Mathematical Geology, Vol. 3, No. 1, 1971

An Algorithm and FORTRAN-IV Program for Large-Scale *Q*-Mode Factor Analysis and Calculation of Factor Scores[1]

J. E. Klovan[2] and J. Imbrie[3]

An algorithm and FORTRAN-IV *computer program,* CABFAC, *for* Q*-mode factor analysis is described. The program will accept up to 1500 items and 50 variables on a moderate-size computer.* KEY WORDS: factor analysis, principal components analysis.

INTRODUCTION

Q-mode factor analysis is a multivariate procedure useful for studying relationships among items. The technique has been recently applied to geologic problems (Imbrie and Van Andel, 1964; Harbaugh and Demirmen, 1964; Klovan, 1966) and programs for its implementation are readily available (Manson and Imbrie, 1964; Klovan, 1968; Ondrick and Srivastava, 1970; Imbrie and Kipp, 1970). Because the method usually involves the computation of an $N \times N$ similarity matrix (where N equals the number of items in the study) and N is, in most studies, usually large, these programs can only be run on large computers. A practical limit of 200 items is usually encountered even on large machines.

The present program, CABFAC, *Ca*lgary and *B*rown *Fac*tor Analysis, is designed to accommodate up to 1500 items on a moderate-sized computer, such as an IBM 360/50. This is made possible by making use of certain properties of the matrices used in the analysis; rather than factoring an $N \times N$ matrix of similarity coefficients an $n \times n$ matrix of cross products is factored, where n is the number of variables used.

DEFINITION OF MATRICES

The following notation will be used:

X is the $N \times n$ raw-data matrix, where N is the number of items (rows) and n is the number of variables (columns).

D is an $N \times N$ diagonal matrix containing the row-sum squares of X.

---

[1] Manuscript received 26 August 1970.

[2] Department of Geology, The University of Calgary (Canada).

[3] Department of Geological Sciences, Brown University (USA).

$W = D^{-1/2}X$ is the row-normalized data matrix, that is, each item in W has unit vector length.

$S = WW'$ is the $N \times N$ cosine-theta matrix, that is, the degree of proportional similarity between each pair of items.

$P = W'W$ is the $n \times n$ matrix of cross products among the n variables.

Λ is the diagonal matrix of nonzero eigenvalues of S.

U is the columnwise orthonormal matrix of eigenvectors associated with Λ.

Q is the principal factor-loadings matrix.

F_p is the principal factor-score matrix, columnwise-orthonormal.

F_{ps} is the scaled, principal factor-score matrix.

V is the varimax factor-loadings matrix.

F_v is the varimax factor-score matrix.

F_{vs} is the scaled, varimax factor-score matrix.

PROCEDURE

The usual solution to a Q-mode factor analysis consists of the following steps:

(1) $W = QF_p'$, the basic factor equation (Harman, 1967).

(2) $WW' = S$, the cosine-theta matrix (Imbrie and Purdy, 1962).

Q, the factor matrix associated with S, is computed such that:

(3) $QQ' = S$, conditional on:

(4) $Q'Q = \Lambda$, and

(5) $F_p'F_p = I$.

It is also true that:

(6) $U'SU = \Lambda$, and it follows that:

(7) $Q = U\Lambda^{-\frac{1}{2}}$.

The factor score matrix may be found from:

$$W = QF_p' \text{ and } Q'W = \Lambda F_p'.$$

(8) $F_p = W'Q\Lambda^{-1}$.

The procedure followed in CABFAC takes advantage of certain properties of these matrices to achieve the same results but with less stringent storage requirements. The steps in this procedure are:

(9) $W'W = P$, the $n \times n$ cross-products matrix.

We also define:

(10) $\overset{*}{U}'P\overset{*}{U} = \overset{*}{\Lambda}$, where $\overset{*}{U}$ and $\overset{*}{\Lambda}$ are the eigenvectors and eigenvalues associated with P, that is, $P = \overset{*}{U}\overset{*}{\Lambda}\overset{*}{U}'$.

Here, note the order of P is equal to the number of columns of X, whereas the order of S is equal to the number of rows of X.

It is known from matrix theory that the nonzero elements of $\overset{*}{\Lambda}$, which are the eigenvalues of $W'W$, are equal to the nonzero elements of Λ, the eigenvalues of WW'.

Thus:

(11) $\Lambda = \overset{*}{\Lambda}$.

Because $\overset{*}{U}$ is orthonormal we may write the equation in step (2) as $S = WW' = W\overset{*}{U}\overset{*}{U}'W'$ or

(12) $S = W\overset{*}{U}(W\overset{*}{U})'$.

Because of (11) we may write $\Lambda = \overset{*}{U}'P\overset{*}{U}$ or $\Lambda = \overset{*}{U}'W'W\overset{*}{U}$ or

(13) $\Lambda = (W\overset{*}{U})'W\overset{*}{U}$.

Comparing (3) with (12) and (4) with (13) we see:

(14) $QQ' = S = W\overset{*}{U}(W\overset{*}{U})'$ and

(15) $Q'Q = \Lambda = (W\overset{*}{U})'W\overset{*}{U}$.

We may therefore write:

(16) $Q = W\overset{*}{U}$.

The principal factor-score matrix becomes $F_p = W'Q\Lambda^{-1} = W'W\overset{*}{U}\Lambda^{-1} = P\overset{*}{U}\Lambda^{-1}$

(17) $= \overset{*}{U}\Lambda\overset{*}{U}'\overset{*}{U}\Lambda^{-1} = \overset{*}{U}$.

Thus, instead of having to store and diagonalize the $N \times N$ matrix S, we can achieve the same results by diagonalizing the $n \times n$ matrix P and perform simple matrix multiplication to arrive at the desired matrix Q.

FACTOR SCORES PROCEDURE

As noted above in (17), the factor scores, which are an estimate of the "composition" of the factors, are obtained directly by the procedure outlined here. Because the size of the numbers obtained in F_p is in part a reflection of the number of variables used in the analysis, the present program, in addition to the factor scores above, adjusts, or scales, the factor scores by multiplying each of them by the square root of the number of variables. The resulting matrix F_{ps} makes possible the comparison of factor scores between studies in which different numbers of variables have been used. If all variables in a study are equally important in a factor, then each *scaled* score equals unity. The minimum absolute value a factor score can attain is zero, indicating that a variable contributes nothing to a factor. The maximum possible absolute value, occurring when other variables have zero scores, is $\sqrt{n}$.

ROTATIONAL PROCEDURE

The principal factor matrix is rotated to satisfy the varimax criterion using the well-known method of Kaiser (1958).

Varimax factor scores are obtained from the expression:

(18) $F_v = F_pT$

where T is the varimax transformation matrix and F_v is the varimax factor-score matrix. The scaled, varimax factor scores F_{vs} also are computed, as is the varimax factor-loadings matrix V.

EXTENSION OF A FACTOR ANALYSIS TO ANOTHER DATA SET

After performing a factor analysis on one set of data from a universe of inquiry there are occasions when it is desirable or necessary to extend these results directly to another data set from the same universe without the usual algebraic manipulation. After factoring foraminiferal data obtained from samples widely distributed over the Atlantic sea bottom, for example, Imbrie and Kipp (1970) wished to describe fossil assemblages from one deep-sea core in terms of the varimax factors gained from the sea-bottom study. Because the same species occur in both sets of data this objective is possible, using the procedure described below and options available in CABFAC. The results achieved are, in this example, considerably different than would be achieved by factoring directly the fossil assemblages from each core. The sea-bottom factors describe the fauna in terms of ecologic variation patterns characteristic of the entire Atlantic ocean, whereas factors obtained by the usual analysis of the core data would describe the faunal variations in terms of ecologic changes occurring at one particular site and representing a much narrower range of variation. By extending the sea-bottom factors to the core samples it is possible to interpret the factor analysis of the fossil record directly in terms of the modern ocean.

Consider a factor analysis of a set of data X in terms of the varimax solution:

$$W_x = V_x F_v'$$

where W_x is the normalized data matrix corresponding to X, and V_x is the corresponding varimax loadings matrix. Because F_v' is columnwise orthonormal, we can solve the equation for V_x by writing

$$V_x = W_x F_v$$

and view F_v as an operator analyzing any normalized data matrix into varimax components by postmultiplication. Thus, given another set of data Y from the same universe of inquiry, and a corresponding normalized data matrix W_y, we can resolve it into the original varimax components by postmultiplication:

$$V_y = W_y F_v$$

Options in CABFAC provide punched output for F_v and W_y. Thus, given F_v from a data set X, one can factor data set Y simply by obtaining W_y and performing the appropriate matrix multiplication.

PROGRAM DESCRIPTION

This program has been designed with four objectives in mind: (1) solution of large data matrices; (2) minimizing the size of core requirements; (3) ease of use; and (4) maximizing the useful output.

As presented here the program will handle a maximum of 1500 items (rows of data matrix) and 50 variables (columns). It will extract a maximum of 10 factors. These restrictions can be expanded by changing the appropriate DIMENSION statements.

To ensure ease of use, the control cards required to run the program have been kept simple. The penalty for this is that only a few options are included. It is felt that individual users may want to add their own options for their particular needs.

The program is designed to do multiple jobs; several sets of data may be stacked for one machine pass.

In many studies it is not possible to specify in advance the optimum number of factors. Our program overcomes this problem as follows. First, the user decides how much of the total information, in terms of sums of squares, he wants to retain. In situations where little is known about the data it is suggested that one should be generous and ask for 99 percent. The program will extract and rotate the number of factors necessary to account for this percentage or a maximum of 10 factors, whichever occurs first. It also will ignore any factors whose eigenvalue falls below 0.0001.

Determining the number of factors in this manner does not, however, insure that all of them will be useful when rotated. In order to avoid several reruns, the following rotational procedure has been adopted. First, all the factors extracted according to the above criterion are rotated. Then, the least significant factor is deleted and the rotation is done again. This process continues until two factors have been rotated, or until less than 75 percent of the total information remains. In this manner the user can see all possible varimax rotations after one pass and select the appropriate one. Factor scores also are computed for each set of rotated factors. Whenever three or two factors are rotated, the varimax matrix is recalculated and printed in row-normalized form so that each row adds up to 1.0. This is for convenience in plotting on a triangular diagram.

If the user knows the number of factors to be rotated prior to the run, he may specify this number on the control card and only one varimax matrix and associated factor scores will be calculated.

PROGRAM INPUT

Control Card 1

Columns 1–2—the number of separate jobs to be run. For each job, control cards 2–6 are necessary and precede each data deck.

Control Card 2

Columns 1–80—alphanumeric title of the job.

Control Card 3

Columns 1–2—NV—the number of variables (columns) of the data matrix (maximum of 50).

Columns 3–6—NS—the number of items (rows) of the data matrix (maximum of 1500).

Columns 7–11—QUIT—the stopping criterion, that is, the total percent of information (sums of squares) to be explained by the factors. For example, 99.9 (a decimal point must be punched) means that factors are to be extracted until they account for 99.9 percent of the information in the data matrix.

Columns 12–13—NROT—the number of factors to be rotated in the varimax rotation (maximum of 10). If this is left blank the number rotated is taken to be that number required to give the percent information required as specified by QUIT. Then the last factor is deleted and the rotation is repeated. This process continues until only two factors are rotated.

Column 14—IPUN—a one causes the row-normalized data matrix, the principal factor matrix, and the unscaled, principal factor scores to be punched. If NROT and IPUN are both nonzero, the desired varimax factor matrix and associated scores also will be punched.

Column 15—ITRANS—a blank or zero results in factoring of the data as read in; a one transforms each variable to a percent of its maximum value; a two transforms each variable to a percent of its range.

Control Cards 4 and 5

Input data format cards. Their form is discussed in the following section. If all the necessary information can be placed on one card, a second blank card *must* follow.

DATA CARDS

The data matrix should be punched rowwise, that is, variables for each item are contained on one or more cards.

The first 12 columns of the first data card for each item are reserved for alphanumeric identification of the item. Thus, the input format card should *always* begin with 3A4. For example, if 15 variables (columns) are present in the data matrix, data cards might be punched up as follows.

The first 12 columns of the data card would contain the name of the item, the first six variables might take up the rest of the card in fields of 10 digits, and the remaining nine variables might be on a second card in fields of eight digits. The format statement to be punched on CARD 4-5 would be (3A4,6F10.3/9F8.2).

MACHINE REQUIREMENTS

The version presented here runs successfully on an IBM 360/50. It uses approximately 170 k bytes of memory.

The card reader, printer, and punch have been assigned variable names. The numeric values of these units may be changed in the first few cards of the main program.

In addition, two scratch tapes or working areas on disc are required. Again, these are assigned variable names and may be changed to suit a particular system.

ACKNOWLEDGMENTS

Discussions with Dr. V. Manson of the New York Museum of Natural History and F. E. Steidler, Esso Research and Engineering Co., contributed much to the development of the algorithm presented here. Design and testing of the program was done at The University of Calgary with the support of a National Research Council of Canada grant to the senior author. Additional testing and development of the factor-extension procedure were carried out at Brown University by the junior author; the research was supported by National Science Foundation Grants NSF-GP-4994, NSF-GA-1303, and NSF-GA-14853.

REFERENCES

Harbaugh, J. W., and Demirmen, F., 1964, Application of factor analysis to petrologic variations of Americus Limestone (Lower Permian), Kansas and Oklahoma: Kansas Geol. Survey Sp. Dist. Publ. 15, 40 p.

Harman, H. H., 1967, Modern factor analysis (2nd ed.): Univ. of Chicago Press, Chicago, 474 p.

Imbrie, J., and Kipp, N. G., 1970, A new micropaleontological method for quantitative paleoclimatology: application to a late Pleistocene Caribbean core, *in* The Late Cenozoic glacial ages: Yale Univ. Press, New Haven, in press.

Imbrie, J., and Purdy, E. G., 1962, Classification of modern Bahamian carbonate sediments: Am. Assoc. Petroleum Geologists Mem. 1, p. 253–272.

Imbrie, J., and Van Andel, T. H., 1964, Vector analysis of heavy-mineral data: Geol. Soc. America Bull., v. 75, no. 11, p. 1131–1156.

Kaiser, H. F., 1958, The varimax criterion for analytic rotation in factor analysis: Psychometrika, v. 23, no. 3, p. 187–200.

Klovan, J. E., 1966, The use of factor analysis in determining depositional environments from grain-size distributions: Jour. Sed. Pet., v. 36, no. 1, p. 115–125.

Klovan, J. E., 1968, Q-mode factor analysis program in FORTRAN-IV for small computers: Kansas Geol. Survey Computer Contr. 20, p. 39–51.

Manson, V., and Imbrie, J., 1964, FORTRAN program for factor and vector analysis of geological data using an IBM 7090 or 7094 computer system: Kansas Geol. Survey Sp. Dist. Publ. 13, 46 p.

Ondrick, C. W., and Srivastava, G. S., 1970, CORFAN–FORTRAN IV computer program for correlation, factor analysis (R- and Q-mode) and varimax rotation: Kansas Geol. Survey Computer Contr. 42, 92 p.

APPENDIX

Listing of CABFAC in FORTRAN-IV

```
C     CABFAC
C     Q-MODE FACTOR ANALYSIS USING DUALITY CONCEPT
C      J.E. KLOVAN
C      DEPT OF GEOLOGY
C     UNIVERSITY OF CALGARY
C     REVISED VERSION JULY 10,1970
C     ------------------------------------------------------------------------
C     CONTROL CARD SET-UP
C     ------------------------------------------------------------------------
C    FIRST CARD COLS 1-2 : THE NUMBER OF JOBS TO BE RUN
C     ------------------------------------------------------------------------
C    FOR EACH JOB THE FOLLOWING CONTROL CARDS PRECEED DATA DECK OF EACH JOB.
C      CONTROL CARD 1:
C         COLS 1 - 80 ANY JOB TITLE
C       CONTROL CARD 2:
C          COLS 1 - 2 = NV = NUMBER OF VARIABLES (MAX. OF 50 )
C          COLS 3 - 6 = NS = THE NUMBER OF SAMPLE POINTS (MAX OF1500 )
C          COLS 7 - 11 = QUIT = STOPPING CRITERION - CUMULATIVE % SUMS OF
C                               SQUARES TO BE EXPLAINED
C          COLS 12 - 13 = NROT = NUMBER OF FACTORS TO BE ROTATED
C          COL  14 = IPUN =  1 IF NORMALIZED DATA,FACTOR LOADINGS AND
C                            SCORES TO BE PUNCHED.
C          COL  15 = ITRANS = 0 FOR RAW DATA
C                           = 1 FOR % MAXIMUM TRANSFORMATION
C                           = 2 FOR % RANGE TRANSFORMATION
C       CONTROL CARDS 3 & 4 INPUT FORMAT STATEMENT
C     THE FIRST 12 CHARACTERS ON THE DATA CARD ARE RESERVED FOR IDENT.
C       STATEMENT NO.3 READS IN THE DATA
C     ------------------------------------------------------------------------
      DIMENSION F(1500,10), NAME(1500,3), COM(1500)
      DIMENSION X(50),R(50,50), A(50,50),FS(50,10)
      DIMENSION Q(10), VAR(10), XX(10), T(10,10)
      DIMENSION TITLE(20), FMT(36)
      EQUIVALENCE (R(1,1),F(1,1)),(A(1,1),F(1,6))
       COMMON KT1,KT2,KT3,KT4,KT5
C     ------------------------------------------------------------------------
C     KT1 = CARD READER
C     KT2 = CARD PUNCH
C      KT3= PRINTER
C      KT4 & KT5 = SCRATCH TAPES
C     ------------------------------------------------------------------------
      KT1 = 5
      KT2 = 7
      KT3 = 6
      KT4 = 9
      KT5 = 8
      READ(KT1,1050) NJOBS
 1050 FORMAT(I2)
 2000  NJOBS = NJOBS -1
       IF(NJOBS) 36,2001,2001
 2001 REWIND KT4
      REWIND KT5
       READ(KT1,1020) TITLE
 1020 FORMAT(20A4)
 1007 FORMAT(1H1,20A4,///)
      WRITE(KT3,1007) TITLE
      READ(KT1,1000)NV,NS,QUIT,NROT,IPUN,ITRANS
 1000 FORMAT(I2,I4,F5.0,I2,2I1)
       READ(KT1,1001) FMT
 1001 FORMAT(18A4/18A4)
      WRITE(KT3,1008) NV,NS
 1008 FORMAT(1HO,'NUMBER OF VARIABLES = ',I3,4X,' NUMBER OF SAMPLES = ',
     1 I4)
```

```
      TRACE = C.
      SN=NS
      VN=NV
      SSN = SQRT(SN)
      SVN = SQRT (VN)
C
      IF(ITRANS.GE.1) CALL TRANS(FMT,NAME,X,A,NV,NS,ITRANS)
C     ---------------------------------------------------------------------
C     A(1,.) = VARIABLE MEANS
C     A(2,.) = VARIABLE STD. DEVS.
C     A(3,.) = VARIABLE MINIMUMS
C     A(4,.) = VARIABLE MAXIMUMS
C     ---------------------------------------------------------------------
C
      DO 1 J=1,NV
      A(1,J) = C.
      A(2,J) = C.
      A(4,J)= -1.0E30
      A(3,J) = 1.0E30
C     ---------------------------------------------------------------------
      DO 1 I=1,NV
    1 R(I,J) = C.
C
C     BEGIN READ IN LOOP AND COMPUTATION OF BASIC STATS.
C
      DO 2 K=1,NS
      COM(K) = C.
      SAMSSQ=C.C
      IF(ITRANS)3,3,4
    3 READ(KT1,FMT )(NAME(K,J),J = 1,3),(X(I),I = 1,NV)
      GO TO 6
    4 READ(KT5)      (NAME(K,J),J = 1,3),(X(I),I = 1,NV)
C
    6 CONTINUE
      DO 5 J = 1,NV
C
      A(1,J) = A(1,J) +X(J)
      A(2,J) = A(2,J) + X(J)**2
      IF(X(J) -A(3,J)) 42,43,43
   42 A(3,J) = X(J)
   43 IF(X(J) -A(4,J)) 41,41,44
   44 A(4,J) = X(J)
   41 CONTINUE
C
C     ---------------------------------------------------------------------
C     COMPUTING VECTOR LENGTHS
C
    5 SAMSSQ= SAMSSQ +X(J)* X(J)
      COM(K) = SQRT(SAMSSQ)
      SAMSSQ = SQRT(SAMSSQ)
C     ---------------------------------------------------------------------
C     NORMALIZE THE DATA AND PUT IT ON KT4
      DO 7 J=1,NV
    7 X(J) = X(J)/SAMSSQ
      WRITE(KT4)      (X(J),J=1,NV)
 1051 FORMAT(3A4,6F10.4/(8F10.4))
      IF(IPUN.GE.1)WRITE(KT2,1051)(NAME(K,J),J = 1,3),(X(I),I = 1,NV)
C     ---------------------------------------------------------------------
C     COMPUTE PSEUDO COS THETA MATRIX
C     ---------------------------------------------------------------------
C
      DO 9 I=1,NV
      DO 9 J=I,NV
    9 R(I,J)= R(I,J)+ X(I)*X(J)
    2 CONTINUE
C     ---------------------------------------------------------------------
C
      REWIND KT4
      REWIND KT5
C
      IF(ITRANS)80,80,81
   80 WRITE(KT3,1010)
 1010 FORMAT(////' GENERAL STATISTICS FOR RAW DATA',///
     11UH  VARIABLE 19X 8HSTANDARD  7X 7HMINIMUM 8X 7HMAXIMUM/9H   NUMBE
     2R 6X 7HAVERAGE 7X 9HDEVIATION 7X 5HVALUE 10X 5HVALUE //)
      GO TO 82
   81 WRITE(KT3,1007) TITLE
      WRITE(KT3,1011)
 1011 FORMAT(////' GENERAL STATISTICS FOR TRANSFORMED DATA',///
     11UH  VARIABLE 19X 8HSTANDARD  7X 7HMINIMUM 8X 7HMAXIMUM/9H   NUMBE
     2R 6X 7HAVERAGE 7X 9HDEVIATION 7X 5HVALUE 10X 5HVALUE //)
```

```
C
C     ------------------------------------------------------------------
C
   82 DO 10 I=1,NV
      DO 16 J = I,NV
   16 R(J,I)= R(I,J)
C
      A(1,I) = A(1,I)/SN
      A(2,I) = SQRT(A(2,I)     /SN -A(1,I)**2)
      WRITE(KT3,1005) I,A(1,I),A(2,I), A(3,I),A(4,I)
 1005 FORMAT(I7,4F15.4)
C
C     ------------------------------------------------------------------
C
   10 TRACE = TRACE + R(I,I)
C     ------------------------------------------------------------------
C     CALL THE EIGENVALUE ROUTINE
C     ------------------------------------------------------------------
      CALL HDIAG (R,NV,I,A,1,X)
C
C     ------------------------------------------------------------------
C     R(I,I) CONTAINS THE EIGENVALUES
C     A CONTAINS THE EIGENVECTORS
C     ------------------------------------------------------------------
C
C     NOW DETERMINE THE RIGHT NUMBER OF EIGENVALUES
C
C     ------------------------------------------------------------------
C
      WRITE(KT3,15)
   15 FORMAT(1HO, 'NO.   EIGENVALUE   CUM. VAR.',//)
      EVSUM = 0.
      DO 11 I = 1,NV
      NF = I
      EVSUM = EVSUM+ (R(I,I) *100. / TRACE)
      WRITE(KT3,14)I,R(I,I), EVSUM
   14 FORMAT(1H ,I2,2X,F10.6    ,3X, F7.2)
      IF(NF-10) 18,13,13
   18 IF(R(I,I) - 0.0001) 13,12,12
   12 IF(EVSUM - QUIT) 11,13,13
   11 CONTINUE
C     ------------------------------------------------------------------
C
C     TRANSFER EIGENVECTORS TO FS; EIGENVALUES TO Q
C
   13 DO 50 J = 1,NF
      Q(J) = R(J,J)
      DO 50 I = 1,NV
   50 FS(I,J) = A(I,J)
      WRITE(KT3,1007) TITLE
      WRITE(KT3,1012) (J,J =  1,NF)
 1012 FORMAT(1HO, '          PRINCIPAL FACTOR LOADINGS ',//,1X,
     1 '   COMM.',I6,1019,/)
C     ------------------------------------------------------------------
C     BEGIN MAJOR LOOP COMPUTING THE PRINCIPAL FACTOR MATRIX
C     ------------------------------------------------------------------
C
      DO 19  I = 1,NF
   19 VAR(I) = 0.
      DO 20 K= 1,NS
      READ(KT4)      (X(I),I=1,NV)
      DO 24 J=1,NF
      F(K,J) =0.
      DO 24 I=1,NV
   24 F(K,J) = X(I)*FS(I,J) +F(K,J)
      COMAL = 0.
      DO 25 J = 1,NF
      COMAL = COMAL + F(K,J)* F(K,J)
   25 VAR(J) = VAR(J) + F(K,J) * F(K,J)
C
      WRITE(KT5) (F(K,I), I = 1,NF)
      WRITE(KT3,1003)(NAME(K,J), J= 1,3),COMAL,(F(K,I), I = 1,NF)
 1003 FORMAT(1H ,3A4,11F9.4)
      IF(IPUN)20,20,21
C     ------------------------------------------------------------------
   21 WRITE(KT2,1004) (NAME(K,J), J = 1,3),(F (K,I), I = 1,NF)
 1004 FORMAT(3A4,10F6.3)
C     ------------------------------------------------------------------
   20 CONTINUE
C
C     ------------------------------------------------------------------
C
C     EXPRESS COL SUMS OF SQUARES AS A % OF TOTAL SUMS OF SQUARES.
C
      DO 26 I = 1,NF
   26 VAR(I) = VAR(I)/TRACE * 100.
```

```
      X(1) = VAR(1)
      DO 27 J = 2,NF
      K = J -1
   27 X(J) = X(K) + VAR(J)
      WRITE(KT3,1015) (VAR(I), I = 1,NF)
 1015 FORMAT(1HO,'VARIANCE     ',8X,10F8.2)
      WRITE(KT3,1016) (X(I), I = 1,NF)
 1016 FORMAT(1H ,'CUM. VARIANCE       ',10F8.2)
C
C     ------------------------------------------------------------------
C
C     COMPUTE PRINCIPAL FACTOR SCORES
C     ------------------------------------------------------------------
C
       WRITE(KT3,1007) TITLE
        WRITE(KT3,1030)(J,J=1,NF)
 1030 FORMAT(1HO, ' PRINCIPAL FACTOR SCORE MATRIX' ,
     1 //,11X,'VAR.', 5X, 10I8,//)
C       COMPUTE VARIMAX F. SCORE MATRIX
      DO 51 I = 1,NV
   51 WRITE(KT3,1031) I,(FS(I,J),J = 1,NF)
 1031 FORMAT(1H ,I2X,(2,6X, 10F8.3)
C
C     COMPUTE SCALED PRINCIPAL FACTOR SCORE MATRIX
C     ------------------------------------------------------------------
C
      WRITE(KT3,1007) TITLE
       WRITE(KT3,1032) (J,J = 1,NF)
 1032 FORMAT(1HO, ' SCALED PRINCIPAL FACTOR SCORE MATRIX',
     1 //,11X,'VAR.', 5X, 10I8,//)
      DO 55 I = 1,NV
      DO 56 J = 1,NF
   56 XX(J) = FS(I,J) * SVN
      IF(IPUN) 55,55,57
   57 WRITE(KT2,1033) I,(FS(I,J),J = 1,NF)
 1033 FORMAT(I2, 8X,10F7.4)
   55 WRITE(KT3,1031) I,(XX(J), J = 1,NF)
C
C     ------------------------------------------------------------------
C
C
      NFF= NF
      IF(NROT.GE.1) NF= NROT
   38 CONTINUE
C
C     ------------------------------------------------------------------
C
C     CALL VARIMAX ROTATION PROCEDURE
C
        CALL VARMAX(NF,NS,NV,TITLE,F, T,COM,NAME,NROT,IPUN)
C
C     ------------------------------------------------------------------
C
C      COMPUTE VARIMAX FACTOR SCORE MATRIX
C     ------------------------------------------------------------------
C
   62 WRITE(KT3,1007) TITLE
      WRITE(KT3,1041) (J, J = 1,NF)
 1041 FORMAT(1HO,' VARIMAX FACTOR SCORE MATRIX',
     1 //,11X,'VAR.', 5X, 10I8,//)
      DO 69 K = 1,NV
      DO 68 I = 1,NF
      XX(I) =0.
      DO 68 J = 1,NF
   68 XX(I) = FS(K,J)*T(J,I) + XX(I)
C
C      COMPUTE SCALED VARIMAX FACTOR SCORE MATRIX
C     ------------------------------------------------------------------
C
      DO 67 J = 1,NF
   67 A(K,J) = XX(J) * SVN
C
C      PUNCH VARIMAX FACTOR SCORE MATRIX IF NEEDED
      IF(IPUN) 69,69,71
   71 IF(NROT)69,69,72
   72 WRITE(KT2,1033) K, (XX(J), J = 1,NF)
   69 WRITE(KT3,1031) K, (XX(J), J = 1,NF)
C
C
C     WRITE SCALED VARIMAX FACTOR SCORE MATRIX
C     ------------------------------------------------------------------
C
      WRITE(KT3,1007) TITLE
      WRITE(KT3,1042) (J ,J = 1,NF)
 1042 FORMAT(1HO,'SCALED VARIMAX FACTOR SCORES'
     1 //,11X,'VAR.', 5X, 10I8,//)
      DO 70 I = 1,NV
   70 WRITE(KT3,1031) I,(A(I,J), J = 1,NF)
C
C     ------------------------------------------------------------------
C
```

```
C      CHECK TO SEE IF ADDITIONAL ROTATIONS ARE REQUIRED
C
       IF(NROT.GE.1) GO TO 2000
        NF = NF - 1
        IF(NF -2) 35,30,30
   30 IF(X(NF) - 75.)  35,31,31
   31 REWIND KT5
       DO 37 K = 1,NS
   37 READ(KT5) (F(K,I), I = 1,NFF)
       GO TO 38
C      ---------------------------------------------------------------------
   35 GO TO 2000
   36 WRITE(KT3,1043)
 1043 FORMAT(1HO,' NORMAL END OF JOB')
       STOP
       END

       SUBROUTINE TRANS          (FMT,NAME,X,A,NV,NS,ITRANS)
       DIMENSION FMT(36), NAME(1500,3),X(50),A(50,50)
        COMMON KT1,KT2,KT3,KT4,KT5
       SN = NS
       DO 1 J=1,NV
       A(1,J) = 0.
       A(2,J) = 0.
       A(3,J) = 1.0E30
     1 A(4,J)= -1.0E30
       DO 2 K=1,NS
     3 READ(KT1,FMT )(NAME(K,J),J = 1,3),(X(I),I = 1,NV)
       WRITE(KT4)     (NAME(K,J),J = 1,3),(X(I),I = 1,NV)
       DO 5 J=1,NV
       A(1,J) = A(1,J) +X(J)
        A(2,J) = A(2,J) + X(J)**2
        IF(X(J) -A(3,J)) 42,43,43
    42 A(3,J) = X(J)
    43 IF(X(J) -A(4,J)) 41,41,44
    44 A(4,J) = X(J)
    41 CONTINUE
     5 CONTINUE
     2 CONTINUE
       WRITE(KT3,1010)
 1010 FORMAT(///' GENERAL STATISTICS FOR UNTRANSFORMED DATA',///
      110H  VARIABLE 19X 8HSTANDARD  7X 7HMINIMUM 8X 7HMAXIMUM/9H    NUMBE
      2R 6X 7HAVERAGE 7X 9HDEVIATION 7X 5HVALUE 10X 5HVALUE //)
       DO 10 I = 1,NV
       A(1,I) = A(1,I)/SN
       A(2,I) = SQRT(A(2,I)     /SN -A(1,I)**2)
    10 WRITE(KT3,1005) I,A(1,I),A(2,I), A(3,I),A(4,I)
 1005 FORMAT(I7,4F15.4)
       REWIND KT4
       DO 20 K = 1,NS
       READ(KT4)      (NAME(K,J),J = 1,3),(X(I),I = 1,NV)
       IF (ITRANS - 1) 25,25,30
    25 DO 26 I = 1,NV
    26 X(I) = X(I) / A(4,I)
       GO TO 20
    30 DO 31 I = 1,NV
    31 X(I) = (X(I) - A(3,I))/(A(4,I)- A(3,I))
    20 WRITE(KT5)      (NAME(K,J),J = 1,3),(X(I),I = 1,NV)
       REWIND KT4
       REWIND KT5
       IF(ITRANS - 1) 50,50,51
    50 WRITE(KT3,1050)
 1050 FORMAT(1HO,'VARIABLES TRANSFORMED TO % OF MAXIMUM VALUE',//)
       GO TO 52
    51 WRITE(KT3,1051)
 1051 FORMAT(1HO,  'VARIABLES TRANSFORMED TO % OF THEIR RANGE',//)
    52 CONTINUE
       RETURN
       END


       SUBROUTINE HDIAG(R,M,L,SS,NR,X)
C      ---------------------------------------------------------------------
       DIMENSION R(50,50),SS(50,50),X(50),D2(50),D3(50),D4(50),E(50)
C      ---------------------------------------------------------------------
       CALL HOW(M, 50,M,R,E,SS,X,D2,D3,D4)
       DO 50 J=1,M
       SM=0.0
       DO 30 I=1,M
       SM=SM+SS(I,J)
    30 CONTINUE
       IF (SM)40,50,50
    40 DO 60 I=1,M
    60 SS(I,J)=-SS(I,J)
    50 CONTINUE
       DO 10 I=1,M
    10 R(I,I)=E(I)
       RETURN
       END
```

```
      SUBROUTINE HOW(LP,NM,M,R,E,V,A,B,W1,W2)
C     ------------------------------------------------------------------
      DIMENSION R(2500),E(50),V(2500),A(50),B(50),W1(50),W2(50)
C     ------------------------------------------------------------------
      IF(LP-1)20,14,4
    4 CALL TRIDI(LP,NM,R,A,B,W1,W2)
      CALL EIGVAL(LP,NM,M,E,A,B,W1,W2)
      IF(M)5,9,5
    5 K=IABS(M)
      J=1
      DO 7 I=1,K
      CALL EIGVEC(LP,NM,R,A,B,E(I),V(J),W1,W2)
    7 J=J+NM
    9 NM1=NM+1
      JJ=NM1
      LP2=LP*NM
      DO 12 I=2,LP2,NM1
      K=I
      DO 10 J=JJ,LP2,NM
      R(K)=R(J)
   10 K=K+1
   12 JJ=JJ+NM1
      GO TO 20
   14 E(1)=R(1)
      V(1)=1.0
      A(1)=R(1)
      B(1)=0.0
   20 RETURN
      END

      SUBROUTINE TRIDI(LP,NM,R,A,B,W,Q)
C     ------------------------------------------------------------------
      DIMENSION R(2500),A(50),B(50),W(50),Q(50)
C     ------------------------------------------------------------------
      LP1=LP-1
      LP2=LP1*NM+LP
      LPP=LP2-NM
      NM1=NM+1
      L=0
      DO 10 I=1,LP2,NM1
      L=L+1
   10 A(L)=R(I)
      B(1)=0.0
      IF(LP-2)99,65,15
   15 KK=0
      DO 50 K=2,LP1
      KL=KK+K
      KV=KK+LP
      KJ=K+1
      SUM=0.0
      DO 20 J=KL,KV
   20 SUM=SUM+R(J)**2
      S= SQRT(SUM)
      B(K)= SIGN(S,-R(KL))
      S=1.0/S
      W(K)= SQRT( ABS(R(KL))*S+1.0)
      X= SIGN(S/W(K),R(KL))
      R(KL)=W(K)
      DO 30 I=KJ,LP
      JJ=I+KK
      W(I)=X*R(JJ)
   30 R(JJ)=W(I)
      DO 35 J=K,LP
      JJ=J+1
      Q(J)=0.0
      L=KK+J
      DO 33 I=K,J
      L=L+NM
   33 Q(J)=Q(J)+R(L)*W(I)
      IF(JJ-LP)34,34,36
   34 DO 35 I=JJ,LP
      L=L+I
   35 Q(J)=Q(J)+R(L)*W(I)
   36 X=0.0
      DO 40 J=K,LP
   40 X=X+W(J)*Q(J)
      X=0.5*X
      DO 45 I=K,LP
   45 Q(I)=X*W(I)-Q(I)
      LL=KK
      KK=KK+NM
      DO 50 I=K,LP
      LL=LL+NM
      DO 50 J=I,LP
      L=LL+J
   50 R(L)=R(L)+Q(I)*W(J)+Q(J)*W(I)
      L=1
```

```
      DO 60 I=1,LP
      X=A(I)
      A(I)=R(L)
      R(L)=X
   60 L=L+NM1
   65 R(LP)=R(LPP)
   99 RETURN
      END


      SUBROUTINE EIGVAL(LP,NM,M,E,A,B,F,W)
C
      DIMENSION E(50),A(50),B(50),F(50),W(50)
C
      BD= ABS(A(1))
      DO 5 I=2,LP
      BD=BD+1.0
    5 BD=AMAX1(BD, ABS(A(I))+B(I)**2)
      DO 6 I=1,LP
      A(I)=A(I)/BD
      B(I)=B(I)/BD
      W(I)=1.0
    6 E(I)=-1.0
      DO 50 K=1,LP
C   8 IF((W(K)-E(K))/AMAX1( ABS(W(K)), ABS(E(K)),1.0E-9)-1.0E-7)50,50,10
    8 IF((W(K)-E(K))/AMAX1( ABS(W(K)), ABS(E(K)),1.0E-7)-1.0E-5)50,50,10
   10 X=(W(K)+E(K))*0.5
      IS2=1
      F(1)=A(1)-X
      IF(F(1))102,104,104
  102 IS1=-1
      N=0
      GO TO 105
  104 IS1=1
      N=1
  105 DO 120 I=2,LP
      IF(B(I))106,113,106
  106 IF(B(I-1))107,114,107
C 107 IF( ABS(F(I-1))+ ABS(F(I-2))-1.0E-15)110,112,112
  107 IF( ABS(F(I-1))+ ABS(F(I-2))-1.0E-7 )110,112,112
C 110 F(I-1)=F(I-1)*1.0E15
  110 F(I-1)=F(I-1)*1.0E7
C     F(I-2)=F(I-2)*1.0E15
      F(I-2)=F(I-2)*1.0E7
  112 F(I)=(A(I)-X)*F(I-1)-B(I)**2*F(I-2)
      GO TO 115
  113 F(I)=(A(I)-X)*ISIGN(1,IS1)
      GO TO 115
  114 F(I)=(A(I)-X)*F(I-1)- SIGN(B(I)**2, FLOAT(IS2))
  115 IS2=IS1
      IF(F(I))116,117,116
  116 FIS1= FLOAT(IS1)
      FIS1=(FIS1**2+0.1)/FIS1
      IS1=IFIX(    ( SIGN(FIS1,F(I))))
      IF(IS2+IS1)117,120,117
  117 N=N+1
  120 CONTINUE
      M=LP-N
      IF(M-K)20,12,12
   12 DO 15 J=K,M
   15 W(J)=X
   20 M=M+1
      IF(LP-M)8,24,24
   24 DO 26 J=M,LP
      IF(X-E(J))8,8,26
   26 E(J)=X
      GO TO 8
   50 CONTINUE
      DO 60 I=1,LP
      A(I)=A(I)*BD
      B(I)=B(I)*BD
   60 F(I)=(W(I)+E(I))*BD*0.5
      J=LP
      K=1
      DO 70 I=1,LP
      IF( ABS(F(K))- ABS(F(J)))63,63, 65
   63 E(I)=F(J)
      J=J-1
      GO TO 70
   65 E(I)=F(K)
      K=K+1
   70 CONTINUE
      IF(ISIGN(1,M))75,80,80
   75 DO 77 I=1,LP
   77 F(I)=E(I)
      J=LP
      DO 79 I=1,LP
      E(I)=F(J)
```

```
   79 J=J-1
   80 CONTINUE
      RETURN
      END

      SUBROUTINE EIGVEC(LP,NM,R,A,B,E,V,P,Q)
C
      DIMENSION R(2500),V(2500),A(50),B(50),P(50),Q(50)
C     ------------------------------------------------------------------
      X=A(1)-E
      Y=B(2)
      LP1=LP-1
      DO 10 I=1,LP1
      IF( ABS(X)- ABS(B(I+1)))4,6,8
    4 P(I)=B(I+1)
      Q(I)=A(I+1)-E
      V(I)=B(I+2)
      Z=-X/P(I)
      X=Z*Q(I)+Y
      IF(LP1-I)5,10,5
    5 Y=Z*V(I)
      GO TO 10
    6 IF(X)8,7,8
    7 X=1.0E-10
    8 P(I)=X
      Q(I)=Y
      V(I)=0.0
      X=A(I+1)-(B(I+1)/X*Y+E)
      Y=B(I+2)
   10 CONTINUE
   20 IF(X)21,28,21
   21 V(LP)=1.0/X
   22 I=LP1
      V(I)=(1.0-Q(I)*V(LP))/P(I)
      X=V(LP)**2+V(I)**2
   25 I=I-1
      IF(I)26,30,26
   26 V(I)=(1.0-(Q(I)*V(I+1)+V(I)*V(I+2)))/P(I)
      X=X+V(I)**2
      GO TO 25
   28 V(LP)=1.0E10
      GO TO 22
   30 X= SQRT(X)
      DO 31 I=1,LP
   31 V(I)=V(I)/X
      J=LP1*NM-NM
      K=LP
      GO TO 42
   32 K=K-1
      J=J-NM
      Y=0.0
      DO 35 I=K,LP
      L=J+I
   35 Y=Y+V(I)*R(L)
      DO 40 I=K,LP
      L=J+I
   40 V(I)=V(I)-Y*R(L)
   42 IF(J)32,44,32
   44 RETURN
      END

       SUBROUTINE VARMAX(MAXF,MAXT,NV,TITLE,F,T,COM,NAME,NROT,IPUN)
C
C     VARIMAX MATRIX ROTATION
C
      DIMENSION F(1500,10),COM(1500),VAR(10),CUM(10),          TITLE(20),
     1 NAME(1500,3)
      DIMENSION  T(10,10)
C
C     MAXT = NO. OF SAMPLES, NS = NO. OF VARIABLES,MAXF = NO.OF FACTORS
C     ------------------------------------------------------------------
C
       COMMON KT1,KT2,KT3,KT4,KT5
    1 FORMAT(2I4)
    2 FORMAT(5X 10F7.4 )
      DO 801 I=1,MAXF
      DO 801 J=1,MAXF
      IF(I-J) 803,802,803
  802 T(I,J)= 1.0
      GO TO 801
  803 T(I,J) = 0.0
  801 CONTINUE
      EPS = 0.06993
  150 DO 103 N = 1, MAXT
      COM(N) = 0.0
      DO 102 M = 1, MAXF
  102 COM(N) = COM(N) + F(N,M) * F(N,M)
```

```
      COM(N) = SQRT (COM(N)  )
      DO 103 M = 1, MAXF
  103 F(N,M) = F(N,M)/COM(N)
      L = MAXF - 1
  104 NOROT = 0
      DO 123 M = 1, L
      K = M + 1
      DO 123 MONE = K, MAXF
      A = 0.0
      B = 0.0
      C = 0.0
      D = 0.0
      DO 105 N = 1, MAXT
      U = F(N,M)**2  -F(N,MONE) **2
      V = F(N,M) * F(N,MONE)* 2.
      A = A + U
      B = B + V
      C = C + U ** 2 - V ** 2
  105 D = D + U * V * 2.0
      R = MAXT
      QNUM = D - 2.0 * A * B / R
      QDEN = C - (A ** 2 - B ** 2) / R
      IF(ABS(QNUM) + ABS(QDEN)) 120,120,106
  106 IF(ABS(QNUM) - ABS(QDEN)) 107,114,111
  107 R = ABS(QNUM/QDEN)
      IF(R - EPS) 109,108,108
  108 CS4TH = COS(ATAN(R))
      SN4TH = SIN(ATAN(R))
      GO TO 115
  109 IF(QDEN) 110,120,120
  110 SNPHI = .70710678
      CSPHI = SNPHI
      GO TO 121
  111 R = ABS(QDEN/QNUM)
      IF(R - EPS) 113,112,112
  112 SN4TH = 1.0/ SQRT(1.0 + R ** 2)
      CS4TH = SN4TH * R
      GO TO 115
  113 CS4TH = 0.0
      SN4TH = 1.0
      GO TO 115
  114 CS4TH = .70710678
      SN4TH = CS4TH
  115 R = SQRT((1.0 + CS4TH) * 0.5)
      CSTH = SQRT((1.0 + R) * 0.5)
      SNTH = SN4TH/(4.0 * CSTH * R)
      IF(QDEN) 116,117,117
  116 CSPHI =.70710678*(CSTH+SNTH)
      SNPHI =.70710678*(CSTH-SNTH)
      GO TO 118
  117 CSPHI = CSTH
      SNPHI = SNTH
  118 IF(QNUM) 119,121,121
  119 SNPHI = - SNPHI
      GO TO 121
  120 NOROT = NOROT + 1
      GO TO 1230
  121 DO 123 N = 1, MAXT
      R = F(N,M) * CSPHI + F(N,MONE) * SNPHI
      F(N,MONE) = F(N,MONE) * CSPHI - F(N,M) * SNPHI
      F(N,M)= R
      IF(N-MAXF) 804,804,123
  804 TP = T(N,M)
      T(N,M)= TP * CSPHI + T(N,MONE) * SNPHI
      T(N,MONE) = -TP * SNPHI + T(N,MONE) * CSPHI
 1230 CONTINUE
  123 CONTINUE
      IF(NOROT -(MAXF * L)/2) 104,124,104
  124 DO 125 N = 1, MAXT
      DO 127 M = 1, MAXF
  127 F(N,M) = F(N,M) * COM(N)
  125 COM(N) = COM(N) ** 2
      WRITE(KT3,60) TITLE
   60 FORMAT(1H1, 20A4,///)
      WRITE(KT3,30)
   30 FORMAT(22HOVARIMAX FACTOR MATRIX  //)
      WRITE(KT3,40) (J,J = 1,MAXF)
   40 FORMAT(1HO,22X,'COMM.',4X,10(I5,4X),/ )
      DO 126 N = 1, MAXT
C
C     PUNCH VARIMAX FACTOR MATRIX IF NEEDED
      IF(NROT)126,126,63
   63 IF(IPUN)126,126,64
   64 WRITE(KT2,1004) (NAME(N,JK),JK=1,3),        (F(N,M),M=1,MAXF
 1004 FORMAT(3A4,10F6.3)
  126 WRITE(KT3,20) N,(NAME(N,JK),JK=1,3),COM(N),(F(N,M),M=1,MAXF)
   20 FORMAT(1H ,I5,2X,3A4,11F9.4)
      DO 200 I = 1,MAXF
```

```
      VAR(I) =0.
200   CUM(I) = 0.
       FN = MAXT
      DO 201 I = 1,MAXF
      DO 201 J = 1,MAXT
201   VAR(I) = VAR(I) + F(J,I) * F(J,I)
       DO 500 I = 1,MAXF
500   VAR(I) = (VAR(I)/ FN) * 100.
       CUM(1) = VAR(1)
       DO 202 I = 2, MAXF
202   CUM(I) = CUM(I-1) + VAR(I)
      WRITE(KT3,502) (VAR(I), I = 1,MAXF)
502   FORMAT(1H0,19X,8HVARIANCE,3X,10(F7.3,2X))
      WRITE(KT3,503)(CUM(I), I = 1,MAXF)
503   FORMAT(1H0,19X,8HCUM. VAR,3X,10(F7.3,2X))
      IF(MAXF - 3) 600,600,601
600   WRITE(KT3,60)(TITLE(I), I = 1,15)
        WRITE(KT3,602)
602   FORMAT(1H0,'  NORMALIZED VARIMAX FACTOR COMPONENTS      ')
       WRITE(KT3,40)(J,J = 1,MAXF)
       DO 603 N = 1, MAXT
       DO 603 M = 1,MAXF
      S = F(N,M)
       F(N,M) = (F(N,M) * F(N,M))/CUM(N)
      IF(S) 605,603,603
605   F(N,M) = -F(N,M)
603   CONTINUE
       DO 604 N = 1,MAXT
604   WRITE(KT3,20) N,(NAME(N,JK),JK=1,3),CUM(N),(F(N,M),M=1,MAXF)
601   CONTINUE
100   RETURN
      END
```