ADDENDUM TO:


The missing data programs have been written to accept a uniform format throughout the data matrix. If variable format from deck to deck or within a deck is desired, the following changes in the programs should be made:

\section*{MSTDIA}

\begin{verbatim}
DIMENSION.............,FORM(13),D(200)
  1 READ INPUT TAPE 7,2,N,LX,NTOTAL
  2 FORMAT(3I6)
  9 FORMAT(13A6)
    READ INPUT TAPE 7,FORM,(D(I),I=1,N)
    L=IB+1
    IF(R(I)-D(L)) 25,40,25
    25 SUM(IB)=SUM(IB)+R(I)
\end{verbatim}

\section*{ZIA}

\begin{verbatim}
DIMENSION.............,Z(22000),D(200)
  1 READ INPUT TAPE 7,2,N,LX,NTOTAL,NONE,NTWO,CARDS
  2 FORMAT(5I6,F2.0)
  9 FORMAT(13A6)
    READ INPUT TAPE 7,FORM,(D(I),I=1,N)
    L=IB+1
    IF(R(I)-D(L)) 25,40,25
    25 SUM(IB)=SUM(IB)+R(I)
    IF(SUM(IB)-D(I)) 84,88,84
    84 IF(SQ(IB)-D(I)) 86,88,86
    86 IF(R(IB)-D(I)) 90,88,90
\end{verbatim}

In above two programs D(200) means that a maximum of 200 variables can be used. This number may be changed if desired, e.g., D(100) or D(300).
Addendum to:
"The JRP Statistical Programs for High Speed Digital Computers",
September, 1961

MIA, AIA, and AIB

```
DIMENSION ..........., FORM(13), D(190)
    (Note: D(200) in AIA and AIB)
    - - -
1 READ INPUT TAPE 7,2,N,LY,LX
2 FORMAT(316)
    - - -
8 FORMAT(13A6)
    READ INPUT TAPE 7, FORM,(D(I), I=1,N)
    DO 32 LA=1,LX
    - - -
12 - - -
   IF(A(I,K)-D(I)) 17,20,17
17 - - -
    (Note: the format for a particular variable cannot be varied from group to group)
```

RIA

```
DIMENSION ..........., FORM(13), D(100)
1 READ INPUT TAPE 7,2,N,NFIRST,NLAST,NEND,COUNT,TVALUE,CARD
2 FORMAT(4I6,F6.0,F4.2,F2.0)
    - - -
3 FORMAT(13A6)
    READ INPUT TAPE 7, FORM,(D(I), I=1,N)
    LG=(JUMP/MFIRST)
    - - -
36 - - -
   IF(A(I,K)-D(I)) 38,42,38
38 IF(A(L,K)-D(L)) 40,42,40
    - - -
```

RIB

```
DIMENSION ..........., FORM(13), D(200)
1 READ INPUT TAPE 7,2,N,NBEGIN,NEND,LY,COUNT,TVALUE,CARD
```

The other changes are the same as for RIA.
CONTROL AND FORMAT CARDS

The control cards are prepared as indicated in the book. "D" in control card #1 is not punched.

The format statement card is punched according to instructions given on pages 13 - 14.

Immediately following the format statement card "missing data statement cards" have to be added (the number of cards has to correspond to the number of data cards per subject). These cards are punched in the same format and columns as the data cards, giving the numbers (e.g., +8888 or +777) used to identify missing data.
THE UNIVERSITY OF MICHIGAN
AND
YPSILANTI STATE HOSPITAL
Schizophrenia and Psychopharmacology
Joint Research Project

THE JRP STATISTICAL PROGRAMS
FOR
HIGH SPEED IBM DIGITAL COMPUTERS

B. K. Radhakrishnan and Nils B. Mattsson
with the assistance of Norman A. Starr

ORA Project 04176

Supported by USPHS grants MY-1972 and MY-1971
Principal investigator: Dr. Ralph W. Gerard

administered through:

OFFICE OF RESEARCH ADMINISTRATION ANN ARBOR

September 1961
TABLE OF CONTENTS

Appendices A through J contain (a) the specific information needed for the use of the respective programs and (b) program print-outs for the 8k memory computer.

<table>
<thead>
<tr>
<th>Acknowledgments</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II. General comments</td>
<td>3</td>
</tr>
<tr>
<td>III. Self-adjusting means and standard deviations program</td>
<td>7</td>
</tr>
<tr>
<td>Appendix A</td>
<td>10</td>
</tr>
<tr>
<td>IV. Self-adjusting standard scores ('z') program</td>
<td>25</td>
</tr>
<tr>
<td>Appendix B</td>
<td>28</td>
</tr>
<tr>
<td>V. Programs for testing significance of difference in mean values</td>
<td>41</td>
</tr>
<tr>
<td>1. t-test program</td>
<td>41</td>
</tr>
<tr>
<td>Appendix C</td>
<td>44</td>
</tr>
<tr>
<td>2. One way analysis of variance program</td>
<td>55</td>
</tr>
<tr>
<td>Appendix D</td>
<td>59</td>
</tr>
<tr>
<td>VI. Correlation programs</td>
<td>77</td>
</tr>
<tr>
<td>1. Inter-correlation program</td>
<td>77</td>
</tr>
<tr>
<td>Appendix E</td>
<td>82</td>
</tr>
<tr>
<td>2. Cross-correlation program</td>
<td>98</td>
</tr>
<tr>
<td>Appendix F</td>
<td>101</td>
</tr>
<tr>
<td>3. R-completion program</td>
<td>112</td>
</tr>
<tr>
<td>Appendix G</td>
<td>114</td>
</tr>
<tr>
<td>VII. Inversion of symmetric matrices and estimation of communalities</td>
<td>125</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Appendix H</td>
<td>130</td>
</tr>
<tr>
<td>VIII. Factor analysis program (principal components)</td>
<td>137</td>
</tr>
<tr>
<td>Appendix I</td>
<td>145</td>
</tr>
<tr>
<td>IX. Orthogonal rotation programs</td>
<td>153</td>
</tr>
<tr>
<td>Appendix J</td>
<td>159</td>
</tr>
<tr>
<td>X. Appendix K</td>
<td>167</td>
</tr>
<tr>
<td>Test data</td>
<td>167</td>
</tr>
<tr>
<td>Test data results</td>
<td>169</td>
</tr>
<tr>
<td>XI. References</td>
<td>173</td>
</tr>
<tr>
<td>XII. Supplement</td>
<td>175</td>
</tr>
<tr>
<td>Program print-outs for a 32k memory computer</td>
<td>175</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

The authors wish to express their sincere thanks to Mr. Ray Adem who drew all the figures, to Miss Alfreda Cole for her help in preparing the manuscript, and to Mr. Alex Bennett and Mr. Sam Adem for checking out computer output.
I. INTRODUCTION

The statistical computer programs introduced in this publication were written for the large scale statistical analyses performed at the Schizophrenia and Psychopharmacology Joint Research Project. The need for programs capable of handling several hundred variables at one time has grown considerably as more and more research establishments have become involved in multidisciplinary studies. The authors, therefore, have undertaken to make this group of programs, that could be regarded as being of general interest, as flexible as possible. The computer used by the Project is the IBM 704 at The University of Michigan. However, as indicated below in Section II, these programs, being general, can be used on any IBM 704 or 709 computer.

All programs, except matrix problems, are divided into two categories

1. Missing data programs

2. No missing data programs

for efficiency in performance and for accomplishment of some optional features discussed elsewhere in this publication.

The machine language used is FORTRAN. Test problems are provided in the appendices to aid users who are not familiar with this language. In case additional information is needed the authors will be glad to furnish it. The address is: Joint Research Project, Box A, Ypsilanti, Michigan; telephone: HUnter 26404.
II. GENERAL COMMENTS

1. The input sub-routine reads in the FORMAT of the data cards. The programs therefore are capable of handling data with any FORMAT. Exceptions are the R-Completion, Inversion, and Varimax programs which require a new format statement card in the Fortran deck if format change is desired.

2. The results are printed out to 3-decimal place accuracy. During computation the words are rounded to the eighth decimal place. The results can be printed out to more places if desired.

3. The changes that are likely to occur in using these programs elsewhere, either on the IBM 704 or the 709, are expected to be very slight. In addition to possible changes in calling sequence for sub-routines and in dimension cards, the tape numbers assigned to the different sections of the programs may call for changes. For example, tape 6 is exclusively designated as an output tape on The University of Michigan IBM 704 Computer, and cannot therefore be used as a scratch tape. It is only necessary to change the tape numbers, or set the control dials on the tape units accordingly.

4. Running time estimates are provided wherever possible. In general, running-time will be much more affected by number of variables than by number of observations. So far, the relation between number of variables and number of observations with respect to running time has not been calculated. Any information on actual running times
will be appreciated. All time estimates given are for the IBM 704 (8K).

5. Arrangement and order of punched cards is indicated below. See Fig. 4, Appendix E.
   a. Control cards
   b. Format card
   c. Data deck
   d. Job number card

6. Several jobs can be processed at one loading of a program.

7. The format for missing data entry in the control cards of all missing data programs is F6.3 which means sign (+ or -), 4 digits to the left and 3 digits to the right of the decimal point. This is the maximum number of significant digits which can be used, if the decimal point is not punched. Any number not used as actual data can be punched to indicate missing data. For example if the data format is
   a. F6.3, missing data will be punched in the control card +9999999, or +8888888, etc.
   b. F4.2, the control card would read +000990, or +000770, etc.

In the data cards missing data would be punched correspondingly +9999999, or +8888888, etc. in the case of F6.3. In the case of F4.2 the entry would be +999, or +777, etc.

When a data format smaller than F6.3 is used the number indicating missing data entry may not exactly correspond to an octal number. In this case it is advisable to punch the exact number for the missing
data entry in the control card including the decimal point. For example

\begin{center}
\begin{tabular}{|c|c|}
\hline
F3.2 & data entry \\
\hline
 & \pm xxxx (e.g. \(+0.356\)) \\
missing data entry & \(\pm 9999\) \\
or & \(+8888\) \\
& etc. \\
& \begin{align*}
control card & \pm 0.000.000 \\
or & \pm 000.000 \\
& etc.
\end{align*}
\hline
\end{tabular}
\end{center}

9. Test data for \(y\) variables and 56 observations are given in appendix K. This information may be used when compiling the programs and to check the output. Sample answers are also provided in appendix K.

9. Included in the references are Anderson (1) and Rao (13). These publications have not been referred to anywhere in the text, although consulted for formulas and matrix theory.

Disclaimer

Although all programs and the special features incorporated in them have been tested by the authors, no warranty, express or implied, is made by the authors as to the accuracy and functioning of the programs and related program material and no responsibility is assumed by the authors in connection therewith.

The authors would appreciate notification of possible discrepancies found by the users of these programs.

Note

The users of the programs are urged to inquire at their respective computing centers about special symbols required in program or other cards (for example control
cards), IBM card columns where to start punching the cards, special cards (for instance an END card) to be included in the program deck, calling sequence for sub-routines, etc.
III. SELF-ADJUSTING MEANS AND STANDARD DEVIATIONS PROGRAM

A. DESCRIPTION OF THE PROGRAM

1. The program is classified into two categories

   a. MSTD1A (missing data)
   b. MSTD2A (no-missing data)

2. Program print-out (see Appendix A)

3. Special code built into program

   To avoid stoppage due to any unforeseen circumstances, the following feature is incorporated. Whenever the variance is exactly zero the program prints out

   \[ \sigma_x = +999.999 \]

   \[ \bar{X} = +999.999 \]

B. SPECIAL FEATURES

1. The input sub-routine reads in the FORMAT of the data cards. The program therefore is capable of handling data with any FORMAT.

2. The program has been so designed that it can be used efficiently for various sizes of matrices. One compilation of the program is sufficient, because the program adjusts the storage space available for data depending on the number of variables included.

3. Once a matrix of data has been punched on cards, the program can, without rearrangement of the matrix, do calculations on
a. The full matrix
b. One or more specified variables
c. One or more sequences of variables

The program can be made to skip calculations on variables which are not to be included. This feature will thus save card sorting and computing time.

C. DISCUSSION

1. Mean:

\[
\bar{X} = \frac{\sum_{i=1}^{N} x_i}{N}
\]

2. Standard deviation

\[
\sigma_X = \sqrt{\frac{N \sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2}{N(N-1)}}
\]

In the case of the missing data program, \(N, \sum X, \sum x^2\) refer only to observations present. The particular calculation loop is skipped whenever missing data are detected by the program.

D. LIMITATIONS

<table>
<thead>
<tr>
<th>Program Category</th>
<th>MAXIMUM NUMBER OF VARIABLES ON AN 8k MEMORY</th>
<th>MAXIMUM NUMBER OF VARIABLES ON A 32k MEMORY</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTD 1A</td>
<td>1400 *</td>
<td>5000 *</td>
</tr>
<tr>
<td>MSTD 2A</td>
<td>1400 *</td>
<td>5000 *</td>
</tr>
</tbody>
</table>

*These are estimates
If the data matrix is very large, say, several hundred variables, it would be advisable to divide it into two or three smaller matrices, and run the jobs on one loading. This can be done because the variables are not interdependent, and it will reduce running time appreciably. As to the number of observations on each variable the storage capacity of the tape unit is the only limiting factor.

E. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Computing time per $({\bar{X}}$ and $\sigma$) is approximately 1 to 3 seconds, depending on number of variables and observations. For example, a matrix of 200 variables and 300 observations requires close to 3 seconds per $({\bar{X}}$ and $\sigma$).

F. TEST PROBLEMS

The test problems have been designed so that they cover possible occurrences, and they are itemized for categories of programs discussed above (MSTDLA, MSTD2A). See examples in Appendix A.

G. OUTPUT

Print-out: Variable no., $\sigma$, $\bar{X}$, $N$ (reduced)
Appendix A

1. ARRANGEMENT OF CARDS (see Appendix E under inter-correlation programs)
   a. Using fortran deck. See Fig. 4.
   b. Using binary deck. See Fig. 4.

2. EXAMPLE PROBLEMS

Means and standard deviations missing data program (MSTD 1A) and
Means and standard deviations no-missing data program (MSTD 2A)

<table>
<thead>
<tr>
<th>Example*</th>
<th>Number of variables</th>
<th>Number of observations**</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>280</td>
<td>369</td>
</tr>
<tr>
<td>B</td>
<td>185</td>
<td>206</td>
</tr>
<tr>
<td>C</td>
<td>96</td>
<td>100</td>
</tr>
<tr>
<td>D</td>
<td>31</td>
<td>100</td>
</tr>
</tbody>
</table>

*See Fig. 1

**The number of observations could be very much larger, say 2,000

Missing data are also to be counted as observations, although during computation the program omits counting missing data.

3. DIMENSION CARDS FOR THE FORTRAN DECK

The program has been written so that there is no need to compile for different dimensions. However, new dimension cards have to be punched in case of a 32k memory.
4. NSTORE = **** CARD: (***) refers to number of digits punched. It means that the core storage available for data is a four digit number on an 8k memory.)

See program print-out in this appendix. A new card is required only if a new compilation is made. This would be the total storage space available for data. The dimension cards and the NSTORE = **** card have been set for an 8k memory in the program print-out.

5. CONTROL CARDS

Means and standard deviations missing data program (MSTD 1A) and Means and standard deviations no-missing data program (MSTD 2A)

a. Control card no. 1:

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-6</td>
</tr>
<tr>
<td>N</td>
<td>LY</td>
</tr>
<tr>
<td>A</td>
<td>+00280</td>
</tr>
<tr>
<td>B</td>
<td>+00185</td>
</tr>
<tr>
<td>C</td>
<td>+00096</td>
</tr>
<tr>
<td>D</td>
<td>+00031</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 1.

2. D refers to missing data entry on cards.

3. Columns 19-26 entry can be left blank in the case of no missing data program.

4. Note the entries for LY. See corresponding sketches in Fig. 1 and also control cards 2 and 3.
5. \( N = \text{total number of variables in the punched matrix} \)

\( \text{LY} = \text{number of sequences of variables (including the case of single variables) to be calculated} \)

\( \text{NTOTAL} = \text{total number of observations (including missing data)} \)

6. The missing data entry in the example is given for data format F5.3.

b. Control card no. 2

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>001</td>
</tr>
<tr>
<td>B</td>
<td>025</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 1

2. The starting point in the matrix is indicated on this card.

Also note, in example D, LY has an entry 3 in control card no. 1. Correspondingly three starting variables are indicated on control card no. 2.
c. Control card no. 3

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>280</td>
</tr>
<tr>
<td>B</td>
<td>179</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 1

2. The last variable to be computed is indicated on this card.

3. The program is designed for a maximum value of LY = 24. However, the program can be compiled for a larger value of LY. The desired number has to be entered in the dimension card before compilation.

6. FORMAT STATEMENT

The sub-routine reads in the format of the data cards as input-data. This makes it possible to handle data of different formats.

Example 1. 9F8.3: i.e., there are 9 variables punched on each card, each variable occupying 8 columns including sign. The decimal point is not punched in the data cards, and no space is left blank for it. (If the decimal point is punched, it overrides the format statement.) There are 4 digits to the left, and 3 to the right of the decimal point (e.g. +0471.535, or +1263.000, or -0000.350).
Example 2. 36F2.0: 36 variables per card, data consists of: sign, 1 digit, no decimal; 24F3.0: 24 variables per card: sign, 2 digits, no decimal. Thus the format card contains starting from column 1: 2F8.3 or 36F2.0, etc. depending on the format of the data cards. Note: the decimal point has to be punched in the format statement card.

Example 3. Format change from card to card and columns not to be read in. If the format card, for example, is punched the following way starting in column 1:

10X,30F2.0/15F4.1/9F3.0,5X,9F3.0

it means that there are three successive data cards per subject, card 1 containing 30 variables of format ±X, card 2 containing 15 variables of format ±XXX, card 3 containing 9 + 9 = 18 variables of format ±XX. 10X and 5X indicate number and position of columns to be skipped. Thus the computer is "told" to read in the data cards as follows:

Data card 1. Do not read first ten columns. Read 30 variables, each occupying two columns (e.g. +5 or +0).

Data card 2. Read 15 variables, each occupying four columns (e.g. -03.4 or +12.4). The decimal point is not punched.

Data card 3. Read 9 variables, each occupying three columns (e.g. +26 or -09). Do not read next five columns. Read 9 variables, each occupying three columns (e.g. -11, or +07).

7. DATA CARDS

The observed values on the variables punched in the card for one subject have to be punched in the same columns for each successive subject. The
cards completed for all subjects form a deck which should also be identified by a deck number, usually punched in the last two or three columns. The subjects should also be identified by number, punched usually to the left of the deck number. A good practice is to reserve column numbers 73 - 80 for identification. If there are variables to fill another, a third, etc., card per subject, a second, third, etc. deck is punched. The format for the missing data program should be the same in all decks. This can be arranged by properly spacing the observed data. This arrangement will provide for identical missing data entries in all decks. However, in the no-missing data program, variable format is permissible. Columns which are not used may be left blank. Very often the last deck does not contain the same number of variables as the other decks, depending on total number of variables. If the same format is stated (see format card) for all decks, e.g., $9F^8.3$ which means 9 variables per card punched with fixed decimal point between the third and fourth digit from the right, the control card will prevent the computer from reading beyond the last variable, say only 4, in the last deck. The data cards have to be sorted so that cards from deck 1, 2, 3, etc., for the first subject, are followed by cards from deck 1, 2, 3, etc., for the second subject, etc. The computer thus reads all variables for subject one, then subject two, etc. In other words, the cards have been sorted in matrix form, variables horizontally, and subjects (observations) vertically. Note that number of observations has to be the same for all variables in
the input cards. If this number varies, for instance in the case of several groups of observations, there are two ways to proceed:

a. Arrange the data cards into two or more matrices, and run them as separate jobs on one loading.

b. Punch "missing data" to "fill up" those variables which have fewer observations. Blank words in the data cards will be read as zero values by the computer.

8. JOB NUMBER CARD

This is the last card of the entire input. Punch any numerical code (e.g. job sequence number) in the first 10 columns of the card. (Example: +000006121, or +000000025). This number will print on top of the output and means that all cards in the analysis have been read into the computer.
FIG 1
*COMPILE FORTRAN, PRINT SAP, PUNCH OBJECT, EXECUTE, DUMP
MSTD1A0
DIMENSION SUM(5850), SQ(5850), COUNT(5850), R(5850), NX(5850), NY(5850)
EQUIVALENCE (R, SUM), (R, COUNT), (R, SQ), (R, NX), (R, NY)
1 READ INPUT TAPE 7, 2 * N, 5 * L, 4 * N, 4 * D
2 FORMAT(316, F8.3)
   NSTORE=5850
   K1=2
   IA=LY
   IB=IA+LY
   READ INPUT TAPE 7, 3 * (NX(I), I=1, IA)
3 FORMAT(2413)
   L=IA+1
   READ INPUT TAPE 7, 3 * (NY(I), I=L, IB)
   N1=0
   DO 5 LA=1, LY
      IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
   IA=LY
   IC=IB+N1
   ID=IC+N1
   IE=ID+N1
   IF=NSTORE-IE
   MFIRST=IF/N
   M1=MFIRST
   M=IE+N
   L=IE+1
   LG=NTOTAL/MFIRST
   MLAST=MFIRST*LG
   IF (NTOTAL-MLAST) 7, 6, 7
6 MLAST=MFIRST
   GO TO 8
7 LG=LG+1
   MLAST=NTOTAL-MLAST
8 CALL INPUT
   READ INPUT TAPE 7, 9 * (R(I), I=L, M)
9 FORMAT(1)
   1 L=IB+1
   DO 14 J=L, IE
14 COUNT(J)=0; 0
   DO 40 J=1, LG
      IA=LY
      IB=IA+LY
      IC=IB+N1
      ID=IC+N1
      IF (LG-J) 15, 15, 16
15 M1=MLAST
   DO 20 KA=K1, M1
      K=M
      K=K+1
      M=M+N
      READ INPUT TAPE 7, 9 * (R(I), I=K, M)
20 CONTINUE
   K1=1
   M=IE
   DO 40 LA=1, LY
      IA=IA+1
   40 MA=NX(LA)
   MB=NY(IA)
DO 40 L=MA,MB
  I=IE-N+L
  IB=IB+1
  IC=IC+1
  ID=ID+1
DO 40 KA=1,M1
I=I+N
IF (R(I)=D) 25,40,25
25 SUM(IB)=SUM(IB)+R(I)
  SQ(IC)=SQ(IC)+R(I)**2
  COUNT(ID)=COUNT(ID)+1
CONTINUE
READ INPUT TAPE 7,41,I
FORMAT(I10)
WRITE OUTPUT TAPE 6,42,I
FORMAT(44H1 MEAN AND SIGMA PROGRAM JOB NUMBER I10)
  IA=LY
  IB=IA+LY
  IC=IB+N1
  ID=IC+N1
  I=29
DO 70 LA=1,LY
  IA=IA+1
  MA=NX(LA)
  MB=NY(LA)
DO 70 L=MA,MB
  IB=IB+1
  IC=IC+1
  ID=ID+1
  CA=(COUNT(ID)*(COUNT(ID)-1.))
  CB=(COUNT(ID)*SQ(IC))-(SUM(IB)**2)
  IF (CB) 45,45,56
  SQ(IC)=+999,999
  IF (COUNT(ID)) 50,50,54
50 SUM(IB)=+999,999
GO TO 58
54 SUM(IB)=SUM(IB)/COUNT(ID)
GO TO 58
56 SQ(IC)=(SQRT(CB/CA))
  SUM(IB)=SUM(IB)/COUNT(ID)
58 I=I+1
IF (I-30) 65,60,70
60 WRITE OUTPUT TAPE 6,61
61 FORMAT(64H1) VARIABLE STANDARD DEVIATION MEAN
  1 NUMBER
  I=0
65 WRITE OUTPUT TAPE 6,66,L,SQ(IC),SUM(IB),COUNT(ID)
66 FORMAT(IH0I14,F19.3,F20.3,F9.0)
70 CONTINUE
GO TO 1
*ASSEMBLE,PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END,0,1
PZE
BCD 1INPUT
PZE ENT

19
REM
ORG 0
REL
ERROR RCD 1ERROR
ENT RTD 7
CLA 7$4
STA CPY
SXD AX1$1
LXA A=1$1
CPY CPY **$1
D=12 TXI *+3$1$1
TSX ERROR$4
TSX ERROR$4
TXL CPY$1$112
LXD AX1$1
TRA 1$4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
*COMPILE FORTRAN, PRINT SAP, PUNCH OBJECT, EXECUTE, DUMP
DIMENSION SUM(5875), SQ(5875), R(5875), NX(5875), NY(5875)
EQUIVALENCE (R, SUM), (R, SQ), (R, NX), (R, NY)
1 READ INPUT TAPE 7, 2, N, LY, NTOTAL
2 FORMAT(3I6)
NSTORE=5875
COUNT=NTOTAL
K1=2
IA=LY
IB=IA+LY
READ INPUT TAPE 7, 3, (NX(I), I=1, IA)
3 FORMAT(24I3)
L=IA+1
READ INPUT TAPE 7, 3, (NY(I), I=1, IB)
N1=0
DO 5 LA=1, LY
IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
IA=LY
IC=IB+N1
IE=IC+N1
IF=NSTORE-IE
MFIRST=IF/N
M1=MFIRST
M=IE+N
L=IE+1
LG=NTOTAL/MFIRST
MLAST=MFIRST*LG
IF (NTOTAL-MLAST) 7, 6, 7
6 MLAST=MFIRST
GO TO 8
7 LG=LG+1
MLAST=NTOTAL-MLAST
8 CALL INPUT
READ INPUT TAPE 7, 9, (R(I), I=1, M)
9 FORMAT(
1   )
L=IB+1
DO 12 J=L, IE
12 SQ(J)=0.0
DO 40 J=1, LG
IA=LY
IB=IA+LY
IC=IB+N1
IF (LG-J) 15, 15, 16
15 M1=MLAST
16 DO 20 KA=K1, M1
K=M
K=K+1
M=M+N
READ INPUT TAPE 7, 9, (R(I), I=K, M)
20 CONTINUE
K1=1
M=IE
DO 40 LA=1, LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
DO 40 L=MA*MB
40 CONTINUE
I=IE-N+L
IB=IB+1
IC=IC+1
DO 40 KA=1*M1
I=I+N
25 SUM(IB)=SUM(IB)+R(I)
SQ(IC)=SQ(IC)+R(I)**2
40 CONTINUE
READ INPUT TAPE 7,41,IA
41 FORMAT(110)
WRITE OUTPUT TAPE 6,42,IA
42 FORMAT(44H1 MEAN AND SIGMA PROGRAM JOB NUMBER 110 )
IA=LY
IB=IA+LY
IC=IB+N1
I=29
CA=(COUNT*(COUNT-1.))
DO 70 LA=1,LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
DO 70 L=MA,MB
IB=IB+1
IC=IC+1
CB=(COUNT*SQ(IC))-(SUM(IB)**2)
IF (CB) 45,45,56
45 SQ(IC)=+999.999
SUM(IB)=SUM(IB)/COUNT
GO TO 58
56 SQ(IC)=(SQRT(CB/CA))
SUM(IB)=SUM(IB)/COUNT
58 I=I+1
IF (I=30) 65,60,70
60 WRITE OUTPUT TAPE 6,61
61 FORMAT(64H1 VARIABLE STANDARD DEVIATION MEAN
1 NUMBER)
I=0
65 WRITE OUTPUT TAPE 6,66,L,SQ(IC),SUM(IB),COUNT
66 FORMAT(1H0114,F19.3,F20.3,F9.0)
70 CONTINUE
GO TO 1
*ASSEMBLE,PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END,0+1
PZE
BCD 1INPUT
PZE ENT
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7,4
STA CPY
SXD AX1,1
LXA A=1,1
CPY
D=12
TXI
TSX ERROR*4
TSX ERROR*4
TXL CPY*1,12
LXD AX1*1
TRA 1,4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
IV. SELF-ADJUSTING STANDARD SCORES ('z') PROGRAM

A. DESCRIPTION OF THE PROGRAM

1. The program is classified into two categories.
   a. Z 1A (missing data)
   b. Z 2A (no-missing data)

2. Program print-out (See Appendix B)

3. Special codes built into the program
   To avoid stoppage due to any unforeseen circumstances, the following features are incorporated.
   a. Whenever the variance is exactly zero the program prints out
      $z_x = +9.999$
   b. Whenever the program detects a missing observation, the program prints out for that particular observation
      $z = +9.999$

B. SPECIAL FEATURES

See means and standard deviations programs.

C. DISCUSSION

1. Mean

$$\bar{X} = \frac{\sum_{i=1}^{N} X_i}{N}$$
2. Standard deviation

\[ \sigma_X = \sqrt{\frac{N \sum x_i^2 - (\sum x_i)^2}{N(N-1)}} \]

3. 'z' Score

\[ z = \frac{(X - \bar{X})}{\sigma_X} \]

Note: in the case of the missing data program N, \( \sum X \), \( \sum X^2 \) refer only to observations present. The particular calculation loop is skipped whenever missing data are detected by the program.

D. LIMITATIONS

<table>
<thead>
<tr>
<th>Program Category</th>
<th>MAXIMUM NUMBER OF VARIABLES ON an 8k memory</th>
<th>a 32k memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z 1A</td>
<td>1200*</td>
<td>4500*</td>
</tr>
<tr>
<td>Z 2A</td>
<td>1200*</td>
<td>4500*</td>
</tr>
</tbody>
</table>

*These are estimates.

If the data matrix is very large, say, several hundred variables, it would be advisable to divide it into two or three smaller matrices, and run the jobs on one loading. This can be done because the variables are not interdependent, and it will reduce running time appreciably. As to the number of observations on each variable the storage capacity of the tape unit is the only limiting factor.
E. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Computation time per 
z is .01 to .03 seconds. For example a matrix of 200 variables and 300 
observations requires .027 seconds per z.

F. TEST PROBLEMS

The test problems have been designed so that they cover possible occurrences, 
and they are itemized for categories of programs discussed above (Z1A, Z2A).

G. OUTPUT

1. Print-out: variable no., σ, $\bar{X}$, N (reduced), and cards: $z_{ij}$ with deck 
   and observation identification, or

2. Print-out: variable no., σ, $\bar{X}$, N (reduced), and $z_{ij}$ with deck and 
   observation identification.
Appendix B

1. ARRANGEMENT OF CARDS

(See Appendix E under intercorrelation programs)

a. Using Fortran deck (Fig. 4)

b. Using binary deck (Fig. 4)

2. EXAMPLE PROBLEMS

Standard Scores missing data program (Z 1A) and

Standard Scores no-missing data program (Z 2A)

<table>
<thead>
<tr>
<th>Example</th>
<th>NUMBER OF VARIABLES</th>
<th>NUMBER OF OBSERVATIONS</th>
<th>OBSERV. NO.</th>
<th>DECK NO.</th>
<th>CARD OUTPUT</th>
<th>PRINT OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>280</td>
<td>309</td>
<td>001</td>
<td>001</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>B</td>
<td>165</td>
<td>206</td>
<td>101</td>
<td>021</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>C</td>
<td>96</td>
<td>200</td>
<td>201</td>
<td>101</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>D</td>
<td>31</td>
<td>100</td>
<td>050</td>
<td>010</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>

Note: a. The number of observations could be very much larger, say 2000. Missing data are also to be counted as observations, although during computation the program omits counting missing data.

b. See Fig. 2.
3. **DIMENSION CARD FOR THE FORTRAN DECK**

The programs have been written so that there is no need to compile for different dimensions. However, new dimension cards have to be punched in case of a 32k memory.

4. **NSTORE = **** CARD**

See program print out. A new card is required only if a new compilation is made. This would be the total storage space available for data. The dimension cards and the **NSTORE = **** card** have been set for an 8k memory in the program print-out.

5. **CONTROL CARDS**

Standard Scores missing data program (Z 1A) and

Standard Scores no-missing data program (Z 2A)

a. Control card no. 1

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-0</td>
</tr>
<tr>
<td>N</td>
<td>LY</td>
</tr>
<tr>
<td>A</td>
<td>+00280</td>
</tr>
<tr>
<td>B</td>
<td>+00160</td>
</tr>
<tr>
<td>C</td>
<td>+00070</td>
</tr>
<tr>
<td>D</td>
<td>+00031</td>
</tr>
</tbody>
</table>

**Note:** NONE refers to row identification of the input matrix (subject or observation number).

NTWO refers to deck identification of the input matrix.
In example A observation no. 1, 2, ..., 309 is punched in columns 75-77, and deck no. 1, 2, ..., n in columns 78-80 of the output cards.

In example B observation numbers will start with 101, and deck numbers with 021. The increment is always = 1.

b. Control card no. 2

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>001</td>
</tr>
<tr>
<td>B</td>
<td>025</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 2

2. See Note 2 under means and standard deviations program, control card no. 1.

3. See Note 3 under means and standard deviation program, control card no. 1.

4. Note the entries for LY. See corresponding sketches in Fig. 2, and also control cards 2 and 3.

5. See Note 5 under means and standard deviations program, control card no. 1.

6. + 1 entry in columns 31-32 calls for card out-put (see IV. G. 1)

- 1 entry in columns 31-32 calls for print-out (see IV. G. 2)
c. Control card no. 3

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>280</td>
</tr>
<tr>
<td>B</td>
<td>179</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 2

2. The last variable to be computed is indicated on this card.

6. FORMAT CARD

See format card under means and standard deviations program.

7. DATA CARDS

See data cards under means and standard deviations program.

8. JOB NUMBER CARD

See job number card under means and standard deviations program.
FIG. 2
*COMPILE FORTRAN* PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP
DIMENSION SUM(5325),SQ(5325),COUNT(5325),R(5325),NX(5325),NY(5325)
I*Z(5325)
EQUIVALENCE (R,*SUM),(R,*COUNT),(R,*SQ),(R,*NX),(R,*NY),(R,*Z)
1 READ INPUT TAPE 7*2,N*LY,NTOTAL,NONSEC,TWO,CARDS,D
2 FORMAT(5I6,F2.0,F8.3)
NSTORE=5325
REWIND 2
K1=2
NPerc=9
IA=LY
IB=IA+LY
READ INPUT TAPE 7*3,(NX(I),I=1,IA)
3 FORMAT(24I3/)
L=IA+1
READ INPUT TAPE 7*3,(NY(I),I=L,IB)
N1=0
DO 5 LA=1,LY
IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
IA=LY
IC=IB+N1
ID=IC+N1
IE=ID+N1
IF=NSTORE-IE
MFIRST=IF/N
M1=MFIRST
LB=N*M1+IE
M=E+N
L=IE+1
LG=NTOTAL/MFIRST
MLAST=MFIRST*LG
IF(NTOTAL-MLAST)7*6,7
6 MLAST=MFIRST
GO TO 8
7 LG=LG+1
MLAST=NTOTAL-MLAST
8 CALL INPUT
READ INPUT TAPE 7*9,(R(I),I=L,M)
9 FORMAT(1)
L=IB+1
DO 14 J=L,IE
14 COUNT(J)=0,0
DO 40 J=1,LG
IA=LY
IB=IA+LY
IC=IB+N1
ID=IC+N1
IF((LG-J))15,15,16
15 M1=MLAST
LB=N*M1+IE
16 DO 20 K1=K1,M1
K=M
K=K+1
M=M+N
READ INPUT TAPE 7*9,(R(I),I=K,M)
20 CONTINUE
K1=1
M=IE
L1=IE+1
WRITE TAPE 2,(R(I),I=L1,LB)
DO 40 LA=1*LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
DO 40 L=MA*MB
I=IE-N+L
IB=IB+1
IC=IC+1
ID=ID+1
DO 40 KA=1*M1
I=I+N
IF (R(I)=D) 25,40,25
25 SUM(IB)=SUM(IB)+R(I)
SQ(IC)=SQ(IC)+R(I)**2
COUNT(ID)=COUNT(ID)+1
CONTINUE
IA=LY
IB=IA+LY
IC=IB+N1
ID=IC+N1
END FILE 2
REWIND 2
L=IE+NPERC
I=29
READ INPUT TAPE 7,41,MA
41 FORMAT(I10)
WRITE OUTPUT TAPE 6,42,MA
42 FORMAT(40H1,Z) PROGRAM JOB NUMBER I10 )
DO 70 LA=1*LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
DO 70 L=MA*MB
IB=IB+1
IC=IC+1
ID=ID+1
CA=(COUNT(ID)*(COUNT(ID)-1))/
CB=(COUNT(ID)*SQ(IC)-(SUM(IB)**2)
IF (CB) 45*45,56
45 SQ(IC)=D
IF (COUNT(ID)) 50,50,54
50 SUM(IB) =D
GO TO 58
54 SUM(IB)=SUM(IB)/COUNT(ID)
GO TO 58
56 SQ(IC)=(SQRT(CB/CA))
SUM(IB)=SUM(IB)/COUNT(ID)
58 I=I+1
IF (I=30) 60,60,70
60 WRITE OUTPUT TAPE 6,61
61 FORMAT(64H1 VARIABLE STANDARD DEVIATION MEAN
1 NUMBER) I=0
65 WRITE OUTPUT TAPE 6,66,L*SQ(IC),SUM(IB),COUNT(ID)
66 FORMAT(1H0114,F19.3,F20.3,F9.0)
CONTINUE
M1=MFIRST
IA=LY
IB=IA+LY
IE=IB+(3*N1)
M=N*M1+IE
L1=IE+1
N2=None-1
DO 140 J=1*LG
IA=LY
IF (LG-J) 75,75,78
M1=MLAST
IB=IA+LY
IE=IB+(3*N1)
M=N*M1+IE

READ TAPE 2*(R(I),I=L1,M)
DO 140 LA=1*LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
MC=MB-MA+1
LX=MC/NPERC
LZ=NPERC*LX
IF (LZ-MC) 82,80,82

LZ=NPERC
GO TO 83

LX=LX+1
LZ=MC-LZ

IG=-N
DO 95 K=1,M1
IB=2*LY
IC=IB+N1
ID=IC+N1
IG=IG+N
ID=ID+IG
DO 95 I=MA*MB
IE=ID+I
IE=IE+N
IB=IB+1
IC=IC+1
IF (SUM(IB)-D) 84,88,84
IF (SQ(IC)-D) 86,88,86
IF (R(IE)-D) 90,88,90
Z(IE)=+9.999
GO TO 95
Z(IE)=(R(IE)-SUM(IB))/SQ(IC)

CONTINUE
IB=2*LY
IC=IB+(2*N1)
IG=-N
DO 130 K=1,M1
N3=NTWO-1
IG=IG+N
ID=IC+IG
IE=ID+MA-1
IE=IE+N
LE=IE
LC=NPERC
100 N2=N2+1
   DO 120 LB=1,LX
   N3=N3+1
   IF (LX-LB) 105,105,110
105 LC=LZ
110 L=LE+1
   LE=LE+LC
   IF (CARDS) 113,120,115
113 WRITE OUTPUT TAPE 6,114,N2,N3,Z(KA),KA=L,LE
114 FORMAT((1HOI6,I3,9F12.3))
   GO TO 120
115 PUNCH 119,Z(KA),KA=L,LE,N2,N3
119 FORMAT((9F8.3),I5,I3)
120 CONTINUE
130 CONTINUE
140 CONTINUE
   REWIND 2
   GO TO 1

*ASSEMBLE*PUNCH OBJECT
   REM INPUT
   ORG 0
   PGM
   PZE END 0+1
   PZE
   BCD 1INPUT
   PZE ENT
   REM
   ORG 0
   REL
ERROR BCD 1ERROR
ENT RTD 7
   CLA 7+4
   STA CPY
   SXD AX1+1
   LXA A=1+1
CPY CPY **+1
D=12 TXI **+3+1+1
   TSX ERROR+4
   TSX ERROR+4
   TXL CPY+1+12
   LXD AX1+1
   TRA 1+4
A=1 PZE 1
AX1 BSS 1
END SYN *
END

*DATA
*COMPILE  JRAN PRINT SAP PUNCH OBJECT EXECUTE DUMP

DIMENSION SUM(5400), SQ(5400), R(5400), NX(5400), NY(5400), Z(5400)
EQUIVALENCE (R, SUM), (R, SX), (R, NX), (R, NY), (R, Z)
1 READ INPUT TAPE 7, 2, NLY, NTOTAL, NONE, NTWO, CARDS
2 FORMAT(516, F2.0)
   COUNT=NTOTAL
   NSTORE=5400
   REWIND 2
   K1=2
   MPRC=9
   IA=LY
   IB=IA+LY
   READ INPUT TAPE 7, 3, (NX(I), I=1, IA)
3 FORMAT(2413)
   L=IA+1
   READ INPUT TAPE 7, 3, (NY(I), I=1, IB)
   N1=0
   DO 5 LA=1, LY
      IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
   IA=LY
   IC=IB+N1
   IE=IC+N1
   IF=NTOTAL-IE
   MFIRST=IF/N
   M1=MFIRST
   LB=N*M1+IE
   M=IE+N
   L=IE+1
   LG=NTOTAL/MFIRST
   MLAST=MFIRST*LG
   IF (NTOTAL-MLAST) 7, 6, 7
6 MLAST=MFIRST
   GO TO 8
7 LG=LG+1
   MLAST=NTOTAL-MLAST
8 CALL INPUT
   READ INPUT TAPE 7, 9, (R(I), I=L, M)
9 FORMAT(1)
   L=IB+1
   DO 14 J=1, IE
14 SUM(J)=0, 0
   DO 40 J=1, LG
      IA=LY
      IB=IA+LY
      IC=IB+N1
      IF (LG-J) 15, 15, 16
15 M1=MLAST
      LB=N*M1+IE
16 DO 20 KA=K1, M1
      K=M
      K=K+1
      M=M+N
      READ INPUT TAPE 7, 9, (R(I), I=K, M)
20 CONTINUE
   K1=1
   M=IE
   L1=IE+1
WRITE TAPE 2*(RI)*I=L1*LB)
DO 40 LA=I*LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
DO 40 L=MA*MB
I=IE-N+L
IB=IB+1
IC=IC+1
DO 40 KA=I*M1
I=I+N
SUM(IB)=SUM(IB)+R(I)
SQ(IC)=SQ(IC)+R(I)**2
CONTINUE
IA=LY
IB=IA+LY
IC=IB+N1
END FILE 2
REWRITE 2
L=IE+NPERC
I=29
READ INPUT TAPE 7*41*MA
WRITE OUTPUT TAPE 6*42*MA
FORMAT(4DH1 Z PROGRAM JOB NUMBER I10 )
CA=(COUNT*(COUNT-1))
DO 70 LA=I*LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
DO 70 L=MA*MB
IB=IB+1
IC=IC+1
CB=(COUNT*SQ(IC))-(SUM(IB)**2)
IF(CB) 45,45,56
SQ(IC)=+9.999
SUM(IB)=SUM(IB)/COUNT
GO TO 58
SQ(IC)=(SQRT(CB/CA))
SUM(IB)=SUM(IB)/COUNT
58 I=I+1
IF(I=30) 65,60,70
WRITE OUTPUT TAPE 6*61
FORMAT(64H1 VARIABLE STANDARD DEVIATION MEAN 1 NUMBER)
I=0
WRITE OUTPUT TAPE 6*66*L*SQ(IC)*SUM(IB)*COUNT
FORMAT(1H0I14,F19*3,F20*3,F9*0)
CONTINUE
M1=MFIRST
IA=LY
IB=IA+LY
IE=IB+(2*N1)
M=N*M1+IE
L1=IE+1
N2=NONE-1
DO 140 J=1*LG
IA=LY
38
IF (LG-J) 75,75,78
75 M1=MLAST
   IB=IA+LY
   IE=IB+(2*N1)
   M=M*M1+IE
78 READ TAPE 2*(R(I),I=L1,M)
   DO 140 LA=1,LY
      IA=IA+1
      MA=NX(LA)
      MB=NY(IA)
      MC=MB-MA+1
      LX=MC/NPERC
      LZ=NPERC*LX
      IF (LZ-MC) 82,80,82
80 LZ=NPERC
   GO TO 83
82 LX=LX+1
   LZ=MC-LZ
83 IG=-N
   DO 95 K=1,M1
      IB=2*LY
      IC=IB+N1
      IG=IG+N
      ID=IC+IG
   DO 95 I=MA,MB
      IE=ID+I
      IE=IE+N
      IB=IB+1
      IC=IC+1
      IF ((SQDIC)-9,999) 90,88,90
88 Z(IE)=+9,999
   GO TO 95
90 Z(IE)=(R(IE)-SUM(IB))/SQ(II)
95 CONTINUE
   IB=2*LY
   IC=IB+N1
   IG=-N
   DO 130 K=1,M1
      N3=NTWO-1
      IG=IG+N
      ID=IC+IG
      IE=ID+MA-1
      IE=IE+N
      LE=IE
   LC=NPERC
100 N2=N2+1
   DO 120 LB=1,LX
      N3=N3+1
   IF (LX-LB) 105,105,110
105 LC=LZ
110 L=LE+1
   LE=LE+LC
   IF (CARDS) 113,120,115
113 WRITE OUTPUT TAPE 6,114,N2,N3*(Z(KA),KA=L,LE)
114 FORMAT(1HOI6,I3,9F12.3)
   GO TO 120
115 PUNCH 119*(Z(KA),KA=L,LE)*N2,N3
119 FORMAT(9F8.3),I5,I3)
120 CONTINUE
130 CONTINUE
140 CONTINUE
   REWIND 2
   GO TO 1
*ASSEMBLE, PUNCH OBJECT
   REM INPUT
   ORG 0
   PGM
   PZE END, 0, 1
   PZE
   BCD 1INPUT
   PZE ENT
   REM
   ORG 0
   REL
   ERROR BCD 1ERROR
   ENT RTD 7
   CLA 7, 4
   STA CPY
   SXD AX1, 1
   LXA A=1, 1
   CPY CPY **, 1
   D=12 TXI **, 3, 1, 1
   TSX ERROR, 4
   TSX ERROR, 4
   TXL CPY, 1, 12
   LXD AX1, 1
   TRA 1, 4
   A=1 PZE 1
   AX1 BSS 1
   END SYN *
   END
*DATA
V. PROGRAMS FOR TESTING SIGNIFICANCE OF DIFFERENCE IN MEAN VALUES

1. t-Test Program

A. DESCRIPTION OF THE PROGRAM

1. The program is classified into two categories
   a. M 1A (missing data)
   b. M 2A (no-missing data)

2. Program print-out (see Appendix C)

3. Special code built into the program

   Whenever the variance of a variable for either one or both of the
   groups considered is exactly zero the program prints \( t = +9.999 \)

B. SPECIAL FEATURES

1. The input sub-routine reads in the FORMAT of the data cards.

2. The program can be used to calculate 't' on either all variables
   or on selected groups of variables in a punched matrix of cards.
   See Fig. 3.

3. The program has been designed to compute 't' values as follows:
<table>
<thead>
<tr>
<th>'t' VALUES ON EACH VARIABLE FOR GROUPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-2*</td>
</tr>
<tr>
<td>1-3  2-3</td>
</tr>
<tr>
<td>1-4  2-4  3-4</td>
</tr>
<tr>
<td>1-5  2-5  3-5  4-5</td>
</tr>
<tr>
<td>1-6  2-6  3-6  4-6  5-6</td>
</tr>
<tr>
<td>-         -         -         -         -</td>
</tr>
<tr>
<td>1-k 2-k 3-k 4-k 5-k...(k-1)-k</td>
</tr>
</tbody>
</table>

Thus at one loading of the program 't' values are computed on all the different combinations of groups taken two at a time \(_{k}^{\binom{k}{2}}\).

*1-2 means group 1 versus group 2.

C. DISCUSSION

1. Mean

\[
\bar{X}_j = \frac{\sum_{i=1}^{N} X_{i,j}}{N} \quad j = 1,2,\ldots,k \text{ groups}
\]

\(N = \text{number of observations in the group}\)

2. Standard deviation

\[
\sigma_j = \sqrt{\left(\frac{\sum_{i=1}^{N} X_{i,j}^2 - (\sum_{i=1}^{N} X_{i,j})^2}{N(N-1)}\right)}
\]
3. Student's t (uncorrelated groups)

\[ t = \frac{\bar{X}_i - \bar{X}_j}{\sqrt{\frac{\sigma_i^2}{N_i} + \frac{\sigma_j^2}{N_j}}} \quad i, j = \text{any two groups} \]

D. LIMITATIONS

<table>
<thead>
<tr>
<th>Program</th>
<th>MAXIMUM NUMBER OF VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>8k memory</td>
</tr>
<tr>
<td>M 1A</td>
<td>800*</td>
</tr>
<tr>
<td>M 2A</td>
<td>800*</td>
</tr>
</tbody>
</table>

* These are estimates.

E. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Actual computing time for M 1A per t is .2 to .5 seconds, depending on number of variables and observations. The M 2A program is about 2 to 3 times faster.

F. TEST PROBLEMS

Test problems have been designed so that they cover possible occurrences, and they are itemized for the categories of programs discussed above.

See examples in Appendix C.

G. OUTPUT

Printout: Group i; Group j; variable no.; N_i, \( \bar{X}_i \), \( \sigma_i \), N_j, \( \bar{X}_j \), \( \sigma_j \), t, df.
Appendix C

1. ARRANGEMENT OF CARDS

(See Appendix E under inter-correlation programs)

a. Using fortran deck (see Fig. 4)

b. Using binary deck (see Fig. 4)

2. EXAMPLE PROBLEMS

   t-test missing data program (M 1A) and

   t-test no-missing data program (M 2A)

<table>
<thead>
<tr>
<th>Example</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>50</td>
<td>100</td>
<td>25</td>
<td>80</td>
<td>68</td>
</tr>
<tr>
<td>B</td>
<td>80</td>
<td>90</td>
<td>100</td>
<td>25</td>
<td>40</td>
</tr>
<tr>
<td>C</td>
<td>60</td>
<td>75</td>
<td>52</td>
<td>68</td>
<td>32</td>
</tr>
<tr>
<td>D</td>
<td>25</td>
<td>23</td>
<td>32</td>
<td>40</td>
<td>28</td>
</tr>
</tbody>
</table>

   * The groups can be identified in the data cards by punching

   a group number in, for example, columns 73-74 (01, 02, etc.).

   The number of groups can vary from 2 to k.

3. DIMENSION CARDS FOR THE FORTRAN DECK

   The user is advised to follow through the steps described under inter-

   correlation programs, where the dimension cards are discussed in detail.

   The procedure to be followed here is approximately the same.
4. MFIRST = *** or ** CARD. (Asterisks indicate number of digits punched.)

For discussion on MFIRST = *** or ** Card, see inter-correlation program.

5. CONTROL CARDS

t-test missing data program (M 1A) and
t-test no-missing data program (M 2A)

a. Control card no. 1

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td>1-6</td>
<td>+00280</td>
</tr>
<tr>
<td></td>
<td>+00185</td>
</tr>
<tr>
<td></td>
<td>+00096</td>
</tr>
<tr>
<td></td>
<td>+00031</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 3.

2. See control cards, note 2, means and standard deviations program.

3. Note entries for LY. See corresponding sketches in Fig. 3, and also control cards 2 and 3. LX = Number of groups.

4. Columns 19-26 entry can be left blank in the case of no-missing data programs.
b. Control card no. 2

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>001</td>
</tr>
<tr>
<td>B</td>
<td>025</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 3

2. The starting point in the matrix is indicated on this card. Also note, in example D, LY has an entry 3 in control card 1. Correspondingly three starting variables are indicated on control card 2. In example C, LY = 2.

c. Control card no. 3

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>280</td>
</tr>
<tr>
<td>B</td>
<td>179</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 3

2. The last variable to be computed is indicated on this card.

3. The program is designed for a value LY = 24. However the program can be compiled for a larger value of LY. The desired number has to be entered in the dimension.
card before compilation.

d. Control card no. 4

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>050</td>
</tr>
<tr>
<td>B</td>
<td>080</td>
</tr>
<tr>
<td>C</td>
<td>060</td>
</tr>
<tr>
<td>D</td>
<td>025</td>
</tr>
</tbody>
</table>

Note: Punch number of observations in each group. Missing data, if any, are also to be counted as observations, although missing data are skipped during computation.

6. FORMAT STATEMENT

See Appendix A under means and standard-deviations program.

7. DATA CARDS

See Appendix A under means and standard-deviations program.

8. JOB NUMBER CARD

See Appendix A under means and standard-deviations program.
THE DATA CARDS FOR GROUPS 1, 2, etc. FOLLOW EACH OTHER WITHOUT ANY IN BETWEEN CARDS TO INDICATE BREAKS.

FIG. 3
*COMPILE FORTRAN PRINT SAP PUNCH OBJECT EXECUTE DUMP

DIMENSION A(200),22 SUM(200),SQUARE(200),COUNT(200),NX(25),NY(25),
1XMEAN(200),DEVMY(200),YMEAN(200),DEVNX(200),LC(200),MLAST(200)

EQUIVALENCE (DEVNX,SQUARE),(DEVMY,LC),(MLAST,YMEAN)

1 READ INPUT TAPE 7,2,N,LY,LX,D

2 FORMAT(3I6,F8.3)

RELINE 3
REWO 4
MFIRST=22
MLAST=MFIRST

READ INPUT TAPE 7,3,(NX(I),I=1,LX)

3 FORMAT(24I3)

READ INPUT TAPE 7,3,(NY(I),I=1,LX)

4 FORMAT(24F3.0)

DO 7 I=1,LX

J=COUNT(I)

LC(I)=(J/MFIRST)

MLAST(I)=(MFIRST*LC(I))

IF (J-MLAST(I)) 6,5,6

5 MLAST(I)=MFIRST

GO TO 7

6 LC(I)=LC(I)+1

MLAST(I)=J-MLAST(I)

7 CONTINUE

K=1

CALL INPUT

READ INPUT TAPE 7,8,(A(I,K),I=1,N)

8 FORMAT(

K=2

DO 32 LA=1,LX

MLAST(LA)

DO 9 J=1,N

SUM(J)=0.0

SQUARE(J)=0.0

COUNT(J)=0.0

9 CONTINUE

DO 20 J=1,LG

IF (LG-J) 10,10,11

M1=MLAST(LA)

10 MLAST(LA)=K1*M1

READ INPUT TAPE 7,8,(A(I,K),I=1,N)

12 CONTINUE

K1=1

DO 20 L=1,LY

NFIRST=NX(L)

NLAST=NY(L)

DO 20 I=NFIRST,NLAST

DO 20 K=1,M1

IF (A(I,K)-D) 17,20,17

17 SUM(I)=SUM(I)+A(I,K)

SQUARE(I)=SQUARE(I)+(A(I,K)**2)

COUNT(I)=COUNT(I)+1

20 CONTINUE

DO 32 L=1,LY

NFIRST=NX(L)

NLAST=NY(L)

49
DO 30 I=NFIRST,NLAST
CA=(COUNT(I)*SQUARE(I)-(SUM(I)**2)
CA=COUNT(I)*(COUNT(I)-1)
IF (CA) 22,22,25
22 XMEAN(I)=999.999
DEVNX(I)=0.0
25 IF (CB) 26,26,27
26 XMEAN(I)=999.999
DEVNX(I)=0.0
GO TO 30
27 DEVNX(I)=SQR(CB/CA)
XMEAN(I)=SUM(I)/COUNT(I)
30 CONTINUE
WRITE TAPE 3,((DEVNX(I),XMEAN(I),COUNT(I)),I=NFIRST,NLAST)
WRITE TAPE 4,((DEVNX(I),XMEAN(I),COUNT(I)),I=NFIRST,NLAST)
32 CONTINUE
END FILE 3
END FILE 4
REWIND 3
REWIND 4
READ INPUT TAPE 7,33,LA
33 FORMAT(I10)
WRITE OUTPUT TAPE 6,34,LA
34 FORMAT(38HU MEAN AND TEE PROGRAM JOB NUMBER 110)
LA=LX-1
DO 70 J=1,LA
DO 35 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
READ TAPE 3,((DEVNX(I),XMEAN(I),COUNT(I)),I=NFIRST,NLAST)
35 CONTINUE
DO 65 K=1,LX
DO 36 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
READ TAPE 4,((DEVNY(I),YMEAN(I),SUM(I)),I=NFIRST,NLAST)
36 CONTINUE
IF (K-J) 65,65,38
38 SKIP=29.
DO 60 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
DO 60 I=NFIRST,NLAST
IF (DEVNX(I)) 39,39,41
39 DEVNX(I)=999.999
TEE=999.999
IF (DEVNY(I)) 40,40,44
40 DEVNY(I) = 999.999
TEE=999.999
GO TO 44
41 IF (DEVNY(I)) 44,44,43
43 TEE=(XMEAN(I)-YMEAN(I))/(SQR((DEVNX(I)**2/COUNT(I))*DEVNY(I)**2/
1SUM(I))))
DF=(COUNT(I)+SUM(I)-2)
SKIP=SKIP+1.
45 IF (30.SKIP) 50,46,50
46 WRITE OUTPUT TAPE 6,48
48 FORMAT(95H1 GROUPS VARIABLE N-TOTAL MEAN SIGMA N-TOTA

50
1L MEAN SIGMA TEE DF )
50 WRITE OUTPUT TAPE 6*B5*J*K*1,COUNT(I)*XMEAN(I)*DEVNX(I),
1SUM(I)*YMEAN(I)*DEVNY(I)*TEE*DF
55 FORMAT(1HC15*14*16*F10*0*2(F11*3)*F6*0*3(F11*3)*F10*0)
60 CONTINUE
65 CONTINUE
REWIND 4
70 CONTINUE
REWIND 3
GO TO 1
*ASSEMBLE,PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END*0*1
PZE
BCD 1INPUT
PZE ENT
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7*4
STA CPY
SXD AX1,1
LXA A=1,1
CPY CPY **1
D=12 TXI **3*1,1
TSX ERROR*4
TXS ERROR*4
TXL CPY*1,12
LXD AX1,1
TRA 1*4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
*Compile FORTRAN* PRINT SAP PUNCH OBJECT EXECUTE DUMP

DIMENSION A(120,40) SUM(120) SQUARE(120) COUNT(120) NX(25) NY(25)
1 XMEAN(120) DEVNY(120) YMEAN(120) DEVNX(120) LC(120) MLAST(120)
EQUIVALENCE (DEVNY*LC)(MLAST*YMEAN)(SUM*XMEAN)(SQUARE*DEVINX)

1 READ INPUT TAPE 7*2*NLY*NLX

2 FORMAT(3I6)
REWRITE 3
REWIND 4
M1=MFIRST
READ INPUT TAPE 7*3*(NX(I),I=1,N)

3 FORMAT(24I3)
READ INPUT TAPE 7*3*(NY(I),I=1,LN)
READ INPUT TAPE 7*4*(COUNT(I),I=1,LX)

4 FORMAT(24F3.0)
DO 7 I=1,N
J=COUNT(I)
LC(I)=(J/MFIRST)
MLAST(I)=(MFIRST*LC(I))
IF (J-MLAST(I)) 6,5,6
MLAST(I)=MFIRST
GO TO 7

6 LC(I)=LC(I)+1
MLAST(I)=J-MLAST(I)

7 CONTINUE
K=1
K1=2
CALL INPUT
READ INPUT TAPE 7*8*(A(I,K),I=1,N)

8 FORMAT(1)
DO 32 LA=1,NX
CA=COUNT(LA)*(COUNT(LA)-1)
M1=MFIRST
LG=LC(LA)
DO 9 J=1,N
SUM(J)=0.0
SQUARE(J)=0.0

9 CONTINUE
DO 20 J=1,LG
IF (LG-J) 10,10,11

10 M1=MLAST(LA)
11 DO 12 K=K1,M1
READ INPUT TAPE 7*8*(A(I,K),I=1,N)

12 CONTINUE
K1=1
DO 20 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
DO 20 I=NFIRST,NLAST
DO 20 K=1,M1
SUM(I)=SUM(I)+A(I,K)
SQUARE(I)=SQUARE(I)+(A(I,K)**2)

20 CONTINUE
DO 32 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
DO 30 I=NFIRST,NLAST
CB=(COUNT(LA)*SQUARE(I)-(SUM(I)**2))
```plaintext
23  XMEAN(I)=SUM(I)/COUNT(LA)
25  IF (CB) 26,26,27
26  DEVX(I)=0.0
   GO TO 30
27  DEVX(I)=SORT(CB/CA)
30  CONTINUE
   WRITE TAPE 3*(((DEVX(I)*XMEAN(I)),I=NFIRST,NLAST)
   WRITE TAPE 4*(((DEVX(I)*XMEAN(I)),I=NFIRST,NLAST)
31  CONTINUE
   END FILE 3
   END FILE 4
   REWIND 3
   REWIND 4
   READ INPUT TAPE 7*33,LA
33  FORMAT(I10)
   WRITE OUTPUT TAPE 6*34,LA
34  FORMAT(38H0, MEAN AND TEE PROGRAM JOB NUMBER I10 )
   LA=LX=1
   DO 70   J=1,LA
   DO 35 L=1,LY
      NFIRST=NX(L)
      NLAST=NY(L)
      READ TAPE 3*(((DEVX(I)*XMEAN(I)),I=NFIRST,NLAST)
35  CONTINUE
   DO 65 K=1,LX
   DO 36 L=1,LY
      NFIRST=NX(L)
      NLAST=NY(L)
      READ TAPE 4*(((DEVX(I)*YMEAN(I)),I=NFIRST,NLAST)
36  CONTINUE
   IF (K-J) 65,65,38
38  SKIP=29,
   DO 60 L=1,LY
      NFIRST=NX(L)
      NLAST=NY(L)
   DO 60 I=NFIRST,NLAST
      IF (DEVX(I)) 39,39,41
39  TEE=999.999
   DEVX(I)=999.999
   IF (DEVX(I)) 40,40,44
40  DEVX(I) = 999.999
   TEE = 999.999
   GO TO 44
41  IF (DEVX(I)) 40,40,43
43  TEE=(XMEAN(I)-YMEAN(I))/(SQRT((DEVX(I)**2/COUNT(J))+((DEVX(I)**2/
1COUNTERK(I))))
44  DF=(COUNT(J)+COUNTERK(I))-2.
   SKIP=SKIP+1.
45  IF (30.-SKIP) 50,50,50
46  WRITE OUTPUT TAPE 6*48
48  FORMAT(95H1 GROUPS VARIABLE N-TOTAL MEAN SIGMA N-TOTA
1L MEAN SIGMA TEE DF )
   SKIP=0.0
50  WRITE OUTPUT TAPE 6*55,J,K,I,COUNT(J),XMEAN(I),DEVX(I),
1COUNTERK(I),YMEAN(I),DEVX(I), TEE,DF
55  FORMAT(1HO15,14,16,F10.0,2(F11.3),F6.0,3(F11.3),F10.0)
60  CONTINUE
65  CONTINUE
```
REWIN D 4
70 CONTINUE
REWIN D 3
GO TO 1
*ASSEMBLE PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END$0+1
PZE
BCD 1INPUT
PZE ENT
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7$4
STA CPY
SXD AX1$1
LXA A=1$1
CPY CPY **$1
D=12 TXI **3$1+1
TSX ERROR$4
TSX ERROR$4
TXL CPY$1+12
LXD AX1$1
TRA 1$4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
2. One Way Analysis of Variance Program

A. DESCRIPTION OF THE PROGRAM

1. This group of programs have been designed so that large numbers of sub-groups on each variable could be analyzed. The programs have been classified into two major categories.
   
a. Missing data program
      
i. A 1A
         
This program computes F-ratios on each variable for all combinations of groups taken two at a time.
      
ii. A 1B
         
This program computes F-ratios on each variable taking into account all of the groups at a time.
   
b. No-missing data program
      
i. A 2A
         
See a.i. above.
      
ii. A 2B
         
See a.ii. above.

2. Program print-out (See Appendix D)

3. Special code built into program

   To avoid stoppage due to any unforeseen circumstances, the following feature is incorporated:

   Whenever the variance of any group considered is exactly zero the program prints out the comment "Please check your data."
B. SPECIAL FEATURES

1. The input sub-routine reads in the FORMAT of the data cards. The program therefore is capable of handling data with any FORMAT.

2. The F-ratios can first be computed on each variable considering all groups at a time. Then the F-ratios can be computed on each variable for all combinations of groups taking two at a time as shown below. (The t-program may be used for the same purpose: \( t = \sqrt{F} \).)

<table>
<thead>
<tr>
<th>GROUP</th>
<th>F-RATIO VALUES ON EACH VARIABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1-2*</td>
</tr>
<tr>
<td>2</td>
<td>1-3 2-3</td>
</tr>
<tr>
<td>3</td>
<td>1-4 2-4 3-4</td>
</tr>
<tr>
<td>4</td>
<td>1-5 2-5 3-5 4-5</td>
</tr>
<tr>
<td>k</td>
<td>1-k 2-k 3-k 4-k...(k-1)-k</td>
</tr>
</tbody>
</table>

*1-2 means group 1 versus group 2.

C. DISCUSSION

\[ k \quad = \quad \text{Number of groups} \]
\[ N_k \quad = \quad \text{Number of observations in a group} \]
\[ N_{tot} \quad = \quad \sum N_k \]
\[ A \quad = \quad (\sum N_k \bar{X}_k^2) - (N_{tot} \bar{X}_{tot}^2) \quad = \quad \text{SS between groups} \]
\[ B \quad = \quad (\sum \bar{X}_{tot}^2) - (N_{tot} \bar{X}_{tot}^2) \quad = \quad \text{SS total} \]
\[ C \quad = \quad B - A \quad = \quad \text{SS within groups} \]
\[ MS_{bg} = \frac{A}{k-1} \]

\[ MS_{wg} \text{ (residual)} = \frac{C}{N_{tot}-k} \]

\[ F = \frac{MS_{bg}}{MS_{wg}} \]

\( MS = \) Mean Square (variance estimate)

The above calculations are done for each variable. Thus in A 1B and A 2B the program will print out one F-ratio per variable. However, in program A 1A and A 2A a total of \( k(k-1)/2 \) F-ratios are computed for each variable.

### D. LIMITATIONS

<table>
<thead>
<tr>
<th>Program Category</th>
<th>MAXIMUM NUMBER OF VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8k memory</td>
</tr>
<tr>
<td>A 1A</td>
<td>500*</td>
</tr>
<tr>
<td>A 1B</td>
<td>500*</td>
</tr>
<tr>
<td>A 2A</td>
<td>500*</td>
</tr>
<tr>
<td>A 2B</td>
<td>500*</td>
</tr>
</tbody>
</table>

* These are estimates.

If the data matrix is very large, say, several hundred variables, it would be advisable to divide it into two or three smaller matrices, and run the jobs on one loading. This can be done because the variables are not interdependent, and it will reduce running time appreciably. As to the number of observations on each variable, the storage capacity of the tape unit is the only limiting factor.
E. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Actual computation time for A 1A is .2 to .5 seconds per F-ratio, depending on number of variables and observations. An actual run of 189 variables and 225 observations (divided into 5 groups), which gave $\frac{189}{5}$ = 1890 F-ratios, required 0.24 seconds per F-ratio.

The A 2A program is about 2 to 3 times faster.

The computation time for A 1B and A 2B should be estimated as if the groups were run two at a time. This will be an overestimate, though safe.

F. TEST PROBLEMS

Test problems have been designed so that they cover possible occurrences.

See examples in Appendix D.

G. OUTPUT

Printout for two groups at a time: variable no.; Group i; Group j; SS; Residual SS; $N_i$; $N_j$; df 1; df 2; F; $\bar{X}_i$; $\sigma_i$; $\bar{X}_j$; $\sigma_j$. Printout for all groups at a time: variable no.; SS; Residual SS; df 1; df 2; F. (Detailed information on the groups can be obtained by running $\binom{k}{2}$ combinations of groups.)
Appendix D

1. ARRANGEMENT OF CARDS

(See Appendix E under inter-correlation programs)

a. Using FORTRAN deck (See Fig. 4)

b. Using binary deck (See Fig. 4)

2. EXAMPLE PROBLEMS

a. i. One way analysis of variance missing-data program (A 1A),
(Two groups at a time)

   ii. One way analysis of variance missing data program (A 1B),
(All groups at a time)

b. i. One way analysis of variance no-missing data program (A 2A), and
(Two groups at a time)

   ii. One way analysis of variance no-missing data program (A 2B)
(All groups at a time)

<table>
<thead>
<tr>
<th>Example</th>
<th>NUMBER OF OBSERVATIONS IN THE GROUPS</th>
<th>NUMBER OF VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1  2  3  4</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>50 100 25 80 68</td>
<td>380</td>
</tr>
<tr>
<td>B</td>
<td>80 90 100 25 40</td>
<td>185</td>
</tr>
<tr>
<td>C</td>
<td>60 75 52 68 52</td>
<td>90</td>
</tr>
<tr>
<td>D</td>
<td>25 23 52 40 28</td>
<td>31</td>
</tr>
</tbody>
</table>

59
5. DIMENSION CARDS FOR THE FORTRAN DECK

The user is advised to follow through the steps described under intercorrelation programs, where the dimension cards are discussed in detail. The procedure to be followed is approximately the same.

4. MFIRST = *** or ** CARD

For discussion on MFIRST = *** or ** Card see inter-correlation programs.

5. CONTROL CARDS

Missing data programs (A 1A), (A 1B), and

No-missing data programs (A 2A), (A 2B)

a. Control card no. 1

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-6</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td>A</td>
<td>+00280</td>
</tr>
<tr>
<td>B</td>
<td>+00135</td>
</tr>
<tr>
<td>C</td>
<td>+00096</td>
</tr>
<tr>
<td>D</td>
<td>+00031</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 3.

2. See control cards, note 2, means and standard deviations program.

3. Note entries for LY. See corresponding skethes in Fig. 3 and also control cards 2 and 3. LX = number of groups.

4. Columns 19-26 entry can be left blank in the case of no-missing data programs.

60
b. Control card no. 2

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-5</td>
</tr>
<tr>
<td>A</td>
<td>001</td>
</tr>
<tr>
<td>B</td>
<td>025</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 3 (Appendix C)

2. The starting point in the matrix is indicated on this card. Also note, in example D, LY has an entry 5 in control card no. 1. Correspondingly three starting variables are indicated on control card no. 2.

c. Control card no. 5

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-5</td>
</tr>
<tr>
<td>A</td>
<td>280</td>
</tr>
<tr>
<td>B</td>
<td>179</td>
</tr>
<tr>
<td>C</td>
<td>018</td>
</tr>
<tr>
<td>D</td>
<td>003</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 5, Appendix C.

2. The last variable to be computed is indicated on this card.
3. The program is designed for a value \( LY = 24 \). However, the program can be compiled for a larger value of \( LY \).

The desired number has to be entered in the dimension card before compilation.

d. Control card no. 4

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-5</td>
</tr>
<tr>
<td>A</td>
<td>050</td>
</tr>
<tr>
<td>B</td>
<td>060</td>
</tr>
<tr>
<td>C</td>
<td>060</td>
</tr>
<tr>
<td>D</td>
<td>025</td>
</tr>
</tbody>
</table>

Note: Punch number of observations in each group. Missing data, if any, are also to be counted as observations although the program omits counting missing data during computation.

6. FORMAT STATEMENT

See Appendix A, means and standard deviations program.

7. DATA CARDS

See Appendix A, means and standard deviations program.

8. JOB NUMBER CARD

See Appendix A, means and standard deviations program.
*COMPILE FORTRAN*PRINT SAP,PUNCH OBJECT*EXECUTE*DUMP 

```
DIMENSION A(200), SUM(200), SQUARE(200), COUNT(200), TOTAL(200),
1 XMEAN(200), DEVNY(200), YMEAN(200), DEVX(200), SOM(200), SGRE(200),
1 LC(200), MLAST(200), NX(25), NY(25)

EQUIVALENCE (MLAST,YMEAN),(LC,DEVX,DEVNY)
1 READ INPUT TAPE 7*2,N*LX*LY*D
2 FORMAT(316,F8.3)
   REWIND 3
   REWIND 4
   MFIRST=17
   READ INPUT TAPE 7*3*(NX(I)*I=1,LY)
3 FORMAT(24I3)
   READ INPUT TAPE 7*3*(NY(I)*I=1,LY)
   READ INPUT TAPE 7*4*(COUNT(I)*I=1,LX)
4 FORMAT(24F3.0)
   DO 7 I=1*LX
       J=COUNT(I)
       LC(I)=(J/MFIRST)
       MLAST(I)=(MFIRST*LC(I))
       IF (J-MLAST(I)) 6,5,6
5 MLAST(I)=MFIRST
   GO TO 7
6 LC(I)=LC(I)+1
   MLAST(I)=J-MLAST(I)
7 CONTINUE
   K=1
   K1=2
   CALL INPUT
   READ INPUT TAPE 7*8*(A(I,K),I=1,N)
8 FORMAT(
   1)
   DO 32 LA=1,LX
       M1=MFIRST
       LG=LC(LA)
       DO 9 J=1,N
           SUM(J)=0.0
           SQUARE(J)=0.0
           COUNT(J)=0.0
   9 CONTINUE
   DO 20 J=1,LG
       IF (LG-J) 10,10,11
10 M1=MLAST(LA)
11 DO 12 K=K1,M1
   READ INPUT TAPE 7*8*(A(I,K),I=1,N)
12 CONTINUE
   K1=1
   DO 20 L=1,LY
       NFIRST=NX(L)
       NLAST=NY(L)
       DO 20 I=NFIRST,NLAST
       DO 20 K=1,M1
       IF (A(I,K)-D) 17,20,17
17 SUM(I)=SUM(I)+A(I,K)
   SQUARE(I)=SQUARE(I)+(A(I,K)**2)
   COUNT(I)=COUNT(I)+1
20 CONTINUE
   DO 32 L=1,LY
       NFIRST=NX(L)
       NLAST=NY(L)
```
```
DO 30 I=NFIRST,NLAST
CB=(COUNT(I)*SQUARE(I))-(SUM(I)**2)
CA=COUNT(I)*(COUNT(I)-1)
IF (COUNT(I)) 22,22,23
22 XMEAN(I)=0.0
DEVNX(I)=0.0
GO TO 30
23 XMEAN(I)=SUM(I)/COUNT(I)
IF (CA) 24,24,25
24 IF (CB) 26,26,27
26 IF (CB) 0.0
GO TO 30
27 DEVNX(I)=SQRT(CB/CA)
30 CONTINUE
WRITE TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIRST
1ST*NLAST)
WRITE TAPE 4,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIRST
1ST*NLAST)
32 CONTINUE
END FILE 3
END FILE 4
REWR 3
REWIND 4
READ INPUT TAPE 7,33,LA
33 FORMAT(I10)
WRITE OUTPUT TAPE 6,34,LA
34 FORMAT(4SH0 ANALYSIS OF VARIANCE PROGRAM JOB NUMBER I10 )
LA=LX-1
DO 70 J=1,LA
DO 35 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
READ TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIRST
1ST*NLAST)
35 CONTINUE
DO 65 K=1,LX
DO 36 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
READ TAPE 4,((SUM(I),SQERE(I),TOTAL(I),YMEAN(I),DEVNY(I)),I=NFIRST
1ST*NLAST)
36 CONTINUE
IF (K-J) 65,65,37
37 SKIP=29
DO 60 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
DO 60 I=NFIRST,NLAST
IF (DEVNX(I)) 38,38,49
38 IF (DEVNY(I)) 39,39,43
39 SKIP=SKIP+1
IF (30-SKIP) 60,40,41
40 WRITE OUTPUT TAPE 6,57
SKIP=0.0
41 WRITE OUTPUT TAPE 6,42,I+J+K
42 FORMAT(IHOI9,I8,I4,43H PLEASE CHECK RAW DATA THANK YOU. )
GO TO 60
43 SKIP = SKIP + 1
   IF (300 - SKIP) 60, 44, 45
44 WRITE OUTPUT TAPE 6, 57
   SKIP = 0
45 WRITE OUTPUT TAPE 6, 57, 1, 6, 7, K, XMEAN(I), YDNY(I)
47 FORMAT(1H01, 1, 8, 1, 4, 1, 7, 6H, PLEASE CHECK YOUR DATA FOR THE TWO GROUPS
   1S
   2(F10, 3))
GO TO 60
49 IF (YDNY(I)) 50, 50, 54
50 SKIP = SKIP + 1
   IF (300 - SKIP) 60, 51, 52
51 WRITE OUTPUT TAPE 6, 57
   SKIP = 0
52 WRITE OUTPUT TAPE 6, 57, 1, 6, 7, K, XMEAN(I), YDNY(X)
53 FORMAT(1H01, 1, 8, 1, 4, 1, 5, 1, 2, 6H, PLEASE CHECK YOUR DATA FOR THE TWO GROUPS
   1S
   2(F10, 3))
GO TO 60
54 TOP1 = ((SUM(I) ** 2) / COUNT(I)) + ((SOM(I) ** 2) / TOTAL(I))
   DF2 = COUNT(I) + TOTAL(I)
   TOP2 = ((SUM(I) + SOM(I)) ** 2) / (DF2)
   XBAR = TOP1 - TOP2
   DF1 = 1
   DF2 = DF2 - 2
   R = (SQUARE(I) + SQUARE(1)) - TOP1
   Z2 = R / DF2
   Z3 = XBAR / Z2
   SKIP = SKIP + 1
   IF (300 - SKIP) 60, 55, 58
55 WRITE OUTPUT TAPE 6, 57
57 FORMAT(1H01, 1, 8, 1, 4, 1, 1, 6H, VARIABLE GROUPS S.S.BETWEEN S.S.RESIDUAL N-1
   1N=2, DF1, DF2, F-RATIO, MEAN, SIGMA, MEAN, SIGMA)
   SKIP = 0
58 WRITE OUTPUT TAPE 6, 59, 1, 6, 7, K, XBAR, R, COUNT(I), TOTAL(I), DF1, DF2, Z3,
   1XMEAN(I), YDNY(I), XMEAN(I), YDNY(I)
59 FORMAT(1H01, 1, 8, 1, 4, 1, 3, 1, 6, 1, 3, 1, 4, 1, 3, 1, 3, 1, 3, 1, 3F10, 3)
60 CONTINUE
65 CONTINUE
   REWIND 4
70 CONTINUE
   REWIND 3
   GO TO 1
*ASSEMBLE, PUNCH OBJECT
   REM INPUT
   ORG 0
   PGM
   PZE END 0, 1
   PZE
   BCD 1INPUT
   PZE ENT
   REM
   ORG 0
   REL
ERROR BCD 1ERROR
   RTD 7
   CLA 7, 4
   STA CPY
   SXD AX 1, 1

65
LXA A=1,1
CPY CPY **,1
D=12 TXI ***3,1,1
TSX ERROR ,4
TSX ERROR ,4
TXL CPY,1,12
LXD AX1,1
TRA 1,4
A=1 PZE 1
AX1 BSS 1
END SYN *
END

*DATA
*COMPILE FORTRAN, PRINT SAP, PUNCH OBJECT, EXEDEUTE, DUMP
DIMENSION A(200), SUM(200), SQUARE(200), COUNT(200), NX(25), NY(25),
XMEAN(200), DEVMY(200), YMEAN(200), DEVNX(200), LC(200), MLAST(200),
SM(200), SQEME(200)
EQUIVALENCE (MLAST, XMEAN), (LC, DEVMY)
1 READ INPUT TAPE 7, 2*N, LY, LX
2 FORMAT(3I6)
   REWIND 3
   REWIND 4
   MFIRST=19
   M1=MFIRST
   READ INPUT TAPE 7, 3*(NX(I), I=1, LY)
3 FORMAT(24I3)
   READ INPUT TAPE 7, 3*(NY(I), I=1, LY)
   READ INPUT TAPE 7, 4*(COUNT(I), I=1, LX)
4 FORMAT(24F3.0)
   DO 7 I=1, LX
       J=COUNT(I)
       LC(I)=(J/MFIRST)
       MLAST(I)=(MFIRST*LC(I))
       IF (J-MLAST(I)) 6, 5, 6
5 MLAST(I)=MFIRST
   GO TO 7
6 LC(I)=LC(I)+1
   MLAST(I)=J-MLAST(I)
7 CONTINUE
   K=1
   K1=2
   CALL INPUT
   READ INPUT TAPE 7, 8*(A(I,K), I=1, N)
8 FORMAT(1)
   DO 32 LA=1, LX
       M1=MFIRST
       LG=LC(LA)
       CA=COUNT(LA)*(COUNT(LA)-1)
       DO 9 J=1, N
           SUM(J)=0, 0
           SQUARE(J)=0, 0
9 CONTINUE
   DO 20 J=1, LG
       IF (LG-J) 10, 10, 11
10 M1=MLAST(LA)
11 DO 12 K=K1, M1
   READ INPUT TAPE 7, 8*(A(I,K), I=1, N)
12 CONTINUE
   K1=1
   DO 20 L=1, LY
       NFIRST=NX(L)
       NLAST=NY(L)
       DO 20 I=NFIRST, NLAST
17 SUM(I)=SUM(I)+A(I,K)
       SQUARE(I)=SQUARE(I)+(A(I,K)**2)
20 CONTINUE
   DO 32 L=1, LY
       NFIRST=NX(L)
       NLAST=NY(L)
   DO 30 I=NFIRST, NLAST

67
CB=(COUNT(LA)*SQUARE(I))-(SUM(I)**2)
23  XMEAN(I)=SUM(I)/COUNT(LA)
25  IF (CB) 26,26,27
26  DEVNX(I)=0
27  GO TO 30
28  DEVNX(I)=SQRT(CB/CA)
30  CONTINUE
31  WRITE TAPE 3,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
32  END FILE 3
33  WRITE TAPE 4,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
34  CONTINUE
35  END FILE 4
36  REWIND 3
37  REWIND 4
38  READ INPUT TAPE 7,33,LA
39  FORMAT(I10)
40  WRITE OUTPUT TAPE 6,34,LA
41  FORMAT(38B0 ANALYSIS OF VARIANCE JOB NUMBER 110)
42  LA=LX-1
43  DO 70 J=1,LA
44  DO 35 L=1,LY
45  NFIRST=NX(L)
46  NLAST=NY(L)
47  READ TAPE 3,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
48  CONTINUE
49  DO 65 K=1,LX
50  DO 36 L=1,LY
51  NFIRST=NX(L)
52  NLAST=NY(L)
53  READ TAPE 4,((SUM(I),SQUARE(I),XMEAN(I),DEVNY(I)),I=NFIRST,NLAST)
54  CONTINUE
55  IF (K-J) 65,65,37
56  DO 60 I=1,LX
57  IF (DEVNX(I)) 38,38,49
58  IF (DEVNY(I)) 39,39,43
59  SKIP=SKIP+1
60  IF (30.-SKIP) 60,40,41
61  WRITE OUTPUT TAPE 6,57
62  SKIP=0
63  WRITE OUTPUT TAPE 6,42,1,J,K
64  FORMAT(10019,18,14,43H PLEASE CHECK YOUR RAW DATA, THANK YOU )
65  GO TO 60
66  SKIP=SKIP+1
67  IF (30.-SKIP) 60,44,45
68  WRITE OUTPUT TAPE 6,57
69  SKIP=0
70  WRITE OUTPUT TAPE 6,47,1,J,K,YMEAN(I),DEVNY(I)
71  FORMAT(10019,18,14,76H PLEASE CHECK YOUR DATA FOR THE TWO GROUP)
72  15
73  GO TO 60
74  IF (DEVNY(I)) 50,50,54
75  SKIP=SKIP+1
76  IF (30.-SKIP) 60,51,52
77  WRITE OUTPUT TAPE 6,57
52 WRITE OUTPUT TAPE 6,53,1, J, K, XMEAN(I), DEVNX(I).
53 FORMAT(1HU19,I8,14,52H PLEASE CHECK YOUR DATA FOR THE TWO GROUP
15 F14.3,F10.3)
      GO TO 60
54 TOP2=((SUM(I)**2)/COUNT(J))+(SUM(I)**2)/COUNT(K)
      DF2=COUNT(J)+COUNT(K)
      XBAR=(SUM(I)+SUM(J))/(DF2)
      DF1=1
      DF2=DF2-2
      R=(SQUARE(I)+SQERE(I))-TOP1
      Z2=R/DF2
      Z3=XBAR/Z2
      SKIP=SKIP+1
      IF (30.-SKIP) 60,55,58
55 WRITE OUTPUT TAPE 6,57
56 FORMAT(118H1 VARIABLE GROUPS S•S•BETWEEN S•S•RESIDUAL N-1
58 1N-2 DF1 DF2 F-RATIO MEAN SIGMA MEAN SIGMA)
59 FORMAT(118H1 VARIABLE GROUPS S•S•BETWEEN S•S•RESIDUAL N-1
61 1N-2 DF1 DF2 F-RATIO MEAN SIGMA MEAN SIGMA)
62 CONTINUE
65 CONTINUE
70 CONTINUE
    REWIND 4
70 CONTINUE
    REWIND 3
    GO TO 1

*ASSEMBLE•PUNCH OBJECT
REM INPUT
PGM
  P2E END*0,1
  P2E
  BCD 1INPUT
  P2E END
REM
  ORG 0
  REL
ERROR BCD 1ERROR
ENT RTD 7
    CLA 7,4
    STA CPY
    SXD AX1+1
    LXA A=1+1
    CPY ***1
    D=12
    TXI +3+1,1
    TXS ERROR+4
    TXS ERROR+4
    TXL CPY+1,12
    LXD AX1+1
    TRA 1,4
    A=1
    P2E 1
    AX1 BSS 1
    END SYN *
    END

*DATA
*COMPILE FORTRAN, PRINT SAPI, PUNCH OBJECT; EXECUTE, DUMP

DIMENSION A(200,17), SUM(200), SQUARE(200), COUNT(200), TOTAL(200),
1 XMEAN(200), DEVNY(200), YMEAN(200), DEVNX(200), SOM(200), SQUARE(200),
1 LC(200), MLAST(200), NX(25), NY(25),
EQUIVALENCE (MLAST, YMEAN), (LC, DEVNY)

1 READ INPUT TAPE 7;2;NY;LY;LX;ID
2 FORMAT(3I6,F8.3)
   REWIND 3
   MFIRST=17
   READ INPUT TAPE 7;3;NY(I);I=1;LY)
3 FORMAT(24I3)
   READ INPUT TAPE 7;3;NY(I);I=1;LY)
   READ INPUT TAPE 7;4;COUNT(I);I=1;LX)
4 FORMAT(24F3.0)
   DO 7 I=1;LY
      J=COUNT(I)
      LC(I)=(J/MFIRST)
      MLAST(I)=(MFIRST*LC(I))
      IF (J-MLAST(I)) 6;5;6
5 MLAST(I)=MFIRST
   GO TO 7
6 LC(I)=LC(I)+1
   MLAST(I)=J-MLAST(I)
7 CONTINUE
   K=1
   K1=2
   CALL INPUT
   READ INPUT TAPE 7;8;I(A(I;K);I=1;N)
8 FORMAT(1)
   DO 32 LA=1;LY
       M1=MFIRST
       LG=LC(LA)
       DO 9 J=1;N
           SUM(J)=0;0
           SQUARE(J)=0;0
           COUNT(J)=0;0
9 CONTINUE
   DO 20 J=1;LG
       IF (LG-J) 10;10;11
10 M1=MLAST(LA)
11 DO 12 K=K1;M1
       READ INPUT TAPE 7;8;I(A(I;K);I=1;N)
12 CONTINUE
   K1=1
   DO 20 L=1;LY
       NFIRST=NX(L)
       NLAST=NY(L)
       DO 20 I=NFIRST;NLAST
           DO 20 K=1;M1
               IF (A(I;K)-D) 17;20;17
17 SUM(I)=SUM(I)+A(I;K)
           SQUARE(I)=SQUARE(I)+(A(I;K)**2)
           COUNT(I)=COUNT(I)+1
20 CONTINUE
   DO 32 L=1;LY
       NFIRST=NX(L)
       NLAST=NY(L)
       DO 30 I=NFIRST;NLAST

70
CB=(COUNT(I)*SQUARE(I)-(SUM(I)**2)
   /COUNT(I)*(COUNT(I)-1))
IF (COUNT(I)) 22,22,23

22 XMEAN(I)=0.0
   DEVNX(I)=0.0
   GO TO 30
23 XMEAN(I)=SUM(I)/COUNT(I)
   IF (CA) 24,24,25
24 DEVNX(I)=0.0
   GO TO 30
25 IF (CB) 26,26,27
26 DEVNX(I)=0.0
   GO TO 30
27 DEVNX(I)=SQRT(CB/CA)
30 CONTINUE
   WRITE TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIN
   1ST,NLAST)
31 CONTINUE
   END FILE 3
   REWIND 3
   READ INPUT TAPE 7,33,LA
32 FORMAT(I10)
   WRITE OUTPUT TAPE 6,34,LA
33 FORMAT(45H0 ANALYSIS OF VARIANCE PROGRAM JOB NUMBER I10 )
   LA=LX=1
   SKIP=29*
   DF1 = LX - 1
   GROUP=LX
   DO 35 I=1,N
   LC(I) = 1
   SQERE(I)=0.0
   SOM(I)=0.0
   TOTAL(I)=0.0
35 YMEAN(I)=0.0
   DO 50 J=1,LX
   DO 36 L=1,LY
   NFIRST = NX(L)
   NLAST = NY(L)
   READ TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFI
   RST,NLAST)
36 CONTINUE
   DO 50 L=1,LY
   NFIRST = NX(L)
   NLAST = NY(L)
   DO 50 I = NFIRST, NLAST
   IF (LC(I)) 50,50,37
37 IF (DEVNX(I)) 38,38,39
38 LC(I) = -1
   GO TO 50
39 YMEAN(I) = YMEAN(I) + ((SUM(I)**2) / COUNT(I))
   TOTAL(I) = TOTAL(I) + COUNT(I)
   SOM(I) = SOM(I) + SUM(I)
   SQERE(I) = SQERE(I) + SQUARE(I)
50 CONTINUE
   DO 70 L=1,LY
   NFIRST = NX(L)
   NLAST = NY(L)
   DO 70 I = NFIRST, NLAST

71
IF (LC(I)) 43,43,54
43 SKIP=SKIP+1.
   IF (30-=SKIP) 70,44,45
44 WRITE OUTPUT TAPE 6,57
   SKIP=0.0
45 WRITE OUTPUT TAPE 6,47,I
47 FORMAT(1H0I9,40H PLEASE CHECK YOUR DATA )
   GO TO 70
54 TOP 2 = (SOM(I)**2) / TOTAL(I)
   DF2=TOTAL(I)-GROUP
   XBAR=YMEAN(I)-TOP2
   R = SQERE(I) - YMEAN(I)
   Z2 = R/DF2
   Z3=XBAR/Z2
   SKIP=SKIP+1.
   IF (30-=SKIP) 70,55,58
55 WRITE OUTPUT TAPE 6,57
57 FORMAT(77H1 VARIABLE S.S.BETWEEN S.S.RESIDUAL DF1
   1 DF2 F-RATIO )
   SKIP=0.0
58 WRITE OUTPUT TAPE 6,59,I,XBAR,R,DF1,DF2,Z3
59 FORMAT(1H0I12,2F16.3,2F10.0,F10.3)
60 CONTINUE
   REWIND 3
   GO TO 1

*ASSEMBLE*PUNCH OBJECT
   REM INPUT
   ORG 0
   PGM
   PZE END,0,1
   PZE
   BCD 1INPUT
   PZE ENT
   REM
   ORG 0
   REL

ERROR
   BCD 1ERROR
   ENT RTD 7
   CLA 7,4
   STA CPY
   SXD AX1,1
   LXA A=1,1
   CPY CPY **,1
   D=12 TXI **3,1,1
   TSX ERROR,4
   TSX ERROR,4
   TXL CPY,1,12
   LXD AX1,1
   TRA 1,4
   A=1 PZE 1
   AX1 BSS 1
   END SYN *
   END

*DATA
*COMPILE FORTRAN, PRINT SAP, PUNCH OBJECT, EXECUTE, DUMP

DIMENSION A(200,19), SUM(200), SQUARE(200), COUNT(200), NX(25), NY(25), 
IXMEAN(200), DEVNY(200), YMEAN(200), DEVNX(200), LC(200), MLAST(200), 
1SOM(200), SQREA(200)
EQUIVALENCE (MLAST, YMEAN), (LC, DEVNY)
1 READ INPUT TAPE 7, 2(N, LY, LX)
2 FORMAT(316)
   REWIND 3
   MFIRST=19
   M1=MFIRST
   READ INPUT TAPE 7, 3(NX(I), I=1, LY)
3 FORMAT(24, I3)
   READ INPUT TAPE 7, 3(NY(I), I=1, LY)
   READ INPUT TAPE 7, 4(COUNT(I), I=1, LX)
4 FORMAT(24F3, 0)
   TOTAL=0, 0
   DO 7 I=1, LX
   J=COUNT(I)
   TOTAL=TOTAL+COUNT(I)
   LC(I)=(J/MFIRST)
   MLAST(I)=(MFIRST*LC(I))
   IF (J-MLAST(I)) 6, 5*6
5 MLAST(I)=MFIRST
   GO TO 7
6 LC(I)=LC(I)+1
   MLAST(I)=J-MLAST(I)
7 CONTINUE
   K=1
   K1=2
   CALL INPUT
   READ INPUT TAPE 7, 8(A(I*K), I=1, N)
8 FORMAT(
   1
   )
   DO 32 LA=1, LX
   M1=MFIRST
   LG=LC(LA)
   CA=COUNT(LA)*(COUNT(LA)-1)
   DO 9 J=1, N
   SUM(J)=0, 0
   SQUARE(J)=0, 0
9 CONTINUE
   DO 20 J=1, LG
   IF (LG-J) 10, 10, 11
10 M1=MLAST(LA)
11 DO 12 K=K1, M1
   READ INPUT TAPE 7, 8(A(I*K), I=1, N)
12 CONTINUE
   K1=1
   DO 20 L=1, LY
   NFIRST=NX(L)
   NLAST=NY(L)
   DO 20 I=NFIRST, NLAST
   DO 20 K=1, M1
   17 SUM(I)=SUM(I)+A(I,K)
   SQUARE(I)=SQUARE(I)+(A(I,K)**2)
20 CONTINUE
   DO 32 L=1, LY
   NFIRST=NX(L)
   NLAST=NY(L)

73
DO 30 I=NFIRST,NLAST
    CB=COUNT(LA)*SQUARE(I)-(SUM(I)**2)
23 XMEAN(I)=SUM(I)/COUNT(LA)
25 IF (CB) 26,26,27
26 DEVNX(I)=0.0
    GO TO 30
27 DEVNX(I)=SQRT(CB/CA)
30 CONTINUE
    WRITE TAPE 3,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
32 CONTINUE
    END FILE 3
    REWIND 3
    READ INPUT TAPE 7,33,LA
33 FORMAT(I10)
    WRITE OUTPUT TAPE 6,34,LA
34 FORMAT(38HO ANALYSIS OF VARIANCE JOB NUMBER 110 )
    LX=LX-1
    DF1 = LX - 1
    GROUP=LX
    DF2=TOTAL-GROUP
    SKIP=29*
    DO 35 I=1,N
    LC (I) = 1
    SQERE (I) = 0.0
    SOM (I) = 0.0
35 YMEAN (I) = 0.0
    DO 50 J = 1, LX
    DO 36 L=1, LY
    NFIRST = NX (L)
    NLAST = NY (L)
    READ TAPE 3,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
36 CONTINUE
    DO 50 L = 1, LY
    NFIRST = NX (L)
    NLAST = NY (L)
    DO 50 I = NFIRST, NLAST
      IF (LC(I)) 50,50,37
    37 IF (DEVNX(I)) 38,38,39
    38 LC(I) = -1
    GO TO 50
39 YMEAN(I)=YMEAN(I)+((SUM(I)**2)/COUNT(J))
    SOM (I) = SOM (I) + SUM (I)
    SQERE (I) = SQERE (I) + SQUARE (I)
50 CONTINUE
    DO 70 L = 1, LY
    NFIRST = NX (L)
    NLAST = NY (L)
    DO 70 I = NFIRST, NLAST
      IF (LC(I)) 43,43,54
43 SKIP=SKIP+1
      IF (30*-SKIP) 70,44,45
44 WRITE OUTPUT TAPE 6,57
    SKIP=0.0
45 WRITE OUTPUT TAPE 6,47,I
47 FORMAT(1H0I9,40H PLEASE CHECK YOUR DATA )
    GO TO 70
54 TOP2=(SOM(I)**2)/TOTAL
    XBAR=YMEAN(I)-TOP2

74
R = SQERE (I) - YMEAN (I)
Z2 = R/DF2
Z3=XBAR/Z2
SKIP=SKIP+1
IF (30.=SKIP) 70,55,58
55 WRITE OUTPUT TAPE 6,57
57 FORMAT(77H1) VARIABLE S*S.BETWEEN S*S.RESIDUAL DF1
   1 DF2 F-RATIO )
   SKIP=0
58 WRITE OUTPUT TAPE 6,59,1,XBAR,R,DF1,DF2,Z3
59 FORMAT(1H0I12,2F16.3,2F10.0,F10.3)
70 CONTINUE
REWRITE 3
GO TO 1
*ASSEMBLE PUNCH OBJECT
  REM INPUT
  ORG 0
  PGM
  PZE END*0,1
  PZE
  BCD 1INPUT
  PZE END
  REM
  ORG 0
  REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7*4
STA CPY
SX SD AX1*1
LXA A=1*1
CPY CPY **,1
D=12 TXI **,3+1*1
TSX ERROR*4
TSX ERROR*4
TXL CPY*1*12
LXD AX1*1
TRA 1*4
A=1 PZE 1
AX1 BSS 1
END SYN *
DATA
VI. CORRELATION PROGRAMS

1. Inter-Correlation Program

A. DESCRIPTION OF THE PROGRAM

1. The program is classified into two categories
   a. R 1A (missing data)
   b. R 2A (no-missing data)

2. Program print out (see Appendix E)

3. Special codes built into the program

   To avoid stoppage of the machine due to any unforeseen circumstances, the following features are incorporated.

   a. Whenever the variance of either of the two variables is exactly zero the program prints $r_{xy} = 9.999$

   \[ t_{xy} = 9.999 \]

   b. Whenever the degrees of freedom of the matched variables is zero the program prints out $r_{xy} = 9.999$

   \[ t_{xy} = 9.999 \]

   c. Whenever $r_{xy} = 1.00$ the program prints out $t_{xy} = 8.888$.

B. SPECIAL FEATURES

1. The input sub-routine reads in the FORMAT of the data cards. The program therefore is capable of handling data with any FORMAT.
2. The following feature, which could be called versatility, is best illustrated by an example. This feature has been found to be very useful when several independent research groups are working on a project. For example, at the Schizophrenia and Psychopharmacology Research Project, data are collected on several variables in each of the following sections: Biochemistry (50), Psychology (100), Social History (75), Physiology (75), Psychiatry (70), and Animal Physiology (20). The numbers in parentheses indicate approximate number of variables studied in each section. It was found necessary to have a program to handle data the following ways:

a. To determine inter-correlations either on all variables or on a selected group of consecutive variables in a punched matrix. See Fig. 5.

b. To determine inter-correlations on one set of consecutive variables and cross-correlations between these and a second set of consecutive variables in the same matrix. No break between the two sets is accepted. The inter-correlations for the second set of variables are not computed. See Fig. 5.

3. The program has been designed so that significant 't' values (e.g. ≥ 1.960) can be printed. This feature is found to be very useful when a research group is handling a large number of variables. See Appendix E.
C. DISCUSSION

1. Pearson product-moment correlation coefficient

\[ r_{xy} = \frac{L_{xy}}{\sqrt{(L_{xx} L_{yy})}} \]

Where,

\[ L_{xy} = N \bar{XY} - \bar{X} \bar{Y} \]
\[ L_{xx} = N \bar{X}^2 - (\bar{X})^2 \]
\[ L_{yy} = N \bar{Y}^2 - (\bar{Y})^2 \]

In the case of the missing data program \( N \), \( \bar{XY} \), \( \bar{X} \), \( \bar{Y} \), \( \bar{X}^2 \), \( \bar{Y}^2 \), refer to matching observations. The particular calculation loop is skipped whenever missing data are detected by the program in either one or both of the variables.

2. Fisher's t for the correlation coefficient

\[ t_r = \frac{r_{xy} \sqrt{(N-2)}}{\sqrt{1 - r_{xy}^2}} \]

3. Degrees of freedom = \( N-2 \)

The corresponding degrees of freedom is computed for each pair of variables. The value of \( N \) is the total number of matching observations, as discussed above.
D. LIMITATIONS

<table>
<thead>
<tr>
<th>Program Category</th>
<th>MAXIMUM NUMBER OF VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8k memory</td>
</tr>
<tr>
<td>R 1A</td>
<td>797*</td>
</tr>
<tr>
<td>R 2A</td>
<td>1100*</td>
</tr>
</tbody>
</table>

* These are estimates.

E. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Some running time estimates, based on actual runs, are listed below.

<table>
<thead>
<tr>
<th>Program Category</th>
<th>NUMBER OF VARIABLES</th>
<th>NUMBER OF OBSERVATIONS</th>
<th>RUNNING TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>R 1A</td>
<td>120</td>
<td>200</td>
<td>30 min.</td>
</tr>
<tr>
<td>R 2A</td>
<td>130</td>
<td>300</td>
<td>17 min.</td>
</tr>
</tbody>
</table>

Computation time for R 1A is .1 to .2 seconds per r, depending on the size of the matrix. The R 2A program is 3 to 5 times faster. (See also G. 1. below).

F. TEST PROBLEMS

Test problems have been designed so that they cover possible occurrences, and they are itemized for the categories of programs discussed above.

See examples in Appendix E.
G. OUTPUT

1. Printout for R 1A:

   Variable j
   
   Variable j; r_{ij}; df_{ij}; t_{r_{ij}}; var. j + 1; r_{i(j + 1)}; df_{i(j + 1)};
   
   t_{r_{i(j + 1)}}; \ldots; var. n; r_{in}; df_{in}; t_{rin}.

   In example A, i = 1,2, \ldots 279, j = i + 1, \ldots, 280.

   In example B, i = 25,26, \ldots 178, j = i + 1, \ldots, 179.

   (See appendix E.)

   Option 't'-value yes will, in addition to above matrix, repeat the
   parts of it where t_{r}’s \geq a specified value (e.g. 1.96).*

2. Printout for R 2A:

   Same as for R 1A, except for degrees of freedom which here is printed
   only once on the same line as variable i.

3. Card output (optional)

   Upper half off-diagonal correlation matrix (see Appendix G). The
   Format is $12F6.3$.

---

*If option t-value yes is used, running time per r has to be estimated between 
.3 and .5 second. This estimate is for the missing data program. The no-
missing data program will be 3 to 5 times faster.
Appendix E

1. ARRANGEMENT OF CARDS

(for details, see program print out)

a. Using FORTRAN deck (Fig. 4)

The FORTRAN deck represents the written program. Although this deck may be used together with the data cards, a better procedure is to run only a short deck of test data cards with the FORTRAN deck. The computer "compiles" the program, computes the data as required, prints out the results, and "cuts" binary cards. The binary deck can then be used, instead of the Fortran deck, in all subsequent runs, provided that the maximum dimensions given are sufficient. New compilation of the program is not needed when using the binary deck. Some computing centers restrict computing time, after compilation, to a minimum.

1. I-D (identification) cards. The same information is punched in both cards: name of scientist, computing center project number, estimated execution time (minutes), estimated number of output pages, estimated card output (if any).

2. Card to instruct the computer to compile Fortran. Usually punched starting in column 1: "*COMPILE FORTRAN, PRINT SAP, PUNCH"
OBJECT, EXECUTE, DUMP

3. Fortran program deck (with any change in dimension cards and MFIRST = *** or ** card, if required).

4. Card to assemble SAP. Usually punched starting in column 1: *ASSEMBLE, PUNCH OBJECT

5. Sub-routine deck

6. Card to inform the computer to start reading control cards, format statement, and data. Usually punched starting in column 1:

*DATA

7. Control card

8. Format statement. See Appendix A.

9. Test data cards sorted in matrix form.

10. Job number card. See Appendix A.

b. Using binary deck (Fig. 4)

1. I-D cards

2. Card punched (starting in column 1:)

*EXECUTE, DUMP

3. Binary program deck

4. Binary sub-routine cards
2. EXAMPLE PROBLEMS

Inter-correlation R 1A (missing data), and

Inter-correlation R 2A (no-missing data)

<table>
<thead>
<tr>
<th>Example</th>
<th>NUMBER OF VARIABLES</th>
<th>NUMBER OF OBSERVATIONS</th>
<th>'t'-VALUE</th>
<th>CARD OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>280</td>
<td>369</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>B</td>
<td>185</td>
<td>206</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>C</td>
<td>96</td>
<td>200</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>D</td>
<td>31</td>
<td>100</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

Note: 1. The number of observations could be very much larger, say 2000. Missing data are also to be counted as observations, although during computation the program omits counting missing data.

**The number of control cards used in other programs is given in the respective appendices.
2. Maximum number of variables these programs can handle are discussed under 'Limitations'.

3. 't' value option yes:

In addition to the regular correlation matrix, the program will print another matrix including only those correlations for which the corresponding 't'-values are, for example, > 1.96. This makes it easier to find the significant correlations. It should be noted that r and t = 9.999 or t = 8.888 indicates unusual situations (see special codes).

3. DIMENSION CARDS FOR THE FORTRAN DECK:

   a. Missing Data R 1A (See example problems A, B, C, and D)

Card 1: (Example A)

DIRECTORY SUM (500), SOM (300), SQUARE (300), SQERE (300), PROD (300)

Card 2:

1A (500,12), TOTAL (300), JA (300,12)

Note: 500+300+300+300+300+(300 x 12) + 300 < 5580

Card 1: (Example B)

DIRECTORY SUM (200), SOM (200), SQUARE (200), SQERE (200), PROD (200)

Card 2:

1A (200,21), TOTAL (200), JA (200,21)

Note: 200+200+200+200+200+(200 x 21) + 200 < 5580
Card 1: (Example C)

DIMENSION SUM (100), SOM (100), SQUARE (100), SØERE (100), PROD (100)

Card 2:

1A (100,49), TOTAL (100), JA (100,49)

Note: 100+100+100+100+100+(100 x 49) + 100 ≤ 5580

Card 1: (Example D)

DIMENSION SUM (50), SOM (50), SQUARE (50), SØERE (50), PROD (50)

Card 2:

1A (50,105), TOTAL (50), JA (50,105)

Note: 50+50+50+50+50+(50 x 105) + 50 ≤ 5580

5580 locations refer to an 8k memory. When using a 32k memory corresponding changes should be made before compilation. Card 2 is a continuation card.

A, B, C, and D refer to the example problems given above. Example A called for 280 variables and 361 observations. The dimension cards are punched to make it possible to handle a maximum of 300 variables. The entries in the dimension cards mean: store 6 times information on 300 variables (sum 300), (som 300), (square 300), (søere 300), (prod 300), (total 300). The double entries are required because of missing data. This requires 6 x 300 = 1800 locations in the computer core. There are 5580 locations available for computing. Thus 5580 minus 1800 leaves 3780 locations which divided by 300 variables gives 12 observations to be computed at one time (the remainder, 180, cannot be used). The terms 1A (300,12) and JA
(300,12) refer to this condition. The computer will thus calculate 12 observations on all variables in one cycle and repeat the calculation as many times as required, in this case \( \frac{569}{12} = 30 \times 12 + 1 \times 9 = 31 \) times.

The maximum number of variables punched into the dimension cards is determined by the amount of variables generally used in analysis. If this number is appreciably smaller than in example A, it is more advantageous to punch dimension cards to handle a maximum of, say, 150 or 100 variables. This makes it possible for the computer to store more observations for each computing cycle. (See examples B, C, and D).

If the number of variables varies largely from analysis to analysis, the best thing to do is to "compile" several programs and get resulting binary cards, which then can be used for jobs of different magnitudes.

b. No missing data R 2A

The method is the same as in R 1A.

Example:

A. (1 card only)

DIMENSION SUM (300), R (300,15), SIGMA (300), PROD (300), JA (300,15)

Note: \( 300 + (300 \times 15) + 300 + 300 \leq 5600 \)

B, C, and D follow the same pattern.

For details see program print-out.
4. **MFIRST = *** or ** CARD**

See program print-out.

Whenever the program is compiled for a different dimension, the number of observations computed on all variables in one cycle will also change.

See cards 2 in the above examples (R 1A).

In these cases:

\[
\begin{align*}
\text{MFIRST} & = 12 \quad (A) \\
\text{MFIRST} & = 21 \quad (B) \\
\text{MFIRST} & = 49 \quad (C) \\
\text{MFIRST} & = 105 \quad (D)
\end{align*}
\]

Corresponding change should be made before compilation.

5. **CONTROL CARDS**

Inter-correlation missing data program (R 1A) and

Inter-correlation no-missing data program (R 2A)

a. Control card no. 1

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-6</td>
</tr>
<tr>
<td>A</td>
<td>+00280</td>
</tr>
<tr>
<td>B</td>
<td>+00185</td>
</tr>
<tr>
<td>C</td>
<td>+00096</td>
</tr>
<tr>
<td>D</td>
<td>+00031</td>
</tr>
</tbody>
</table>
Note:  1. See Fig. 5.

2. D refers to missing data entry on cards.

3. +1 calls for card output, -1 no card output.

4. +196 calls for 't' value option yes and -196 calls for 't' value option no. (In this example a t of \( > 1.96 \) is called for)

5. Note the entries 37-44 (can be left blank in the case of no-missing data program).

6. FORMAT STATEMENT

   See Appendix A under means and standard-deviations program.

7. DATA CARDS

   See Appendix A under means and standard-deviations program.

8. JOB NUMBER CARD

   See Appendix A under means and standard-deviations program.
ANY SPECIAL INSTRUCTIONS TO THE COMPUTER OPERATOR (e.g. PLEASE USE FULL LENGTH BINARY TAPES, OR PRINT TWO COPIES OF OUTPUT, ETC.) SHOULD BE PUNCHED OR WRITTEN ON A CARD AND PLACED IN FRONT OF THE 1-D CARDS.
EXAMPLES: R1A, R2A
INPUT MATRICES A, B, C, AND D

FIG. 5
*COMPILE FORTRAN PRINT SAP PUNCH OBJECT EXECUTE DUMP
DIMENSION SUM(155), SOM(155), SQUARE(155), SQERE(155), PROD(155),
SA(155), BA(155), TOTAL(155), JA(155), BA(155), 30
1 READ INPUT TAPE 7*2*N, NFIRST, NLAST, NEND, COUNT, TVALE, CARD, D
2 FORMAT(4I6*F6*.0*F4*.2*F2*.0*F8*.3)
EQUIVALENCE (JA*A)
REWIND 3
REWIND 4
M1 = MFIRST
JUMP = COUNT
K = 1
CALL INPUT
READ INPUT TAPE 7*3*(A(I*K), I=1*N)
3 FORMAT(
1 K1=2
LG=(JUMP/MFIRST)
MLAST=(M1*LG)
IF (JUMP-MLAST) 5*,4*,5
4 MLAST=MFIRST
GO TO 7
5 LG=LG+1
MLAST=JUMP-MLAST
7 ASSIGN 21 TO LOOP
ASSIGN 28 TO NEW
ASSIGN 36 TO JUMP
DO 50 LA=1*LG
IF (LG-LA) 8*,8*,10
8 M1=MLAST
10 DO 11 K=K1*M1
READ INPUT TAPE 7*3*(A(I*K), I=1*N)
11 CONTINUE
K1=1
GO TO LOOP*(21,22)
21 ASSIGN 22 TO LOOP
JTAPE=4
ITAPE=3
GO TO 23
22 ASSIGN 21 TO LOOP
JTAPE=3
ITAPE=4
23 DO 44 I=NFIRST, NLAST
GO TO NEW*(28,32)
28 DO 30 J=NEND
SOM(J)=0.0
SUM(J)=0.0
TOTAL(J)=0.0
SQUARE(J)=0.0
SQERE(J)=0.0
PROD(J)=0.0
30 CONTINUE
32 L1=1+1
GO TO JUMP*(34,36)
34 READ TAPE ITAPE*((PROD(J)*SOM(J)*SUM(J)*SQERE(J),
ISQUARE(J)*TOTAL(J)), J=L1, NEND)
36 DO 42 L=L1, NEND
DO 42 K=1*M1
IF (A(I*K)-D) 38*,42*,38

92
38 IF (A(I,K)-D) 40,42,40
40 PROD(L)=PROD(L)+(A(I,K)*A(L,K))
   SOM(L)=SOM(L)+A(I,K)
   SUM(L)=SUM(L)+A(L,K)
   SQERE(L)=SQERE(L)+(A(I,K)**2)
   SQUARE(L)=SQUARE(L)+(A(L,K)**2)
   TOTAL(L)=TOTAL(L)+1*
42 CONTINUE
   WRITE TAPE JTAPE *((PROD(J),SOM(J),SUM(J),SQERE(J),
                     SQUARE(J),TOTAL(J)),J=L1,NEND)
44 CONTINUE
   REWIND JTAPE
   END FILE JTAPE
   REWIND JTAPE
49 ASSIGN 32 TO NEW
   ASSIGN 34 TO JUMP
50 CONTINUE
   READ INPUT TAPE 7,52,NEW
52 FORMAT(110)
   WRITE OUTPUT TAPE 6,53,NEW
53 FORMAT(38H1 CORRELATION MATRIX JOB NUMBER 110 )
   DO 54 L=NFIRST,NEND
      JA(L+1)=L
54 CONTINUE
   DO 90 I=NFIRST,NLAST
      L1=I+1
      READ TAPE JTAPE *((PROD(J),SOM(J),SUM(J),SQERE(J),
                     SQUARE(J),TOTAL(J)),J=L1,NEND)
      DO 80 L=L1,NEND
      IF (TOTAL(L)-2*) 60,60,61
50 PROD(L)=9.999
52 SUM(L)=9.999
   TOTAL(L)=TOTAL(L)-2*
   GO TO 80
51 CA=((TOTAL(L)*(TOTAL(L)-1*))
52 CC=((TOTAL(L)*SQUARE(L))-(SUM(L)**2))
53 CB=((TOTAL(L)*SQERE(L))-(SOM(L)**2))
54 IF (CB) 60,60,64
55 IF (CC) 60,60,66
56 CB=SQRT(CB/CA)
   CC=SQRT(CC/CA)
57 CB=CA*CB*CC
      CC=((TOTAL(L)*PROD(L))-(SOM(L)*SUM(L)))
58 PROD(L)=CC/CB
   CC=PROD(L)**2
   TOTAL(L)=TOTAL(L)-2*
   IF (1.-CC) 71,71,72
   SUM(L)=8.888
   GO TO 80
61 CA=(TOTAL(L)*(TOTAL(L)-1*))
62 CC=((TOTAL(L)*SQUARE(L))-(SUM(L)**2))
63 CB=((TOTAL(L)*SQERE(L))-(SOM(L)**2))
64 IF (CB) 60,60,64
65 IF (CC) 60,60,66
66 CB=SQRT(CB/CA)
   CC=SQRT(CC/CA)
67 CB=CA*CB*CC
   CC=((TOTAL(L)*PROD(L))-(SOM(L)*SUM(L)))
68 PROD(L)=CC/CB
   CC=PROD(L)**2
   TOTAL(L)=TOTAL(L)-2*
   IF (1.-CC) 71,71,72
69 SUM(L)=8.888
   GO TO 80
70 SUM(L)=(PROD(L)/(SQRT(1.-CC)))*(SQRT(TOTAL(L)))
72 CONTINUE
   WRITE OUTPUT TAPE 6,6,1,*((JA(L,1),PROD(L),TOTAL(L),SUM(L)),L=L1,
                     1,NEND)
76 FORMAT(1H017/((3(1H 110,F9.3,F9.0,F9.3))*)
   IF (CARD) 83,90,81
77 PUNCH 82,(PROD(L),L=L1,NEND)
78 FORMAT(12F6.3)
79 IF (TVALUE) 90,90,84
84 L=0
85 DO 87 M=L+1
86 L=L+1
87 CONTINUE
88 WRITE OUTPUT TAPE 6,6,1,((JA(M),PROD(M),TOTAL(M),SUM(M)),M=1,L)
90 CONTINUE
*ASSEMBLE PUNCH OBJECT
*REM INPUT
*ORG 0
*PGM
*PZE END 0+1
*PZE
*BCD 1INPUT
*PZE ENT
*REM
*ORG 0
*REL
*ERROR BCD 1ERROR
*ENT RTD 7
*CLA 7+4
*STA CPY
*SXD AX1+1
*LX A=A+1
*CPY **+1
*D=12 TXI **3+1+1
*TSX ERROR+4
*TSX ERROR+4
*TXL CPY+1+12
*LXD AX1+1
*TRA 1+4
*A=1 PZE 1
*AX1 BSS 1
*END SYM *
*END
*DATA
1 READ INPUT TAPE 7,2,N,NFIRST,NLAST,NEND,COUNT,TVALUE,CARD
2 FORMAT(4I6,F6.0,F4.2,F2.0)
   REWIND 3
   REWIND 4
   MFIRST=33
   M1=MFIRST
   K=1
   CALL INPUT
   READ INPUT TAPE 7,3,(R(I,K),I=1,N)
3 FORMAT(
   1)
   K1=2
   JUMP=COUNT
   LG=(JUMP/MFIRST)
   MLAST=(MFIRST*LG)
   IF (JUMP-MLAST) 5,4,5
4 MLAST=MFIRST
   GO TO 7
5 LG=LG+1
   MLAST=JUMP-MLAST
7 DO 12 J=NFIRST,NEND
   SIGMA(J)=0.0
   SUM(J)=0.0
12 CONTINUE
   ASSIGN 21 TO LOOP
   ASSIGN 28 TO NEW
   ASSIGN 36 TO JUMP
   DO 50 LA=1*LG
   IF (LG-LA) 14,14,16
14 M1=MLAST
16 DO 18 K=K1,M1
   READ INPUT TAPE 7,3,(R(I,K),I=1,N)
18 CONTINUE
   K1=1
   DO 20 I=NFIRST,NEND
   DO 20 K=1*M1
   SUM(I)=SUM(I)+R(I,K)
   SIGMA(I)=SIGMA(I)+(R(I,K)**2)
20 CONTINUE
   GO TO LOOP*(21,22)
21 ASSIGN 22 TO LOOP
   JTAPE=4
   ITAPE=3
   GO TO 23
22 ASSIGN 21 TO LOOP
   JTAPE=3
   ITAPE=4
23 DO 44 I=NFIRST,NLAST
   GO TO NEW*(28,32)
28 DO 30 J=I,NEND
   PROD(J)=0.0
30 CONTINUE
32 L1=I+1
   GO TO JUMP*(34,36)
34 READ TAPE ITAPE*(PROD(J),J=L1,NEND)
DO 42 L=L1*NEND
DO 42 K=1*M1
40 PROD(L)=PROD(L)+(R(I,K)*R(L,K))
42 CONTINUE
   WRITE TAPE JTAPE,(PROD(J),J=L1,NEND)
44 CONTINUE
   REWIND JTAPE
   END FILE JTAPE
   REWIND JTAPE
49 ASSIGN 32 TO NEW
   ASSIGN 34 TO JUMP
50 CONTINUE
   READ INPUT TAPE 7*52,NEW
52 FORMAT(I10)
   WRITE OUTPUT TAPE 6*53,NEW
53 FORMAT(38H1, CORRELATION MATRIX, JOB NUMBER I10 )
   CA=COUNT*(COUNT-1*)
   DF=COUNT-2*
   SDF=SQRT(DF)
   DO 55 I=NFIRST,NEND
      JA(I+1)=I
      CB=(COUNT*SIGMA(I)-(SUM(I)**2))
      SIGMA(I)=SQRT(CB/CA)
55 CONTINUE
   DO 90 I=NFIRST,NLAST
      L=I+1
      READ TAPE JTAPE,(PROD(J),J=L1,NEND)
   DO 80 L=L1,NEND
      TOP=(COUNT*PROD(L)-(SUM(I)*SUM(L)))
      BELOW=CA*(SIGMA(I)*SIGMA(L))
      IF (BELOW) 60,60,65
   60 PROD(L)=9*999
      TEE(L)=9*999
      GO TO 80
   65 PROD(L)=TOP/BELLOW
      CC=PROD(L)**2
      IF (I-CC) 71,71,72
   71 TEE(L)=8*888
      GO TO 80
   72 TEE(L)=(PROD(L)/(SQRT(1-CC)))*SDF
   80 CONTINUE
      WRITE OUTPUT TAPE 6*61,DF,((JA(L),1),PROD(L),TEE(L)),L=L1,NEND)
   6 FORMAT(I017,F8.0/(4(I8,F10.3,F10.4)))
      IF (CARD) 83,90,81
   81 PUNCH 82,(PROD(L),L=L1,NEND)
   82 FORMAT(12F6.3)
   83 IF (TVVALUE) 90,90,84
   84 L=0
   DO 87 M=L1,NEND
      CC=ABS(TEE(M))
      IF (CC-TVVALUE) 87,86,86
   86 L=L+1
   PROD(L)=PROD(M)
   TEE(L)=TEE(M)
   JA(L+2)=JA(M+1)
   87 CONTINUE
   IF (L) 90,90,88
   88 WRITE OUTPUT TAPE 6*61,DF,((JA(M),2),PROD(M),TEE(M)),M=1,L)
CONTINUE
REWIND JTAPE
GO TO 1
*ASSEMBLE,PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END,0*1
PZE
BCD 1INPUT
PZE ENTER
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7,4
STA CPY
SXD AX1,1
LXA A=1*1
CPY CPY ***1
D=12 TXI **+3,1*1
TSX ERROR+4
TSX ERROR+4
TXL CPY+1*12
LXD AX1+1
TRA 1,4
A=1 PZE 1
AX1 BSS 1
END SYN *
END

*DATA
2. Cross-Correlation Program

A. DESCRIPTION OF THE PROGRAM

1. The program is classified into two categories
   a. R 1B (missing data)
   b. R 2B (no-missing data)

2. Program print out (see Appendix)

3. Special codes
   See inter-correlation programs.

B. SPECIAL FEATURES

1. See inter-correlation programs.

2. The program has been designed to
   a. Compute cross-correlations between the variables in two sections
      of a punched matrix, or between the variables in two matrices, or
   b. Compute correlations for one or more variables with all the rest
      in a punched matrix. See Fig. 6.
   c. The program has been designed so that significant 't' values can
      be printed. This feature is found to be very useful when a research
      group is handling a large number of variables. See intercorrelation
      program.

C. DISCUSSION

See inter-correlation programs.
D. LIMITATIONS

<table>
<thead>
<tr>
<th>Program</th>
<th>Maximum Number of Variables on a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>8k memory</td>
</tr>
<tr>
<td>R 1B</td>
<td>797*</td>
</tr>
<tr>
<td>R 2B</td>
<td>1100*</td>
</tr>
</tbody>
</table>

*These are estimates.

E. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Some running time estimates, based on actual runs are listed below.

<table>
<thead>
<tr>
<th>Program</th>
<th>NUMBER OF VARIABLES</th>
<th>NUMBER OF OBSERVATIONS</th>
<th>RUNNING TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R 1B</td>
<td>70 vs 115</td>
<td>200</td>
<td>35 min.</td>
</tr>
<tr>
<td>R 2B</td>
<td>70 vs 120</td>
<td>300</td>
<td>17 min.</td>
</tr>
</tbody>
</table>

Computation time for R 1B is .1 to .2 seconds per r, depending on the size of the matrix. The R 2B program is > to 5 times faster.*

F. TEST PROBLEMS

Test problems have been designed so that they cover possible occurrences, and they are itemized for the categories of programs discussed above.

See Appendix F.

*If option t-value yes (see example problems) is used the estimated computing time per r is .3 to .5 seconds (R 1B).
G. OUTPUT

1. Print-out. The method of printing out is approximately the same as for intercorrelations. Consider first example A (Fig. 6). The print-out will consist of 80 submatrices: variable 1 with 81, 1 with 82, ..., 1 with 280; variable 2 with 81, 2 with 82, ..., 2 with 280; ...; variable 80 with 82, ..., 80 with 280. Next consider example B. Here the count will start at variable 5 with 61, ..., through 173. The last matrix will be for variable 50 with 61, ... through 173.

2. Card output (optional): The submatrices are punched out in format 12F6.3.
Appendix F

1. ARRANGEMENT OF CARDS

(for details, see program print-out)

a. Using Fortran deck (See Fig. 4)

b. Using binary deck (See Fig. 4)

See inter-correlation program (Appendix E).

2. EXAMPLE PROBLEMS

Cross-correlation missing data program (R 1B) and

Cross-correlation no-missing data program (R 2B)

<table>
<thead>
<tr>
<th>Example</th>
<th>Number of Variables</th>
<th>Number of Observations</th>
<th>'t' value</th>
<th>Card Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sec-1</td>
<td>Sec-2</td>
<td>Total</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>80</td>
<td>200</td>
<td>280</td>
<td>250</td>
</tr>
<tr>
<td>B</td>
<td>60</td>
<td>120</td>
<td>180</td>
<td>350</td>
</tr>
<tr>
<td>C</td>
<td>40</td>
<td>60</td>
<td>100</td>
<td>600</td>
</tr>
<tr>
<td>D</td>
<td>20</td>
<td>30</td>
<td>50</td>
<td>95</td>
</tr>
</tbody>
</table>

Note: See note under inter-correlation program, example problems.

3. DIMENSION CARDS FOR THE FORTRAN DECK

The dimension cards are discussed in great detail under inter-correlation programs. The user is advised to follow through the steps. The procedure to be followed is the same.
4. MFIRST = *** or ** CARD

See Appendix E.

5. CONTROL CARDS

Cross-correlation missing data program R 1B, and

Cross-correlation no-missing data program R 2B

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-5</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td>A</td>
<td>+00280</td>
</tr>
<tr>
<td>B</td>
<td>+00173</td>
</tr>
<tr>
<td>C</td>
<td>+00100</td>
</tr>
<tr>
<td>D</td>
<td>+00050</td>
</tr>
</tbody>
</table>

Note: 1. See Fig. 6

2. See note under inter-correlation program, control card no. 1

3. In example B (see also Fig. 6) N could be stated = 180.

b. Control card no. 2

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-5</td>
</tr>
<tr>
<td>A</td>
<td>061</td>
</tr>
<tr>
<td>B</td>
<td>061</td>
</tr>
<tr>
<td>C</td>
<td>025</td>
</tr>
<tr>
<td>D</td>
<td>018</td>
</tr>
</tbody>
</table>
c. Control card no. 3

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>280</td>
</tr>
<tr>
<td>B</td>
<td>173</td>
</tr>
<tr>
<td>C</td>
<td>025</td>
</tr>
<tr>
<td>D</td>
<td>018</td>
</tr>
</tbody>
</table>

Note: 1. LY in control card no. 1 indicates the number of times the calculations are to be repeated. See entries for C and D in control cards 2 and 3.

2. If two separate matrices are used, the procedure in example B should be followed. Let us assume, for example, that the format is 12F6.3, that is 12 variables per card, each word occupying 6 card columns (xxxxxx, 3 decimals). In this case five decks of cards are needed to complete punching the first matrix of 50 variables. The fifth deck will thus contain data on only two variables. Ten words will be blank in this deck. The second matrix of 120 variables (61 through 180) is complete with ten decks of cards, each deck containing data on 12 variables. The two matrices combined thus have 170 variables. When the decks for the two matrices are combined, the fifth deck has to be treated as if it were complete, and the blank words (51 through
60) counted as variables. These blank variables are omitted during computation. The last variable to be included (in example B) is no. 173, and the total variable count should be \textit{stated} = 173. Total count could also be \textit{stated} = 180 (the full matrix). Control card 3 will anyway stop calculations beyond 173.

6. FORMAT STATEMENT

See Appendix A under means and standard-deviations program.

7. DATA CARDS

See Appendix A under means and standard-deviations program.

8. JOB NUMBER CARD

See Appendix A under means and standard-deviations program.
VARIABLES

1, 2, 3, 80, 81, 280

A

CROSS-CORRELATIONS FOR VARIABLES 1-80 VS. 81-280 ARE COMPUTED (NO INTER-CORRELATIONS WITHIN THE TWO SECTIONS WILL BE COMPUTED)

1
2
3

C

CORRELATIONS BETWEEN ANY ONE (HERE #25, 50 AND 99) VARIABLE AND EACH ONE OF THE REMAINING VARIABLES

1
2
3

D

CORRELATIONS BETWEEN ANY ONE VARIABLE (HERE #18 AND 26) AND EACH ONE OF A SELECTED GROUP OF THE REMAINING VARIABLES

1
2
3

EXAMPLES: R1B, R2B

INPUT MATRICES A, B, C, AND D

FIG. 6

105
*COMPILE FORTRAN, PRINT SAP, PUNCH OBJECT, EXECUTE, DUMP
DIMENSION SUM(200), SOM(200), SQUARE(200), SQERE(200), PROD(200),
1 A(250,17), TOTAL(200), JA(250,17), Nx(25), N(25)
EQUIVALENCE (JA, A)
1 READ INPUT TAPE 7, 2, N, NBEGIN, NEND, LY, COUNT, TVALUE, CARD, D
2 FORMAT (4I6, F6.0, F4.2, F2.0, F8.3)
  REWIND 3
  REWIND 4
  MFIRST = 17
  M1 = MFIRST
  K = 1
  READ INPUT TAPE 7, 3, (NX(I), I = 1, N)
3 FORMAT (2413)
  READ INPUT TAPE 7, 3, (NY(I), I = 1, N)
  CALL INPUT
  READ INPUT TAPE 7, 4, (A(I, K), I = 1, N)
4 FORMAT (1, )
  K1 = 2
  JUMP = COUNT
  LG = (JUMP / MFIRST)
  MLAST = (MFIRST * LG)
  IF (JUMP = MLAST) 7, 5, 7
5 MLAST = MFIRST
  GO TO 8
7 LG = LG + 1
  MLAST = JUMP = MLAST
8 ASSIGN 21 TO LOOP
  ASSIGN 28 TO NEW
  ASSIGN 36 TO JUMP
  NA = NEND - NBEGIN + 1
  K = NBEGIN - 1
  DO 50 LA = 1, LG
  IF (LG = LA) 9, 9, 10
9 M1 = MLAST
10 DO 11 K = K1, M1
  READ INPUT TAPE 7, 4, (A(I, K), I = 1, N)
11 CONTINUE
  K1 = 1
  GO TO LOOP *(21, 22)
21 ASSIGN 22 TO LOOP
  JTAPE = 4
  ITAPE = 3
  GO TO 23
22 ASSIGN 21 TO LOOP
  JTAPE = 3
  ITAPE = 4
23 DO 44 LE = 1, LY
  NFIRST = NX(LE)
  NLAST = NY(LE)
  DO 44 I = NFIRST, NLAST
  M = 0
  GO TO NEW *(28, 32)
28 DO 30 J = 1, NA
  SOM(J) = 0.0
  SUM(J) = 0.0
  TOTAL(J) = 0.0
  SQUARE(J) = 0.0
  SQERE(J) = 0.0
30 CONTINUE
PROCEDURE
30 CONTINUE
32 GO TO JUMP(34,36)
34 READ TAPE ITAPE,((PROD(J),SOM(J),SUM(J),SQERE(J),
1SQUARE(J),TOTAL(J)),J=1,NA)
36 DO 42 L=NBEGIN,NEND
M=M+1
DO 42 K=1,M
IF (A(I,K)-D) 38,42,38
38 IF (A(L,K)-D) 40,42,40
40 PROD(M)=PROD(M)+(A(I,K)*A(L,K))
SOM(M)=SOM(M)+A(I,K)
SUM(M)=SUM(M)+A(L,K)
SQERE(M)=SQERE(M)+(A(I,K)**2)
SQUARE(M)=SQUARE(M)+(A(L,K)**2)
TOTAL(M)=TOTAL(M)+1
42 CONTINUE
WRITE TAPE JTAPE,((PROD(J),SOM(J),SUM(J),SQERE(J),
1SQUARE(J),TOTAL(J)),J=1,NA)
44 CONTINUE
REWIND ITAPE
END FILE JTAPE
REWIND JTAPE
49 ASSIGN 32 TO NEW
ASSIGN 34 TO JUMP
50 CONTINUE
READ INPUT TAPE 7,52,NEW
52 FORMAT(110)
WRITE OUTPUT TAPE 6,53,NEW
53 FORMAT(38H1, CORRELATION MATRIX, JOB NUMBER 110 )
DO 54 M=1,NA
54 JA(M,1)=K+M
DO 90 L=1,LY
NFIRST=NX(LE)
NLAST=NY(LE)
DO 90 L=NFIRST,NLAST
READ TAPE JTAPE,((PROD(J),SOM(J),SUM(J),SQERE(J),
1SQUARE(J),TOTAL(J)),J=1,NA)
DO 80 M=1,NA
IF (TOTAL(M)-2) 60,60,61
60 PROD(M)=9.999
SUM(M)=9.999
TOTAL(M)=TOTAL(M)-2
GO TO 80
61 CA=(TOTAL(M)*(TOTAL(M)-1))
CC=((TOTAL(M)*SQUARE(M))-SUM(M)**2)
CB=((TOTAL(M)*SQERE(M))-SUM(M)**2)
62 IF (CB) 60,60,64
64 IF (CC) 60,60,66
66 CB=SQRT(CB/CA)
CC=SQRT(CC/CA)
CB=CA*CB*CC
CC=1.0*(TOTAL(M)*PROD(M)-(SOM(M)*SUM(M))
PROD(M)=CC/CB
TOTAL(M)=TOTAL(M)-2
CC=PROD(M)**2
IF (1=CC) 71,71,72
71 SUM(M)=8.888
GO TO 80
72 SUM(M) = (PROD(M) / (SQRT(1 - CC)) * (SQRT(TOTAL(M)))
80 CONTINUE
   WRITE OUTPUT TAPE 6, 6, 1, ((JA(M), PROD(M), TOTAL(M), SUM(M), M=1, NA)
6 FORMAT(1HO17/3(1H10,F9.3,F9.0,F9.3))
   IF (CARD) 83, 90, 81
81 PUNCH 82, (PROD(M), M=1, NA)
82 FORMAT(12F6.3)
83 IF (TVALUE) 90, 90, 84
84 L = 0
   DO 87 M = 1, NA
      CC = ABSF(SUM(M))
      IF (CC = TVALUE) 87, 86, 86
86 L = L + 1
      SUM(L) = SUM(M)
      PROD(L) = PROD(M)
      TOTAL(L) = TOTAL(M)
      JA(L, 2) = JA(M, 1)
87 CONTINUE
   IF (L) 90, 90, 88
88 WRITE OUTPUT TAPE 6, 6, 1, ((JA(M, 2), PROD(M), TOTAL(M), SUM(M), M=1, L)
90 CONTINUE
   REWIND JTAPE
   GO TO 1
*ASSEMBLE PUNCH OBJECT
REM INPUT
REM
PGM
PZE END=0, 1
BCD 1 INPUT
PZE
REM
ORG 0
REL
ERROR BCD 1 ERROR
ENT RTD 7
CLA 7, 4
STA CPY
SXD AX1, 1
LXA A=1, 1
CPY CPY **, 1
D=12
TXI *+3, 1, 1
TSX ERROR, 4
TSX ERROR, 4
TXL CPY, 1, 12
LXD AX1, 1
TRA 1, 4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
*COMPILE FORTRAN PRINT SAP PUNCH OBJECT EXECUTE DUMP

DIMENSION SUM(150),R(150,32),SIGMA(150),PROD(150),JA(150,32),
    1TE(150),NX(25),NY(25)
EQUIVALENCE (JA,R)

1 READ INPUT TAPE 7,2,N,NBEGIN,NEND,LY,COUNT,TVALUE,CARD
2 FORMAT(4I6,F6.0,F4,2,F2.0)
   REWIND 3
   REWIND 4
   MFIRST=32
   M1=MFIRST
   K=1
   READ INPUT TAPE 7,3,N(X(I),I=1,LY)
3 FORMAT(24I3)
   READ INPUT TAPE 7,3,N(Y(I),I=1,LY)
   CALL INPUT
   READ INPUT TAPE 7,4,(R(I,K),I=1,N)
4 FORMAT(1)
   K1=2
   JUMP=COUNT
   LG=(JUMP/MFIRST)
   MLAST=(MFIRST*LG)
   IF (JUMP-MLAST) 6,5,6
5 MLAST=MFIRST
   GO TO 7
6 LG=LG+1
   MLAST=JUMP-MLAST
7 DO 10 J=1,NEND
   SIGMA(J)=0.0
   SUM(J)=0.0
10 CONTINUE
   ASSIGN 21 TO LOOP
   ASSIGN 28 TO NEW
   ASSGN 36 TO JUMP
   NA=NEND-NBEGIN+1
   K=NBEGIN-1
   DO 50 LA=1,LG
      IF (LG-LA) 11,11,13
11 M1=MLAST
12 DO 14 K=K1,M1
   READ INPUT TAPE 7,4,(R(I,K),I=1,N)
13 CONTINUE
   K1=1
   DO 20 I=1,NEND
      DO 20 K=1,M1
      SUM(I)=SUM(I)+R(I,K)
      SIGMA(I)=SIGMA(I)+(R(I,K)**2)
20 CONTINUE
   GO TO LOOP,(21,22)
21 ASSIGN 22 TO LOOP
   ITPAE=3
   JTAPE=4
   GO TO 23
22 ASSIGN 21 TO LOOP
   JTAPE=3
   ITPAE=4
23 DO 44 LE=1,LY
   NFIRST=NX(LE)
   NLAST=NY(LE)
DO 44 I=NFIRST,NLAST
M=0
GO TO NEW,(28,32)
28 DO 30 J=1,NA
PROD(J)=0.0
30 CONTINUE
32 GO TO JUMP,(34,36)
34 READ TAPE ITAPE,(PROD(J),J=1,NA)
36 DO 42 L=NBEGIN,NEND
M=M+1
DO 42 K=1,M
40 PROD(M)=PROD(M)+R(I,K)*R(L,K)
42 CONTINUE
WRITE TAPE JTAPE,(PROD(J),J=1,NA)
44 CONTINUE
REWRIND ITAPE
END FILE ITAPE
REWRIND JTAPE
49 ASSIGN 32 TO NEW
ASSIGN 34 TO JUMP
50 CONTINUE
READ INPUT TAPE 7,52,NEW
52 FORMAT(I10)
WRITE OUTPUT TAPE 6,53,NEW
53 FORMAT(8H1,15X,85H CORRELATION MATRIX JOB NUMBER 110 )
DO 54 M=1,NA
54 JA(M,1)=K+M
CA=COUNT*(COUNT-1.0)
DF=COUNT-2.0
SDF=SQR(T(DF)
DO 55 I=NFIRST,NEND
CB=(COUNT*SIGMA(I)-(SUM(I)**2)
SIGMA(I)=SQR(T(CB/CA)
55 CONTINUE
DO 90 LE=1,LY
NFIRST=NX(LE)
LAST=NY(LE)
DO 90 I=NFIRST,NLAST
M=0
READ TAPE ITAPE,(PROD(J),J=1,NA)
DO 80 L=NBEGIN,NEND
M=M+1
TOP=(COUNT*PROD(M)-(SUM(I)*SUM(L))
BELOW=CA*(SIGMA(I)*SIGMA(L))
IF (BELOW) 60,60,65
60 PROD(M)=9.999
TEE(M)=9.999
GO TO 80
65 PROD(M)=TOP/BELOW
CC=PROD(M)**2
IF (1.0-CC) 71,71,72
71 TEE(M)=8.888
GO TO 80
72 TEE(M)=(PROD(M)/(SQR(T(1.0-CC))))*SDF
80 CONTINUE
WRITE OUTPUT TAPE 6,8,1,DF*,(*(JA(M,1),PROD(M),TEE(M)),M=1,NA)
8 FORMAT(14H017,F8.0/(4(1H I8,F10.3,F10.3)))
IF (CARD) 83,90,81

110
PUNCH 82*(PROD(M)*M=1*NA)
FORMAT(12F6.3)
IF (TVALUE) 90,90,84
L=0
DO 87 M=1,NA
CC=ABSF(TEE(M))
IF (CC=TVALUE) 87,86,86
L=L+1
TEE(L)=TEE(M)
PROD(L)=PROD(M)
JA(L,2)=JA(M,1)
87 CONTINUE
IF (L) 90,90,88
WRITE OUTPUT TAPE 6,8,1,DF,((JA(M,2),PROD(M),TEE(M)),M=1,L)
90 CONTINUE
REWIND JTAPE
GO TO 1
*ASSEMBLE,PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END 0,1
PZE
BCD 1INPUT
PZE ENT
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7,4
STA CPY
SXD AX1+1
LXA A=1+1
CPY CPY **,1
D=12 TXI *+3+1,1
TSX ERROR*4
TSX ERROR*4
TXL CPY*1,12
LXD AX1*1
TRA 1*4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
3. R-COMPLETION PROGRAM

A. DESCRIPTION OF THE PROGRAM

Program print out (see Appendix C)

B. SPECIAL FEATURES

1. The program has been designed to
   a. Accept as input data the upper half of the off-diagonal elements
      of a correlation matrix (see inter-correlation program).
   b. Eliminate variables not desired for further analysis and
      rearrange the rows and columns of the correlation matrix so
      that the resulting matrix will be symmetric ($r_{ij} = r_{ji}; i \neq j$).
      The original order of variables will not be disturbed.
   c. Insert either 1's or row maxima into the diagonal of the
      correlation matrix, be it the original or the one described
      in b.

2. The matrix described in b (or the original unchanged matrix) can be
   used as input data for the matrix inversion and estimation of
   communalities program presented in section VII of this publication.

3. The matrix described in c can be used as input data for the factor
   analysis program presented in section VIII of this publication if no
   other estimate of communalities is desired.

C. LIMITATIONS

This program can be used for a maximum of about 2500 variables (8k memory),
although in the present form (see print-out) it will accept a maximum of 998 variables. If larger matrices are to be processed, format statement no. 15 has to be changed to read: FORMAT(18I4/). Corresponding changes in the control cards have to be made (see Appendix G, control cards 2, 3, 4, etc.).

D. RUNNING TIME ESTIMATES

The compilation time is approximately five minutes. Computing time for very large matrices should not exceed 5 minutes. A matrix of order 25 requires approximately 6 seconds.

E. TEST PROBLEMS

The test problems have been designed so that they cover possible occurrences.

F. OUTPUT

1. Print-out: Complete correlation matrix, including diagonal elements called for, with row identification. Omitted variables are not included.

2. Card output:
   a. For inversion program: upper half off-diagonal correlation matrix (12F6.3).
   b. For factor analysis: full matrix with diagonal elements called for (12F6.3).
Appendix G

1. ARRANGEMENT OF CARDS (See Appendix E)
   a. Using FORTRAN deck. See Fig. 4.
   b. Using binary deck. See Fig. 4.

2. EXAMPLE PROBLEMS

<table>
<thead>
<tr>
<th>EXAMPLE</th>
<th>NUMBER OF VARIABLES</th>
<th>DIAGONAL ELEMENT</th>
<th>INVERS</th>
<th>CARDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>280</td>
<td>1.0</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>B</td>
<td>180</td>
<td>Max</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>C</td>
<td>146</td>
<td>1.0</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>D</td>
<td>96</td>
<td>1.0</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>E</td>
<td>31</td>
<td>Max</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>

Note: 1. The completed correlation matrix is printed out with row identification.

2. INVERS-yes calls for card output for the inversion program.

   CARDS-yes calls for card output for the factor analysis program.

3. The card output for the inversion program contains the upper half off-diagonal elements only. The print-out, however, gives the complete matrix including the diagonal
elements called for. It is suggested that this option of the program be used only in the case that the input matrix is of order $> 144$, and the final matrix size is $\leq 72$ (using an $8k$ memory machine; for a $32k$ memory the corresponding numbers are $>320$ and $\leq 160$). Otherwise, when these limits are:

8k memory, $\leq 144$ and $\leq 72$

32k memory, $\leq 320$ and $\leq 160$

an option for completing the correlation matrix is provided in the inversion program which accepts matrices within these limits.

4. The card output for the factor analysis program contains the full matrix with diagonal elements called for.

5. DIMENSION CARDS FOR THE FORTRAN DECK

Under inter-correlation programs the dimension cards are discussed in great detail. The user is advised to follow through the steps. The procedure to be followed is approximately the same.

4. MFIRST = *** or ** CARD

For discussion on MFIRST = *** or ** card, see inter-correlation programs.

(Number observations corresponds here to number of variables)

5. CONTROL CARDS

a. Control card no. 1
<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-6</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td>A</td>
<td>+00280</td>
</tr>
<tr>
<td>B</td>
<td>+00180</td>
</tr>
<tr>
<td>C</td>
<td>+00072</td>
</tr>
<tr>
<td>D</td>
<td>+00094</td>
</tr>
<tr>
<td>E</td>
<td>+00027</td>
</tr>
</tbody>
</table>

Note: 1. N refers to the desired size of the new matrix. In example (C) 74 variables, in example (D) 2 variables, and in example (E) 4 variables are to be omitted.

2. NA refers to the input matrix.

3. See Fig. 7.

b. Control card no. 2, 3, 4, ... etc.

For every variable to be omitted 999 is punched in the location of the control card as indicated below. This will make the program skip that variable. Three columns are reserved for each variable. Thus on one card 24 variables can be controlled. Cards are punched from column 1 through 72. If more than 998 variables are processed (see Limitations), 18 variables per card can be controlled, and the skip-punch will be 9999.

EXAMPLE A

1. See Fig. 7

2. Since all variables are required in the completed correlation
matrix, the control cards are made out as follows.

On each card 24 variables can be controlled. Thus in this problem \((280/24) = 12\) cards are required. All cards will be blank. Thus 12 blank cards are inserted after control card no. 1.

EXAMPLE B

1. See Fig. 7.

2. All variables are required in the completed correlation matrix. A total of \((180/24) = 8\) blank cards serve as control cards.

EXAMPLE C

In this problem 74 variables are to be skipped. The procedure is the same as in examples D and E. A total of \((146/24) = 7\) cards are required.

EXAMPLE D

1. See Fig. 7.

2. In this problem variables 18 and 93 are to be omitted. The size of the final output matrix is therefore \((96-2) = 94\).

3. The control cards are made out as follows. A total of \((96/24) = 4\) cards are required.
In each card 72 columns are used. Variable no. 93, in the 3-digit format, ends in column $3 \times 93 = 279$. This number divided by 72 gives 3 cards + 63 columns. Obviously, 999 for variable no. 93 has to be punched in columns 61-63 of control card no. 5 (which is the fourth card of this sequence). Control card no. 1 is discussed above.

EXAMPLE E

1. See Fig. 7.

2. In this problem variables 7, 13, 24, 31 are to be omitted.

3. A total of $(31/24) = 2$ cards are required.

<table>
<thead>
<tr>
<th>CONTROL CARD NO.</th>
<th>VARIABLE NO.</th>
<th>IBM CARD COLUMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1 thru 24</td>
<td>Blank 999</td>
</tr>
<tr>
<td>3</td>
<td>25 thru 31</td>
<td>Blank 999</td>
</tr>
</tbody>
</table>

6. FORMAT STATEMENT

Input format is 12F6.3. No format card is needed. For format change see 7b. below.
7. DATA CARDS

a. See Appendix A, means and standard-deviations program.

b. The upper half off-diagonal elements of the correlation matrix is the input. The cards can be obtained as output from the correlation programs, the format being F 6.3 including decimal point (e.g. + 0.563). If the input correlation matrix is punched separately and a format different from 12F6.3 is used, the program has to be recompiled with corresponding change in Format Statement no. 7 (see printout). F6.3 is preferred by the authors. The decimal point may be left out (e.g. + 00563) or punched (e.g. + 0.563).

For example, a matrix of 34 x 34 should be punched the following way:

<table>
<thead>
<tr>
<th>CARD NO.</th>
<th>IBM CARD COLUMNS</th>
<th>Row no. Card no.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-6</td>
<td>7-12</td>
<td>55-60</td>
</tr>
<tr>
<td>7-72</td>
<td>73-80</td>
<td></td>
</tr>
</tbody>
</table>

1. \( r_{1-2} \) \( r_{1-3} \) \( \ldots \) \( r_{1-13} \) 00010001
2. \( r_{1-14} \) \( r_{1-15} \) \( \ldots \) \( r_{1-25} \) 00010002
3. \( r_{1-26} \) \( r_{1-27} \) \( \ldots \) Blank 00010003
4. \( r_{2-3} \) \( r_{2-4} \) \( \ldots \) \( r_{2-14} \) 00020001
5. \( r_{2-15} \) \( r_{2-16} \) \( \ldots \) \( r_{2-26} \) 00020002
\( \ldots \) \( \ldots \) \( \ldots \) \( \ldots \)
63. \( r_{33-34} \) Blank 00330001

\( *r_{1-2} \) means the correlation for variable 1 with 2.

8. JOB NUMBER CARD

See Appendix A, means and standard-deviations program.
EXAMPLE C FOLLOW THE PROCEDURE OF D AND E.

D

1, 2 1, 3 .................. 1, 96
2, 3 .................. 2, 96
18, 19 .................. 18, 96
93, 94 .................. 93, 96
95, 96

E

1, 2 1, 3 .................. 1, 31
2, 3 .................. 2, 31
7, 8 .................. 7, 31
13, 14 .................. 13, 31
24, 25 .................. 24, 31
30, 31

THE ABOVE MATRICES INDICATE THE ARRANGEMENT OF INPUT CARDS.
THE UPPER HALF OFF-DIAGONAL ELEMENTS ARE USED.
*COMPILE FORTRAN PRINT SAP, PUNCH OBJECT, EXECUTE, DUMP
DIMENSION R(200, 25), C(200), JA(200)
1 READ INPUT TAPE 7*2*N, NA, DIAG, INVERS, CARDS
2 FORMAT(216*F2.0,12*F2.0)
   READ INPUT TAPE 7*3*(JA(I), I=1, NA)
3 FORMAT(2413)
   REWIND 2
   REWIND 3
   REWIND 4
   MFIRST=25
   LG=N-NA
   L=1
   M=1
   M1=NA-1
   ASSIGN 140 TO JUMP
   IF (LG) 6*16*16
6 DO 15 K=1*M1
   J=M
   L=L+1
   LG=0
   ASSIGN 8 TO NEW
   DO 10 I=L, NA
      GO TO JUMP,(140, 145)
140 ASSIGN 145 TO JUMP
   CALL INPUT
   145 READ INPUT TAPE 7*4*(C(I), I=L, NA)
4 FORMAT(1)
   IF (I=JA(I)) 10*7*7
    7 GO TO NEW,(8*9)
   8 LG=LG+1
      M=M+1
   ASSIGN 9 TO NEW
   9 J=J+1
   C(J)=C(I)
10 CONTINUE
   IF (LG) 15*15*12
12 IF (INVERS) 14*14*13
13 PUNCH 120*(C(I), I=M, N)
14 WRITE TAPE 2*(C(I), I=M, N)
15 CONTINUE
   GO TO 18
16 ASSIGN 150 TO JUMP
   DO 17 K=1*M1
      L=L+1
   GO TO JUMP,(150, 155)
150 ASSIGN 155 TO JUMP
   CALL INPUT
155 READ INPUT TAPE 7*4*(C(I), I=L, NA)
   WRITE TAPE 2*(C(I), I=L, NA)
17 CONTINUE
18 READ INPUT TAPE 7, 19, NEW
19 FORMAT(110)
   WRITE OUTPUT TAPE 6*20, NEW
20 FORMAT(40H1, COMPLETED CORRELATION MATRIX 110 )
   END FILE 2
   REWIND 2
   ASSIGN 32 TO NEW
   LG=N/MFIRST

121
MLAST=MFIRST*LG
IF (N=MLAST) 23*,22*,23
22 MLAST=MFIRST
GO TO 24
23 LG=LG+1
MLAST=N-MLAST
24 L=1
IA=1
KA=0
IB=0
M1=0
L1=1
DO 36 LA=1*LG
IF (LG-LA) 26*,26*,27
26 M1=MLAST-1
ASSIGN 31 TO NEW
GO TO 28
27 M1=MFIRST
28 DO 30 K=1,M1
L=L+1
READ TAPE 2,(R(I,K),I=L,N)
30 CONTINUE
GO TO NEW,(31*,32*)
31 M1=M1+1
32 IB=IB+M1
I1=0
DO 35 I=IA,IB
I1=I1+1
K1=KA
DO 35 K=1,M1
K1=K1+1
IF (I1-K) 34*,33*,35
33 R(I,K)=1,
GO TO 35
34 R(I,K)=R(K1,I1)
35 CONTINUE
WRITE TAPE 3,(R(I,K),I=IA,N),K=1,M1
IA=IA+M1
KA=KA+M1
36 CONTINUE
END FILE 3
REWIND 3
REWIND 2
I2=0
DO 80 LA=1*LG
I1=0
IC=1
DO 55 LB=1*LA
IF (LG-LB) 40*,40*,42
40 M1=MLAST
GO TO 43
42 M1=MFIRST
43 READ TAPE 3,(R(I,K),I=IC,N),K=1,M1
IC=IC+M1
IF (LB-LA) 45*,55*,55
45 IF (LG-LA) 46*,46*,47
46 M2=MLAST
GO TO 48
M2=MFIRST
DO 50 K=1*M1
I1=I1+1
IB=IB+1
DO 50 I=1*M2
R(I1+I)=R(IB+K)
50 CONTINUE
55 CONTINUE
WRITE TAPE 4*((R(I*K),I=1*N)*K=1*M1)
REWIND 3
I2=12+M1
80 CONTINUE
END FILE 4
REWIND 4
KA=0
DO 135 LA=1*LG
86=86*87
IF (LG-LA) 86*86*87
GO TO 88
M1=MLAST
87 M1=MFIRST
90 READ TAPE 4*((R(I*K),I=1*N)*K=1*M1)
DO 130 K=1*M1
95 CMAX=CMIN
100 CONTINUE
R(KA*K)=CMAX
113 IF (CARDS) 122*122*115
115 PUNCH 120*(R(I*K),I=1*N)
120 FORMAT(12F6.3)
122 WRITE OUTPUT TAPE 6*(R(I*K),I=1*N)
125 FORMAT(1H0*14/(1H 13F9.3))
130 CONTINUE
135 CONTINUE
REWIND 4
GO TO 1
*ASSEMBLE*PUNCH OBJECT
REM INPUT
ORG 0
PGM
PZE END"0"1
PZE
BCD 1INPUT
PZE ENT
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7*4
STA CPY
SXD AX1*1
123
LXA A=1
CPY CPY **1
D=12 TXI ++311
TSX ERROR'4
TSX ERROR'4
TXL CPY'1112
LXD AX11
TRA 1'4
A=1 PZE 1
AX1 BSS 1
END SYN *
END
*DATA
VII. INVERSION OF SYMMETRIC MATRICES AND ESTIMATION OF COMMUNALITIES

A. DESCRIPTION OF THE PROGRAM

1. Program print out (See Appendix II).

2. This program is especially designed for estimating diagonal elements. Depending on the need of the research worker, the program can be instructed to go through any one of the steps discussed under B. In addition, the program can go through the steps discussed in the R-completion program, provided that the resulting smaller matrix is of order $\leq 72$, and the original matrix $\leq 144$ (8k memory), or $\leq 160$ and $\leq 320$ (32k memory).

3. Special codes built into the program

To avoid stoppage due to any unforeseen circumstances, the following feature is incorporated:

Whenever the diagonal pivot element is exactly zero, i.e. the input matrix is singular, the program prints out the comment 'the diagonal element is zero'. In this case some of the variables used may not be independent of each other.

B. DISCUSSION

1. Input upper-half off-diagonal elements of the correlation matrix.

2. Select as a first estimate 1.0 for the diagonal elements.
3. Single precision inversion of the symmetric matrix (12):

Repeat steps (a) through (c) N times for \( n = 1, 2, 3 \ldots N \), where \( N \) is the order of the matrix.

a. Compute for temporary use the quantities

\[ p_i = a_{in} \times a_{nn}^{-1} \quad \text{for } i < n \quad (i = 1, 2 \ldots N) \]

\[ p_i = a_{ni} \times a_{nn}^{-1} \quad \text{for } i > n \]

b. Replace each element \( a_{ij} \) by the element \( a_{ij}^* \) where:

\[ a_{ij}^* = a_{ij} - p_i \times a_{jn} \quad \text{for } j < n \]

\[ a_{ij}^* = a_{ij} - p_i \times a_{nj} \quad \text{for } j > n \]

c. Replace

1. \( a_{in} \) by \( a_{in}^* = p_i \) for \( i < n \)

2. \( a_{ni} \) by \( a_{ni}^* = p_i \) for \( i > n \)

3. \( a_{nn} \) by \( a_{nn}^* = -a_{nn}^{-1} \)

d. Change signs of all elements and the resultant matrix is the inverted matrix. Print out the inverted matrix.

e. Compute \( A \times A^{-1} \) and print out the result.

\[ A \] is the original correlation matrix and

\[ A^{-1} \] is the inverted matrix.

4. Compute \( E_{jj} = 1/a_{jj}^{jj} \) where \( a_{jj}^{jj} \) refers to the diagonal element of the inverted matrix, and \( E \) is a diagonal matrix \((N \times N)\).

5. Estimation of diagonal elements

There are two options available. See note for a suggestion.
a. Insert squared multiple correlations ($R^2$) in the diagonal ($2; 15$)

1. Compute $r_{jj} = 1 - E_{jj}$

2. Insert $r_{jj}$ in the diagonal of the input correlation matrix.
   Punch output and print out.

b. Estimate diagonal elements and adjust off-diagonal elements of the correlation matrix (Image or 'P' matrix) ($3; 9$).

1. Replace all elements of the inverted correlation matrix by
   
   \[ A^*_jk = (E_{jj} \times A^{-1}_{jk} \times E_{kk}) + A_{jk} \]
   
   where:
   \[ A_{jk} \text{ refers to the input correlation matrix with l's in the diagonal.} \]

2. Replace all diagonal elements of the resultant matrix by
   
   \[ A^{**}_{jj} = A^*_{jj} - 2E_{jj} \]

3. Print out and punch adjusted matrix.

Note: When inserting squared multiple correlations into the diagonal as estimates for communalities the off-diagonal elements may not conform to the concept of a Gramian matrix. The use of the Kaiser-Guttman ($3; 9$) 'P' or Image matrix described in this program is a good solution to the problem. This procedure will adjust the off-diagonal elements in relation to the diagonal.

C. LIMITATIONS

The dimension cards for the program have been set for an 8k memory (see print-out in Appendix H). The inversion section of the program can handle
a matrix of order 72. However, using a 32k memory the order of the matrix can be about 160. The limitations of the R-completion section of the program are given above (see description of program). All computations are single precision.

D. RUNNING TIME ESTIMATES

The compilation is approximately five minutes. Some running time estimates, based on actual runs, are listed below:

<table>
<thead>
<tr>
<th>Final Matrix Size</th>
<th>Option</th>
<th>Running Time minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$18 \times 18$</td>
<td>$\mathbb{R}^2$</td>
<td>.5</td>
</tr>
<tr>
<td>$20 \times 20$</td>
<td>'P'-(Image-matrix)</td>
<td>1.7</td>
</tr>
<tr>
<td>$20 \times 50$</td>
<td>'P'-(Image-matrix)</td>
<td>4.1</td>
</tr>
</tbody>
</table>

The R-completion part of the program requires less than 20 seconds for maximum matrix size.

E. TEST PROBLEMS

See examples in Appendix H.

F. OUTPUT (see example problems in Appendix H)

1. Print out of the complete resultant matrices (with row identification):
   a. $R^{-1}_1, R^{-1}_1 R^{-1}_1, (R_1 + \text{diag } R^2_1)$
   b. $R^{-1}_1, R^{-1}_1 R^{-1}_1, P$
   c. $R_1$ with unities or row maxima in the diagonal
2. Card output, without row identification, can be obtained (optional)

for \((R_1 + \text{diag } R_1^2)\), \(P\), \(R_1\), or \(R_1^{-1}\) above.
Appendix H

1. ARRANGEMENT OF CARDS

(See Appendix E, inter-correlation program)

a. Using FORTRAN deck (see Fig. 4)

b. Using binary deck (see Fig. 4)

2. EXAMPLE PROBLEMS

<table>
<thead>
<tr>
<th>Example</th>
<th>$R_1$</th>
<th>$R_0$</th>
<th>DIAG. ELEM. FIRST</th>
<th>MATRIX INVERSION</th>
<th>DIAG. ELEM. FINAL</th>
<th>CARD OUT-PUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>70</td>
<td>110</td>
<td>1.0</td>
<td>no</td>
<td>1.0</td>
<td>yes</td>
</tr>
<tr>
<td>B</td>
<td>60</td>
<td>60</td>
<td>1.0</td>
<td>yes</td>
<td>$R^2$</td>
<td>yes</td>
</tr>
<tr>
<td>C</td>
<td>50</td>
<td>58</td>
<td>1.0</td>
<td>yes</td>
<td>$R^{-1}$</td>
<td>no</td>
</tr>
<tr>
<td>D</td>
<td>40</td>
<td>50</td>
<td>1.0</td>
<td>yes</td>
<td>$P$</td>
<td>yes</td>
</tr>
<tr>
<td>E</td>
<td>35</td>
<td>40</td>
<td>max</td>
<td>no</td>
<td>max</td>
<td>yes</td>
</tr>
<tr>
<td>F</td>
<td>50</td>
<td>30</td>
<td>max</td>
<td>no</td>
<td>max</td>
<td>no</td>
</tr>
</tbody>
</table>

Note:  
$R_1$ = Size of matrix after elimination of variables 
$R_0$ = Size of original input matrix 
$R^2$ = Squared multiple correlations (see discussion, 5a) 
$P$ = Image matrix (see discussion, 5b) 
$R^{-1}$ = Inverted matrix (no calculations beyond inversion) 
Max = Rox maxima
3. DIMENSION CARDS FOR THE FORTRAN DECK

The dimension cards have been set for an 8k memory. See inter-correlation programs for discussion on dimension cards. See also the program print-out in this appendix.

4. CONTROL CARDS

a. Control card no. 1

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-3</td>
</tr>
<tr>
<td>A</td>
<td>070</td>
</tr>
<tr>
<td>B</td>
<td>060</td>
</tr>
<tr>
<td>C</td>
<td>050</td>
</tr>
<tr>
<td>D</td>
<td>040</td>
</tr>
<tr>
<td>E</td>
<td>035</td>
</tr>
<tr>
<td>F</td>
<td>030</td>
</tr>
</tbody>
</table>

Note: $N = R_1$ in example problems.
NA = $R_0$ in example problems.

DIAG: +1 inserts l's in the diagonal of the input matrix.
-1 inserts row maxima (absolute values) in the diagonal of the input matrix. In this case the program cannot go through the matrix inversion loop. The choice is either to punch or not to punch output cards.
SMCINV: +1 calls for matrix inversion.
-1 calls for skipping the inversion loop, and the choice is either to punch or not to punch out-put cards.
+0 = no further calculations available. +0 should be punched when using row maxima.

SMC: +1 calls for calculation of $R^2$'s, which automatically will be inserted in the diagonal. The off-diagonal elements will not change.
-1 calls for computation of the Image matrix ($P$).
+0 = no further calculations required (or available; compare SMCINV).

CARDS: +1 calls for out-put cards.
-1 no out-put cards.

b. Control card no. 2, 3, 4, etc.

Same as in the R-completion program (see Appendix G).

3. FORMAT CARD

No FORMAT card is needed.

This program will accept only one format: 12F6.3, which also is the format of the out-put cards of the correlation programs and the R-completion program. If data cards are punched separately for this program, the procedure
outlined in Appendix G under data cards should be followed.

If a different input format is desired, a new program should be compiled, where the proper program card (statement no. 4 in the program print-out) has been changed.

6. JOB NUMBER CARD

See Appendix A, means and standard deviations program.
*COMPILE FORTRAN PRINT SAP PUNCH OBJECT EXECUTE DUMP

DIMENSION A(72,72),E(72), R(72), JA(144), F(144)
EQUIVALENCE (E(1),F(1)), (R(1),F(73))

READ INPUT TAPE 7, N, NA, DIAG, SMCINV, SMC, CARDS

FORMAT (2I3,4F2.0)
RENUMD 2
RENUMD 3
L=1
M=NA-1
ASSIGN 39 TO NEW
ASSIGN 86 TO JUMP
READ INPUT TAPE 7, 3, (JA(I), I = 1, NA)

FORMAT (24I3)
I=0
DO 18 IB=1,M
  J=I+1
  L=L+1
  IA=L
  READ INPUT TAPE 7, 4, (F(K), K = L, NA)
  FORMAT (12F6.3)
  IF (IB-JA(IB)) 18, 10, 10
10  I=I+1
  DO 15 K=IA, NA
    IF (K-JA(K)) 15, 12, 12
12  J = J + 1
    A(I,J) = F(K)
15  CONTINUE
18  CONTINUE
  DO 20 I=1,N
    A(I,I)=0.0
    DO 20 K=1,N
      IF (I=K) 19, 20, 20
19    A(K,I)=A(I,K)
20    CONTINUE
    IF (DIAG) 24, 21, 21
21  DO 23 I=1,N
      A(I,I)=1.0
23  CONTINUE
    IF (SMCINV) 100, 36, 36
100  ASSIGN 190 TO JUMP
    GO TO 105
24  DO 35 I=1,N
      CMAX=0.0
      DO 30 K=1,N
        CMIN=ABSF(A(I,K))
        IF (CMAX-CMIN) 25, 30, 30
30      CMAX=CMIN
35    CONTINUE
    A(I,I)=CMAX
35    CONTINUE
    GO TO 100
36  DO 38 I=1,N
38    WRITE TAPE 2, (A(I,K), K=1,N)
    END FILE 2
    REWIND 2
    DO 81 I=1,N
      GO TO NEW*(39, 81)
39    IF (A(I,I)) 42, 40, 42
40    ASSIGN 81 TO NEW
ASSIGN 84 TO JUMP
GO TO 81
42 DO 48 K=1,N
   IF (I-K) 44,48,46
44 R(K)=A(I,K)/A(I,I)
   GO TO 48
46 R(K)=A(K,I)/A(I,I)
48 CONTINUE
   DO 65 J=1,N
      IF (J-I) 50,65,50
50 DO 60 K=1,N
      IF (K-I) 56,60,52
52 IF (J-K) 54,56,64
54 A(J,K)=A(J,K)-R(J)*A(I,K)
   GO TO 60
56 IF (J-K) 58,58,60
58 A(J,K)=A(J,K)-R(J)*A(K,I)
60 CONTINUE
65 CONTINUE
   DO 80 K=1,N
      IF (K-I) 67,69,75
67 A(K,I)=R(K)
   GO TO 80
69 A(I,I)=1./A(I,I)
   GO TO 80
75 A(I,K)=R(K)
80 CONTINUE
81 CONTINUE
905 READ INPUT TAPE 7,82,1
92 FORMAT(110)
   WRITE OUTPUT TAPE 6,83,1
93 FORMAT(44H1 INVERTED MATRIX JOS NUMBER I10 )
   GO TO JUMP (84,86,190)
94 WRITE OUTPUT TAPE 6,85
95 FORMAT(40H1 DIAGONAL ELEMENT IS ZERO )
   GO TO 1
96 DO 95 I=1,N
   DO 90 K=1,N
      IF (I-K) 87,88,90
87 A(I,K)=-A(I,K)
   A(K,I)=A(I,K)
88 A(I,K)=-A(I,K)
90 CONTINUE
   WRITE OUTPUT TAPE 6,94,1*(A(I,K)*K=1,N)
94 FORMAT(1H018/*(1H 10F11.3))
95 CONTINUE
   DO 145 I=1,N
      READ TAPE 2*(R(K)*K=1,N)
   DO 130 L=1,N
      E(L)=0.0
   DO 130 J=1,N
      E(L)=E(L)+(R(J)*A(J,L))
130 CONTINUE
   WRITE OUTPUT TAPE 6,135,1*(E(K)*K=1,N)
135 FORMAT(1H018/*(1H 16F7.3))
145 CONTINUE
   REWIND 2
DO 150 I=1*N
150 E(I)=1./A(I,I)
   IF (SMC) 155,190,170
155 DO 165 I=1,N
   READ TAPE 2*(R(K)*K=1,N)
   DO 160 K=1,N
160 A(I,K)=(E(I)*A(I,K)*E(K))+R(K)
   A(I,I)=A(I,I)-(2.*E(I))
165 CONTINUE
   GO TO 190
170 DO 180 I=1,N
   READ TAPE 2*(A(I,K)*K=1,N)
180 A(I,I)=1.-E(I)
190 DO 250 I=1,N
   IF (CARDS) 220,225,195
195 PUNCH 200*(A(I,K)*K=1,N)
200 FORMAT(12F6.3)
220 WRITE OUTPUT TAPE 6*94*(A(I,K)*K=1,N)
250 CONTINUE
   GO TO 1

*DATA
VIII. FACTOR ANALYSIS PROGRAM

(Principal Components)

A. DESCRIPTION OF THE PROGRAM

Program print-out (see Appendix I)

B. SPECIAL FEATURES

1. The input sub-routine reads in the FORMAT of the data cards. The program, therefore, is capable of handling data with any format. The R-completion and the inversion programs are designed to punch output cards, which can be used directly as input for the factor analysis program. The format of these cards is 12F6.3. If cards are to be specially key-punched for this program, the complete matrix, including diagonal elements, has to be punched.

2. When to stop factoring

Any of the options given below will stop factoring, depending on which condition is first met. It is possible to stop factoring on any desired option by preparing the control card so that the occurrence of the other options before the desired one becomes unlikely. Regardless of these options the computer will automatically stop factor extraction if a negative latent root is obtained during computation. In this case it is evident that the requirement of positive semi-definiteness of the reduced correlation matrix has
been violated (5, pp. 28-29 and 187). The program will print out all the roots and the corresponding factor loadings computed up to this point. A card output will also be obtained automatically. The latent roots for real symmetric positive semi-definite matrices are always real and non-negative. Any estimates of communalities may lead to negative latent roots. The corresponding factors will be imaginary and cannot be used. Unities in the diagonal will preserve the Gramian properties of the matrix. This should also be the case when using the P (Image) matrix, according to Kaiser (11).

**Option 1 - Maximum number of factors**
The estimated upper limit of the number of factors to be extracted is read into the computer (see control card). A good estimate would be from one-sixth (1/6) to one-third (1/3) of the number of variables involved (5, p. 563; 11). Extraction of factors will stop at this limit, only, if none of the other conditions has been satisfied.

**Option 2 - Value of last latent root**
This option deals with the question of lower bound for the number of common factors (4; 9, p. 563; 8). If squared multiple correlations (2; 15) are used in the diagonal of the correlation matrix, the lower bound would, according to Guttman (4), be equal to the number of positive latent roots. An alternative lower bound, though weaker, also given by Guttman, would be the number of latent roots greater than one of the observed correlation matrix. Kaiser (8; 11) has studied
this question very thoroughly, and his conclusion is: "A best answer
to the question of the number of factors is the number of latent roots
greater than one of the observed correlation matrix". This lower
bound will, in general, correspond to one-sixth \(\frac{1}{6}\) to one-third
\(\frac{1}{3}\) of the number of variables.
The desired value of the last latent root to be computed (e.g. 1.0,
or 0.5) has to be entered in the control card. The computer will stop
factoring when a latent root equal to or smaller than this value has
been obtained, provided that none of the other conditions has been
satisfied.

Option 3 - Ratio first/last latent root

An estimated maximum ratio, for example 20, should be entered in the
control card. This option will stop factoring, if none of the other
conditions has been met. This option has been built into the program
to prevent the extraction of an excessive number of factors with
comparatively small loadings. If this option is not desired, the
ratio can be set high enough, say 50, to make the occurrence of this
condition most unlikely.

Option 4 - Percent variance extracted (5, pp. 160 and 363)
a. The percent variance extracted may be calculated either from
total number of variables \(\sigma^2\) per variable = 1.0) or from total
starting communality \(\Omega_{est}^2\). The choice is entered in the control
card. If unities are used in the principal diagonal, either choice
will give the same answer.
b. A decision concerning the percent of total variance to be extracted has to be made. For instance, if 85% of the total variance is considered sufficient, the number 85 has to be entered into the control card. The extraction of factors will stop when this percentage has been reached, providing none of the other conditions, given above, has been satisfied.

Option 5 - Degree of accuracy (Epsilon)

(See Discussion)

The desired accuracy (Epsilon) of the eigen-vector has to be entered in the control card. Iteration carried to third or fourth decimal place accuracy is in general considered sufficient. The program, however, will accept any accuracy required.

The above options have been provided to eliminate unnecessary computer time which otherwise would be used for computation of factors with no practical significance.

C. DISCUSSION

The iterative method is based upon two papers by Hotelling (6; 7). The steps followed by this program (5; 14) are indicated below:

1. Compute

\[ d_{j1} = \frac{\sum_{k=1}^{n} r_{jk} a_{kl}}{n} \]  

\[(j = 1,2,\ldots n)\]

where

the first estimate for \( a_{kl} = (1,1,\ldots,1) \) and \( n \) is the order of the matrix \( R \).
2. Find

\[ E = \text{Max} (d_{jl}) \quad (j = 1, 2, \ldots n) \]

3. Compute

\[ a_{kl}^* = \frac{1}{E} d_{kl} \quad (k = 1, 2, \ldots n) \]

4. Compare

\[ \text{Abs.} \left| (a_{kl}) - (a_{kl}^*) \right| \leq \text{Epsilon} \quad (k = 1, 2, \ldots n) \]

If this has been achieved, proceed to step 6. If not, go to step 5.

5. Replace

\[ a_{kl} \text{ by } a_{kl}^* \]

and repeat steps 1 through 4.

Note: Epsilon is a predetermined level of decimal accuracy. E is the largest positive latent root of R.

6. Compute factor loadings

\[ C_k = E \times a_{kl}/\sqrt{(\sum_{j=1}^{n} a_{jl}^2)} \quad (k = 1, 2, \ldots n) \]

7. Compute the first residual matrix

\[ R_1 = R - Q_1 \]

where

\[ Q_1 = C_1C_1' \]

is the n x n symmetric matrix of products of first-factor coefficients appearing in the column vector C_1.

8. The 2nd, 3rd, ..., mth positive latent roots (in descending order of magnitude), corresponding eigen-vectors and factor loadings are computed using the 1st, 2nd, ..., (m-1)th residual matrices in place of R, repeating steps 1 through 6.
9. Compute new communalities

$$h_k^2 = \sum_{j=1}^{m} \frac{c_{kj}^2}{c_{kj}^2} \quad (k = 1, 2, \ldots, n)$$

10. Compute sum of latent roots and sum of communalities

$$\frac{\sum E_j}{m} \quad \text{and} \quad \frac{\sum h_k^2}{n}$$

D. LIMITATIONS

Maximum matrix size, 8k memory, 1100 x 1100

Maximum matrix size, 32k memory, 5000 x 5000

E. RUNNING TIME ESTIMATES

Compilation time is approximately five minutes.

There are no time estimates available for large matrices. A matrix 36 x 36 which gave five factors (Epsilon = .0001, and stopped at lower bound: latent root $\leq .5$) required 5.5 minutes computing time. However, this job was run on a compilation of maximum 180 x 180 variables, which caused the computer to calculate first 24 and then 12 rows (see program print-out, dimension card and MFIRST = 24 card). On a maximum 68 x 68 compilation (the whole matrix fits into the core) the time required for above job is approximately 1.8 minutes.

Computing time is bound to vary, even for matrices of equal size, depending on how many iterations are required for convergence, accuracy (Epsilon), and number of factors extracted. Convergence is usually fast for the first
few factors, and tends to slow down as extraction goes on. In the case that two or more consecutive latent roots are near equal in size, convergence will be very slow.

In general, the program is very fast for matrices of order $\leq 68$ (8k memory), and $\leq 165$ (32k memory). In the range 69 to 200 (8k) and 166 to 350 (32k) the program is fairly fast, provided that it has been compiled for these upper limits. For larger matrices the program will be comparatively slow, the slower the larger the matrix. This supports the use of the R-completion program to eliminate questionable or unnecessary measures.

The restrictions of the inversion program should also be considered in this connection.

F. TEST PROBLEMS

Test problems for different matrix sizes and options are given in Appendix I.

G. OUTPUT

1. Print-out:
   a. Option ending factor extraction
   b. Factor loadings (1,2,...,m) corresponding to test $z_j$ horizontally, with row identification (1,2,...,n)
   c. Latent roots (1,2,...m)
   d. Sum of latent roots
   e. Communalities (1,2,...n)
   f. Sum of communalities
   g. Sum of starting communalities
2. Card output:

The n x m matrix of factor coefficients, format 12F6.5
Appendix I

1. ARRANGEMENT OF CARDS

(See Appendix E, inter-correlation programs)

   a. Using FORTRAN deck (Fig. 4)

   b. Using binary deck (Fig. 4)

2. EXAMPLE PROBLEMS

<table>
<thead>
<tr>
<th>Example</th>
<th>MATRIX ORDER</th>
<th>TOTAL VARIANCE</th>
<th>NO. OF FACTORS</th>
<th>LOWER BOUND</th>
<th>LAST ROOT</th>
<th>% VAR. EXTR.</th>
<th>FIRST/ EPSILON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n</td>
<td>option 4a</td>
<td>option 1</td>
<td>option 2</td>
<td>option 3</td>
<td>option 4b</td>
<td>option 5</td>
</tr>
<tr>
<td>A</td>
<td>180</td>
<td>n</td>
<td>20</td>
<td>1.50</td>
<td>25</td>
<td>85</td>
<td>.001</td>
</tr>
<tr>
<td>B</td>
<td>100</td>
<td>$\Sigma h_{est}^2$</td>
<td>12</td>
<td>1.00</td>
<td>20</td>
<td>95</td>
<td>.0001</td>
</tr>
<tr>
<td>C</td>
<td>60</td>
<td>n</td>
<td>12</td>
<td>0.50</td>
<td>50</td>
<td>100</td>
<td>.0001</td>
</tr>
<tr>
<td>D</td>
<td>30</td>
<td>$\Sigma n_{est}$</td>
<td>7</td>
<td>0.10</td>
<td>20</td>
<td>50</td>
<td>.00001</td>
</tr>
</tbody>
</table>

3. DIMENSION CARD FOR THE FORTRAN DECK

   For discussion on dimension cards see Appendix E under inter-correlation programs. The procedure to be followed is about the same. The reader is also referred to (VIII. E).

4. MFIRST = ** or * CARD

   See inter-correlation programs, Appendix E. (Here number of rows corresponds to number of observations in the inter-correlation program.)
5. CONTROL CARD

<table>
<thead>
<tr>
<th>Example</th>
<th>N</th>
<th>TOTCOM</th>
<th>NO</th>
<th>CHECK</th>
<th>END</th>
<th>PASENT</th>
<th>EPSLON</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>+00180</td>
<td>+1</td>
<td>+20</td>
<td>+150</td>
<td>+25</td>
<td>+085</td>
<td>+00100</td>
</tr>
<tr>
<td>B</td>
<td>+00100</td>
<td>-1</td>
<td>+15</td>
<td>+100</td>
<td>+20</td>
<td>+095</td>
<td>+00010</td>
</tr>
<tr>
<td>C</td>
<td>+00060</td>
<td>+1</td>
<td>+12</td>
<td>+050</td>
<td>+30</td>
<td>+100</td>
<td>+00010</td>
</tr>
<tr>
<td>D</td>
<td>+00030</td>
<td>-1</td>
<td>+07</td>
<td>+010</td>
<td>+20</td>
<td>+090</td>
<td>+00000</td>
</tr>
</tbody>
</table>

See example problems for explanation.

6. FORMAT CARD

See (VIII. B) special features, and also Appendix A, means and standard deviations program.

7. DATA CARDS

The card output from the R-completion or the inversion program may be used as input data for the factor analysis program. If cards are specially key-punched for this program the procedure outlined in Appendix G (R-completion) has to be followed. However, the complete symmetric matrix has to be punched, including the principal diagonal.

8. JOB NUMBER CARD

See Appendix A, means and standard deviations program.
DIMENSION A(180), C(180), E(40), R(180, 12)

1 READ INPUT TAPE 7,2
2 FORMAT(16,F2.0,13,F4.2,F3.0,F4.2,F6.5)
   REWIND 2
   REWIND 3
   REWIND 4
   MFIRST=24
   M1=MFIRST
   ASSIGN 33 TO LOOP
   ASSIGN 75 TO LOOK
   ASSIGN 175 TO LOAD
   ASSIGN 182 TO LOAN
   ASSIGN 190 TO NEW
   LG=N/MFIRST
   MLAST=MFIRST*LG
   IF (N-MLAST) 7,6
   MLAST=MFIRST
   GO TO 8
   LG=LG+1
   MLAST=N-MLAST
8 IF (N-MFIRST) 9,9,10
9 ASSIGN 34 TO LOOP
   ASSIGN 76 TO LOOK
   ASSIGN 176 TO LOAD
   ASSIGN 185 TO LOAN
   ASSIGN 55 TO NEW
10 VER=0.0
   K=1
   K1=2
   KA=0
   LC=1
   SQM=0.0
   ITAPE=2
   JTAPE=3
   IF (TOTCOM) 12,12,13
12 VAR=0.0
   GO TO 14
13 VAR=N
14 CALL INPUT
   READ INPUT TAPE 7,15,(R(I,K)),I=1,N
15 FORMAT("
   )
   DO 45 LA=1,LG
   IF (LG-LA) 25,25,30
   M1=MLAST
25 DO 32 K=K1,M1
   READ INPUT TAPE 7,15,(R(I,K)),I=1,N
32 CONTINUE
   K1=1
   GO TO LOOP*(33,34)
33 WRITE TAPE JTAPE*(R(I,K)),I=1,N,K=1,M1
34 IF (TOTCOM) 36,36,45
36 DO 44 K=1,M1
   KA=KA+1
   DO 44 I=1,N
   IF (I-KA) 44,44,44
40 VAR=VAR+R(I,K)
44 CONTINUE
CONTINUE
END FILE JTAPE
REWIN D JTAPE
READ INPUT TAPE 7,46,J
FORMAT(110)
WRITE OUTPUT TAPE 6,47,J
FORMAT(43H1 FACTOR ANALYSIS PROGRAM JOB NUMBER 110
DO 50 J=1,N
50 C(J)=0.0
DO 58 I=1,N
58 A(I)=1.
SUM=0.0
DO 65 J=1,N
MA=0
M1=MFIRST
DO 92 LA=1,1,1
IF (LG.LA) 70,70,75
70 M1=MLAST
G0 1000K,(75,76)
READ TAPE JTAPE,,(R(I,K),I=1,N),K=1,M1)
DO 90 90 I=1,N
KA=MA
DO 90 K=1,M1
KA=KA+1
D(I)=D(I)+(R(I,K)*A(KA))
CONTINUE
MA=MA+M1
CONTINUE
REWIN D JTAPE
I=1
CMAX=ABSF(D(I))
DO 100 J=2,N
CMIN=ABSF(D(J))
IF (CMAX=CMIN) 95,100,100
95 CMAX=CMIN
I=J
CONTINUE
DO 105 J=1,N
105 A(J)=D(J)/D(I)
KR=1
DO 120 J=1,N
DIFF=ABSF(A(J)-C(J))
IF (DIFF=EPSLON) 110,110,115
110 KB=KR+1
115 C(J)=A(J)
CONTINUE
IF (KB=N) 60,60,130
DO 135 J=1,N
135 SUM=SUM+(C(J)**2)
E(LC)=D(I)
IF (E(LC)) 240,140,140
140 SOM=SOM+E(LC)
ROOT=SQR(CMAX)
SUM=SQR(SUM)
DO 145 J=1,N
C(J)=ROOT*(C(J)/SUM)
VER=VER+(C(J)**2)
145
WRITE TAPE 4*(C(J),J=1,N)
WRITE OUTPUT TAPE 6*345*(C(J),J=1,N)
PUNCH 28U*(C(J),J=1,N)
IF(E(LC)-CHECK) 235*235*150
SUIT=VER/VAR
IF(QUIT=PASENT) 155*230*230
STOP=E(1)/E(LC)
IF(STOP=END) 160*225*225
IF(LC=NO) 165*225*225
M1=MFIRST
M=0
DO 185 LA=1,LC
IF (LG=LA) 175,175,175
M1=MLAST
GO TO LOAD*(175*176)
READ TAPE ITAPE*(*(R(I,K),I=1,N),K=1,M1)
DO 180 I=1,N
KA=MA
DO 180 K=1,M1
KA=KA+1
R(I,K)=R(I,K)-(C(I)*C(KA))
CONTINUE
GO TO LOAD*(182*185)
WRITE TAPE ITAPE*(*(R(I,K),I=1,N),K=1,M1)
MA=MA+M1
CONTINUE
END FILE ITAPE
REWIND ITAPE
REWIND JTAPE
LC=LC+1
GO TO NEW*(190*225*55)
JTAPE=2
ITAPE=3
ASSIGN 200 TO NEW
GO TO 55
JTAPE=3
ITAPE=2
ASSIGN 190 TO NEW
GO TO 55
WRITE OUTPUT TAPE 6*221
FORMAT(4,H0) DESIRED NUMBER OF FACTORS EXTRACTED
GO TO 250
WRITE OUTPUT TAPE 6*226
FORMAT(55,H0) LATENT ROOT LESS OR EQUAL TO FRACTION OF 1ST ROOT
GO TO 250
WRITE OUTPUT TAPE 6*231
FORMAT(35,H0) REQUIRED PERCENTAGE ACHIEVED
GO TO 250
WRITE OUTPUT TAPE 6*236
FORMAT(55,H0) LATENT ROOT LESS THAN OR EQUAL TO DESIRED VALUE
GO TO 250
WRITE OUTPUT TAPE 6*241
FORMAT(30,H0) FIRST NEGATIVE LATENT ROOT
LC=LC-1
END FILE 4
REWIND 4
REWIND ITAPE
REWIND JTAPE
WRITE OUTPUT TAPE 6*255
255 FORMAT(20H0 FACTORS )
   SUM=0.0
   DO 270 I=1,N
270 C(I)=0.0
   IF (LC-MFIRST) 300,300,272
   DO 298 J=1,N
      DO 295 K=1,LC
         READ TAPE 4*(A(I),I=1,N)
         D(K)=A(J)
         C(J)=C(J)+(D(K)**2)
      CONTINUE
   REWIND 4
   PUNCH 280*(D(I),I=1,LC)
280 FORMAT (12FS,3)
   WRITE OUTPUT TAPE 6*285,J*(D(I),I=1,LC)
285 FORMAT(1H0/10F11,3))
298 CONTINUE
   GO TO 312
300 DO 305 K=1,LC
305 READ TAPE 4*, (R(I,K),I=1,N)
   REWIND 4
   DO 310 I=1,N
   PUNCH 280*, (R(I,K),K=1,LC)
   WRITE OUTPUT TAPE 6*285*,(R(I,K),K=1,LC)
   DO 307 J=1,LC
307 C(I)=C(I)+(R(I,J)**2)
310 CONTINUE
   DO 314 I=1,N
314 SUM=SUM+C(I)
   WRITE OUTPUT TAPE 6*315
315 FORMAT(20H0 LATENT ROOTS )
   WRITE OUTPUT TAPE 6*320*,(E(J),J=1,LC)
320 FORMAT(1H0/16F8,3))
   WRITE OUTPUT TAPE 6*330
330 FORMAT(30H0 SUM OF LATENT ROOTS )
   WRITE OUTPUT TAPE 6*335*,SOM
335 FORMAT(1H0/20F3))
   WRITE OUTPUT TAPE 6*340
340 FORMAT(20H0 COMMUNALITIES )
   WRITE OUTPUT TAPE 6*345*,(C(I),I=1,N)
345 FORMAT (1H0/10F11,3))
   WRITE OUTPUT TAPE 6*350
350 FORMAT(30H0 SUM OF COMMUNALITIES )
   WRITE OUTPUT TAPE 6*335*, SUM
   WRITE OUTPUT TAPE 6*351
351 FORMAT(40H0 SUM OF STARTING COMMUNALITIES
   WRITE OUTPUT TAPE 6*335*,VAR
   GO TO 1
*ASSEMBLE, PUNCH OBJECT
REM INPUT
ORG 0
PGRM
PZE END,0,1
PZE
BCD 1INPUT
PZE ENT
REM
ORG 0
REL
ERROR BCD 1ERROR
ENT RTD 7
CLA 7 4
STA CPY
SXD AX1 1
LXA A=1 1
CPY CPY * 1
D=12 TXI ++3 1 1
TSX ERROR 4
TSX ERROR 4
TXL CPY * 1 12
LXD AX1 1
TRA 1 4
A=1 PZE 1
AX1 BSS 1
END SYN *
END

*DATA
IX. ORTHOGONAL ROTATION PROGRAMS

A. DESCRIPTION OF THE PROGRAM

See program print-out (Appendix J).

B. SPECIAL FEATURES

1. The program is designed to accept large input matrices. See limitations.

2. Three options for rotational procedure are given.
   
   Option 1. Raw Varimax, where the initial factor solution is rotated $(5; 10)$.
   
   Option 2. Normal Varimax, where the vectors representing the variables are extended to unit length in the common-factor space. After rotation they are brought back to their original length $(5; 10)$.
   
   Option 3. Quartimax rotation $(5)^*.$

3. The angle of rotation.

   When the machine finds the angle of rotation, it makes the rotation only if this angle is greater than a specified value. This value has to be entered in the control card (see control card in Appendix J).


   Convergence is here defined as the maximization of a function, $f(b_{jp})$, of the raw or normalized factor loadings being rotated (see Discussion).

and is achieved when the value of the function no longer increases by more than a specified decimal accuracy. This decimal accuracy has to be entered in the control card (criterion accuracy). The maximization of $f(b_{jq})$ is called the rotational criterion.

The convergence is dependent on the smallest specified angle of rotation and the number of iterations through the full $m(m-1)/2$ transformations.

A specified number of iterations has to be entered in the control card. If convergence takes place before this number of iterations has been reached, the program automatically prints out the results. However, in case convergence is not achieved within the specified cycles of operation, computation stops, and the program prints out:

"No convergence even after desired iterations." All other computational operations are carried out in any case, and a print-out of the incompletely rotated factors will result. Convergence is usually fast in the first few cycles and becomes slower as cycles are continued. It is suggested that the number of iterations be stated about 20-25, although convergence usually takes place in about 5-10 iterations.

C. DISCUSSION

<table>
<thead>
<tr>
<th>Varimax method</th>
<th>Quartimax method</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_{jp}$, $a_{jq}$ = loadings of variable $z_j$ on factors $p$ and $q$ ($j = 1, 2, \ldots, n$)</td>
<td>Same</td>
</tr>
<tr>
<td>$h^2_j$ = communality of variable $z_j$</td>
<td>Same</td>
</tr>
</tbody>
</table>

154
Varimax method

\[ w = \text{angle of rotation} \]

\[ x_j = a_{jp} \text{ (raw) or } a_{jp}/h_j \text{ (normalized)} \]

\[ y_j = a_{jq} \text{ (raw) or } a_{jq}/h_j \text{ (normalized)} \]

\[ X_j, Y_j = \text{rotated loadings} \]

\[ u_j = x_j^2 - y_j^2 \quad v_j = 2x_j y_j \]

\[ A = \sum u_j \quad B = \sum v_j \]

\[ C = \sum (u_j^2 - v_j^2) \quad D = 2\sum u_j v_j \]

Quartimax method

\[ \tan 4w = \frac{nD-2AB}{nC-(A^2-B^2)} \]

\[ \tan 4w = \frac{D}{C} \]

\[ \begin{pmatrix} X_j \\ Y_j \end{pmatrix} = \begin{pmatrix} x_j \\ y_j \end{pmatrix} \begin{pmatrix} \cos w & -\sin w \\ \sin w & \cos w \end{pmatrix} \]

The angle of rotation is given by

\[ (X_j, Y_j) = (x_j, y_j) \begin{pmatrix} \cos w & -\sin w \\ \sin w & \cos w \end{pmatrix} \]

\[ X_j \text{ and } Y_j \text{ are final for the Raw Varimax and the Quartimax.} \]

\[ h_j X_j \text{ and } h_j Y_j \text{ (denormalized) are final for the Normal Varimax.} \]

Raw Varimax criterion:

\[ V_R = n \sum_{p=1}^{m} \sum_{j=1}^{n} b_{jp}^4 - \sum_{p=1}^{m} \left( \sum_{j=1}^{n} b_{jp}^2 \right)^2 \]

where \( b_{jp} \) represent the factor loadings after rotation

Normal Varimax criterion:

\[ V_N = n \sum_{p=1}^{m} \sum_{j=1}^{n} \left( b_{jp}/h_j \right)^2 - \sum_{p=1}^{m} \left( \sum_{j=1}^{n} b_{jp}/h_j \right)^2 \]

The original expressions, given by Kaiser, have been here multiplied by \( n^2 \) for simplicity, because it has no effect on the maximization process. Quartimax
criterion (Neuhaus and Wrigley)*:

\[ q = \sum_{j=1}^{n} \sum_{p=1}^{m} b_{jp} \]

The rotation is carried out for two factors at a time. The procedure involves successive pairings of factors \( p < q = 1, 2, \ldots, m \), and going through the full \( m(m-1)/2 \) transformations for each cycle. Cycles of operations are continued until convergence is achieved (see Special Features IX. B. 3, 4).

D. LIMITATIONS

The program can handle a matrix of over 1000 columns (factors) and practically unlimited number of rows (tests). The capacity of the tape unit is the only limiting factor as to number of rows. Naturally, the use of dimensions of this magnitude is absurd, even from a theoretical point of view. From a practical point of view, however, it is important to stay as much as possible within the core capacity of the computer used. The use of tape for storage slows down the program, the more tape the slower the program. (This applies to all programs.)

Some ideal largest dimensions are given below. These matrices fit into the core of the computer and save computing time considerably (see MFIRST = card in Appendix J)

<table>
<thead>
<tr>
<th>Memory</th>
<th>Tests</th>
<th>Factors</th>
<th>Tests</th>
<th>Factors</th>
<th>Tests</th>
<th>Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>8k</td>
<td>150</td>
<td>32</td>
<td>200</td>
<td>24</td>
<td>300</td>
<td>15</td>
</tr>
<tr>
<td>32k</td>
<td>300</td>
<td>75</td>
<td>500</td>
<td>50</td>
<td>700</td>
<td>35</td>
</tr>
</tbody>
</table>

*For reference see footnote, page 153.
These are safe estimates.
If the number of factors, in above examples, is doubled, the program will
still be fairly fast.

E. RUNNING TIME ESTIMATE

Compilation time is approximately five minutes.

Rotation is extremely fast. Normal Varimax rotation of 10 factors (54
tests) required less than one minute computation time to go through 5
cycles (iterations). Smallest rotation angle was 1° and the criterion
accuracy = .000001.

F. TEST PROBLEMS

Test problems for various matrix sizes and specifications are given in
Appendix J.

G. OUTPUT

Print-out:

1. Number iterations carried

2. Final solution:

   Rotated factor loadings (l,2,...,m) corresponding to test z_j horizontally, with row identification (l,2,...,n)

3. Sum of squares of columns

4. Communalties (l,2,...,n)

5. Sum of communalities

Card output (optional): 12 rotated factor loadings per card, row identification in columns 1-5.
APPENDIX J

1. ARRANGEMENT OF CARDS

(See inter-correlation program, Appendix E)

   a. Using FORTRAN deck (Fig. 4)

   b. Using binary deck (Fig. 4)

2. EXAMPLE PROBLEMS

<table>
<thead>
<tr>
<th>Example</th>
<th>NUMBER OF TESTS</th>
<th>NUMBER OF FACTORS</th>
<th>TANGENT ANGLE</th>
<th>CRITERION ACCURACY</th>
<th>NO. OF ITER.</th>
<th>RAW VMAX</th>
<th>NORM VMAX QUARTIMAX</th>
<th>CARD OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>200</td>
<td>24</td>
<td>.03490</td>
<td>.0000100</td>
<td>25</td>
<td>Normal</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>150</td>
<td>21</td>
<td>.01746</td>
<td>.0000010</td>
<td>20</td>
<td>Raw</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>100</td>
<td>15</td>
<td>.00870</td>
<td>.0000001</td>
<td>15</td>
<td>Q-Max</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

Note: 1. TAN ANGLE = tangent of the angle of rotation. Rotation will take place if the angle is greater than the specified value. Large matrices will converge slowly if this angle is very small. Some values for ready reference:

\[
\begin{align*}
\tan 5^\circ &= .0875, \tan 4^\circ = .0699, \tan 3^\circ = .0524, \tan 2^\circ = .0349, \\
\tan 1^\circ &= .01746, \tan 30' = .00873, \tan 15' = .00435, \tan 6' = .00175, \\
\tan 3' &= .00087, \tan 1' = .00029
\end{align*}
\]

2. See special feature (IX. B).

3. DIMENSION CARD

The procedure to be followed is approximately the same as for inter-correlation programs. However, in the rotation program the number of tests
corresponds to the number of observations in the inter-correlation pro-
gram. The second entry in \( R(200,20) \), i.e. 20 (see dimension card in the
program print-out), corresponds to the maximum number of factors compiled
for. This entry should be greater than, or at least equal to the number
of factors in a particular problem. The first entry (here 200) is the
number of rows (tests) processed in one cycle. If there are, for example,
300 tests and 20 factors, \( R(200,20) \) would process 200 x 20 in the first
and 100 x 20 in the second cycle of operation.

In general, it is advantageous to compile the program to fit the problem,
or if there are many jobs of different dimensions, compile several pro-
grams (see Limitations, IX. D).

If all jobs are equal to or smaller than the dimensions fitting into the
computer core, one compilation for these maximum dimensions is enough.

4. MFIRST = ** or *** CARD

Refer to the first entry in \( R(200,20) \). See program print-out in this
appendix and also explanation in Appendix E.

5. CONTROL CARD

<table>
<thead>
<tr>
<th>Example</th>
<th>IBM CARD COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-6</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td>A</td>
<td>+00200</td>
</tr>
<tr>
<td>B</td>
<td>+00150</td>
</tr>
<tr>
<td>C</td>
<td>+00100</td>
</tr>
</tbody>
</table>

See example problems for explanation.
6. FORMAT CARD

No format card is needed. Input format is 12F6.5. See inversion program for desired format change.

7. DATA CARDS

The card output from the factor analysis program serves as input for the rotation program.

8. JOB NUMBER CARD

See means and standard deviations program, Appendix A.
*COMPILE FORTRAN PRINT SAP PUNCH OBJECT EXECUTE DUMP
VAR200Z

DIMENSION C(200),R(200),E(4),H(200)

1 READ INPUT TAPE 7,2,N,L,CHECK,CRIT,ITER,NOMVAR,CARD
2 FORMAT(216,F6.5,F8.7,13,212)
   RENIND 3
   RENIND 4
   MFIRST=200
   M1=MFIRST
   ITR=ITER
   JTAPE=4
   ITAPE=3
   L=0
   ASSIGN 45 TO JUMP
   TESTS=N
   ASSIGN 80 TO LOOK
   ASSIGN 95 TO NEW
   ASSIGN 143 TO LOOM
   ASSIGN 190 TO JEMP
   WRITE OUTPUT TAPE 6,6
   FORMAT(23H1 INPUT FACTORS )
   IF (N=MFIRST) 3,3,4
   ASSIGN 44 TO JUMP
   LG=N/MFIRST
   MLAST=MFIRST*LG
   IF (N-MLAST) 6,5,6
   MLAST=MFIRST
   GO TO 7
   LG=LG+1
   MLAST=N-MLAST
   7 DO 11 I=1,N
   11 H(I)=0
   DO 35 LA=1, LG
   IF (LG-LA) 12,12,15
   12 M1=MLAST
   DO 22 I=1,M1
   READ INPUT TAPE 7,2,V, R(I,K), K=1,L,C
   20 FORMAT(12F6.3)
   WRITE OUTPUT TAPE 6,199,V, R(I,K), K=1,L,C
   22 CONTINUE
   IF (NOMVAR) 28,28,23
   23 DO 27 I=1,M1
   L=L+1
   DO 25 K=1,L
   25 H(L)=H(L)+(R(I,K)**2)
   H(L)=SORT(H(L))
   DO 26 K=1,L
   R(I,K)=R(I,K)/H(L)
   26 CONTINUE
   28 WRITE TAPE ITAPE*,((R(I,K), I=1,M1), K=1,L,C)
   30 CONTINUE
   END FILE ITAPE
   REWIND ITAPE
   READ INPUT TAPE 7,31,LOOP
   31 FORMAT(11G)
   IF (NOMVAR) 33,35,37
   33 WRITE OUTPUT TAPE 6,34,LOOP
   34 FORMAT(43H1 RAW VARIMAX PROGRAM JOB NO. 110 )
   GO TO 39

162
35 WRITE OUTPUT TAPE 6,36,LOOP
36 FORMAT(43H1, QUARTIMAX PROGRAM JOB NO. 110)
     GO TO 39
37 WRITE OUTPUT TAPE 6,36,LOOP
38 FORMAT(43H1, NORMAL VARIMAX JOB NO. 110)
39 ASSIGN 110 TO LOOP
     M=LC=1
     XCRIT=0.0
40 M1=MFIRST
    DO 140 J=1,M
     L=J+1
    DO 140 K=L,LC
     SUMX=0.0
     SUMY=0.0
     SQXSQY=0.0
     SUMXY=0.0
    DO 54 LA=1,LC
       IF (LG=LAL) 42,42,45
42 M1=MLAST
    GO TO JUMP(44,45,46)
44 ASSIGN 46 TO JUMP
    ASSIGN 82 TO LOOK
    ASSIGN 100 TO NEW
    ASSIGN 140 TO LOOP
    ASSIGN 144 TO LOOM
    ASSIGN 192 TO JEMP
45 READ TAPE ITAPE(((R(I,K)*L=1,M1)*K=1,LC))
46 IF (NOMVAR) 47,51,47
47 DO 50 I=1,M1
    X=(R(I,J)**2) - (R(I,K)**2)
    Y=2*R(I,J)*R(I,K)
    SUMX=SUMX+X
    SUMY=SUMY+Y
    P=X**2
    S=Y**2
    SQXSQY=SQXSQY+P-S
50 CONTINUE
    SUMXY=(TESTS*SUMXY) -(2*(SUMX*SUMY))
    SQXSQY=(TESTS*SQXSQY) -(SUMX**2)+(SUMY**2)
    GO TO 54
51 DO 52 I=1,M1
    X=(R(I,J)**2) - (R(I,K)**2)
    Y=2*R(I,J)*R(I,K)
    P=X**2
    S=Y**2
    SUMXY=SUMXY+2*(X*Y)
    SQXSQY=SQXSQY+(P-S)
52 CONTINUE
54 CONTINUE
    REWIND ITAPE
    M1=MFIRST
    Z=ATN1(SUMXY,SQXSQY)
    IF(Z=3.1415927) 60,60,55
55 Z=Z-6.2831853
50 Z=Z+2.5*Z
    IF (ABS(Z)-CHECK) 140,140,70
70 F1=COS(Z)
F2 = SIN(Z)
DO 100 LA = 1, LG
72 IF (LG = LA) 75, 75, 80
75 M1 = MLAST
GO TO LOOk (80, 82)
80 READ TAPE ITAPE, ((R(I,K1), I = 1, M1), K1 = 1, LC)
82 DO 90 I = 1, M1
   TEMP = (R(I,J)*F1) + (R(I,K)*F2)
   R(I,K) = -R(I,J)*F2 + R(I,K)*F1
   R(I,J) = TEMP
90 CONTINUE
GO TO NEXT (95, 100)
95 WRITE TAPE ITAPE, ((R(I,K1), I = 1, M1, K1 = 1, LC)
100 CONTINUE
END FILE ITAPE
REWIND ITAPE
REWIND TAPE
GO TO LOOP (110, 120, 140)
110 ASSIGN 120 TO LOOP
   JTAPE = 3
   ITAPE = 4
   GO TO 140
120 ASSIGN 110 TO LOOP
   JTAPE = 4
   ITAPE = 3
140 CONTINUE
M1 = MFIRST
REWIND ITAPE
REWIND TAPE
SUMX = 0.0
SUMY = 0.0
DO 141 K = 1, LC
   C(K) = 0.0
141 E(K) = 0.0
   DO 148 LA = 1, LG
   IF (LG = LA) 142, 142, 143
142 M1 = MLAST
   GO TO LOOP (143, 144)
143 READ TAPE ITAPE, ((R(I,K1), I = 1, M1), K1 = 1, LC)
144 DO 148 K = 1, LC
   DO 148 I = 1, M1
   IF (NOMVAR) 145, 146, 145
145 C(K) = C(K) + (R(I,K)**2)
146 E(K) = E(K) + (R(I,K)**4)
148 CONTINUE
   DO 152 K = 1, LC
   SUMX = SUMX + (C(K)**2)
   SUMY = SUMY + E(K)
152 CONTINUE
   IF (NOMVAR) 153, 154, 153
153 SUMY = (TESTS*SUMY) - (SUMX)
154 XCRIT = SUMY - XCRIT
   IF (XCRIT) 159, 155, 155
155 IF (XCRIT - CRIT) 165, 165, 156
156 XCRIT = SUMY
   ITR = ITR - 1
   IF (ITR) 157, 157, 40
157 WRITE OUTPUT TAPE 6, 158
158 IF (NCONV < 1.0) THEN
  WRITE OUTPUT TAPE 6,160
  XCRIT = SUMY
  GO TO 170
160WRITE OUTPUT TAPE 6,160
161FORMAT (55HO, NO CONVERGENCE EVEN AFTER DESIRED ITERATIONS )
162XCRIT = SUMY
163GO TO 170
164WRITE OUTPUT TAPE 6,172
165FORMAT (20HO, CRITERION VALUE DECREASED FOR SOME UNKNOWN REASON )
166I = IER = ITR
167WRITE OUTPUT TAPE 6,175
168FORMAT (33HO, NO OF ITERATIONS ), 110
169M1 = M(1)
170J = 0
171L = 0
172DO 178 I = 1,N
173C(I) = 0.0
174DO 180 I = 1,LC
175E(I) = 0.0
176DO 205 I = 1,LC
177IF (LG - LA) > 185.185.190
178M1 = M(LAST)
179WRITE OUTPUT TAPE 6,190
180IF (CARD) = 202, 202, 202
181READ TAPE ITAPE, (R(I, K), I = 1, M1), K = 1, LC
182WRITE OUTPUT TAPE 6,192
183IF (NLVAR) = 198, 198, 194
184L = L + 1
185DO 195 K = 1, LC
186R(I, K) = R(I, K) * H(L)
187WRITE OUTPUT TAPE 6,199
188FORMAT (1H15, 5X, (15F7.3))
189IF (CARD) = 202, 202, 202
190FORMAT (15, 3X, (12F6.3))
191201J = J + 1
192C(I) = C(I) + (R(I, K) * 2)
193E(K) = E(K) + (R(I, K) * 2)
194DO 205 K = 1, LC
195CONTINUE
196DO 210 K = 1, LC
197SUMY = SUMY + E(K)
198WRITE OUTPUT TAPE 6,220
199FORMAT (40HO, SUM OF SQUARES OF COLUMNS )
200WRITE OUTPUT TAPE 6,225
201FORMAT (1H0(F16.3, 12F8.3))
202FORMAT (1HO(F16.3))
203WRITE OUTPUT TAPE 6,240
204FORMAT (20HO, COMMUNALITIES )
205WRITE OUTPUT TAPE 6,245
206FORMAT (3HO, SUM OF COMMUNALITIES )
207WRITE OUTPUT TAPE 6,250
208FORMAT (3HO, CRITERION VALUE )
209WRITE OUTPUT TAPE 6,255
210XCHIT
211GO TO 1
X. APPENDIX K

1. TEST DATA

Test data for 56 observations and 9 variables are provided in Table I. This information, when punched on cards, can be used for checking purposes:

a. Whenever a new compilation of a program is made

b. When control cards for the different features of the programs are made out

In order to test programs requiring matrix input, the test data deck must first be processed using the proper preceding program or programs. The test data format is 9F8.3. The data card entry for missing data is +0009999, though in the no missing data programs this number is treated as a real observation (i.e. +9.999). The control card entry for missing data should be +009.999 (eight punched columns, preferably including the decimal point). There are 9 missing data "observations" on each variable. The test data observations have been divided into four groups for checking out the t-test and analysis of variance programs:

<table>
<thead>
<tr>
<th>Group</th>
<th>Number observations</th>
<th>Number missing data</th>
<th>N reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>Total</td>
<td>56</td>
<td>9</td>
<td>47</td>
</tr>
</tbody>
</table>
### TABLE I

<table>
<thead>
<tr>
<th>TEST DATA</th>
<th>&amp;</th>
<th>TEST DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAR. 1</td>
<td>&amp;</td>
<td>VAR. 1</td>
</tr>
<tr>
<td>VAR. 2</td>
<td>&amp;</td>
<td>VAR. 2</td>
</tr>
<tr>
<td>VAR. 3</td>
<td>&amp;</td>
<td>VAR. 3</td>
</tr>
<tr>
<td>VAR. 4</td>
<td>&amp;</td>
<td>VAR. 4</td>
</tr>
<tr>
<td>VAR. 5</td>
<td>&amp;</td>
<td>VAR. 5</td>
</tr>
<tr>
<td>VAR. 6</td>
<td>&amp;</td>
<td>VAR. 6</td>
</tr>
<tr>
<td>VAR. 7</td>
<td>&amp;</td>
<td>VAR. 7</td>
</tr>
<tr>
<td>VAR. 8</td>
<td>&amp;</td>
<td>VAR. 8</td>
</tr>
<tr>
<td>VAR. 9</td>
<td>&amp;</td>
<td>VAR. 9</td>
</tr>
<tr>
<td>S</td>
<td>&amp;</td>
<td>S</td>
</tr>
<tr>
<td>D</td>
<td>&amp;</td>
<td>D</td>
</tr>
</tbody>
</table>

+011300000040000030000000400000100000050000010000007000000130000010001170000043000062000012100001480000400000111000007200000850000120012700000530000120000016000006000000970000006000000740000010000113000006600000105000010200000720000017000000600000065000001000011300000750000030000001050000106000009700000128000007000000600000010050101130000470000041000011600000123000001100000113000006700000089000010060113000080000003000000111000001700000044000001140000077000000600000010070117000007700000530000012000001120000050000000900000012400001250000010800116000006000000350000011500000105000007000000115000006000000100001090001000000999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999
2. **TEST DATA RESULTS**

Only a few examples are given in Tables II-V to serve as checking points when using the test data deck. The assumption is made that the whole output is right if part of it is right.

**TABLE II**

TEST DATA RESULTS

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Program and Variable Identification</th>
<th>Missing Data</th>
<th>No Missing Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>i = 1 i = 3</td>
<td>i = 1 i = 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>j = 2 j = 4</td>
<td>j = 2 j = 4</td>
</tr>
<tr>
<td>( \bar{X}_i )</td>
<td>14.351 4.513</td>
<td>MSTDA1A</td>
<td>+13.652 + 5.394</td>
</tr>
<tr>
<td>( \bar{X}_j )</td>
<td>5.406 11.819</td>
<td>MSTDA1A</td>
<td>+ 6.144 +11.527</td>
</tr>
<tr>
<td>( \sigma_i )</td>
<td>3.015 2.347</td>
<td>MSTDA1A</td>
<td>+ 3.194 + 2.957</td>
</tr>
<tr>
<td>( \sigma_j )</td>
<td>1.546 1.547</td>
<td>MSTDA1A</td>
<td>+ 2.213 + 1.567</td>
</tr>
<tr>
<td>N_1</td>
<td>47 47</td>
<td>MSTDA1A</td>
<td>56 56</td>
</tr>
<tr>
<td>N_j</td>
<td>47 47</td>
<td>MSTDA1A</td>
<td>56 56</td>
</tr>
<tr>
<td>z_{10,i}</td>
<td>+ 9.999* 9.999*</td>
<td>Z1A</td>
<td>- 1.143 + 1.557</td>
</tr>
<tr>
<td>z_{15,j}</td>
<td>- 0.263 +0.117</td>
<td>Z1A</td>
<td>- 0.517 + 0.302</td>
</tr>
<tr>
<td>r_{ij}</td>
<td>- 0.310 +0.226</td>
<td>R1A</td>
<td>- 0.559 - 0.148</td>
</tr>
<tr>
<td>t_{r_{ij}}</td>
<td>- 2.184 +1.555</td>
<td>R1A</td>
<td>- 4.956 - 1.100</td>
</tr>
<tr>
<td>df_{ij}</td>
<td>45 45</td>
<td>R1A</td>
<td>54 54</td>
</tr>
</tbody>
</table>

*+9.999 = missing data.*
# TABLE III

## TEST DATA RESULTS

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Program and Variable Identification</th>
<th>Missing Data</th>
<th>No Missing Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Program Code</td>
<td>Group $K = 2$, $L = 3$</td>
<td>Group $K = 2$, $L = 3$</td>
</tr>
<tr>
<td>$N_i^K$</td>
<td>13 13 M1A</td>
<td>14 14 M2A</td>
<td></td>
</tr>
<tr>
<td>$N_i^L$</td>
<td>13 13 and</td>
<td>14 14 and</td>
<td></td>
</tr>
<tr>
<td>$\bar{X}_i^K$</td>
<td>11.077 6.308 A1A</td>
<td>11.000 6.571 A2A</td>
<td></td>
</tr>
<tr>
<td>$\bar{X}_i^L$</td>
<td>11.538 6.692 &quot;</td>
<td>11.428 6.928 &quot;</td>
<td></td>
</tr>
<tr>
<td>$\sigma_i^K$</td>
<td>1.706 2.323 &quot;</td>
<td>1.664 2.440 &quot;</td>
<td></td>
</tr>
<tr>
<td>$\sigma_i^L$</td>
<td>1.450 1.797 &quot;</td>
<td>1.453 1.940 &quot;</td>
<td></td>
</tr>
<tr>
<td>df for $t_{i,K,L}^2$</td>
<td>24 24 M1A</td>
<td>26 26 M2A</td>
<td></td>
</tr>
<tr>
<td>df 3 for $F_{i,K,L}^2$</td>
<td>1 1 A1A</td>
<td>1 1 A2A</td>
<td></td>
</tr>
<tr>
<td>df 2 for $F_{i,K,L}^2$</td>
<td>24 24 &quot;</td>
<td>26 26 &quot;</td>
<td></td>
</tr>
<tr>
<td>$t_{i,K,L}^2$</td>
<td>-0.743 -0.472 M1A</td>
<td>-0.726 -0.429 M2A</td>
<td></td>
</tr>
<tr>
<td>$F_{i,K,L}^2$</td>
<td>0.552 0.223 A1A</td>
<td>0.527 0.184 A2A</td>
<td></td>
</tr>
<tr>
<td>all 4 groups</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>df1 $i$</td>
<td>3 3 A1B</td>
<td>3 3 A2B</td>
<td></td>
</tr>
<tr>
<td>df2 $i$</td>
<td>43 43 &quot;</td>
<td>52 52 &quot;</td>
<td></td>
</tr>
<tr>
<td>$F_{i}^2$</td>
<td>12.639 0.739 &quot;</td>
<td>9.986 3.963 &quot;</td>
<td></td>
</tr>
</tbody>
</table>

170
TABLE IV

INVERSION AND ESTIMATION OF COMMUNALITIES PROGRAM

<table>
<thead>
<tr>
<th>Row/Column</th>
<th>$R^{-1}$</th>
<th>$P^*$</th>
<th>Row/Column</th>
<th>$R^{-1}$</th>
<th>$P$</th>
<th>$R^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>0.067</td>
<td>-0.290</td>
<td>1,1</td>
<td>1.769</td>
<td>0.435</td>
<td>0.435</td>
</tr>
<tr>
<td>1,3</td>
<td>0.174</td>
<td>-0.077</td>
<td>3,3</td>
<td>1.760</td>
<td>0.432</td>
<td>0.432</td>
</tr>
<tr>
<td>2,5</td>
<td>0.318</td>
<td>-0.081</td>
<td>5,5</td>
<td>1.745</td>
<td>0.427</td>
<td>0.427</td>
</tr>
</tbody>
</table>

*The original correlations were: $1,2 = -0.310; 1,3 = -0.133; 2,5 = -0.177.$

TABLE V

FACTOR ANALYSIS AND ROTATION PROGRAMS

<table>
<thead>
<tr>
<th>Row $i$</th>
<th>Col. $j$</th>
<th>$FL_{ij}$</th>
<th>$h_i^2$</th>
<th>$l_j$</th>
<th>$FL_{ij}^*$</th>
<th>$\Sigma FL_{ij}^2$</th>
<th>$\Sigma h_i^2$</th>
<th>$\Sigma L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>0.143</td>
<td>0.294</td>
<td>1.564</td>
<td>0.057</td>
<td>1.320</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2,1</td>
<td>0.552</td>
<td>0.500</td>
<td>2.126</td>
<td>0.046</td>
<td>1.600</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7,3</td>
<td>-0.357</td>
<td>0.607</td>
<td>0.812</td>
<td>-0.769</td>
<td>1.581</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


$FL =$ factor loading, $h_i^2 =$ communality, $L =$ latent root, $FL^* =$ rotated factor loading

Note: The sample answers in Tables IV and V are based on the missing data correlation program.

In Table V, $R^2$'s were used in the diagonal and 3 factors were extracted. Decimal accuracy used = .0001.

Sum of starting communalities ($\Sigma h_{est}^2$) = 4.585

171
The option used for rotation was the Normal Varimax, smallest angle of rotation 1°, criterion accuracy = .000001.

Criterion value = 32.578.

Rotation cycles (iterations) needed = 1.
XI. REFERENCES


XII. SUPPLEMENT

Program print-outs for a 32k memory computer

While this publication was in press the IBM-704 (8k) computer at The University of Michigan Computing Center was replaced by the IBM-709. The authors thus had the opportunity to test all programs on this larger machine (32k) after necessary changes had been made in the program cards.

Although the 704 programs need only slight changes to be accepted by the 709, the complete print-outs are given in this supplement.

Approximate maximum dimensions have been indicated under "Limitations" for each program in Sections III - IX above.

Time estimates are not available. In general, the 32k machine will be two to four times faster for programs using tape storage on the 8k, because part of it (or perhaps all of it) will fit into the core of the larger machine.

The 709 program print-outs can be identified by the code (e.g. MSTD1A) on the extreme right of the first line of each print-out.

Note that when the Format is read as input data, beginning and closing parentheses have to be punched, e.g. (9F8.3).
$\textsc{compile fortran} \text{print object} \text{punch object} \text{execute} \text{dump}$

\begin{verbatim}
DIMENSION SUM(24000), SQ(24000), COUNT(24000), R(24000), NX(24000), NY(24000), FORM(13)
EQUIVALENCE (R, SUM), (R, COUNT), (R, SQ), (R, NX), (R, NY)
1 READ INPUT TAPE 7*2*N,LY,NTOTAL,D
2 FORMAT(3I6,F8.3)
   NSTORE=24000
   IA=LY
   IB=IA+LY
   READ INPUT TAPE 7*3*(NX(I), I=1,1A)
3 FORMAT(24I3)
   L=IA+1
   READ INPUT TAPE 7*3*(NY(I), I=L,1B)
   N1=0
   DO 5 LA=1,LY
   IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
   IA=LY
   IC=IB+N1
   ID=IC+N1
   IE=ID+N1
   IF=NSTORE-IE
   MFIRST=IF/N
   M1=MFIRST
   M=IE
   LG=NTOTAL/MFIRST
   MLAST=MFIRST*LG
   IF (NTOTAL-MLAST) 7,6,7
6 MLAST=MFIRST
   GO TO 8
7 LG=LG+1
   MLAST=NTOTAL-MLAST
8 READ INPUT TAPE 7*9*(FORM(I), I=1,13)
9 FORMAT(13A6)
   L=IB+1
   DO 14 J=L,IE
14 COUNT(J)=0*0
   DO 40 J=1,LG
   IA=LY
   IB=IA+LY
   IC=IB+N1
   ID=IC+N1
   IF (LG-J) 15,15,16
15 M1=MLAST
16 DO 20 KA=1,M1
   K=M
   K=K+1
   M=M+N
   READ INPUT TAPE 7*FORM,(R(I), I=K,M)
20 CONTINUE
   M=IE
   DO 40 LA=1,LY
   IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
   DO 40 L=MA,MB
   I=IE-N+L
   IB=IB+1
   IC=IC+1
\end{verbatim}
ID=ID+1
DO 40 KA=1*M1
   I=I+N
   IF (R(I)-D) 25,40,25
25   SUM(IB)=SUM(IB)+R(I)
   SQ(IC)=SQ(IC)+R(I)**2
   COUNT(ID)=COUNT(ID)+1
40   CONTINUE
READ INPUT TAPE 7,41,IA
FORMAT(10)
WRITE OUTPUT TAPE 6,42,IA
FORMAT(44H1 MEAN AND SIGMA PROGRAM JOB NUMBER I10 )
   IA=LY
   IB=IA+LY
   IC=IB+N1
   ID=IC+N1
   I=29
   DO 70 LA=1,LY
       IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
   DO 70 L=MA,MB
       IB=IB+1
       IC=IC+1
       ID=ID+1
   CA=(COUNT(ID)*(COUNT(ID)-1))*(COUNT(ID)-1))
   CB=(COUNT(ID)*SQ(IC))-(SUM(IB)**2)
   IF (CB) 45,45,56
   SQ(IC)=**999,999
   IF (COUNT(ID)) 50,50,54
50   SUM(IB)=**999,999
   GO TO 58
54   SUM(IB)=SUM(IB)/COUNT(ID)
   GO TO 58
56   SQ(IC)=(SQRT(CB/CA))
   SUM(IB)=SUM(IB)/COUNT(ID)
58   I=I+1
   IF (I>30) 65,60,70
60   WRITE OUTPUT TAPE 6,61
61   FORMAT(64H1 VARIABLE STANDARD DEVIATION MEAN 1 NUMBER )
   I=0
65   WRITE OUTPUT TAPE 6,66,L*SQ(IC)*SUM(IB)*COUNT(ID)
66   FORMAT(1H0I14,F19.3,F20.3,F9.0)
70   CONTINUE
   GO TO 1
END
$DATA
$COMPIL FORTRAN PRINT OBJECT PUNCH OBJECT EXECUTE DUMP
MSTD2A
DIMENSION SUM(22000), SQ(22000), R(22000), NX(22000), NY(22000), FORM(1, 13)
EQUIVALENCE (R, SUM), (R, SQ), (R, NX), (R, NY)
1 READ INPUT TAPE 7, 2, N, LY, NTOTAL
2 FORMAT(3I6)
   NSTORE=22000
   COUNT=NTOTAL
   IA=LY
   IB=IA+LY
   READ INPUT TAPE 7, 3, (NX(I), I=1, IA)
3 FORMAT(2413)
   L=IA+1
   READ INPUT TAPE 7, 3, (NY(I), I=L, IB)
   N1=0
   DO 5 LA=1, LY
   IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
   IA=LY
   IC=IB+N1
   IE=IC+N1
   IF=NSTORE-IE
   MFIRST=IF/N
   M1=MFIRST
   M=IE
   LG=NTOTAL/MFIRST
   MLAST=MFIRST*LG
   IF (NTOTAL-MLAST) 7, 6, 7
6 MLAST=MFIRST
   GO TO 8
7 LG=LG+1
   MLAST=NTOTAL-MLAST
8 READ INPUT TAPE 7, 9, (FORM(I), I=1, 13)
9 FORMAT(13A6)
   L=IB+1
   DO 12 J=L, IE
12 SQ(J)=0.0
   DO 40 J=1, LG
   IA=LY
   IB=IA+LY
   IC=IB+N1
   IF (LG-J) 15, 15, 16
15 M1=MLAST
16 DO 20 KA=1, M1
   K=M
   K=K+1
   M=M+N
   READ INPUT TAPE 7, FORM, (R(I), I=K, M)
20 CONTINUE
   M=IE
   DO 40 LA=1, LY
   IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
   DO 40 L=MA, MB
   I=IE-N+L
   IB=IB+1
   IC=IC+1
   DO 40 KA=1, M1
I=I+N
25 SUM(IB)=SUM(IB)+R(I)
   SQ(IC)=SQ(IC)+R(I)**2
40 CONTINUE
   READ INPUT TAPE 7*,IA
41 FORMAT(1I0)
   WRITE OUTPUT TAPE 6*,IA
42 FORMAT(44H1 MEAN AND SIGMA PROGRAM JOB NUMBER I10 )
   IA=LY
   IB=IA+LY
   IC=IB+N1
   I=29
   CA=(COUNT*(COUNT-1*))
   DO 70 LA=1*LY
      IA=IA+1
      MA=NX(LA)
      MB=NY(IA)
   DO 70 L=MA,MB
      IB=IB+1
      IC=IC+1
   CB=(COUNT*SQ(IC)-(SUM(IB)**2)
   IF (CB) 45,45,56
45 SQ(IC)=+999.999
   SUM(IB)=SUM(IB)/COUNT
   GO TO 58
56 SQ(IC)=(SQRT(CB/CA))
   SUM(IB)=SUM(IB)/COUNT
58 I=I+1
   IF (I-30) 65,60,70
60 WRITE OUTPUT TAPE 6*,IA
61 FORMAT(64H1 VARIABLE STANDARD DEVIATION MEAN
   1 NUMBER)
   I=0
65 WRITE OUTPUT TAPE 6*,IA,SQ(IC),SUM(IB),COUNT
66 FORMAT(1H014,F19.3,F20.3,F9.0)
70 CONTINUE
   GO TO 1
END
$DATA
$COMPILE FORTRAN$CLASS$PRINT OBJECT$PUNCH OBJECT$EXECUTE$DUMP
DIMENSION SUM(22000), SQ(22000), COUNT(22000), R(22000), NX(22000),
1 NY(22000), FORM(13), Z(22000)
EQUIVALENCE (R*SUM),(R*CQUN), (R*SQ), (R*NX), (R*NY), (R*Z)
1 READ INPUT TAPE 7,2*N*L*NTOTAL=None*NTWO, CARDS, D
2 FORMAT(516,F2.0,F8.3)
   NSTORE=22000
   REWIND 2
   NPERC=9
   IA=LY
   IB=IA+L
   READ INPUT TAPE 7,3*(NX(I), I=1, IA)
3 FORMAT(24I3)
   L=IA+1
   READ INPUT TAPE 7,3*(NY(I), I=L, IB)
   N1=0
   DO 5 LA=1,LY
   IA=IA+1
5 N1=N1+NY(IA)-NX(LA)+1
   IA=LY
   IC=IB+N1
   ID=IC+N1
   IE=ID+N1
   IF=NSTORE-IE
   MFIRST=IF/N
   M1=MFIRST
   LB=N*M1+IE
   M=IE
   L=IE+1
   LG=NTOTAL/MFIRST
   MLA=N/MLAST
   IF (NTOTAL-MLAST) 7,6,7
6 MLA=N/MLAST
   GO TO 8
7 LG=LG+1
   MLA=NTOTAL-MLAST
8 READ INPUT TAPE 7,9*(FORM(I), I=1,13)
9 FORMAT(13A6)
   L=IB+1
   DO 14 J=L,IE
14 COUNT(J)=0,0
   DO 40 J=1, LG
   IA=LY
   IB=IA+LY
   IC=IB+N1
   ID=IC+N1
   IF (LG-J) 15,15,16
15 M1=MLAST
   LB=N*M1+IE
16 DO 20 KA=1,M1
   K=M
   K=K+1
   M=M+N
   READ INPUT TAPE 7, FORM, (R(I), I=K,M)
20 CONTINUE
   K1=1
   M=IE
   L1=IE+1
   WRITE TAPE 2, (R(I), I=L1, LB)
DO 40 LA=1*LY
   IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
DO 40 L=MA*MB
   I=IE-N+L
   IB=IB+1
   IC=IC+1
   ID=ID+1
DO 40 KA=1*M1
   I=I+N
   IF (R(I)-D) 25,40,25
25 SUM(IB)=SUM(IB)+R(I)
   SQ(IC)=SQ(IC)+R(I)**2
   COUNT(ID)=COUNT(ID)+1
40 CONTINUE
   IA=LY
   IB=IA+LY
   IC=IB+N1
   ID=IC+N1
   END FILE 2
   REWIND 2
   L=IE+NPERC
   I=29
   READ INPUT TAPE 7,41,MA
41 FORMAT(I10)
   WRITE OUTPUT TAPE 6,42,MA
42 FORMAT(40H1,2 PROGRAM JOB NUMBER I10 )
   DO 70 LA=1*LY
   IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
DO 70 L=MA*MB
   IB=IB+1
   IC=IC+1
   ID=ID+1
   CA=(COUNT(ID)*(COUNT(ID)-1))/
   CB=(COUNT(ID)*SQ(IC)-(SUM(IB)**2)
   IF (CB) 45,45,56
   SQ(IC)=D
   IF (COUNT(ID)) 50,50,54
50 SUM(IB) =D
   GO TO 58
54 SUM(IB)=SUM(IB)/COUNT(ID)
   GO TO 58
56 SQ(IC)=SQRT(CB/CA)
   SUM(IB)=SUM(IB)/COUNT(ID)
58 I=I+1
   IF (I-30) 65,60,70
60 WRITE OUTPUT TAPE 6,61
61 FORMAT(64H1, VARIABLE STANDARD DEVIATION MEAN
1 NUMBER)
   I=0
65 WRITE OUTPUT TAPE 6,66,LY,SQ(IC),SUM(IB),COUNT(ID)
66 FORMAT(1H0114,F19.3,F20.3,F9.0)
70 CONTINUE
   M1=MFIRST
   IA=LY
IB=IA+LY
IE=IB+(3*N1)
M=N*M1+IE
L1=IE+1
N2=NONE-1
DO 140 J=1, LG
IA=LY
IF (LG-J) 75, 75, 78
75 M1=MLAST
IB=IA+LY
IE=IB+(3*N1)
M=N*M1+IE
READ TAPE 2*(R(I), I=L1, M)
DO 140 LA=1, LY
IA=IA+1
MA=NX(LA)
MB=NY(IA)
MC=MB-MA+1
LX=MC/NPERC
LZ=NPERC*LX
IF (LZ=MC) 82, 80, 82
80 LZ=NPERC
GO TO 83
82 LX=LX+1
LZ=MC-LZ
83 IG=N
DO 95 K=1, M1
IB=2*LY
IC=IB+N1
ID=IC+N1
IG=IG+N
ID=ID+IG
DO 95 I=MA, MB
IE=ID+I
IE=IE+N
IB=IB+1
IC=IC+1
IF (SUM(IB)-D) 84, 88, 84
84 IF (SQ(IE)-D) 86, 88, 86
86 IF (R(IE)-D) 90, 88, 90
88 Z(IE)=+.999
GO TO 95
90 Z(IE)=(R(IE)-SUM(IB))/SQ(IE)
95 CONTINUE
IB=2*LY
IC=IB+(2*N1)
IG=-N
DO 130 K=1, M1
N3=NTWO-1
IG=IG+N
ID=IC+IG
IE=ID+MA-1
IE=IE+N
LE=IE
LC=NPERC
100 N2=N2+1
DO 120 LB=1, LX
N3=N3+1
IF (LX=LB) 105,105,110
105 LC=LZ
110 L=LE+1
    LE=LE+LC
    IF (CARDS) 113,120,115
113 WRITE OUTPUT TAPE 6,114,N2,N3,(Z(KA),KA=L,LE)
114 FORMAT((1H0I6,I3,9F12.3))
    GO TO 120
115 PUNCH 119,(Z(KA),KA=L,LE),N2,N3
119 FORMAT(9F8.3,I5,I3)
120 CONTINUE
130 CONTINUE
140 CONTINUE
    REWIND 2
    GO TO 1
END
$DATA
$COMPILE FORTRAN$PRINT OBJECT$PUNCH OBJECT$EXECUTE$DUMP

DIMENSION SUM(22000), SQ(22000), R(22000), NX(22000), NY(22000),
Z(22000), FORM(13)
EQUIVALENCE (R, SUM), (R, SQ), (R, NX), (R, NY), (R, Z)

1 READ INPUT TAPE 7, 2, N, L, N, TOTAL, NONE, NTWO, CARDS
2 FORMAT (5I6, F2.0)
COUNT = NTOTAL
NSTORE = 22000
REWIND 2
NPERC = 9
IA = LY
IB = IA + LY
READ INPUT TAPE 7, 3, (NX(I), I = 1, IA)
3 FORMAT (24I3)
L = IA + 1
READ INPUT TAPE 7, 3, (NY(I), I = L, IB)
N1 = 0
DO 5 LA = 1, LY
IA = IA + 1
5 N1 = N1 + NY(IA) - NX(LA) + 1
IA = LY
IC = IB + N1
IE = IC + N1
IF = NSTORE - IE
MFIRST = IF / N
M1 = MFIRST
LB = N * M1 + IE
M = IE
LG = NTOTAL / MFIRST
MLAST = MFIRST * LG
IF (NTOTAL - MLAST) 7, 6, 7
6 MLAST = MFIRST
GO TO 8
7 LG = LG + 1
MLAST = NTOTAL - MLAST
8 READ INPUT TAPE 7, 9, (FORM(I), I = 1, 13)
9 FORMAT (13A6)
L = IB + 1
DO 14 J = L, IE
14 SUM(J) = 0.0
DO 40 J = 1, LG
IA = LY
IB = IA + LY
IC = IB + N1
IF (LG - J) 15, 15, 16
15 M1 = MLAST
LB = N * M1 + IE
16 DO 20 KA = 1, M1
K = M
K = K + 1
M = M + N
READ INPUT TAPE 7, FORM, (R(I), I = K, M)
20 CONTINUE
M = IE
L1 = IE + 1
WRITE TAPE 2, (R(I), I = L1, LB)
DO 40 LA = 1, LY
IA = IA + 1
MA = NX(LA)

184
MB=NY(IA)
DO 40 L=MA*MB
   I=IE-N+L
   IB=IB+1
   IC=IC+1
    DO 40 KA=1*M1
   I=I+N
   25 SUM(IB)=SUM(IB)+R(I)
   SQ(IC)=SQ(IC)+R(I)**2
40 CONTINUE
   IA=LY
   IB=IA+LY
   IC=IB+N1
END FILE 2
RECEIVE 2
L=IE+NPERC
I=29
READ INPUT TAPE 7,41,MA
41 FORMAT(110)
WRITE OUTPUT TAPE 6,42,MA
42 FORMAT(40H1 Z PROGRAM JOB NUMBER I10 )
   CA=(COUNT*(COUNT-1.))
   DO 70 LA=1,LY
   IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
   DO 70 L=MA*MB
   IB=IB+1
   IC=IC+1
   CB=(COUNT*SQ(IC))-(SUM(IB)**2)
   IF (CB) 45,45,56
45 SQ(IC)=+9.999
   SUM(IB)=SUM(IB)/COUNT
   GO TO 58
56 SQ(IC)=SQRT(CB/CA))
   SUM(IB)=SUM(IB)/COUNT
58 I=I+1
   IF (I=30) 65,60,70
60 WRITE OUTPUT TAPE 6,61
61 FORMAT(64H1 VARIABLE STANDARD DEVIATION MEAN
1 NUMBER)
   I=0
65 WRITE OUTPUT TAPE 6,66,LY,SQ(IC),SUM(IB),COUNT
66 FORMAT(1H0114,F19.3,F20.3,F9.0)
70 CONTINUE
   M1=MFIRST
   IA=LY
   IB=IA+LY
   IE=IB+(2*N1)
   M=N*M1+IE
   L1=IE+1
   N2=NONE-1
    DO 140 J=1,LY
   IA=LY
   IF (LY-J) 75,75,78
75 M1=MLAST
   IB=IA+LY
   IE=IB+(2*N1)
M=N*M1+IE
78 READ TAPE 2*(R(I),I=L1,M)
   DO 140 LA=1,LY
   IA=IA+1
   MA=NX(LA)
   MB=NY(IA)
   MC=MB-MA+1
   LX=MC/NPERC
   LZ=NPERC*LX
   IF (LZ=MC) 82,80,82
80 LZ=NPERC
   GO TO 83
82 LX=LX+1
   LZ=MC=LZ
83 IG=-N
   DO 95 K=1,M1
   IB=2*LY
   IC=IB+N1
   IG=IG+N
   ID=IC+IG
   DO 95 I=MA,MB
   IE=ID+I
   IE=IE+N
   IB=IB+1
   IC=IC+1
   IF (SQ(IC)=-9.999) 90,88,90
88 Z(IE)=+9.999
   GO TO 95
90 Z(IE)=(R(IE)-SUM(IB))/SQ(IC)
95 CONTINUE
   IB=2*LY
   IC=IB+N1
   IG=-N
   DO 130 K=1,M1
   N3=NTWO-1
   IG=IG+N
   ID=IC+IG
   IE=ID+MA-1
   IE=IE+N
   LE=IE
   LC=NPERC
100 N2=N2+1
   DO 120 LB=1,LX
   N3=N3+1
   IF (LB-LB)<105,105,110
105 LC=LZ
110 L=LE+1
   LE=LE+LC
   IF (CARDS)<113,120,115
113 WRITE OUTPUT TAPE 6*(114*N2*N3*(Z(KA),KA=L,LE)
114 FORMAT((1H016,13,9F12,3))
   GO TO 120
115 PUNCH 119*(Z(KA),KA=L,LE)*N2*N3
119 FORMAT(9F8,3,I5,13)
120 CONTINUE
130 CONTINUE
140 CONTINUE
   REWIND 2
   GO TO 1
END
$DATA
$COMPILE FORTRAN PRINT OBJECT PUNCH OBJECT EXECUTE DUMP

DIMENSION A(190, 250), SUM(20), SQUARE(10), COUNT(20), NX(25), NY(25)
1*XMEAN(190), YMEAN(190), DEVN(190), LXC(190), LYM(190),
1FORM(13)
EQUIVALENCE (DEVY*, SQUARE), (DEVN*, LC), (MLAST*, YMEAN)
1 READ INPUT TAPE 7*2*N*L*LY*LX*D
2 FORMAT(316*, F8*3)
REWIND 3
REWIND 4
MFIRST = 120
M1 = MFIRST
READ INPUT TAPE 7*3*(NX(I), I=1*, LY)
3 FORMAT(24I3)
READ INPUT TAPE 7*3*(NY(I), I=1*, LY)
READ INPUT TAPE 7*4*(COUNT(I), I=1*, LX)
4 FORMAT(24F3*0)
DO 7 I = 1*, LX
J = COUNT(I)
LC(I) = I / MFIRST
MLAST(I) = (MFIRST / LC(I))
IF (J = MLAST(I)) 6*5*6
5 MLAST(I) = MFIRST
GO TO 7
6 LC(I) = LC(I) + 1
MLAST(I) = J - MLAST(I)
7 CONTINUE
READ INPUT TAPE 7*8*(FORM(I), I=1*, 13)
8 FORMAT(13A6)
DO 32 LA = 1*, LX
M1 = MFIRST
LG = LC(LA)
DO 9 J = 1*, N
SUM(J) = 0.0
SQUARE(J) = 0.0
COUNT(J) = 0.0
9 CONTINUE
DO 20 J = 1*, LG
IF (LG - J) 10, 10, 11
10 M1 = MLAST(LA)
11 DO 12 K = 1*, M1
READ INPUT TAPE 7*, FORM*(A(I, K), I=1*, N)
12 CONTINUE
K1 = 1
DO 20 L = 1*, LY
NFIRST = NX(L)
NLAST = NY(L)
DO 20 I = NFIRST, NLAST
DO 20 K = 1*, M1
IF (A(I, K) < D) 17, 20, 17
17 SUM(I) = SUM(I) + A(I, K)
SQUARE(I) = SQUARE(I) + (A(I, K)**2)
COUNT(I) = COUNT(I) + 1
20 CONTINUE
DO 32 L = 1*, LY
NFIRST = NX(L)
NLAST = NY(L)
DO 30 I = NFIRST, NLAST
CB = (COUNT(I) * SQUARE(I)) - (SUM(I)**2)
CA = COUNT(I) * (COUNT(I) - 1*)
187
IF (CA) 22,22,25
22 XMEAN(I)=999.999
   DEVNX(I)=0.0
25 IF (CB) 26,26,27
26 XMEAN(I)=999.999
   DEVNX(I)=0.0
   GO TO 30
27 DEVNX(I)=SQRT(CB/CA)
   XMEAN(I)=SUM(I)/COUNT(I)
30 CONTINUE
   WRITE TAPE 3*,((DEVNX(I),XMEAN(I),COUNT(I)),I=NFIRST,NLAST)
   WRITE TAPE 4*,((DEVNX(I),XMEAN(I),COUNT(I)),I=NFIRST,NLAST)
32 CONTINUE
   END FILE 3
   END FILE 4
   REWIND 3
   REWIND 4
   READ INPUT TAPE 7,8,33,LA
33 FORMAT(I10)
   WRITE OUTPUT TAPE 6,34,LA
34 FORMAT(38H0 MEAN AND TEE PROGRAM JOB NUMBER I10 )
   LA=LX=1
   DO 70 J=1,LA
   DO 35 L=1,LY
   NFIRST=NX(L)
   NLAST=NY(L)
   READ TAPE 3*,((DEVNX(I),XMEAN(I),COUNT(I)),I=NFIRST,NLAST)
35 CONTINUE
   DO 65 K=1,LX
   DO 36 L=1,LY
   NFIRST=NX(L)
   NLAST=NY(L)
   READ TAPE 4*,((DEVNY(I),YMEAN(I),SUM(I)),I=NFIRST,NLAST)
36 CONTINUE
   IF (K-J) 65,65,38
38 SKIP=29
   DO 60 L=1,LY
   NFIRST=NX(L)
   NLAST=NY(L)
   DO 60 I=NFIRST,NLAST
   IF (DEVNX(I)) 39,39,41
39 DEVNX(I)=999.999
   TEE=999.999
   IF (DEVNY(I)) 40,40,44
40 DEVNY(I)=999.999
   TEE=999.999
   GO TO 44
41 IF (DEVNY(I)) 40,40,43
43 TEE=(XMEAN(I)-YMEAN(I))/(SQRT((DEVNX(I)**2/COUNT(I))+(DEVNY(I)**2/
   SUM(I))))
44 DF=(COUNT(I)+SUM(I))-2.
   SKIP=SKIP+1.
45 IF (30-SKIP) 50,46,50
46 WRITE OUTPUT TAPE 6,48
48 FORMAT(95H1 GROUPS VARIABLE N-TOTAL MEAN SIGMA N-TOTA
1L MEAN SIGMA TEE DF )
   SKIP=0.0
50 WRITE OUTPUT TAPE 6,55,J,K,I,COUNT(I),XMEAN(I),DEVNX(I),
SUM(I) * YMEAN(I) * DEVNY(I) * TEE * DF
55 FORMAT(1H0I5, I4, 16, F10.0, 2(F11.3), F6.0, 3(F11.3), F10.0)
60 CONTINUE
65 CONTINUE
REWIND 4
70 CONTINUE
REWIND 3
GO TO 1
END

$DATA
$COMPILE FORTRAN$PRINT OBJECT$PUNCH OBJECT$EXECUTE$DUMP

**M2A200**

DIMENSION A(200), SUM(200), SQUARE(200), COUNT(200), NX(25), NY(25)

1*XMEAN(200), DEVNY(200), YMEAN(200), DEVNX(200), LC(200), MLAST(200),
1*FORM(13)

EQUIVALENCE (DEVNY, LC), (MLAST, XMEAN), (SUM, XMEAN), (SQUARE, DEVNX)

1 READ INPUT TAPE 7, 2*N, LY, LX

2 FORMAT(316)

RE WIND 3
REWIND 4
M FIRST=100
M1=M FIRST
READ INPUT TAPE 7, 3*(NX(I), I=1, LY)

3 FORMAT(2413)

READ INPUT TAPE 7, 3*(NY(I), I=1, LY)
READ INPUT TAPE 7, 4*(COUNT(I), I=1, LX)

4 FORMAT(24F3.0)

DO 7 I=1, LX
J=COUNT(I)
LC(I)=(J/MFIRST)
MLAST(I)=(M FIRST*LC(I))
IF (J-MLAST(I)) 6, 5, 6

5 MLAST(I)=M FIRST
GO TO 7

6 LC(I)=LC(I)+1
MLAST(I)=J-MLAST(I)

7 CONTINUE

READ INPUT TAPE 7, 8*(FORM(I), I=1, 13)

8 FORMAT(13A6)

DO 32 LA=1, LX
CA=COUNT(LA)*(COUNT(LA)-1*)
M 1=M FIRST
LG=LC(LA)
DO 9 J=1, N
SUM(J)=0.*0.
SQUARE(J)=0.*0.

9 CONTINUE

DO 20 J=1, LG
IF (LG-J) 10, 11, 11

10 M 1=MLAST(LA)

11 DO 12 K=1, M 1
READ INPUT TAPE 7, FORM(I), A(I,K), I=1, N

12 CONTINUE

DO 20 L=1, LY
N FIRST=NX(L)
NLAST=NY(L)
DO 20 I=N FIRST, NLAST
DO 20 K=1, M 1
16 SUM(I)=SUM(I)+A(I,K)
SQUARE(I)=SQUARE(I)+(A(I,K)**2)

20 CONTINUE

DO 32 L=1, LY
N FIRST=NX(L)
NLAST=NY(L)
DO 30 I=N FIRST, NLAST
CB=(COUNT(LA)*SQUARE(I)-(SUM(I)**2)

23 XMEAN(I)=SUM(I)/COUNT(LA)

25 IF (CB) 26, 26, 27

26 DEVNX(I)=0.*0.
GO TO 30

190
27 DEVNX(I)=SQRT(CB/CA)  
30 CONTINUE  
   WRITE TAPE 3,((DEVNX(I),XMEAN(I)),I=NFIRST,NLAST)  
   WRITE TAPE 4,((DEVNX(I),XMEAN(I)),I=NFIRST,NLAST)  
32 CONTINUE  
   END FILE 3  
   END FILE 4  
   REWIND 3  
   REWIND 4  
   READ INPUT TAPE 7,33,LA  
33 FORMAT(I10)  
   WRITE OUTPUT TAPE 6,34,LA  
34 FORMAT(38H0 MEAN AND TEE PROGRAM JOB NUMBER 110 )  
   LA=LX=1  
   DO 70 J=1,LA  
   DO 35 L=1,LY  
   NFIRST=NX(L)  
   NLAST=NY(L)  
   READ TAPE 3,((DEVNX(I),XMEAN(I)),I=NFIRST,NLAST)  
35 CONTINUE  
   DO 65 K=1,LX  
   DO 36 L=1,LY  
   NFIRST=NX(L)  
   NLAST=NY(L)  
   READ TAPE 4,((DEVNY(I),YMEAN(I)),I=NFIRST,NLAST)  
36 CONTINUE  
   IF (K-J) 65,65,38  
38 SKIP=29.  
   DO 60 L=1,LY  
   NFIRST=NX(L)  
   NLAST=NY(L)  
   DO 60 I=NFIRST,NLAST  
   IF (DEVNX(I)) 39,39,41  
39 TEE=999.999  
   DEVNX(I)=999.999  
   IF (DEVNY(I)) 40,40,44  
40 DEVNY(I) = 999.999  
   TEE = 999.999  
   GO TO 44  
41 IF (DEVNY(I)) 40,40,43  
43 TEE=(XMEAN(I)-YMEAN(I))/((SQRT(((DEVNX(I)**2/COUNT(J))+(DEVNY(I)**2/  
   1COUNT(K))))  
44 DF=(COUNT(J)+COUNT(K))-2.  
   SKIP=SKIP+1.  
45 IF (30-SKIP) 50,46,50  
46 WRITE OUTPUT TAPE 6,48  
48 FORMAT(95H1 GROUPS VARIABLE N-TOTAL MEAN SIGMA N-TOTA  
   1L MEAN SIGMA TEE DF )  
   SKIP=0.0  
50 WRITE OUTPUT TAPE 6,55,J,K,I,COUNT(J)XMEAN(I)DEVNX(I),  
1COUNT(K)YMEAN(I)DEVNY(I)TEE,DF  
55 FORMAT(1H015,i4,i6,f10.0,i2(f11.3,f6.0,3(f11.3,f10.0))  
60 CONTINUE  
65 CONTINUE  
67 CONTINUE  
69 CONTINUE  
70 CONTINUE  
72 CONTINUE  
74 CONTINUE  
76 CONTINUE  
78 CONTINUE  
80 CONTINUE  
82 CONTINUE  
84 CONTINUE  
86 CONTINUE  
88 CONTINUE  
90 CONTINUE  
92 CONTINUE  
94 CONTINUE  
96 CONTINUE  
98 CONTINUE  
100 CONTINUE  
102 CONTINUE  
104 CONTINUE  
106 CONTINUE  
108 CONTINUE  
110 END  
$DATA
191
$COMPILE FORTRAN PRINT OBJECT PUNCH OBJECT EXECUTE DUMP  

A1A200  

DIMENSION A(200), SUM(200), SQUARE(200), COUNT(200), TOTAL(200),  
1 XMEAN(200), DEVNY(200), YMEAN(200), DEVNX(200), SOM(200), SQERE(200),  
1 LC(200), MLAST(200), NX(25), NY(25), FORM(13)  

EQUIVALENCe (MLAST, YMEAN), (LC, DEVNY)  

1 READ INPUT TAPE 7*2*N,LY,LX,D  
2 FORMAT(316,F8.3)  
3 REWIND 3  
4 REWIND 4  
5 MFIRST=100  
6 READ INPUT TAPE 7*3*(NX(I),I=1,LY)  
7 FORMAT(2*4I3)  
8 READ INPUT TAPE 7*3*(NY(I),I=1,LY)  
9 READ INPUT TAPE 7*4*(COUNT(I),I=1,LX)  
10 FORMAT(2*4F3.0)  
11 DO 7 I=1,LX  
12 J=COUNT(I)  
13 LC(I)=(J/MFIRST)  
14 MLAST(I)=(MFirst*LC(I))  
15 IF (J-MLAST(I)) 6,5,6  
16 MLAST(I)=MFirst  
17 GO TO 7  
18 LC(I)=LC(I)+1  
19 MLAST(I)=J-MLAST(I)  
20 CONTINUE  
21 K=1  
22 READ INPUT TAPE 7*8*(FORM(I),I=1,13)  
23 FORMAT(13A6)  
24 DO 32 LA=1,LX  
25 M1=MFirst  
26 LG=LC(LA)  
27 DO 30 J=1,N  
28 SUM(J)=0.0  
29 SQUARE(J)=0.0  
30 COUNT(J)=0.0  
31 CONTINUE  
32 DO 20 J=1,LG  
33 IF (LG-J) 10,10,11  
34 M1=MLAST(LA)  
35 CONTINUE  
36 DO 12 K=1,M1  
37 READ INPUT TAPE 7*FORM,(A(I,K),I=1,N)  
38 CONTINUE  
39 DO 20 L=1,LY  
40 NFirst=NX(L)  
41 NLAST=NY(L)  
42 DO 20 I=NFirst,NLast  
43 DO 20 K=1,M1  
44 IF (A(I,K)-D) 17,20,17  
45 SUM(I)=SUM(I)+A(I,K)  
46 SQUARE(I)=SQUARE(I)+(A(I,K)**2)  
47 COUNT(I)=COUNT(I)+1  
48 CONTINUE  
49 DO 32 L=1,LY  
50 NFirst=NX(L)  
51 NLAST=NY(L)  
52 DO 30 I=NFirst,NLast  
53 CB=(COUNT(I)*SQUARE(I)-(SUM(I)**2)  
54 CA=COUNT(I)*(COUNT(I)-1)  
55 IF (COUNT(I)) 22,22,23  

192
22 XMEAN(I)=0.0
    DEVNX(I)=0.0
    GO TO 30
23 XMEAN(I)=SUM(I)/COUNT(I)
    IF (CA) 24,24,25
24 DEVNX(I)=0.0
    GO TO 30
25 IF (CB) 26,26,27
26 DEVNX(I)=0.0
    GO TO 30
27 DEVNX(I)=SQRT(CB/CA)
30 CONTINUE
    WRITE TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIRST
    1ST,NLAST)
    WRITE TAPE 4,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIRST
    1ST,NLAST)
32 CONTINUE
    END FILE 3
    END FILE 4
    REWIND 3
    REWIND 4
    READ INPUT TAPE 7,33,LA
33 FORMAT(I10)
    WRITE OUTPUT TAPE 6,34,LA
34 FORMAT(45H0 ANALYSIS OF VARIANCE PROGRAM JOB NUMBER 110 )
    LA=LX-1
    DO 70 J=1,LA
    DO 35 L=1,LY
    NFIRST=NX(L)
    NLAST=NY(L)
    READ TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIRST
    1ST,NLAST)
35 CONTINUE
    DO 65 K=1,LX
    DO 36 L=1,LY
    NFIRST=NX(L)
    NLAST=NY(L)
    READ TAPE 4,((SUM(I),SQERE(I),TOTAL(I),YMEAN(I),DEVNY(I)),I=NFIRST
    1ST,NLAST)
36 CONTINUE
    IF (K-J) 65,65,37
37 SKIP=29*
    DO 60 L=1,LY
    NLAST=NX(L)
    NLAST=NY(L)
    DO 60 I=NFIRST,NLAST
    IF (DEVNX(I)) 38,38,49
38 IF (DEVNY(I)) 39,39,43
39 SKIP=SKIP+1*
    IF (30-SKIP) 60,40,41
40 WRITE OUTPUT TAPE 6,57
    SKIP=0*
41 WRITE OUTPUT TAPE 6,42,1,J,K
42 FORMAT(1HOI9,18,14,43H PLEASE CHECK RAW DATA* THANK YOU. )
    GO TO 60
43 SKIP=SKIP+1*
    IF (30-SKIP) 60,44,45
44 WRITE OUTPUT TAPE 6,57

193
Skip=0.0
45 Write output tape 6,47,i,j,k,ymean(i),devny(i)
47 Format(i,1h019,18,14,76h) Please check your data for the two group
1s
49 If (devny(i)) 50,50,54
50 Skip=skip+1.
     If (30*-skip) 60,51,52
51 Write output tape 6,57
Skip=0.0
52 Write output tape 6,53,i,j,k,xmean(i),devnx(i)
53 Format(i,1h019,18,14,52h) Please check your data for the two group
1s
     f14.3,f10.3)
54 Top1=((sum(i)**2)/count(i))+(som(i)**2)/total(i))
     df2=Count(i)+total(i)
     Top2=((sum(i)+som(i)**2)/(df2)
     xbar=top1-top2
     df1=1.
     df2=df2-2.
     r=(square1)+square(i)-top1
     z2=r/df2
     z3=xbar/z2
     skip=skip+1.
     if (30*-skip) 60,55,58
55 Write output tape 6,57
57 Format(118h1 variable groups s.s.between s.s.residual n-1
1n-2 df1 df2 f-ratio mean sigma mean sigma)
Skip=0.0
58 Write output tape 6,59,i,j,k,xbar,r,count(i),total(i),df1,df2,z3,
1xmean(i),devnx(i),ymean(i),devny(i)
59 Format(i,1h019,18,14,f13.3,f14.3,f15.0,f9.3,f11.3,f10.3)
60 Continue
65 Continue
70 Continue
Rewind 3
Go to 1
End
$Data
$COMPILE FORTRAN, PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP

DIMENSION A(200*100), SUM(200), SQUARE(200), COUNT(200), NX(25), NY(25)
1*XMEAN(200), DEVNY(200), YMEAN(200), DEVNX(200), LC(200), MLAST(200),
15*SUM(200), SQUARE(200), FORM(13)
EQUIVALENCE (MLAST, YMEAN) (LC, DEVNY)

1 READ INPUT TAPE 7*2*N*LY*LX
2 FORMAT(3I6)
   REWIND 3
   REWIND 4
   MFIRST=100
   M1=MFIRST
   READ INPUT TAPE 7*3*(NX(I), I=1*LY)
3 FORMAT(24I3)
   READ INPUT TAPE 7*3*(NY(I), I=1*LY)
   READ INPUT TAPE 7*4*(COUNT(I), I=1*LX)
4 FORMAT(24F9.0)
   DO 7 I=1*LX
      J=COUNT(I)
      LC(I)=J/MFIRST
      MLAST(I)=MFIRST*LC(I)
      IF (J-MLAST(I)) 6, 5, 6
5 MLAST(I)=MFIRST
   GO TO 7
6 LC(I)=LC(I)+1
   MLAST(I)=J-MLAST(I)
7 CONTINUE
   READ INPUT TAPE 7*8*(FORM(I), I=1*13)
8 FORMAT(13A6)
   DO 32 LA=1*LX
      M1=MFIRST
      LG=LC(LA)
      CA=COUNT(LA)*(COUNT(LA)-1)
   DO 9 J=1=N
      SUM(J)=0.0
      SQUARE(J)=0.0
9 CONTINUE
   DO 20 J=1, LG
   IF (LG-J) 10, 10, 11
10 M1=MLAST(LA)
11 DO 12 K=1, M1
   READ INPUT TAPE 7, FORM(A(I*K), I=1, N)
12 CONTINUE
   DO 20 L=1,LY
      NFIRST=NX(L)
      NLAST=NY(L)
   DO 20 I=NFIRST, NLAST
   DO 20 K=1, M1
17 SUM(I)=SUM(I)+A(I*K)
   SQUARE(I)=SQUARE(I)+(A(I,K)**2)
20 CONTINUE
   DO 32 L=1,LY
      NFIRST=NX(L)
      NLAST=NY(L)
   DO 30 I=NFIRST, NLAST
      CB=(COUNT(LA)*SQUARE(I))-SUM(I)**2)
23 XMEAN(I)=SUM(I)/COUNT(LA)
25 IF (CB) 26, 26, 27
26 DEVNX(I)=0.0
   GO TO 30
27  DEVNX(I)=SQRT(CB/CA)
30  CONTINUE
    WRITE TAPE 3*1(SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
    WRITE TAPE 4*1(SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
32  CONTINUE
    END FILE 3
    END FILE 4
    REWIND 3
    REWIND 4
    READ INPUT TAPE 7*33,LA
33  FORMAT(I10)
    WRITE OUTPUT TAPE 6*34,LA
34  FORMAT(38HC, ANALYSIS OF VARIANCE  JOB NUMBER I10 )
    LA=LX-1
    DO 70 J=1,LA
    DO 35 L=1,LY
        NFIRST=NX(L)
        NLAST=NY(L)
        READ TAPE 3*1(SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
35  CONTINUE
    DO 65 K=1,LX
    DO 36 L=1,LY
        NFIRST=NX(L)
        NLAST=NY(L)
        READ TAPE 4*1(SOM(I),SQERE(I),YMEAN(I),DEVNY(I)),I=NFIRST,NLAST)
36  CONTINUE
    IF (K=J) 65,65,37
37  SKIP=29,
    DO 60 L=1,LY
        NFIRST=NX(L)
        NLAST=NY(L)
        DO 60 I=NFIRST,NLAST
        IF (DEVNX(I)) 38,38,49
38  IF (DEVNY(I)) 39,39,43
39  SKIP=SKIP+1.
    IF (30-SKIP) 60,40,41
40  WRITE OUTPUT TAPE 6*57
    SKIP=0*0
41  WRITE OUTPUT TAPE 6*42,1*J,K
42  FORMAT(IH0I9,18,14,43H PLEASE CHECK YOUR RAW DATA,THANK YOU. )
    GO TO 60
43  SKIP=SKIP+1.
    IF (30-SKIP) 60,44,45
44  WRITE OUTPUT TAPE 6*57
    SKIP=0*0
45  WRITE OUTPUT TAPE 6*47,1*J,K,YMEAN(I),DEVNY(I)
47  FORMAT(IH0I9,18,14,76H PLEASE CHECK YOUR DATA FOR THE TWO GROUP
    1S
    GO TO 60
49  IF (DEVNY(I)) 50,50,54
50  SKIP=SKIP+1.
    IF (30-SKIP) 60,51,52
51  WRITE OUTPUT TAPE 6*57
    SKIP=0*0
52  WRITE OUTPUT TAPE 6*53,1*J,K,XMEAN(I),DEVNX(I)
53  FORMAT(IH0I9,18,14,52H PLEASE CHECK YOUR DATA FOR THE TWO GROUP
    1S
    GO TO 60
    F14,3,F10,3)
TOPL = ((SUM(I)**2)/COUNT(J)) + ((SUM(I)**2)/COUNT(K))
DF2 = COUNT(J) + COUNT(K)
TOP2 = ((SUM(I)**2)/COUNT(J)) / (DF2)
XBAR = TOPL - TOP2
DF1 = 1
DF2 = DF2 - 2
R = (SQUARE(I) + SQERE(I)) - TOPL
Z2 = R / DF2
Z3 = XBAR / Z2
SKIP = SKIP + 1
IF (30 < SKIP) 60, 55, 58
55 WRITE OUTPUT TAPE 6, 57
57 FORMAT(118H1, VARIABLE GROUPS S.S.BETWEEN S.S.RESIDUAL N-1
1N-2, DF1, DF2, F-RATIO, MEAN, SIGMA, MEAN, SIGMA)
SKIPI = 0
58 WRITE OUTPUT TAPE 6, 59, XBAR, COUNT(J), COUNT(K), DF1, DF2, Z3,
1XMEAN(I), DEVN(I), YMEAN(I), DEVN(Y)
60 CONTINUE
65 CONTINUE
REWIND 4
70 CONTINUE
REWIND 3
GO TO 1
END
$DATA

197
$\text{COMPILe FORTRAN, PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP}
\text{A1B200}
\text{DIMENSION A(200), SUM(200), SQUARE(200), COUNT(200), TOTAL(200),}
\text{1XMEAN(200), DEVN(200), YMEAN(200), DEVNX(200), SOM(200), SQERE(200),}
\text{LC(200), MLAst(200), N(25), NY(25), FORM(13)}
\text{EQUIVALENCE (MLAST, YMEAN), (LC, DEVN)}
1 \text{READ INPUT TAPE 72*N*LY*LX*D}
2 \text{FORMAT(316, F8.3)}
\text{REWOIND 3}
\text{MFIRST=100}
\text{READ INPUT TAPE 723*(NX(I), I=1, LY)}
3 \text{FORMAT(2413)}
\text{READ INPUT TAPE 723*(NY(I), I=1, LY)}
\text{READ INPUT TAPE 724*(COUNT(I), I=1, LX)}
4 \text{FORMAT(24F3.0)}
\text{DO 7 I=1, LX}
\text{J=COUNT(I)}
\text{LC(I)=(J/MFIRST)}
\text{MLAST(I)=(MFIRST*LC(I))}
\text{IF (J=MLAST(I)) 656}
5 \text{MLAST(I)=MFIRST}
\text{GO TO 7}
6 \text{LC(I)=LC(I)+1}
\text{MLAST(I)=J-MLAST(I)}
7 \text{CONTINUE}
\text{READ INPUT TAPE 728*(FORM(I), I=1, 13)}
8 \text{FORMAT(13A6)}
\text{DO 32 LA=1, LX}
\text{M1=MFIRST}
\text{LG=LC(LA)}
\text{DO 9 J=1, N}
\text{SUM(J)=0.0}
\text{SQUARE(J)=0.0}
\text{COUNT(J)=0.0}
9 \text{CONTINUE}
\text{DO 20 J=1, LG}
\text{IF (LG=J) 10, 10, 11}
10 \text{M1=MLAST(LA)}
11 \text{DO 12 K=1, M1}
\text{READ INPUT TAPE 72FORM, (A(I, K), I=1, N)}
12 \text{CONTINUE}
\text{DO 20 L=1, LY}
\text{NFirst=NX(L)}
\text{NLAST=NY(L)}
\text{DO 20 L=NFirst, NLAST}
\text{DO 20 K=1, M1}
\text{IF (A(I, K)-D) 17, 20, 17}
17 \text{SUM(I)=SUM(I)+A(I, K)}
\text{SQUARE(I)=SQUARE(I)+(A(I, K)**2)}
\text{COUNT(I)=COUNT(I)+1}
20 \text{CONTINUE}
\text{DO 32 L=1, LY}
\text{NFirst=NX(L)}
\text{NLAST=NY(L)}
\text{DO 30 I=NFirst, NLAST}
\text{CB=(COUNT(I)*SQUARE(I)-(SUM(I)**2)2)
\text{CA=COUNT(I)*(COUNT(I)-1)}
\text{IF (COUNT(I)) 22, 22, 23}
22 \text{XMEAN(I)=0.0}
\text{DEVNX(I)=0.0}

GO TO 30
23 XMEAN(I)=SUM(I)/COUNT(I)
   IF (CA) 24,24,25
24 DEVNX(I)=0*0
   GO TO 30
25 IF (CB) 26,26,27
26 DEVNX(I)=0*0
   GO TO 30
27 DEVNX(I)=SQRT(CB/CA)
30 CONTINUE
   WRITE TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIR
1ST,NLAST)
32 CONTINUE
   END FILE 3
   REWIND 3
   READ INPUT TAPE 7,33,LA
33 FORMAT(I10)
   WRITE OUTPUT TAPE 6,34,LA
34 FORMAT(45H0 ANALYSIS OF VARIANCE PROGRAM JOB NUMBER       110 )
   LA=LX-1
   SKIP=29*
   DF1 = LX - 1
   GROUP=LX
   DO 35 I=1,N
   LC (I) = 1
   SQERE (I) = 0*0
   SOM (I) = 0*0
   TOTAL (I) = 0*0
35 YMEAN (I) = 0*0
   DO 50 J = 1, LX
   ND = 1, LY
   NFIRST = NX (L)
   NLAST = NY (L)
   READ TAPE 3,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIR
1ST,NLAST)
36 CONTINUE
   DO 50 L = 1, LY
   NFIRST = NX (L)
   NLAST = NY (L)
   DO 50 I = NFIRST, NLAST
   IF (LC(I)) 50,50,37
37 IF (DEVNX(I)) 38,38,39
38 LC(I) = -1
   GO TO 50
39 YMEAN (I) = YMEAN (I) + ((SUM(I)**2) / COUNT(I))
   TOTAL (I) = TOTAL (I) + COUNT (I)
   SOM (I) = SOM (I) + SUM (I)
   SQERE (I) = SQERE (I) + SQUARE (I)
50 CONTINUE
   DO 70 L = 1, LY
   NFIRST = NX (L)
   NLAST = NY (L)
   DO 70 I = NFIRST, NLAST
   IF (LC(I)) 43,43,54
43 SKIP=SKIP+1.
   IF (30+SKIP) 70,44,45
44 WRITE OUTPUT TAPE 6,57
   SKIP=0*0
45 WRITE OUTPUT TAPE 6,47,I
47 FORMAT(1HOI9.40H PLEASE CHECK YOUR DATA )
GO TO 70
54 TOP 2 = (SOM(I)**2) / TOTAL (I)
   DF2=TOTAL(I)-GROUP
   XBAR=YMEAN(I)-TOP2
   R = SQERE (I) - YMEAN (I)
   Z2 = R/DF2
   Z3=XBAR/Z2
   SKIP=SKIP+1.
   IF (30.=SKIP) 70,55,58
55 WRITE OUTPUT TAPE 6,57
57 FORMAT(77H1 VARIABLE S*S*BETWEEN S*S*RESIDUAL DF1
   1 DF2 F-RATIO )
   SKIP=0
58 WRITE OUTPUT TAPE 6,59,1,XBAR,R,DF1,DF2,Z3
59 FORMAT(1HOI12,2F16.3,2F10.0,F10.3)
70 CONTINUE
   REWIND 3
   GO TO 1
END
$DATA
$SCOMPILERE PRINT OBJECT$PUNCH OBJECT$EXECUTE$DUMP A2B200
DIMENSION A(200*100),SUM(200),SQUARe(200),COUNT(200),NX(25),NY(25)
&& XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),LC(200),MLAST(200),
1SUM(200),SOERE(200),FORM(13)
EQUIVALENCE (MLAST,YMEAN),(LC,DEVNY)
1READ INPUT TAPE 7,2,N,LX
2FORMAT(316)
REWRiND 3
MFIRST=100
M1=MFIRST
READ INPUT TAPE 7,3,(NX(I),I=1,LX)
3FORMAT(2413)
READ INPUT TAPE 7,3,(NY(I),I=1,LY)
READ INPUT TAPE 7,4,(COUNT(I),I=1,LX)
4 FORMAT(24F3,0)
TOTAL=0.0
DO 7 I=1,LX
J=COUNT(I)
TOTAL=TOTAL+COUNT(I)
LC(I)=(J/MFIRST)
MLAST(I)=(MFIRST*LC(I))
IF (J-MLAST(I)) 6,5,6
5 MLAST(I)=MFIRST
GO TO 7
6 LC(I)=LC(I)+1
MLAST(I)=J-MLAST(I)
7 CONTINUE
READ INPUT TAPE 7,8,(FORM(I),I=1,13)
8 FORMAT(13A6)
DO 32 LA=1,LX
M1=MFIRST
LG=LC(LA)
CA=COUNT(LA)*(COUNT(LA)-1)
DO 9 J=1,N
SUM(J)=0.0
SQUARE(J)=0.0
9 CONTINUE
DO 20 J=1,LG
IF (LG-J) 10,10,11
M1=MLAST(LA)
10 M1=M1*M1
READ INPUT TAPE 7,FORM,(A(I,K),I=1,N)
12 CONTINUE
DO 32 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
DO 20 I=NFIRST,NLAST
DO 20 K=1,M1
17 SUM(I)=SUM(I)+A(I,K)
SQUARE(I)=SQUARE(I)+(A(I,K)**2)
20 CONTINUE
DO 32 L=1,LY
NFIRST=NX(L)
NLAST=NY(L)
DO 30 I=NFIRST,NLAST
CB=(COUNT(LA)*SQUARE(I)-(SUM(I)**2)
23 XMEAN(I)=SUM(I)/COUNT(LA)
25 IF (CB) 26,26,27
26 DEVNX(I)=0.0

201
GO TO 30
27  DEVNX(I)=SQRT(CB/CA)
30  CONTINUE
   WRITE TAPE 3,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
32  CONTINUE
   END FILE 3
   REWIND 3
   READ INPUT TAPE 7,33,LA
33  FORMAT(I10)
   WRITE OUTPUT TAPE 6,34,LA
34  FORMAT(38H0, ANALYSIS OF VARIANCE JOB NUMBER 110 )
   LA=LX-1
   DF1 = LX - 1
   GROUP=LX
   DF2=TOTAL-GROUP
   SKIP=29,
   DO 35 I=1,N
      LC (I) = 1
      SQERE (I) = 0.0
      SOM (I) = 0.0
35  YMEAN (I) = 0.0
   DO 50 J = 1, LX
      DO 36 L=1, LY
         NFIRST = NX (L)
         NLAST = NY (L)
         READ TAPE 3,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
36  CONTINUE
   DO 50 L = 1, LY
      NFIRST = NX (L)
      NLAST = NY (L)
   DO 50 I = NFIRST, NLAST
50  IF (LC(I)) 50,50,37
37  IF (DEVNX(I)) 38,38,39
38  LC(I) = -1
   GO TO 50
39  YMEAN(I)=YMEAN(I)+((SUM(I)**2)/COUNT(J))
   SOM (I) = SOM (I) + SUM (I)
   SQERE (I) = SQERE (I) + SQUARE (I)
50  CONTINUE
   DO 70 L = 1, LY
   NFIRST = NX (L)
   NLAST = NY (L)
   DO 70 I = NFIRST, NLAST
70  IF (LC(I)) 43,43,54
43  SKIP=SKIP+1.
    IF (30.*-SKIP) 70,44,45
44  WRITE OUTPUT TAPE 6,57
   SKIP=0.0
45  WRITE OUTPUT TAPE 6,47,I
47  FORMAT(1H019,'40H PLEASE CHECK YOUR DATA ')
   GO TO 70
54  TOP2=(SOM(I)**2)/TOTAL
    XBAR=YMEAN(I)-TOP2
    R = SQERE (I) - YMEAN (I)
72  Z2 = R/DF2
    Z3=XBAR/Z2
   SKIP=SKIP+1.
   IF (30.*-SKIP) 70,55,58
55 WRITE OUTPUT TAPE 6
57 FORMAT(77H1 VARIABLE S*S*BETWEEN S*S*RESIDUAL DF1
1 DF2 F-RATIO )
   SKIP=0.0
58 WRITE OUTPUT TAPE 6
59 FORMAT(1H0I12,2F16.3,2F10.0,F10.3)
70 CONTINUE
   REWIND 3
   GO TO 1
   END
$DATA
*COMPILE FORTRAN, PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP

DIMENSION SUM(100), SOM(100), SQUARE(100), SQUARE(100), PROD(100),
LA(100, 225), TOTAL(100), JA(100, 225), FORM(13)
1 READ INPUT TAPE 7, 2*N, NFIRST, NLAST, NEND, COUNT, TVALUE, CARD, D
2 FORMAT(4I6, F6.0, F4.2, F2.0, F8.3)
   EQUIVALENCE (JA, A)
   REWIND 3
   REWIND 4
   MFIRST=225
   M1=MFIRST
   JUMP=COUNT
   READ INPUT TAPE 7, (FORM(I), I=1, 13)
3 FORMAT(13A6)
   LG=(JUMP/MFIRST)
   MLAST=(MFIRST*LG)
   IF (JUMP-MLAST) 5, 4, 5
4 MLAST=MFIRST
   GO TO 7
5 LG=LG+1
   MLAST=JUMP-MLAST
7 ASSIGN 21 TO LOOP
   ASSIGN 28 TO NEW
   ASSIGN 36 TO JUMP
   DO 50 LA=1, LG
   IF (LG-LA) 8, 8, 10
8 M1=MLAST
10 DO 11 K=1, M1
   READ INPUT TAPE 7, FORM(A(I,K), I=1, N)
   CONTINUE
   GO TO LOOP*(21, 22)
21 ASSIGN 22 TO LOOP
   JTAPE=4
   ITAPE=3
   GO TO 23
22 ASSIGN 21 TO LOOP
   JTAPE=3
   ITAPE=4
23 DO 44 I=1, NFIRST*NEND
   GO TO NEW*(28, 32)
28 DO 30 J=1, NEND
   SOM(J)=0.0
   SUM(J)=0.0
   TOTAL(J)=0.0
   SQUARE(J)=0.0
   SQERE(J)=0.0
   PROD(J)=0.0
30 CONTINUE
32 L1=I+1
   GO TO JUMP*(34, 36)
34 READ TAPE ITAPE,**(PROD(J), SOM(J), SUM(J), SQERE(J),
   SQUARE(J), TOTAL(J), J=L1*NEND)
36 DO 42 L=L1*NEND
   DO 42 K=1, M1
   IF (A(I,K)-D) 38, 42, 38
   IF (A(L,K)-D) 40, 42, 40
40 PROD(L)=PROD(L)+(A(I,K)*A(L,K))
   SOM(L)=SOM(L)+A(I,K)
   SUM(L)=SUM(L)+A(L,K)
   SQERE(L)=SQERE(L)+(A(I,K)**2)
SQUARE(L) = SQUARE(L) + (A(L,K) ** 2)
TOTAL(L) = TOTAL(L) + 1

42 CONTINUE
WRITE TAPE JTAPE = ((PROD(J) * SOM(J) * SUM(J) * SQERE(J) ,
1 * SQUARE(J) * TOTAL(J)) * J = L1 * NEND)
44 CONTINUE
REWIND ITAPE
END FILE JTAPE
REWIND JTAPE

49 ASSIGN 32 TO NEW
ASSIGN 34 TO JUMP
50 CONTINUE
READ INPUT TAPE 7 * 52 * NEW
52 FORMAT(110)
WRITE OUTPUT TAPE 6 * 53 * NEW
53 FORMAT(38H1 CORRELATION MATRIX JOB NUMBER 110)
DO 54 L = NFIRST * NEND
JAIL1 = L
54 CONTINUE
DO 90 I = NFIRST * NLAST
L1 = I + 1
READ TAPE JTAPE = ((PROD(J) * SOM(J) * SUM(J) * SQERE(J) ,
1 * SQUARE(J) * TOTAL(J)) * J = L1 * NEND)
DO 80 L = L1 * NEND
IF (TOTAL(L) = 2) 60 * 60 * 61
60 PROD(L) = 9.999
SUM(L) = 9.999
TOTAL(L) = TOTAL(L) - 2
GO TO 80
61 CA = (TOTAL(L) * (TOTAL(L) - 1))
CC = (TOTAL(L) * SQUARE(L) - (SUM(L) ** 2))
CB = (TOTAL(L) * SQERE(L) - (SUM(L) ** 2))
62 IF (CB) 60 * 60 * 64
64 IF (CC) 60 * 60 * 66
66 CB = SQRT(CB / CA)
CC = SQRT(CC / CA)
CB = CA * CB * CC
CC = (TOTAL(L) * PROD(L) - (SOM(L) * SUM(L))
PROD(L) = CC / CB
CC = PROD(L) ** 2
TOTAL(L) = TOTAL(L) - 2
IF (1 * CC) T1, T1, T2
71 SUM(L) = 0.888
GO TO 80
72 SUM(L) = (PROD(L) / (SQRT(1 - CC))) * (SQRT(TOTAL(L)))
80 CONTINUE
WRITE OUTPUT TAPE 6 * 6, 1, ((JA(L,1) * PROD(L) * TOTAL(L) * SUM(L)) , L = L1 ,
1 * INEND)
IF (CARD) 83 * 80 * 81
81 PUNCH 82 * (PROD(L) * L = L1 * NEND)
82 FORMAT(12F6.3)
83 IF (TVALUE) 90 * 90 * 84
84 L = 0
DO 87 M = L1, NEND
CC = ABSF(SUM(M))
IF (CC - TVALUE) 87 * 86 * 86

205
86 L=L+1
    SUM(L)=SUM(M)
    PROD(L)=PROD(M)
    TOTAL(L)=TOTAL(M)
    JA(L,2)=JA(M,1)
87 CONTINUE
    IF (L) 90,90,88
88 WRITE OUTPUT TAPE 6,6,1,((JA(M,2),PROD(M),TOTAL(M),SUM(M)),M=1,L)
90 CONTINUE
    REWIND JTAPE
    GO TO 1
END
$DATA
$COMPILE FORTRAN$ PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP
DIMENSION SUM(140), R(140), SIGMA(140), PROD(140), JA(140)
ITEE(140), FORM(13)
EQUIVALENCE (JA, R)
1 READ INPUT TAPE 7*2*N, NFIRST, NLAST, NEND, COUNT, TVALUE, CARD
2 FORMAT (416, F6.0, F4.2, F2.0)
REWORK 3
REWIND 4
MFIRST=140
M1=MFIRST
READ INPUT TAPE 7*3*(FORM(I), I=1, 13)
FMTHERAT 13A6
JUMP=COUNT
LG=(JUMP/MFIRST)
MLAST=(MFIRST*LG)
IF (JUMP-MLAST) 5, 4, 5, 5
4 MLAST=MFIRST
GO TO 7
5 LG=LG+1
MLAST=JUMP-MLAST
7 DO 12 J=NFIRST, NEND
SIGMA(J)=0.0
SUM(J)=0.0
12 CONTINUE
ASSIGN 21 TO LOOP
ASSIGN 28 TO NEW
ASSIGN 36 TO JUMP
DO 50 LA=1*LG
IF (LG-LA) 14, 14, 16
14 M1=MLAST
16 DO 18 K=1*M1
READ INPUT TAPE 7, FORM*(R(I, K), I=1, N)
18 CONTINUE
DO 20 I=NFIRST, NEND
DO 20 K=1*M1
SUM(I)=SUM(I)+R(I, K)
SIGMA(I)=SIGMA(I)+(R(I, K)**2)
20 CONTINUE
GO TO LOOP, (21, 22)
21 ASSIGN 22 TO LOOP
JTAPE=4
ITAPE=3
GO TO 23
22 ASSIGN 21 TO LOOP
JTAPE=3
ITAPE=4
23 DO 44 I=NFIRST, NLAST
GO TO NEW*(28, 32)
28 DO 30 J=1*NEND
PROD(J)=0.0
30 CONTINUE
32 L1=I+1
GO TO JUMP, (34, 36)
34 READ TAPE ITAPE*, (PROD(J), J=L1, NEND)
36 DO 42 L=L1, NEND
DO 42 K=1*M1
PROD(L)=PROD(L)+((R(I, K)*R(L, K))
42 CONTINUE
WRITE TAPE JTAPE*, (PROD(J), J=L1, NEND)
CONTINUE
REWRITE ITAPE
END FILE ITAPE
REWRITE JTAPE

ASSIGN 32 TO NEW
ASSIGN 34 TO JUMP

CONTINUE
READ INPUT TAPE 7,52,NEW

FORMAT(I10)
WRITE OUTPUT TAPE 6,53,NEW

FORMAT(38H1 CORRELATION MATRIX JOB NUMBER 110 )
CA=COUNT*(COUNT-1)
DF=COUNT-2
SDF=SQRT(DF)
DO 55 I=NFIRST,NEND
JA(I+1)=I
CB=(COUNT*SIGMA(I))-(SUM(I)**2)
SIGMA(I)=SQRT(CB/CA)

CONTINUE
DO 90 I=NFIRST,NLAST
L1=I+1
READ TAPE JTAPE,(PROD(J),J=L1,NEND)
DO 80 L=L1,NEND
TOP=(COUNT*PROD(L))-(SUM(I)*SUM(L))
BELOW=CA*(SIGMA(I)*SIGMA(L))
IF (BELOW) 60,60,65

PROD(L)=9.999
TEE(L)=9.999
GO TO 80

PROD(L)=TOP/BELLO
CC=PROD(L)**2
IF (1-CC) 71,71,72

TEE(L)=8.888
GO TO 80

TEE(L)=(PROD(L)/(SQRT(1-CC)))*SDF

CONTINUE
WRITE OUTPUT TAPE 6,6,I,DF,(JA(L+1),PROD(L),TEE(L)),L=L1,NEND

FORMAT(1HO7,F8.0/4(1H F8.0,F10.3,F10.3))
IF (CARD) 83,90,81

PUNCH 82,(PROD(L),L=L1,NEND)

FORMAT(12F6.3)

IF (VALUE) 90,90,84

L=0
DO 87 M=L1,NEND
CC=ABS(TEE(M))
IF (CC-VALUE) 87,86,86

L=L+1
PROD(L)=PROD(M)
TEE(L)=TEE(M)
JA(L+2)=JA(M+1)

CONTINUE
IF (L) 90,90,88

WRITE OUTPUT TAPE 6,6,I,DF,(JA(M+2),PROD(M),TEE(M)),M=1,L

CONTINUE
REWRITE JTAPE
GO TO 1

$DATA
$COMPILE FORTRAN, PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP

DIMENSION SUM(200), SOM(200), SQUARE(200), SQERE(200), PROD(200),
1A(200,115), TOTAL(200), J(200,115), N(25), NY(25), FORM(13)

EQUIVALENCES (J, A)
1 READ INPUT TAPE 7, 2, N, NBEGIN, NEND, L, COUNT, T, VALUE, CARD, D
2 FORMAT(4I6, F6.0, F4.2, F2.0, F8.3)
RECORD 3
RECORD 4
MFIRST=115
M1=MFIRST
READ INPUT TAPE 7, 3, (NX(I), I=1, L, LY)
3 FORMAT(2413)
RECORD 5
READ INPUT TAPE 7, 3, (NY(I), I=1, L, LY)
READ INPUT TAPE 7, 4, (FORM(I), I=1, 13)
4 FORMAT(13A6)
JUMP=COUNT
LG=(JUMP/MFIRST)
MLAST=(MFIRST*LG)
IF (JUMP-MLAST) 7, 5, 7
5 MLAST=MFIRST
GO TO 8
LG=LG+1
MLAST=JUMP-MLAST
8 ASSIGN 21 TO LOOP
ASSIGN 28 TO NEW
ASSIGN 36 TO JUMP
NA=NEND-NBEGIN+1
K=NBEGIN-1
DO 50 LA=1, LG
IF (LG-LA) 9, 9, 10
IF 11
9 M1=MLAST
10 DO 11 K=1, M1
READ INPUT TAPE 7, FORM, (A(I), K), I=1, N)
11 CONTINUE
GO TO LOOP, 21, 22
21 ASSIGN 22 TO LOOP
JTAXE=4
ITAPE=3
GO TO 23
22 ASSIGN 21 TO LOOP
JTAXE=3
ITAPE=4
DO 44 LE=1, L
MFIRST=NX(LE)
MLAST=NY(LE)
DO 44 I=MFIRST, NLAST
M=0
GO TO NEW, 28, 32
28 DO 30 J=1, NA
SUM(J)=0.0
SUM(J)=0.0
TOTAL(J)=0.0
SQUARE(J)=0.0
SQERE(J)=0.0
PROD(J)=0.0
30 CONTINUE
32 GO TO JUMP, 34, 36
34 READ TAPE ITAPE, (( PROD(J), SOM(J), SUM(J), SQERE(J),
1SQUARE(J), TOTAL(J)), J=1, NA)
DO 42 L=NBEGIN*NEND
  M=M+1
  DO 42 K=1,M1
    IF (A(I,K)-D) 38,42,38
  38 IF (A(L,K)-D) 40,42,40
  40 PROD(M)=PROD(M)+(A(I,K)*A(L,K))
  SOM(M)=SOM(M)+A(I,K)
  SUM(M)=SUM(M)+A(L,K)
  SQERE(M)=SQERE(M)+(A(I,K)**2)
  SQUARE(M)=SQUARE(M)+(A(L,K)**2)
  TOTAL(M)=TOTAL(M)+1*
  42 CONTINUE
  WRITE TAPE JTape *( (PROD(J), SOM(J), SUM(J), SQERE(J),
   1 SQUARE(J), TOTAL(J)) , J=1,NA )
  44 CONTINUE
  REWIND JTape
  END FILE JTape
  REWIND JTape
  49 ASSIGN 32 TO NEW
  ASSIGN 34 TO JUMP
  50 CONTINUE
  READ INPUT TAPE 7*,52*,NEW
  52 FORMAT(110)
  WRITE OUTPUT TAPE 6*,53*,NEW
  53 FORMAT(38H1 CORRELATION MATRIX JOB NUMBER 110 )
  DO 54 M=1,NA
  54 JA(M,1)=K+M
  DO 90 LE=1,LY
    NFIRST=NX(LE)
    NLAST=NY(LE)
    DO 90 I=NFIRST,NLAST
      PROD(JTAPE) *( (PROD(J), SOM(J), SUM(J), SQERE(J),
       1 SQUARE(J), TOTAL(J)) , J=1,NA )
      90 CONTINUE
  IF (TOTAL(M)-2*) 60*,60*,61
  60 PROD(M)=9,999
  SUM(M)=9,999
  TOTAL(M)=TOTAL(M)-2*
  GO TO 80
  61 CA=(TOTAL(M)* (TOTAL(M)-1* )
  CC=((TOTAL(M)*SQUARE(M))-(SUM(M)**2))
  CB=((TOTAL(M)*SQERE(M))-(SOM(M)**2))
  62 IF (CB) 60*,60*,64
  64 IF (CC) 60*,60*,66
  66 CB=SQRT(CB/CA)
  CC=SQRT(CC/CA)
  CB=CA*CB*CC
  CC=((TOTAL(M)*PROD(M))-(SUM(M)*SUM(M)))
  PROD(M)=CC/CB
  TOTAL(M)=TOTAL(M)-2*
  CC=PROD(M)**2
  IF (1*-CC) 71,71,72
  71 SUM(M)=8,888
  GO TO 80
  72 SUM(M)=(PROD(M)/(SQRT(1*-CC)))*(SQRT(TOTAL(M)))
  80 CONTINUE
  WRITE OUTPUT TAPE 6*,61*,(JA(M,1), PROD(M), TOTAL(M), SUM(M)) M=1,NA)
  6 FORMAT(1H0,7/(1H 110,F9.3,F9.0,F9.3,1H 110,F9.3,F9.0,F9.3,1H 110,F
19.3, F9.0, F9.3))
   IF (CARD) 83, 90, 81
81  PUNCH 82*(PROD(M)*M=1, NA)
82  FORMAT(12F6.3)
83  IF (TVALUE) 90, 90, 84
84  L=0
   DO 87 M=1, NA
     CC=ABS(F(SUM(M))
     IF (CC=TVALUE) 87, 86, 86
86  L=L+1
     SUM(L)=SUM(M)
     PROD(L)=PROD(M)
     TOTAL(L)=TOTAL(M)
     JA(L, 2)=JA(M, 1)
87  CONTINUE
     IF (L) 90, 90, 88
88  WRITE OUTPUT TAPE 6, 6, I, ((JA(M, 2), PROD(M), TOTAL(M), SUM(M)), M=1, L)
90  CONTINUE
   REWIND JTAPE
   GO TO 1
END
$DATA
$COMPILE FORTRAN, PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP
R2B200
DIMENSION SUM(200), R(200, 100), SIGMA(200), PROD(200), JA(200, 100),
ITEE(200), NX(25), NY(25), FORM(13)
EQUIVALENCE (JA*R)
1 READ INPUT TAPE 7, 2, N, NBEGIN, NEND, LY, COUNT, TVALUE, CARD
2 FORMAT(4I6, F6.0, F4.2, F2.0)
REWRIND 3
REWRIND 4
MFIRST = 100
M1 = MFIRST
READ INPUT TAPE 7, 3, (NX(I), I = 1, LY)
3 FORMAT(24I3)
READ INPUT TAPE 7, 3, (NY(I), I = 1, LY)
READ INPUT TAPE 7, 4, (FORM(I), I = 1, 13)
4 FORMAT(13A6)
    JUMP = COUNT
    LG = (JUMP/MFIRST)
    MLAST = (MFIRST*LG)
    IF (JUMP-MLAST) 6, 5, 6
5 MLAST = MFIRST
GO TO 7
6 LG = LG + 1
MLAST = JUMP-MLAST
7 DO 10 J = 1, NEND
    SIGMA(J) = 0.0
    SUM(J) = 0.0
10 CONTINUE
    ASSIGN 21 TO LOOP
ASSIGN 28 TO NEW
ASSIGN 36 TO JUMP
NA = NEND-NBEGIN+1
K = NBEGIN-1
DO 50 LA = 1, LG
IF (LG-LA) 11, 11, 13
11 MLAST = MLAST
13 DO 14 K = 1, M1
    READ INPUT TAPE 7, FORM, (R(I, K), I = 1, N)
14 CONTINUE
    DO 20 I = 1, NEND
    DO 20 K = 1, M1
        SUM(I) = SUM(I) + R(I, K)
        SIGMA(I) = SIGMA(I) + (R(I, K)**2)
20 CONTINUE
    GO TO LOOP*(21, 22)
21 ASSIGN 22 TO LOOP
    ITAPE = 3
    JTAPE = 4
    GO TO 23
22 ASSIGN 21 TO LOOP
    JTAPE = 3
    ITAPE = 4
23 DO 44 LE = 1, LY
    NFIRST = NX(LE)
    NLAST = NY(LE)
    DO 44 I = NFIRST, NLAST
        M = 0
        GO TO NEW*(28, 32)
28 DO 30 J = 1, NA
    PROD(J) = 0.0
212
30 CONTINUE
32 GO TO JUMP*(34.36)
34 READ TAPE ITAPE*(PROD(J),J=1,NA)
36 DO 42 L=NBEG1,NEND
   M=M+1
   DO 40 K=1,M
40 PROD(M)=PROD(M)+(R(I,K)*R(L,K))
42 CONTINUE
   WRITE TAPE JTAPE*(PROD(J),J=1,NA)
44 CONTINUE
   REWIND ITAPE
   END FILE JTAPE
   REWIND JTAPE
49 ASSIGN 32 TO NEW
   ASSIGN 34 TO JUMP
50 CONTINUE
   READ INPUT TAPE 7,52,NEW
52 FORMAT(I10)
   WRITE OUTPUT TAPE 6,53,NEW
53 FORMAT(38H1 CORRELATION MATRIX JOB NUMBER I10 )
   DO 54 M=1,NA
54 JA(M,1)=K+M
   CA=COUNT*(COUNT-1.)*
   DF=COUNT-2.*
   SDF=SQRT(DF)
   DO 55 I=FIRST,NEND
   CB=(COUNT*SIGMA(I)-(SUM(I)**2)
   SIGMA(I)=SQRT(CB/CA)
55 CONTINUE
   DO 90 LE=1,LY
   FIRST=NX(LE)
   NLAST=NY(LE)
   DO 90 I=FIRST,NLAST
   M=0
   READ TAPE JTAPE*(PROD(J),J=1,NA)
   DO 80 L=NBEG1,NEND
   M=M+1
   TOP=(COUNT*PROD(M)-(SUM(I)*SUM(L))
   BELOW=CA*(SIGMA(I)*SIGMA(L))
   IF (BELOW) 60,60,65
60 IF ((TOP**2)-BELOW) 9999,9999
   TEE(M)=9999
   GO TO 80
65 PROD(M)=TOP/BELLOW
   CC=PROD(M)**2
   IF (1.-CC) 71,71,72
71 TEE(M)=8888
   GO TO 80
72 TEE(M)=(PROD(M)/(SQRT(1.-CC)))/SDF
80 CONTINUE
   WRITE OUTPUT TAPE 6,8,I,DF*,((JA(M,1)*PROD(M),TEE(M)),M=1,NA)
8 FORMAT(1H017,F8.0/4(1H I8,F10.3,F10.3))
   IF (CARD) 83,90,81
81 PUNCH 82*(PROD(M),M=1,NA)
82 FORMAT(12F6.3)
83 IF (TVALUE) 90,90,84
84 L=0
   DO 87 M=1,NA
213
CC=ABS(TEE(M))
IF (CC=VALUE) 87,86,86

86 L=L+1
TEE(L)=TEE(M)
PROD(L)=PROD(M)
JA(L,2)=JA(M,1)

87 CONTINUE
IF (L) 90,90,88

88 WRITE OUTPUT TAPE 6,8,I,DF,J((JA(M,2),PROD(M),TEE(M)),M=1,L)

90 CONTINUE
REWIND JTAPE
GO TO 1
END

$DATA
$COMPILE FORTRAN PRINT OBJECT PUNCH OBJECT EXECUTE DUMP

DIMENSION R(200,100),C(200),JA(200),FORM(13)
1 READ INPUT TAPE 7,*N,NA,DIAG,INVERS,CARDS
2 FORMAT(216,F2.0,12,F2.0)
    READ INPUT TAPE 7,*J(A(I),I=1,NA)
3 FORMAT(24I3)
    REWIND 2
    REWIND 3
    REWIND 4
    MFIRST=100
    LG=N-NA
    READ INPUT TAPE 7,*4,FORM(I),I=1,13)
4 FORMAT(13A6)
    L=1
    M=1
    M1=NA-1
    IF (LG) 6,16,16
6 DO 15 K=1,M1
    J=M
    L=L+1
    LG=0
    ASSIGN 8 TO NEW
    READ INPUT TAPE 7,*FORM,(C(I),I=L,NA)
7 DO 10 I=L,NA
8 LG=LG+1
9 GO TO NEW,(8,9)
10 CONTINUE
11 IF (LG) 15,15,12
12 IF (INVERS) 14,14,13
13 PUNCH 120,(C(I),I=M,N)
14 WRITE TAPE 2,*(C(I),I=M,N)
15 CONTINUE
16 DO 17 K=1,M1
    L=L+1
    READ INPUT TAPE 7,*4,(C(I),I=L,NA)
17 WRITE TAPE 2,*(C(I),I=L,NA)
18 CONTINUE
19 READ INPUT TAPE 7,19,NEW
20 FORMAT(40H1,COMPLETED CORRELATION MATRIX 110)
21 END FILE 2
22 REWIND 2
23 ASSIGN 32 TO NEW
24 LG=N/MFIRST
25 MLAST=MFIRST*LG
26 IF (N-MLAST) 23,22,23
27 MLAST=MFIRST
28 GO TO 24
29 LG=LG+1
30 MLAST=N-MLAST
31 L=1
32 IA=1
KA=0
IB=0
M1=0
L1=1
DO 36 LA=1*LG
IF (LG-LA) 26,26,27
26 M1=MLAST-1
   ASSIGN 31 TO NEW
   GO TO 28
27 M1=MFIRST
28 DO 30 K=1,M1
   L=L+1
   READ TAPE 2,(R(I,K),I=L,N)
30 CONTINUE
   GO TO NEW,(31,32)
31 M1=M1+1
32 IB=IB+M1
   I1=0
   DO 35 I=IA,IB
      I1=I1+1
      K1=KA
   DO 35 K=1,M1
      K1=K1+1
      IF (I1-K) 34,33,35
33 R(I,K)=1.
   GO TO 35
34 R(I,K)=R(K1,I1)
35 CONTINUE
   WRITE TAPE 3,((R(I,K),I=IA,N),K=1,M1)
      IA=IA+M1
      KA=KA+M1
36 CONTINUE
   END FILE 3
   REWIND 3
   REWIND 2
   I2=0
   DO 80 LA=1*LG
      I1=0
      IC=1
   DO 80 LB=1*LA
      IF (LG-LB) 40,40,42
40 M1=MLAST
   GO TO 43
42 M1=MFIRST
43 READ TAPE 3,((R(I,K),I=IC,N),K=1,M1)
      IC=IC+M1
      IF (LB-LA) 45,55,55
45 IF (LG-LA) 46,46,47
46 M2=MLAST
   GO TO 48
47 M2=MFIRST
48 DO 50 K=1,M1
      I1=I1+1
      IB=I2
   DO 50 I=1,M2
      IB=IB+1
      R(I1,I)=R(IB,K)
50 CONTINUE
55 CONTINUE
60 WRITE TAPE 4,((R(I,K),I=1,N),K=1,M1)
   Rewind 3
   I2=I2+M1
80 CONTINUE
   END FILE 4
   Rewind 4
   KA=0
   DO 135 LA=1,2
      IF (LG-LA) 86,86,87
86 M1=MLAST
   Go To 88
87 M1=MFIRST
88 READ TAPE 4,((R(I,K),I=1,N),K=1,M1)
   DO 130 K=1,M1
      KA=KA+1
      IF (DIAG) 89,89,113
89 CMAX=0.0
   DO 100 I=1,N
      IF (I-KA) 90,100,90
90 CMIN=ABSF(R(I,K))
   IF (CMAX-CMIN) 95,100,100
95 CMAX=CMIN
100 CONTINUE
   R(KA,K)=CMAX
113 IF (CARDS) 122,122,115
115 PUNCH 120,((R(I,K),I=1,N)
120 FORMAT(10F6.3)
122 WRITE OUTPUT TAPE 6,125,KA,((R(I,K),I=1,N)
125 FORMAT(1H0,14/(1H 13F9.3))
130 CONTINUE
135 CONTINUE
   Rewind 4
   Go To 1
END
$DATA
$Compilation, FORTRAN, PRINT OBJECT, PUNCH OBJECT, EXECUTE, DUMP
DIMENSION A(150,150), E(150), R(150), JA(300), F(300)
EQUVALENCE (E(1), F(1)), (R(1), F(151))
1 READ INPUT TAPE 7,2, N, NA, DIAG, SMCINV, SMC, CARDS
2 FORMAT(2I3,14F2.0)
   REWIND 2
   REWIND 3
   L=1
   M=NA-1
   ASSIGN 39 TO NEW
   ASSIGN 86 TO JUMP
   READ INPUT TAPE 7,3, (JA(I), I = 1, NA)
3 FORMAT(24I3)
   I=0
   DO 18 IB=1,M
      J=I+1
      L=L+1
      IA=L
      READ INPUT TAPE 7,4, (F(K), K = L, NA)
4 FORMAT(12F6.3)
   IF (IB=JA(IB)) 18, 10, 10
10 I=I+1
   DO 15 K = IA, NA
      IF (K=JA(K)) 15, 12, 12
12 J = J + 1
   A(I,J) = F(K)
15 CONTINUE
18 CONTINUE
   DO 20 I=1,N
      A(I,I)=0.0
   DO 20 K=1,N
      IF (I=K) 19, 20, 20
19 A(K,I)=A(I,K)
20 CONTINUE
IF (DIAG) 24, 21, 21
21 DO 23 I=1,N
      A(I,I)=1.
23 CONTINUE
IF (SMCINV) 100, 36, 36
100 ASSIGN 190 TO JUMP
   GO TO 105
24 DO 35 I=1,N
      CMAX=0.0
      DO 30 K=1,N
         CMIN=ABSF(A(I,K))
         IF (CMAX=CMIN) 25, 30, 30
25 CMAX=CMIN
30 CONTINUE
   A(I,I)=CMAX
35 CONTINUE
   GO TO 100
36 DO 38 I=1,N
38 WRITE TAPE 2, (A(I,K), K=1,N)
   END FILE 2
   REWIND 2
   DO 81 I=1,N
   GO TO NEW *(39, 81)
39 IF (A(I,I)) 42, 40, 42
40 ASSIGN 81 TO NEW
218
ASSIGN 84 TO JUMP
GO TO 81
42 DO 48 K=1,N
    IF (I-K) 44,48,46
44 R(K)=A(I+K)/A(I+I)
    GO TO 48
46 R(K)=A(K+I)/A(I+I)
48 CONTINUE
    DO 65 J=1,N
        IF (J-I) 50,65,50
60 DO 60 K=1,N
        IF (K-I) 56,60,52
62 IF (J-K) 54,56,60
64 A(J+K)=A(J+K)-R(J)*A(I+K)
      GO TO 60
66 IF (J-K) 58,60,60
68 A(J+K)=A(J+K)-R(J)*A(K+I)
60 CONTINUE
    DO 80 K=1,N
        IF (K-I) 67,69,75
67 A(K+I)=R(K)
      GO TO 80
69 A(I+I)=-1./A(I+I)
      GO TO 80
71 A(I+K)=R(K)
75 A(I+K)=R(K)
80 CONTINUE
81 CONTINUE
105 READ INPUT TAPE 7,82,I
82 FORMAT(110)
    WRITE OUTPUT TAPE 6,83,I
83 FORMAT(44H1 INVERTED MATRIX JOB NUMBER 110 )
      GO TO JUMP,(84,86,190)
84 WRITE OUTPUT TAPE 6,85
85 FORMAT(40H1 DIAGONAL ELEMENT IS ZERO )
      GO TO 1
86 DO 95 I=1,N
    DO 90 K=1,N
        IF (I-K) 87,88,90
87 A(I+K)=-A(I+K)
        A(K+I)=A(I+K)
      GO TO 90
88 A(I+K)=-A(I+K)
90 CONTINUE
    WRITE OUTPUT TAPE 6,94,I,(A(I+K),K=1,N)
94 FORMAT(1H018/(1H 16F11.3))
95 CONTINUE
    DO 145 I=1,N
      READ TAPE 2*(R(K),K=1,N)
    DO 130 L=1,N
      E(L)=0.0
    DO 130 J=1,N
      E(L)=E(L)+(R(J)*A(J,L))
130 CONTINUE
    WRITE OUTPUT TAPE 6,135,I,(E(K),K=1,N)
135 FORMAT(1H018/(1H 16F7.3))
145 CONTINUE
REWIND 2
DO 150 I=1,N
150 E(I)=1./A(I,I)
   IF (SMC) 155,190,170
155 DO 165 I=1,N
   READ TAPE 2*(R(K),K=1,N)
   DO 160 K=1,N
   160 A(I,K)=(E(I)*A(I,K)*E(K))+R(K)
   A(I,I)=A(I,I)-(2.*E(I))
165 CONTINUE
   GO TO 190
170 DO 180 I=1,N
   READ TAPE 2*(A(I,K),K=1,N)
180 A(I,I)=1.-E(I)
   IF (CARDS) 220,220,195
195 PUNCH 200*(A(I,K),K=1,N)
200 FORMAT(12F6.3)
220 WRITE OUTPUT TAPE 6*94*1*(A(I,K),K=1,N)
250 CONTINUE
   GO TO 1
END
$COMPILE FORTRAN*PRINT OBJECT*PUNCH OBJECT*EXECUTE*DUMP
DIMENSION A(150),C(150),D(150),E(150),R(150,150),FORM(13)
1 READ INPUT TAPE 7*2*N,TOTCOM,NO,CHECK,END,PASENT,EPSLON
2 FORMAT(16,F2.0,13,F4.2,F3.0,F4.2,F6.5)
   REWIND 2
   REWIND 3
   REWIND 4
   MFIRST=150
   M1=MFIRST
   ASSIGN 33 TO LOOP
   ASSIGN 75 TO LOOK
   ASSIGN 175 TO LOAD
   ASSIGN 182 TO LOAN
   ASSIGN 190 TO NEW
   LG=N/MFIRST
   MLAST=MFIRST*LG
   IF(N-MLAST) 7*6,7
6 MLAST=MFIRST
   GO TO 8
7 LG=LG+1
   MLAST=N-MLAST
8 IF (N-MFIRST) 9,9,10
9 ASSIGN 34 TO LOOP
   ASSIGN 76 TO LOOK
   ASSIGN 176 TO LOAD
   ASSIGN 185 TO LOAN
   ASSIGN 55 TO NEW
10 VER=0.0
   KA=0
   LC=1
   SOM=0.0
   ITAPE=2
   JTape=3
   IF (TOTCOM) 12,12,13
12 VAR=0.0
   GO TO 14
13 VAR=N
14 READ INPUT TAPE 7*15*(FORM(I),I=1,13)
15 FORMAT(13A6)
   DO 45 LA=1,LG
   IF (LG-LA) 25,25,30
25 M1=MLAST
30 DO 32 K=1,M1
   READ INPUT TAPE 7*FORM,(R(I,K),I=1,N)
   READ INPUT TAPE 7*FORM,(R(I,K),I=1,N)
   READ INPUT TAPE 7*FORM,(R(I,K),I=1,N)
   READ INPUT TAPE 7*FORM,(R(I,K),I=1,N)
   READ INPUT TAPE 7*FORM,(R(I,K),I=1,N)
32 CONTINUE
   GO TO LOOP*(33,34)
33 WRITE TAPE JTape,((R(I,K),I=1,N),K=1,M1)
34 IF (TOTCOM) 36,36,45
36 DO 44 K=1,M1
   KA=KA+1
   DO 44 I=1,N
   IF (I-KA) 44,44,44
40 VAR=VAR+R(I,K)
44 CONTINUE
45 CONTINUE
   END FILE JTape
   REWIND JTape
   READ INPUT TAPE 7*46,J
46 FORMAT(I10)
WRITE OUTPUT TAPE 6,47,J
47 FORMAT(43H1 FACTOR ANALYSIS PROGRAM JOB NUMBER I10 )
   DO 50 J=1,N
50  C(J)=0.0
   DO 55 I=1,N
55  A(I)=1.
      SUM=0.0
   DO 65 J=1,N
65  D(J)=0.0
      MA=0
      M1=FIRST
      DO 92 LA=1, LG
         IF (LG-LA) 70,70,75
70     M1=LAST
74  GO TO LOOK,(75,76)
75  READ TAPE JTAPE,(R(I,K),I=1,N),K=1,M1
76  DO 90 I=1,N
       KA=MA
       DO 90 K=1,M1
         KA=KA+1
         D(I)=D(I)+(R(I,K)*A(KA))
90     CONTINUE
      MA=MA+M1
92  CONTINUE
      REWIND JTAPE
      I=1
      CMAX=ABS(F(D(I))
      DO 100 J=2,N
         CMIN=ABS(F(D(J))
         IF (CMAX-CMIN) 95,100,100
95     CMAX=CMIN
      I=J
100  CONTINUE
      DO 105 J=1,N
       A(J)=D(J)/D(I)
      KB=1
      DO 120 J=1,N
         DIFF=ABS(A(J)-C(J))
         IF (DIFF-EPSILON) 110,110,115
110    KB=KB+1
115   C(J)=A(J)
120  CONTINUE
      IF (KB-N) 60,60,130
130  DO 135 J=1,N
135  SUM=SUM+(C(J)**2)
      E(LC)=D(I)
      IF(E(LC)) 240,140,140
140  SUM=SUM+E(LC)
      ROOT=SQR(CMAX)
      SUM=SQR(SUM)
      DO 145 J=1,N
         C(J)=ROOT*(C(J)/SUM)
      145   VER=VER+(C(J)**2)
      WRITE TAPE 4,C(J),J=1,N
      WRITE OUTPUT TAPE 6,345,C(J),J=1,N
      PUNCH 280,C(J),J=1,N
      IF(E(LC)-CHECK) 235,235,150
150  QUIT=VER/VAR
IF(QUIT=PRESENT) 155,230,230
155 STOP=E(1)/E(LC)
IF(STOP=END) 160,225,225
160 IF(LC=NO) 165,220,220
165 M1=MFIRST
MA=0
DO 185 LA=1,LA
IF (LG=LA) 170,170,175
170 M1=MLAST
174 GO TO LOAD*(175,176)
175 READ TAPE ITAPE*((R(I,K),I=1,N),K=1,M1)
176 DO 180 I=1,N
KA=MA
DO 180 K=1,M1
KA=KA+1
R(I,K)=R(I,K)-(C(I)*C(KA))
180 CONTINUE
GO TO LOAD*(182,185)
182 WRITE TAPE ITAPE*((R(I,K),I=1,N),K=1,M1)
MA=MA+M1
185 CONTINUE
END FILE ITAPE
REWIND ITAPE
REWIND JTape
LC=LC+1
GO TO NEW*(190,200,55)
190 JTape=2
ITAPE=3
ASSIGN 200 TO NEW
GO TO 55
200 JTape=3
ITAPE=2
ASSIGN 190 TO NEW
GO TO 55
220 WRITE OUTPUT TAPE 6,221
221 FORMAT(40HO DESIRED NUMBER OF FACTORS EXTRACTED )
GO TO 250
225 WRITE OUTPUT TAPE 6,226
226 FORMAT(55HO LATENT ROOT LESS OR EQUAL TO FRACTION OF 1ST ROOT )
GO TO 250
230 WRITE OUTPUT TAPE 6,231
231 FORMAT(35HO REQUIRED PERCENTAGE ACHieved )
GO TO 250
235 WRITE OUTPUT TAPE 6,236
236 FORMAT(55HO LATENT ROOT LESS THAN OR EQUAL TO DESIRED VALUE )
GO TO 250
240 WRITE OUTPUT TAPE 6,241
241 FORMAT(30HO FIRST NEGATIVE LATENT ROOT )
LC=LC-1
250 END FILE 4
REWIND 4
REWIND JTape
REWIND ITAPE
WRITE OUTPUT TAPE 6,255
255 FORMAT(20HO FACTORS )
SUM=0.0
DO 270 I=1,N
270 C(I)=0.0

223
IF (LC-MFIRST) 300, 300, 272
272 DO 298 J=1,N
  DO 295 K=1,LC
  READ TAPE 4*(A(I),I=1,N)
  D(K)=A(J)
  C(J)=C(J)+(D(K)**2)
295 CONTINUE
  REWIND 4
  PUNCH 280*(D(I),I=1,LC)
280 FORMAT (12F6.3)
  WRITE OUTPUT TAPE 6, 285*(D(I),I=1,LC)
285 FORMAT (1HO18/(1HO10F11.3))
298 CONTINUE
  GO TO 312
300 DO 305 K=1,LC
305 READ TAPE 4*, (R(I,K),I=1,N)
  REWIND 4
  DO 310 I=1,N
    PUNCH 280*, (R(I,K)*K=1,LC)
    WRITE OUTPUT TAPE 6, 285*(R(I,K)*K=1,LC)
  DO 307 J=1,LC
307 C(I)=C(I)+(R(I,J)**2)
310 CONTINUE
312 DO 314 I=1,N
314 SUM=SUM+C(I)
  WRITE OUTPUT TAPE 6, 315
315 FORMAT (20HO, LATENT ROOTS )
  WRITE OUTPUT TAPE 6, 320*(E(J),J=1,LC)
320 FORMAT (1HO(F16.3,12F8.3))
  WRITE OUTPUT TAPE 6, 330
330 FORMAT (30HO, SUM OF LATENT ROOTS )
  WRITE OUTPUT TAPE 6, 335, SOM
335 FORMAT (1HO(F20.3))
  WRITE OUTPUT TAPE 6, 340
340 FORMAT (20HO, COMMUNALITIES )
  WRITE OUTPUT TAPE 6, 345*(C(I),I=1,N)
345 FORMAT (1HO10F11.3)
  WRITE OUTPUT TAPE 6, 350
350 FORMAT (30HO, SUM OF COMMUNALITIES )
  WRITE OUTPUT TAPE 6, 335, SUM
  WRITE OUTPUT TAPE 6, 351
351 FORMAT (40HO, SUM OF STARTING COMMUNALITIES )
  WRITE OUTPUT TAPE 6, 335, VAR
  GO TO 1
END
$DATA
$COMPIL\_FORTRAN,P\_PRINT\_OBJECT,P\_PUNCH\_OBJECT,EXECUTE,DUMP \n
DIMENSION C(300),R(300),E(50),H(300) \n1 READ INPUT TAPE 7$2,N,LC,CHECK,CRT,ITER,NOMVAR,CARD \n2 FORMAT(216,F6.5,F8.7,I3,212) \n   REWIND 3 \n   REWIND 4 \n   MFIRST=300 \n   M1=MFIRST \n   ITR=ITER \n   JTAPE=4 \n   ITAPE=3 \n   L=0 \n   ASSIGN 45 TO JUMP \n   TESTS=N \n   ASSIGN 80 TO LOOK \n   ASSIGN 95 TO NEW \n   ASSIGN 143 TO LOOM \n   ASSIGN 190 TO JEMP \n   WRITE OUTPUT TAPE 6$8 \n8 FORMAT(23H1,INPUT FACTORS) \n   IF (N-MFIRST) 3*3,4 \n3 ASSIGN 44 TO JUMP \n4 LG=N/MFIRST \n   MLAST=MFIRST*LG \n   IF (N-MLAST) 6*5,6 \n5 MLAST=MFIRST \n   GO TO 7 \n6 LG=LG+1 \n   MLAST=N-MLAST \n7 DO 11 I=1,N \n11 H(I)=0.0 \n   DO 30 LA=1,LG \n   IF (LG=LA) 12,12,15 \n12 M1=MLAST \n15 DO 22 I=1,M1 \n   READ INPUT TAPE 7$20, (R(I,K),K=1,LC) \n20 FORMAT(12F6.3) \n   WRITE OUTPUT TAPE 6$199,1*(R(I,K),K=1,LC) \n22 CONTINUE \n   IF (NOMVAR) 28,28,23 \n23 DO 27 I=1,M1 \n   L=L+1 \n25 H(L)=H(L)+(R(I,K)**2) \n   H(L)=SQRT(H(L)) \n   DO 26 K=1,LC \n26 R(I,K)=R(I,K)/H(L) \n27 CONTINUE \n28 WRITE_TAPE ITAPE,((R(I,K),I=1,M1),K=1,LC) \n30 CONTINUE \n   END FILE ITAPE \n   REWIND ITAPE \n   READ INPUT TAPE 7$31,LOOP \n31 FORMAT(110) \n   IF (NOMVAR) 33,35,37 \n33 WRITE OUTPUT TAPE 6$34,LOOP \n34 FORMAT(43H1,RAW VARIMAX PROGRAM JOB NO:*110) \n   GO TO 39 \n35 WRITE OUTPUT TAPE 6$36,LOOP
36 FORMAT(43H1 QUARTIMAX PROGRAM JOB NO. I10)
37 WRITE OUTPUT TAPE 6*38*LOOP
38 FORMAT(43H1 NORMAL VARIMAX JOB NO. I10)
39 ASSIGN 110 TO LOOP
   M=LC-1
   XCRIT =0.0
40 M1=MFIRST
   DO 140 J=1*M
      L=J+1
   DO 140 K=L*LC
      SUMX=0.0
      SUMY=0.0
      SQXSQY=0.0
      SUMXY=0.0
   DO 54 LA=1*LG
      IF (LG=LA) 42,42,45
      M1=MLAST
      GO TO JUMP,44,45,46)
44 ASSIGN 46 TO JUMP
   ASSIGN 82 TO LOOK
   ASSIGN 100 TO NEW
   ASSIGN 140 TO LOOP
   ASSIGN 144 TO LOOM
   ASSIGN 192 TO JEMP
45 READ TAPE ITAPE,((R(I,K1),I=1,M1),K1=1,LC)
46 IF (NOMVAR) 47,51,47
47 DO 50 I=1,1*M
   X=(R(I,J)**2)-(R(I,K)**2)
   Y=2*R(I,J)*R(I,K)
   SUMX=SUMX+X
   SUMY=SUMY+Y
   P=X**2
   S=Y**2
   SUMXY=SUMXY+(2*(X*Y))
   SQXSQY=SQXSQY+P-S
50 CONTINUE
   SUMXY=(TESTS*SUMXY)-(2*(SUMX*SUMY))
   SQXSQY=(TESTS*SQXSQY)-(SUMX**2)+(SUMY**2)
51 CONTINUE
50 CONTINUE
51 CONTINUE
   REWIND ITAPE
   M1=MFIRST
   Z=ATN1(SUMXY,SQXSQY)
   IF(Z=3,1415927) 55,56,55
55 Z=Z-6,2831853
56 Z=0.25*Z
57 IF (ABSZ(Z)>=CHECK) 140,140,70
58 F1=COS(Z)
59 F2=SIN(Z)
DO 100 LA=1*LG
72 IF (LG-LA) 75,75,80
75 M1=MLAST
   GO TO LOOK*(80,82)
80 READ TAPE ITAPE*((R(I,K1),I=1*M1),K1=1*LC)
82 DO 90 I=1*M1
     TEMP=(R(I,J)*F1)+(R(I,K)*F2)
     R(I,K)=-R(I,J)*F2+R(I,K)*F1
     R(I,J)=TEMP
90 CONTINUE
   GO TO NEW*(95,100)
95 WRITE TAPE JTAPE*((R(I,K1),I=1*M1),K1=1*LC)
100 CONTINUE
   END FILE JTAPE
   REWIND JTAPE
   REWIND ITAPE
   GO TO LOOP*(110,120,140)
110 ASSIGN 120 TO LOOP
   JTAPE=3
   ITAPE=4
   GO TO 140
120 ASSIGN 110 TO LOOP
   JTAPE=4
   ITAPE=3
140 CONTINUE
   M1=MFIRST
   REWIND ITAPE
   REWIND JTAPE
   SUMX=0*0
   SUMY=0*0
   DO 141 K=1*LC
        C(K)=0*0
141 E(K)=0*0
   DO 148 LA=1*LG
        IF (LG-LA) 142,142,143
142 M1=MLAST
   GO TO LOOP*(143,144)
143 READ TAPE ITAPE*((R(I,K1),I=1*M1),K1=1*LC)
144 DO 148 K=1*LC
145 IF (NOMVAR) 145,146,145
146 C(K)=C(K)+(R(I,K)**2)
148 CONTINUE
    DO 152 K=1*LC
         SUMX=SUMX+(C(K)**2)
         SUMY=SUMY+E(K)
152 CONTINUE
    IF (NOMVAR) 153,154,153
153 SUMY=(TESTS*SUMY)-(SUMX)
154 XCRTIT=SUMY-XCRTIT
    IF (XCRTIT) 159,155,155
155 XCRTIT-CRTIT) 165,165,156
156 XCRTIT=SUMY
   ITR=ITR-1
    IF (ITR) 157,157,40
157 WRITE OUTPUT TAPE 6*158
158 FORMAT(55H0   NO CONVERGENCE EVEN AFTER DESIRED ITERATIONS
XCRIT=SUMY
GO TO 170
159 WRITE OUTPUT TAPE 6,160
160 FORMAT(5SH0 CRITERION VALUE DECREASED FOR SOME UNKNOWN REASON )
165 XCRIT=SUMY
170 SUMY=0.0
WRITE OUTPUT TAPE 6,172
172 FORMAT(20H0 FACTORS )
I=ITER=ITR
WRITE OUTPUT TAPE 6,175,I
175 FORMAT(30H0 NO OF ITERATIONS I10)
M1=MFIRST
J=0
L=0
DO 178 I=1,N
178 C(I)=0.0
DO 180 I=1,LC
180 E(I)=0.0
DO 205 LA=1,ILG
182 IF (LG=LA) 185,185,190
185 M1=MLAST
GO TO JEMP,(190,192)
190 READ TAPE ITAPE,((R(I,K),I=1,M1),K=1,LC)
192 DO 205 I=1,M1
IF (NOMVAR) 198,198,194
194 L=L+1
DO 195 K=1,LC
195 R(I,K)=R(I,K)*H(L)
198 WRITE OUTPUT TAPE 6,199,I,(R(I,K),K=1,LC)
199 FORMAT(1H6,5X,(15F7.3))
IF (CARD) 202,202,200
200 PUNCH 201,I,(R(I,K),K=1,LC)
201 FORMAT(I5,3X,(12F6.3))
202 J=J+1
DO 205 K=1,LC
C(J)=C(J)+(R(I,K)**2)
E(K)=E(K)+(R(I,K)**2)
205 CONTINUE
DO 210 K=1,LC
210 SUMY=SUMY+E(K)
WRITE OUTPUT TAPE 6,220
220 FORMAT(40H0 SUM OF SQUARES OF COLUMNS )
WRITE OUTPUT TAPE 6,225,(E(J),J=1,LC)
225 FORMAT (1H0(F16.3,12F8.3))
235 FORMAT (1H0(F20.3)
WRITE OUTPUT TAPE 6,240
240 FORMAT(20H0 COMMUNALITIES )
WRITE OUTPUT TAPE 6,245,(C(I),I=1,N)
245 FORMAT (1H0(F10.3)
WRITE OUTPUT TAPE 6,250
250 FORMAT(30H0 SUM OF COMMUNALITIES )
WRITE OUTPUT TAPE 6,235,255
WRITE OUTPUT TAPE 6,255
255 FORMAT(30H0 CRITERION VALUE )
WRITE OUTPUT TAPE 6,235,XCRIT
GO TO 1
END
$DATA