's g²(det W) for Example 2. The value of $\hat{g}$ shown is the number of groups estimated by the procedure.

To test the PMG procedure the value of c was set to two, three, and four. Twenty-five iterations were run using the defined error matrix. The variance criterion was again used in the clustering.

A value of g(c) was computed for c at θ = .9, .85, .8, and .75 and λ = 10%, 1%, and .1%. If g(c) was the same for c = 2, 3, and 4, for any significance level at one probability level, then $\hat{g}$ was set equal to g(c). The value of g(c) and the probabilities θ, where g(c) = g(c+1) = g(c+2), are listed in Table 5. A value of $\hat{g}$ = 2 is predicted for 60% of the cases, and $\hat{g}$ = 1 is predicted for the other 40%. Dendrograms of the frequency measure are in Fig. 14. Dividing the dendrograms into the number of groups joined at the level where $d^*_{JJ'} = 0$, there are 2 groups in all 10 cases. The groups are not as distinct as the dendrogram in Example 1.

The properties of the data sets that caused the PMG procedure to miscalculate the number of groups are not apparent from Table 3 or even from the dendrograms in Fig. 11. Fig. 15, however, indicates the difference between two of the data sets, 4 and 7. Data set 4 has a clear division between the two groups represented by the two symbols in the figure; however, data set 7 does not. The PMG procedure estimated only one group, because of the lack of any clear cut division between the two groups or any two groups. A plot of the probabilities of group membership for the

Table 5. Values of $\hat{g}$ estimated by the PMG procedure for Example 2

Data set	Number of groups estimated	Probability, θ[1] used for estimate
1	2	0.85
2	2	0.80
3	-	----[2]
4	2	0.85
5	2	0.85
6	2	0.80
7	1	C.85
8	1	0.80
9	1	0.85
10	1	0.80

[1] θ= .9, .85, and .8 was tested for λ = 10., 1.0, 0.1.

[2] g(c) did not remain constant for three consecutive values of c for any tested value of θ.

ORNL DWG 78-1806

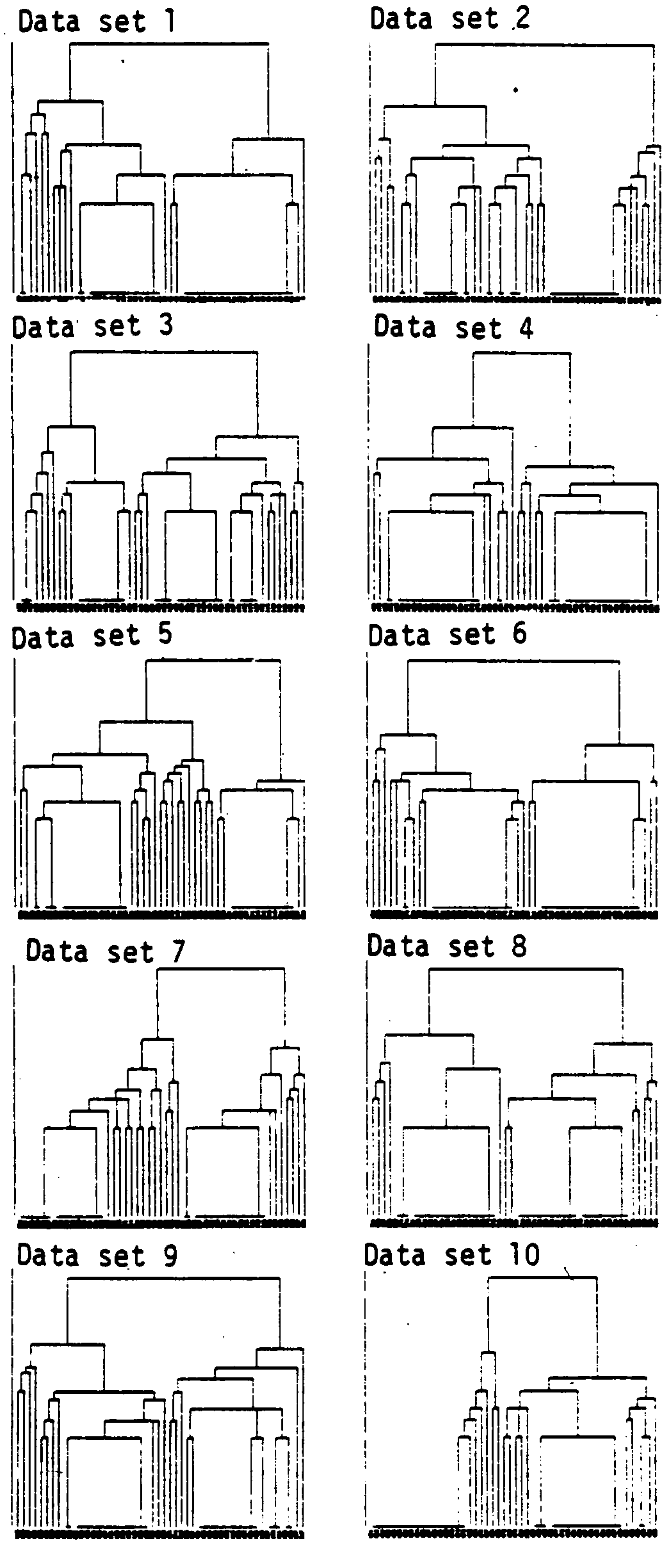


Figure 14. Dendrogram of frequency measure for Example 2. Complete linkage is the clustering criterion used.

ORNL DWG 78-6705

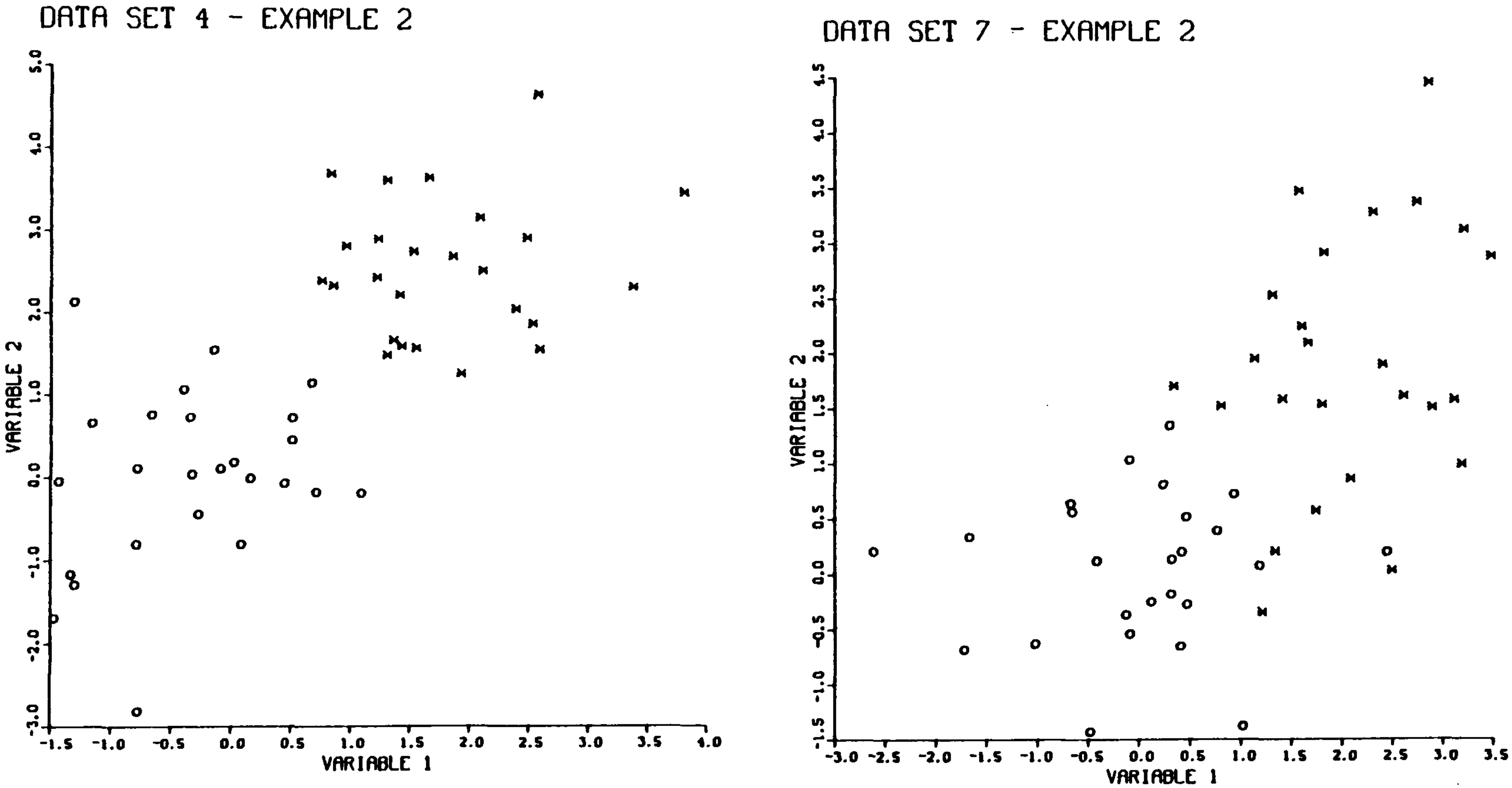


Figure 15. Two-dimensional plots of Example 2, data sets 4 and 7.

samples when $g(c) = 2$ for $a_0 = 20$ is in Fig. 16 for data set 7. There is a large number of samples which have a probability of being in both groups. When a lower value of a_0 is used, these samples link the two groups into one.

A comparison of misclassification for the PMG method with the regular variance hierarchical method and with the linear discriminant function is in Table 6. The linear discriminant function (LDF) is the theoretical best division of data into groups given the groups' means and covariance matrices. The number misclassified by the LDF is calculated using Eq. (4.4-16) in Tou and Gonzalez [1974] for the error distorted data sets, Y. The number misclassified by the variance method is calculated for the error distorted set plus a collection of 10 data sets generated from means (0,0) and (2,2) and covariance matrix of .9I. The number misclassified by both the frequency measure and the PMG procedure is given.

The PMG method improves the classification ability of the variance criteria. The improvement is better than the 10 data sets generated from an error-free distribution. The PMG misclassification is almost as small as the theoretical misclassification of the perturbed data. That is, in this particular example, using the error improves the classification ability almost as much as knowing all the distribution parameters of the data.

ORNL DWG 78-6706

PROBABILITY OF EACH GROUP FOR EACH SAMPLE

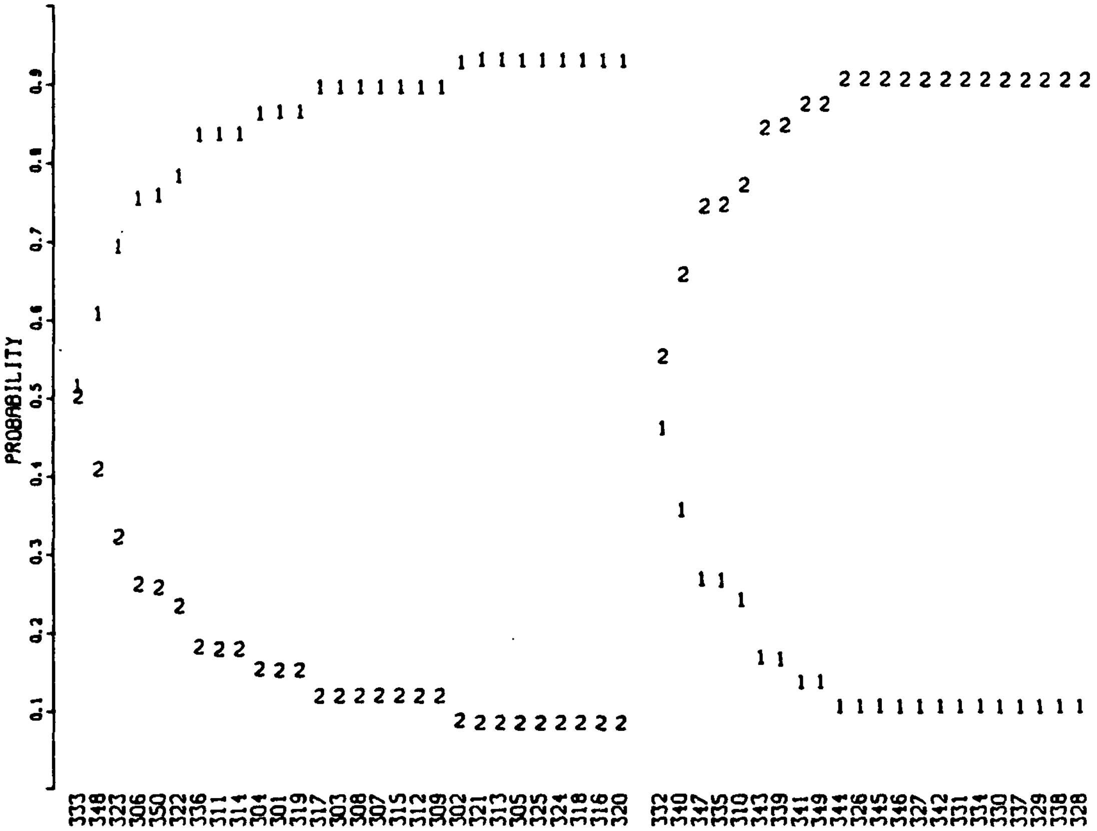


Figure 16. Probability of group membership for each sample using the PMG procedure for Example 2, data set 7.

Table 6. A comparison of the classification abilities of the PMG procedure with the hierarchical variance criterion (V) and the linear discriminant function (LDF)

	Number misclassified				
	Experimental error				No error
Data Set	LDF	V	PMG procedure	Frequency measure	V
1	5	10	4	5	4
2	4	5	3	5	5
3	6	7	7	7	5
4	0	0	0	0	2
5	5	9	8	9	6
6	2	4	4	4	6
7	3	5	5	5	3
8	0	2	3	2	3
9	2	5	3	5	4
10	5	3	4	3	5
Average	3.2[1]	5.0[2]	4.1	4.5	4.3

[1] Theoretical value = 3.7 from Bayne et al. [1978]

[2] Theoretical value = 5.7 from Bayne et al. [1978]

The second test reemphasizes the ability of the dendrograms, NORMIX, Beale's F statistic, Calinski and Harabasz Kg, and the PMG method to select the proper number of groups in the data. The large standard deviation about the mean used in generating the data sets caused some difficulty. The division of the dendrograms was more subjective, the F statistic and the Kg misjudged some of the data sets, and the PMG procedure results are not as decisive as in the first example. The PMG procedure definitely improved the misclassification of variance criteria.

4.3 Practical Application

The third example is more difficult to analyze since no *a priori* information about the number of groups is available. It is given here to illustrate the practical application of the methods. The data consist of concentration measurements of 10 different elements for 53 stream sediment samples collected around Llano, Texas as part of the National Uranium Resource Evaluation Project [Nichols et al., 1976]. Before any cluster analysis is performed on the data, it is transformed by logarithms to approximately normalize the data. The variables are multiplied by subjective weights derived to emphasize measurements hypothesized to be important to uranium geochemistry.

A dendrogram (again using variance criterion) of the data is in Fig. 17. No subgroups of samples are sufficiently distinct to warrant formation of a specific number of clusters. A subjective division is separation between pairs 153 and 1023; 64 and 1826; 1773 and 1851; 275 and 73; and 157 and 1843, forming six clusters.

The optimization criteria of MICKA are in Fig. 18 for $\hat{g}$ = 2, 3, 4, 5, 6, 7, and 8. The only apparent aid in determining the number of groups is the gradual increase for $\hat{g}$ = 1 to 7 and then the drastic decrease of the trace of $B^{-1}W$ at $\hat{g}$ = 8. Although this criterion has not helped in choosing the number of groups in the previous two examples, it suggests seven groups in the Llano data.

The NORMIX results in Fig. 19 suggest five, seven, or eight groups in the data. The significance tests for accepting five groups over four, for accepting seven over six, and for accepting eight over seven are all above or equal to .50 with the highest probability at seven groups. The log likelihood estimate, however, has continued to increase for each increase in $\hat{g}$.

The ISODATA procedure was also run on the Llano data. The input number of groups and parameters defining inter- and intragroup distances were varied, but the procedure did not converge. Whenever a higher number of groups was input, a higher number of groups was output, possibly signifying a hierarchical structure in the data.

ORNL DWG 78-1434

LLANO PILOT SURVEY STREAM SEDIMENT
53 SAMPLES CLUSTERED USING 10 VARIABLES.   DATE PLOTTED IS 01-10-78
CLUSTERING CRITERION -- MINIMUM (WITHIN-CLUSTER)
STANDARD DEVIATION
NORMALIZATION -- EACH VARIABLE IS DIVIDED BY ITS STANDARD DEVIATION
AND MULTIPLIED BY WEIGHTS CALCULATED FROM INPUT RANKS.

VARIABLE	WEIGHT	MEAN	STD.DEV.
L- U	0.295	0.300	0.367
L-AS	0.118	0.921	0.448
L-SE	0.147	-0.597	0.647
L-BA	0.088	5.083	0.478
L-CR	0.029	2.848	0.855
L-CU	0.088	1.477	0.643
L-MN	0.029	5.214	0.780
L-TI	0.029	7.373	0.800
L- V	0.118	3.049	0.538
L-ZR	0.059	4.902	0.753

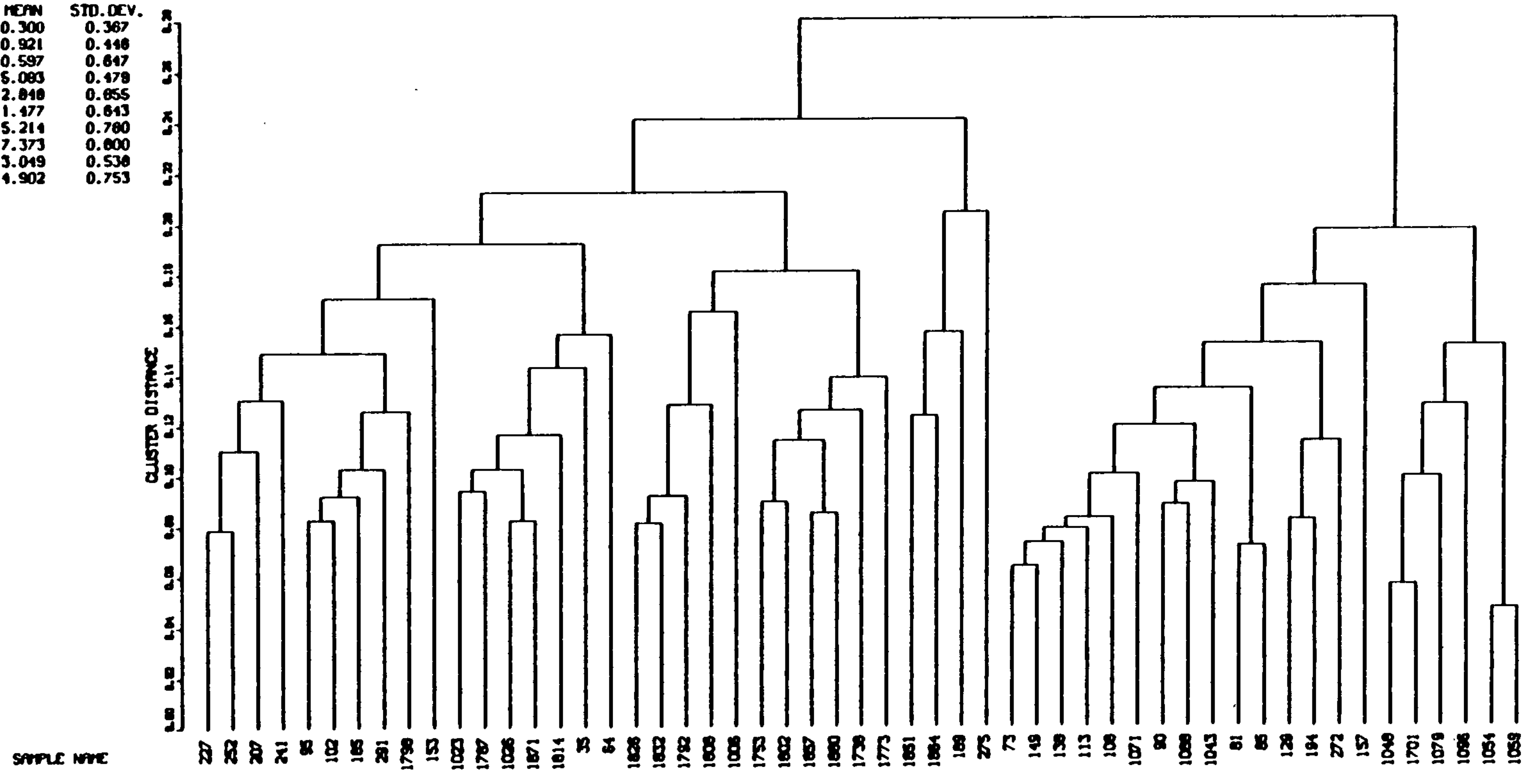


Figure 17. Dendrogram using variance criterion for Example 3.

ORNL DWG 78-2216

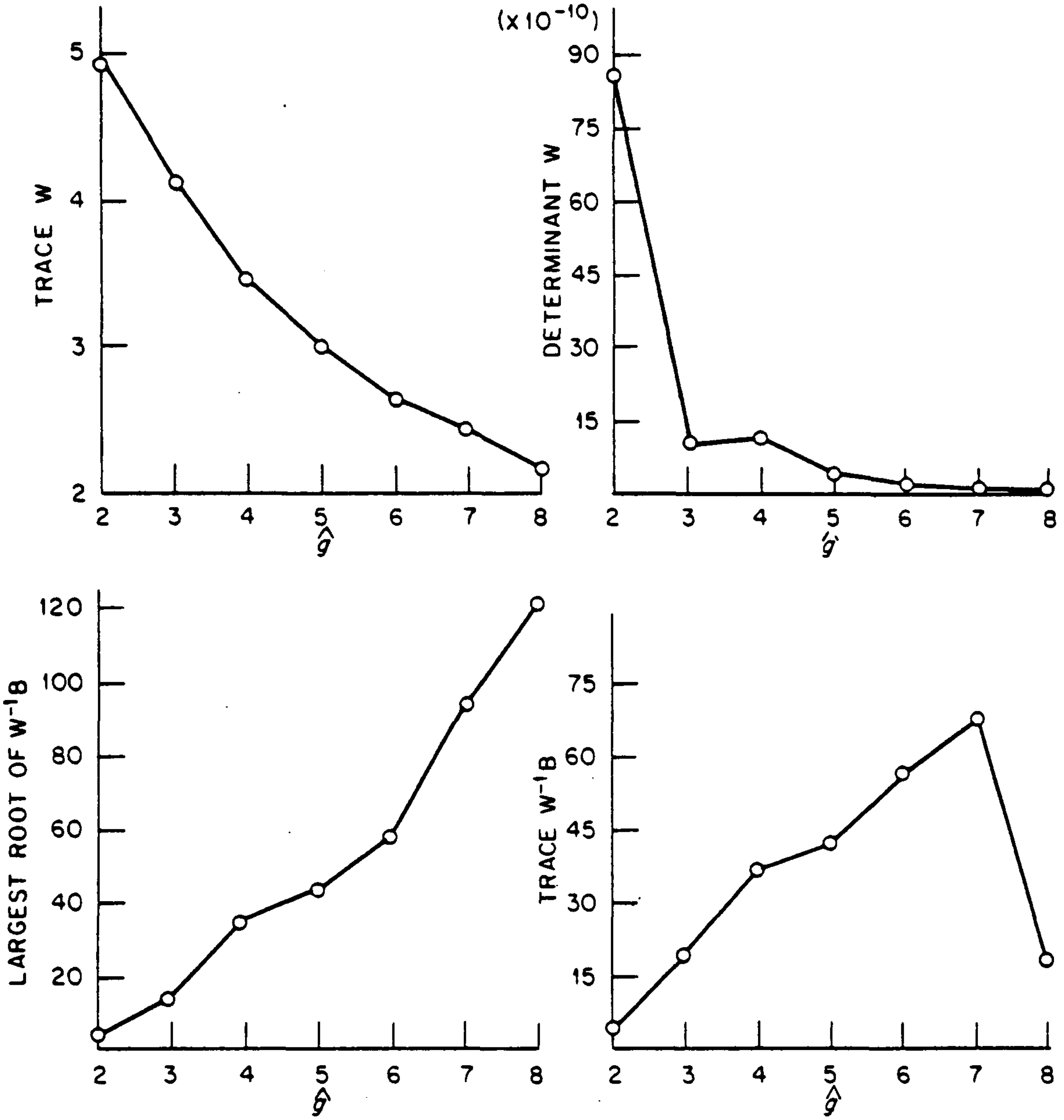


Figure 18. Plot of $\hat{g}$ versus optimization values using dispersion matrices criteria for Example 3.

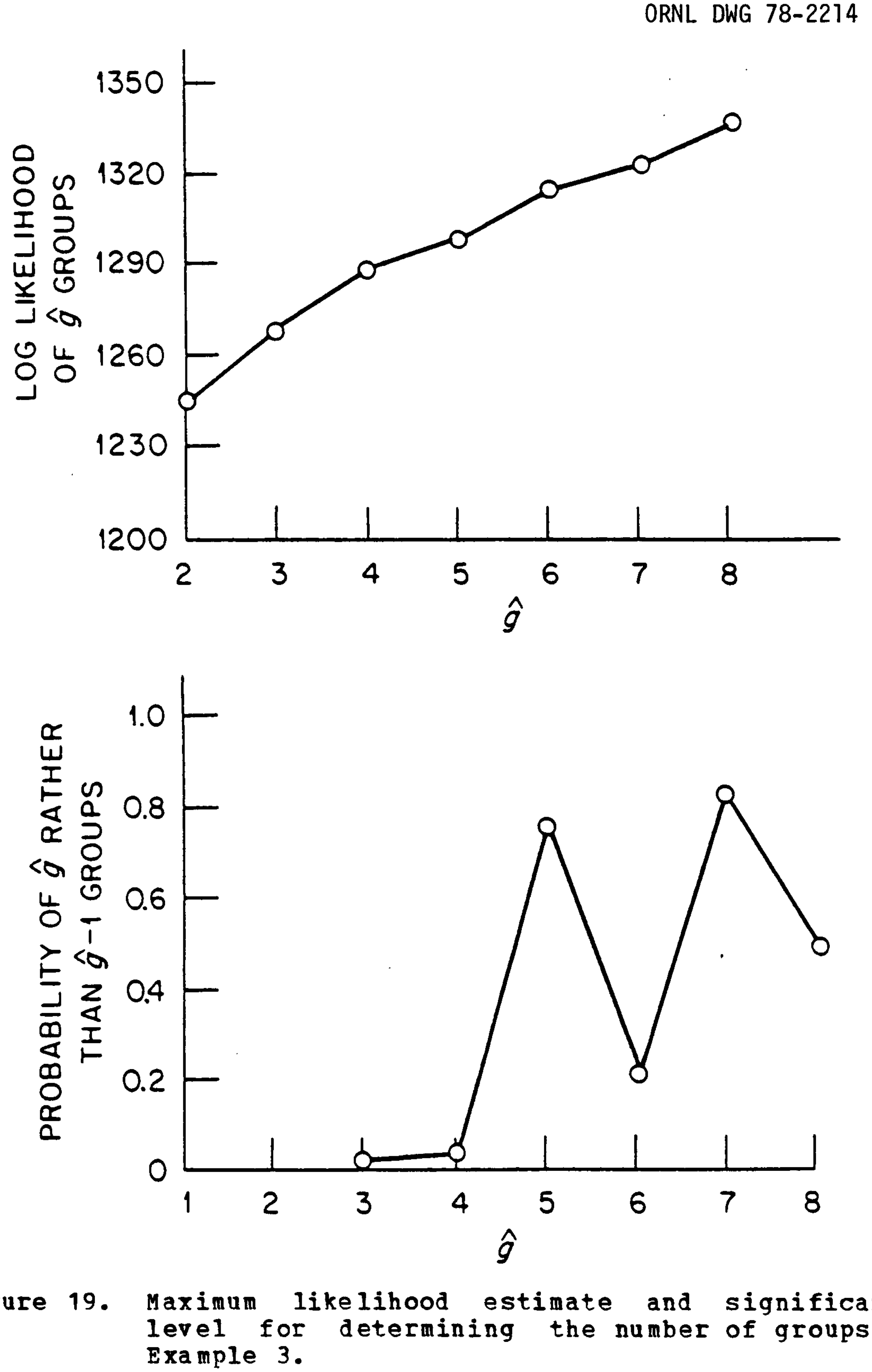


Figure 19. Maximum likelihood estimate and significance level for determining the number of groups in Example 3.

Both Beale's F statistic and Calinski and Harabasz Kg agreed with the theoretical number of groups in the first two examples; they both suggest a hierarchical grouping or a number of groups greater than eight on the Llano data (Fig. 20). The g^2(det W) indicator has a relative minimum at $\hat{g}$ = 3, but does decrease below that minimum at $\hat{g} = 6$.

The PMG method was run for c = 3, 4, 5, 6, 7, and 8. Fifty iterations were used in each case, with the error in each variable determined using a coefficient of variation as given in Kane and Larson [1977] and a truncated normal distribution of $\mu = (0,0)$ and $\Sigma = .2I$ (Eq. 20). A dendrogram of the frequency measure for each result is given in Fig. 21.

Table 7 lists g(c) for each c and each value of θ and λ. The first section with $\theta = .9$ suggests 7 groups (g(5) = g(6) = g(7) = 7 for λ = 10%, 1%, and .1%) or 8 groups (g(6) = g(7) = g(8) = 8 for λ = 10%, 1%, and .1%). However, for θ = .85, .8, and .75, 7 groups are indicated since g(6), g(7), and g(8) are all equal to 7 at one or more levels of λ. Note also that g(9) = 7 for $\theta = .75$ and $\lambda = .1\%$. A very low a_0 was necessary before g(9) = 7 since the 7 natural groups were forced to subdivide into 9 groups.

The value of $\hat{g} = 7$ agrees with the results obtained by clustering the frequency measure since there are 7 groups in both the dendrograms for c = 7 and c = 8. The groups predicted by PMG procedure and the frequency measure are

ORNL DWG 78-2215

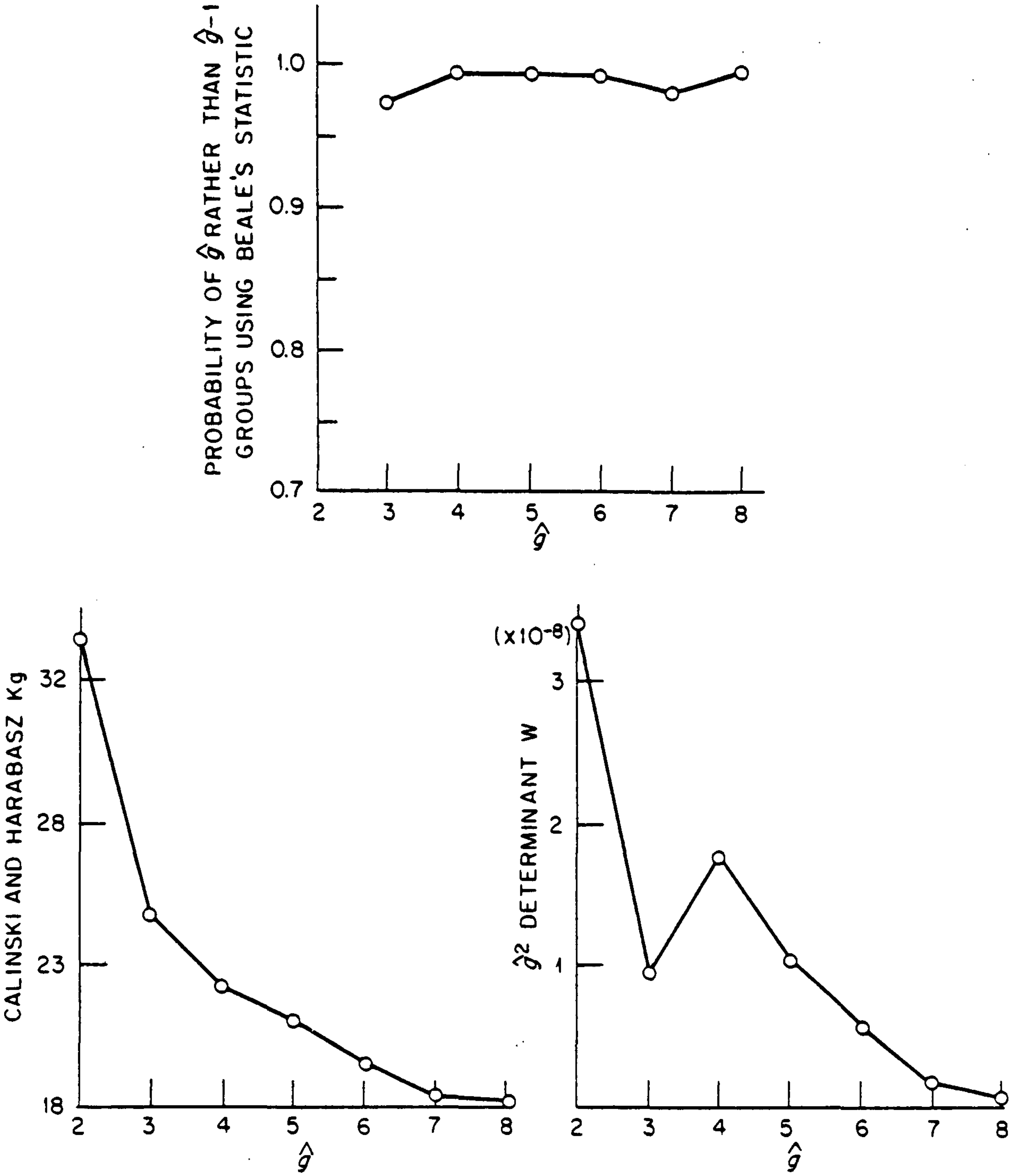


Figure 20. Results of Beale's F Statistic, Calinski and Harabasz's Kg and Marriot's g^2(det W) for Example 3.

ORNL DWG 78-1804

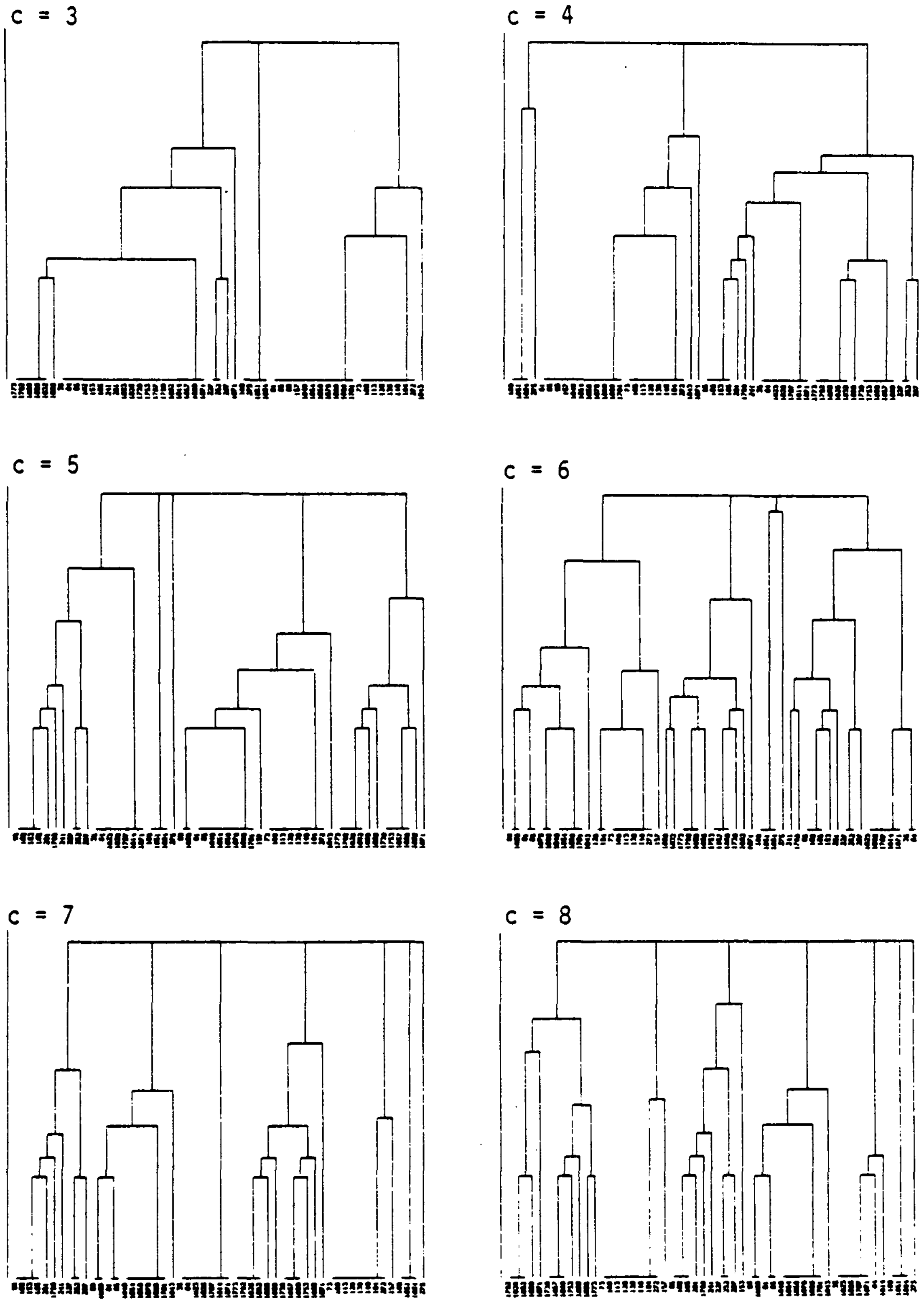


Figure 21. Dendrogram of frequency matrix for c = 3 to c = 8 for Example 3.

Table 7. Values of $\hat{g}$ estimated by the PMG procedure for Example 3

Probability	Number of clusters,c	λ =10.%	λ =1.0%	λ =0.1%
0.90	3	4	3	3
	4	5	4	3
	5	7	7	6
	6	8	8	7
	7	8	8	7
	8	9	9	8
	9	12	12	12
0.85	3	3	3	3
	4	4	3	3
	5	7	6	6
	6	8	7	7
	7	8	7	7
	8	9	7	7
	9	12	9	9
0.80	3	3	3	2
	4	3	3	3
	5	6	6	5
	6	7	7	7
	7	7	7	7
	8	7	7	7
	9	9	9	9
0.75	3	3	2	2
	4	3	3	3
	5	6	5	4
	6	7	7	7
	7	7	7	7
	8	7	7	7
	9	9	8	7

identical. A plot of the probability of each group for each sample is in Fig. 22. Note that the sixth group is very distinct, consisting of only samples 189, 1832, and 1864. In addition, sample 275 is an outlier.

ORNL DWG 78-6707

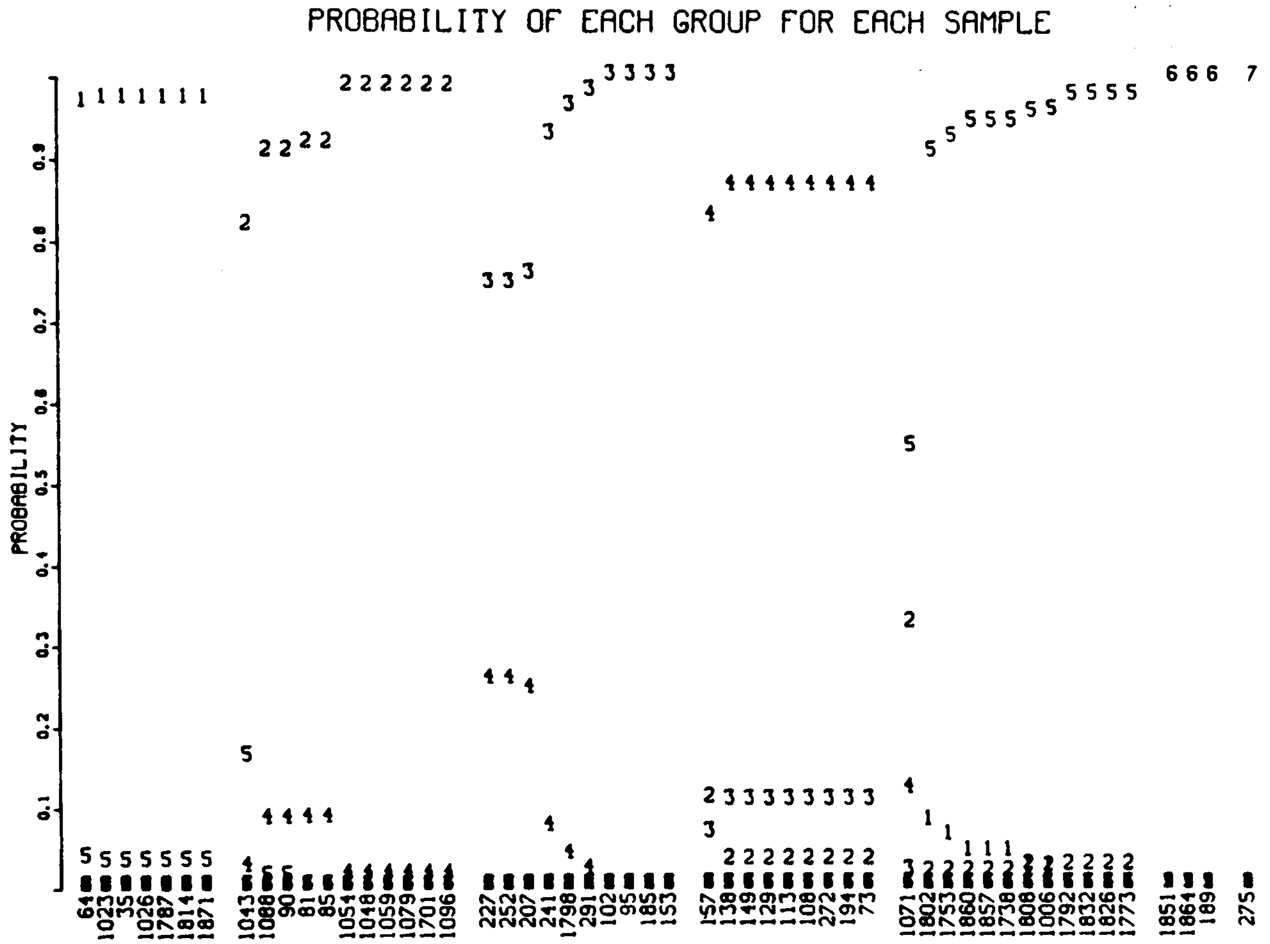


Figure 22. Probability of group membership for each sample using the PMG procedure for Example 3.

CHAPTER 5

CONCLUSIONS

5.1 Results

The PMG method is as successful in determining the number of groups in a data set as the other methods given in the literature for the three different examples tried. The analysis method has the additional advantage of using the experimental error and of defining the groups using a probabilistic association measure. The PMG procedure and the frequency measure give two different methods of examining the output.

The procedure seems unaffected by group size and shape. The three mathematical indicators which correctly predicted the number of groups in the first two examples are based upon the trace or determinant of W. Everitt [1974] states that using a trace W criterion forces a spherical shape on the data and that the determinant W criterion assumes that all groups are of the same shape. In addition, these two parameters become very small for small groups and therefore, may not represent the groups accurately.

The definition and separation of the groups are also an advantage of the PMG procedure. The separation between group 6 and all other groups in Example 3 is apparent from the analysis (Fig. 22); whereas, to determine these results would require further calculations using other methods. In addition, only the combination of error analysis with the grouping may be able to show this type of group separation.

The biggest disadvantage of the Monte Carlo analysis method is the time required for clustering of all of the perturbed data sets instead of just one. In the first example set, the extra time spent in doing the extra calculations was not advantageous since the hierarchical analysis as well as most of the other methods gave the correct results. However, in the third example, although we cannot be certain that there are or are not seven groups in the data, a great deal more information about the group divisions and separations can be learned from the output of the analysis method than from any combination of the other methods.

5.2 Future Work

The PMG procedure could be applied to other clustering algorithms besides hierarchical clustering. A comparison of results between two algorithms using the procedure would show the difference in the clustering methods as well as the overall stability of the procedure. A study of the best combinations of θ and λ necessary for determining g(c)--especially when the data are forced into a larger number of groups than are present, would also aid the method.

LIST OF REFERENCES

LIST OF REFERENCES

Anderberg, M.R. (1973), Cluster Analysis for Applications, New York: Academic Press.

Anderson, T.W. and Darling, D.A. (1952), "Asymptotic Theory of Certain Goodness of Fit Criteria Based on Stochastic Processes," Annals of Mathematic Statistics, 23,293.

Ball, G.H. and Hall, D.J. (1965), "ISODATA, A Novel Method of Data Analysis and Pattern Classification," AD 699616, Stanford Research Institute, Menlo Park, California.

Bayne, C.K., Beauchamp, J.J. , Begovich, C.L., and Kane, V.E. (1978), "Monte Carlo Comparison of Clustering Procedures," submitted for publication.

Begovich, C.L. and Larson, N.M., (1976), "A User's Manual for RECOG-ORNL," ORNL/CSD/TM-21, Oak Ridge National Laboratory, Oak Ridge, Tennessee.

Borucki, W.J., Card, D.H., and Lyle, G.C. (1975), "A Method of Using Cluster Analysis to Study Statistical Dependence in Multivariate Data," IEEE Transactions on Computers, C-24,12,1183-1191.

Brownlee, K.A. (1960), Statistical Theory and Methodology in Science and Engineering, New York: John Wiley and Sons, Inc.

Bryan, J.K. and Tebbe, D.L. (1970), "Generation of Multivariate Gaussian Data," Information Science Series MOU-IS-DN-7, University of Missouri, Columbia, Missouri.

Cormack, R.M. (1971), "A Review of Classification," Journal of the Royal Statistical Society, 134, 321-347.

Dorofeyuk, A.A. (1971), "Automatic Classification Algorithms," Automation and Control, 32,1928-1958.

Dubes, R. and Jain, A.K. (1976), "Clustering Techniques: The Users Dilemma," Pattern Recognition, 8,247-260.

Duewer, D.L., Koskinen, J.R., and Kowalski, B.R. (1975), "ARTHUR," available from B. R. Kowalski, Laboratory for Chemometrics, Department of Chemistry, University of Washington, Seattle, Washington.

Everitt, B. (1974), Cluster Analysis, London: Hienemann Education Books Ltd.

Friedman, H.P. and Rubin, J. (1971), "On Some Invariant Criteria for Grouping Data," Journal of American Statistical Association ,66,1159-1178.

Fukunaga, K. (1972), Introduction to Statistical Pattern Recognition, New York: Academic Press.

Fukunaga, K. and Hostetler, L.D. (1975), "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition," IEEE Transactions on Information Theory, IT-21,32-40.

Gorenstein, S. (1967), "Testing a Random Number Generator," Communications of the ACM, 10,111-118.

Gower, J.C. (1967), "Comparisons of Some Methods of Cluster Analysis," Biometrics, 23,623-637.

Halton, J.H. (1970), "A Retrospective and Prospective Survey of Monte Carlo Methods," SIAM Review, 12,1-63.

Hartigan, J.A. (1975), Clustering Algorithms, New York: John Wiley.

Koontz, W.L.G. and Fukunaga, K. (1972), "Asymptotic Analysis of a Nonparametric Clustering Technique," IEEE Transactions on Computers, C-21,967-974.

Koontz, W.L.G., and Narendra, P.M., and Fukunaga, K. (1976), "A Graph-Theoretical Approach to Nonparametric Cluster Analysis," IEEE Transactions on Computers, C-25,936-944.

International Business Machines Corporation (1970), "System/360 Scientific Subroutine Package, Version III, Programmer's Manual," IBM Corporation, New York.

International Mathematical and Statistical Libraries (1977), "IMSL Library, Edition 6," IMSL, Texas.

Kane, V.E., Baer, T., and Begovich, C.L. (1977), "Principal Component Testing for Outliers," K/UR-7,Oak Ridge Gaseous Diffusion Plant, Oak Ridge, Tennessee.

Kane, V.E. and Larson, N.M. (1976), "Clustering Problems for Geochemical Data," Proceedings of the Second ERDA Statistical Symposium, G. Tietjen and K. Campbell (ed.), LA-6758-C,239-261.

Kendall, M.G. and Smith, B.B. (1938), "Randomness and Random Sampling Numbers," Journal of the Royal Statistical Society, 51,147-167.

Kinderman, A.J. and Ramage, J.G. (1976), "Computer Generation of Normal Random Variables," Journal of the American Statistical Association, 71,893-896.

Kittler, J. (1976), "A Locally Sensitive Method for Cluster Analysis," Pattern Recognition, 8,23-33.

Lance, G.N. and Williams, W.T. (1967), "A General Theory of Classification Sorting Strategies, I. Hierarchical Systems," Computer Journal, 9,373-380.

---------------- (1968), "A General Theory of Classification Sorting Strategies, II. Clustering Systems," Computer Journal, 10,271-277.

Larson, N.M., Begovich, C.L., and Nall, V.C. (1977), "A User's Manual for the Hierarchical Cluster Analysis Code DENDRO," ORNL/CSD/TM-20, Oak Ridge National Laboratory, Oak Ridge, Tennessee.

Levene, H. and Wolfowitz, J. (1944),"The Covariance Matrix of Runs Up and Down," Annals of Mathematical Statistics, 15,58-69.

Lindgren, B.W. (1976), Statistical Theory, New York: Macmillan Publishing Co., Inc.

Ling, R.F. (1971), "Cluster Analysis," Technical Report No. 18, Department of Statistics, Yale University.

MacLaren, M.D. and Marsaglia, G. (1965), "Uniform Random Number Generators," Journal of the Association for Computing Machinery, 12,83-89.

Marriot, F.H.C. (1971), "Practical Problems in a Method of Cluster Anaysis," Biometrics, 27,501-514.

McRae, D.J. (1970), "MICKA: A FORTRAN IV Iterative K-means Cluster Analysis Program," CTB/McGraw-Hill, Del Monte Research Park, California.

Nagy, G. (1968), "State of the Art in Pattern Recognition," Proceedings of the IEEE, 56,836-862.

Nichols, C.E., Kane, V.E., Minkin, S.C., and Cagle, G.W. (1976), "Hydrogeochemical and Stream Sediment Pilot Survey of Llano Area, Texas," Report K-TL-602, Oak Ridge Gaseous Diffusion Plant, Oak Ridge, Tennessee.

Rand, W.M. (1971), "Objective Criteria for the Evaluation of Clustering Methods," Journal of the American Statistical Association, 66,846.

Shapiro, S. and Wilk, M. (1965), "An Analysis of Variance Test for Normality," Biometrika, 52,591-611.

Shreider, Y.A. (1964), Methods of Statistical Testing, Amsterdam: Elsvier Publishing Company.

Slagle, J.R., Chang, C.L., and Lee, R.C.T. (1974), "Experiment with Some Clustering Analysis Algorithms," Pattern Recognition, 6,181.

Sokal, R.R. and Sneath, P.H.A. (1963), Principles of Numerical Taxonomy, San Francisco: W.H.Freeman and Company.

Tou, J.T. and Gonzalez, R.C. (1977), Pattern Recognition Principles, Massachusetts: Addison-Wesley Publishing Company.

Westley, G.A. and Watts, J.A. (ed.), (1970), "The Computing Technology Center Numerical Analysis Library," CTC-39, Union Carbide Corporation, Oak Ridge, TN.

Wolfe, J.H. (1971), "NORMIX 360 Computer Program," Research Memorandum SRM 72-4, California: Naval Personnel and Training Research Laboratory.

----------(1971), "A Monte Carlo Study of the Sampling Distribution of the Likelihood Ratio for Mixtures of Multinormal Distributions," Research Memorandum SRM 72-2, California: Naval Personnel and Training Research Laboratory.

APPENDICES

APPENDIX A

PMG PROGRAM DESCRIPTION

The program listings for the three programs PMGPER, PMGCLS, and PMGEST are included in this Appendix. Flowcharts of the programs are in Figs. A1, A2, and A3.

The PMGPER program runs on the IBM360/91, using a FORTRAN G compiler with optimization level two, and requires approximately 270 K of core and 60 seconds execution time for the 50 iterations of the 53 sample data set of Example 3. PMGCLS and PMGEST both run interactively on the PDP-10, each requiring less than 10 seconds execution time for the same problem and one value of c. A listing of the input required for each program follows.

The input required by PMGPER is similar to that required by DENDRO [Larson et al., 1977]. The reader is referred to that report for a more detailed description of the parameters.

Card 1 Variables=IFLAG, NORM, MET, INV, LOG, NGRPS, NTIMES, ITRN, IERR Format=(3I5,10X,6I5)

IFLAG indicates which clustering criterion is to be used:

IFLAG=1 for single linkage clustering criterion.

IFLAG=2 for complete linkage clustering criterion.

IFLAG=3 for group average clustering criterion.

IFLAG=4 for weighted average clustering criterion.

IFLAG=5 for centroid clustering criterion.

ORNL DWG 78-6920

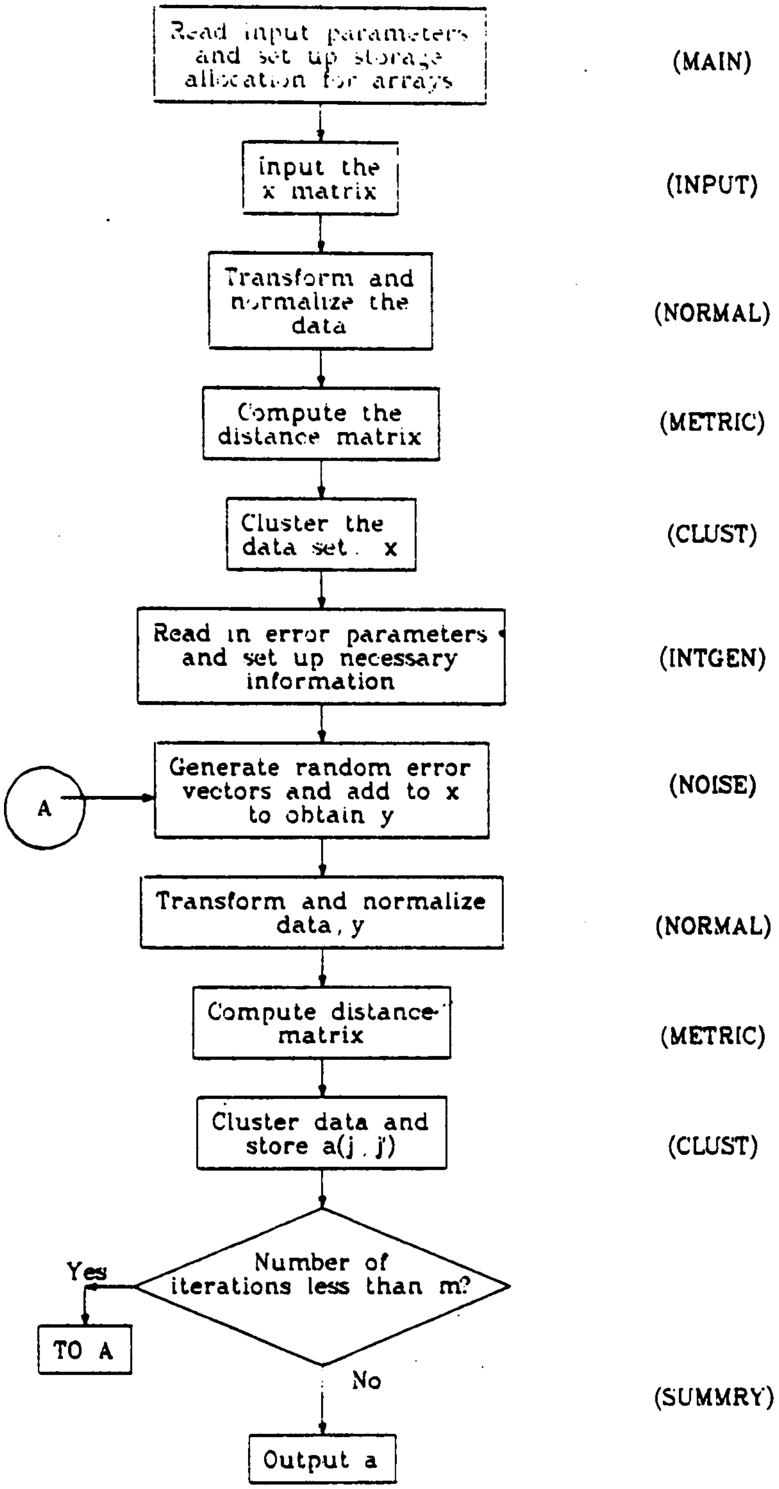

Read input parameters and set up storage allocation for arrays
(MAIN)
Input the x matrix
(INPUT)
Transform and normalize the data
(NORMAL)
Compute the distance matrix
(METRIC)
Cluster the data set . x
(CLUST)
Read in error parameters and set up necessary information
(INTGEN)
A
Generate random error vectors and add to x to obtain y
(NOISE)
Transform and normalize data, y
(NORMAL)
Compute distance matrix
(METRIC)
Cluster data and store a(j , j')
(CLUST)
Number of iterations less than m?
Yes
TO A
No
(SUMMRY)
Output a

Figure A1. Flowchart of PMGPER. This program perturbs data set X, clusters the data sets Y , and stores the resulting cluster information in the frequency matrix.

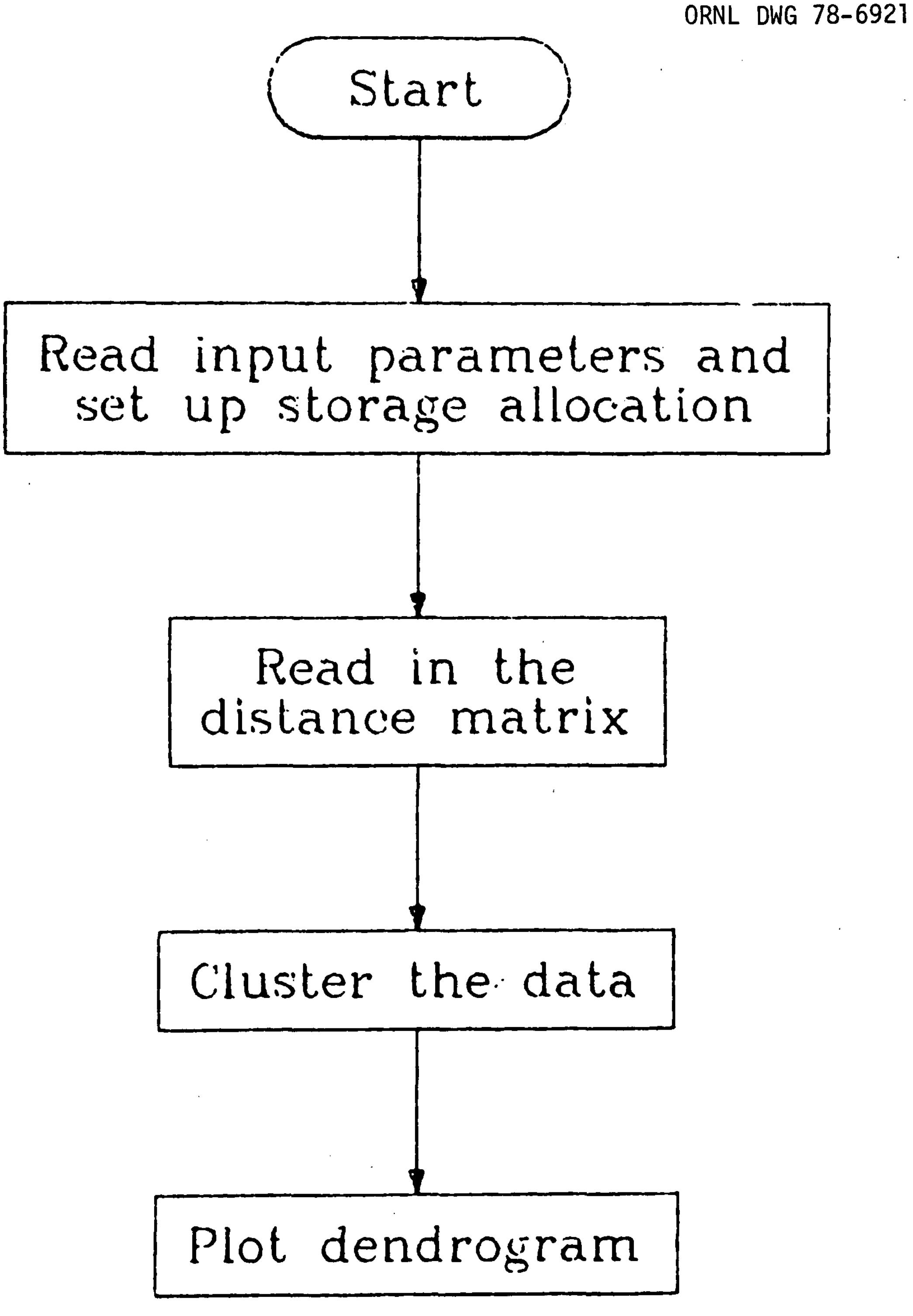

ORNL DWG 78-6921
Start
Read input parameters and set up storage allocation
Read in the distance matrix
Cluster the data
Plot dendrogram

Figure A2. Flowchart of PMGCLS. This program clusters the frequency matrix.

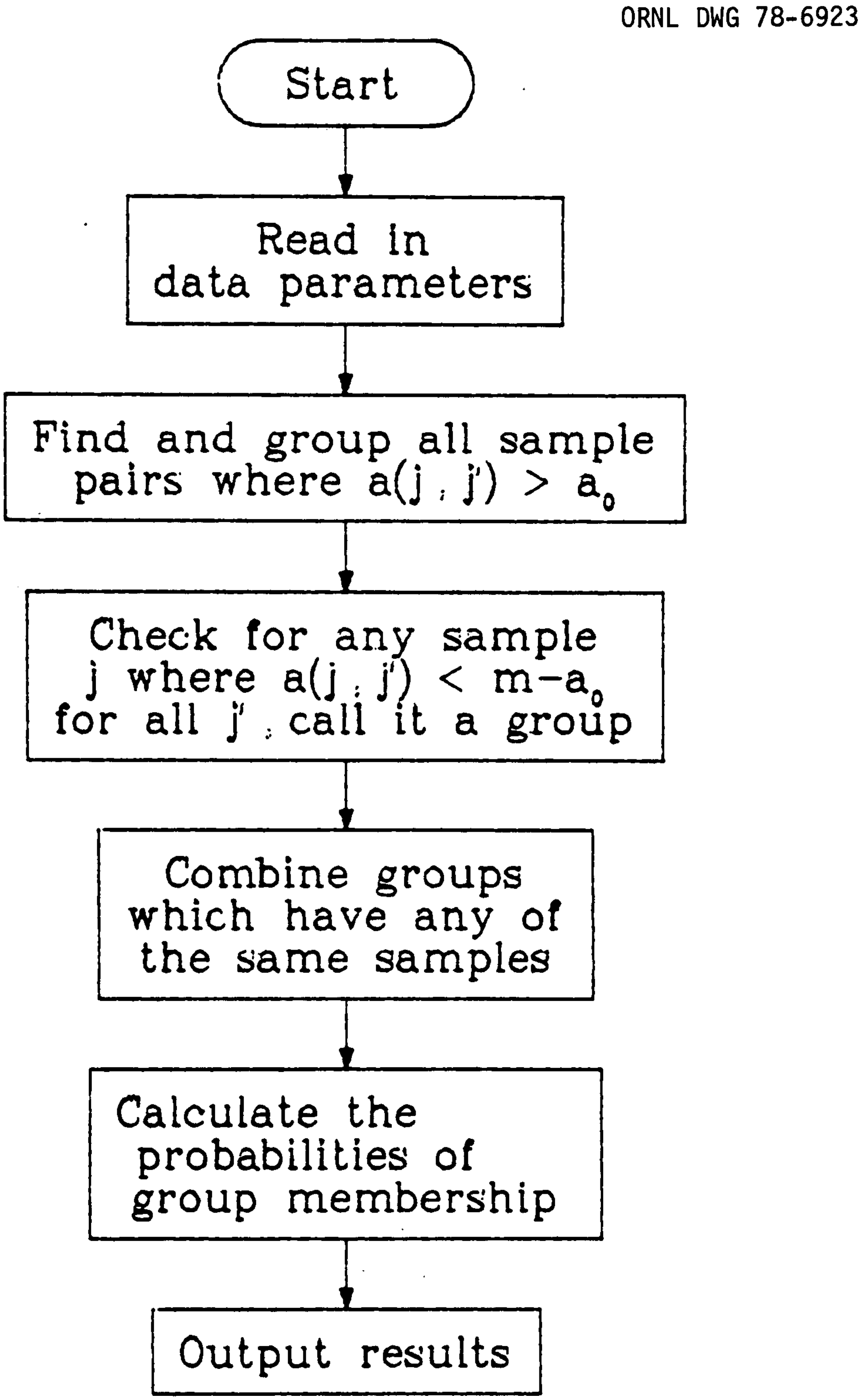

ORNL DWG 78-6923
Start
Read in data parameters
Find and group all sample pairs where $a(j\,;\,j') > a_0$
Check for any sample j where $a(j\,;\,j') < m-a_0$ for all j' ; call it a group
Combine groups which have any of the same samples
Calculate the probabilities of group membership
Output results

Figure A3. Flowchart of PMGEST. This program performs the PMG procedure.

IFLAG=6 for median criterion.

IFLAG=7 for Ward's method clustering criterion.

IFLAG=8 for variance method.

NORM indicates what to use to normalize the data:

NORM=0 for unnormalized data.

NORM=1 for divide by maximum.

NORM=2 for divide by standard deviation.

NORM=3 for multiply by input weights.

NORM=4 for divide by standard deviation and then multiply by input weights.

MET indicates what distance measure to be used:

MET=1 for the square of the Euclidean metric.

MET=2 for the Euclidean metric.

MET=3 for the square of a normalized Euclidean metric.

MET=4 for normalized Euclidean metric.

MET=5 for Pearson correlation based metric.

MET=6 for Spearman correlation based metric.

MET=7 for city-block distance.

INV is used to indicate how the data is to be read:

INV=0 for all variables for a given sample .

INV=anything else for all samples for a given variable.

LOG is used with the correlation metrics to determine threshold values:

LOG=0 use a large negative threshold.

LOG=1 use a threshold of 0.

LCG=2 use a threshold of -10.

LOG=3 read in a threshold value for all variables.

NGRPS is the number of values of c the data are to be clustered into for storing the frequency matrix.

NTIMES is the number of Monte Carlo iterations to be performed.

ITRN is the transformation to be used on the data:

ITRN=0 for no transformation.

ITRN=1 for transforming with logarithms.

ITRN=2 for transforming with exponentiation.

ITRN=3 for transforming with square root.

IERR determines the error distribution:

IERR=1 use normal distribution.

IERR=2 use normal distribution with coefficient of variation.

IERR=3 use uniform distribution.

Card 2 Variables=IGRPS Format=(16I5)

IGRPS is the value of c, the number of clusters to use to determine the frequency matrix. There are NGRPS values; a frequency matrix is computed for each value.

Card 3 Variables= NAME, NFEAT, NSAMP, NSNAM, NVNAM Format=(A4,4I5)

NAME is a four-character name for the data set.

NFEAT is the number of variables.

NSAMP is the number of samples or observations.

NSNAM is the number of four-character words in each sample name.

NVNAM is the number of four-character words in each variable name.

Card 4 Variables=FMT Format=(20A4)

FMT is either a format statement to be used to read all variables for one sample and then three descriptors: a floating point property, a class name and a sample name or it is the characters 'BCD '. If FMT is 'BCD ', then the next two cards are required.

Card 5 Variables=FMT Format=(20A4)

FMT is a format to read only the variables for all samples. This card is required only if FMT of Card 4 is 'BCD '.

Card 6 Variables=FMT2 Format=(20A4)

FMT2 is a format to read only the three descriptors, a floating point property, a class name and a sample name. This card is required only if FMT of Card 4 is 'BCD '.

Card 7 Variables=TITLE Format=(20A4)

TITLE is a three-line title used for identification purposes.

Card 8 Variables=DATA, PP, CL, SAMPLE Format=FMT (Card 4) or FMT and FMT2 (Cards 5 and 6)

DATA is the sample data.

PP is a floating point property of the data which is used only for identity purposes.

CL is a class name which is used only for identity purposes.

SAMPLE is a sample name which is used only for identity purposes.

Card 9 Variables=VAR Format=(20A4)

VAR is the array containing the variables' names.

Card 10 Variables=CUT Format=(8F10.0)

CUT is the lower bound of each variable. It is used to check the generated values of Y.

Card 11 Variables=WEIG Format=(8F10.0)

WEIG is the weights for each variable. This card is in the input deck only if NORM indicates that weights are going to be used.

Card 12 Variables=ICONST Format=(8I10)

ICONST is the array containing initial seeds to be used for the uniform number generator. One seed is required for each variable.

Card 13 Variables=SIGMA or BLIM, ULIM Format=(8F10.0)

If IERR is equal to 1 or 2, then SIGMA is read in, where SIGMA is the lower diagonal form of the covariance matrix of the normal distribution of the error. If IERR is equal to 3, BLIM and ULIM are read in, where BLIM is the lower limit and ULIM is the upper limit of the uniform interval used to characterize the error.

Card 14 Variables=CV Format=(8F10.0)

CV is the coefficient of variation of each variable, read in only if IERR = 2.

Card 15 Variables=TRUN Format=(8F10.0)

TRUN is the upper limit of the noise allowed when a normal distribution is used for generating the random error. If TRUN is 0, TRUN is set to be 100000.

The input required by PMGCLS is prompted by the program upon execution. The user is asked for a value of IFLAG to use for the clustering criterion of $d^*_{jj'}$. The options are the same as given in PMGPER. The user is also asked to enter the number of samples in the data set. The only other required input is the data set output by PMGPER. These data consist of a title, a list of sample names, and the frequency matrix.

The input required by PMGEST is also prompted by the program. The first request is for the number of samples in the data set. The value of a_0 or CRITNO is the next input. The number of Monte Carlo iterations is also requested. As in PMGCLS, the only other data required are the card output from PMGPER. The program listings for PMGPER, PMGCLS, and PMGEST follow.

```
C       PMGPER PROGRAM LISTING
C
C
C-----------------------------------------------------------
C       CLUSTERING PROGRAM
C       ORIGINAL VERSION (VERY DIFFERENT FROM THIS) OBTAINED
C         FROM ERNIE HALL, USC, LOS ANGELES
C
C       THIS VERSION IS WRITTEN TO BE PART OF THE
C       PMG PROCEDURE.  THE DATA IS READ IN,
C       PERTURBED BY EXPERIMENTAL ERROR, AND
C       CLUSTERED INTO C CLUSTERS.  THE
C       OCCURENCE OF EACH SAMPLE WITH ANY OTHER SAMPLE
C       IN THE SAME CLUSTER IS STORED IN A FREQUENCY
C       MATRIX WHICH IS OUTPUT FOR FURTHER
C       ANALYSIS BY PMGCLS OR PMGEST.
C-----------------------------------------------------------
C
      COMMON/ALWAYS/TITLE(30),DATE(5),TODAY(2),IFLAG,
     > NORM,MET,NSQUAR,INV,LOG,NGRP,ITRN
      INTEGER DIMEN
      COMMON/HOLD/NSIZE,A(30000)
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),TITCLS(5),
     A  PTALK(20)
C
C
      DIMENSION DDATE(5),IGRP(10)
      DATA DDATE/'*** ',' TOD','AYS ','DATE',' IS '/
C
C
      NSIZE=30000
C
      DO 10 I=1,5
   10 DATE(I)=DDATE(I)
C
   20 CALL LOOSE(1)
C
      CALL IDAY(TODAY)
      WRITE(6,10200)TODAY
C
C
C ---------------------------------------------------------
C
C       IFLAG=1 FOR SINGLE LINKAGE CRITERION
C       IFLAG=2 FOR COMPLETE LINKAGE CRITERION
C       IFLAG=3 FOR GROUP AVERAGE CRITERION
C       IFLAG=4 FOR WEIGHTED AVERAGE CRITERION
C       IFLAG=5 FOR CENTROID CRITERION
C       IFLAG=6 FOR MEDIAN CRITERION
C       IFLAG=7 FOR MINIMUM INCREASE IN (WITHIN-CLUSTER)
C       SUM OF SQUARES
C       IFLAG=8 FOR MINIMUM (WITHIN-CLUSTER)
C       STANDARD DEVIATION
```

```
C
C       NORM=0 FOR UNNORMALIZED DATA
C       NORM=1 FOR DIVIDE BY MAX
C       NORM=2 FOR DIVIDE BY STANDARD DEVIATION
C       NORM=3 FOR MULTIPLY BY INPUT WEIGHTS
C       NORM=4 FOR DIVIDE BY STANDARD DEVIATION
C       AND MULTIPLY BY INPUT WEIGHT
C       NORM=5 FOR DIVIDE BY ROBUST STANDARD DEVIATION
C       NORM=6 FOR DIVIDE BY ROBUST STANDARD DEVIATION AND
C              MULTIPLY BY INPUT WEIGHT
C
C       MET=1 FOR EUCLIDEAN METRIC (SQUARED)
C       MET=2 FOR EUCLIDEAN METRIC (NOT SQUARED)
C       MET=3 FOR NORMALIZED EUCLIDEAN METRIC (SQUARED)
C       MET=4 FOR NORMALIZED EUCLIDEAN METRIC (NOT SQUARED)
C       MET=5 FOR PEARSON CORRELATION - BASED METRIC
C       MET=6 FOR SPEARMAN CORRELATION - BASED METRIC
C       MET=7 FOR CITY-BLOCK DISTANCE
C
C       NGRP=NUMBER OF DIFFERENT CLUSTERS DIVISIONS
C       FOR WHICH FREQUENCY MATRIX WILL BE OUTPUT.
C
C
C       NTIMES=NUMBER OF MONTE CARLO ITERATIONS (M)
C
C       ITRN=0, NO TRANSFORMATION OF THE DATA
C       ITRN=1, USE LOGARITHM TO TRANSFORM THE DATA
C       ITRN=2, USE EXPONENTIATION TO TRANFORM THE DATA
C       ITRN=4, USE SQUARE ROOT OF THE DATA
C
C
C       IERR=1, EXPERIMENTAL ERROR IS FROM
C       NORMAL DISTRIBUTION
C       IERR=2, EXPERIMENTAL ERROR IS FROM
C       NORMAL DISTRIBTUTION, BUT USE COEFFICIENTS OF
C       VARIATION
C       IERR=3, EXPERIMENTAL ERROR IF FROM
C       UNIFORM DISTRIBUTION
C
C       IGRPS=NGRPS VALUE OF C, THE NUMBER OF CLUSTERS
C       TO DIVIDE THE DATA INTO TO OBTAIN A,
C       THE FREQUENCY MATRIX
C
C
C ---------------------------------------------------------
C
C *** INPUT PARAMETERS
      READ(5,10000,END=70)IFLAG,KNORM,MET,NSQUAR,KINV,
     A   LOG,NGRP,NTIMES,ITRN,IERR
      READ(5,10000) (IGRP(I),I=1,NGRP)
      IF (ITRN.EQ.0) ITRN=1
      WRITE(6,10100)IFLAG,KNORM,MET,NSQUAR,KINV,
     A   LOG,NGRP,NTIMES,ITRN,IERR
```

```
      IF (IFLAG.EQ.0) IFLAG=2
      IF (IPLOT.EQ.0) IPLOT=1
      IF (MET.EQ.0) MET=1
      IF (IFLAG.EQ.5.AND.MET.NE.1) WRITE(6,10400)
      IF (IFLAG.EQ.7.AND.MET.NE.1) WRITE(6,10500)
      IF (IFLAG.EQ.8.AND.MET.NE.1) WRITE(6,10600)
C
      NORM=KNORM
      INV=KINV
C
C
C *** READ DATA PARAMETERS
C
C       NAME IS THE DATA SET NAME
C       (TO BE IGNORED AFTER READING IN)
C       NFEAT IS THE NUMBER OF FEATURES,
C       NVNAM THE NUMBER OF (FOUR-BYTE)
C       WORDS USED TO DESCRIBE THE FEATURE
C       NSAMP IS THE NUMBER OF SAMPLES,
C       NSNAM THE NUMBER OF (FOUR-BYTE)
C       WORDS USED TO DESCRIBE THE SAMPLE
C
      READ(5,10300) NAME,NFEAT,NSAMP,NSNAM,NVNAM
      IF (NSNAM.EQ.0) NSNAM=1
      IF (NVNAM.EQ.0) NVNAM=1
C
      IF (INV.EQ.0) GO TO 30
C *** INVERT ORDER OF PARAMETERS IF INV .NE. 0
      N=NFEAT
      NFEAT=NSAMP
      NSAMP=N
      N=NVNAM
      NVNAM=NSNAM
      NSNAM=N
C
C
   30 CONTINUE
      CALL TALK(NFEAT,NSAMP)
C
C
C      PREPARE ARRAYS FOR CALLS TO SUBROUTINES
C      INPUT AND NORMAL (PLUS PREPARE OTHER
C      ARRAYS THAT NEED KEEPING)
C
      LLWEIG=DIMEN(NFEAT)
C                 WEIGHT
      LLDIST=DIMEN(NSAMP)
C                 DISTAN OR DIST
      NS2=NSNAM*NSAMP
      LLSAMP=DIMEN(NS2)
C                 SAMPLE (NAMES)
      N=0
      NF2=NVNAM*NFEAT
      LLVAR=DIMEN(NF2)
C                 VAR -- VARIABLE NAMES
      N=NSAMP*(NSAMP-1)/2
      LLSQUA=DIMEN(N)
C                 SQUA -- DISTANCE BETWEEN SAMPLES
      LLEXTR=DIMEN(NSAMP)
C                 EXTRA -- ARRAY FOR STORING OLD
C                          VARIANCES FOR MINIMUM
C                          STANDARD DEVIATION CRITERION
      N=NSAMP*NFEAT
      LLDATA=DIMEN(N)
C                 DATA
      LLCUT=DIMEN(NFEAT+1)
C                 TO STORE LOWEST MEASUREABLE VALUES
      LIHOLD=DIMEN(N)
C.                TO STORE DATA
      LLCON=DIMEN(NFEAT)
C                 TO STORE SEEDS FOR UNIFORM GENERATOR
      LLVAL=DIMEN(NFEAT)
C                 TO STORE NORMAL VALUES
      LLXH=DIMEN(4*NFEAT)
C                 TO STORE MEAN,SD,SKEWNESS,AND KURTOSIS
      N=(NFEAT*(NFEAT+1)/2+1)
      LLTA=DIMEN(N)
      LLSIG=DIMEN(N)
      K=0
      IF (IERR.EQ.2) K=NFEAT
      LLCV=DIMEN(K)
C                 TO STORE TRANSFORMATION ARRAY.
      N=NFEAT
C
      CALL INPUT(A(LLDATA),A(LLSAMP),A(LLVAR),A(LLCUT),
     A  A(LLHOLD),NFEAT,NSAMP)
C
      LLX=DIMEN(NSAMP)
      CALL NORMAL(A(LLWEIG),A(LLDATA),A(LLVAR),A(LLX),
     A A(LLCUT),NFEAT,NSAMP,1)
C
C
   40 CONTINUE
C
C     PREPARE ARRAYS FOR CALL TO SUBROUTINE METRIC
C
      N=0
      IF (MET.EQ.5.OR.MET.EQ.6) N=NSAMP
      LLTHRS=DIMEN(N)
C                 THRESH -- THRESHOLD FOR NOT COUNTING
C                           THAT SAMPLE IN CORR-DISTANCES
      N=0
      IF (MET.EQ.5.OR.MET.EQ.6) N=NFEAT
      LLXX=DIMEN(N)
C                 XX
      LLYY=DIMEN(N)
```

```
C                 YY
      LLIR=DIMEN(N)
C                 IR
      LLR=DIMEN(N)
C                 R
C
      CALL METRIC(A(LLDATA),A(LLSQUA),A(LLXX),A(LLYY),
     A  A(LLIR),A(LLR),A(LLTHRS),A(LLEXTR),
     A  A(LLSAMP),NSAMP,NFEAT)
C
C
C     PREPARE ARRAYS FOR CALLS TO SUBROUTINE CLUSTER
C
C
      LLKLUS=DIMEN(NSAMP+1)
C                 KLUSTR
      N=NSAMP/2+1
      LLMARR=DIMEN(N)
C                 MARRAY
      LLJARR=DIMEN(N)
C                 JARRAY
      NS=NSAMP/2+1
      LLMCEL=DIMEN(NS)
C                 MCEL
      LLJCEL=DIMEN(NS)
C                 JCEL
      LLONCL=DIMEN(N)
C                 ONDCL
      LLINCL=DIMEN(N)
C                 INDCL
C
      NK=((NSAMP*(NSAMP-1))/2*NGRP)/2+1
      LLKNT=DIMEN(NK)
      DO 50 N=1,NK
      A(LLKNT-1+N)=0.0
   50 CONTINUE
      CALL CLUST(A(LLDIST),A(LLKLUS),A(LLMARR),A(LLJARR),
     A  A(LLDATA),A(LLMCEL),A(LLJCEL),A(LLONCL),A(LLINCL),
     B  A(LLSQUA),A(LLEXTR),A(LLKNT),IGRP,NFEAT,NSAMP)
C
C
C     PREPARE ARRAYS FOR CALL TO SUBROUTINE DENDRO
C
      CALL INTGEN(A(LLTA),A(LLCON),A(LLSIG),A(LLCV),
     A  TRUN,IERR,NFEAT,BLIM,ULIM)
      DO 60 MTIME=2,NTIMES
      CALL NOISE(A(LLVAR),A(LLDATA),A(LLHOLD),A(LLCON),
     A  A(LLVAL),A(LLTA),A(LLXH),A(LLCV),TRUN,IERR,
     B  MTIME,NFEAT,NSAMP,BLIM,ULIM)
      CALL NORMAL(A(LLWEIG),A(LLDATA),A(LLVAR),A(LLX),
     A  A(LLCUT),NFEAT,NSAMP,MTIME)
      CALL METRIC(A(LLDATA),A(LLSQUA),A(LLXX),A(LLYY),
     A  A(LLIR),A(LLR),A(LLTHRS),A(LLEXTR),A(LLSAMP),
     B  NSAMP,NFEAT)
      CALL CLUST(A(LLDIST),A(LLKLUS),A(LLMARR),A(LLJARR),
     A  A(LLDATA),A(LLMCEL),A(LLJCEL),A(LLONCL),
     B  A(LLINCL),A(LLSQUA),A(LLEXTR),
     C  A(LLKNT),IGRP,NFEAT,NSAMP)
   60 CONTINUE
      CALL SUMMRY(A(LLKNT),A(LLSAMP),TITLE,NSAMP,NGRP)
      GO TO 20
   70 STOP
C
C
C
10000 FORMAT(16I5)
10100 FORMAT('0IFLAG,NORM,MET,NSQUAR,INV,',
     A  'LOG,NGRP,NTIMES,ITRN',5X,16I5)
10200 FORMAT('1TODAY IS ',2A4)
10300 FORMAT(A4,2I5,15X,2I5)
10400 FORMAT('0FOR CENTROID CRITERION,',
     A  ' YOU HAVE CHOSEN NOT TO DO EUCLIDEAN METRIC',
     B  ' (SQUARED).  THIS IS NOT A TRUE CENTROID.')
10500 FORMAT('0FOR INCREASE-IN-VARIANCE CRITERION,',
     A  ' YOU HAVE CHOSEN NOT TO DO EUCLIDEAN METRIC ',
     B  '(SQUARED).  THIS IS NOT A TRUE ',
     C  'INCREASE-IN-VARIANCE.')
10600 FORMAT('0FOR STANDARD-DEVIATION CRITERION,',
     A  ' YOU HAVE CHOSEN NOT TO DO EUCLIDEAN METRIC ',
     B  '(SQUARED).  THIS IS NOT A TRUE STANDARD',
     >' DEVIATION.')
      END
      SUBROUTINE SUMMRY(KNT,SAMP,TITLE,NSAMP,NGRP)
      INTEGER*2 KNT(NGRP,1)
      DIMENSION SAMP(NSAMP),TITLE(20)
      NN=((NSAMP-1)*NSAMP)/2
      DO 10 K=1,NGRP
      WRITE(07,10000) TITLE
      WRITE(07,10000) SAMP
      WRITE(07,10100) (KNT(K,I),I=1,NN)
   10 CONTINUE
      RETURN
10000 FORMAT(20A4)
10100 FORMAT(20I4)
      END
      SUBROUTINE INTGEN(A,ICONST,SIGMA,CV,TRUN,IERR,
     A  NFEAT,BLIM,ULIM)
      DIMENSION A(1),ICONST(1),SIGMA(1),CV(1)
      READ(5,10000) (ICONST(I),I=1,NFEAT)
      GO TO (10,10,140),IERR
   10 NF=((NFEAT+1)*NFEAT)/2
      READ(5,10100) (SIGMA(I),I=1,NF)
      IF (IERR.EQ.2) READ(5,10100) (CV(I),I=1,NFEAT)
      READ(5,10100) TRUN
      IF (TRUN.EQ.0.0) TRUN=1.E5
      NO=0
      DO 20 J=1,NFEAT
      DO 20 I=1,J
      NO=NO+1
      IF (SIGMA(NO).NE.0.0.AND.I.NE.J) GO TO 30
      A(NO)=SQRT(SIGMA(NO))
   20 CONTINUE
      GO TO 90
   30 IF (SIGMA(1).LT.0.0) GO TO 150
      A(1) = SQRT(SIGMA(1))
      L = 1
      DO 80 I=2,NFEAT
      I1 = I - 1
      M = 0
      M1 = L + 1
      N = 1
      DO 60 J=1,I1
      N = N + J - 1
      I = L + 1
      A(L) = SIGMA(L)
      IF (J.EQ.1) GO TO 50
      L1 = L - 1
      N1 = N
      DO 40 K=M1,L1
      A(L) = A(L) - A(N1)*A(K)
   40 N1 = N1 + 1
   50 M = M + J
   60 A(L) = A(L)/A(M)
      K = L
      L = L + 1
      A(L) = SIGMA(L)
      DO 70 J=M1,K
   70 A(L) = A(L) - A(J)*A(J)
      IF (A(L).LT.0.0) GO TO 150
   80 A(L) = SQRT(A(L))
   90 WRITE(6,10300)
      NOB=1
      NOE=1
  100 WRITE(6,10200) (SIGMA(J),J=NOB,NOE)
      IF (NOE.EQ.NF) GO TO 110
      NCB=NOB+1
      NOE=NOE+2
      IF (NOE.GT.NF) NOE=NF
      GO TO 100
  110 NOB=1
      NOE=1
      WRITE(6,10400)
  120 WRITE(6,10200) (A(J),J=NOB,NOE)
      IF (NOE.EQ.NF) GO TO 130
      NOB=NOB+1
      NOE=NOE+2
      IF (NOE.GT.NF) NOE=NF
      GO TO 120
  130 RETURN
  140 READ(5,10100) BLIM,ULIM
      RETURN
  150 WRITE(6,10500) (SIGMA(I),I=1,NF)
      STOP 104
10000 FORMAT(8I10)
10100 FORMAT(8F10.3)
10200 FORMAT('0',10G10.3/(1X,10G10.3))
10300 FORMAT('0COVARIANCE MATRIX'//)
10400 FORMAT('0TRANSFORMATION MATRIX'//)
10500 FORMAT(' COVARIANCE MATRIX IS NOT ',
     A  'POSITIVE DEFINITE.'/ (10G10.3))
      END
      SUBROUTINE NOISE(VAR,DATA,HOLD,ICONST,VALUE,TA,X,
     A  CV,TRUN,IERR,NO,NFEAT,NSAMP,BLIM,ULIM)
      DIMENSION VAR(1),DATA(NFEAT,1),HOLD(NFEAT,1),
     A  ICONST(1),VALUE(1),TA(1),X(4,1),CV(1)
      DO 10 J=1,NFEAT
      DO 10 K=1,4
      X(K,J)=0.0
   10 CONTINUE
      DO 30 J=1,NSAMP
      IF (IERR.NE.3) CALL NORM(ICONST,VALUE,
     A  TA,NFEAT,DATA(1,J),TRUN)
      IF (IERR.EQ.3) CALL UNIF(ICONST,VALUE,BLIM,ULIM,NFEAT)
      DO 20 K=1,NFEAT
      V=VALUE(K)
      X(1,K)=X(1,K)+V
      X(2,K)=X(2,K)+V*V
      X(3,K)=X(3,K)+V*V*V
      X(4,K)=X(4,K)+V*V*V*V
      IF (IERR.EQ.2) V=CV(K)*HOLD(K,J)*V
      DATA(K,J)=HOLD(K,J)+V
   20 CONTINUE
   30 CONTINUE
      FN=1./FLOAT(NSAMP)
      WRITE(6,10100) NO
      DO 50 K=1,NFEAT
      DO 40 L=1,4
      X(L,K)=X(L,K)*FN
   40 CONTINUE
      EMB=X(2,K)-X(1,K)*X(1,K)
      EMC=X(3,K)-3.*X(1,K)*X(2,K)+2.*X(1,K)*X(1,K)*X(1,K)
      EMD=X(4,K)-4.*X(1,K)*X(3,K)+6.*X(1,K)*X(1,K)*X(2,K)
     A    -3.*X(1,K)*X(1,K)*X(1,K)*X(1,K)
      CURT=EMD/(EMB*EMB)
      SIG=SQRT((EMB*NSAMP)/(NSAMP-1.))
      SKEW=EMC/(EMB*SQRT(EMB))
      WRITE(6,10200) K,X(1,K),SIG,SKEW,CURT
   50 CONTINUE
CDO 3000 J=1,NSAMP
CWRITE(6,10200) (DATA(I,J),I=1,NFEAT)
   60 CONTINUE
      RETURN
```

```
10000 FORMAT(1X,10G11.3)
10100 FORMAT(' FOR GROUPING NUMBER ',I5/
     A   4X,'VARIABLE',10X,'MEAN',9X,
     B 'STANDARD',9X, 'SKEWNESS',6X,'KURTOSIS'/
     C  34X,'DEVIATION'//)
10200 FORMAT(8X,I4,3X,4(F10.3,5X))
      END
      SUBROUTINE NORM(ICONST,VALUE,TA,NFEAT,X,TRUN)
      DIMENSION ICONST(1),VALUE(1),TA(1),X(1)
      DOUBLE PRECISION KR
      COMMON/SEED/ISEED
      N1=1
      DO 30 J=1,NFEAT
      ISEED=ICONST(J)
   10 X(J)=KR(J)
      ICONST(J)=ISEED
      V=0.0
      DO 20 I=1,J
      V=V+TA(N1)*X(I)
      N1=N1+1
   20 CONTINUE
      IF (ABS(V).GE.TRUN) GO TO 10
      VALUE(J)=V
   30 CONTINUE
      RETURN
      END
      SUBROUTINE UNIF(ICONST,VALUE,BLEVEL,ULEVEL,NFEAT)
      COMMON/SEED/ISEED
      DIMENSION ICONST(1),VALUE(1)
      DO 10 J=1,NFEAT
      ISEED=ICONST(J)
      U=RAN(J)
      VALUE(J)=(ULEVEL-BLEVEL)*U+BLEVEL
      ICONST(J)=ISEED
   10 CONTINUE
      RETURN
      END
      SUBROUTINE TALK(NFEAT,NSAMP)
C
C       PURPOSE -- PRINT OUT INFORMATION REGARDING OPTIONS
C       USED FOR THIS DENDROGRAM
C
C       FEBRUARY 21, 1977
C
C
      COMMON/ALWAYS/TITLE(30),DATE(5),TODAY(2),IFLAG,
     > NORM,MET,NSQUAR,INV,LOG,NGRP,ITRN
C
C
      WRITE(6,10000) NFEAT,NSAMP
C
      GO TO (10,20,30,40,50,60,70,80),IFLAG
   10 WRITE(6,10100)
```

```
      GO TO 90
   20 WRITE(6,10200)
      GO TO 90
   30 WRITE(6,10300)
      GO TO 90
   40 WRITE(6,10400)
      GO TO 90
   50 WRITE(6,10500)
      GO TO 90
   60 WRITE(6,10600)
      GO TO 90
   70 WRITE(6,10700)
      GO TO 90
   80 WRITE(6,10800)
   90 CONTINUE
C
      IF (NORM.EQ.0) GO TO 100
      GO TO (110,120,130,140,150,160),NORM
  100 WRITE(6,10900)
      GO TO 170
  110 WRITE(6,11000)
      GO TO 170
  120 WRITE(6,11100)
      GO TO 170
  130 WRITE(6,11200)
      GO TO 170
  140 WRITE(6,11300)
      GO TO 170
  150 WRITE(6,11400)
      GO TO 170
  160 WRITE(6,11500)
  170 CONTINUE
C
      GO TO (180,190,200,210,220,230,240),MET
  180 IF (IFLAG.EQ.7.OR.IFLAG.EQ.8) GO TO 250
      WRITE(6,11600)
      GO TO 250
  190 WRITE(6,11700)
      GO TO 250
  200 WRITE(6,11800)
      GO TO 250
  210 WRITE(6,11900)
      GO TO 250
  220 WRITE(6,12000)
      GO TO 250
  230 WRITE(6,12100)
      GO TO 250
  240 WRITE(6,12200)
  250 CONTINUE
C
C
      IF (INV.NE.0) WRITE(6,12400)
      WRITE(6,12500)
      RETURN
C
10000 FORMAT('0THERE ARE',I4,' VARIABLES AND',
     A  I4,' SAMPLES.')
10100 FORMAT(' CLUSTERING CRITERION -- SINGLE LINKAGE')
10200 FORMAT(' CLUSTERING CRITERION -- COMPLETE LINKAGE')
10300 FORMAT(' CLUSTERING CRITERION -- GROUP AVERAGE')
10400 FORMAT(' CLUSTERING CRITERION -- WEIGHTED AVERAGE')
10500 FORMAT(' CLUSTERING CRITERION -- CENTROID')
10600 FORMAT(' CLUSTERING CRITERION -- MEDIAN')
10700 FORMAT(' CLUSTERING CRITERION -- ',
     A  'MINIMUM INCREASE IN (WITHIN-CLUSTER) ',
     B  'SUM OF SQUARES')
10800 FORMAT(' CLUSTERING CRITERION -- ',
     A  'MINIMUM (WITHIN-CLUSTER) STANDARD DEVIATION')
10900 FORMAT(' NORMALIZATION        -- RAW DATA IS USED')
11000 FORMAT(' NORMALIZATION        -- ',
     A  'EACH VARIABLE IS DIVIDED BY ITS MAXIMUM')
11100 FORMAT(' NORMALIZATION        -- ',
     A  'EACH VARIABLE IS DIVIDED BY ITS ',
     B  'STANDARD DEVIATION.')
11200 FORMAT(' NORMALIZATION        -- ',
     A  'EACH VARIABLE IS MULTIPLIED BY AN INPUT WEIGHT.')
11300 FORMAT(' NORMALIZATION        -- ',
     A  'EACH VARIABLE  IS DIVIDED BY ITS',
     B  'STANDARD DEVIATION AND ',
     C  'MULTIPLIED BY AN INPUT WEIGHT.')
11400 FORMAT(' NORMALIZATION     --',
     A  'EACH VARIABLE IS DIVIDED',
     > ' BY ITS ROBUST STANDARD DEVIATION.')
11500 FORMAT(' NORMALIZATION     --EACH VARIABLE IS DIVIDED
',
     A 'BY ITS ROBUST STANDARD DEVIATION ',
     B  'AND MULTIPLIED BY THE'
     B ' ABOVE WEIGHT.')
11600 FORMAT(' METRIC               -- EUCLIDEAN (SQUARED)')
11700 FORMAT(' METRIC               -- EUCLIDEAN ',
     A  '(NOT SQUARED)')
11800 FORMAT(' METRIC               -- ',
     A  'NORMALIZED EUCLIDEAN (SQUARED)')
11900 FORMAT(' METRIC               -- ',
     A  'NORMALIZED EUCLIDEAN (NOT SQUARED)')
12000 FORMAT(' METRIC               -- ',
     A  '( 1.-ABS(PEARSON CORRELATION) )')
12100 FORMAT(' METRIC               -- ',
     A  '( 1.-ABS(SPEARMAN CORRELATION) )')
12200 FORMAT(' METRIC               -- CITY BLOCK')
12300 FORMAT('0DATA SET IS THE SAME AS THAT ',
     A  'USED FOR THE PREVIOUS DENDROGRAM.')
12400 FORMAT('0(DATA SET IS INVERTED FROM THE USUAL.)')
12500 FORMAT('0')
      END
C
```

```
C
C
      INTEGER FUNCTION DIMEN(MANY)
C
C       PURPOSE -- KEEP TRACK OF DIMENSIONS
C       FEBRUARY 21, 1977
C
      COMMON/HOLD/NSIZE,A(1)
      DATA KOUNT/1/
      DIMEN=KOUNT
      KOUNT=KOUNT+MANY
      IF (KOUNT.GT.NSIZE) WRITE(6,10000) NSIZE,KOUNT
      RETURN
C
      ENTRY LOOSE(MANY)
      KOUNT=MANY
      RETURN
C
10000 FORMAT(' AVAILABLE SIZE=',I5,
    A  ' BUT YOU NEED',I6,'.   ERROR!')
      END
C
C
C
      SUBROUTINE INPUT(DATA,SAMPLE,VAR,CUT,HOLD,
     A  NFEAT,NSAMP)
C
C       PURPOSE -- READ IN THE DATA SET (THE INTENTION IS TO
C       BE  COMPATIBLE WITH RECOG INPUT).  ALSO PRINT THE
C       DATA SET.
C
C       FEBRUARY 21, 1977
C
C
      COMMON/ALWAYS/TITLE(30),DATE(5),TODAY(2),IFLAG,
     > NORM,MET,NSQUAR,INV,LOG,NGRP,ITRN
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),TITCLS(5),
     >  PTALK(20)
      DIMENSION DATA(NFEAT,1),SAMPLE(NSNAM,1),VAR(NVNAM,1),
     > CUT(1),HOLD(NFEAT,1)
      DIMENSION FMT(20),FMT2(20)
      DATA BD/4HBCD /
C
C
C
C
      IF (INV.NE.0) GO TO 50
C
C *** READ FORMAT INFORMATION
C
      READ(05,10000) FMT
      IF (FMT(1).NE.BD) GO TO 20
C
```

```
C *** ONE VERSION OF RECOG'S INPUT
      READ(05,10000)FMT
      READ(05,10000)FMT2
      READ(05,10100)TITLE
C         NOTE THAT THIS IS ACTUALLY THREE CARDS
C
C       SORT OUT TITLE -- ELIMINATE BLANK SPACES --
      CALL SORT(TITLE)
C
      DO 10 I=1,NSAMP
      READ(05,FMT) (DATA(J,I),J=1,NFEAT)
      READ(05,FMT2) PP,CL, (SAMPLE(J,I),J=1,NSNAM)
   10 CONTINUE
      GO TO 40
C
C
C *** ANOTHER VERSION OF RECOG'S INPUT
   20 READ(05,10100)  TITLE
      CALL SORT(TITLE)
      DO 30 I=1,NSAMP
      READ(05,FMT) (DATA(J,I),J=1,NFEAT),PP,CL,
     A  (SAMPLE(J,I),J=1,NSNAM)
   30 CONTINUE
C
   40 WRITE(6,10800)  TITLE
      READ(05,10000) ((VAR(J,I),J=1,NVNAM),I=1,NFEAT)
      READ(5, 10900)  (CUT(K),K=1,NFEAT)
      GO TO 70
C
C
C
C *** INVERTED READ
   50 READ(05,10000)FMT
      READ(05,10000)FMT
      READ(05,10000)FMT2
      READ(05,10100)TITLE
      CALL SORT(TITLE)
      WRITE(06,10800)TITLE
C
      DO 60 J=1,NFEAT
      READ(05,FMT) (DATA(J,I),I=1,NSAMP)
   60 READ(05,FMT2) PP,CL, (VAR(K,J),K=1,NVNAM)
      READ(5, 10000) ((SAMPLE(K,I),K=1,NSNAM),I=1,NSAMP)
C
C
C
C *** PRINT OUT THE INPUT
C
   70 IF (NVNAM.GT.1) GO TO 80
      WRITE(6,10200) (VAR(1,I),I=1,NFEAT)
      GO TO 130
   80 N=(NFEAT-1)/10+1
      MIN=1
      MAX=10
      WRITE(6,10400)
      DO 120 K=1,N
      IF (MAX.GT.NFEAT) MAX=NFEAT
      MINL=1
      MAXL=2
      DO 90 KK=1,NVNAM,2
      WRITE(6,10300) ((VAR(J,I),J=MINL,MAXL),I=MIN,MAX)
      MINL=MINL+2
      MAXL=MAX+2
      IF (MAXL.GT.NVNAM) GO TO 100
   90 CONTINUE
      GO TO 110
  100 WRITE(6,10200) (VAR(NVNAM,I),I=MIN,MAX)
  110 CONTINUE
      MIN=MIN+10
      MAX=MAX+10
  120 CONTINUE
  130 CONTINUE
      WRITE(6,10500)
      IF (NSNAM.GT.1) GO TO 150
      DO 140 I=1,NSAMP
  140 WRITE(6,10600) SAMPLE(1,I),(DATA(J,I),J=1,NFEAT)
      GO TO 170
  150 DO 160 I=1,NSAMP
  160 WRITE(6,10700) (SAMPLE(K,I),K=1,2),
     A  (DATA(J,I),J=1,NFEAT)
  170 CONTINUE
      DO 180 I=1,NSAMP
      DO 180 J=1,NFEAT
      HOLD(J,I)=DATA(J,I)
  180 CONTINUE
C
C
      RETURN
C
C
C
10000 FORMAT(20A4)
10100 FORMAT(10A4)
10200 FORMAT('0      VARIABLES'/
     A  (10X,A4,7X,A4,7X,A4,7X,A4,7X,A4,7X,A4,
     B  7X,A4,7X,A4,7X,A4,7X,A4))
10300 FORMAT(12X,2A4,3X,2A4,3X,2A4,3X,2A4,3X,
     A  2A4,3X,2A4,3X,2A4,3X,2A4,3X,2A4,3X,2A4)
10400 FORMAT('0      VARIABLES')
10500 FORMAT('0SAMPLES')
10600 FORMAT('0',A4,2X,10G11.3/(7X,10G11.3))
10700 FORMAT('0',2A4,10G11.3/(9X,10G11.3))
10800 FORMAT(1X,10A4)
10900 FORMAT(8F10.0)
      END
C
```

```
C
C
      SUBROUTINE SORT(TITLE)
      DIMENSION TITLE(30)
      DATA BLANK/'    '/
C
      DO 10 J=1,10
      IF (TITLE(J).NE.BLANK) GO TO 60
   10 CONTINUE
      DO 20 J=11,20
      IF (TITLE(J).NE.BLANK) GO TO 40
   20 CONTINUE
      DO 30 J=1,10
      TITLE(J)=TITLE(J+20)
   30 TITLE(J+20)=BLANK
      GO TO 90
   40 DO 50 J=1,20
      TITLE(J)=TITLE(J+10)
   50 TITLE(J+10)=BLANK
      GO TO 90
   60 DO 70 J=11,20
      IF (TITLE(J).NE.BLANK) GO TO 90
   70 CONTINUE
      DO 80 J=11,20
      TITLE(J)=TITLE(J+10)
   80 TITLE(J+10)=BLANK
   90 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE NORMAL(WEIGHT,DATA,VAR,X,CUT,
     A  NFEAT,NSAMP,MTIME)
C
C
C       PURPOSE -- NORMALIZE THE DATA
C                  NORM=0       UNNORMALIZED
C                  NORM=1       DIVIDE BY MAX
C                  NORM=2       DIVIDE BY STANDARD DEVIATION
C                  NORM=3       MULTIPLY BY INPUT WEIGHTS
C                  NORM=4       DO BOTH 2 AND 3
C                  NORM=5       DIVIDE BY ROBUST STANDARD
C                               DEVIATION
C                  NORM=6       DO 5 AND 3
C
C       FEBRUARY 21, 1977
C
C
      COMMON/ALWAYS/TITLE(30),DATE(5),TODAY(2),IFLAG,
     > NORM,MET,NSQUAR,INV,LOG,NGRP,ITRN
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),TITCLS(5),
     A  PTALK(20)
```

```
C
      DIMENSION WEIGHT(1),DATA(NFEAT,1),VAR(NVNAM,1),X(1),
     A  CUT(1),WEIG(8)
C
C
      DO 10 J=1,NSAMP
      DO 10 K=1,NFEAT
      IF (DATA(K,J).LT.CUT(K)) DATA(K,J)=CUT(K)
   10 CONTINUE
      GO TO (80,20,40,60),ITRN
      GO TO 80
   20 DO 30 J=1,NSAMP
      DO 30 K=1,NFEAT
      DATA(K,J)=ALOG(DATA(K,J))
   30 CONTINUE
      GO TO 80
   40 DO 50 J=1,NFEAT
      DO 50 K=1,NSAMP
      DATA(K,J)=EXP(DATA(K,J))
   50 CONTINUE
      GO TO 80
   60 DO 70 J=1,NSAMP
      DO 70 K=1,NFEAT
      DATA(K,J)=SQRT(DATA(K,J))
   70 CONTINUE
   80 CONTINUE
C
      IF (NORM.EQ.0) GO TO 90
      IF (NORM.GT.6) GO TO 90
      GO TO (100,160,220,160,290,290),NORM
C
C
C *** NC NORMALIZATION
   90 RETURN
C
C *** NORMALIZE BY DIVIDING BY MAX
  100 WRITE(6,10000)
      DO 130 IFEAT=1,NFEAT
      F=0.
      DO 110 ISAMP=1,NSAMP
      IF (F.GT.DATA(IFEAT,ISAMP)) GO TO 110
      F=DATA(IFEAT,ISAMP)
  110 CONTINUE
      F=1./F
      WEIGHT(IFEAT)=F
      WRITE(6,10100) IFEAT,F,(VAR(J,IFEAT),J=1,NVNAM)
      DO 120 ISAMP=1,NSAMP
  120 DATA(IFEAT,ISAMP)=DATA(IFEAT,ISAMP)*F
  130 CONTINUE
      F=1.0/NFEAT
      DO 150 IFEAT=1,NFEAT
      WEIGHT(IFEAT)=F
      DO 140 ISAMP=1,NSAMP
  140 DATA(IFEAT,ISAMP)=DATA(IFEAT,ISAMP)*F
  150 CONTINUE
      RETURN
C
C *** NORMALIZE BY DIVIDING BY STANDARD DEVIATION
  160 WRITE(6,10400)
      DO 190 IFEAT=1,NFEAT
      F=0.
      G=0.
      DO 170 ISAMP=1,NSAMP
      F=F+DATA(IFEAT,ISAMP)
  170 G=G+DATA(IFEAT,ISAMP)**2
      G=(G-F*F/NSAMP)/(NSAMP-1)
      IF (G.LE.1.E-6) G=1.0
      G=SQRT(G)
      F=1./G
      WRITE(6,10500)IFEAT,G,F,(VAR(J,IFEAT),J=1,NVNAM)
      DO 180 ISAMP=1,NSAMP
  180 DATA(IFEAT,ISAMP)=DATA(IFEAT,ISAMP)*F
  190 CONTINUE
      IF (NORM.EQ.4) GO TO 220
      F=1.0/NFEAT
      DO 210 IFEAT=1,NFEAT
      WEIGHT(IFEAT)=F
      DO 200 ISAMP=1,NSAMP
  200 DATA(IFEAT,ISAMP)=DATA(IFEAT,ISAMP)*F
  210 CONTINUE
      RETURN
C
C *** NORMALIZE BY MULTIPLYING BY INPUT WEIGHTS
  220 N=(NFEAT-1)/8+1
      IFEAT=0
      IF (MTIME.NE.1) GO TO 260
      DO 250 II=1,N
      READ(5,10200) (WEIG(I),I=1,8)
      DO 240 I=1,8
      IFEAT=IFEAT+1
      IF (IFEAT.GT.NFEAT) GO TO 240
      WEIGHT(IFEAT)=WEIG(I)
      WRITE(6,10100)IFEAT,WEIG(I),(VAR(J,IFEAT),J=1,NVNAM)
      F=WEIG(I)
      DO 230 ISAMP=1,NSAMP
  230 DATA(IFEAT,ISAMP)=DATA(IFEAT,ISAMP)*F
  240 CONTINUE
  250 CONTINUE
      RETURN
  260 DO 280 IFEAT=1,NFEAT
      F=WEIGHT(IFEAT)
      DO 270 ISAMP=1,NSAMP
      DATA(IFEAT,ISAMP)=DATA(IFEAT,ISAMP)*F
  270 CONTINUE
  280 CONTINUE
      RETURN
```

```
C
C *** NORMALIZE BY DIVIDING BY ROBUST STANDARD DEVIATION
  290 IL=NSAMP/2
      DO 450 I=1,NFEAT
      DO 300 J=1,NSAMP
      X(J)=DATA(I,J)
  300 CONTINUE
      DO 320 J=2,NSAMP
      M=J
  310 IF (X(M).GE.X(M-1)) GO TO 320
      A=X(M-1)
      X(M-1)=X(M)
      X(M)=A
      M=M-1
      IF (M.GT.1) GO TO 310
  320 CONTINUE
      WRITE(06,10600)
      DO 330 J=1,NSAMP
      IF (X(J).GT.CUT(I)) GO TO 340
  330 CONTINUE
  340 IC=J-1
      IF (IC.GT.IL) GO TO 380
      FN=NSAMP
      IA=.05*FN
      IB=.95*FN
      ID=AMAX0(IC,IA,1)
      ID1=ID+1
      IR=0
  350 IR=IR+1
      IF (ID1+IR.GT.NSAMP) GO TO 400
      IF (X(ID1).EQ.X(ID1+IR)) GO TO 350
      L=ID+IR
      PB=FLOAT(L)/FLOAT(NSAMP+1)
      IR=0
  360 IR=IR+1
      IF (IB-IR.LE.0) GO TO 410
      IF (X(IB).EQ.X(IB-IR)) GO TO 360
      IU=IB-IR
      PT=FLOAT(IU)/FLOAT(NSAMP+1)
      IUP=NSAMP-ID
      XMEAN=0.0
      DO 370 J=ID1,IUP
      XMEAN=XMEAN+X(J)
  370 CONTINUE
      FN=1./FLOAT(NSAMP-2*ID)
      XMEAN=XMEAN*FN
      CALL MDNRIS(PB,X1,IER)
      B=-X1
      CALL MDNRIS(PT,X2,IER)
      T=X2
      STDV=((X(IU)+X(IU+1))-(X(L)+X(L+1)))/(2.*(B+T))
      WRITE(6,10700) I,XMEAN,STDV,(VAR(J,I),J=1,NVNAM)
      WT=1./STDV
      GO TO 430
  380 WRITE(6,10800) I,(VAR(J,I),J=1,NVNAM)
C       CALCULATE REGULAR STANDARD DEVIATION WHEN THERE ARE
C       NOT ENOUGH POINTS FOR ROBUST
      F=0.
      G=0.
      DO 390 J=1,NSAMP
      F=F+X(J)
      G=G+X(J)*X(J)
  390 CONTINUE
      XMEAN=F/NSAMP
      G=(G-F*F/NSAMP)/(NSAMP-1)
      STDV=SQRT(G)
      WT=1./STDV
      GO TO 430
  400 WRITE(6,10900)
      GO TO 420
  410 WRITE(6,11000)
  420 XMEAN=0.0
      STDV=0.0
      WT=0.0
      GO TO 430
  430 DO 440 J=1,NSAMP
      DATA(I,J)=DATA(I,J)*WT
  440 CONTINUE
  450 CONTINUE
      IF (NORM.EQ.6) GO TO 220
      RETURN
C
C
C
10000 FORMAT('0VARIABLE          WEIGHT (DIVISION BY MAX)')
10100 FORMAT(1X,I4,10X,E16.4,T7,2A4/(6X,2A4))
10200 FORMAT(8F10.2)
10300 FORMAT('0VARIABLE         WEIGHT (AS INPUT ON CARDS)')
10400 FORMAT('0VARIABLE      STANDARD DEVIATION    WEIGHT',
     > '(1./ST.DEV.)')
10500 FORMAT(1X,I3,6X,E16.4,3X,E15.4,T7,2A4/(6X,2A4))
10600 FORMAT('0 VARIABLE     MEAN   ROBUST ST.DEV. NAME')
10700 FORMAT(I8,F12.3,F12.3,3X,(5A4))
10800 FORMAT(' TOO MUCH DATA BELOW DETECTION ',
     A 'LIMIT FOR VARIABLE', I5,1X,5A4)
10900 FORMAT(' CANNOT FIND IR')
11000 FORMAT(' CANNOT FIND IU')
      END
C
C
C
      SUBROUTINE METRIC (DATA,SQUA,XX,YY,IR,R,THRESH,EXTRA,
     > SAMPLE,NSAMP,NFEAT)
C
C       PURPOSE -- INITIALIZE THE METRIC ARRAY "SQUA"
C        (ALSO THE AUXILLIARY ARRAY "EXTRA" IF NEEDED)
```

```
C
C       FEBRUARY 21, 1977
C
      DIMENSION DATA(NFEAT,1),SQUA(1),XX(1),YY(1),IR(1),
     A  R(1),THRESH(1),EXTRA(1),SAMPLE(NSNAM,1)
C
      COMMON/ALWAYS/TITLE(37),IFLAG,NORM,MET,NSQUAR,
     A   INV,LOG,NGRP,ITRN
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),TITCLS(5),
     A  PTALK(20)
      INTEGER*2 IR
C
      DATA EPS/0.01/
      DIMENSION URE(7)
      DATA URE/'  PH','M-AK','  BC',' U/U',
     A  'T-AK', 'P-AK','PH¬P'/
      DATA NOLOG/7/
C
C
C
      AMAX=0.
      LOC=0
      ILIMIT=NSAMP-1
      GO TO (10,10,10,10,50,50,260),MET
C
C *** EUCLIDEAN METRIC
   10 DO 30 I=1,ILIMIT
      ILIM=I+1
      DO 30 J=ILIM,NSAMP
      DIST=0.
      DO 20 K=1,NFEAT
      DIST=DIST+(DATA(K,I)-DATA(K,J))**2
   20 CONTINUE
      IF (DIST.GT.AMAX) AMAX=DIST
      LOC=LOC+1
      SQUA(LOC)=DIST
   30 CONTINUE
      SAMAX=SQRT(AMAX)
      CALL CHANGE (SQUA,AMAX,SAMAX,NSAMP)
      WRITE(6,10000)AMAX
      IF (IFLAG.NE.8) GO TO 290
      DO 40 I=1,NSAMP
   40 EXTRA(I)=0.
      GO TO 290
C
C
C *** CORRELATION-BASED METRICS
   50 CONTINUE
      IF (LOG.NE.0) GO TO 70
      DO 60 I=1,NSAMP
   60 THRESH(I)=-1.E25
      GO TO 150
   70 GO TO (80,100,140),LOG
   80 DO 90 I=1,NSAMP
   90 THRESH(I)=0.
      GO TO 150
  100 DO 130 I=1,NSAMP
      DO 110 J=1,NOLOG
      IF (SAMPLE(1,I).EQ.URE(J)) GO TO 120
  110 CONTINUE
      THRESH(I)=-10.
      GO TO 130
  120 THRESH(I)=0.
  130 CONTINUE
      GO TO 150
  140 READ(05,10100) (THRESH(I),I=1,NSAMP)
  150 WRITE(06,10200) (SAMPLE(1,I),THRESH(I),I=1,NSAMP)
      NSQUAR=1
C
      IF (MET.EQ.6) GO TO 200
C
C
C *** PEARSON CORRELATION-BASED METRIC
      DO 190 I=1,ILIMIT
      ILIM=I+1
      DO 190 J=ILIM,NSAMP
      NF=0
      DO 160 K=1,NFEAT
      IF (DATA(K,I).LT.THRESH(I)) GO TO 160
      IF (DATA(K,J).LT.THRESH(J)) GO TO 160
      NF=NF+1
      XX(NF)=DATA(K,I)
      YY(NF)=DATA(K,J)
  160 CONTINUE
      IF (NF.LT.10) GO TO 180
      A=0.
      B=0.
      C=0.
      D=0.
      E=0.
      DO 170 K=1,NF
      A=A+XX(K)*YY(K)
      B=B+XX(K)*XX(K)
      C=C+YY(K)*YY(K)
      D=D+XX(K)
      E=E+YY(K)
  170 CONTINUE
      B=B-D*D/NF
      C=C-E*E/NF
      A=A-D*E/NF
      A=A/SQRT(B*C)
      IF (A.LT.0.) A=-A
      LOC=LOC+1
      DIST=1.-A
      SQUA(LOC)=DIST
      IF (DIST.GT.AMAX) AMAX=DIST
      GO TO 190
  180 LOC=LOC+1
      SQUA(LOC)=1.0
  190 CONTINUE
      WRITE(6,10000)AMAX
      GO TO 290
C
C
C *** SPEARMAN CORRELATION-BASED METRIC
  200 CONTINUE
      DO 250 I=1,ILIMIT
      ILIM=I+1
      DO 240 J=ILIM,NSAMP
      NF=0
      DO 210 K=1,NFEAT
      IF (DATA(K,I).LT.THRESH(I)) GO TO 210
      IF (DATA(K,J).LT.THRESH(J)) GO TO 210
      NF=NF+1
      XX(NF)=DATA(K,I)
      YY(NF)=DATA(K,J)
  210 CONTINUE
      IF (NF.LT.10) GO TO 230
      D=NF*(NF+1.)*(NF+1.)/4.
      CALL RANK(XX,NF,IR,R,EPS)
      CALL RANK(YY,NF,IR,R,EPS)
      A=0.
      B=0.
      C=0.
      DO 220 K=1,NF
      A=A+XX(K)*YY(K)
      B=B+XX(K)*XX(K)
  220 C=C+YY(K)*YY(K)
      A=A-D
      B=B-D
      C=C-D
      A=A/SQRT(B*C)
      IF (A.LT.0.) A=-A
      LOC=LOC+1
      DIST=1.-A
      SQUA(LOC)=DIST
      IF (DIST.GT.AMAX) AMAX=DIST
      GO TO 240
  230 LOC=LOC+1
      SQUA(LOC)=1.0
  240 CONTINUE
  250 CONTINUE
      WRITE(6,10000)AMAX
      GO TO 290
C
C
C *** CITY BLOCK DISTANCE
  260 CONTINUE
      NSQUAR=1
      DO 280 I=1,ILIMIT
      ILIM=I+1
      DO 280 J=ILIM,NSAMP
      DIST=0.
      DO 270 K=1,NFEAT
  270 DIST=DIST+ABS(DATA(K,I)-DATA(K,J))
      IF (DIST.GT.AMAX) AMAX=DIST
      LOC=LOC+1
      SQUA(LOC) =DIST
  280 CONTINUE
      SAMAX=AMAX
      WRITE(6,10000) SAMAX
C
  290 RETURN
C
C
C
10000 FORMAT('0MAXIMUM DISTANCE IS',F10.4)
10100 FORMAT(8F10.2)
10200 FORMAT('0NAME THRESHOLD'/(1X,A4,F10.2))
      END
C
C
C
      SUBROUTINE CHANGE(SQUA,AMAX,SAMAX,NSAMP)
C
C       PURPOSE -- CHANGE FROM METRIC 1 TO METRIC 2, 3, OR 4
C
C       FEBRUARY 21, 1977
C
      DIMENSION SQUA(1)
C
      COMMON/ALWAYS/TITLE(37),IFLAG,NORM,MET,NSQUAR,
     A  INV,LOG,NGRP,ITRN
C
C
C
      NT=NSAMP*(NSAMP-1)/2
      GO TO (10,40,60,80),MET
C
   10 IF (IFLAG.EQ.7) GO TO 20
      IF (NSQUAR.EQ.0) AMAX=SAMAX
      RETURN
   20 DO 30 I=1,NT
   30 SQUA(I)=SQUA(I)*0.5
      AMAX=AMAX*0.5
      IF (NSQUAR.EQ.0) AMAX=SQRT(AMAX)
      RETURN
C
   40 CONTINUE
      NSQUAR=1
      DO 50 I=1,NT
   50 SQUA(I)=SQRT(SQUA(I))
```

```
      AMAX=SQRT(AMAX)
      RETURN
C
   60 CONTINUE
      DO 70 I=1,NT
   70 SQUA(I)=SQUA(I)/AMAX
      AMAX=1.
      RETURN
C
   80 CONTINUE
      NSQUAR=1
      DO 90 I=1,NT
   90 SQUA(I)=SQRT(SQUA(I))/SAMAX
      AMAX=1.
      RETURN
C
      END
C
C
C
      SUBROUTINE RANK (X,N,IR,R,EPS)
C
C       PURPOSE -- ORDER THE SAMPLES ACCORDING TO RANK
C
C       FEBRUARY 21, 1977
C
C
      DIMENSION X(1),IR(1),R(1)
      INTEGER*2 IR
C                    SAVE X VECTOR AND INITIALIZE THE
C                    PERMUTATION VECTOR
      DO 10 I = 1,N
      R(I) = X(I)
   10 IR(I) = I
C                    SORT ELEMENTS OF VECTOR R INTO
C                    ASCENDING SEQUENCE SAVING
C                    PERMUTATIONS
      CALL VSORTP (R,N,IR)
      N1 = N-1
      L = 1
   20 DO 60 J = L,N1
      JJ = J
      Y = R(J)
      IF (ABS(Y-R(J+1)).GT.EPS) GO TO 60
C                    COUNT THE NUMBER OF TIES
      K = 1
      J2 = J+2
      IF (J2.GT.N) GO TO 40
      DO 30 I = J2,N
      IF (ABS(Y-R(I)).GT.EPS) GO TO 40
   30 K = K+1
   40 Y = J+.5*K
      K1 = K+1
      DO 50 I = 1,K1
      JJ = J+I-1
   50 X(IR(JJ)) = Y
      GO TO 70
   60 X(IR(J)) = J
   70 L = JJ+1
      IF (L.LE.N1) GO TO 20
      IF (L.EQ.N) X(IR(N)) = N
      RETURN
      END
C
C
C
      SUBROUTINE VSORTP (A,LA,IR)
C
      DIMENSION A(1),IU(21),IL(21),IR(1)
      INTEGER*2 IR
C
      M=1
      I=1
      J=LA
      R=.375
   10 IF (I.EQ.J) GO TO 100
   20 IF (R.GT..5898437) GO TO 30
      R=R+3.90625E-2
      GO TO 40
   30 R=R-.21875
   40 K=I
C                    SELECT A CENTRAL ELEMENT OF THE
C                    ARRAY AND SAVE IT IN LOCATION T
      IJ=I+(J-I)*R
      T=A(IJ)
      IT=IR(IJ)
C                    IF FIRST ELEMENT OF ARRAY IS GREATER
C                    THAN T, INTERCHANGE WITH T
      IF (A(I).LE.T) GO TO 50
      A(IJ)=A(I)
      A(I)=T
      T=A(IJ)
      IR(IJ)=IR(I)
      IR(I)=IT
      IT=IR(IJ)
   50 L=J
C                    IF LAST ELEMENT OF ARRAY IS LESS THAN
C                    T, INTERCHANGE WITH T
      IF (A(J).GE.T) GO TO 70
      A(IJ)=A(J)
      A(J)=T
      T=A(IJ)
      IR(IJ)=IR(J)
      IR(J)=IT
      IT=IR(IJ)
C                    IF FIRST ELEMENT OF ARRAY IS GREATER
```

```
C                    THAN T, INTERCHANGE WITH T
      IF (A(I).LE.T) GO TO 70
      A(IJ)=A(I)
      A(I)=T
      T=A(IJ)
      IR(IJ)=IR(I)
      IR(I)=IT
      IT=IR(IJ)
      GO TO 70
   60 TT=A(L)
      A(L)=A(K)
      A(K)=TT
      ITT=IR(L)
      IR(L)=IR(K)
      IR(K)=ITT
C                    FIND AN ELEMENT IN THE SECOND HALF OF
C                    THE ARRAY WHICH IS SMALLER THAN T
   70 L=L-1
      IF (A(L).GT.T) GO TO 70
C                    FIND AN ELEMENT IN THE FIRST HALF OF
C                    THE ARRAY WHICH IS GREATER THAN T
   80 K=K+1
      IF (A(K).LT.T) GO TO 80
C                    INTERCHANGE THESE ELEMENTS
      IF (K.LE.L) GO TO 60
C                    SAVE UPPER AND LOWER SUBSCRIPTS OF
C                    THE ARRAY YET TO BE SORTED
      IF (L-I.LE.J-K) GO TO 90
      IL(M)=I
      IU(M)=L
      I=K
      M=M+1
      GO TO 110
   90 IL(M)=K
      IU(M)=J
      J=L
      M=M+1
      GO TO 110
C                    BEGIN AGAIN ON ANOTHER PORTION OF
C                    THE UNSORTED ARRAY
  100 M=M-1
      IF (M.EQ.0) RETURN
      I=IL(M)
      J=IU(M)
  110 IF (M.GT.21) WRITE(6,10000)M
      IF (J-I.GE.1) GO TO 40
      IF (I.EQ.1) GO TO 10
      I=I-1
  120 I=I+1
      IF (I.EQ.J) GO TO 100
      T=A(I+1)
      IT=IR(I+1)
      IF (A(I).LE.T) GO TO 120
      K=I
  130 A(K+1)=A(K)
      IR(K+1)=IR(K)
      K=K-1
      IF (T.LT.A(K)) GO TO 130
      A(K+1)=T
      IR(K+1)=IT
      GO TO 120
C
10000 FORMAT(' IN VSORTP, M=',I3)
      END
C
C
C
      SUBROUTINE CLUST(DISTAN,KLUSTR,MARRAY,JARRAY,
     A  DATA,MCEL,JCEL,ONDCL,INDCL,SQUA,EXTRA,
     B  KNT,IGRP,NNFEAT,NNSAMP)
C
C        PURPOSE -- DO THE CLUSTER ANALYSIS !!
C
C        FEBRUARY 21, 1977
C
C
      DIMENSION DISTAN(1),KLUSTR(2,1),MARRAY(1),JARRAY(1),
     A  DATA(NNFEAT,1),MCEL(1),JCEL(1),SQUA(1),EXTRA(1),
     B  ONDCL(1),INDCL(1),IGRP(1)
      INTEGER*2 KLUSTR,MARRAY,JARRAY,MCEL,JCEL,
     A  ONDCL,INDCL,KNT(NGRP,1)
C
      COMMON/ALWAYS/TITLE(37),IFLAG,NORM,MET,NSQUAR,
     A  INV,LOG,NGRP,ITRN
C
C
      NSAMP=NNSAMP
      NFEAT=NNFEAT
C
C
C  INITIALIZE TO NSAMP CLASSES
C
      NCL=NSAMP
      ITER=0
      NCLUST=0
      DO 10 I=1,NSAMP
      MCEL(I)=1
      JCEL(I)=0
      ONDCL(I)=I
      MARRAY(I)=I
      JARRAY(I)=0
      INDCL(I)=0
   10 CONTINUE
C
C
C ***  START LOOP
```

```
C
   20 CONTINUE
      ITER=ITER+1
C
C  FIND CANDIDATES FOR CLUSTERING
C
      ISUB=0
      JSUB=0
      AMIN=1.E35
      CALL INTER(SQUA,EXTRA,MARRAY,ONDCL,MCEL,
     A  NSAMP,XNI,XNJ,ISUB,JSUB,AMIN,IFLAG,NCL,JJJ,ITER)
      SAMIN=AMIN
      IF (NSQUAR.EQ.0) SAMIN=SQRT(AMIN)
      KLUSTR(1,ITER)=ISUB
      KLUSTR(2,ITER)=JSUB
      DISTAN(ITER)=SAMIN
C
C  ISUB AND JSUB ARE THE CLUSTER NUMBERS WHICH MINIMIZE
C       THE CRITERION.
C  FORM A CLUSTER
C
      NCLUST=NCLUST+1
      ILIM=MCEL(ISUB)
      XNI=ILIM
      INDIS=ONDCL(ISUB)
C               NUMBER OF SAMPLES IN CLUSTER ISUB
      JLIM=MCEL(JSUB)
      XNJ=JLIM
      INDJS=ONDCL(JSUB)
C               NUMBER OF SAMPLES IN CLUSTER JSUB
      JJJ=MARRAY(INDJS)
C               NUMBER OF THE FIRST SAMPLE IN JSUB
C
C  LOAD INTO CLUSTER ARRAY
C
      INDEX=0
      INDCL(NCLUST)=INDEX+1
      DO 30 I=1,ILIM
      INDEX=INDEX+1
   30 JARRAY(INDEX)=MARRAY(INDIS+I-1)
      DO 40 J=1,JLIM
      INDEX=INDEX+1
   40 JARRAY(INDEX)=MARRAY(INDJS+J-1)
      JCEL(NCLUST)=ILIM+JLIM
C
C  COPY REST OF MATRIX INTO JARRAY.
C
      DO 60 I=1,NCL
      IF (I.EQ.ISUB) GO TO 60
      IF (I.EQ.JSUB) GO TO 60
      NCLUST=NCLUST+1
      INDOLD=ONDCL(I)
      INDCL(NCLUST)=INDEX+1
      ILIM=MCEL(I)
      AMIN=0.
      DO 50 J=1,ILIM
      INDEX=INDEX+1
      JARRAY(INDEX)=MARRAY(INDOLD+J-1)
   50 CONTINUE
      JCEL(NCLUST)=ILIM
   60 CONTINUE
C
      IST=INDCL(I)-1
   70 CONTINUE
C
C  REINITIALIZE
C
      DO 80 J=1,NSAMP
   80 MARRAY(J)=0
C
C  COPY JARRAY INTO MARRAY
C
      DO 90 I=1,NCLUST
      JLIM=JCEL(I)
      MCEL(I)=JLIM
      JCEL(I)=0
      ONDCL(I)=INDCL(I)
   90 CONTINUE
      DO 100 J=1,NSAMP
      MARRAY(J)=JARRAY(J)
      JARRAY(J)=0
  100 CONTINUE
      NCL=NCLUST
      NCLUST=0
      DO 110 IN=1,NGRP
      IF (NCL.EQ.IGRP(IN)) GO TO 120
  110 CONTINUE
      GO TO 20
C
C *** END OF LOOP
C
C
  120 CONTINUE
      JLIM=NSAMP
      WRITE(06,10400) (MCEL(I),I=1,NCL)
      WRITE(06,10500) (MARRAY(J),J=1,JLIM)
      ISTART=1
      DO 160 J=1,NCL
      KK=MCEL(J)
      IF (KK.LT.2) GO TO 150
      IOK=MCEL(J)+ISTART-1
      DO 140 K=2,KK
      NOJ=ISTART+K-1
      NMJ=MARRAY(NOJ-1)
      DO 140 NOK=NOJ,IOK
      NMK=MARRAY(NOK)
      IF (NMJ.LT.NMK) GO TO 130
      NOA=((NMJ-1)*(NMJ-2))/2+NMK
      GO TO 140
  130 NOA=((NMK-1)*(NMK-2))/2+NMJ
  140 KNT(IN,NOA)=KNT(IN,NOA)+1
  150 ISTART=ISTART+KK
  160 CONTINUE
      IF (IN.EQ.1) RETURN
      GO TO 20
      RETURN
C
C
C
10000 FORMAT(1X,'ITER=',I5,2X,2I5,' AMIN=',F10.4)
10100 FORMAT(' ITERATION',I4,'  NUMBER OF CLUSTERS  ',I4)
10200 FORMAT(1X,'CLUSTER NUMBER',I4,
     A  ' SAMPLES ',25I4, /,(24X,25I4))
10300 FORMAT(' ILIM = ',I4)
10400 FORMAT(10X,'FINAL ORDERING  '/
     A  ' CLUSTERS END AT ',(12I5))
10500 FORMAT(20I5)
      END
C
C
      SUBROUTINE INTER(SQUA,EXTRA,MARRAY,ONDCL,MCEL,
     A  NSAMP,XNI,XNJ,ISUB,JSUB,AMIN,IFLAG,NCL,JJJ,ITER)
C
C       PURPOSE -- (1) UPDATE DISTANCE-BETWEEN-CLUSTER ARRAY
C         SQUA AS REQUIRED BY PREVIOUS ITERATION
C                  (2) FIND WHICH TWO CLUSTERS ARE TO BE
C         COMBINED NEXT (ISUB AND JSUB)
C
C       FEBRUARY 21, 1977
C
C
C
      DIMENSION SQUA(1),MARRAY(1),MCEL(1),EXTRA(1),ONDCL(1)
      INTEGER*2 MARRAY,MCEL,ONDCL
C
C
C
      IF (ITER.EQ.1) GO TO 100
C          IE, IF THIS IS THE FIRST ITERATION, GO TO 100
C
      GO TO (10,20,30,40,50,60,160,190),IFLAG
C
C
C *** SINGLE LINKAGE CRITERION
C
   10 AI=.5
      AJ=.5
      B=0.0
      G=-.5
      GO TO 70
C
C
C *** COMPLETE LINKAGE CRITERION
C
   20 AI=.5
C
      AJ=.5
      B=0.0
      G=.5
      GO TO 70
C
C
C *** GROUP AVERAGE CRITERION
C
   30 AI=XNI/(XNI+XNJ)
      AJ=XNJ/(XNJ+XNI)
      B=0.0
      G=0.0
      GO TO 70
C
C
C *** WEIGHTED AVERAGE CRITERION
C
   40 AI=.5
      AJ=.5
      B=0.0
      G=0.0
      GO TO 70
C
C
C *** CENTROID CRITERION
C
   50 AI=XNI/(XNI+XNJ)
      AJ=XNJ/(XNI+XNJ)
      B=-(XNI*XNJ)/((XNJ+XNI)*(XNJ+XNI))
      G=0.0
      GO TO 70
C
C
C *** MEDIAN CRITERION
C
   60 AI=.5
      AJ=.5
      B=-.25
      G=0.0
C
C
C *** (FOR THE FIRST SIX CRITERIA)
   70 CONTINUE
      IF (NCL.EQ.2) GO TO 150
C
C       COMPUTE DISTANCES TO NEWLY FORMED CLUSTER
```

```
C        AND STORE DISTANCES IN SQUA
C
      DO 80 J=2,NCL
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,J)
      DIF=SQUA(KI)-SQUA(KJ)
      IF (DIF.LT.0.0) DIF=-DIF
      SQUA(KI)=AI*SQUA(KI)+AJ*SQUA(KJ)+B*SQUA(IJ)+G*DIF
C
C             CHECK FOR MINIMUM DISTANCES
C
      IF (AMIN.LE.SQUA(KI)) GO TO 80
      AMIN=SQUA(KI)
      ISUB=1
      JSUB=J
   80 CONTINUE
C
C
   90 IILOW=2
      GO TO 110
C
  100 IILOW=1
C        (FOR FIRST ITERATION)
      IF (IFLAG.EQ.8) GO TO 210
C
C             CHECK THE REST OF THE CLUSTERS FOR
C            MINIMUM DISTANCES
C
  110 LIMIT=NCL-1
      DO 140 I=IILOW, LIMIT
      ILOWER=I+1
      IN1=ONDCL(I)
      DO 140 K=ILOWER,NCL
      IN2=ONDCL(K)
      IF (MARRAY(IN1).LT.MARRAY(IN2)) GO TO 120
      MA=MARRAY(IN1)
      MI=MARRAY(IN2)
      GO TO 130
  120 MA=MARRAY(IN2)
      MI=MARRAY(IN1)
  130 IK=(MI-1)*NSAMP-MI*(MI+1)/2+MA
      IF (AMIN.LE.SQUA(IK)) GO TO 140
      AMIN=SQUA(IK)
      ISUB=I
      JSUB=K
  140 CONTINUE
      RETURN
C
C *** LAST ITERATION FOR ALL BUT THE MINIMUM VARIANCE AND
C        STANDARD DEVIATION
  150 CONTINUE
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,2)
      DIF=SQUA(KI)-SQUA(KJ)
      IF (DIF.LT.0.0) DIF=-DIF
      AMIN=AI*SQUA(KI)+AJ*SQUA(KJ)+B*SQUA(IJ)+G*DIF
      ISUB=1
      JSUB=2
      RETURN
C
C
C *** MINIMUM VARIANCE CRITERION
C
  160 LIMIT=NCL-1
      XMIJ=XNI+XNJ
      IF (NCL.EQ.2) GO TO 180
      DO 170 J=2,NCL
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,J)
      A=MCEL(J)
      XM=XMIJ+A
      SQUA(KI)=((XNI+A)/XM)*SQUA(KI)+((XNJ+A)/XM)
     A   *SQUA(KJ)-(A/XM)*SQUA(IJ)
      IF (AMIN.LE.SQUA(KI)) GO TO 170
      AMIN=SQUA(KI)
      ISUB=1
      JSUB=J
  170 CONTINUE
      GO TO 90
C
C *** LAST ITERATION FOR MINIMUM VARIANCE
  180 CONTINUE
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,2)
      A=MCEL(2)
      XM=XMIJ+A
      AMIN=((XNI+A)/XM)*SQUA(KI)+((XNJ+A)/XM)*SQUA(KJ)-
     A   (A/XM)*SQUA(IJ)
      ISUB=1
      JSUB=2
      RETURN
C
C
C *** MINIMUM STANDARD DEVIATION CRITERION
C
  190 LIMIT=NCL-1
      XMIJ=XNI+XNJ
      III=MARRAY(1)
      X=EXTRA(III)+EXTRA(JJJ)
      IF (NCL.EQ.2) GO TO 250
      DO 200 K=2,NCL
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,K)
      A=MCEL(K)
      D=A+XMIJ
      D=1./(D*(D-1.))
      IN1=ONDCL(K)
      KKK=MARRAY(IN1)
      A=EXTRA(KKK)
      SQUA(KI)=SQUA(KI)+SQUA(KJ)+SQUA(IJ)-A-X
      D=SQUA(KI)*D
      IF (AMIN.LE.D) GO TO 200
      AMIN=D
      ISUB=1
      JSUB=K
  200 CONTINUE
      EXTRA(III)=SQUA(IJ)
      IILOW=2
C
C       CHECK REST OF CLUSTER PAIRS FOR MINIMUM
C       STANDARD DEVIATION
  210 LIMIT=NCL-1
      DO 240 I=IILOW, LIMIT
      A=MCEL(I)
      ILOWER=I+1
      IN1=ONDCL(I)
      DO 240 K=ILOWER,NCL
      IN2=ONDCL(K)
      IF (MARRAY(IN1).LT.MARRAY(IN2)) GO TO 220
      MA=MARRAY(IN1)
      MI=MARRAY(IN2)
      GO TO 230
  220 MA=MARRAY(IN2)
      MI=MARRAY(IN1)
  230 IK=(MI-1)*NSAMP-MI*(MI+1)/2+MA
      B=MCEL(K)+A
      B=1./(B*(B-1.))
      B=B*SQUA(IK)
      IF (AMIN.LE.B) GO TO 240
      AMIN=B
      ISUB=I
      JSUB=K
  240 CONTINUE
      RETURN
C
C *** FINAL ITERATION FOR MINIMUM STANDARD DEVIATION
  250 CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,2)
      A=MCEL(2)
      D=A+XMIJ
      D=1./(D*(D-1.))
      IN2=ONDCL(2)
      KKK=MARRAY(IN2)
      A=EXTRA(KKK)
      EXTRA(III)=SQUA(IJ)
      AMIN=SQUA(KI)+SQUA(KJ)+SQUA(IJ)-A-X
      AMIN=AMIN*D
      ISUB=1
      JSUB=2
      RETURN
C
      END
C
C
C
```

```
      SUBROUTINE INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,J)
C
C    PURPOSE -- LOCATE POSITION IN DISTANCE ARRAY "SQUA" FOR
C      DISTANCE BETWEEN CLUSTERS III AND JJJ (IJ),
C      DISTANCE BETWEEN CLUSTERS KKK AND III (KI),
C      AND DISTANCE BETWEEN CLUSTERS KKK AND JJJ (KJ).
C
C       FEBRUARY 21, 1977
C
C
      DIMENSION MARRAY(1),ONDCL(1)
      INTEGER*2 MARRAY,ONDCL
C
      IN1=ONDCL(1)
      IN2=ONDCL(J)
      IF (MARRAY(IN1).LT.MARRAY(IN2)) GO TO 10
      MA=MARRAY(IN1)
      MI=MARRAY(IN2)
      GO TO 20
   10 MA=MARRAY(IN2)
      MI=MARRAY(IN1)
   20 KI=NSAMP*(MI-1)-MI*(MI+1)/2+MA
      IF (JJJ.LT.MARRAY(IN2)) GO TO 30
      MA=JJJ
      MI=MARRAY(IN2)
      GO TO 40
   30 MA=MARRAY(IN2)
      MI=JJJ
   40 KJ=(MI-1)*NSAMP-MI*(MI+1)/2+MA
      IF (MARRAY(IN1).LT.JJJ) GO TO 50
      MA=MARRAY(IN1)
      MI=JJJ
      GO TO 60
   50 MA=JJJ
      MI=MARRAY(IN1)
   60 IJ=(MI-1)*NSAMP-MI*(MI+1)/2+MA
      RETURN
C
      END
```

```
C       PMGCLS PROGRAM LISTING
      COMMON/ALWAYS/STORY(30),DATE(5),TODAY(2),IFLAG,NORMA,
     > NORMB,MET,NSQUAR,LOG,NEWDAT,INV,NFACTR,
     > IPLOT,MPLOT,NCLAS,ITURN,IDELX,ISTORY,IPTALK
      INTEGER DIMEN
      COMMON/HOLD/NSIZE,A(10000)
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),
     >  TITCLS(5),PTALK(20)
C
C
      DIMENSION DDATE(5)
      DATA DDATE/'*** ',' TOD','AYS ','DATE',' IS '/
C
C
      NSIZE=10000
      NUMBER=0
      CALL COMPRS
C
      DO 10 I=1,5
   10 DATE(I)=DDATE(I)
      NSNAM=1
C
   20 NUMBER=NUMBER+1
C
C
C
      TYPE 10000
      ACCEPT 10100,IFLAG
      TYPE 10200
      ACCEPT 10300,NSAMP
      CALL BGNPL(NUMBER)
      CALL LOOSE(1)
      LLDIST=DIMEN(NSAMP)
      N=((NSAMP-1)*NSAMP)/2
      LLSQUA=DIMEN(N)
      LLSAMP=DIMEN(NSAMP)
C                        SQUA -- DISTANCE BETWEEN SAMPLES
      LLEXTR=DIMEN(NSAMP)
C
      LLKLUS=DIMEN(2*NSAMP+1)
C                        KLUSTR
      N=NSAMP
      LLMARR=DIMEN(N)
C                        MARRAY
      LLJARR=DIMEN(N)
C                        JARRAY
      NS=NSAMP
      LLMCEL=DIMEN(NS)
C                        MCEL
      LLJCEL=DIMEN(NS)
C                        JCEL
      LLONCL=DIMEN(N)
C                        ONDCL
```

```
      LLINCL=DIMEN(N)
C                       INDCL
C
      CALL INPUT(A(LLSQUA),A(LLSAMP),NSAMP)
      CALL CLUST(A(LLDIST),A(LLSQUA),A(LLEXTR),
     > A(LLKLUS),A(LLMARR),A(LLJARR),A(LLMCEL),
     > A(LLJCEL),A(LLONCL), A(LLINCL),NFEAT,NSAMP)
C
C
C *** PREPARE ARRAYS FOR CALL TO SUBROUTINE DENDRO
C
      CALL LOOSE(LLMARR)
      LLIPOS=DIMEN(NS)
C                       IPOS
      LLNC=DIMEN(NS)
C                       NC
      LLNC2=DIMEN(NS)
C                       NC2
      N=NSAMP+NSAMP
      LLSTOR=DIMEN(N)
C                       STORE
      LLSTR2=DIMEN(N)
C                       STORE2
      N=5*NSAMP
      LLXXXX=DIMEN(N)
C                       XXXX
C
      CALL DENDRO(A(LLDIST),A(LLSAMP),A(LLKLUS),
     > A(LLIPOS),A(LLNC),A(LLNC2),A(LLSTOR),A(LLSTR2)
     > ,A(LLXXXX),NFEAT,NSAMP)
C
      CALL ENDGR(0)
      CALL ENDPL(0)
      CALL DONEPL(0)
   30 STOP
C
C
10000 FORMAT(' CLUSTERING CRITERION?')
10100 FORMAT(I)
10200 FORMAT(' NO OF SAMPLES?')
10300 FORMAT(2I)
      END
C
C
C
      SUBROUTINE INPUT(SQUA,ISAMP,NSAMP)
      COMMON/TIL/TITLE(16)
      DIMENSION SQUA(1),ISAMP(1)
      NN=((NSAMP-1)*NSAMP)/2
      TYPE 10000
      ACCEPT 10100,ITER
      DIV=1./ITER
      READ(45,10300) TITLE
      READ(45,10400)  (ISAMP(I),I=1,NSAMP)
      READ(45,10200) (SQUA(I),I=1,NN)
      DO 10 J=1,NN
      SQUA(J)=ASIN(SQRT(1.-SQUA(J)*DIV))
   10 CONTINUE
      RETURN
10000 FORMAT(' NO OF ITERATIONS?')
10100 FORMAT(I)
10200 FORMAT(20F4.0)
10300 FORMAT(16A5)
10400 FORMAT(20A4)
      END
C
C
C
C
C
C
      SUBROUTINE CLUST(DISTAN,SQUA,EXTRA,KLUSTR,MARRAY,
     > JARRAY,MCEL,JCEL,ONDCL,INDCL,NNFEAT,NNSAMP)
C
C       PURPOSE -- DO THE CLUSTER ANALYSIS !!
C
C       JULY 22, 1977
C
C
      DIMENSION DISTAN(1),SQUA(1),EXTRA(1),KLUSTR(2,1),
     > MARRAY(1),JARRAY(1),MCEL(1),JCEL(1),ONDCL(1),INDCL(1)
C
      COMMON/ALWAYS/STORY(30),DATE(5),TODAY(2),
     > IFLAG,NORMA,NORMB,MET,NSQUAR,LOG,NEWDAT,INV,
     > NFACTR,IPLOT,MPLOT,NCLAS,ITURN,IDELX,
     > ISTORY,IPTALK
C
C
      NSAMP=NNSAMP
      NFEAT=NNFEAT
C
C
C  INITIALIZE TO NSAMP CLASSES
C
      NCL=NSAMP
      ITER=0
      NCLUST=0
      DO 10 I=1,NSAMP
      MCEL(I)=1
      JCEL(I)=0
      ONDCL(I)=I
      MARRAY(I)=I
      JARRAY(I)=0
      INDCL(I)=0
   10 CONTINUE
C
```

```
C
C *** START LOOP
C
   20 CONTINUE
      ITER=ITER+1
C
C  FIND CANDIDATES FOR CLUSTERING
C
      ISUB=0
      JSUB=0
      AMIN=1.E10
      CALL INTER(SQUA,EXTRA,MARRAY,MCEL,ONDCL,
     > NSAMP,XNI,XNJ,ISUB,JSUB,AMIN,IFLAG,NCL,JJJ,ITER)
      SAMIN=AMIN
      IF (NSQUAR.EQ.0) SAMIN=SQRT(AMIN)
      KLUSTR(1,ITER)=ISUB
      KLUSTR(2,ITER)=JSUB
      DISTAN(ITER)=SAMIN
      WRITE(06,10000)ITER,ISUB,JSUB,SAMIN
C
C  ISUB AND JSUB ARE THE CLUSTER NUMBERS WHICH MINIMIZE
C       THE CRITERION.
C  FORM A CLUSTER
C
      NCLUST=NCLUST+1
      ILIM=MCEL(ISUB)
      XNI=ILIM
      INDIS=ONDCL(ISUB)
C       NUMBER OF SAMPLES IN CLUSTER ISUB
      JLIM=MCEL(JSUB)
      XNJ=JLIM
      INDJS=ONDCL(JSUB)
C       NUMBER OF SAMPLES IN CLUSTER JSUB
      JJJ=MARRAY(INDJS)
C       NUMBER OF THE FIRST SAMPLE IN JSUB
C
C  LOAD INTO CLUSTER ARRAY
C
      INDEX=0
      INDCL(NCLUST)=INDEX+1
      DO 30 I=1,ILIM
      INDEX=INDEX+1
   30 JARRAY(INDEX)=MARRAY(INDIS+I-1)
      DO 40 J=1,JLIM
      INDEX=INDEX+1
   40 JARRAY(INDEX)=MARRAY(INDJS+J-1)
      JCEL(NCLUST)=ILIM+JLIM
C
C  COPY REST OF MATRIX INTO JARRAY.
C
      DO 60 I=1,NCL
      IF (I.EQ.ISUB) GO TO 60
      IF (I.EQ.JSUB) GO TO 60
      NCLUST=NCLUST+1
      INDOLD=ONDCL(I)
      INDCL(NCLUST)=INDEX+1
      ILIM=MCEL(I)
      AMIN=0.
      DO 50 J=1,ILIM
      INDEX=INDEX+1
      JARRAY(INDEX)=MARRAY(INDOLD+J-1)
   50 CONTINUE
      JCEL(NCLUST)=ILIM
   60 CONTINUE
C
      WRITE(06,10100)ITER,NCLUST
      DO 70 I=1,NCLUST
      ILIM=JCEL(I)
      IF (ILIM.LE.1) GO TO 70
      IST=INDCL(I)-1
      IF (ILIM.GT.10) WRITE(06,10200)
     > I,(JARRAY(IST+J),J=1,ILIM)
C        WRITE(06,10300)ILIM
   70 CONTINUE
C
C  REINITIALIZE
C
      DO 80 J=1,NSAMP
   80 MARRAY(J)=0
C
C  COPY JARRAY INTO MARRAY
C
      DO 90 I=1,NCLUST
      JLIM=JCEL(I)
      MCEL(I)=JLIM
      JCEL(I)=0
      ONDCL(I)=INDCL(I)
   90 CONTINUE
      DO 100 J=1,NSAMP
      MARRAY(J)=JARRAY(J)
      JARRAY(J)=0
  100 CONTINUE
      NCL=NCLUST
      NCLUST=0
      IF (NCL.LE.1) GO TO 110
      GO TO 20
C
C *** END OF LOOP
C
C
  110 CONTINUE
      JLIM=MCEL(1)
      WRITE(06,10300)
      WRITE(06,10400) (MARRAY(J),J=1,JLIM)
      RETURN
C
```

```
C
C
10000 FORMAT(1X,'ITER=',I5,2X,2I5,' AMIN=',F10.4)
10100 FORMAT(' ITERATION',I4,'  NUMBER OF CLUSTERS  ',I4)
10200 FORMAT(1X,'CLUSTER NUMBER',I4,
     > ' SAMPLES ',25I4, /,(24X,25I4))
10300 FORMAT(10X,'FINAL ORDERING  ')
10400 FORMAT(20I5)
      END
C
C
      SUBROUTINE INTER(SQUA,EXTRA,MARRAY,MCEL,ONDCL,NSAMP,
     > XNI,XNJ,ISUB,JSUB,AMIN,IFLAG,NCL,JJJ,ITER)
C
C  PURPOSE -- (1) UPDATE DISTANCE-BETWEEN-CLUSTER ARRAY
C         SQUA AS REQUIRED BY PREVIOUS ITERATION
C    (2) FIND WHICH TWO CLUSTERS ARE TO BE COMBINED
C         NEXT (ISUB AND JSUB)
C
C       JULY 22, 1977
C
C
C
      DIMENSION SQUA(1),EXTRA(1),MARRAY(1),MCEL(1),ONDCL(1)
C
C
C
      IF (ITER.EQ.1) GO TO 100
C       IE, IF THIS IS THE FIRST ITERATION, GO TO 100
C
      GO TO (10,20,30,40,50,60,160,190),IFLAG
C
C
C *** SINGLE LINKAGE CRITERION
C
   10 AI=.5
      AJ=.5
      B=0.0
      G=-.5
      GO TO 70
C
C
C *** COMPLETE LINKAGE CRITERION
C
   20 AI=.5
C
      AJ=.5
      B=0.0
      G=.5
      GO TO 70
C
C
C *** GROUP AVERAGE CRITERION
```

```
C
   30 AI=XNI/(XNI+XNJ)
      AJ=XNJ/(XNJ+XNI)
      B=0.0
      G=0.0
      GO TO 70
C
C
C *** WEIGHTED AVERAGE CRITERION
C
   40 AI=.5
      AJ=.5
      B=0.0
      G=0.0
      GO TO 70
C
C
C *** CENTROID CRITERION
C
   50 AI=XNI/(XNI+XNJ)
      AJ=XNJ/(XNI+XNJ)
      B=-(XNI*XNJ)/((XNJ+XNI)*(XNJ+XNI))
      G=0.0
      GO TO 70
C
C
C *** MEDIAN CRITERION
C
   60 AI=.5
      AJ=.5
      B=-.25
      G=0.0
C
C
C *** (FOR THE FIRST SIX CRITERIA)
   70 CONTINUE
      IF (NCL.EQ.2) GO TO 150
C
C     COMPUTE DISTANCES TO NEWLY FORMED CLUSTER AND
C     STORE DISTANCES IN SQUA
C
      DO 80 J=2,NCL
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,J)
      DIF=SQUA(KI)-SQUA(KJ)
      IF (DIF.LT.0.0) DIF=-DIF
      SQUA(KI)=AI*SQUA(KI)+AJ*SQUA(KJ)+B*SQUA(IJ)+G*DIF
C
C     CHECK FOR MINIMUM DISTANCES
C
CTYPE *,KI,AMIN
      IF (AMIN.LE.SQUA(KI)) GO TO 80
      AMIN=SQUA(KI)
      ISUB=1
      JSUB=J
   80 CONTINUE
C
C
   90 IILOW=2
      GO TO 110
C
  100 IILOW=1
C         (FOR FIRST ITERATION)
      IF (IFLAG.EQ.8) GO TO 210
C
C    CHECK THE REST OF THE CLUSTERS FOR
C         MINIMUM DISTANCES
C
  110 LIMIT=NCL-1
      DO 140 I=IILOW, LIMIT
      ILOWER=I+1
      IN1=ONDCL(I)
      DO 140 K=ILOWER,NCL
      IN2=ONDCL(K)
      IF (MARRAY(IN1).LT.MARRAY(IN2)) GO TO 120
      MA=MARRAY(IN1)
      MI=MARRAY(IN2)
      GO TO 130
  120 MA=MARRAY(IN2)
      MI=MARRAY(IN1)
C 130       IK=(MI-1)*NSAMP-MI*(MI+1)/2+MA
  130 IK=((MA-2)*(MA-1))/2+MI
CTYPE *,IK,SQUA(IK),AMIN
      IF (AMIN.LE.SQUA(IK)) GO TO 140
      AMIN=SQUA(IK)
      ISUB=I
      JSUB=K
  140 CONTINUE
      RETURN
C
C *** LAST ITERATION FOR ALL BUT THE MINIMUM VARIANCE AND
C         STANDARD DEVIATION
  150 CONTINUE
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,2)
      DIF=SQUA(KI)-SQUA(KJ)
      IF (DIF.LT.0.0) DIF=-DIF
      AMIN=AI*SQUA(KI)+AJ*SQUA(KJ)+B*SQUA(IJ)+G*DIF
      ISUB=1
      JSUB=2
      RETURN
C
C
C *** MINIMUM VARIANCE CRITERION
C
  160 LIMIT=NCL-1
      XMIJ=XNI+XNJ
      IF (NCL.EQ.2) GO TO 180
      DO 170 J=2,NCL
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,J)
      A=MCEL(J)
      XM=XMIJ+A
      SQUA(KI)=((XNI+A)/XM)*SQUA(KI)+((XNJ+A)/XM)
     > *SQUA(KJ)-(A/XM)*SQUA(IJ)
      IF (AMIN.LE.SQUA(KI)) GO TO 170
      AMIN=SQUA(KI)
      ISUB=1
      JSUB=J
  170 CONTINUE
      GO TO 90
C
C *** LAST ITERATION FOR MINIMUM VARIANCE
  180 CONTINUE
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,2)
      A=MCEL(2)
      XM=XMIJ+A
      AMIN=((XNI+A)/XM)*SQUA(KI)+((XNJ+A)/XM)*SQUA(KJ)-
     > (A/XM)*SQUA(IJ)
      ISUB=1
      JSUB=2
      RETURN
C
C
C *** MINIMUM STANDARD DEVIATION CRITERION
C
  190 LIMIT=NCL-1
      XMIJ=XNI+XNJ
      III=MARRAY(1)
      X=EXTRA(III)+EXTRA(JJJ)
      IF (NCL.EQ.2) GO TO 250
      DO 200 K=2,NCL
      CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,K)
      A=MCEL(K)
      D=A+XMIJ
      D=1./(D*(D-1.))
      IN1=ONDCL(K)
      KKK=MARRAY(IN1)
      A=EXTRA(KKK)
      SQUA(KI)=SQUA(KI)+SQUA(KJ)+SQUA(IJ)-A-X
      D=SQUA(KI)*D
      IF (AMIN.LE.D) GO TO 200
      AMIN=D
      ISUB=1
      JSUB=K
  200 CONTINUE
      EXTRA(III)=SQUA(IJ)
      IILOW=2
C
C       CHECK REST OF CLUSTER PAIRS FOR MINIMUM
C       STANDARD DEVIATION
  210 LIMIT=NCL-1
```

```
      DO 240 I=IILOW, LIMIT
      A=MCEL(I)
      ILOWER=I+1
      IN1=ONDCL(I)
      DO 240 K=ILOWER,NCL
      IN2=ONDCL(K)
      IF (MARRAY(IN1).LT.MARRAY(IN2)) GO TO 220
      MA=MARRAY(IN1)
      MI=MARRAY(IN2)
      GO TO 230
  220 MA=MARRAY(IN2)
      MI=MARRAY(IN1)
C 230       IK=(MI-1)*NSAMP-MI*(MI+1)/2+MA
  230 IK=((MA-2)*(MA-1))/2+MI
      B=MCEL(K)+A
      B=1./(B*(B-1.))
      B=B*SQUA(IK)
      IF (AMIN.LE.B) GO TO 240
      AMIN=B
      ISUB=I
      JSUB=K
  240 CONTINUE
      RETURN
C
C *** FINAL ITERATION FOR MINIMUM STANDARD DEVIATION
  250 CALL INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,2)
      A=MCEL(2)
      D=A+XMIJ
      D=1./(D*(D-1.))
      IN2=ONDCL(2)
      KKK=MARRAY(IN2)
      A=EXTRA(KKK)
      EXTRA(III)=SQUA(IJ)
      AMIN=SQUA(KI)+SQUA(KJ)+SQUA(IJ)-A-X
      AMIN=AMIN*D
      ISUB=1
      JSUB=2
      RETURN
C
      END
C
C
C
      SUBROUTINE INT2(MARRAY,ONDCL,NSAMP,JJJ,KI,KJ,IJ,J)
C
C  PURPOSE -- LOCATE POSITION IN DISTANCE ARRAY "SQUA" FOR
C     DISTANCE BETWEEN CLUSTERS III AND JJJ (IJ),
C     DISTANCE BETWEEN CLUSTERS KKK AND III (KI),
C     AND DISTANCE BETWEEN CLUSTERS KKK AND JJJ (KJ).
C
C        JULY 22, 1977
C
C
```

```
      DIMENSION MARRAY(1),ONDCL(1)
C
      IN1=ONDCL(1)
      IN2=ONDCL(J)
      IF (MARRAY(IN1).LT.MARRAY(IN2)) GO TO 10
      MA=MARRAY(IN1)
      MI=MARRAY(IN2)
      GO TO 20
   10 MA=MARRAY(IN2)
      MI=MARRAY(IN1)
C  20 KI=NSAMP*(MI-1)-MI*(MI+1)/2+MA
   20 KI=((MA-1)*(MA-2))/2+MI
      IF (JJJ.LT.MARRAY(IN2)) GO TO 30
      MA=JJJ
      MI=MARRAY(IN2)
      GO TO 40
   30 MA=MARRAY(IN2)
      MI=JJJ
C  40 KJ=(MI-1)*NSAMP-MI*(MI+1)/2+MA
   40 KJ=((MA-2)*(MA-1))/2+MI
      IF (MARRAY(IN1).LT.JJJ) GO TO 50
      MA=MARRAY(IN1)
      MI=JJJ
      GO TO 60
   50 MA=JJJ
      MI=MARRAY(IN1)
C  60 IJ=(MI-1)*NSAMP-MI*(MI+1)/2+MA
   60 IJ=((MA-2)*(MA-1))/2+MI
      RETURN
C
      END
C
C
C
      SUBROUTINE DENDRO(DIST,SAMPLE,KLUSTR,IPOS,NC,NC2,
     > STORE,STORE2,XXXX,NFEAT,NSAMP)
C
C
C       PURPOSE -- PLOT DENDROGRAMS !!!
C
C       JULY 22, 1977
C
C
      DIMENSION DIST(1),SAMPLE(NSNAM,1),KLUSTR(2,1),
     > IPOS(1),NC(1),NC2(1),STORE(2,1),STORE2(2,1),
     > XXXX(NSAMP,5)
C
      COMMON/ALWAYS/STORY(30),DATE(5),TODAY(2),
     > IFLAG,NORMA,NORMB,MET,NSQUAR,LOG,NEWDAT,
     > INV,NFACTR,IPLOT,MPLOT,NCLAS,ITURN,IDELX,
     > ISTORY,IPTALK
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),TITCLS(5),
     > PTALK(20)
```

```
      DIMENSION X(4),Y(4)
C
C
C
C
C
      DELX=0.2
C
      XINC=0.
      XTALK=XINC-2.0
C
C *** INITIALIZE ARRAYS
      DO 10 J=1,NSAMP
   10 NC(J)=J
C
      CALL VARSAM(SAMPLE,DIST,NFEAT,NSAMP,XINC,YINC,DELX)
C
C *** WRITE SAMPLE NAMES ALONG X-AXIS
C *** VERTICAL SAMPLE NAMES
      XINC=XINC+DELX*1.1
      SIZE=0.12
      IF (DELX.LE.SIZE) SIZE=DELX*0.6
      USIZE=SIZE*0.8
      YINC=-0.5
      IF (NOS.GT.4) YINC=YINC-0.4
      YINC2=YINC-0.515
      YINCA=YINC2-0.035
      YINC3=YINC2-0.40
      CALL ANGLE(90.)
      DO 20 J=1,NSAMP
      CALL HEIGHT(SIZE)
      CALL MESSAG(SAMPLE(1,IPOS(J)),4,XINC,YINC)
      XINC=XINC+DELX
   20 CONTINUE
      CALL RESET('ANGLE')
   30 DO 40 J=1,NSAMP
      STORE(1,IPOS(J))=J
      STORE(2,J)=0.0
   40 CONTINUE
C
      NSTEP=NSAMP-1
      DO 130 K=1,NSTEP
C
      XXXX(K,1)=STORE(1,KLUSTR(1,K))
      XXXX(K,2)=STORE(1,KLUSTR(2,K))
      XXXX(K,3)=STORE(2,KLUSTR(1,K))
      XXXX(K,4)=STORE(2,KLUSTR(2,K))
      XXXX(K,5)=DIST(K)
C
C
C *** REINITIALIZE ARRAYS FOR NEXT ITERATION
      XN=(STORE(1,KLUSTR(1,K))+STORE(1,KLUSTR(2,K)))/2.
      NO=1
      M=KLUSTR(1,K)
      J=KLUSTR(2,K)
      IXL=MINO(M,J)
      IXG=MAXO(M,J)
      DO 110 M=1,NSAMP
      IF (NC(M)-IXL) 50,100,60
   50 NO=NC(M)+1
      GO TO 70
   60 NO=NC(M)
   70 IF (NC(M)-IXG) 90,100,80
   80 NO=NC(M)-1
   90 CONTINUE
      STORE2(1,NO)=STORE(1,NC(M))
      STORE2(2,NO)=STORE(2,NC(M))
      NC2(M)=NO
      GO TO 110
  100 CONTINUE
      STORE2(1,1)=XN
      STORE2(2,1)=DIST(K)
      NC2(M)=1
  110 CONTINUE
      DO 120 M=1,NSAMP
      NC(M)=NC2(M)
      STORE(1,M)=STORE2(1,M)
      STORE(2,M)=STORE2(2,M)
  120 CONTINUE
  130 CONTINUE
C
C *** ORDER "UP-ACROSS-DOWN" LINES ON XXXX(K,1)
      N=NSTEP
      DO 160 K=2,NSTEP
      NSWAP=0
      N=N-1
      DO 150 KK=2,N
      IF (XXXX(KK-1,1).LE.XXXX(KK,1)) GO TO 150
      DO 140 J=1,5
      A=XXXX(KK,J)
      XXXX(KK,J)=XXXX(KK-1,J)
  140 XXXX(KK-1,J)=A
      NSWAP=NSWAP+1
  150 CONTINUE
      IF (NSWAP.EQ.0) GO TO 170
  160 CONTINUE
C
C *** DRAW "UP-ACROSS-DOWN" LINES
  170 K=1
  180 X(1)=XXXX(K,1)
      X(2)=X(1)
      X(3)=XXXX(K,2)
      X(4)=X(3)
      Y(1)=XXXX(K,3)
      Y(2)=XXXX(K,5)
      Y(3)=Y(2)
      Y(4)=XXXX(K,4)
      GO TO 200
  190 X(1)=XXXX(K,2)
      X(2)=X(1)
      X(3)=XXXX(K,1)
      X(4)=X(3)
      Y(1)=XXXX(K,4)
      Y(2)=XXXX(K,5)
      Y(3)=Y(2)
      Y(4)=XXXX(K,3)
  200 CALL CURVE(X,Y,4,0)
C
C *** PREPARE NEXT LINES
      K=K+1
      IF (K.GT.NSTEP) GO TO 210
      A=(X(4)-XXXX(K,1))**2+(Y(4)-XXXX(K,3))**2
      B=(X(4)-XXXX(K,2))**2+(Y(4)-XXXX(K,4))**2
      IF (A.LE.B) GO TO 180
      GO TO 190
C
  210 CONTINUE
C
      WRITE(6,10000)
      RETURN
C
C
10000 FORMAT('0PLOT COMPLETED')
10100 FORMAT(15A4)
10200 FORMAT(F10.3)
      END
C
C
C
      INTEGER FUNCTION DIMEN(MANY)
C
C       PURPOSE -- KEEP TRACK OF DIMENSIONS
C       JULY 22, 1977
C
      COMMON/HOLD/NSIZE,A(1)
      DATA KOUNT/1/
      DIMEN=KOUNT
      KOUNT=KOUNT+MANY
      IF (KOUNT.GT.NSIZE) WRITE(6,10000) NSIZE,KOUNT
      RETURN
C
      ENTRY LOOSE(MANY)
      KOUNT=MANY
      RETURN
C
10000 FORMAT(' AVAILABLE SIZE=',I5,
     > ' BUT YOU NEED',I6,'.   ERROR!')
      END
      SUBROUTINE VARSAM(SAMPLE,DIST,NFEAT,NSAMP,XINC,
     > YINC,DELX)
C
C
C         PURPOSE -- TO WRITE VARIABLES AND SAMPLES TO
C         LEFT OF PLOT
C
C         NML
C         JULY 22, 1977
C
C
      COMMON/ALWAYS/STORY(30),DATE(5),TODAY(2),
     > IFLAG,NORMA,NORMB,MET,NSQUAR,LOG,NEWDAT,
     > INV,NFACTR,IPLOT,MPLOT,NCLAS,ITURN,IDELX,
     > ISTORY,IPTALK
      COMMON/NAM/NSNAM,NVNAM,TITSAM(5),TITPP(5),TITCLS(5),
     > PTALK(20)
C
      DIMENSION SAMPLE(1),DIST(1)
C
      DATA LBLANK/'    '/
      COMMON/TIL/TITLL(16)
C
C
   10 CONTINUE
      CALL NOBRDR
      XMAX=NSAMP*DELX+XINC+0.5
      PAGEX=XMAX+1.0
      CALL PAGE(PAGEX,11.0)
      CALL HEIGHT(0.12)
      CALL PHYSOR(0.,0.5)
      CALL TITLE(LBLANK,1,LBLANK,0,LBLANK,0,XMAX,8.5)
   20 CONTINUE
      NSTEP=NSAMP-1
C
C     FIND THE MAXIMUM IN THE Y DIRECTION.
      YMAX=0.
      DO 30 J=1,NSTEP
      IF (YMAX.LT.DIST(J)) YMAX=DIST(J)
   30 CONTINUE
      YINC=0.0
   40 CONTINUE
C
C *** INITIALIZE PLOT
      CALL ENDGR(0)
      CALL OREL(XINC,YINC)
      XMAX=NSAMP*DELX
      CALL TITLE(LBLANK,1,LBLANK,0,LBLANK, 1,XMAX,8.9)
      CALL ANGLE(90.)
      XMAX=NSAMP
      CALL GRAF(0.0,1.0,XMAX,0.0,'SCALE',YMAX)
      CALL MESSAG('CLUSTER DISTANCE',16,-0.5,3.0)
      CALL RESET('ANGLE')
      CALL MESSAG(TITLL,59,0.0,9.0)
      CALL MESSAG(
     > ' CLUSTER ANALYSIS ON MONTE CARLO RESULTS USING $' ,
     > 100,0.0,9.3)
      GO TO (50,60,70,80,90,100,110,120),IFLAG
   50 CALL MESSAG('SINGLE LINKAGE',14,'ABUT','ABUT')
      GO TO 130
   60 CALL MESSAG('COMPLETE LINKAGE',16,'ABUT','ABUT')
      GO TO 130
   70 CALL MESSAG('GROUP AVERAGE',13,'ABUT','ABUT')
      GO TO 130
   80 CALL MESSAG('WEIGHTED AVERAGE',16,'ABUT','ABUT')
      GO TO 130
   90 CALL MESSAG('CENTROID',8,'ABUT','ABUT')
      GO TO 130
  100 CALL MESSAG('MEDIAN',6,'ABUT','ABUT')
      GO TO 130
  110 CALL MESSAG('WARDS METHOD',11,'ABUT','ABUT')
      GO TO 130
  120 CALL MESSAG('STANDARD DEVIATION',19,'ABUT','ABUT')
  130 CALL MESSAG(' FOR THE CLUSTERING CRITERION$',
     > 100,'ABUT','ABUT')
      RETURN
C
C
10000 FORMAT('1THERE ARE',I3,' VARIABLES AND WEIGHTS,',
     > ' WHICH IS TOO MANY TO WRITE ON THE PLOT./'
     >' INSTEAD THEY ARE LISTED HERE FOR YOUR CONVENIENCE'/
     >/'0VARIABLE       WEIGHT')
10100 FORMAT(13X,E10.3,T5,2A4/(4X,18A4))
10200 FORMAT('0THERE ARE',I3,
     > ' SAMPLES, WHICH IS TOO MANY TO LIST ON THE PLOT.')
      END
```

```
C       PMGEST PROGRAM LISTING
      DIMENSION KNT(3000),CA(3000),IGRPS(100,25),NC(100),
     > VEC(100),PROB(100,25),TITLE(16),SAMNO(100),SCU(100),
     > OUTL(100),IH(100)
C***    N IS THE NUMBER OF SAMPLES
      TYPE 10100
      ACCEPT 10400,N
C***    CRITNO IS THE A0 TO USE TO SET UP GROUPS.
      TYPE 10700
      ACCEPT 10800,CRITNO
      PRINT 10900,CRITNO
C***    NIT IS THE NUMBER OF ITERATIONS USED
C***    IN MONTE CARLO ITERATIONS (M)
      TYPE 11000
      ACCEPT 10400,NIT
      DIVFAC=1./NIT
      NA=((N-1)*(N))/2
C***    READ OUTPUT FROM PMGPER AS UNIT 25
      READ(26,11500) TITLE
      READ(26,11600) (SAMNO(I),I=1,N)
      READ(26,10600)  (KNT(I),I=1,NA)
   10 PRINT 11700,TITLE
      PRINT 11800,(SAMNO(K),K=1,N)
      IOUT=0
      OUTLC=NIT-CRITNO
C***    FIND THE SAMPLE PAIRS WHERE THE FREQUENCY IS
C***    IS GREATER THAN A0.
      DO 20 I=1,NA
      CA(I)=KNT(I)*DIVFAC
      IF  (KNT(I).GE.CRITNO)  CA(I)=1.0
   20 CONTINUE
C***    CHECK FOR ANY OUTLIERS (FREQUENCY LESS THAN M-A0)
      DO 40 I=1,N
      DO 30 J=1,N
      IF  (J.EQ.I)  GO TO 30
      NO=((I-1)*(I-2))/2+J
      IF  (J.GT.I)  NO=((J-1)*(J-2))/2+I
      IF  (KNT(NO).GT.OUTLC)  GO TO 40
   30 CONTINUE
C***    SAMPLE I IS AN OUTLIER.
      IOUT=IOUT+1
      OUTL(IOUT)=I
   40 CONTINUE
C***    SET UP GROUPS BY COMBINING SAMPLE
C***    PAIRS WITH FREQUENCY GREATER THAN A0.
      NO=0
      NGRPS=0
      DO 110 J=2,N
      L1=J-1
      DO 100 L=1,L1
      NO=NO+1
C***    SKIP THE REST OF THIS LOOP IF FREQUENCY
C***    OF OCCURENCE <A0.
```

```
      IF (CA(NO).NE.1.0) GO TO 100
      IMATCH=0
      IF (NGRPS.EQ.0) GO TO 90
C***    FIRST COMBINE ONLY THE SAMPLE PAIRS
C***    WHOSE FREQUENCY OF OCCURENCE > A0 FOR EVERY MEMBER
C***    OF THIS GROUP.
      DO 80 IG=1,NGRPS
      NG=NC(IG)
      DO 50 IIG=1,NG
      IV=L
      IF (IGRPS(IIG,IG).EQ.J) GO TO 60
      IV=J
      IF (IGRPS(IIG,IG).EQ.L) GO TO 60
   50 CONTINUE
      GO TO 80
   60 DO 70 IIG=1,NG
      IA=IGRPS(IIG,IG)
      IF (IV.EQ.IA) GO TO 100
      INO=((IA-1)*(IA-2))/2+IV
      IF (IV.GT.IA) INO=((IV-1)*(IV-2))/2+IA
      IF (CA(INO).NE.1.0) GO TO 50
   70 CONTINUE
C***     A MATCH WAS FOUND,
C***    THAT IS ADD SAMPLE IV TO GROUP IG.
      IMATCH=1
      NC(IG)=NC(IG)+1
      IGRPS(NC(IG),IG)=IV
   80 CONTINUE
      IF (IMATCH.EQ.1) GO TO 100
C***    NO MATCH WAS FOUND FOR A PREVIOUSLY EXISTING GROUP,
C***    SO FORM A NEW ONE.
   90 NGRPS=NGRPS+1
      NC(NGRPS)=2
      IGRPS(1,NGRPS)=J
      IGRPS(2,NGRPS)=L
  100 CONTINUE
  110 CONTINUE
C***    THE GROUPS NOW FORMED HAVE ALL MEMBERS
C***    WHOSE FREQUENCY OF OCCURENCE IS > A0
C***    FOR ALL PAIRS.  THIS IS FOUND FIRST,
C***    AND THEN THE GROUPS WITH COMMON SAMPLES
C***    ARE COMBINED TO FORM THE GROUPS DESCRIBED
C***    IN THE PMG PROCEDURE.
      IF (IOUT.EQ.0) GO TO 130
      DO 120 I=1,IOUT
      NGRPS=NGRPS+1
      IGRPS(1,NGRPS)=OUTL(I)
      NC(NGRPS)=1
  120 CONTINUE
  130 CONTINUE
      IF (NGRPS.EQ.1) GO TO 220
C***    NOW COMBINE ALL GROUPS WITH SAMPLES IN COMMON.
      DO 210 J=2,NGRPS
      J1=J-1
      DO 200 K=1,J1
      NNM=0
      NG1=NC(K)
      NG2=NC(J)
      DO 150 N2=1,NG2
      IJ=IGRPS(N2,J)
      DO 140 N1=1,NG1
      IF (IJ.EQ.IGRPS(N1,K)) GO TO 150
  140 CONTINUE
      NNM=NNM+1
      IH(NNM)=IJ
  150 CONTINUE
      IF (NNM.EQ.NG2) GO TO 200
      NGRPS=NGRPS-1
      NC(K)=NC(K)+NNM
      IF (NNM.EQ.0) GO TO 170
      DO 160 NN=1,NNM
      IGRPS(NG1+NN,K)=IH(NN)
  160 CONTINUE
  170 CONTINUE
      DO 190 LL=J,NGRPS
      NN=NC(LL+1)
      DO 180 NI=1,NN
      IGRPS(NI,LL)=IGRPS(NI,LL+1)
  180 CONTINUE
      NC(LL)=NN
  190 CONTINUE
      GO TO 130
  200 CONTINUE
  210 CONTINUE
  220 CONTINUE
      PRINT 12000,NGRPS
C***    CALCULATE THE PROBABILITIES OF SAMPLE MEMBERSHIP.
  230 DO 330 J=1,NGRPS
      PRINT 11200,J
      NNU=0
      DO 240 L=1,N
  240 VEC(L)=0.0
      NG=NC(J)
      DO 250 IG=1,NG
      L=IGRPS(IG,J)
      VEC(L)=1.
      SCU(IG)=SAMNO(L)
  250 CONTINUE
      PRINT 11300,(SCU(NN),NN=1,NG)
      DO 320 L=1,N
      PROB(L,J)=0.0
      L1=L-1
      SUM=0.0
      IF (L1.LE.1) GO TO 270
      DO 260 M=1,L1
      NO=((L-1)*(L-2))/2+M
      SUM=SUM+VEC(M)*CA(NO)
  260 CONTINUE
      IF (L.EQ.N) GO TO 290
  270 L2=L+1
      DO 280 M=L2,N
      NO=((M-1)*(M-2))/2+L
      SUM=SUM+VEC(M)*CA(NO)
  280 CONTINUE
  290 DIV=NC(J)
      IF (VEC(L).NE.1) GO TO 300
      SUM=SUM+1.
      DIV=DIV+1.
  300 SUM=SUM/DIV
      PROB(L,J)=SUM
  310 CONTINUE
  320 CONTINUE
  330 CONTINUE
      DO 360 J=1,N
      SSUM=0.0
      DO 340 K=1,NGRPS
      SSUM=SSUM+PROB(J,K)
  340 CONTINUE
      DO 350 K=1,NGRPS
      IF (SSUM.EQ.0.0) GO TO 350
      PROB(J,K)=PROB(J,K)/SSUM
  350 CONTINUE
  360 CONTINUE
      ICH=0
C***    WRITE OUT SOME INFORMATION TO UNIT 25 TO
C***    PLOT THE SAMPLE MEMBERSHIP PROBABILITIES IF DESIRED.
      WRITE(25,11500) TITLE
      WRITE(25,10200) N,NGRPS,CRITNO
      WRITE(25,11900) (SAMNO(I),I=1,N)
C***    WRITE OUT THE PROBABILITIES TO THE
C***    LINE PRINTER UNIT.
      DO 370 K=1,NGRPS
      PRINT 11100,K,(PROB(L,K),L=1,N)
      WRITE(25,10300) (PROB(L,K),L=1,N)
  370 CONTINUE
      TYPE 12200,NGRPS
      TYPE 10000
      ACCEPT 10500,ANS
      IF (ANS.NE.'Y') STOP
      TYPE 10700
      ACCEPT 10800,CRITNO
      GO TO 10
      STOP
10000 FORMAT(' AGAIN?')
10100 FORMAT(' NUMBER OF SAMPLES?')
10200 FORMAT(2I5,2F10.3)
10300 FORMAT(8F10.3)
10400 FORMAT(I)
10500 FORMAT(A1)
```

```
10600 FORMAT(20I4)
10700 FORMAT(' CRITNO?')
10800 FORMAT(2G)
10900 FORMAT(' THE CRITICAL NO. USED FOR THIS RUN IS',F10.3)
11000 FORMAT(' NUMBER OF ITERATIONS?')
11100 FORMAT(' GROUP',I5/10F9.2/(10F9.2))
11200 FORMAT(' CLUSTER UNIT ',I5/)
11300 FORMAT((1X,10(A4,1X)))
11500 FORMAT(16A5)
11600 FORMAT(20A4)
11700 FORMAT(1X,16A5)
11800 FORMAT(' SAMPLE NUMBERS'/ (10(3X,A4,2X)))
11900 FORMAT(' SAMPLE NUMBERS'/(20A4))
12000 FORMAT(' THE FOLLWING ARE PROBABILITIES FOR',
     > ' THE ',I5,' GROUPS FOUND.')
12200 FORMAT(' THE NO. OF GROUPS =',I5)
      END
```

APPENDIX B

GENERATING AND TESTING RANDOM NUMBERS

A uniform random number generator for an interval I is defined as a method which selects points within I such that each point has an equally likely possibility of occuring. Actual methods for generating uniform numbers in use today usually consist of using a recurrence relation on a digital computer. These procedures specify an initial number α_0, each α_i is calculated from α_{i-1} by some algorithm. The numbers generated by these methods are not truly random, but they can be shown to satisfy statistical tests for randomness and can be assumed to approximate true randomness. Such a generator is then called a pseudorandom number generator [Shreider, 1964].

Statistical testing plays an important part in determining the usefulness of pseudorandom number generators. It is not possible to prove that a generator is truly random since that would necessitate generating an infinite set. Statistical tests can give a qualitative comparison between tested sequences [Halton, 1970]. Such tests can be used to select a pseudorandom number generator for use in a particular system [Gorenstein, 1967].

A collection of six standard statistical tests is used to test the different attributes of locally available pseudorandom generators. These tests are implemented in

TESTRN, a FORTRAN program. Each test is described below. The results of testing the four generators are given and discussed. A description of input and a listing of TESTRN are also contained in this Appendix.

Tests were applied to 10 sets of 5000 uniformly generated random numbers from the interval I from 0 to 1. The first four tests are performed for each set. The last two tests require longer sequences; the entire collection of generated numbers was used for them. In each case, expected values are compared with actual values. Test statistics are applied and the obtained significance level for the hypothesis that the sequence is uniformly random is computed.

The first test calculates the first, second, and third moments for each batch in order to test the uniformity of the sequences over the entire interval I. The calculated moments are then compared with the expected values of one-half, one-third, and one-fourth, respectively, for a uniformly distributed random variable in I [Gorenstein, 1967]. The results are tabulated for each set.

The second test is a frequency test [Kendall, 1938] which is applied to test the uniformity of the generated numbers over small subintervals within I. One hundred equal subintervals were used in testing each batch. The calculated quantity of numbers occurring within each interval is compared to the expected value for a uniform distribution. The chi-square statistic was used to test the goodness-of-fit of the observed frequency to the expected frequency

$$t^2 = \sum_{i=1}^{100} \frac{(q_i - N/100)^2}{N/100} , \qquad \text{(B1)}$$

where N is the total number of random numbers in one set and q_i is the quantity of generated numbers occurring in the i-th interval. The resultant t^2 has approximately a chi-square distribution with 99 degrees of freedom.

In a set of generated numbers $z_1, z_2, \ldots , z_N$, a run up of length s is defined as a subsequence of s successive numbers starting at z_α such that

$$\begin{aligned} &z_{i+\alpha} > z_{j+\alpha} , i > j, \ j = 1, 2, \ldots, s-1 \ , \\ &z_\alpha < z_{\alpha-1} , \qquad \text{(B2)} \\ &z_{s+\alpha+1} < z_{s+\alpha} . \end{aligned}$$

The run down is defined similarly. Levene [1964] gives the expected number of runs of length s in a sequence of uniformly generated random numbers as

$$E(r_s) = 2N \frac{s^2 + 3s + 1}{(s + 3)!} - 2\frac{(s^3 + 3s^2 - s - 4)}{(s + 3)!} , \qquad \text{(B3)}$$

where N is the number of random numbers generated and s is the length of the run. The expected number of runs greater than or equal to s is

$$E(r \geq r_s) = 2N\frac{s+1}{(s+2)!}\;\frac{2(s^2+s-1)}{(s+2)!} \qquad \text{(B4)}$$

In TESTRN, runs of length four or less are tabulated, with the number of runs of length five or greater also being calculated. The total number of runs up and runs down can be compared to check for bias of the generator. The chi-square statistic can be used to test the significance of the results

$$t^2 = \sum_{s=1}^{4} \frac{(E(r_s)-T(r_s))^2}{E(r_s)} + \frac{E(r_s \geq 5)-T(r_s \geq 5)}{E(r_s \geq 5)} \text{(B5)}$$

where $T(r_s)$ is actual tabulation of runs of length s. The statistic t^2 has approximately a chi-square distribution with four degrees of freedom.

The Cramer von Mises test was the fourth test used [Shreider, 1964]. This test does not involve a grouping of the numbers, but instead compares the empirical and hypothetical cumulative distribution functions [Anderson, 1952]. To apply the test, the generated numbers are ordered and the test statistic is computed

$$Nw^2 = 1/12N + \mathop{}_{\nu=1}^{N} [z_{(\nu)} -(2\nu-1/2N)^2 , \qquad \text{(B6)}$$

where $z_{(\nu)}$ is the ordered sequence of the generated numbers. The reported significance level in TESTRN is computed from the approximation in Anderson [1952, Eq. 4.35].

The fifth test is a serial or gap test [Kendall, 1938], which computes the two-dimensional frequency of a pair of generated numbers separated by a specified gap. The resultant frequencies indicate the tendency of two generated numbers to occur together. The frequencies are tabulated as follows:

$$q_{ij} = q_{ij} + 1$$

if z_b is in interval i and $z_{b+\beta}$ is in interval j where q_{ij} is the number of pairs occurring in the i-th and j-th interval, and β is the specified gap length. An interval size of 0.10 was used in TESTRN. Up to five different gaps (β) can be specified as input to TESTRN. The chi-square statistic used to calculate a significance level for the serial test is

$$t^2 = \sum_{i=1}^{10} \sum_{i=1}^{10} \frac{(q_{ij} - N/100)^2}{N/100} \quad , \qquad \text{(B7)}$$

for 99 degrees of freedom.

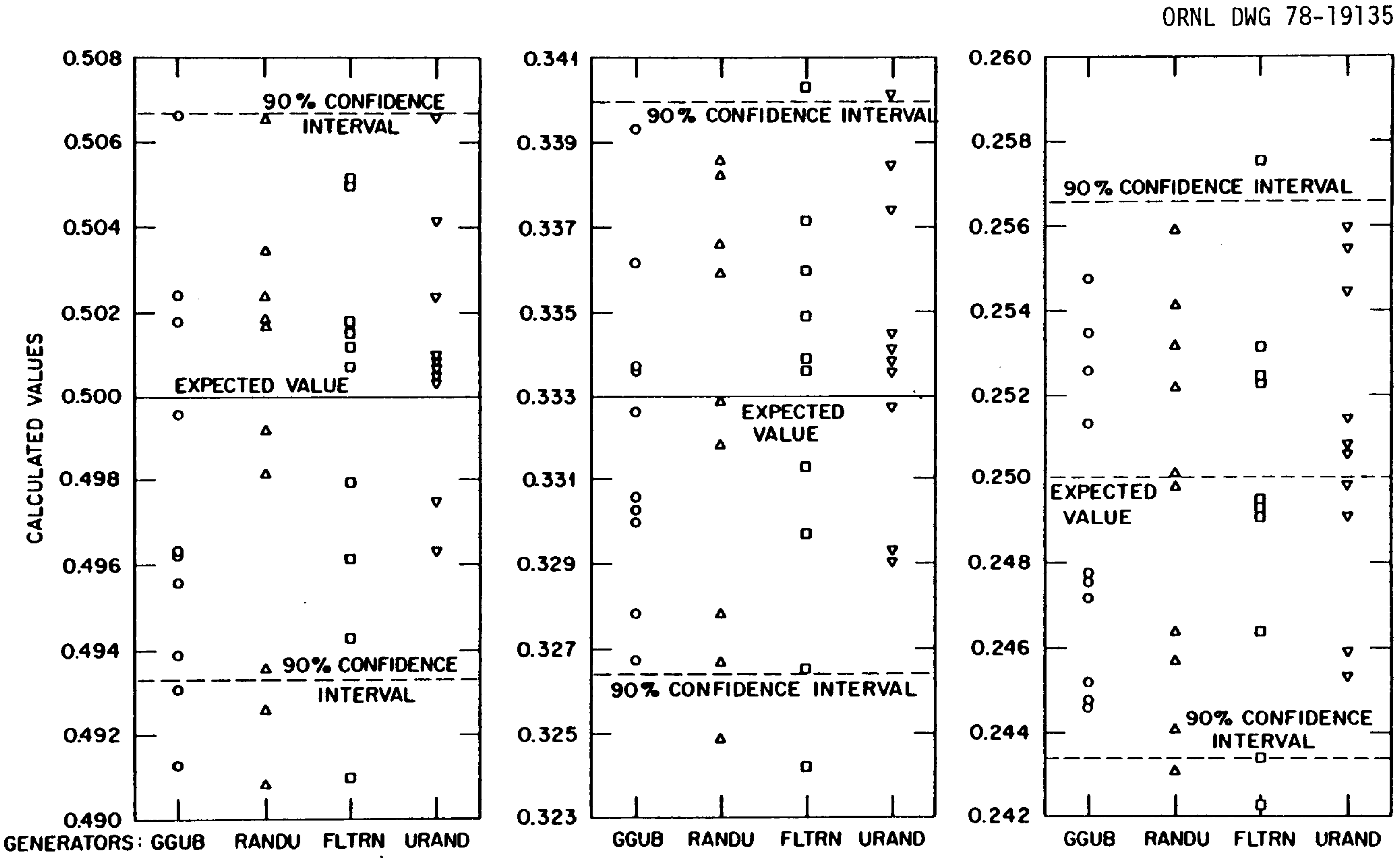
The final test is one suggested by MacLaren and Marsaglia [1965] to evaluate the behavior of n-tuples $(z_1, z_2, \ldots, z_n)$. If

$$\begin{aligned} z_M &= (\max(z_{i+1}, z_{i+2}, \ldots, z_{i+n})^n) \quad , \\ z_m &= 1 - (1 - \min(z_{i+1}, z_{i+2}, \ldots, z_{i+n})^n) \quad , \end{aligned} \qquad \text{(B8)}$$

then both z_M and z_m should be uniformly distributed

[MacLaren and Marsaglia, 1965]. In TESTRN, n-tuples (n = 5, 10, 15, 20) were tested by checking for the uniformity of functions of the maximum and minimum of these vectors. For each set of 20 random variables generated, z_M and z_m were calculated for four 5-tuples, two 10-tuples, and one 15- and 20-tuple. The frequency of z_M and z_m in subintervals of size 0.01 was calculated for all sets. The statistic used for this test is Eq. (B1), where q_i indicates the frequency of z_M or z_m. The statistic is approximately a chi-square with 99 degrees of freedom.

The four pseudouniform random number generators tested include FLTRN [Westley and Watts,1970], GGUB [International Mathematical and Statistical Libraries, 1977], RANDU [International Business Machines, 1970], and URAND [McRae, 1970]. GGUB and RANDU are package routines from the IMSL routine and IBM libraries, respectively. FLTRN is a generator available on the system at Oak Ridge National Laboratory. The generator URAND is a 3^{19} congruential uniform generator and is used in MICKA [McRae, 1970]. The results of the test are given in Figs. B1-B7. In Fig. B1 the first, second, and third moments are compared. Figures B2-B4 compare significance levels for the tests indicated for each batch of uniform numbers generated. Figures B5, B6, and B7 give the significance levels for the tests done on the entire set. All significance levels are for the hypothesis that the numbers generated are random. Table B1



First moment

Second moment

Third moment

Figure B1. Moments of simulated data sets of 5000.

ORNL DWG 77-19316

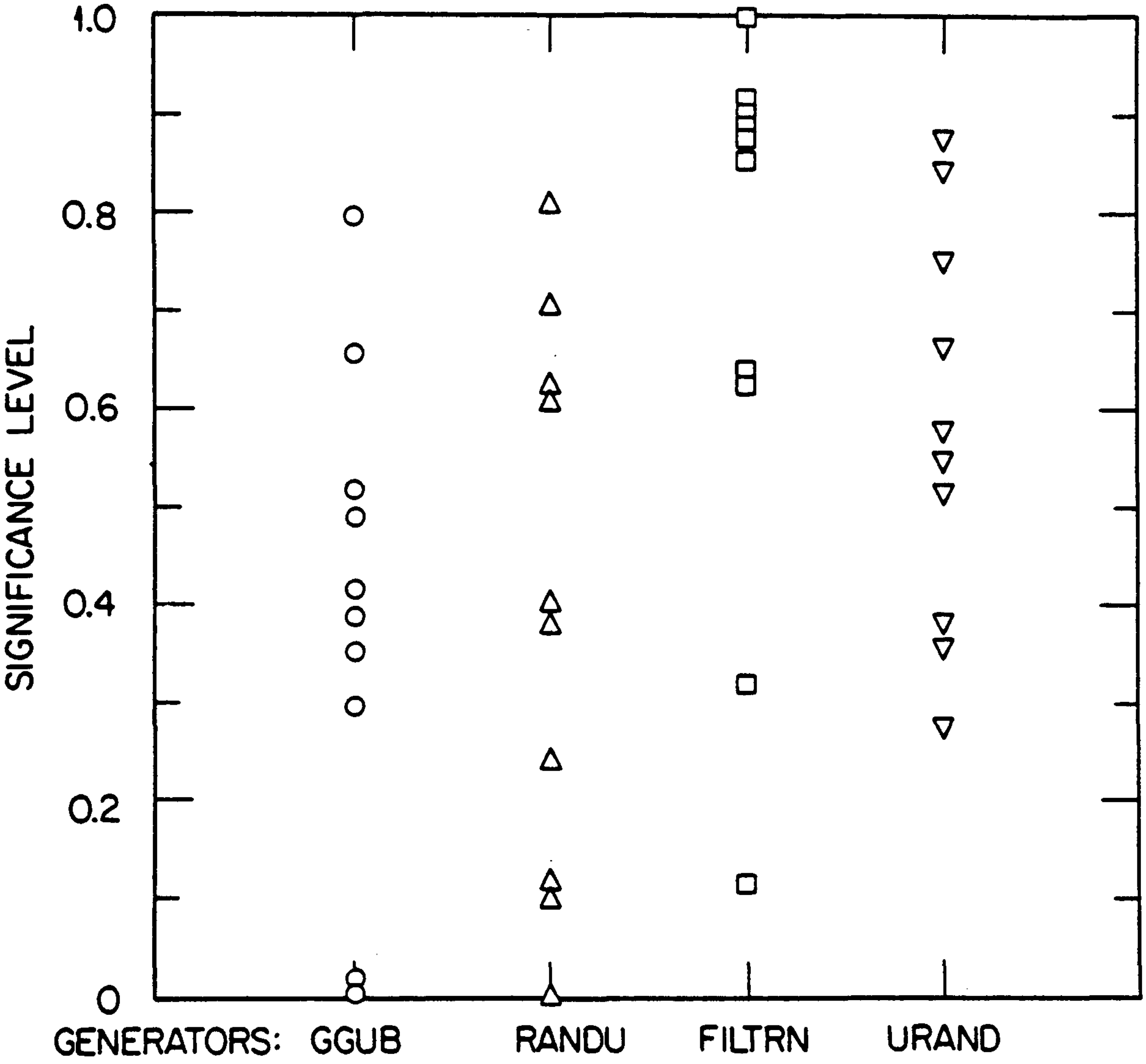


Figure B2. Frequency test results of sample data sets of size 5000.

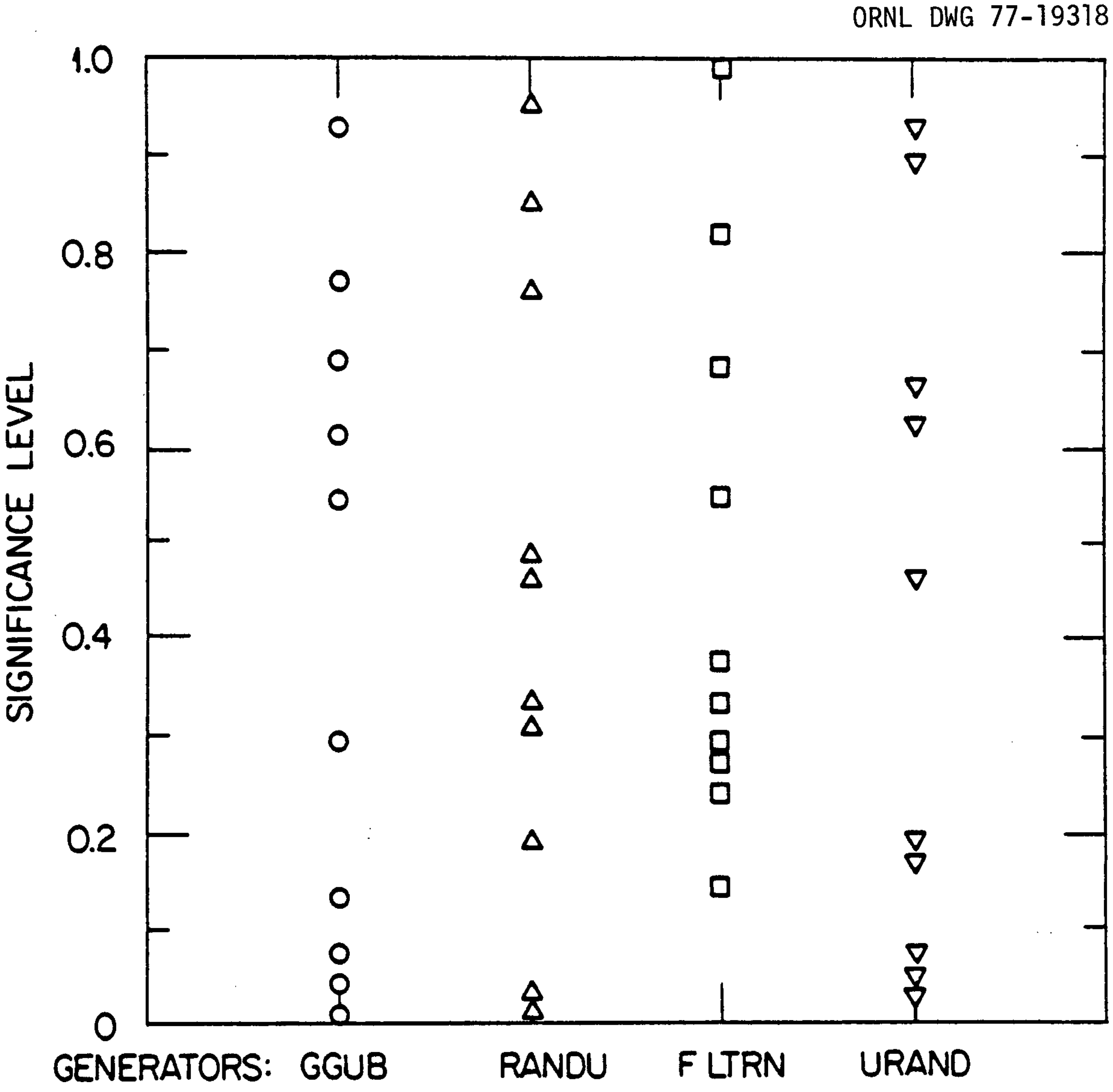


Figure B3. Significance level for runs test.

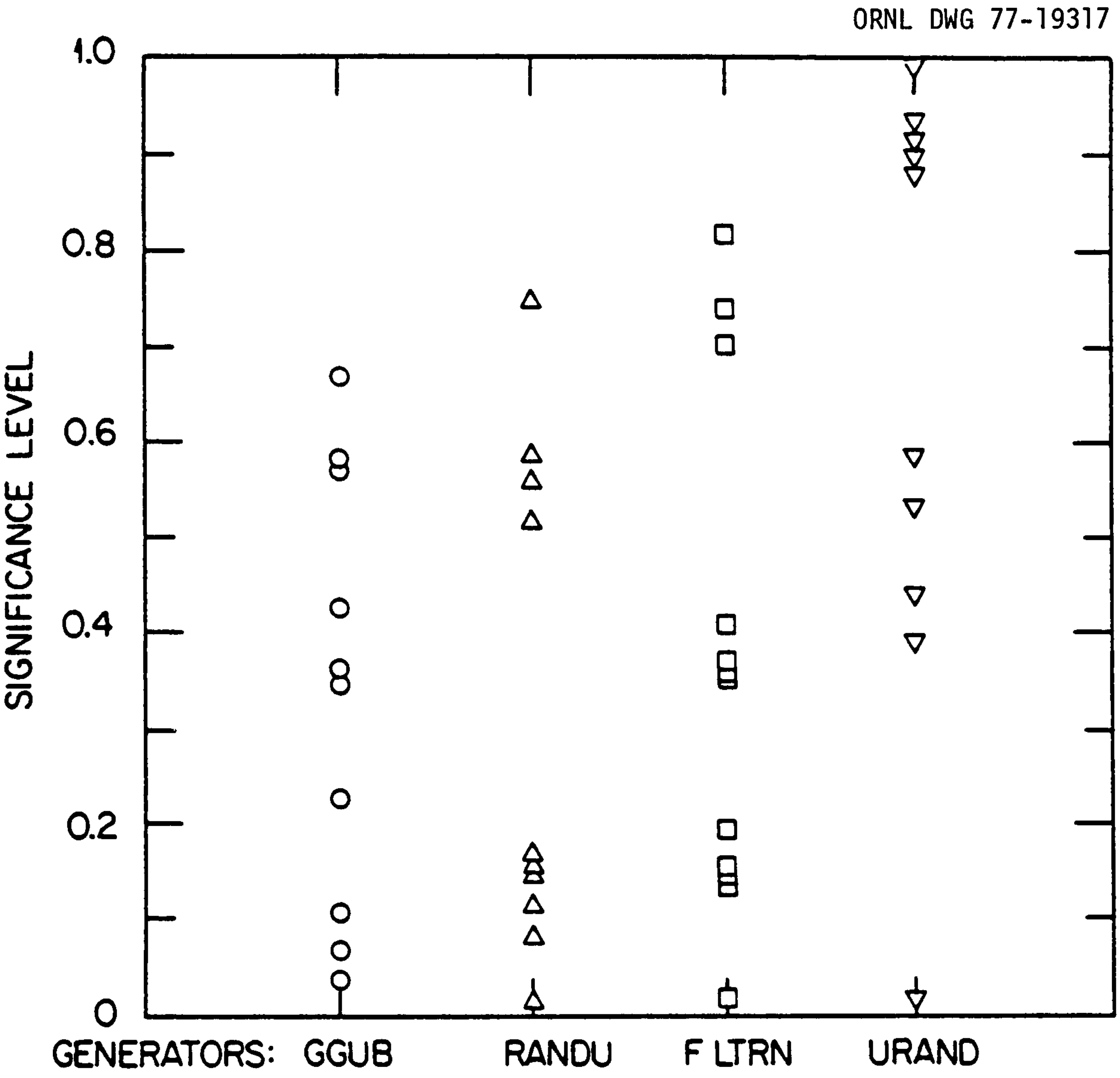


Figure B4. Significance level for Cramer von Mises test.

ORNL DWG 77-19319

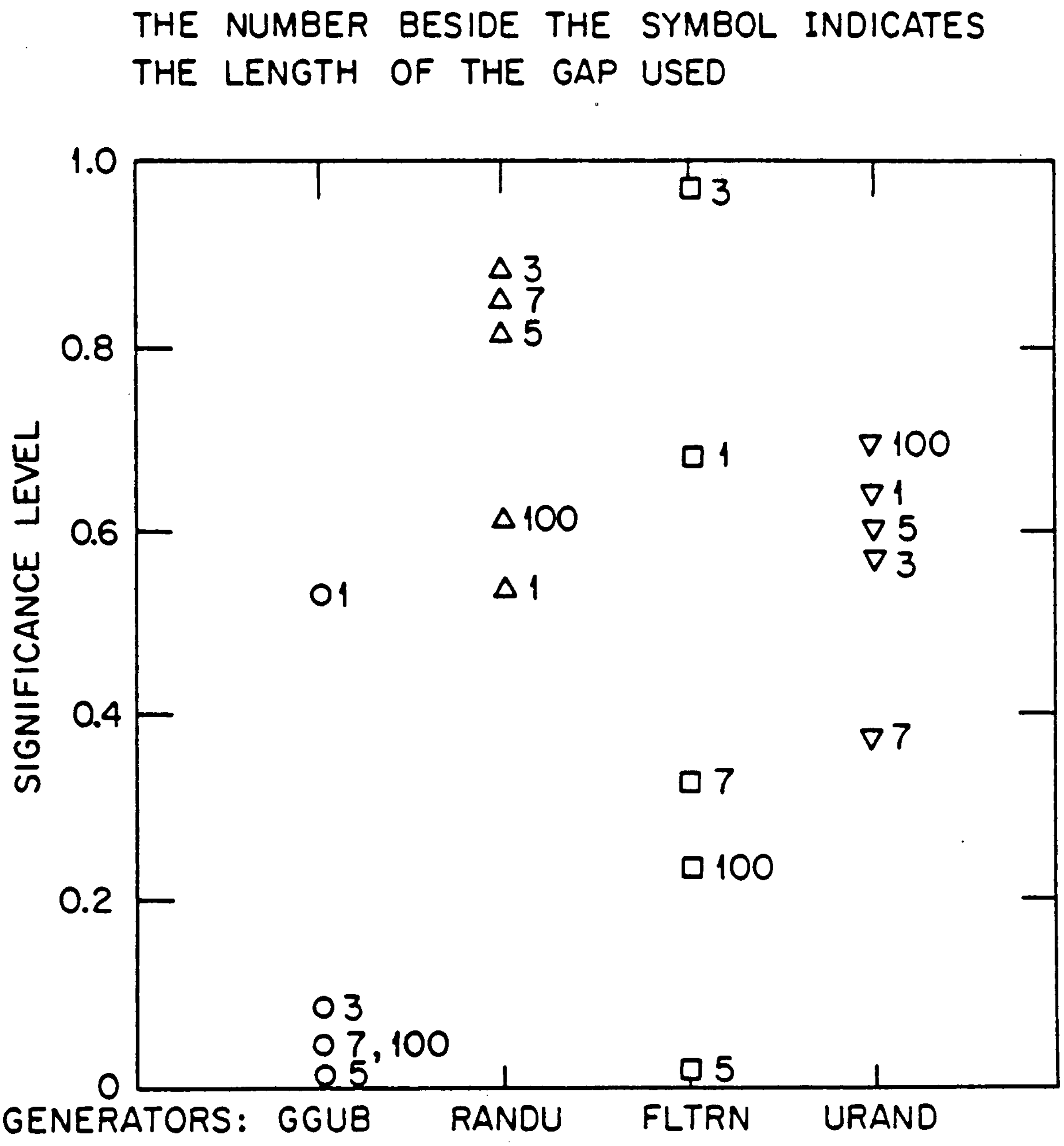


Figure B5. Significance level for the gap test.

ORNL DWG 77-19320

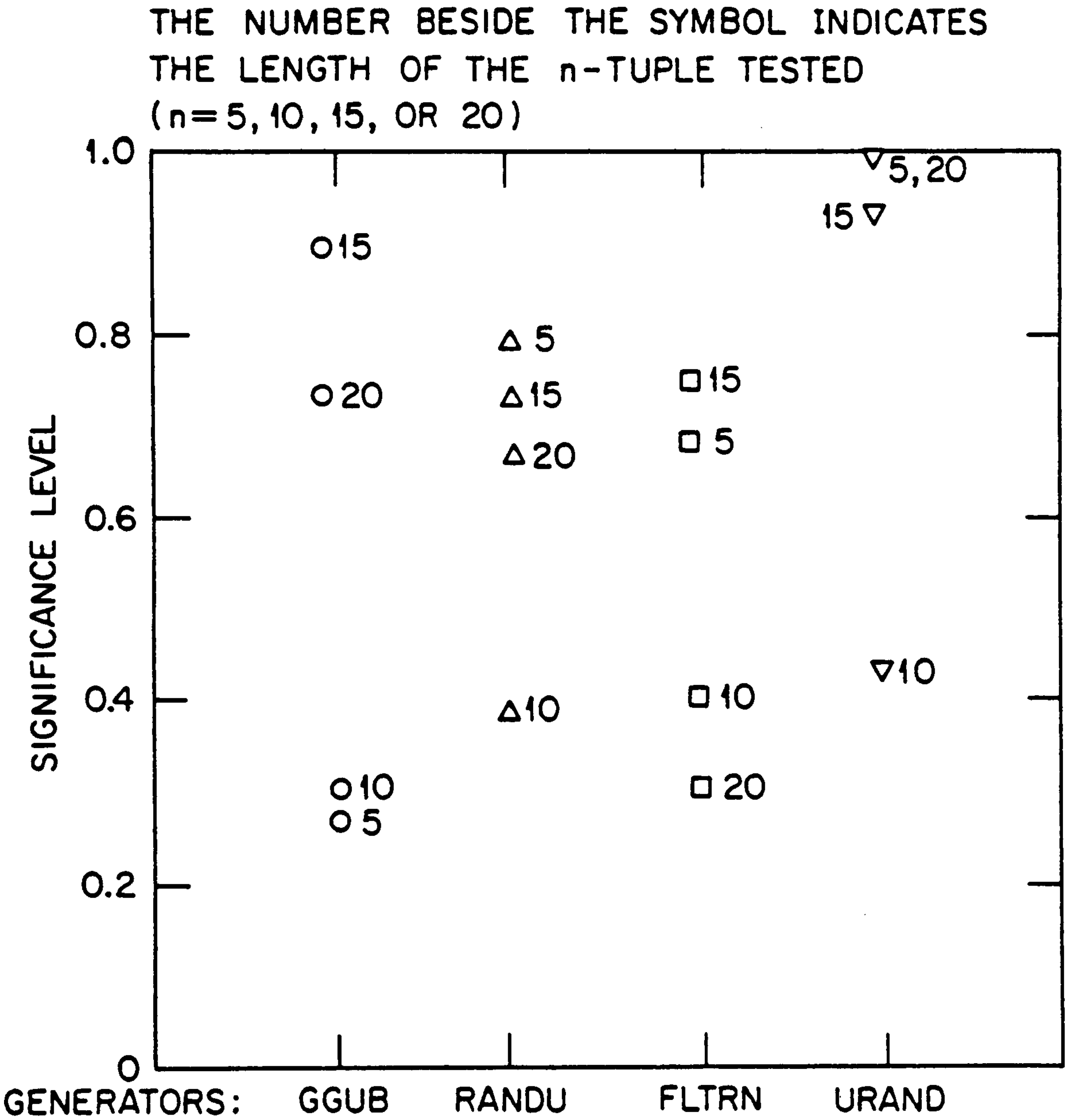


Figure B6. Significance level for the maximum of an n-tuple.

ORNL DWG 77-19321

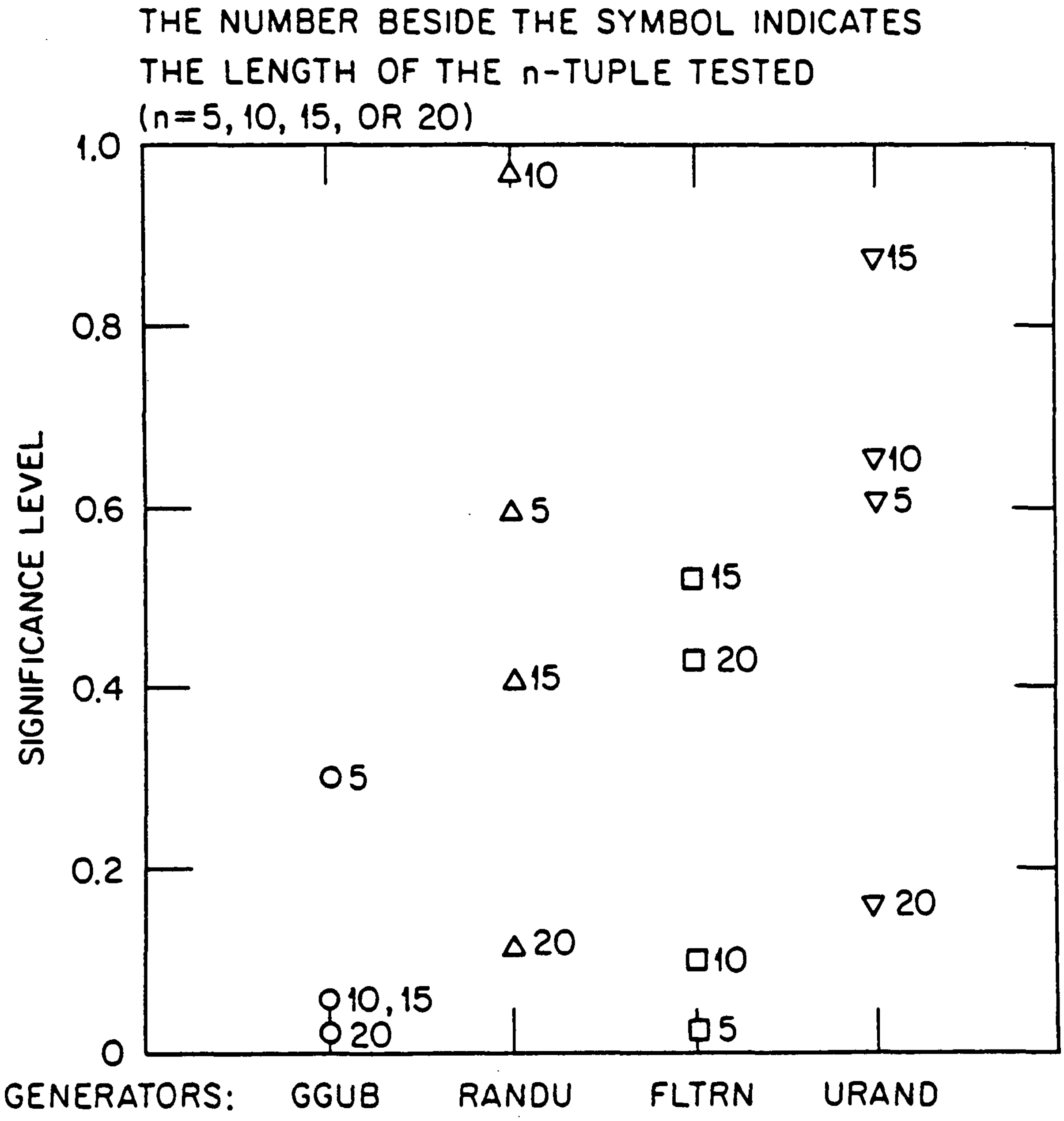


Figure B7. Significance level for the minimum of an n-tuple.

Table B1. Number of significance levels below 0.10 for four pseudorandom number generators[1]

Name	Frequency test	Runs test	Cramer von Mises	Gap test	Maximum n-tuple	Minimum n-tuple	Total
GGUB	2	3	2	3	0	3	13
RANDU	2	2	2	0	0	1	7
FLTRN	1	0	1	1	0	2	5
URAND	0	3	1	0	0	0	4

[1] Testing H_0:the sequences generated are uniformly random versus H:the sequences are not uniformly random.

lists the order of the generators by the number of significance levels which occurred below 0.10.

FLTRN and URAND give the best results in the sequences tested. URAND gave slightly better results in the tests shown in Fig. B6 and also required less time than FLTRN to generate 5000 numbers. Therefore, URAND was chosen for use as a uniform number generator.

TESTRN is designed to evaluate an arbitrary number of pseudo-uniform random number generators using six statistical tests of uniformity. The input parameters to TESTRN are given below, followed by a description of the required subroutine RANDYR. Figure B8 is a flowchart of the program.

The first input card is a general data card; the next four describe the generator and should be supplied for every one tested.

Card 1 Variables=NGEN, NRAN, NTIMES Format=(3I5)

NGEN is the number of generators to be tested.

NRAN is the size of each data set to be generated (<10,000).

NTIMES is the number of times the data sets of size NRAN are to be generated.

Card 2 Variables=TITLE Format=(20A4)

TITLE is the alphanumeric title for the pseudouniform random number generator.

ORNL DWG 78-6922

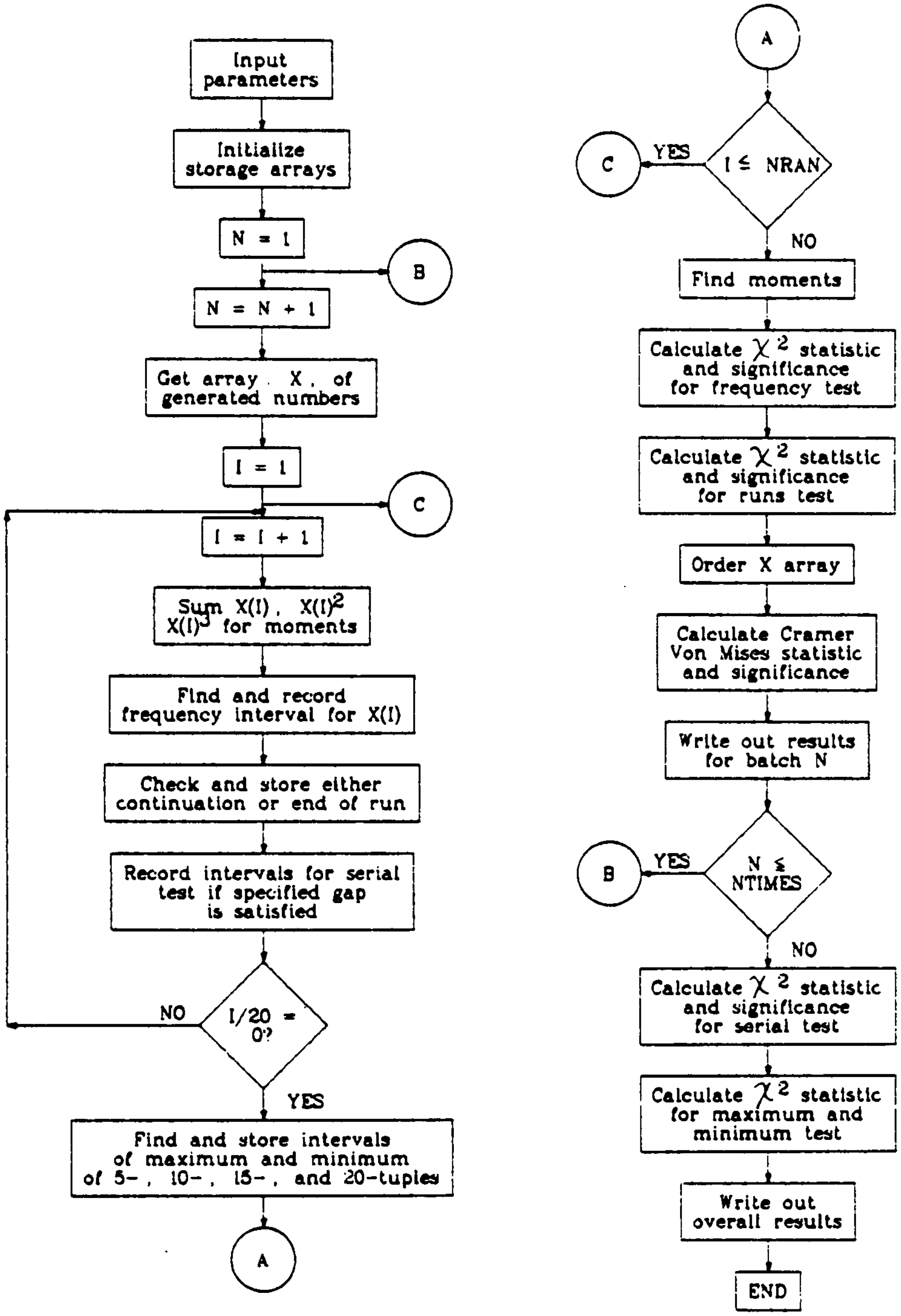

Input parameters
Initialize storage arrays
N = 1
B
N = N + 1
Get array X, of generated numbers
I = 1
C
I = I + 1
Sum X(I), X(I)^2 X(I)^3 for moments
Find and record frequency interval for X(I)
Check and store either continuation or end of run
Record intervals for serial test if specified gap is satisfied
NO
I/20 = 0?
YES
Find and store intervals of maximum and minimum of 5-, 10-, 15-, and 20-tuples
A
A
C
YES
I ≤ NRAN
NO
Find moments
Calculate χ^2 statistic and significance for frequency test
Calculate χ^2 statistic and significance for runs test
Order X array
Calculate Cramer Von Mises statistic and significance
Write out results for batch N
B
YES
N ≤ NTIMES
NO
Calculate χ^2 statistic and significance for serial test
Calculate χ^2 statistic for maximum and minimum test
Write out overall results
END

Figure B8. Flowchart of TESTRN. This program tests the hypothesis that a sequence of generated numbers is uniform.

Card 3 Variables=NOPT Format=(I5)

NOPT=0 (default) means an initalization factor(seed) will be read in; if NOPT is not equal to 0 no initialization of the generator will be done.

Card 4 Variables=FMT Format=(20A4)

FMT is the format to be used in reading in the initalization factor; the variable read in will be a double length word. This card should not be in the input if NOPT is not equal to 0.

Card 5 Variable=RINT Format=FMT

RINT is the initialization factor(seed). This card should not be in the input if NOPT is not equal to 0.

The user-supplied routine RANDYR should generate NRAN random numbers for NG generators. RANDYR is called once for every data set. The argument list of RANDYR is ORD, NRAN, RINT, NG, and NT where

ORD is an array of NRAN pseudouniform random numbers upon return to RANDYR.

NRAN is the number of pseudorandom numbers to be generated.

RINT is the initialization factor read in if NOPT=0.

NG is the sequential number of the generator that is to be tested.

NT is the number of sets.

An example of RANDYR is given in the program listing.

```
C         TESTRN PROGRAM LISTING
          DIMENSION IFINT(100),ISINT(10,10,10),
     A    IFMAX(4,100),NR(10),IFMIN(4,100),
     B    ORD(10000),OX(4),ON(4),TITLE(20),
     C    NRUNU(7),NRUND(7),RUNS(8),EXPR(8)
          DATA EXPM1/0.5/,EXPM2/.33333333/,EXPM3/.25/
          LOGICAL PLU
          DATA DF/99.0/
          READ(5,10100) NGEN,NRAN,NTIMES
          NG=1
          FN=1./FLOAT(NRAN)
          FN1=1./FLOAT(NRAN-1)
          READ(5,10200) NS,(NR(M),M=1,NS)
2         NT=1
          READ(5,10000) TITLE
          READ(5,10100) NOPT
          WRITE(6,10010) TITLE
          WRITE(6,10020) NRAN,NTIMES
          NT=1
          DO 5 K=1,100
          DO 5 J=1,4
          IFMAX(J,K)=0
          IFMIN(J,K)=0
5         CONTINUE
          DO 15 J=1,10
          DO 15 K=1,10
          DO 15 L=1,10
          ISINT(L,K,J)=0
15        CONTINUE
C***      SET UP EXPECTED NO. OF RUNS
          FAC=6.
          DO 20 J=1,7
          P=J
          FAC=FAC*(P+3.)
          P2=P*P
          P3=P2*P
          EXPR(J)=2.*(NRAN*(P2+3.*P+1.)-(P3+3.*P2-P-4.))/FAC
20        CONTINUE
          TE=(4.*NRAN-2.)/6.
          EXPR(8)=(16.*NRAN-142.)*2.7557391E-7
1         I=1
C***      ZERO OUT ALL STORAGE ARRAYS
C***      IFINT STORES COUNTS FOR THE INTERVALS OF
C***      FREQUENCY TEST
C***      IFMAX AND IFMIN STORE COUNTS FOR MAX
C***      AND MIN OF N TUPLES(N=5,10,15,20)
C***      ISINT STORES COUNTS FOR INTERVALS OF SERIAL TEST
C***      XM1,XM2,XM3 ARE THE FIRST,SECOND, AND THIRD MOMENTS
C***      NRUNU,NRUND STORE THE NUMBER OF 1-7 RUNS UP AND DOWN
C***      ITRUNU,ITRUND, STORE TOTAL
C***      RUNS UP AND DOWN.
          DO 10 K=1,100
          IFINT(K)=0
```

```
10        CONTINUE
          XM1=0.0
          XM2=0.0
          XM3=0.0
          DO 30 J=1,7
          NRUNU(J)=0
          NRUND(J)=0
30        CONTINUE
          ITRUND=0
          MAXRUU=0
          PLU=.FALSE.
          NRUN=0
          ITRUNU=0
          MAXRUD=0
          CALL RANGET(ORD,NRAN,TIME,NOPT,NT,NTIMES,NG)
          XRL=ORD(1)
          DO 100 I=1,NRAN
          XR=ORD(I)
C***    FIND MOMENTS
110       XM1=XM1+XR
          XM2=XM2+XR*XR
          XM3=XM3+XR*XR*XR
C***    FIND INTERVAL FOR FREQUENCY TEST
          INF=XR*100+1
          IFINT(INF)=IFINT(INF)+1
C***    CHECK FOR RUNS
          XF=XRL-XR
          IF(XF) 200,205,210
200       IF(PLU)GO TO 202
C***      IF RANDOM NUMBER>LAST RANDOM NUMBER
C***      BEGINNING OR CONTINUING
C***      A RUN UP
C***      IFPLU TRUE, BEGINNING A RUN UP
C***      IF PLU FALSE, CONTINUING A RUN UP
          NRUN=NRUN+1
          GO TO 220
202       PLU=.FALSE.
          ITRUND=ITRUND+1
          IF(NRUN.LE.7) NRUND(NRUN)=NRUND(NRUN)+1
          IF(NRUN.GT.MAXRUD) MAXRUD=NRUN
205       NRUN=1
          GO TO 220
C***      IF RANDOM NUMBER <LAST RANDOM NUMBER
C***      BEGINNING OR CONTINUING A RUN DOWN
C***      IF PLU FALSE, BEGINNING A RUN DOWN
C***      IF PLU TRUE, CONTINUING A RUN DOWN
210       IF(.NOT. PLU) GO TO 212
          NRUN=NRUN+1
          GO TO 220
212       PLU=.TRUE.
          ITRUNU=ITRUNU+1
          IF(NRUN.LE.7) NRUNU(NRUN)=NRUNU(NRUN)+1
          IF(NRUN.GT.MAXRUU) MAXRUU=NRUN
```

```
        NRUN=1
220     CONTINUE
       XRL=XR
        INS=XR*10+1
C**  CHECK FOR SERIAL TEST
        DO 300 M=1,NS
        IF(I.LT.NR(M)) GO TO 300
        JNS=ORD(I-NR(M))*10+1
        ISINT(JNS,INS,M)=ISINT(JNS,INS,M)+1
300     CONTINUE
        IF(MOD(I,20).NE.0) GO TO 100
C***  CHECK FOR MAX AND MIN OF N-TUPLES
C***    OX CONTAINS MAXIMUM OF THE 4 5-TUPLES
C***    ON CONTANS  MINIMUM OF THE 4 5-TUPLES
C***    IFMAX(1,*) AND FIMIN(1,*) STORES COUNTS OF
C***    MAX AND MIN OF  5-TUPLES
C***    IFMAX(2,*) AND IFMIN(2,*) STORES COUNTS OF
C***    MAX AND MIN OF 10-TUPLES
C***    IFMAX(3,*) AND IFMIN(3,*) STORES COUNTS OF
C***    MAX AND MIN OF 15-TUPLES
C***    IFMAX(4,*) AND IFMIN(4,*) STORES COUNTS OF
C***    MAX AND MIN OF 20-TUPLES
350     DO 500 M=1,4
        IADD=5*(M-1)+I-20
        OX(M)= AMAX1(ORD(1+IADD),ORD(2+IADD),ORD(3+IADD),
     A  ORD(4+IADD), ORD(5+IADD))
        IOX=(OX(M)**5)*100.+1
        IFMAX(1,IOX)=IFMAX(1,IOX)+1.
        ON(M) =AMIN1(ORD(1+IADD),ORD(2+IADD),ORD(3+IADD),
     A ORD(4+IADD),  ORD(5+IADD))
        ION=(1.-(1.-ON(M))**5)*100.+1.
        IFMIN(1,ION)=IFMIN(1,ION)+1
500     CONTINUE
        O1X=OX(1)
        IF(OX(2).GT.O1X)  O1X=OX(2)
        O2X=OX(3)
        IF(OX(4).GT.O2X)  O2X=OX(4)
        J1X=(O1X**10)*100.+1.
        J2X=(O2X**10)*100.+1.
        IFMAX(2,J1X)=IFMAX(2,J1X)+1
        IFMAX(2,J2X)=IFMAX(2,J2X)+1
        O1N=ON(1)
        IF(ON(2).LT.O1N)  O1N=ON(2)
        O2N=ON(3)
        IF(ON(4).LT.O2N)  O2N=ON(4)
        J1N=(1.-(1.-O1N)**10)*100.+1.
        J2N=(1.-(1.-O2N)**10)*100.+1.
        IFMIN(2,J1N)=IFMIN(2,J1N)+1
        IFMIN(2,J2N)=IFMIN(2,J2N)+1
        IF(OX(3).GT.O1X)  O1X=OX(3)
        IF(ON(3).LT.O1N)  O1N=ON(3)
        J1X=(O1X**15)*100.+1.
        J1N=(1.-(1.-O1N)**15)*100.+1.
```

```
          IFMAX(3,J1X)=IFMAX(3,J1X)+1
          IFMIN(3,J1N)=IFMIN(3,J1N)+1
          IF(OX(4).GT.O1X)  O1X=OX(4)
          IF(ON(4).LT.O1N)  O1N=ON(4)
          J1X=(O1X**20)*100.+1.
          J1N=(1.-(1.-O1N)**20)*100.+1.
          IFMAX(4,J1X)=IFMAX(4,J1X)+1
          IFMIN(4,J1N)=IFMIN(4,J1N)+1
100       CONTINUE
C***      FIND MOMENTS
1000      XM1=XM1*FN
          XM2=XM2*FN
          XM3=XM3*FN
          CHI=0.0
C***      FIND CHI SQUARE DISTRIBUTION FOR FREQUENCY TEST
          RN=0.01*NRAN
          DO 1100 N=1,100
          SUM=IFINT(N)-RN
          CHI=CHI+SUM*SUM
1100      CONTINUE
          CHI=CHI/RN
          WRITE(6,11100)  NT,XM1,EXPM1,XM2,EXPM2,XM3,EXPM3
          CALL MDCH(CHI,DF,P,IER)
          P=1.-P
          WRITE(6,11200)CHI,P
          WRITE(6,11300)
          IF(PLU)  GO TO 1120
          ITRUND=ITRUND+1
          IF(NRUN.LE.7)  NRUND(NRUN)=NRUND(NRUN)+1
          IF(NRUN.GT.MAXRUD)  MAXRUD=NRUN
          GO TO 1122
1120      IF(NRUN.LE.7)  NRUNU(NRUN)=NRUNU(NRUN)+1
          IF(NRUN.GT.MAXRUU)  MAXRUU=NRUN
1122      CONTINUE
          ITRUNT=ITRUNU+ITRUND
          TR=0.0
          CHI=0.0
          DO 1150 L=1,7
          NTRL=NRUNU(L)+NRUND(L)
          TR=TR+NTRL
          SUM=NTRL-EXPR(L)
          CHI=CHI+SUM*SUM
1150      WRITE(6,11350)  L,NRUNU(L),NRUND(L),NTRL,EXPR(L)
          WRITE(6,11355)  ITRUNU,ITRUND,ITRUNT,TE
          MAXFUN=MAXRUU
          IF(MAXRUD.GT.MAXRUN)MAXRUN=MAXRUD
          WRITE(6,11365)  MAXRUU,MAXRUD,MAXRUN
          SUM=ITRUNT-TR-EXPR(8)
          CHI=CHI+SUM*SUM
          CHI=CHI/7.
          CALL MDCH(CHI,7.,P,IER)
          P=1.-P
          WRITE(6,11375)CHI,P
```

```
C***    ORDER THE LAST NRAN RANDOM NUMBERS FOR
C***    CRAMER VON MISES TEST
        DO 1200 J=2,NRAN
        M=J
 1225   IF(ORD(M).GE.ORD(M-1)) GO TO 1200
        ORDH=ORD(M)
        ORD(M)=ORD(M-1)
        ORD(M-1)=ORDH
        M=M-1
        IF(M.GT.1) GO TO 1225
 1200   CONTINUE
C***    DO SUMMATION FOR CRAMER VON MISES TEST
        FN2=1./FLOAT(2*NRAN)
        SUM=0.0
        DO 1300 J=1,NRAN
        FN1=J*2-1
        FAC=ORD(J)-FN1*FN2
        SUM=SUM+FAC*FAC
 1300   CONTINUE
        OMEG=1./FLOAT(12*NRAN)+SUM
        CALL CUM(OMEG,P)
        P=1.-P
        WRITE(6,11400) OMEG,P
C***    ANALYSIS COMPLETED FOR THIS PASS
C***    CHECK IF SHOULD DO ANOTHER PASS
        NT=NT+1
        IF(NT.LE.NTIMES) GO TO 1
 2000   CONTINUE
C***    FIND CHI-SQUARE DISTRIBUTION FOR SERIAL TEST
        WRITE(6,11450)
        DO 2200 J=1,NS
        CHI=0.0
        NT=NRAN*NTIMES-NR(J)*NTIMES
        RNTD=0.01*NT
        DO 2100 K=1,10
        DO 2100 L=1,10
        SUM=ISINT(L,K,J)-RNTD
        CHI=CHI+SUM*SUM
 2100   CONTINUE
        CHI=CHI/RNTD
        CALL MDCH(CHI,DF,P,IER)
        P=1.-P
        WRITE(6,11500) NR(J),CHI,P
 2200   CONTINUE
        WRITE(6,11600)
        DO 3000 L=1,4
        LEN=L*5
        IFAC=LEN
        IF(L.EQ.3) IFAC=20
        RFAC=NTIMES*NRAN*0.01/FLOAT(IFAC)
        CHI1=0.0
        CHI2=0.0
        DO 2500 J=1,100
```

```
          SUM=IFMAX(L,J)-RFAC
          CHI1=CHI1+SUM*SUM
          SUM=IFMIN(L,J)-RFAC
          CHI2=CHI2+SUM*SUM
2500      CONTINUE
          CHI1=CHI1/RFAC
          CHI2=CHI2/RFAC
          CALL MDCH(CHI1,DF,P1,IER)
          CALL MDCH(CHI2,DF,P2,IER)
          P1=1.-P1
          P2=1.-P2
          WRITE(6,11650) LEN,CHI1,P1,CHI2,P2
3000      CONTINUE
          WRITE(6,11800) NRAN,TIME
          NG=NG+1
          IF(NG.LE.NGEN) GO TO 2
          STOP
10000     FORMAT(20A4)
10100     FORMAT(3I5)
10200     FORMAT(16I5)
10010     FORMAT('1TEST FOR QUALITY OF PSEUDO ',
      A   'RANDOM NUMBERS'///
      B   ' GENERATOR NAME: ',20A4///)
10020     FORMAT(' A DATA SET OF SIZE',I5,' WAS GENERATED ',
      A   I5,' TIMES FOR THIS TEST.'//)
11100     FORMAT('0*************** RESULTS FROM DATA SET ',I5,
      A   ' ***************'//
      B   ' MOMENTS',10X,'CALCULATED',5X,'EXPECTED'/
      C   5X,'FIRST',5X,F10.5,5X,F10.5/
      D   5X,'SECOND',4X,F10.5,5X,F10.5/
      E   5X,'THIRD',5X,F10.5,5X,F10.5////)
11200     FORMAT(' FREQUENCY TEST (CHI-SQUARE)'//
      A   5X,'STATISTIC = '
      B   3X,F10.2/5X,'SIGNIFICANCE = ',F10.2////)
11300     FORMAT(' RESULTS OF RUN TEST'//
      A   5X,'NUMBER OF RUNS',6X,'SUCESSIVE',
      B   5X,'SUCCESSIVE',8X,'TOTAL'/
      C   7X,'OF LENGTH',9X,'INCREASES',5X,
      D   'DECREASES',9X,'RUNS',9X,
      E   'EXPECTED'/)
11350     FORMAT(16X,I5,4X,3(I7,8X),F10.2)
11355     FORMAT(4X,' TOTAL',15X,3(I7,8X),F10.2)
11365     FORMAT(4X,' LONGEST RUN' ,9X,3(I7,8X)/)
11375     FORMAT(4X,' RESULTANT CHI-SQUARE STATISTIC= '
      A   ,F10.2/4X,' SIGNIFICANCE = ',F10.4////)
11400     FORMAT(' CRAMER VON MISES STATISTIC = ',F10.3/
      A   5X,'SIGNIFICANCE = ',F10.3////)
11450     FORMAT(' OVERALL SERIAL TEST RESULTS:'/)
11500     FORMAT(  4X,' FOR GAP OF',I5,
      A   ' CHI-SQUARE STATISTIC= ',F10.2/
      A   4X,' RESULTANT SIGNIFICANCE = ',F10.4////)
11600     FORMAT(' OVERALL MAX AND MIN TEST RESULTS'//
      A   4X,' LENGTH OF N-TUPLE',6X,'CHI-SQUARE MAXIMUM',12X,
     B  'CHI-SQUARE MINIMUM'/
     C  25X,'STATISTIC',10X,'SIGNIFICANCE',
     D  9X,'STATISTIC',10X,'SIGNIFICANCE'/)
11650   FORMAT(15X,I5,3X,F10.2,3(10X,F10.2))
11800   FORMAT('0'//' THE TIME REQUIRED TO GENERATE ',
     A  I6,' RANDOM NUMBERS WAS ON THE AVERAGE ',
     B  F8.3,' SECONDS.')
        END
        SUBROUTINE RANGET(ORD,NRAN,TIME,NOPT,NT,NTIMES,NG)
        REAL*8 RINT
        DIMENSION ORD(1),FMT(20)
        IF(NT.GT.1) GO TO 10
        ITT=0
        RINT=0.0D0
        IF(NOPT.NE.0) GO TO 10
        READ(5,10100) FMT
        READ(5,FMT) RINT
10      IT1=ICLOCK(0)
        CALL RANDYR(ORD,NRAN,RINT,NG,NT)
        IT2=ICLOCK(0)
        ITT=IT2-IT1+ITT
        IF(NT.LT.NTIMES) RETURN
        TIME=(ITT/FLOAT(NTIMES))*0.01
        RETURN
10100   FORMAT(20A4)
        END
        DOUBLE PRECISION FUNCTION KR(N)
        IMPLICIT REAL*8(A-H,O-Z)
        DATA P1,P2,P3,P4,PM/.8840702298758D0,
     A  .9113127802887O3D0,.9587208247904 63D0,
     B  .9733109541738 98D0,.9866554770869488D0/
        DATA C1,C2/.479727404222441D0,2.216035867166471D0/
        DATA B1,B2,B3/-.595507138015 94D0,1.105473661022 07D0,
     A  -.63083480192196D0/
        DATA H1,H2,H3/.0533775495068 86D0,.0492644963731 28D0,
     A.03424050375O111D0/
        DATA R1,R2,R3/.805577924423817D0,.8728349766717 9D0,
     A  .7555915316676O1D0/
        DATA SL,SRTP/.180025191068563D0,.398942280401432 7D0/
        DATA CC,TC/1.1311316354441 8D0,2.455407482284127D0/
C===MISING VARIABLE
        U=RAN(1)
C===BIG TRIANGLE
        IF(U.GT.P1) GO TO 10
        KR=C2*(RAN(1)+CC*U-1.)
        RETURN
C***    SMALL TRIANGLE
10      IF(U.GT.P4) GO TO 50
        IF(U.GT.P3) GO TO 30
        C=C1
        IF(U.GT.P2) GO TO 20
        B=B1
        H=H1
        R=R1
        GO TO 40
20      B=B2
        H=H2
        R=R2
        GO TO 40
30      C=C2
        B=B3
        H=H3
        R=R3
C***TRIANGLE REJECTION
40      V=RAN(1)
        W=RAN(1)
        Z=V-W
        KR=DMIN1(V,W)*B+C
        IF(DMAX1(V,W).LE.R) GO TO 45
        IF(KR.LT.0.) GO TO 40
        IF(H*DABS(Z).GT.SRTP*DEXP(-.5*KR*KR)+SL*(KR-C2))
     A  GO TO 40
45      KR=DSIGN(KR,Z)
        RETURN
C***TAIL
50      V=RAN(1)
        KR=TC-DLOG(RAN(1))
        IF(V*V*KR.GT.TC) GO TO 50
        KR=DSIGN(DSQRT(2.*KR),U-PM)
```

```
RETURN
END
DOUBLE PRECISION FUNCTION RAN(N)
IMPLICIT REAL*8(A-H,O-Z)
COMMON/SEED/IRAND
IRAND=IRAND*1162261467
IF(IRAND.LT.0) IRAND=-IRAND
RAN=FLOAT(IRAND)*0.4656612873E-9
RETURN
END
```

ORNL/CSD/TM-65

INTERNAL DISTRIBUTION

1. C. K. Bayne
2. J. J. Beauchamp
3-22. C. L. Begovich
23. J. L. Bledsoe
24. H. P. Carter/A. A. Brooks/ CSD Library
25. K. N. Fischer
26. D. A. Gardiner
27. D. G. Gosslee
28. V. E. Kane
29. B. D. Murphy
30. C. W. Nestor, Jr.
31. A. C. Olson
32. M. R. Patterson
33. J. S. Wassom
34. D. A. Wolf
35-36. Central Research Library
37. Document Reference Section Y-12
38-40 Laboratory Records
41. Laboratory Records - RC
42. ORNL Patent Office

EXTERNAL DISTRIBUTION

43. Chief, Mathematics and Geoscience Branch, Department of Energy, Washington, DC 20545.

44. Office of Assistant Manager for Energy Research and Development, DOE/ORO.

45. J. N. Rogers, Division 8324, Sandia Laboratories, Livermore, CA 94550.

46-72. Technical Information Center, Department of Energy, P. O. Box 62, Oak Ridge, TN 37830.