Computers & Geosciences Vol. 19, No. 3, pp. 303–342, 1993
Printed in Great Britain

0098-3004/93 $6.00 + 0.00
Pergamon Press Ltd

PRINCIPAL COMPONENTS ANALYSIS (PCA)*

ANDRZEJ MAĆKIEWICZ[1] and WALDEMAR RATAJCZAK[2]

[1]Department of Mathematics, Technical University of Poznań, Piotrowo 3a, Poznań and
[2]Institute of Socio-Economic Geography and Spatial Planning, Adam Mickiewicz University,
Fredry 10, Poznań, Poland

(Received 20 May 1992; accepted 15 September 1992)

Abstract—Principal Components Analysis (PCA) as a method of multivariate statistics was created before the Second World War. However, the wider application of this method only occurred in the 1960s, during the "Quantitative Revolution" in the Natural and Social Sciences.

The main reason for this time-lag was the huge difficulty posed by calculations involving this method. Only with the advent and development of computers did the almost unlimited application of multivariate statistical methods, including principal components, become possible.

At the same time, requirements arose for precise numerical methods concerning, among other things, the calculation of eigenvalues and eigenvectors, because the application of principal components to technical problems required absolute accuracy.

On the other hand, numerous applications in Social Sciences gave rise to a significant increase in the ability to interpret these nonobservable variables, which is just what the principal components are. In the application of principal components, the problem is not only to do with their formal properties but above all, their empirical origins.

The authors considered these two tendencies during the creation of the program for principal components. This program—entitled PCA—accompanies this paper. It analyzes consecutively, matrices of variance–covariance and correlations, and performs the following functions:

—the determination of eigenvalues and eigenvectors of these matrices,
—the testing of principal components,
—the calculation of coefficients of determination between selected components and the initial variables, and the testing of these coefficients,
—the determination of the share of variation of all the initial variables in the variation of particular components,
—construction of a dendrite for the initial set of variables,
—the construction of a dendrite for a selected pattern of the principal components,
—the scatter of the objects studied in a selected coordinate system.

Thus, the PCA program performs many more functions especially in testing and graphics, than PCA programs in conventional statistical packages. Included in this paper are a theoretical description of principal components, the basic rules for their interpretation and also statistical testing.

Key Words: Principal Components Analysis, Variance–covariance matrix, Coefficients of determination, Eigenvalues, Eigenvectors, Correlation matrix, Bartlett's statistics, FORTRAN 77.

DESCRIPTION OF THE PRINCIPAL COMPONENTS METHOD

The basic aim of the analysis utilizing principal components is a reduction of the dimensions of the observation space in which given objects are studied (Kendall, 1983; Jackson and Hearne, 1973). The reduction is obtained by creating new linear combinations of variables characterizing the objects studied. These combinations, termed principal components, must satisfy certain mathematical and statistical conditions. They will be discussed in detail in subsequent sections.

The starting point in the principal components method is an observation matrix $\mathbf{X}$ in which column vectors list observations characterizing an object with respect to random variables $X_1, X_2, \ldots, X_p$.

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{22} & \cdots & x_{1n} \\ x_{21} & x_{12} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{p1} & x_{p2} & \cdots & x_{pn} \end{bmatrix}.$$

Each column vector represents a point in a p-dimensional space. Because the observation matrix $\mathbf{X}$ is compiled for a sample of the entire population (numbers p and n are finite), the variance–covariance matrix $\mathbf{S}$ derived from observations of random variables is an estimator of general variance–covariance matrix $\boldsymbol{\Sigma}$, whereas the vector of mean values $\bar{\mathbf{x}}$ is an estimator of the general vector $\mathbf{U}$. Thus as mentioned the task of the principal components method is to determine linear combinations with a maximum

*Due to circumstances beyond the Publisher's control, this article appears in print without author corrections.

variance. Thus, the problem essentially is replacing the set of initial variables with their linear combinations, that is new variables with special properties. These new variables are termed principal components and are written in the form:

$$\mathbf{V} = \mathbf{A}'\mathbf{X} \tag{1}$$

where

V is a matrix of the new variables,
A is a matrix of orthonormal eigenvectors of matrix S, and
X is the observation matrix.

Transformation (1) is possible after determinantal Equation (2) has been solved.*

$$|\mathbf{S} - l\mathbf{I}| = 0 \tag{2}$$

where

S is a variance–covariance matrix of order $(p \times p)$,
l is the characteristic root of the determinantal equation, and
I is a unit matrix of order $(p \times p)$.

Equation (2) is a polynomial of degree p with respect to unknown l, hence it has p roots which can be ordered in such a way that

$$l_1 \geqslant l_2 \geqslant l_3 \geqslant \cdots \geqslant l_p \geqslant 0.$$

Because there is an orthonormal column eigenvector $\mathbf{A}_i$ corresponding to each root l_i, the variable V_1 derived from Equation (2) has the maximum value l_1 (maximum variance) and is termed the first principal component.

Because the sum $l_1 + l_2 + \cdots + l_p = \mathrm{tr}\,\mathbf{S}$ and is equal to the sum of the variances of matrix **S** (i.e $\sigma_{11} + \sigma_{22} + \cdots + \sigma_{pp}$), $l_1, l_2, \ldots, l_p$ defines the share of variability of particular principal components in the total variance of matrix **S**. If we consider the quotients

$$\frac{l_1}{\mathrm{tr}\,\mathbf{S}}100, \quad \frac{l_2}{\mathrm{tr}\,\mathbf{S}}100, \quad \ldots, \quad \frac{l_p}{\mathrm{tr}\,\mathbf{S}}100, \tag{3}$$

we get the percent share of each component in the variance of matrix **S**. The algorithm for the calculation of principal components is such that this is a decreasing sequence, which indicates that $l_1/\mathrm{tr}\,\mathbf{S}\,100$ is the largest quantity. Quantity l_2 corresponds to variable V_2, which therefore is termed the second principal component. It is clear that there are as many principal components as initial variables.

Each root l_i has its corresponding column vector A_i such that

$$(\mathbf{S} - l\mathbf{I})\mathbf{A}_i = 0 \quad \text{or} \quad \mathbf{S}\mathbf{A}_i = l_i\mathbf{A}_i. \tag{4}$$

*It also is possible to derive principal components by substituting the variance–covariance matrix **S** with correlation matrix **R**, that is by solving the equation $|\mathbf{R} - l\mathbf{I}| = 0$.

Because vectors $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_p$ are orthonormal, that is

$$\mathbf{A}_i'\mathbf{A}_i = 1, \quad \mathbf{A}_j\mathbf{A}_i = 0 \quad \text{for } i \neq j, \tag{5}$$

and they satisfy (2) and (4), we have

$$\mathbf{A}_i'\mathbf{S}\mathbf{A}_i = l_i, \quad \mathbf{A}_i'\mathbf{S}\mathbf{A}_j = 0 \quad \text{for } i \neq j, \tag{6}$$

$$\mathbf{I} = \mathbf{A}_1\mathbf{A}_1' + \cdots + \mathbf{A}_p\mathbf{A}_p' \tag{7}$$

and

$$\mathbf{S} = l_1\mathbf{A}_1\mathbf{A}_1' + l_2\mathbf{A}_2\mathbf{A}_2' + \cdots + l_p\mathbf{A}_p\mathbf{A}_p'. \tag{8}$$

Expression (8) is termed a spectral decomposition of a matrix **S**.

The basic property of the new variables is the lack of correlation among them (in contrast to the initial variables). The variance of the ith component is l_i, or

$$\mathrm{Var}(\mathbf{A}_i\mathbf{X}) = l_i, \tag{9}$$

whereas

$$\mathrm{Cov}(\mathbf{A}_i\mathbf{X}, \mathbf{A}_j\mathbf{X}) = 0 \quad \text{for } i \neq j.$$

Because the primary aim of principal components analysis is the reduction of the dimensions of the observation space, it is necessary at some stage to decide on how many new variables should be taken into account further study. To help with the decision, the ratio of characteristic roots to the trace of the matrix is considered. For example, if the expression $l_1/\mathrm{tr}\,\mathbf{S}\,100$ has a great value (e.g. 90%), the set of initial variables is replaced with the first component V_1. When the ratio is not so high, the next components are taken into account. Naturally, the elimination of some components from further analysis cannot follow solely from the researcher's subjective evaluation of the $l_i/\mathrm{tr}\,\mathbf{S}\,100$ quotient, but must result from the testing of the components. This problem is discussed in detail in the next section.

An important issue in Principal Components Analysis is the interpretation of the components, to help determine, after the reduction of the observation space, which initial variables have the greatest shares in the variance of particular principal components. This information can be obtained using coefficients of determination established between the components and the initial variables. It should be added that interpretation of the components differs slightly depending on whether **S** or **R** is used.

*Interpretation of principal components derived from variance–covariance matrix **S***

The coefficient of correlation between the ith component and the jth initial variable is defined by the equation:

$$r_{ij} = \frac{a_{ij}\sqrt{l_i}}{S_j}. \tag{10}$$

Hence the coefficient of determination has the form:

$$r_{ij}^2 = \frac{a_{ij}^2 l_i}{S_j^2}, \tag{11}$$

where a_{ij}^2 is the square of the element of the eigenvector A_i corresponding to the ith component and jth initial variable, l_i is the variance of the ith component, and S_j^2 is the variance of variable j.

On the basis of (8) and (9), making use of the variances and eigenvectors of all the principal components, it is possible to reconstruct the variance–covariance matrix **S**. Naturally, the product $l_1\mathbf{A}_1\mathbf{A}_1'$ has the greatest share in this reconstruction. Moreover, in for example matrix $l_1\mathbf{A}_1\mathbf{A}_1'$ the elements on the main diagonal are estimates (supplied with the help of the first component) of the variance of the jth initial variable, which can be computed from a general expression:

$$\hat{\mathrm{V}}\mathrm{ar}\, X_j = l_1 a_{1j}^2, \tag{12}$$

whereas the remaining elements are estimates of covariance, with the final element (matrix) of spectral decomposition $l_p\mathbf{A}_p\mathbf{A}_p'$ bringing the estimated variance and covariance up to real values. Given (12), (11) can be written in the form:

$$r_{ij}^2 = \frac{\hat{\mathrm{V}}\mathrm{ar}\, X_j}{\mathrm{Var}\, X_j} = \frac{\hat{S}_j^2}{S_j^2}. \tag{13}$$

It is clear that the coefficient of determination between component i and variable j is a ratio of the estimated variance of variable j to its real variance. If we consider any (ith) matrix from the spectral decomposition, then, by summing up the elements on the main diagonal

$$\sum_{j=1}^{p} \hat{S}_j^2 = \sum_{j=1}^{p} l_i a_{ij}^2$$

$$\sum_{j=1}^{p} \hat{S}_j^2 = l_i \sum_{j=1}^{p} a_{ij}^2$$

we get

$$\sum_{j=1}^{p} \hat{S}_j^2 = l_i. \tag{14}$$

Thus, by adding together the estimated variances of particular variables we arrive at the variance of the ith component. This relationship can be the theoretical basis for the interpretation of the components. Using (14), we can write (3) alternatively as

$$\frac{\sum_{j=1}^{p} \hat{S}_j^2}{\mathrm{tr}\,\mathbf{S}} 100. \tag{15}$$

Expression (15) assumes the greatest value for the first component; however, this measure should be used carefully. It follows from (14) that almost the entire variance of the ith component is made up of the estimated variance of a single variable, for example one that has a high absolute value in comparison with the remaining ones, that is one with high variance. Hence the necessity of a skillful formulation of the observation matrix, variables of which should be of a similar order of measure (15), the following dependence is proposed for use in the interpretation of the components:

$$w = \frac{\sum_{i,j}^{p} r_{ij}^2}{p} 100 \tag{16}$$

where p is the number of variables of the observation matrix, and r_{ij}^2 is the coefficient of the determination between the ith component and jth initial variable.

Equation (16) shows the percent of the variance of all the variables accounted for by the ith component. The results of measure (16) applied to components derived from the covariance matrix are as a rule lower than those of measure (15), because in reality one (e.g. the first) component seldom accounts for more than 50% of the variance of all the variables included in the observation matrix. When a variable in the observation matrix exceeds all others markedly in value, expression (15) gives a high value of the first component, and expression (16) a low one. It is so because the variance of the component depends in this situation on a single variable only, and may not be the most important one either (its significance depends solely on the adopted units of measurement). As can be seen, expression (16) shows the actual share of the ith component in the variance of all the variables. It is worth noting that the numerator of (16) is easy to calculate because the following equation holds:

$$\sum_{i,j}^{p} r_{ij}^2 = l_i \sum_{j=1}^{p} \frac{a_{ij}^2}{S_j^2}. \tag{17}$$

The variance of all the initial variables is accounted for by the component if:

$$\sum_{i,j}^{p} r_{ij}^2 = l_i \sum_{j=1}^{p} \frac{a_{ij}^2}{S_j^2} = p.$$

*Interpretation of principal components derived from the correlation matrix **R***

The discrepancies between evaluations of principal components obtained on the basis of formulae (15) and (16) do not occur if we derive them from a correlation matrix (i.e. if use is made of normalized initial variables). This is so because the following dependencies hold by virtue of an appropriate transformation of the covariance matrix (Anderson, 1958):

$$r_{ij} = a_{ij}\sqrt{l_i}, \tag{18}$$

$$r_{ij}^2 = a_{ij}^2 l_i, \tag{19}$$

$$\sum_{i,j}^{p} r_{ij}^2 = \sum_{i,j}^{p} a_{ij}^2 l_i,$$

$$\sum_{i,j}^{p} r_{ij}^2 = l_i. \tag{20}$$

Or,

$$\frac{l_i}{\mathrm{tr}\,\mathbf{S}} 100 = \frac{\sum_{i,j}^{p} r_{ij}^2}{p} 100.$$

Thus, when the correlation matrix is used, in order to determine what part of the variance of all the initial variables is accounted for (in percent) by the ith component, it is possible to employ either measure (15) or (16). There is no such optionality in the situation of the variance–covariance matrix.

TESTING PRINCIPAL COMPONENTS

Because matrices **S** and **R** are only estimators of matrices Σ and **P** (covering the entire population), the results obtained must be subjected to a verification procedure. There are several approaches to the testing of principal components (cf. Mardia, Kent, and Bibby, 1979; Anderson, 1984). They are discussed next and then used in the program. It must be emphasized that the tests used in this program are based on the asymptotic distributions of the roots and vectors of variance–covariance and correlation matrices. Thus, the size of the sample used in the calculation process may have considerable influence on the results of the testing (see Anderson, 1984, p. 468).

Testing principal components derived from variance–covariance matrix ***S***

Testing the hypothesis about the ratio of the sum of the least characteristic roots to the sum of all roots. If the sum of the characteristic roots of a few final components with relation to the trace of matrix **S** is relatively small, then there is a justifiable temptation to eliminate these components from further analysis. They can be rejected after the following hypothesis has been verified:

$$H_0\colon\ f(\lambda) = \frac{\lambda_{k+1} + \cdots + \lambda_p}{\lambda_1 + \cdots + \lambda_p} \geqslant \delta \tag{21}$$

against an alternative hypothesis:

$$H_1\colon\ f(\lambda) < \delta,$$

where δ is assumed known.

The asymptotic variance $f(\mathbf{l})$ is determined from the equation:

$$2\left(\frac{\delta}{\operatorname{tr}\Sigma}\right)^2(\lambda_1^2 + \cdots + \lambda_k^2) + 2\left(\frac{1-\delta}{\operatorname{tr}\Sigma}\right)^2(\lambda_{k+1}^2 + \cdots + \lambda_p^2). \tag{22}$$

Hypothesis H_0 is rejected if $\sqrt{n}\,[f(\mathbf{l}) - \delta]$ is smaller than the corresponding point of the standardized normal distribution multiplied by the root from (22) in which estimators are substituted for real values, that is:

$$2\left(\frac{\delta}{\operatorname{tr}\mathbf{S}}\right)^2(l_1^2 + \cdots + l_k^2) + 2\left(\frac{1-\delta}{\operatorname{tr}\mathbf{S}}\right)^2(l_{k+1}^2 + \cdots + l_p^2) \tag{23}$$

whereas

$$f(\mathbf{l}) = \frac{l_{k+1} + \cdots + l_p}{l_1 + \cdots + l_p}.$$

Determination of the confidence interval for the proportion of variance accounted for the successive principal components. The point value of the proportion of variance accounted for by successive principal components is determined from the equation:

$$\hat{\psi} = \frac{l_1 + \cdots + l_k}{l_1 + \cdots + l_p}. \tag{24}$$

However, the determination of the confidence interval for quantity $\hat{\psi}$ may supply further significant information about the boundary values of the interval in which the true value of ψ may be contained.

Assuming that

$$\hat{\alpha} = \frac{l_1^2 + \cdots + l_k^2}{l_1^2 + \cdots + l_p^2},$$

the variance of estimator $\hat{\psi}$ is calculated from the equation:

$$\hat{\tau}^2 = \frac{2\operatorname{tr}\mathbf{S}^2}{(n-1)(\operatorname{tr}\mathbf{S})^2}(\hat{\psi}^2 - 2\hat{\alpha}\hat{\psi} + \hat{\alpha}^2). \tag{25}$$

In turn, the confidence interval of $\hat{\psi}$ (for $\alpha = 0.05$) has the form:

$$\hat{\psi} \pm 1.96(\hat{\tau}^2)^{1/2}. \tag{26}$$

Testing the hypothesis that $(p-k)$ eigenvalues of matrix Σ are equal. After p principal components have been determined, it is advisable to verify the hypothesis that $(p-k)$ eigenvalues are equal. Such a situation is termed isotropy and may suggest that variance is equal in all the directions of $(p-k)$-dimensional space spanned by the last $(p-k)$ eigenvectors. The test helps determine the number of principal components used to describe objects under study.

In the testing procedure use is made of Bartlett's asymptotic approximation:

$$\left(n - \frac{2p+11}{6}\right)(p-k)\log\left(\frac{a_0}{g_0}\right) \sim \chi^2(p-k+2)(p-k-1)/2 \tag{27}$$

where

$$a_0 = \frac{l_{k+1} + \cdots + l_p}{p-k}, \quad g_0 = (l_{k+1} \times \cdots \times l_p)^{1/(p-k)},$$

n is the number of observations, and the number of degrees of freedom is calculated from the equation: $df = \frac{1}{2}(p-k+2)(p-k+1)$.

With the help of statistics (27) the zero hypothesis about the equality of eigenvalues of matrix Σ is tested:

$$H_0\colon\ \lambda_p = \lambda_{p-1} = \cdots = \lambda_{k+1}.$$

Hypothesis H_0 usually is tested sequentially, taking $k = 0$, $k = 1$, etc. If $\chi^2_{\text{obl}} \geqslant \chi^2_{\alpha,df}$, then H_0 is rejected.

Testing principal components derived from correlation matrix ***R***

Testing the hypothesis that $(p-k)$ least eigenvalues of matrix ***P*** *are equal.* When principal components are derived from correlation matrix **R** (which is the estimator of the general correlation matrix **P**), to test the hypothesis that $(p-k)$ of its eigenvalues are equal use is made of Bartlett's statistics having the form:

$$(n-1)(p-k)\log\left(\frac{a_0}{g_0}\right), \tag{28}$$

where

$$a_0 = \frac{l_{k+1} + \cdots + l_p}{p-k},$$

$$g_0 = (l_{k+1} \times \cdots \times l_p)^{1/(p-k)},\ 0 < k < p-1.$$

This expression can be treated as chi-square with $\frac{1}{2}(p-k+2)(p-k-1)$ degrees only if the first k components account for a relatively large portion of variance. This statistic is used to test the hypothesis:

$$H_0:\ \lambda_p = \lambda_{p-1} = \cdots = \lambda_{k+1}.$$

If $\chi^2_{\text{obl}} \geqslant \chi^2_{\alpha,df}$, then hypothesis H_0 is rejected.

INSTRUCTIONS FOR THE PCA SYSTEM

The Principal Components Analysis program (henceforth termed PCA) has been written in the FORTRAN-77 programming language. Being fully compatible with the FORTRAN-77 standard it can be used without further modifications on any computer equipped with a compiler of this language.

The high quality of the compilers used and the carefully selected (and tested) numerical procedures performing the main part of the calculations should guarantee a reliable and efficient performance of the program.

In this paper the PCA program is presented in the Appendix together with some comments relative to the Operating System DOS (for the IBM PC).

System requirements

The basis for a successful run of the PCA on IBM PC type machines is compliance with the system requirements and the constraints of the operating system DOS 3.30 (or later).

The PCA system is designed to be used in machines with at least 640 kbyte of random access memory (RAM). It does not use extended memory (provided that the user has not changed parameter settings in the beginning of the source code). If it is to be run in a given PC system, it is necessary to modify (once) the CONFIG.SYS file stored in this system by introducing into it the sentences

```
FILES = 20
BUFFERS = 10.
```

This is an indispensable modification, because the operation of the PCA program utilizes many more files simultaneously than the DOS standard allows.

Each time the PCA system is initiated, all programs residing in the RAM (e.g. Side-Kick, Norton Commander, etc.) should be removed.

Data preparation

In order to simplify as much as possible the process of data input and the derivation of results in the PCA system, a "file-to-file" method of information transmission has been introduced. Thus data can be prepared earlier, independently of the system in question, using an appropriate wordprocessor and stored in a file with a specified name facilitating its identification. The results, in turn (also made into a file), also can be inspected, analyzed, or printed independently of the PCA system (with reservations as discussed). Further on, we present a sample data file 'd1-pca' which will serve to demonstrate how to operate the PCA program and how it runs. This file has been prepared according to a pattern which must be followed in any other data file.

```
6
29
1.08   7.43  0.60  1.27   8.00  0.36
1.00   9.01  0.71  1.08   9.01  0.36
1.13   7.19  0.49  1.24   8.14  0.40
1.03   6.24  0.55  1.82   6.63  0.47
1.04   7.07  0.57  1.50   7.35  0.38
1.17   7.63  0.59  0.88   8.90  0.42
0.89   7.16  0.67  1.38   6.37  0.33
1.04   9.05  0.65  1.60   9.40  0.38
1.04  10.23  0.81  1.80  10.60  0.37
1.15   6.49  0.51  0.95   7.49  0.41
1.13   6.24  0.43  1.11   7.06  0.40
1.14   6.38  0.53  1.23   7.26  0.41
1.05   7.21  0.63  1.20   7.54  0.38
1.15   6.97  0.54  0.95   7.99  0.40
1.06   7.07  0.53  1.13   7.49  0.38
1.03   7.31  0.67  1.17   7.49  0.36
1.05   8.63  0.62  1.38   9.09  0.39
1.06   6.05  0.46  1.29   6.42  0.38
1.05   7.86  0.45  2.09   8.28  0.37
1.19   5.34  0.42  0.93   6.33  0.43
0.95   6.76  0.49  1.63   6.44  0.35
1.12   6.37  0.53  1.0    7.13  0.39
0.95   5.63  0.53  1.60   5.34  0.33
0.94   7.46  0.55  1.82   6.92  0.33
1.04   6.76  0.53  1.08   7.26  0.39
1.04   8.27  0.67  1.27   8.62  0.38
1.07   9.12  0.51  1.07   9.75  0.38
1.05   9.01  0.74  1.38   9.49  0.39
1.04   6.95  0.51  1.50   7.22  0.35
```

There is a single figure in the first row of the 'd1-pca' file (further on interpreted as P—the number of variables) satisfying the condition $1 \leqslant P \leqslant 50$.

The second row contains one integer (N—the number of objects) which satisfies the condition $3 \leqslant N \leqslant 200$.

Next come elements of the observation matrix (in lexicographic order) in such a way that the figure placed in the ith row in the jth position represents the value of the jth variable of the ith object.

The PCA system does not destroy the input data file.

The operation of the system and its control

Let us assume, for the sake of simplicity, that the PCA system to be operated appears, together with a suitable data file, on a selected path of a hard disc, and that the output file will be put on this path, too. To start the system, the user should write the name

PCA

and press the ENTER key. The course which the computation process will take depends crucially on the user's answers to questions asked by the system. The final part of this documentation contains an example of a dialog with the system and the results for which it provided a basic (for the d1-pca data file). Next we describe the principal elements of this dialog.

After the display of introductory information, the system asks the user for the name of the file. It should be noted that the name should be given in single quotation marks. With the assumptions mentioned previously and in the example now discussed, the correct answer to the question about the file name is

d1-pca.

The answer should be followed by pressing the ENTER key, and this holds for each next answer.

Then follows the question about the name of the output file. In our example the answer may be

wyn.1st.

It is assumed in the PCA system that the full name of the data file and an output file may be arbitrary, but its length should not exceed 60 characters.

When the names of the two files have been given, the processing and physical formulation of the output file takes place. At the same time the first few eigenvalues of the variance–covariance matrix are displayed.

Calculations are suspended temporarily when the following message appears.

EXAMINE YOUR OUTPUT FILE IN ORDER
TO CHOOSE A NEW COORDINATE
SYSTEM THEN PRESS <ENTER>
IT IS USUALLY POSSIBLE TO EXECUTE
ANY OS COMMAND NOW
Execution suspended:

This is connected with the need for the user to evaluate the results obtained thus far and to make a decision about further calculations. If the information displayed on the screen so far is not sufficient to make such a decision, it is recommended to follow the DOS command:

type wyn.1st | more

(or its equivalent, depending on the name selected for the output file). This command will result in the part of the output file so far being displayed in page order on the screen.

Having completed the PCA analysis with the use of the variance–covariance matrix, the system proceeds automatically to the second stage, namely a PCA analysis making use of the correlation matrix. The questions that follow and the form of dialog are similar to those in the first stage of calculations.

The PCA program makes possible an analysis of the objects under study in new systems of coordinates (principal components) selected by the user. In the figure, the position of an object is denoted by a number from 1 to 9, depending on how many objects have identical coordinates in a new system. If there are more than nine objects sharing the same coordinates, a special graphic sign appears. On the right-hand side of the figure identification numbers of the objects are printed.

Running the PCA system for the exemplary data series D1-PCA—Results (file WYN.LST)

```
                    *********************************
                    * PRINCIPAL COMPONENTS ANALYSIS *
                    *********************************
 *   FOR DATA FILE  D1-PCA                                             *
 NUMBER OF VARIABLES =    6                          NUMBER OF CASES= 29
 <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
              VAR.-COVARIANCE MATRIX WAS TAKEN INTO ACCOUNT
 <><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
              DESCRIPTIVE PARAMETERS OF ORIGINAL VARIABLES
```

VARIABLE	MEAN	VARIANCE	STAND. DEV.
X(1)	0.105793E+01	0.4864685E-02	0.6974729E-01
X(2)	0.734103E+01	0.1300064E+01	0.1140204E+01
X(3)	0.568621E+00	0.8680856E-02	0.9317111E-01
X(4)	0.132241E+01	0.9271486E-01	0.3044912E+00
X(5)	0.775897E+01	0.1416264E+01	0.1190069E+01
X(6)	0.381724E+00	0.9039239E-03	0.3006533E-01

```
              ##### VAR.-COVARIANCE MATRIX #####
      COL( 1)         COL( 2)         COL( 3)         COL( 4)         COL( 5)
1   0.486468D-02  -0.158013D-01  -0.229941D-02  -0.128019D-01   0.160151D-01
2  -0.158013D-01   0.130006D+01   0.777635D-01   0.779941D-01   0.124907D+01
3  -0.229941D-02   0.777635D-01   0.868086D-02   0.365850D-02   0.652537D-01
4  -0.128019D-01   0.779941D-01   0.365850D-02   0.927149D-01  -0.205957D-02
5   0.160151D-01   0.124907D+01   0.652537D-01  -0.205957D-02   0.141626D+01
6   0.151046D-02  -0.728109D-02  -0.670036D-03  -0.289727D-02   0.335006D-02
      COL( 6)
1   0.151046D-02
2  -0.728109D-02
3  -0.670036D-03
4  -0.289727D-02
5   0.335006D-02
6   0.903924D-03
*******************************************************************************
                  TESTING THE PRINCIPAL COMPONENTS
                  (FOR VARIANCE-COVARIANCE MATRIX)
___ NO.__CRITICAL DELTA__ LEFTCONFID.____ RIGHTCONFID.
     1    100          89.20              96.00
     2      4          97.60              99.10
     3      1          99.80              99.90
     4      1         100.00             100.00
     5      0           0.00               0.00
     6      0           0.00               0.00
 BARTLET'S TEST REJECTS EIGENVECTORS NO.
     5
     6
          *********  NON-ZERO EIGENVALUES  *********
    NR.                LAMBDA            PERCENTAGE OF TRACE
     1          0.2613593905742508E+01        92.57
     2          0.1631374931478640E+00         5.78
     3          0.4304858866144624E-01         1.52
     4          0.3247796870052141E-02         0.12
     5          0.3270209024478446E-03         0.01
     6          0.1384170300235664E-03         0.00
    SELECTED EIGENVECTORS (FROM 1 TO   2 ) OF THE VAR.-COVARIANCE MATRIX
          A( 1)          A( 2)
1  -0.0001155169   0.1594767997
2  -0.6902897618  -0.5309246613
3  -0.0387275414  -0.0614179494
4  -0.0208237984  -0.6376345788
5  -0.7221949295   0.5291913710
6   0.0010306523   0.0478813818
****************** VALUES OF THE PRINCIPAL COMPONENTS ******************
           V( 1)                                  V( 2)
     1   -0.1095584883804461D+02          1   -0.3684137088632902D+00
     2   -0.1277621780691318D+02          2   -0.5711549374131692D+00
     3   -0.1088636639158975D+02          3   -0.1311308911594027D+00
     4   -0.9154394532615441D+01          4   -0.8119405490473295D+00
     5   -0.1024152023331932D+02          5   -0.6714900809364176D+00
     6   -0.1173532222750920D+02          6    0.2681910528735984D+00
     7   -0.9597303384400675D+01          7   -0.1193822078561360D+01
     8   -0.1309397415020718D+02          8   -0.7065554935458379D+00
     9   -0.1478552145730417D+02          9   -0.8358495501355532D+00
    10   -0.9928464507290487D+01         10    0.8389599924975855D-01
    11   -0.9445589847792558D+01         11   -0.1117015714194618D+00
    12   -0.9693031858935081D+01         12   -0.1607771123805059D+00
    13   -0.1047145550469394D+02         13   -0.4560731083644219D+00
    14   -0.1062207319044447D+02         14    0.9132649503541419D-01
    15   -0.1033337592670940D+02         15   -0.3558322408807868D+00
    16   -0.1050531742480956D+02         16   -0.5230999872863084D+00
    17   -0.1257441980887520D+02         17   -0.5034207331926865D+00
    18   -0.8857152675013044D+01         18   -0.4782461294909233D+00
    19   -0.1146614062732514D+02         19   -0.9664908816771722D+00
    20   -0.8292967215995477D+01         20    0.1062143760071039D+00
    21   -0.9369962434875720D+01         21   -0.1082235996258010D+01
    22   -0.9587472449555461D+01         22   -0.8175395451033177D-01
    23   -0.7796465582402760D+01         23   -0.1048686945368204D+01
    24   -0.1020611846632740D+02         24   -0.1327259443636310D+01
    25   -0.9952227459998097D+01         25   -0.2837886044528224D+00
    26   -0.1198613878838233D+02         26   -0.4960124751256308D+00
```

```
        27   -0.1337860765538985D+02          27   -0.2071740963255488D+00
        28   -0.1313025519510916D+02          28   -0.5008657100114776D+00
        29   -0.1006250738816002D+02          29   -0.6743253643014311D+00
.......................................................................
   COEFFICIENTS OF DETERMINATION AND CORRELATION BETWEEN COMPONENTS V(I)
                     AND ORIGINAL VARIABLES X(J)
 BETWEEN V( 1) AND ORIGINAL VARIABLES FROM X(1) TO X( 6)
X( 1)   R^2= 0.00000717  R=    -0.00267755  SIGNIFICANT FOR  ALPHA   >  0.1
X( 2)   R^2= 0.95793511  R=    -0.97874160  SIGNIFICANT FOR  ALPHA   < 0.001
X( 3)   R^2= 0.45155994  R=    -0.67198210  SIGNIFICANT FOR  ALPHA   < 0.001
X( 4)   R^2= 0.01222387  R=    -0.11056160  SIGNIFICANT FOR  ALPHA   >  0.1
X( 5)   R^2= 0.96250418  R=    -0.98107298  SIGNIFICANT FOR  ALPHA   < 0.001
X( 6)   R^2= 0.00307136  R=     0.05541985  SIGNIFICANT FOR  ALPHA   >  0.1
 TOTAL PERCENTAGE OF VARIABILITY OF ALL ORIGINAL VARIABLES
    ACCOUNTED FOR BY V( 1)   MEASURE W( 1) = 39.79%
 BETWEEN V( 2) AND ORIGINAL VARIABLES FROM X(1) TO X( 6)
X( 1)   R^2= 0.85289210  R=     0.92352158  SIGNIFICANT FOR  ALPHA   < 0.001
X( 2)   R^2= 0.03537160  R=    -0.18807339  SIGNIFICANT FOR  ALPHA   >  0.1
X( 3)   R^2= 0.07088949  R=    -0.26625080  SIGNIFICANT FOR  ALPHA   >  0.1
X( 4)   R^2= 0.71539870  R=    -0.84581245  SIGNIFICANT FOR  ALPHA   < 0.001
X( 5)   R^2= 0.03225781  R=     0.17960461  SIGNIFICANT FOR  ALPHA   >  0.1
X( 6)   R^2= 0.41376644  R=     0.64324680  SIGNIFICANT FOR  ALPHA   < 0.001
 TOTAL PERCENTAGE OF VARIABILITY OF ALL ORIGINAL VARIABLES
    ACCOUNTED FOR BY V( 2)   MEASURE W( 2) = 35.34%
===========================================================================
    CONSTRUCTING A DENDRITE FOR THE EUCLIDEAN DISTANCE MATRIX
              ORIGINAL VARIABLES SPACE IS CONSIDERED
.D(  1,   3) =  0.307083
..D(  3,   6) =  0.955405
...D(  6,  26) =  0.815476
....D( 26,  17) =  0.604401
.....D( 17,   2) =  0.502295
.....D( 17,  28) =  0.564624
......D( 28,   8) =  0.257682
......D( 28,  27) =  0.478748
.......D( 27,   9) =   1.60577
..D(  3,  14) =  0.397367
.D(  1,  13) =  0.516817
..D( 13,   5) =  0.386523
...D(  5,  24) =  0.672532
...D(  5,  29) =  0.189209
..D( 13,  15) =  0.192614
...D( 15,  25) =  0.389871
....D( 25,  10) =  0.394462
.....D( 10,  12) =  0.379341
......D( 12,  22) =  0.265895
.......D( 22,  11) =  0.210000
........D( 11,  18) =  0.695916
.........D( 18,   4) =  0.614980
..........D(  4,  21) =  0.605806
...........D( 21,   7) =  0.513614
.........D( 18,  20) =  0.814125
.........D( 18,  23) =   1.20764
..D( 13,  16) =  0.125698
.D(  1,  19) =  0.979388
  MEAN                    =          0.55868870
  STD. DEV=               =          0.32723257
  MEAN + 2*(STD.DEV)      =          1.21315384
    CONSTRUCTING A DENDRITE FOR THE EUCLIDEAN DISTANCE MATRIX
    DISTANCE MATRIX IN A COORDINATE SYSTEM WITH AXES  V( 1)-V( 2) IS CONSIDERED
.D(  1,   3) =  0.247247
..D(  3,  14) =  0.345454
.D(  1,  16) =  0.476347
..D( 16,   5) =  0.302669
...D(  5,  24) =  0.656724
....D( 24,   7) =  0.623267
.....D(  7,  21) =  0.253249
......D( 21,   4) =  0.345730
.......D(  4,  18) =  0.446883
........D( 18,  20) =  0.812342
........D( 18,  23) =   1.20435
...D(  5,  29) =  0.179036
..D( 16,  13) =  0.750947E-1
```

```
...D( 13,  15) =  0.170629
....D( 15,  25) =  0.387897
.....D( 25,  12) =  0.286905
......D( 12,  10) =  0.339548
......D( 12,  22) =  0.131862
.......D( 22,  11) =  0.145009
.D(  1,  19) =  0.786190
..D( 19,  26) =  0.701248
...D( 26,   6) =  0.804311
...D( 26,  17) =  0.588328
....D( 17,   2) =  0.212862
.....D(  2,   8) =  0.345401
......D(  8,  28) =  0.208865
.......D( 28,  27) =  0.384622
........D( 27,   9) =   1.54099
  MEAN                  =        0.46439481
  STD. DEV=             =        0.33063257
  MEAN + 2*(STD.DEV)    =        1.12565994

           SCATTERED PLOT IN V( 1)-V( 2) COORDINATE SYSTEM
      +....+....+....+....+....+....+....+....+....+....+....+....+.
 0.600+                                                           +
      .                                                           .
      .                                                           .
      .                                                           .
      .                                                           .
 0.400+                                                           +
      .                                                           .
      .                                                           .
      .                    1                                      .     6
      .                                                           .
 0.200+                                                           +
      .                                                           .
      .                                           1               .    20
      .                           1    1                          .    14   10
      .                                                           .
 0.000+....+....+....+....+....+....+....+....+....+....+....+....++
      .                                                           .
      .                                  1                        .    22
      .                         1         1                       .     3   11
      .                                 1                         .    12
-0.200+         1                                                 +    27
      .                                                           .
      .                                1                          .    25
      .                                                           .
      .                         1   1                             .     1   15
-0.400+                                                           +
      .                            1                              .    13
      .                  1                    1                   .    26   18
      .          1   1             1                              .    28   17   16
      .             1                                             .     2
-0.600+                                                           +
      .                                                           .
      .                              11                           .     5   29
      .           1                                               .     8
      .                                                           .
-0.800+                                     1                     +     4
      .1                                                          .     9
      .                                                           .
      .                                                           .
      .                      1                                    .    19
-1.000+                                                           +
      .                                              1            .    23
      .                                    1                      .    21
      .                                                           .
      .                                                           .
-1.200+                                  1                        +     7
      .                                                           .
      .                                                           .
      .                              1                            .    24
      .                                                           .
-1.400+....+....+....+....+....+....+....+....+....+....+....+....+.
   -15.000   -13.500   -12.000   -10.500    -9.000    -7.500    -6.000
        -14.250   -12.750   -11.250    -9.750    -8.250    -6.750
```

POINTS WERE PLOTTED IN THE FOLLOWING ORDER :

NO.	X(I)	Y(I)	OBJECT NO.
1	-0.1174E+02	0.2682E+00	6
2	-0.8293E+01	0.1062E+00	20
3	-0.1062E+02	0.9133E-01	14
4	-0.9928E+01	0.8390E-01	10
5	-0.9587E+01	-0.8175E-01	22
6	-0.1089E+02	-0.1311E+00	3
7	-0.9446E+01	-0.1117E+00	11
8	-0.9693E+01	-0.1608E+00	12
9	-0.1338E+02	-0.2072E+00	27
10	-0.9952E+01	-0.2838E+00	25
11	-0.1096E+02	-0.3684E+00	1
12	-0.1033E+02	-0.3558E+00	15
13	-0.1047E+02	-0.4561E+00	13
14	-0.1199E+02	-0.4960E+00	26
15	-0.8857E+01	-0.4782E+00	18
16	-0.1313E+02	-0.5009E+00	28
17	-0.1257E+02	-0.5034E+00	17
18	-0.1051E+02	-0.5231E+00	16
19	-0.1278E+02	-0.5712E+00	2
20	-0.1024E+02	-0.6715E+00	5
21	-0.1006E+02	-0.6743E+00	29
22	-0.1309E+02	-0.7066E+00	8
23	-0.9154E+01	-0.8119E+00	4
24	-0.1479E+02	-0.8358E+00	9
25	-0.1147E+02	-0.9665E+00	19
26	-0.7796E+01	-0.1049E+01	23
27	-0.9370E+01	-0.1082E+01	21
28	-0.9597E+01	-0.1194E+01	7
29	-0.1021E+02	-0.1327E+01	24

```
==========================================================================
NUMBER OF VARIABLES =    6                              NUMBER OF CASES= 29
<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
                 CORRELATION MATRIX WAS TAKEN INTO ACCOUNT
<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
```

Now the same analysis as for the variance–covariance matrix will be done.

REFERENCES

Anderson, T. W., 1958, An introduction to multivariate statistical analysis: John Wiley & Sons, New York, 675 p.

Anderson, T. W., 1984, An introduction to multivariate statistical analysis (2nd ed.): John Wiley & Sons, New York, 675 p.

Jackson, J. E., and Hearne, F. T., 1973, Relationships among coefficients of vectors used in principal components: Technometrics v. 15, no. 3, p. 601–610.

Kendall, M. G., 1938, The geographical distribution of crop productivity in England: Roy. Statist. Soc., v. 102, p. 21–48.

Mardia, K. V. Kent, J. T., and Bibby, J. M., 1979, Multivariate analysis: Academic Press, London, 521 p.

APPENDIX

FORTRAN Program Listing

```
      PROGRAM PCA
**********************************************************************
*                PRINCIPAL COMPONENTS ANALYSIS PROGRAM               *
*                      LANGUAGE: FORTRAN77                           *
*                              BY                                    *
*     ANDRZEJ MACKIEWICZ            WALDEMAR RATAJCZAK               *
*     TECHNICAL UNIVERSITY          ADAM MICKIEWICZ UNIVERSITY       *
*                        POZNAN, POLAND                              *
*                                                                    *
*     ............FLOPPY DISK DELIVERED BY REQUEST.............      *
**********************************************************************
*     MAXIMUM NUMBER OF VARIABLES IS SET AT 50;                      *
*     MAXIMUM NUMBER OF CASES IS SET AT    200;                      *
```

```
*     FOR CHANGING SIZES REPLACE 50 AND 200 IN THE PARAMETER          *
*     STATEMENT THREE LINES BELOW.                                    *
      INTEGER I,J,J1,K,L,LA,LB,N,M,NM,MM,LL,IERR,PRMODE
      PARAMETER (NM=50,MM=200)
*     ------------  ^ ---- ^  ------ CHANGED ?.
      PARAMETER (PRMODE=1)
*     ----------------- ^ ----- CONVERT THIS PARAMETER TO 0 IF YOUR
*                      PRINTER CAN PLOT ONLY 133 CHARACTERS PER LINE
*                      (STANDARD PRMODE VALUE IS 1)
      REAL               RINFM,RXXXM,RINF
      DOUBLE PRECISION   XINF,XMIN,XXXF,XXXN
      PARAMETER (RXXXM= 3.35E+38, XXXF= 1.7D+308, XXXN= 2.3D-308)
*     -------------------- ^ -------------- ^ ----------- ^ -CHANGED ?.
*     VAX CONSTANTS:     1.7E+38         1.7D+38         5.9D-39
*
*     THESE PARAMETERS ARE MACHINE-DEPENDENT (HERE THEY HAVE BEEN
*     SET FOR THE IBM PC OR ANY OTHER MACHINE EQUIPPED WITH THE
*     INTEL 8087, 80287, 80387 OR COMPATIBLE. SUGGESTIONS FOR THE VAX
*     ARE PRESENTED TWO LINES BELOW THE LAST PARAMETER STATEMENT).
*     RXXXM - IS THE LARGEST VALUE POSITIVE (NORMALIZED) REAL DATA,
*     XXXF  - IS THE LARGEST VALUE POSITIVE (NORMALIZED) DOUBLE
*             PRECISION DATA,
*     XXXN  - IS THE SMALLEST VALUE POSITIVE (NORMALIZED) DOUBLE
*             PRECISION DATA,
*     TO ALTER THESE PARAMETERS FOR A PARTICULAR ENVIRONMENT
*     CHECK THE MANUAL OF IT OR SEE THE FOLLOWING PAPER:
*     FOX P.A., HALL A.D., SCHRYER N.L.,*FRAMEWORK FOR A
*     PORTABLE LIBRARY*, ACM TRANSACTIONS ON MATHEMATICAL
*     SOFTWARE, VOL. 4, NO. 2, JUNE 1978, PP. 177-188.
*     ---------------------------------------------------------------
      INTEGER            BB(MM),PROC(NM),H1(MM),R1(MM),ISORT(MM),NSTEPY
      DOUBLE PRECISION   A(NM,NM),D(NM),E(NM),X(NM),S(NM),AC(NM)
      DOUBLE PRECISION   AS1,TRACE,BX,DX,R,SX1,LCO(NM)
      DOUBLE PRECISION   U(MM),V(MM),RCO(NM)
      REAL               TQUANT,Y(MM,MM),WDIS(MM*(MM-1)/2),DD(MM),P(4)
      LOGICAL            ST
      CHARACTER          INSET*60,OUTSET*60,STR*9,YON*1
      EQUIVALENCE        (Y,WDIS),(DD,X),(E,U),(S,V)
      COMMON             RINFM,XINF,XMIN
      RINFM= -RXXXM
      XINF= XXXF
      XMIN= XXXN
      WRITE(*,'(12X,''PRINCIPAL COMPONENTS ANALYSIS-INTERACTION PROGRAM
     & ''//)')
      WRITE(*,'(12X,''*************************************************
     & '',////)')
      WRITE(*,'(''   *******  GIVE THE PATH NAME OF THE INPUT FILE - AT
     &MOST 60 CHARACTERS *******''/)')
      READ(*,5) INSET
      WRITE(*,'(''   ******* GIVE THE NAME OF THE OUTPUT FILE - AT MOST
     &60 CHARACTERS  *******''/)')
      READ(*,5) OUTSET
    5 FORMAT (A)
      OPEN(2,STATUS='UNKNOWN',FILE=OUTSET)
      WRITE(2,'(22X, ''**********************************'')')
```

```
      WRITE(2,'(22X, ''* PRINCIPAL COMPONENTS ANALYSIS *'')')
      WRITE(2,'(22X, ''*********************************''/)')
      WRITE(2,'(/, ''* '', ''  FOR DATA FILE  '',A60,''* '')') INSET
      ST=.FALSE.
10    IF (ST) THEN
         OPEN(1,STATUS='OLD',FILE=INSET)
         ELSE
         OPEN(1,STATUS='UNKNOWN',FILE=INSET)
      ENDIF
      READ(1,*) N
      IF (N.GT.NM) THEN
         WRITE(2,'(/9X,'' MAXIMUM NUMBER OF VARIABLES IS EQUAL TO NM''
     &/'' ARRAY SIZE IN THE CALLING PROGRAM MUST BE CHANGED FOR YOUR''/
     &    ,21X,''DATA BEFORE USE ''/)')
         STOP
      ENDIF
      READ(1,*) M
      IF (M.GT.MM) THEN
         WRITE(2,'(/9X,'' MAXIMUM NUMBER OF CASES IS EQUAL TO MM''
     &/'' ARRAYS SIZE IN THE CALLING PROGRAM MUST BE CHANGED FOR YOUR''/
     &    ,21X,''DATA BEFORE USE ''/)')
         STOP
      ENDIF
      IF(M.LT.3) THEN
         WRITE(2,'(/15X,''TOO FEW CASES. MINIMUM NUMBER OF CASES IS EQUA
     &L TO 3''/)')
         STOP
      ENDIF
      WRITE (2,'(//)')
      WRITE(2,'('' NUMBER OF VARIABLES = '',I3,33X,''NUMBER OF CASES='',
     &I3)') N,M
      IF (M.LE.300) THEN
         NSTEPY=50
         ELSE IF (M.LE.600) THEN
         NSTEPY=100
         ELSE
         NSTEPY=200
      ENDIF
      WRITE(2,*)
      WRITE(2,*) ('<>',I=1,39)
      IF(ST) THEN
      WRITE(2,'(/19X,''CORRELATION MATRIX WAS TAKEN INTO ACCOUNT''/)')
      ELSE
      WRITE(2,'(/17X,''VAR.-COVARIANCE MATRIX WAS TAKEN INTO ACCOUNT''/
     &)')
      ENDIF
      WRITE(2,*) ('<>',I=1,39)
*-----------------------------------------------
*     DETERMINING THE VARIANCE-COVARIANCE MATRICES
*-----------------------------------------------
      CALL COVM (NM,M,N,S,A,D,E)
      WRITE(2,*)
      WRITE(2,'(17X,''DESCRIPTIVE PARAMETERS OF ORIGINAL '',
     &''VARIABLES ''/)')
      WRITE(2,'(4X,''VARIABLE'',8X,''MEAN'',13X,''VARIANCE'',7X,
```

```
   &''STAND. DEV.''/)')
    DO 20 I=1,N
    WRITE(2,'(4X,''X('',I3,'')'',3X,E16.6,3X,2E16.7)') I,S(I),A(I,I),
   & DSQRT(A(I,I))
20  AC(I)=A(I,I)
*-----------------------------------------------------
*     OPTIONAL DETERMINATION OF THE CORRELATION MATRIX
*-----------------------------------------------------
      IF (ST) THEN
         DO 40 I=2,N
            DO 30 J=1,I-1
               A(I,J)=A(I,J)/DSQRT(A(I,I)*A(J,J))
 30         CONTINUE
 40      CONTINUE
         DO 50 I=1,N
            A(I,I)=1
 50      CONTINUE
          WRITE(2,'(//16X,'' #####  CORRELATION MATRIX  #####''/)')
          ELSE
          WRITE(2,'(//16X,'' #####  VAR.-COVARIANCE MATRIX  #####'')')
      ENDIF
*-----------------------------------------------------------------
*       PRINTING THE CORRELATION MATRIX AND THE VAR.-COVAR MATRIX
*-----------------------------------------------------------------
      DO 70 I=1,N
        DO 60 J=1,I
           A(J,I)=A(I,J)
 60     CONTINUE
 70   CONTINUE
      K=N
      DO 110 L=0,K-1,5
         WRITE(2,*)
         J1=MIN0(L+5,K)
         WRITE(2,120)(I,I=L+1,J1)
         WRITE(2,*)
         DO 100 J=1,N
            WRITE(2,130) J,(A(J,I),I=L+1,J1)
 100     CONTINUE
 110  CONTINUE
 120  FORMAT (5X,5(3X:,'COL(',I2,')',5X))
 130  FORMAT (I3,5(1X:,D14.6))
*----------------------------------------------
*     DETERMINING  EIGENVECTORS AND EIGENVALUES
*----------------------------------------------
      CALL TRED2(NM,N,A,D,E,A)
      CALL TQL2(NM,N,D,E,A,IERR)
      IF (IERR.GT.0) THEN
         WRITE(2,*) ' QL ALGORITHM FAILS - FUTURE PROGRESS IMPOSSIBLE'
         STOP
      ENDIF
*-----------------------------------------------------
*      TRACE OF THE MATRIX CONSIDERED IS NOW DETERMINED
*-----------------------------------------------------
      TRACE=0
      DO 140 I=1,N
```

```
        TRACE=TRACE+DMAX1(0D0,D(I))
  140 CONTINUE
*------------------
*     BARTLET'S TEST
*------------------
      IF (.NOT.ST) THEN
        CALL BARTLE (N,M,D,TRACE,PROC,LCO,RCO)
        ELSE
        CALL BART2(N,M,D)
      ENDIF
*---------------------------
*     PRINTING OF EIGENVALUES
*---------------------------
      WRITE(2,'(/10X,'' *********  NON-ZERO EIGENVALUES  *********''/)')
      WRITE(2,'(5X,'' NR.  '',15X,'' LAMBDA'',8X,
     & ''PERCENTAGE OF TRACE''/)')
      DO 150 I=1,N
         L=N-I+1
         IF (D(L).LE.0D0) THEN
             GOTO 170
             ELSE
             WRITE(2,160) I,D(L),SNGL(100D0*D(L)/TRACE)
          ENDIF
  150 CONTINUE
  160 FORMAT(5X,I3,10X,E22.16,7X,F6.2)
*-------------------------------------
*     PRINTING EIGENVALUES ON THE SCREEN
*-------------------------------------
  170 IF(ST) THEN
      WRITE(*,'(/10X,''+++  CORRELATION MATRIX IS CONSIDERED  +++ ''/)')
      ELSE
      WRITE(*,'(/10X,''-  VAR.-COVARIANCE  MATRIX IS CONSIDERED  -''/)')
      ENDIF
      WRITE(*,'(/10X,'' ******* FIRST NO-ZERO EIGENVALUES  *******''/)')
      WRITE(*,'(5X,'' NR.  '',15X,'' LAMBDA'',8X,
     & ''PERCENTAGE OF TRACE''/)')
      DO 180 I=1,AMINO(10,N)
         L=N-I+1
         IF (D(L).LE.0D0) THEN
             GOTO 190
             ELSE
             WRITE(*,160) I,D(L),SNGL(100D0*D(L)/TRACE)
          ENDIF
 180  CONTINUE
 190  WRITE(*,'(//1X,'' GIVE THE NUMBER OF THE PRINCIPAL COMPONENTS YOU
     &WANT TO CONSIDER ''/)')
      READ(*,*) K
*---------------------
*    END OF MONITORING
*---------------------
      IF(K.GT.0) THEN
         IF(ST) THEN
            WRITE(2,*) ('.',I=1,78)
            WRITE(2,'(/5X, '' SELECTED EIGENVECTORS (FROM '',I1,
     &      '' TO'',I3, '' ) OF THE CORRELATION MATRIX'')') 1,K
```

```
          ELSE
          WRITE(2,'(/5X, '' SELECTED EIGENVECTORS (FROM '',I1,
     &    '' TO '',I3, '' ) OF THE VAR.-COVARIANCE MATRIX'')') 1,K
        ENDIF
*--------------------------------
*       PRINTING THE EIGENVECTORS
*--------------------------------
        DO 230 L=0,K-1,5
          WRITE(2,*)
          J1=MIN0(L+5,K)
          WRITE(2,240) (I,I=L+1,J1)
          WRITE(2,*)
          DO 220 J=1,N
            WRITE(2,250) J,(A(J,N-I+1),I=L+1,J1)
 220      CONTINUE
 230    CONTINUE
      ENDIF
 240  FORMAT (5(10X:,'A(',I2,')'))
 250  FORMAT (I3,5(1X:,F14.10))
      WRITE(2,'(//)')
      WRITE(2,'('' ****************** VALUES OF THE PRINCIPAL COMPONENTS
     & ******************''/)')
      DO 280 I=1,K
        CLOSE(1,STATUS='KEEP')
        OPEN(1,STATUS='OLD',FILE=INSET)
        READ(1,*) N
        READ(1,*) M
        WRITE(2,*)
        WRITE(2,'(12X,'' V('',I2,'')''/)') I
        DO 270 J=1,M
          READ(1,*) (X(LL),LL=1,N)
          BX=0D0
          DO 260 LA=1,N
            BX=BX+X(LA)*A(LA,N-I+1)
 260      CONTINUE
          WRITE(2,'(8X,I3,3X,D23.16)') J,BX
 270    CONTINUE
 280  CONTINUE
*------------------------------------------------------------------
*     DETERMINATION AND CORRELATION COEFFICIENTS ARE NOW CALCULATED
*------------------------------------------------------------------
      WRITE(2,*)
      WRITE(2,*) ('.',I=1,72)
      WRITE(2,*) '   COEFFICIENTS OF DETERMINATION AND CORRELATION BETWE
     &EN COMPONENTS V(I) '
      WRITE(2,'(23X, '' AND ORIGINAL VARIABLES X(J)'')')
      WRITE(2,*)
      J1=M-2
      P(1)=0.1
      P(2)=0.05
      P(3)=0.01
      P(4)=0.001
      DO 320 I=1,K
      WRITE(2,*)
      WRITE(2,'(''  BETWEEN V('',I3,'') AND ORIGINAL VARIABLES FROM X(1)
```

```
 & TO X('',I3,'')'')') I,N
  WRITE(2,*)
  AS1=0
     DO 310 J=1,N
        R=A(J,N-I+1)* A(J,N-I+1) *D(N-I+1)
        IF (.NOT.ST)  R=R/AC(J)
        AS1=AS1+R
        DX=DSIGN(DSQRT(R),A(J,N-I+1))
        DO 290 LA=1,4
           CALL TQ(J1,P(LA),TQUANT)
           BX=TQUANT*TQUANT
           TQUANT=SQRT(BX/(BX+J1))
           IF (DABS(DX).LE.TQUANT) THEN
              L=LA-1
              GOTO 300
           ENDIF
 290       CONTINUE
           L=5
 300       IF (L.EQ.0) THEN
               STR=' >  0.1 '
               ELSEIF(L.EQ.1) THEN
               STR=' =  .1  '
               ELSEIF(L.EQ.2) THEN
               STR=' = 0.05 '
               ELSEIF(L.EQ.3) THEN
               STR=' = 0.01 '
               ELSEIF(L.EQ.4) THEN
               STR=' = 0.001'
               ELSE
               STR=' < 0.001'
           ENDIF
           WRITE(2,'(''X('',I3,'')   R^2='',F11.8,''  R= '',F14.8, ''  S
    &IGNIFICANT FOR  ALPHA '',1X,A9)') J,R,DX,STR
 310     CONTINUE
      WRITE(2,*)
      WRITE(2,'('' TOTAL PERCENTAGE OF VARIABILITY OF ALL ORIGINAL VARIA
    &BLES'')')
      WRITE(2,'(''     ACCOUNTED FOR BY V('',I3,'')   MEASURE W('',I3,''
    &) ='',F6.2, ''%'')') I,I,AS1*100.0/DBLE(N)
 320  CONTINUE
*-----------------------------------------------------------------
*  EUCLIDEAN DISTANCES (IN ORIGINAL VARIABLE SPACE) ARE NOW DETERMINED
*-----------------------------------------------------------------
      WRITE(2,*)
      WRITE(2,*) ('=',I=1,78)
      IF(.NOT.ST) THEN
         CLOSE(1,STATUS='KEEP')
         OPEN(1,STATUS='OLD',FILE=INSET)
         READ(1,*) N
         READ(1,*) M
         DO 330 J=1,M
            READ(1,*) (Y(LL,J),LL=1,N)
 330     CONTINUE
         LL=1
         DO 370 I=1,M-1
```

```
            DO 340 K=1,N
               X(K)=Y(K,I)
 340        CONTINUE
            DO 360 J=I+1,M
               SX1=0.0
               DO 350 K=1,N
                 SX1=SX1+(X(K)-Y(K,J))**2
 350           CONTINUE
               Y(J,I)=SQRT(SX1)
 360        CONTINUE
            Y(I,I)=0.0
 370     CONTINUE
         DO 390 I=2,M
            DO 380 J=1,I-1
              Y(J,I)=Y(I,J)
 380        CONTINUE
 390     CONTINUE
         DO 410 J=2,M
           DO 400 I=1,J-1
              WDIS(LL)=Y(I,J)
              LL=LL+1
 400       CONTINUE
 410     CONTINUE
         CLOSE(1,STATUS='KEEP')
         WRITE(2,'(/5X,''CONSTRUCTING A DENDRITE FOR THE EUCLIDEAN DISTA
     &NCE MATRIX'')')
         WRITE(2,'(/15X,''ORIGINAL VARIABLES SPACE IS CONSIDERED'',/)')
         RINF=RXXXM*1E-1
         CALL DENDRI(M,WDIS,RINF,1,1,BB,DD,H1,R1,LL,PRMODE)
         CLOSE(1,STATUS='KEEP')
      ENDIF
*---------------------------------------------------------------
*     DISTANCE MATRIX IN COORDINATE SYSTEM V(1)-V(2) IS NOW COMPUTED
*---------------------------------------------------------------
      CLOSE(1,STATUS='KEEP')
 420  CLOSE(2,STATUS='KEEP')
      WRITE(*,*)'EXAMINE YOUR OUTPUT FILE IN ORDER TO CHOOSE A NEW COORD
     &INATE SYSTEM'
      WRITE (*,*)'THEN PRESS <ENTER>'
      WRITE(*,*) 'IT IS USUALLY POSSIBLE TO EXECUTE ANY OS COMMAND NOW'
      PAUSE
 415  WRITE(*,*)'TYPE CAPITAL Y IF YOU WANT TO CONSIDER A NEW PRINCIPAL
     & COMPONENT '
      WRITE(*,*) 'COORDINATE SYSTEM OR TYPE CAPITAL N          '
      READ(*,425) YON
      IF(YON.NE.'Y' .AND. YON.NE.'N') GOTO 415
 425  FORMAT(A1)
      OPEN(2,STATUS='OLD',ACCESS='APPEND',FILE=OUTSET)
      IF(YON.EQ.'N') GOTO 490
 430  WRITE(*,*) 'GIVE THE NUMBER OF THE FIRST COORDINATE AXIS '
      READ(*,*) LA
      WRITE(*,*) 'GIVE THE NUMBER OF THE SECOND COORDINATE AXIS '
      READ(*,*) LB
      IF(LA.GT.N.OR.LB.GT.N) THEN
         WRITE (*,*) 'DATA ERROR - TRY AGAIN'
```

```
         GO TO 430
      ENDIF
      OPEN(1,STATUS='OLD',FILE=INSET)
      READ(1,*) N
      READ(1,*) M
      DO 440 J=1,M
         READ(1,*) (Y(J,LL),LL=1,N)
 440  CONTINUE
      CLOSE(1,STATUS='KEEP')
*--------------------
*     DATA FOR PLOTTING
*--------------------
      LA=N+1-LA
      LB=N+1-LB
      DO 460 J=1,M
         BX=0D0
         DX=0D0
         DO 450 K=1,N
            BX=BX+DBLE(Y(J,K))*A(K,LA)
            DX=DX+DBLE(Y(J,K))*A(K,LB)
 450     CONTINUE
         V(J)=DX
         U(J)=BX
 460  CONTINUE
      LL=1
      DO 480 I=2,M
        DO 470 J=1,I-1
           WDIS(LL)=SNGL(DSQRT((U(J)-U(I))**2+(V(J)-V(I))**2))
           LL=LL+1
 470    CONTINUE
 480  CONTINUE
      WRITE(2,'(/5X,''CONSTRUCTING A DENDRITE FOR THE EUCLIDEAN DISTANCE
     & MATRIX'')')
      WRITE(2,'(/5X,''DISTANCE MATRIX IN A COORDINATE SYSTEM WITH AXES
     &V('',I2,'')-V('',I2,'') IS CONSIDERED'',/)') N+1-LA,N+1-LB
      RINF=RXXXM*1E-1
      CALL DENDRI(M,WDIS,RINF,1,1,BB,DD,H1,R1,LL,PRMODE)
      WRITE(2,*)
      WRITE(2,'(/17X,'' SCATTERED PLOT IN V('',I2,'')-V('',I2,'') COORDI
     &NATE SYSTEM''/)') N+1-LA,N+1-LB
      CALL PLOTDR (U,V,M,ISORT,60,NSTEPY,PRMODE)
      WRITE(2,'(//)')
      WRITE(2,'(14X,''POINTS WERE PLOTTED IN THE FOLLOWING ORDER :'')')
      WRITE(2,'(12X,60A1)') ('-',I=1,47)
      WRITE(2,'(12X,'' NO. *  X(I)     *    Y(I)      * OBJECT NO.'')')
      WRITE(2,'(12X,60A1)') ('-',I=1,47)
      DO 485 I=1,M
         WRITE(2,'(12X,I3,2X,E11.4,3X,E11.4,6X,I3)')
     &                     I,U(ISORT(I)),V(I),ISORT(I)
 485  CONTINUE
      WRITE(2,*)
      GOTO 420
*-----------------------------------------------------------------
*     PRINCIPAL COMPONENTS ANALYSIS FOR THE CORRELATION MATRIX OR STOP
*-----------------------------------------------------------------
```

```
 490  WRITE(2,*) ('=',I=1,78)
      IF(.NOT.ST)THEN
         ST=.TRUE.
         CLOSE(1,STATUS='KEEP')
         GOTO 10
      ENDIF
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE TQ(N,P,TQUANT)
************************************
*     STUDENT'S T-QUANTILES        *
************************************
      INTEGER N
      REAL    P,TQUANT,HP,A,B,C,D,X,Y
      IF (N.LT.1 .OR. P.LT..0 .OR. P.GT.1.0) THEN
         TQUANT=0
         RETURN
      ENDIF
      HP=1.57077963268
      IF (N.EQ.1) THEN
         X=P*HP
         TQUANT=COS(X)/SIN(X)
      ELSEIF (N.EQ.2) THEN
         TQUANT=SQRT(2.0/(P*(2.0-P))-2.0)
      ELSE
         A=1.0/(N-.5)
         B=48.0/(A*A)
         C=((20700.0*A/B-98.0)*A-16.0)*A+96.36
         D=((94.5/(B+C)-3.0)/B+1.0)*SQRT(A*HP)*N
         X=D*P
         Y=X**(2.0/N)
         IF(Y.GT..05+A) THEN
            IF(ABS(P-.1).LT.1D-7) THEN
               X=-1.644854
            ELSEIF(ABS(P-.05).LT.1D-7) THEN
               X=-1.959964
            ELSEIF(ABS(P-.01).LT.1D-7) THEN
               X=-2.575829
            ELSEIF(ABS(P-.001).LT.1D-7) THEN
               X=-3.290527
            ELSE
               X=1D30
            ENDIF
            Y=X*X
            IF(N.LT.5) C=C+.3*(N-4.5)*(X+.6)
            C=(((.05*D*X-5.0)*X-7.0)*X-2.0)*X+B+C
            Y=(((((.04*Y+6.3)*Y+36.0)*Y+94.5)/C-Y-3.0)/B+1.0)*X
            Y=A*Y*Y
            IF (Y.GT..1) THEN
               Y=EXP(Y)-1.0
            ELSE
               Y=((Y+4.0)*Y+12.0)*Y*Y/24.0+Y
            ENDIF
         ELSE
            Y=((1.0/(((N+6.0)/(N*Y)-.089*D-.822)*(N+2.0)*3.0)
```

```
     &        +.5/(N+4.0))*Y-1.0)*(N+1.0)/(N+2.0)+1.0/Y
            ENDIF
            TQUANT=SQRT(N*Y)
         ENDIF
         END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE DENDRI (N,D,INF,S,PR,B,C,H,R,ST,PRMODE)
*********************************************************
*     PLOTTING THE MINIMUM SPANNING TREE                *
*********************************************************
      INTEGER   N,PR,ST,PRMODE
      INTEGER   B(N),H(N),S1,S2,I,J,K,K1,L,M1,R(N),S,II,JJ
      REAL      C(N),D((N*(N-1))/2),DI,INF,MIN,SEL,MEAN,STDEV
      CHARACTER LINOUT*200,SHORT*133
      MEAN=0.0
      STDEV=0.0
      SEL=0D0
      IF (N.LT.1) THEN
         ST=2
         RETURN
      ENDIF
      IF (PRMODE.EQ.1) THEN
         JJ=200
         ELSE
         JJ=133
      ENDIF
      ST=0
      INF=INF+1.0E0
      DO 10 I=1,N
         H(I)=I
         C(I)=INF
         B(I)=0
   10 CONTINUE
      H(S)=N
      S1=S
      DO 30 I=N-1,1,-1
         MIN=INF+1E0
         S2=((S1-1)*(S1-2))/2
         DO 20 K=1,I
            J=H(K)
            IF (S1.GT.J) THEN
               L=S2+J
               ELSE
               L=((J-1)*(J-2))/2+S1
            ENDIF
            DI=D(L)
            IF (DI.LT.C(J)) THEN
               C(J)=DI
               B(J)=S1
               ELSE
               DI=C(J)
            ENDIF
            IF (DI.LT.MIN) THEN
               MIN=DI
               K1=K
```

```
          ENDIF
 20    CONTINUE
       S1=H(K1)
       H(K1)=H(I)
 30 CONTINUE
    DO 40 I=1,N
       IF (B(I).EQ.0) GOTO 40
       IF (C(I).LT.INF-1E0) THEN
          SEL=SEL+C(I)
          ELSE
          SEL=INF-1E0
          ST=1
       ENDIF
40 CONTINUE
   IF (PR.EQ.0.OR.S.NE.1) RETURN
   DO 50 I=1,N
      H(I)=0
50 CONTINUE
   DO 60 I=2,N
      J=B(I)
      H(J)=H(J)+1
60 CONTINUE
   R(1)=1
   J=1
   K=1
   DO 100 I=2,N
70    IF (H(K).EQ.0) THEN
         J=J-1
         K=R(J)
         GOTO 70
      ELSE
         H(K)=H(K)-1
         DO 80 M1=2,N
            IF (K.EQ.B(M1)) GOTO 90
80       CONTINUE
90       IF (PRMODE.EQ.1) THEN
             WRITE(LINOUT(1:),'(200A)') ('.',II=1,200)
             IF (J.LE.JJ-25) THEN
                WRITE(LINOUT(J+1:),'(''D('',I4,'','',I4,'') = '',
  &             G12.6E1)') K,M1,C(M1)
             ELSE
                WRITE(LINOUT((JJ-25):),'(A)')' SORRY, LINE TOO LONG '
             ENDIF
             WRITE(2,'(200A)') LINOUT
         ELSE
             WRITE(SHORT(1:),'(133A)') ('.',II=1,133)
             IF (J.LE.JJ-25 ) THEN
                WRITE(SHORT(J+1:),'(''D('',I4,'','',I4,'') = '',
  &             G12.6E1)') K,M1,C(M1)
             ELSE
                WRITE(SHORT((JJ-25):),'(A)') ' SORRY, LINE TOO LONG '
             ENDIF
             WRITE(2,'(133A)') SHORT
         ENDIF
         MEAN=MEAN+C(M1)
```

```
          J=J+1
          K=M1
          R(J)=K
          B(M1)=-B(M1)
        ENDIF
  100 CONTINUE
      DO 110 I=2,N
        B(I)=IABS(B(I))
  110 CONTINUE
      MEAN=MEAN/FLOAT(N-1)
      DO 120 I=2,N
        SEL=(C(I)-MEAN)
        STDEV=STDEV+SEL*SEL
  120 CONTINUE
      STDEV=SQRT(STDEV/FLOAT(N-1))
      WRITE(2,*)
      WRITE(2,*)' MEAN                   =  ', MEAN
      WRITE(2,*)' STD. DEV=              =  ', STDEV
      WRITE(2,*)' MEAN + 2*(STD.DEV)     =  ', MEAN+2*STDEV
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE COVM (MN,M,N,S,A,D,E)
*****************************************************************
*     CALCULATION OF MEANS AND THE VARIANCE-COVARIANCE MATRIX   *
*****************************************************************
      INTEGER             MN,M,N,I,J,N1,I1,J1
      DOUBLE PRECISION    DR,DIJ,DIV
      DOUBLE PRECISION    D(N),E(N),S(N),A(MN,N)
      DO 20 I=1,N
        D(I)=0D0
        S(I)=0D0
        DO 10 J=1,N
          A(I,J)=0D0
 10     CONTINUE
 20   CONTINUE
 30   N1=0
      DO 60 I=1,M
        READ(1,*) (E(J),J=1,N)
        N1=N1+1
        DIV=1D0/DBLE(N1)
        DO 50 I1=1,N
          DR=E(I1)-S(I1)
          D(I1)=DR
          S(I1)=S(I1)+DR*DIV
          DO 40 J1=1,I1
            DIJ=DR*D(J1)
            A(I1,J1)=A(I1,J1)+DIJ-DIJ*DIV
 40       CONTINUE
 50     CONTINUE
 60   CONTINUE
      DO 80 I=1,N
        DO 70 J=1,I
          A(I,J)=A(I,J)/DBLE(M)
 70     CONTINUE
 80   CONTINUE
      END
```

```
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE BART2(N,M,D)
**************************************************
*     BARTLET'S TEST FOR THE COVARIANCE MATRIX    *
**************************************************
      INTEGER          N,M,I,IER,J,K1,P
      REAL             CHIPRO
      DOUBLE PRECISION AO,AOO,D(N)
      LOGICAL          LOGI
      LOGI=.FALSE.
      P=0
      DO 10 I=N,1,-1
         IF (D(I).GT.1D-13) P=P+1
 10   CONTINUE
      J=MINO(N-2,P)
      DO 30 K1=0,J
         AO=0DO
         AOO=1DO
         DO 20 I=K1+1,N
            AO=AO+D(N-I+1)
            AOO=AOO*DABS(D(N-I+1))
 20      CONTINUE
         AO=AO/(N-K1)
         AOO=AOO**(1.DO/(N-K1))
         IF(AO/AOO.GE.1D30) THEN
            LOGI=.TRUE.
            GOTO 40
         ENDIF
         CALL CDTR(SNGL((M-1)*(P-K1)*DLOG(AO/AOO)),
     &             REAL((P-K1+2)*(P-K1-1)*0.5),CHIPRO,IER)
         IF(1.0-CHIPRO.GT.0.05.OR.IER.NE.0) GOTO 40
 30   CONTINUE
 40   WRITE(2,*)
      WRITE(2,*) ('*',I=1,80)
      WRITE(2,*)
      WRITE(2,'(20X,''TESTING THE PRINCIPAL COMPONENTS '')')
      WRITE(2,'(25X,''(FOR CORRELATION MATRIX)'')')
      WRITE(2,*)
      IF(IER.NE.0.OR.LOGI) THEN
         K1=N
         WRITE (2,*) 'BARTLET''S TEST CAN NOT BE USED HERE '
      ENDIF
      WRITE(2,*)
      IF (K1.LT.N) THEN
         WRITE(2,*) ' BARTLET''S TEST REJECTS EIGENVECTORS NO. '
         DO 50 I=K1+1,N
            WRITE(2,'(5X,I3)') I
 50      CONTINUE
      ENDIF
      WRITE(2,*)
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE BARTLE(N,M,D,TRACE,PROC,LCO,RCO)
**************************************************
*                BARTLET'S TEST                  *
**************************************************
```

```
      INTEGER          N,M,K,I,IER,J,L,K1,PROC(N)
      REAL             CHIPRO
      DOUBLE PRECISION A0,A00,DELTA,TR1,TR2,TR3,TRACE,D(N),LCO(N),RCO(N)
      LOGICAL LOGI
      LOGI=.FALSE.
      DO 20 K1=0,N-2
         A0=0D0
         A00=1D0
         DO 10 I=K1+1,N
           A0=A0+DABS(D(N-I+1))
           A00=A00*DABS(D(N-I+1))
 10      CONTINUE
         A0=A0/(N-K1)
         A00=A00**(1.D0/(N-K1))
         IF(A0/A00.GE.1D30) THEN
            LOGI=.TRUE.
            GOTO 30
         ENDIF
         CALL CDTR(SNGL((M-(2*N+11)/6D0)*(N-K1)*DLOG(A0/A00)),
     &         REAL((N-K1+2)*(N-K1-1)*0.5),CHIPRO,IER)
         IF (IER.NE.0) GOTO 30
         IF(1.0-CHIPRO.GT.0.05) GOTO 30
 20   CONTINUE
 30   IF(LOGI.OR.IER.NE.0) THEN
         K1=N
      ENDIF
*
*   TESTING PRINCIPAL COMPONENTS
*
      DO 40 I=K1+1,N
         PROC(N-I+1)=0
         LCO(N-I+1)=0D0
         RCO(N-I+1)=0D0
 40   CONTINUE
      DO 80 K=0,K1-1
         TR1=0
         TR2=0
         TR3=0
         DO 50 I=1,K
           TR1=TR1+D(N-I+1)**2
 50      CONTINUE
         DO 60 I=K+1,N
            TR2=TR2+D(N-I+1)**2
            TR3=TR3+D(N-I+1)
 60      CONTINUE
         DO 70 L=0,100
           DELTA=0.01*DBLE(L)
        IF(DSQRT(DBLE(M))*(TR3/TRACE-DELTA)-1.96*DSQRT(DABS(2*(DELTA
     &        /TRACE))**2*TR1+2*((1-DELTA)/TRACE)**2*TR2).LT.0) THEN
           PROC(N-K)=L
           GOTO 80
        ENDIF
 70      CONTINUE
         PROC(N-K)=100
 80   CONTINUE
```

```
*
*     CONFIDENCE INTERVAL IS NOW DETERMINED
*
      TR1=0D0
      DO 90 I=1,N
         TR1=D(I)*D(I)+TR1
 90   CONTINUE
      DO 110 K=1,K1
         DELTA=0D0
         TR2=0D0
         DO 100 J=1,K
            TR2=TR2+D(N-J+1)**2
            DELTA=DELTA+D(N-J+1)
 100     CONTINUE
         TR3=DELTA/TRACE
         A00=TR2/TR1
         A0=(2D0*TR1*(TR3**2-2D0*TR3*A00+A00**2))/((M-1)*TRACE**2)
         RCO(N-K+1)=1.96*DSQRT(DABS(A0))
         LCO(N-K+1)=TR3-RCO(N-K+1)
         RCO(N-K+1)=TR3+RCO(N-K+1)
 110  CONTINUE
      WRITE(2,*)
      WRITE(2,*) ('*',I=1,80)
      WRITE(2,*)
      WRITE(2,'(20X,''TESTING THE PRINCIPAL COMPONENTS '')')
      WRITE(2,'(20X,''(FOR VARIANCE-COVARIANCE MATRIX)'')')
      WRITE(2,*)
      IF(LOGI.OR.IER.NE.0) THEN
         WRITE (2,*) ' BARTLET''S TEST CAN NOT BE USED HERE '
      ELSE
      WRITE(2,*)'___ NO.__CRITICAL DELTA__ LEFTCONFID.____ RIGHTCONFID.'
      WRITE(2,*)
      DO 120 I=N,1,-1
         WRITE(2,'(I8,2X,I5,6X,F12.2,4X,F17.2)')
     &N-I+1,PROC(I), IDNINT(LCO(I)*1000)/10D0,IDNINT(RCO(I)*1000)/10D0
 120  CONTINUE
      ENDIF
      WRITE(2,*)
      IF (K1.LT.N) THEN
         WRITE(2,*) ' BARTLET''S TEST REJECTS EIGENVECTORS NO. '
         DO 130 I=K1+1,N
            WRITE(2,'(5X,I3)') I
 130     CONTINUE
      ENDIF
      WRITE(2,*)
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE TQL2(NM,N,D,E,Z,IERR)
**************************************************************
*     THIS SUBROUTINE FINDS THE EIGENVALUES AND EIGENVECTORS   *
*     OF A SYMMETRIC TRIDIAGONAL MATRIX BY THE QL METHOD.      *
*           It was taken from the LINPACK library              *
**************************************************************
      INTEGER I,J,K,L,M,N,II,L1,L2,NM,MML,IERR
      DOUBLE PRECISION D(N),E(N),Z(NM,N)
```

```
      DOUBLE PRECISION C,C2,C3,DL1,EL1,F,G,H,P,R,S,S2,TST1,TST2,PYTHAG
      IERR = 0
      IF (N .EQ. 1) GO TO 1001
      DO 100 I = 2, N
  100 E(I-1) = E(I)
      F = 0.0D0
      TST1 = 0.0D0
      E(N) = 0.0D0
      DO 240 L = 1, N
         J = 0
         H = DABS(D(L)) + DABS(E(L))
         IF (TST1 .LT. H) TST1 = H
         DO 110 M = L, N
            TST2 = TST1 + DABS(E(M))
            IF (TST2 .EQ. TST1) GO TO 120
  110    CONTINUE
  120    IF (M .EQ. L) GO TO 220
  130    IF (J .EQ. 30) GO TO 1000
         J = J + 1
         L1 = L + 1
         L2 = L1 + 1
         G = D(L)
         P = (D(L1) - G) / (2.0D0 * E(L))
         R = PYTHAG(P,1.0D0)
         D(L) = E(L) / (P + DSIGN(R,P))
         D(L1) = E(L) * (P + DSIGN(R,P))
         DL1 = D(L1)
         H = G - D(L)
         IF (L2 .GT. N) GO TO 145
         DO 140 I = L2, N
  140       D(I) = D(I) - H
  145    F = F + H
         P = D(M)
         C = 1.0D0
         C2 = C
         EL1 = E(L1)
         S = 0.0D0
         MML = M - L
         DO 200 II = 1, MML
           C3 = C2
           C2 = C
           S2 = S
           I = M - II
           G = C * E(I)
           H = C * P
           R = PYTHAG(P,E(I))
           E(I+1) = S * R
           S = E(I) / R
           C = P / R
           P = C * D(I) - S * G
           D(I+1) = H + S * (C * G + S * D(I))
           DO 180 K = 1, N
              H = Z(K,I+1)
              Z(K,I+1) = S * Z(K,I) + C * H
              Z(K,I) = C * Z(K,I) - S * H
```

```
 180       CONTINUE
 200     CONTINUE
         P = -S * S2 * C3 * EL1 * E(L) / DL1
         E(L) = S * P
         D(L) = C * P
         TST2 = TST1 + DABS(E(L))
         IF (TST2 .GT. TST1) GO TO 130
 220     D(L) = D(L) + F
 240  CONTINUE
      DO 300 II = 2, N
         I = II - 1
         K = I
         P = D(I)
         DO 260 J = II, N
           IF (D(J) .GE. P) GO TO 260
           K = J
           P = D(J)
 260     CONTINUE
         IF (K .EQ. I) GO TO 300
         D(K) = D(I)
         D(I) = P
         DO 280 J = 1, N
            P = Z(J,I)
            Z(J,I) = Z(J,K)
            Z(J,K) = P
 280     CONTINUE
 300  CONTINUE
      GO TO 1001
1000  IERR = L
1001  RETURN
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE TRED2(NM,N,A,D,E,Z)
*************************************************************
*     THIS SUBROUTINE REDUCES A REAL SYMMETRIC MATRIX TO A    *
*     SYMMETRIC TRIDIAGONAL MATRIX USING AND ACCUMULATING     *
*     ORTHOGONAL SIMILARITY TRANSFORMATIONS.                  *
*          It was taken from the LINPACK library              *
*************************************************************
      INTEGER I,J,K,L,N,II,NM,JP1
      DOUBLE PRECISION A(NM,N),D(N),E(N),Z(NM,N)
      DOUBLE PRECISION F,G,H,HH,SCALE
      DO 100 I = 1, N
         DO 80 J = I, N
  80       Z(J,I) = A(J,I)
         D(I) = A(N,I)
 100  CONTINUE
      IF (N .EQ. 1) GO TO 510
      DO 300 II = 2, N
         I = N + 2 - II
         L = I - 1
         H = 0.0D0
         SCALE = 0.0D0
         IF (L .LT. 2) GO TO 130
         DO 120 K = 1, L
```

```
120       SCALE = SCALE + DABS(D(K))
        IF (SCALE .NE. 0.0D0) GO TO 140
130     E(I) = D(L)
        DO 135 J = 1, L
           D(J) = Z(L,J)
           Z(I,J) = 0.0D0
           Z(J,I) = 0.0D0
135     CONTINUE
        GO TO 290
140     DO 150 K = 1, L
           D(K) = D(K) / SCALE
           H = H + D(K) * D(K)
150     CONTINUE
        F = D(L)
        G = -DSIGN(DSQRT(H),F)
        E(I) = SCALE * G
        H = H - F * G
        D(L) = F - G
        DO 170 J = 1, L
170        E(J) = 0.0D0
        DO 240 J = 1, L
           F = D(J)
           Z(J,I) = F
           G = E(J) + Z(J,J) * F
           JP1 = J + 1
           IF (L .LT. JP1) GO TO 220
           DO 200 K = JP1, L
              G = G + Z(K,J) * D(K)
              E(K) = E(K) + Z(K,J) * F
200        CONTINUE
220        E(J) = G
240     CONTINUE
        F = 0.0D0
        DO 245 J = 1, L
           E(J) = E(J) / H
           F = F + E(J) * D(J)
245     CONTINUE
        HH = F / (H + H)
        DO 250 J = 1, L
250        E(J) = E(J) - HH * D(J)
        DO 280 J = 1, L
           F = D(J)
           G = E(J)
           DO 260 K = J, L
260           Z(K,J) = Z(K,J) - F * E(K) - G * D(K)
           D(J) = Z(L,J)
           Z(I,J) = 0.0D0
280     CONTINUE
290     D(I) = H
300 CONTINUE
    DO 500 I = 2, N
        L = I - 1
        Z(N,L) = Z(L,L)
        Z(L,L) = 1.0D0
        H = D(I)
```

```
        IF (H .EQ. 0.0D0) GO TO 380
        DO 330 K = 1, L
  330      D(K) = Z(K,I) / H
        DO 360 J = 1, L
        G = 0.0D0
        DO 340 K = 1, L
  340      G = G + Z(K,I) * Z(K,J)
        DO 360 K = 1, L
          Z(K,J) = Z(K,J) - G * D(K)
  360   CONTINUE
  380   DO 400 K = 1, L
  400      Z(K,I) = 0.0D0
  500 CONTINUE
  510 DO 520 I = 1, N
        D(I) = Z(N,I)
        Z(N,I) = 0.0D0
  520 CONTINUE
      Z(N,N) = 1.0D0
      E(1) = 0.0D0
      RETURN
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      DOUBLE PRECISION FUNCTION PYTHAG(A,B)
**********************************************************************
*   FINDS DSQRT(A**2+B**2) WITHOUT OVERFLOW OR DESTRUCTIVE UNDERFLOW  *
**********************************************************************
      DOUBLE PRECISION A,B
      DOUBLE PRECISION P,R,S,T,U
      P = DMAX1(DABS(A),DABS(B))
      IF (P .EQ. 0.0D0) GO TO 20
      R = (DMIN1(DABS(A),DABS(B))/P)**2
   10 CONTINUE
      T = 4.0D0 + R
      IF (T .EQ. 4.0D0) GO TO 20
      S = R/T
      U = 1.0D0 + 2.0D0*S
      P = U*P
      R = (S/U)**2 * R
      GO TO 10
   20 PYTHAG = P
      RETURN
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      SUBROUTINE PLOTDR(X,Y,NOBS,KEY,NSX,NSY,PRMODE)
************************************************************
*             PLOTTING SCATTEGRAM - DRIVER                 *
************************************************************
      INTEGER PRMODE,NOBS,NSX,NSY
      DOUBLE PRECISION X(NOBS),Y(NOBS),XAXIS(28),T
      INTEGER KEY(NOBS),IA(133),IXLINE(133)
      DOUBLE PRECISION XSTEP,YSTEP,XMN,XMX,XMIN,YMN,YMX,YMIN
      INTEGER J,MAXA,MAXYA,MAXB,MAXYB,NSX1,NSY1
      INTEGER POM,LOGNB2,I,K,L,M,NN
      LOGNB2=INT(ALOG(FLOAT(NOBS))*1.4426950+1E-5)
      DO 10 I=1,NOBS
```

```
         KEY(I)=I
   10  CONTINUE
        M=NOBS
        DO 40 NN=1,LOGNB2
          M=M/2
          K=NOBS-M
          DO 30 J=1,K
            I=J
   20       CONTINUE
            L=I+M
            IF(Y(L).GT.Y(I)) THEN
              T=Y(I)
              POM=KEY(I)
              Y(I)=Y(L)
              KEY(I)=KEY(L)
              Y(L)=T
              KEY(L)=POM
              I=I-M
              IF(I.GE.1)GO TO 20
            ENDIF
   30     CONTINUE
   40   CONTINUE
        YMX = Y(1)
        YMN = Y(NOBS)
        XMX = X(1)
        XMN = XMX
        DO 50 J=2,NOBS
           XMN = DMIN1(X(J),XMN)
           XMX = DMAX1(X(J),XMX)
   50   CONTINUE
        IF (YMN.EQ.YMX) THEN
           YMN = YMN - 1.0D0
           YMX = YMX + 1.0D0
        ENDIF
        IF (XMN.EQ.XMX) THEN
           XMN = XMN - 1.0D0
           XMX = XMX + 1.0D0
        ENDIF
        NSX1 = NSX
        NSY1 = NSY
        CALL PLOTAX(XMN,XMX,NSX1,XMIN,XSTEP,MAXA,MAXB)
        IF (XMIN.EQ.XMN) THEN
           XMIN = XMIN - XSTEP
           NSX1 = NSX1 + 1
        ENDIF
        CALL PLOTAX(YMN,YMX,NSY1,YMIN,YSTEP,MAXYA,MAXYB)
        IF (YMIN.EQ.YMN) THEN
           YMIN = YMIN - YSTEP
           NSY1 = NSY1 + 1
        ENDIF
        CALL PLOTMN(X,Y,XMIN,XSTEP,NSX1,YMIN,YSTEP,NSY1,NOBS,MAXA,
      &MAXB,KEY,MAXYA,MAXYB,IA,IXLINE,XAXIS,PRMODE)
        RETURN
        END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
```

```
      SUBROUTINE PLOTAX(ZMN,ZMX,NSTEP,ZNMIN,STEP,MAXA,MAXB)
************************************************************
*         PLOTTING SCATTEGRAM - AUXILIARY SUBROUTINE       *
************************************************************
      DOUBLE PRECISION STEP,ZMN,ZMX,ZNMIN
      INTEGER MAXA,MAXB,NSTEP
      DOUBLE PRECISION AR,RINT,RNSTPZ,TENN,ZNM,ZNMAX
      INTEGER I,II,IT,J,MINT
      DOUBLE PRECISION R(9)
      DATA R /0.1D0,0.15D0,0.2D0,0.25D0,0.4D0,0.5D0,0.6D0,0.75D0,0.8D0/
      RNSTPZ = NSTEP
      RINT = (ZMX-ZMN)/(RNSTPZ+0.1D0)
   10 MINT = DLOG10(RINT) - 2.0D0
      TENN = 10.0D0**MINT
      DO 30 J=1,11
         DO 20 I=1,9
            AR = R(I)
            IF (AR*TENN.GE.RINT) GO TO 40
   20    CONTINUE
         TENN = TENN*10.0D0
   30 CONTINUE
   40 STEP = TENN*AR
      IT = ZMN/STEP
      AR = STEP*DBLE(IT)
   50 IF ((AR-STEP*0.05D0).GT.ZMN) THEN
          AR = AR - STEP
          GO TO 50
       ENDIF
       ZNMIN = AR
       ZNMAX = ZNMIN + STEP*(RNSTPZ+0.04D0)
       IF (ZNMAX.LT.ZMX) THEN
          RINT = RINT*1.05D0
          GO TO 10
       ENDIF
       MAXA = DLOG10(DABS(ZNMAX)) + 1.0D0
       IF (DABS(ZNMIN).GT.DABS(ZNMAX)) MAXA = DLOG10(DABS(ZNMIN)) +
      & 1.0D0
       MAXA = MAX0(0,MAXA)
       MAXB = DLOG10(STEP) - 2.5D0
       MAXB = -MIN0(MAXB,0)
       IF (MAXA+MAXB.GE.10) GO TO 70
       ZNM = ZNMIN
       DO 60 II=1,10
          I = II - 1
          IF (ZNM+STEP*(RNSTPZ+0.04D0).LT.ZMX) GO TO 70
          ZNMIN = ZNM
          ZNM = (ZNMIN*10.0D0**(MAXB-I))
          IF (ZNM.LT.0.0D0) ZNM = ZNM - 1.0D0
          ZNM = DINT(ZNM)/10.0D0**(MAXB-I)
  60   CONTINUE
  70   RETURN
       END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
       SUBROUTINE PLOTMN(X,Y,XMIN,XSTEP,NSTEPX,YMIN,YSTEP,NSTEPY,
      & NOBS,MAXA,MAXB,KEY,MAXYA,MAXYB,IA,IXLINE,XAXIS,PRMODE)
```

```
*************************************************************
*          PLOTTING SCATTEGRAM - MAIN SUBROUTINE            *
*************************************************************
      DOUBLE PRECISION XMIN,XSTEP,YMIN,YSTEP,Z,Q
      INTEGER MAXA,MAXB,MAXYA,MAXYB,NOBS,NSTEPX,NSTEPY,P,L1,K1,NN
      DOUBLE PRECISION X(NOBS),XAXIS(28),Y(NOBS)
      INTEGER IA(133),IXLINE(133),KEY(NOBS)
      DOUBLE PRECISION ZDIFF,YAXIS,YDIFF
      INTEGER I,IAY,IFOR,II,ISP1,ISP2,ISPACE,IXAX,IXZERO,
     & IYZERO,J,J1,K,MAXF,MAXYF,NCOUNT,NEND,NSTPX2,PRMODE
      LOGICAL LXE,LYE
      CHARACTER MA(27),MB(22),MO(17),ME(39),NB(10)
      INTEGER ICODE(39),IFMA(27),IFMB(22),IFMO(17),NUMB(10)
      DATA MO  /'(',' ',' ','X',',','2','8','(','F',' ','.',' ',',',
     &' ','X',')',')'/
      DATA MA /'(','1','X',',','F','1','1','.',' ',',',' ',
     &' ',' ','A','1',',','1','X',',','3','0','(','I','5',':',')',')'/
      DATA MB /'(','1','2','X',',',' ',' ',' ','A','1',',','1',
     &'X',',','3','0','(','I','5',':',')',')'/
      DATA ME /'.','+',' ','1','2','3','4','5',
     &'6','7','8','9','A','B','C','D','E','F','G','H','I','J','K','L',
     &'M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','*'/
      DATA  NB /'0','1','2','3','4','5','6','7','8','9'/
      DO 1 I=1,17
 1      IFMO(I)=ICHAR(MO(I))
      DO 2 I=1,27
 2      IFMA(I)=ICHAR(MA(I))
      DO 3 I=1,22
 3      IFMB(I)=ICHAR(MB(I))
      DO 4 I=1,39
 4      ICODE(I)=ICHAR(ME(I))
      DO 5 I=1,10
 5      NUMB(I)=ICHAR(NB(I))
      YDIFF = 0.0D0
      ZDIFF = 0.0D0
      IXZERO = 0
      IYZERO = 0
      IF (XMIN.NE.0.0D0) THEN
         IF (DABS((XMIN/XSTEP)-DINT(XMIN/XSTEP+DSIGN(0.5D0,XMIN))).
     &   GE.0.00005D0) ZDIFF = DMOD(XMIN,XSTEP)
         IF (ZDIFF.NE.0.0D0) THEN
            IF (ZDIFF.LT.0.0D0) ZDIFF = XSTEP + ZDIFF
            XMIN = XMIN - ZDIFF
            NSTEPX = NSTEPX + 1
         ENDIF
         IF (XMIN.NE.0.0D0)
     &      IXZERO = -IDINT(XMIN/XSTEP+DSIGN(0.5D0,XMIN))
      ENDIF
      IF (YMIN.NE.0.0D0) THEN
         IF (DABS((YMIN/YSTEP)-DINT(YMIN/YSTEP+DSIGN(0.5D0,YMIN))).
     &   GE.0.00005D0) YDIFF = DMOD(YMIN,YSTEP)
         IF (YDIFF.NE.0.0D0) THEN
            IF (YDIFF.LT.0.0D0) YDIFF = YSTEP + YDIFF
            YMIN = YMIN - YDIFF
            NSTEPY = NSTEPY + 1
```

```
      ENDIF
    ENDIF
    IF (YMIN.NE.0.0D0)
   &    IYZERO = -IDINT(YMIN/YSTEP+DSIGN(0.5D0,YMIN))
    NSTPX2 = NSTEPX + 2
    DO 10 I=1,NSTPX2
       IXLINE(I) = ICODE(1)
       IF (MOD(IABS(IXZERO-I+1),5).EQ.0) THEN
          IXLINE(I) = ICODE(2)
          NEND = (I-1)/5 + 1
          XAXIS(NEND) = XMIN + XSTEP*DBLE(I-1)
       ENDIF
10  CONTINUE
    IFOR = MOD(IXZERO,5)
    MAXYF = MAXYA + MAXYB + 2
    IF (MAXYF.LE.11) THEN
       IFMA(9) = NUMB(MAXYB+1)
       LYE = .FALSE.
    ELSE
       IFMA(5) = ICODE(17)
       IFMA(9) = NUMB(4)
       LYE = .TRUE.
    ENDIF
    J1 = NSTPX2/100
    J = J1 + 1
    IFMA(11) = NUMB(J)
    IFMB(6) = NUMB(J)
    J = NSTPX2/10 + 1 - J1*10
    IFMA(12) = NUMB(J)
    IFMB(7) = NUMB(J)
    J = MOD(NSTPX2,10) + 1
    IFMA(13) = NUMB(J)
    IFMB(8) = NUMB(J)
    IF (MOD(IABS(IYZERO-(NSTEPY+1)),5).NE.0) THEN
       DO 20 I=1,22
           MB(I)=CHAR(IFMB(I))
20     CONTINUE
       WRITE (2,MB) (IXLINE(I),I=1,NSTPX2)
    ELSE
       YAXIS = YMIN + YSTEP*DBLE(NSTEPY+1)
       DO 30 I=1,27
         MA(I)=CHAR(IFMA(I))
30     CONTINUE
       WRITE (2,MA) YAXIS,(IXLINE(I),I=1,NSTPX2)
    ENDIF
    NCOUNT = 1
    DO 110 J=1,NSTEPY
      K = NSTEPY - J + 1
      DO 40 I=1,NSTEPX
         IA(I) = 3
40    CONTINUE
      K1=NCOUNT
50    IF (NCOUNT.GT.NOBS) GO TO 70
      IAY = (Y(NCOUNT)-YMIN)/YSTEP + 0.5D0
      IF (IAY.EQ.0) IAY = 1
```

```
      IF (IAY.LT.K) GO TO 70
      II = KEY(NCOUNT)
      IXAX = (X(II)-XMIN)/XSTEP + 0.5D0
      IF (IXAX.EQ.0) IXAX = 1
      IF (IA(IXAX).GT.3) THEN
         IA(IXAX) = IA(IXAX) + 1
      ELSE
         IA(IXAX) = 4
         IF (NCOUNT.GT.K1) THEN
            L1=NCOUNT-1
            Z=Y(NCOUNT)
            Q=X(KEY(NCOUNT))
            P=KEY(NCOUNT)
60            IF (L1.GE.K1) THEN
                IF(Q.LT.X(KEY(L1))) THEN
                  Y(L1+1)=Y(L1)
                  KEY(L1+1)=KEY(L1)
                  L1=L1-1
                  GOTO 60
                ENDIF
              ENDIF
              Y(L1+1)=Z
              KEY(L1+1)=P
         ENDIF
      ENDIF
      NCOUNT = NCOUNT + 1
      GO TO 50
70    DO 80 I=1,NSTEPX
         IF (IA(I).EQ.3) THEN
            IF (K.EQ.IYZERO) THEN
               IA(I) = 1
               IF (MOD(IABS(IXZERO-I),5).EQ.0) IA(I) = 2
            ELSE
               IF (I.EQ.IXZERO) THEN
                  IA(I) = 1
                  IF (MOD(IABS(IYZERO-K),5).EQ.0) IA(I) = 2
               ENDIF
            ENDIF
         ENDIF
         II = MIN0(IA(I),39)
         IA(I) = ICODE(II)
80    CONTINUE
      IF (PRMODE.EQ.1) THEN
           NN=30
         ELSE
           NN=10
      ENDIF
      IF (MOD(IABS(IYZERO-K),5).NE.0) THEN
         DO 90 I=1,22
             MB(I)=CHAR(IFMB(I))
90       CONTINUE
         WRITE (2,MB) ICODE(1),(IA(I),I=1,NSTEPX),ICODE(1),
     &         (KEY(I),I=K1,MIN0(K1+NN,NCOUNT-1))
      ELSE
         YAXIS = YMIN + YSTEP*DBLE(K)
```

```
          DO 100 I=1,27
             MA(I)=CHAR(IFMA(I))
100       CONTINUE
          WRITE (2,MA) YAXIS,ICODE(2),(IA(I),I=1,NSTEPX),ICODE(2),
   &               (KEY(I),I=K1,MINO(K1+NN,NCOUNT-1))
       ENDIF
110 CONTINUE
    IF (MOD(IABS(IYZERO),5).NE.0) THEN
       DO 120 I=1,22
          MB(I)=CHAR(IFMB(I))
120    CONTINUE
       WRITE (2,MB) (IXLINE(I),I=1,NSTPX2)
      ELSE
       YAXIS = YMIN
       DO 130 I=1,27
          MA(I)=CHAR(IFMA(I))
130    CONTINUE
       WRITE (2,MA) YAXIS,(IXLINE(I),I=1,NSTPX2)
    ENDIF
    IFOR = MOD(IXZERO,5)
    IF (IFOR.LT.0) IFOR = IFOR + 5
    ISP1 = 11 + IFOR
    IF (MAXA+MAXB+2.LE.9) ISP1 = ISP1 - MAXA
    I = ISP1/10 + 1
    IFMO(2) = NUMB(I)
    I = MOD(ISP1,10) + 1
    IFMO(3) = NUMB(I)
    MAXF = MAXA + MAXB + 2
    IF (MAXF.LE.9) THEN
       ISPACE = 10 - MAXF
       LXE = .FALSE.
       IFMO(10) = NUMB(MAXF+1)
       IFMO(12) = NUMB(MAXB+1)
       IFMO(14) = NUMB(ISPACE+1)
    ELSE
       IFMO(9) = ICODE(17)
       IFMO(10) = NUMB(10)
       IFMO(12) = NUMB(3)
       IFMO(14) = NUMB(2)
       LXE = .TRUE.
    ENDIF
    DO 140 I=1,17
         MO(I)=CHAR(IFMO(I))
140 CONTINUE
    WRITE (2,MO) (XAXIS(II),II=1,NEND,2)
    ISP2 = ISP1 + 5
    I = ISP2/10 + 1
    IFMO(2) = NUMB(I)
    I = MOD(ISP2,10) + 1
    IFMO(3) = NUMB(I)
    DO 150 I=1,17
        MO(I)=CHAR(IFMO(I))
150 CONTINUE
    WRITE (2,MO) (XAXIS(II),II=2,NEND,2)
    IF (LXE) IFMO(9) = ICODE(18)
```

```
      IF (LYE) IFMA(5) = ICODE(18)
      RETURN
      END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
       SUBROUTINE CDTR (X,G,P,IER)
*****************************************************************
*     COMPUTES P(X) = PROBABILITY THAT RANDOM VARIABLE          *
*     DESCRIBED ACCORDING TO THE CHI-SQUARE DISTRIBUTION WITH   *
*     G DEGREES OF FREEDOM, IS LESS THAN OR EQUAL TO X          *
*****************************************************************
       REAL               X,G,P
       INTEGER            I,IER
       REAL               PT2,X1,NORMAL,RINFM
       DOUBLE PRECISION   A,Z,DGAM,EPS,W,W1,B,Z1,HALF,ONE,THRTEN,THRD
       DOUBLE PRECISION   DGAMMA,FUNC,XINF,XMIN
       COMMON  RINFM,XINF,XMIN
       DATA    EPS/1.0D-6/,HALF/5.D-1/,THRTEN/13.D0/,ONE/1.D0/
       DATA    THRD/.3333333333333333D0/,PT2/.2222222E0/
       FUNC(W,A,Z)=W*DEXP(A*DLOG(Z)-Z)
       IF (G.LT..5. OR. G.GT.2.E5 .OR. X.LT.0.0) THEN
          P=RINFM
          WRITE (2,*) 'DATA ERROR IN THE SUBROUTINE CDTR '
          IER=1
          RETURN
       ENDIF
       IER=0
       IF (X .LE. 1.E-12) THEN
          P=0.0
          RETURN
       ENDIF
       IF(G.GT.66.) THEN
          IF(X.LT.2.0) THEN
        P=0.0
        RETURN
          ENDIF
          X1=((X/G)**THRD-(ONE-PT2/G))/SQRT(PT2/G)
          IF (X1 .GT. 5.0) THEN
        P=1.0
        RETURN
          ENDIF
          IF (X1 .LT. -18.8055) THEN
        IER=1
        P=0.0
        RETURN
          ENDIF
          P=NORMAL(X)
          RETURN
       ENDIF
       IF (X .GT. 200.) THEN
          P=1.0
          RETURN
       ELSE
          A=HALF*G
          Z=HALF*X
          DGAM = DGAMMA(A)
```

```
        W=DMAX1(HALF*A,THRTEN)
        IF (Z.LT.W) THEN
      IF (G.GT.25. .AND. X.LT.2.) THEN
           P=0.0
           RETURN
        ENDIF
        W=ONE/(DGAM*A)
        W1=W
           DO 10 I=1,50
         B=I
         W1=W1*Z/(A+B)
         IF (W1 .LE. EPS*W) GO TO 20
         W=W+W1
   10          CONTINUE
   20        P=FUNC(W,A,Z)
        RETURN
          ENDIF
          Z1=ONE/Z
          B=A-ONE
          W1=B*Z1
          W=ONE+W1
        DO 30 I=2,50
           B=B-ONE
           W1=W1*B*Z1
           IF (W1 .LE. EPS*W) GO TO 40
           W=W+W1
   30       CONTINUE
   40     W=Z1*FUNC(W,A,Z)
          P=ONE-W/DGAM
       ENDIF
       END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      DOUBLE PRECISION FUNCTION DGAMMA (X)
*****************************************
*     EVALUATES GAMMA FUNCTION          *
*****************************************
       DOUBLE PRECISION  X
       REAL              RINFM
       DIMENSION         P(9),Q(8),Z(7)
       DOUBLE PRECISION  A,B,DEN,P,Z,Q,PI,T,TOP,BIG1,XINF,XMIN,Y,SIGN,R
       INTEGER           I,J
       LOGICAL           MFLAG
       COMMON            RINFM,XINF,XMIN
       DATA              P/-5.966047488753637D01, 5.864023793062003D01,
     &                     -1.364106217165365D03,-8.117569271425580D02,
     &                     -1.569414683149179D04,-1.525979925758372D04,
     &                     -7.264059615964330D04,-8.972275718101010D-01,
     &                      3.349618189847578D00/
       DATA              Q/ 4.103991474182904D02,-2.262590291514875D03,
     &                      2.494325576714903D03, 2.362106244383048D04,
     &                     -5.741873227396418D04,-7.257239715408240D04,
     &                     -9.491399521686949D00,-3.255006939455704D01/
       DATA              Z/ 8.40596949829D-04,    -5.9523334141881D-04,
     &                      7.9365078409227D-04, -2.777777777769526D-03,
     &                      8.333333333333333D-02,9.189385332046727D-01,
```

```
     &                        -1.7816839846D-03/
      DATA      PI/3.141592653589793D0/,BIG1/34.844D0/
      MFLAG = .FALSE.
      T = X
      IF (DABS(T).GT.XMIN) THEN
         IF (DABS(T).GE.BIG1) THEN
       DGAMMA = XINF
       RETURN
         ELSE
       DGAMMA = XINF
       IF (T.LE.0.0D0) THEN
          DGAMMA = -XINF
          RETURN
       ENDIF
         ENDIF
      ENDIF
      IF (T.LE.0.0D0) THEN
         MFLAG = .TRUE.
         T = -T
         R = DINT(T)
         SIGN = 1.0D0
         IF (DMOD(T,2.0D0).EQ.0.0D0) SIGN = -1.0D0
         R = T-R
         IF (R.NE.0.0D0) THEN
       R = PI/DSIN(R*PI)*SIGN
       T = T+1.0D0
         ELSE
       DGAMMA = XINF
       IF (SIGN.EQ.-1.0D0) THEN
          DGAMMA = -XINF
          RETURN
       ENDIF
         ENDIF
      ENDIF
      IF (T.GT.12.0D0) THEN
         TOP = DLOG(T)
         TOP = T*(TOP-1.0D0)-.5D0*TOP
         T = 1.0D0/T
         B = T*T
         A = Z(7)
         DO 10 J = 1,5
10           A = A*B+Z(J)
         Y = A*T+Z(6)+TOP
         Y = DEXP(Y)
         IF (MFLAG) Y = R/Y
         DGAMMA = Y
      ELSE
         I = T
         A = 1.0D0
         IF (I.GT.2) THEN
           DO 20 J=3,I
         T = T-1.0D0
         A = A*T
20          CONTINUE
         ELSE
```

```
        I = I+1
        IF (I .EQ.1) THEN
           A = A/(T*(T+1.0D0))
           T = T+2.0D0
        ENDIF
        IF (I .EQ. 2) THEN
           A = A/T
           T = T+1.0D0
        ENDIF
          ENDIF
          TOP = P(8)*T+P(9)
          DEN = T+Q(8)
          DO 30 J=1,7
        TOP = TOP*T+P(J)
        DEN = DEN*T+Q(J)
   30      CONTINUE
          Y = (TOP/DEN)*A
          IF (MFLAG) Y = R/Y
          DGAMMA = Y
       ENDIF
       END
* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> *
      REAL FUNCTION NORMAL(X)
**********************************************************************
*     EVALUATES THE LOWER TAIL AREA OF THE STANDARIZED NORMAL CURVE  *
**********************************************************************
      REAL X
      REAL Y,W
      DOUBLE PRECISION Z
      IF (X.EQ.0.0) THEN
         Z=0D0
      ELSE
         Y=ABS(X)*0.5
         IF (Y.GT.3.0) THEN
            Z=1.0
         ELSE
            IF (Y.LT.1) THEN
               W=Y*Y
               Z=((((((((0.000124818987D0*W
     &          -0.001075204047D0)*W + 0.005198775019D0)*W
     &          -0.019198292004D0)*W + 0.059054035642D0)*W
     &          -0.151968751364D0)*W + 0.319152932694D0)*W
     &          -0.531923007300D0)*W + 0.797884560593D0)*Y*2D0
            ELSE
               Y=Y-2.0
               Z=((((((((((((( -0.000045255659D0*Y
     &          +0.000152529290D0)*Y - 0.000019538132D0)*Y
     &          -0.000676904986D0)*Y + 0.001390604284D0)*Y
     &          -0.000794620820D0)*Y - 0.002034254874D0)*Y
     &          +0.006549791214D0)*Y - 0.010557625006D0)*Y
     &          +0.011630447319D0)*Y - 0.009279453341D0)*Y
     &          +0.005353579108D0)*Y - 0.002141268741D0)*Y
     &          +0.000535310849D0)*Y + 0.999936657524D0
            ENDIF
         ENDIF
      ENDIF
```

```
IF(X.GT.0) THEN
  NORMAL=REAL((Z+1D0)*0.5D0)
ELSE
  NORMAL=REAL((1D0-Z)*0.5D0)
ENDIF
END
```