ADDENDUM TO:

"The JRP Statistical Programs for High Speed Digital Computers", September, 1961.

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:

MSTDIA

```
      DIMENSION............,FORM(13),D(200)
      - - - -
    1 READ INPUT TAPE 7,2,N,LY,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)
      - - - -
```

ZIA

```
      DIMENSION............,Z(22000),D(200)
      - - - -
    1 READ INPUT TAPE 7,2,N,LY,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(IC)-D(I)) 86,88,86
   86 IF(R(IE)-D(I)) 90,88,90
      - - - -
```

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).

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(3I6)
      - - - -
    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.

NBM:jbw
#670
March 13, 1962

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 OF CONTENTS (Concluded)

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 F8.3 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. F8.3, missing data will be punched in the control card +9999999, or +8888888, etc.

   b. F4.2, the control card would read +0009990, or +0007770, etc.

   In the data cards missing data would be punched correspondingly +9999999, or +8888888, etc. in the case of F8.3. In the case of F4.2 the entry would be +999, or +777, etc.

   When a data format smaller than F8.3 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

F5.2	data entry	±xxxx (e.g. + 0356)
	missing data entry or etc.	+9999 +8888
	control card or etc.	+099.990 +088.880

8. Test data for 9 variables and 56 observations are given in appendix K. This information may be used when compiling the programs and to check the out-put. 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

   $\sigma_X = +999.999$

   $\overline{X} = +999.999$

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

$$\sigma_X = \sqrt{\left(\frac{N\sum_{i=1}^{N} X_i^2 - (\sum_{i=1}^{N} X_i)^2}{N(N-1)}\right)}$$

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

Program Category	MAXIMUM NUMBER OF VARIABLES ON AN 8k MEMORY	MAXIMUM NUMBER OF VARIABLES ON A 32k MEMORY
MSTD 1A	1400 *	5000 *
MSTD 2A	1400 *	5000 *

*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 σ) 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 σ).

F. TEST PROBLEMS

The test problems have been designed so that they cover possible occurences, and they are itemized for categories of programs discussed above (MSTD1A, MSTD2A). See examples in Appendix A.

G. OUTPUT

Print-out: Variable no., σ, $\bar{X}$, 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)

Example*	Number of variables	Number of observations**
A	280	369
B	185	206
C	96	100
D	31	100

*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:

	IBM CARD COLUMNS			
Example	1-6	7-12	13-19	19-26
	N	LY	NTOTAL	D
A	+00280	+00001	+00369	+0009999 (or +009.999)
B	+00185	+00001	+00206	+0009999 (or +009.999)
C	+00096	+00002	+00200	+0009999 (or +009.999)
D	+00031	+00003	+00100	+0009999 (or +009.999)

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 = total number of variables in the punched matrix

LY = number of sequences of variables (including the case of single variables) to be calculated

NTOTAL = total number of observations (including missing data)

6. The missing data entry in the example is given for <u>data format</u> F5.3.

b. Control card no. 2

Example	IBM CARD COLUMN				
	1-3	4-6	7-9	10-12	13-15...69-72
A	001	Blank			
B	025	Blank			
C	018	040	Blank		
D	003	018	026	Blank	

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

Example	IBM CARD COLUMN				
	1-3	4-6	7-9	10-12	13-15...69-72
A	280	Blank			
B	179	Blank			
C	018	080	Blank		
D	003	018	026	Blank	

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.

<u>Example 1</u>. 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: 9F8.3 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., 9F8.3 which means 9 variables per card punched with fixed decimal point between the third and fourth digit from the right, the <u>control card</u> 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.

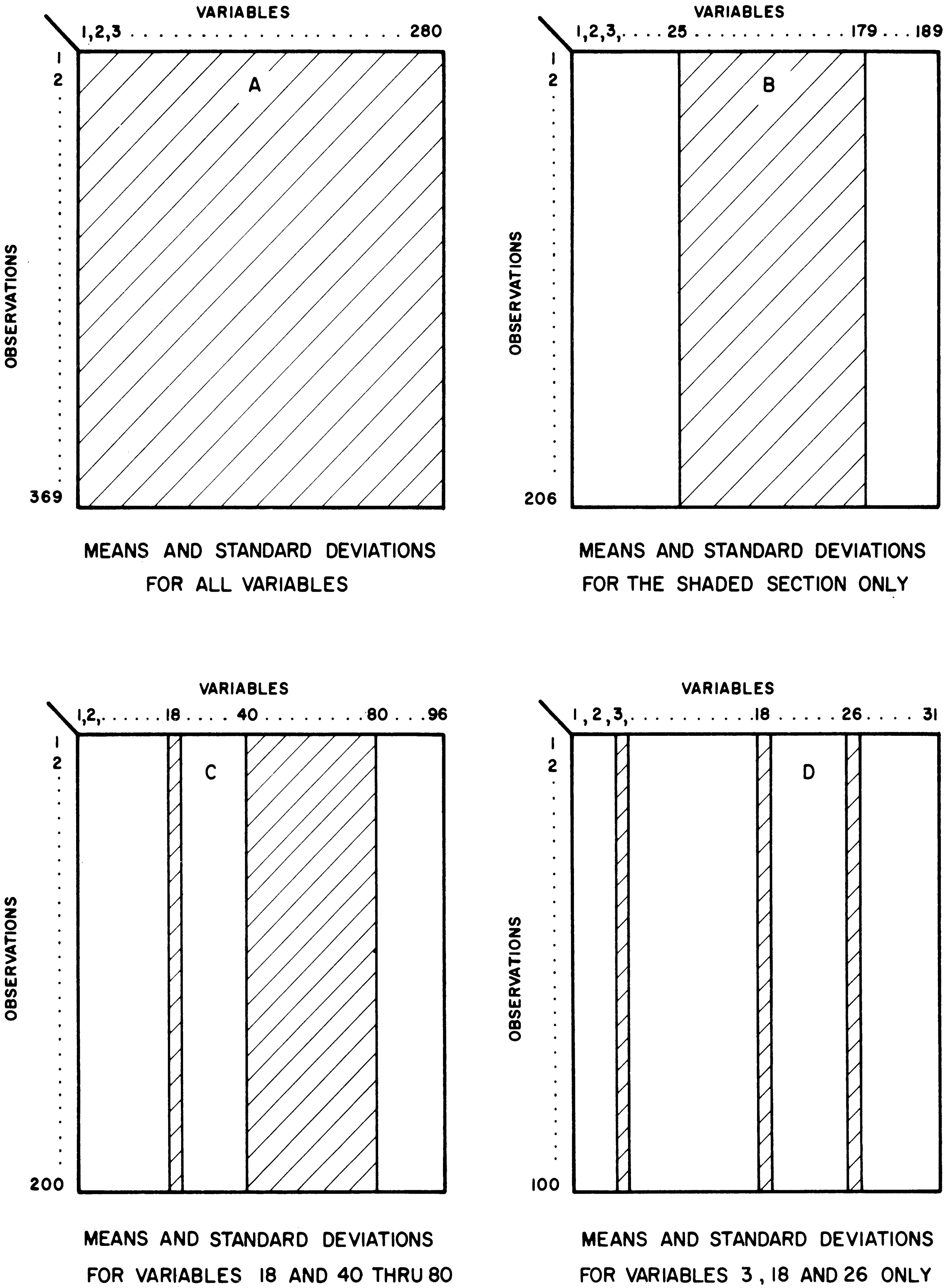

VARIABLES
1,2,3 280
OBSERVATIONS
1
2
369
A
MEANS AND STANDARD DEVIATIONS
FOR ALL VARIABLES
VARIABLES
1,2,3, ... 25 . . . . 179 . . . 189
OBSERVATIONS
1
2
206
B
MEANS AND STANDARD DEVIATIONS
FOR THE SHADED SECTION ONLY
VARIABLES
1,2, ... 18 . . . . 40 . . . . 80 . . . 96
OBSERVATIONS
1
2
200
C
MEANS AND STANDARD DEVIATIONS
FOR VARIABLES 18 AND 40 THRU 80
VARIABLES
1 , 2 , 3, ... 18 . . . . . 26 31
OBSERVATIONS
1
2
100
D
MEANS AND STANDARD DEVIATIONS
FOR VARIABLES 3 , 18 AND 26 ONLY

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,LY,NTOTAL,D
    2 FORMAT(3I6,F8.3)
      NSTORE=5850
      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=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                               )
      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=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
      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
      READ INPUT TAPE 7,41,IA
   41 FORMAT(I10)
      WRITE OUTPUT TAPE 6,42,IA
   42 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.))
      CB=(COUNT(ID)*SQ(IC))-(SUM(IB)**2)
      IF (CB) 45,45,56
   45 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
*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            MSTD2A0
      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=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+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 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
```

```
      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(I10)
      WRITE OUTPUT TAPE 6,42,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 6,61
   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(1H0I14,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    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
```

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

$$\sigma_X = \sqrt{\left(\frac{N \sum_{i=1}^{N} X_i^2 - \left(\sum_{i=1}^{N} X_i \right)^2}{N(N-1)} \right)}$$

3. 'z' Score

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

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

D. LIMITATIONS

Program Category	MAXIMUM NUMBER OF VARIABLES ON an 8k memory	a 32k memory
Z 1A	1200*	4500*
Z 2A	1200*	4500*

*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)

Example	NUMBER OF VARIABLES	NUMBER OF OBSERVATIONS	OBSERV. NO.	DECK NO.	CARD OUTPUT	PRINT OUT
A	280	369	001	001	yes	no
B	185	206	101	021	no	yes
C	96	200	201	101	yes	no
D	31	100	050	010	no	yes

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

	IBM CARD COLUMNS						
Example	1-6	7-12	13-18	19-24	25-30	31-32	33-40
	N	LY	NTOTAL	NONE	NTWO	CARD	D
A	+00280	+00001	+00369	+00001	+00001	+1	+0009999
B	+00185	+00001	+00206	+00101	+00021	-1	+0009999
C	+00096	+00002	+00200	+00201	+00101	+1	+0009999
D	+00031	+00003	+00100	+00030	+00010	-1	+0009999

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,... 369 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

Example	IBM CARD COLUMNS				
	1-3	4-6	7-9	10-12	13-15...69-72
A	001	Blank			
B	025	Blank			
C	018	040	Blank		
D	003	018	026	Blank	

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

Example	IBM CARD COLUMNS				
	1-3	4-6	7-9	10-12	13-15...69-72
A	280	Blank			
B	179	Blank			
C	018	080	Blank		
D	003	018	026	Blank	

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.

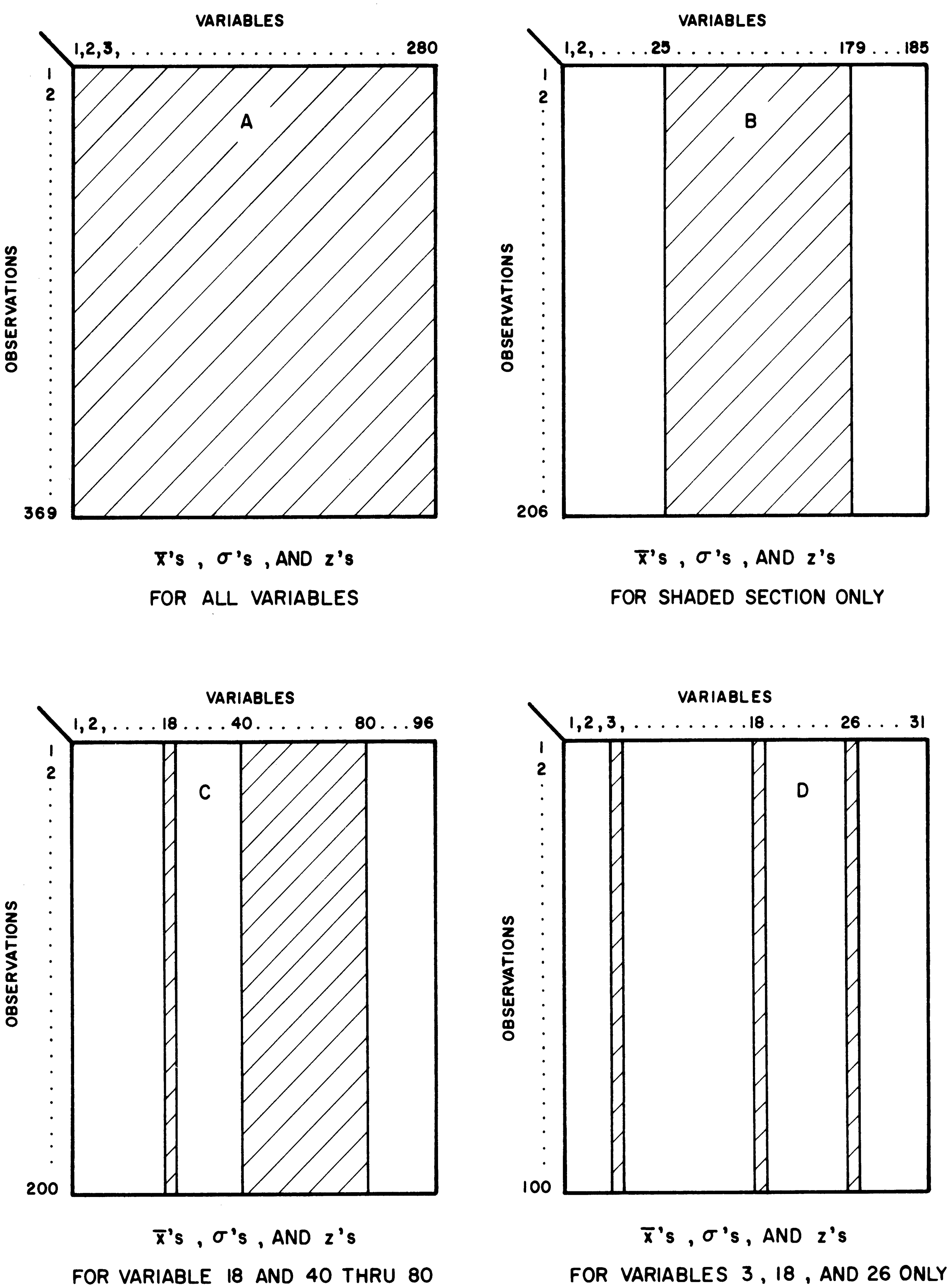

VARIABLES
1,2,3, ... 280
OBSERVATIONS
1
2
369
A
$\overline{X}$'s , σ's , AND z's
FOR ALL VARIABLES
VARIABLES
1,2, ... 25 . . . . 179 . . . 185
OBSERVATIONS
1
2
206
B
$\overline{X}$'s , σ's , AND z's
FOR SHADED SECTION ONLY
VARIABLES
1, 2, ... 18 . . . . 40 . . . . 80 . . . 96
OBSERVATIONS
1
2
200
C
$\overline{X}$'s , σ's , AND z's
FOR VARIABLE 18 AND 40 THRU 80
VARIABLES
1, 2, 3, ... 18 . . . . . . 26 . . . 31
OBSERVATIONS
1
2
100
D
$\overline{X}$'s , σ's, AND z's
FOR VARIABLES 3 , 18 , AND 26 ONLY

FIG. 2

```
*COMPILE FORTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP                    Z1A00000
      DIMENSION SUM(5325),SQ(5325),COUNT(5325),R(5325),NX(5325),NY(5325)
     1,Z(5325)
      EQUIVALENCE (R,SUM),(R,COUNT),(R,SQ),(R,NX),(R,NY),(R,Z)
    1 READ INPUT TAPE 7,2,N,LY,NTOTAL,NONE,NTWO,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 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,(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      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) 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
   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
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
   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(IC)-D) 86,88,86
86 IF (R(IE)-D) 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+(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
*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 , JRTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP             Z2A00000
      DIMENSION SUM(5400),SQ(5400),R(5400),NX(5400),NY(5400),Z(5400)
      EQUIVALENCE (R,SUM),(R,SQ),(R,NX),(R,NY),(R,Z)
    1 READ INPUT TAPE 7,2,N,LY,NTOTAL,NONE,NTWO,CARDS
    2 FORMAT(5I6,F2.0)
      COUNT=NTOTAL
      NSTORE=5400
      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
      IE=IC+N1
      IF=NSTORE-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=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=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
      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
      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   )
      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,L,SQ(IC),SUM(IB),COUNT
   66 FORMAT(1H0I14,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,LG
      IA=LY
```

```
      IF (LG-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 (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
*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:

't' VALUES ON EACH VARIABLE FOR GROUPS				
1-2*				
1-3	2-3			
1-4	2-4	3-4		
1-5	2-5	3-5	4-5	
1-6	2-6	3-6	4-6	5-6
-	-	-	-	-
1-k	2-k	3-k	4-k	5-k...(k-1)-k

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

*1-2 means group 1 versus group 2.

C. DISCUSSION

1. Mean

$$\bar{X}_j = \frac{\sum_{i=1}^{N} X_{ij}}{N}$$

j = 1,2,...k groups

N = number of observations in the group

2. Standard deviation

$$\sigma_j = \sqrt{\left(\frac{N \sum_{i=1}^{N} X_{ij}^2 - (\sum X_{ij})^2}{N(N-1)}\right)}$$

3. Student's t (uncorrelated groups)

$$t = \frac{\bar{X}_i - \bar{X}_j}{\sqrt{\left(\frac{\sigma_i^2}{N_i} + \frac{\sigma_j^2}{N_j}\right)}} \qquad i,j = \text{any two groups}$$

D. LIMITATIONS

Program Category	MAXIMUM NUMBER OF VARIABLES	
	8k memory	32k memory
M 1A	800*	3000*
M 2A	800*	3000*

* 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, $\bar{X}_i$, σ_i, N_j, $\bar{X}_j$, σ_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)

Example	NUMBER OF OBSERVATIONS IN THE GROUPS*					NUMBER OF VARIABLES
	1	2	3	4	5	
A	50	100	25	80	68	280
B	80	90	100	25	40	185
C	60	75	52	68	32	96
D	25	23	32	40	28	31

* 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 follwed 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

Example	IBM CARD COLUMN			
	N	LY	LX	D
	1-6	7-12	13-18	19-26
A	+00280	+00001	+00005	+0009999
B	+00185	+00001	+00005	+0009999
C	+00096	+00002	+00005	+0009999
D	+00031	+00003	+00005	+0009999

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

Example	IBM CARD COLUMN			
	1-3	4-6	7-9	10-12...69-72
A	001			
B	025			
C	018	040		
D	003	018	026	

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

Example	IBM CARD COLUMN			
	1-3	4-6	7-9	10-12...69-72
A	280			
B	179			
C	018	080		
D	003	018	026	

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

Example	IBM CARD COLUMN					
	1-3	4-6	7-9	10-12	13-15	16-18...69-72
A	050	100	025	080	068	
B	080	090	100	025	040	
C	060	075	052	048	032	
D	025	023	032	040	028	

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

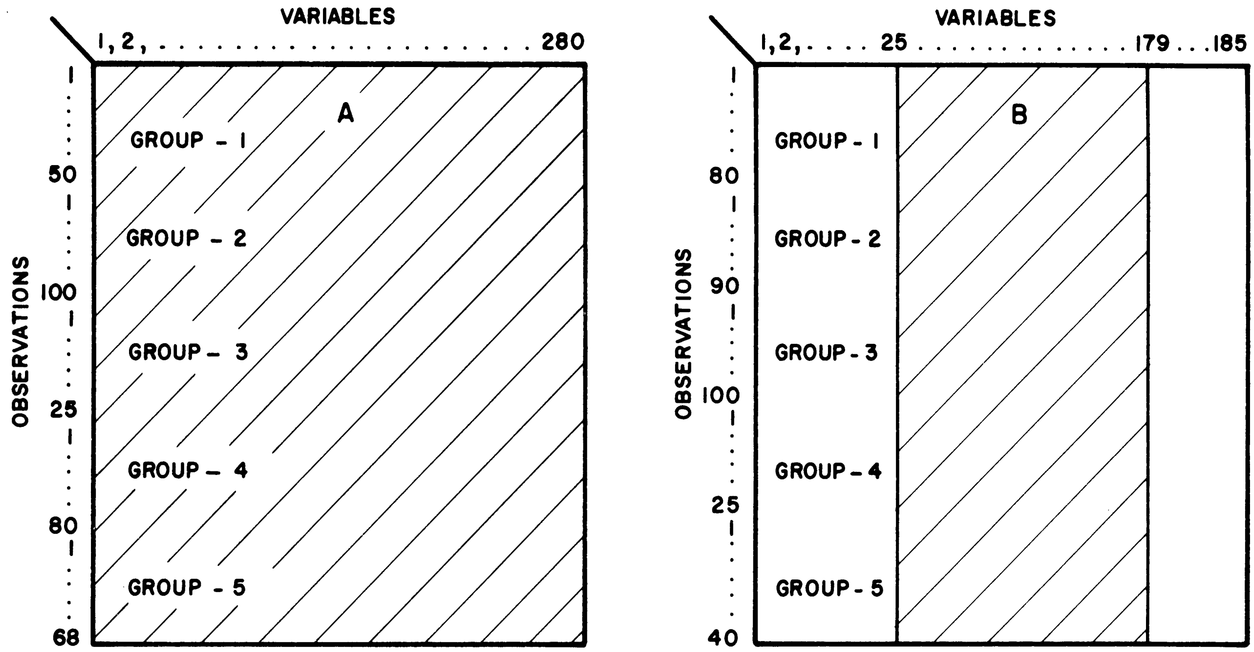
See Appendix A under means and standard-deviations program.

7. DATA CARDS

See Appendix A under means and standard-deviations program.

8. JOB NUMBER CARD

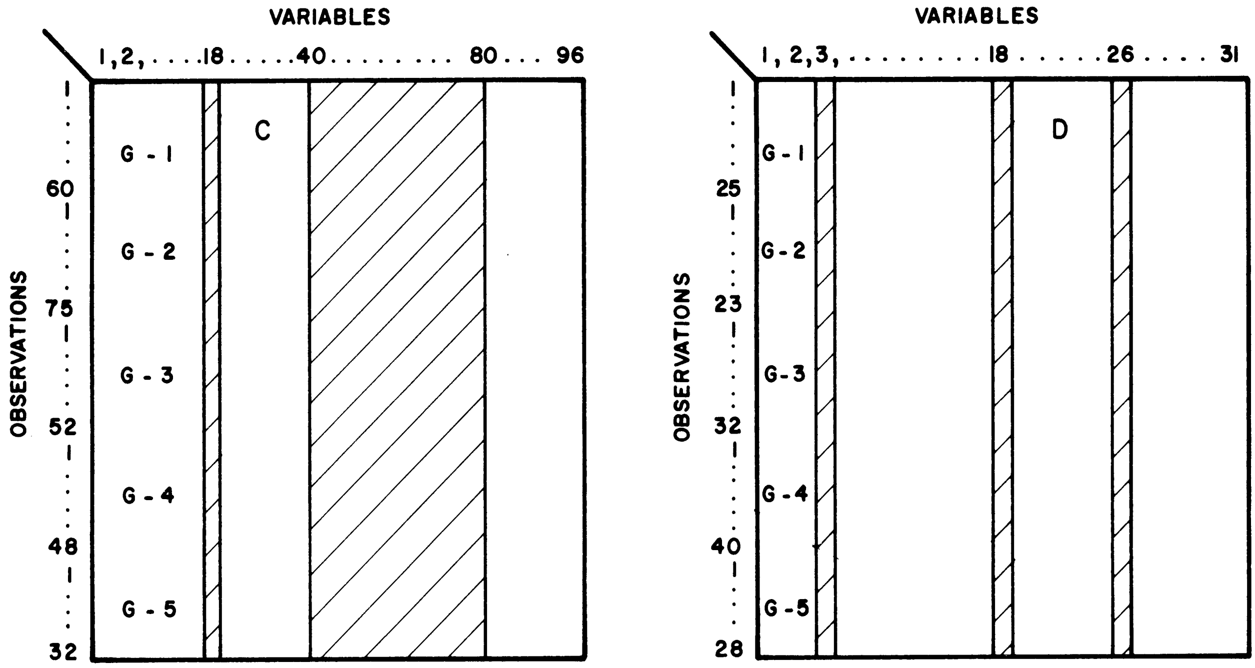
See Appendix A under means and standard-deviations program.



COMPUTE t FOR ALL VARIABLES

COMPUTE t FOR SHADED SECTION ONLY

THE DATA CARDS FOR GROUPS 1, 2, etc. FOLLOW EACH OTHER WITHOUT ANY IN BETWEEN CARDS TO INDICATE BREAKS.



COMPUTE t FOR VARIABLE 18 AND VARIABLES 40 THRU 80

COMPUTE t FOR VARIABLES 3, 18, AND 26 ONLY

FIG. 3

```
*COMPILE FORTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP              M1A20022
      DIMENSION A(200,22),SUM(200),SQUARE(200),COUNT(200),NX(25),NY(25),
     1XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),LC(200),MLAST(200)
      EQUIVALENCE (DEVNX,SQUARE),(DEVNY,LC),(MLAST,YMEAN)
    1 READ INPUT TAPE 7,2,N,LY,LX,D
    2 FORMAT(3I6,F8.3)
      REWIND 3
      REWIND 4
      MFIRST=22
      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
      CALL INPUT
      READ INPUT TAPE 7,8,(A(I,K),I=1,N)
    8 FORMAT(
     1                                     )
      K1=2
      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 (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 7,33,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/
     1SUM(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),
     1SUM(I),YMEAN(I),DEVNY(I),TEE,DF
   55 FORMAT(1H0I5,I4,I6,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
        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              M2A1204
      DIMENSION A(120,40),SUM(120),SQUARE(120),COUNT(120),NX(25),NY(25),
     1XMEAN(120),DEVNY(120),YMEAN(120),DEVNX(120),LC(120),MLAST(120)
      EQUIVALENCE (DEVNY,LC),(MLAST,YMEAN),(SUM,XMEAN),(SQUARE,DEVNX)
    1 READ INPUT TAPE 7,2,N,LY,LX
    2 FORMAT(3I6)
      REWIND 3
      REWIND 4
      MFIRST=40
      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
      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
   16 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,((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   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)),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(1H0I5,I4,I6,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
        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: $t = \sqrt{F}$.)

GROUP	F-RATIO VALUES ON EACH VARIABLE			
1	1-2*			
2	1-3	2-3		
3	1-4	2-4	3-4	
4	1-5	2-5	3-5	4-5
-	-	-	-	-
k	1-k	2-k	3-k	4-k...(k-1)-k

*1-2 means group 1 versus group 2.

C. DISCUSSION

k = Number of groups

N_k = Number of observations in a group

$N_{tot} = \Sigma N_k$

$A = (\Sigma N_k \bar{X}_k{}^2) - (N_{tot} \bar{X}^2{}_{tot})$ = SS between groups

$B = (\Sigma X^2{}_{tot}) - (N_{tot} \bar{X}^2{}_{tot})$ = SS total

$C = B - A$ = SS within groups

$$MS_{bg} = \frac{A}{k-1}$$

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

$$F = \frac{MS_{bg}}{MS_{wg}} \qquad MS = \text{Mean Square (variance estimate)}$$

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

Program Category	MAXIMUM NUMBER OF VARIABLES 8k memory	32k memory
A 1A	500*	2000*
A 1B	500*	2000*
A 2A	500*	2000*
A 2B	500*	2000*

* 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 189 $\binom{5}{2}$ = 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; df2; F; $\bar{X}_i$; σ_i; $\bar{X}_j$; σ_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)

Example	NUMBER OF OBSERVATIONS IN THE GROUPS					NUMBER OF VARIABLES
	1	2	3	4	5	
A	50	100	25	80	68	280
B	80	90	100	25	40	185
C	60	75	52	68	32	96
D	25	23	32	40	28	31

3. 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

Example	IBM CARD COLUMN			
	1-6	7-12	13-18	19-26
	N	LY	LX	D
A	+00280	+00001	+00005	+0009999
B	+00185	+00001	+00005	+0009999
C	+00096	+00002	+00005	+0009999
D	+00031	+00003	+00005	+0009999

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.

b. Control card no. 2

Example	IBM CARD COLUMN			
	1-3	4-6	7-9	10-12...69-72
A	001			
B	025			
C	018	040		
D	003	018	026	

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 3 in control card no. 1. Correspondingly three starting variables are indicated on control card no. 2.

c. Control card no. 3

Example	IBM CARD COLUMN			
	1-3	4-6	7-9	10-12...69-72
A	280			
B	179			
C	018	080		
D	003	018	026	

Note: 1. See Fig. 3, 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

Example	IBM CARD COLUMN					
	1-3	4-6	7-9	10-12	13-15	16-18...69-72
A	050	100	025	080	068	
B	080	090	100	025	040	
C	060	075	052	048	032	
D	025	023	032	040	028	

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                    A1A2000
      DIMENSION A(200,17),SUM(200),SQUARE(200),COUNT(200),TOTAL(200),
     1XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),SOM(200),SQERE(200),
     1LC(200),MLAST(200),NX(25),NY(25)
      EQUIVALENCE (MLAST,YMEAN),(LC,DEVNY)
    1 READ INPUT TAPE 7,2,N,LY,LX,D
    2 FORMAT(3I6,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 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)
      WRITE TAPE 4,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIR
     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    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=NFIRS
     1T,NLAST)
   35 CONTINUE
      DO 65 K=1,LX
      DO 36 L=1,LY
      NFIRST=NX(L)
      NLAST=NY(L)
      READ TAPE 4,((SOM(I),SQERE(I),TOTAL(I),YMEAN(I),DEVNY(I)),I=NFIRST
     1,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(1H0I9,I8,I4,43H    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
      SKIP=0.0
   45 WRITE OUTPUT TAPE 6,47,I,J,K,YMEAN(I),DEVNY(I)
   47 FORMAT(1H0I9,I8,I4,76H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
     1S                              2(F10.3))
      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,I,J,K,XMEAN(I),DEVNX(I)
   53 FORMAT(1H0I9,I8,I4,52H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
     1S        F14.3,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)+SQERE(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(1H0I9,I8,I4,F13.3,F14.3,4F5.0,F9.3,F11.3,3F10.3)
   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
 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,EXEDUTE,DUMP                A2A2000
      DIMENSION A(200,19),SUM(200),SQUARE(200),COUNT(200),NX(25),NY(25),
     1XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),LC(200),MLAST(200),
     1SOM(200),SQERE(200)
      EQUIVALENCE (MLAST,YMEAN),(LC,DEVNY)
    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
      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,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
      WRITE TAPE 4,((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(38H0  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,((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,((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,I,J,K
   42 FORMAT(1H0I9,I8,I4,43H    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 WR+TE OUTPUT TAPE 6,47,I,J,K,YMEAN(I),DEVNY(I)
   47 FORMAT(1H0I9,I8,I4,76H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
     1S                          2(F10.3))
      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,I,J,K,XMEAN(I),DEVNX(I)
   53 FORMAT(1H0I9,I8,I4,52H     PLEASE CHECK YOUR DATA FOR THE TWO GROUP
     1S        F14.3,F10.3)
      GO TO 60
   54 TOP1=((SUM(I)**2)/COUNT(J))+((SOM(I)**2)/COUNT(K))
      DF2=COUNT(J)+COUNT(K)
      TOP2=((SUM(I)+SOM(I))**2)/(DF2)
      XBAR=TOP1-TOP2
      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
   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(J),COUNT(K),DF1,DF2,Z3,
     1XMEAN(I),DEVNX(I),YMEAN(I),DEVNY(I)
   59 FORMAT(1H0I9,I8,I4,F13.3,F14.3,4F5.0,F9.3,F11.3,3F10.3)
   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
 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              A1B2000
      DIMENSION A(200,17);SUM(200);SQUARE(200);COUNT(200);TOTAL(200);
     1XMEAN(200),DEVNY(200);YMEAN(200);DEVNX(200);SOM(200);SQERE(200),
     1LC(200),MLAST(200);NX(25);NY(25)
      EQUIVALENCE (MLAST,YMEAN);(LC;DEVNY)
    1 READ INPUT TAPE 7;2;N;LY;LX;D
    2 FORMAT(3I6,F8.3)
      REWIND 3
      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 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     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=NFIRS
     1T,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(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)
   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
        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               A2B2000
      DIMENSION A(200,19),SUM(200),SQUARE(200),COUNT(200),NX(25),NY(25),
     1XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),LC(200),MLAST(200),
     1SOM(200),SQERE(200)
      EQUIVALENCE (MLAST,YMEAN),(LC,DEVNY)
    1 READ INPUT TAPE 7,2,N,LY,LX
    2 FORMAT(3I6)
      REWIND 3
      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)
      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)
```

```
      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(38H0  ANALYSIS OF VARIANCE   JOB NUMBER  I10  )
      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
      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
```

```
      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)
   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
       TSX ERROR,4
       TXL CPY,1,12
       LXD AX1,1
       TRA 1,4
 A=1   PZE 1
 AX1   BSS 1
 END   SYN *
       END
*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. $\geq$ 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,

$$L_{xy} = N\Sigma XY - \Sigma X \Sigma Y$$

$$L_{xx} = N\Sigma X^2 - (\Sigma X)^2$$

$$L_{yy} = N\Sigma Y^2 - (\Sigma Y)^2$$

In the case of the missing data program N, ΣXY, ΣX, ΣY, ΣX^2, ΣY^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

Program Category	MAXIMUM NUMBER OF VARIABLES	
	8k memory	32k memory
R 1A	797*	3000*
R 2A	1100*	4200*

* 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.

Program Category	NUMBER OF VARIABLES	NUMBER OF OBSERVATIONS	RUNNING TIME
R 1A	120	200	30 min.
R 2A	130	300	17 min.

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 i

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

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

In example B, i = 25,26,...178, j = i + 1, ..., 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 cards)

2. Card to instruct the computer to compile Fortran. Usually punched starting in column 1: *COMPILE FORTRAN, PRINT SAP, PUNCH (1 card)

OBJECT, EXECUTE, DUMP

3. Fortran program deck (with any change in dimension cards and MFIRST = *** or ** card, if required). (1 deck)

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

5. Sub-routine deck (1 deck)

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

7. Control card (1 card)**

8. Format statement. See Appendix A. (1 card)

9. Test data cards sorted in matrix form. (n decks)

10. Job number card. See Appendix A. (1 card)

b. Using binary deck (Fig. 4)

1. I-D cards (2 cards)

2. Card punched (starting in column 1:) *EXECUTE, DUMP (1 card)

3. Binary program deck (1 deck)

4. Binary sub-routine cards (2 cards)

5. *DATA (1 card)

6. Control card (1 card)**

7. Format statement. See Appendix A. (1 card)

8. Data cards sorted in matrix form (n decks)

9. Job number card. See Appendix A. (1 card)

2. EXAMPLE PROBLEMS

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

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

Example	NUMBER OF VARIABLES	NUMBER OF OBSERVATIONS	't'-VALUE	CARD OUTPUT
A	280	369	yes	yes
B	185	206	no	yes
C	96	200	yes	no
D	31	100	no	no

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, $\geq$ 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)

DIMENSION SUM (300), SOM (300), SQUARE (300), SQERE (300), PROD (300)

Card 2:

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

Note: 300+300+300+300+300+(300 x 12) + 300 $\leq$ 5580

Card 1: (Example B)

DIMENSION 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 $\leq$ 5580

Card 1: (Example C)

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

Card 2:

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

Note: 100+100+100+100+100+(100 x 49) + 100 $\leq$ 5580

Card 1: (Example D)

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

Card 2:

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

Note: 50+50+50+50+50+(50 x 105) + 50 $\leq$ 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), (sqere 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 369/12 = 30 x 12 + 1 x 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 x 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:

MFIRST = 12 (A)

MFIRST = 21 (B)

MFIRST = 49 (C)

MFIRST = 105 (D)

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

Example	IBM CARD COLUMNS							
	1-6	7-12	13-18	19-24	25-30	31-34	35-36	37-44
	N	NFIRST	NLAST	NEND	COUNT	't' VALUE	CARD	D
A	+00280	+00001	+00279	+00280	+00369	+196	+1	+0009999
B	+00185	+00025	+00178	+00179	+00206	-196	+1	+0009999
C	+00096	+00001	+00040	+00096	+00200	+196	-1	+0009999
D	+00031	+00005	+00025	+00031	+00100	-196	-1	+0009999

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 $\geq$ 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.

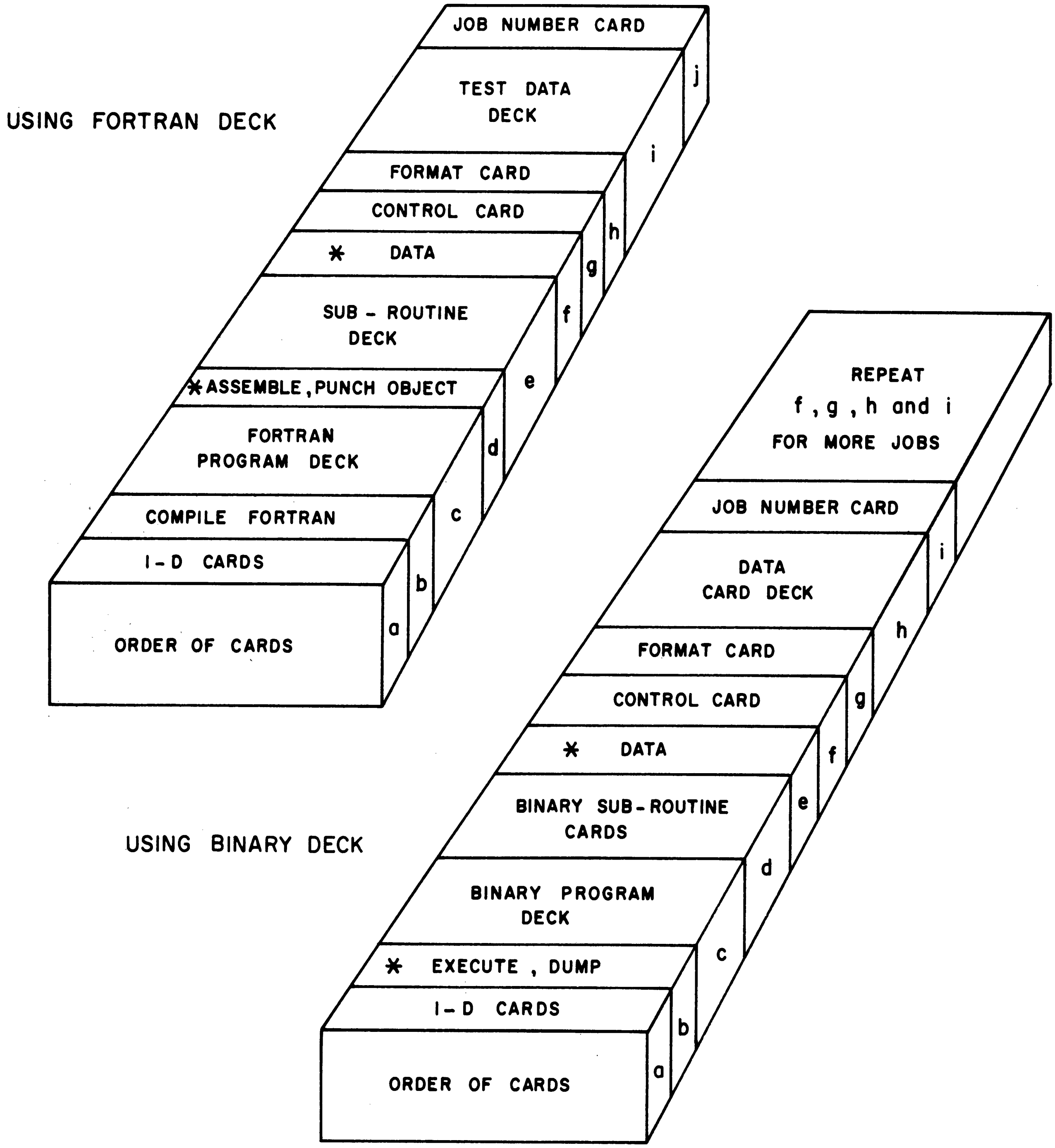
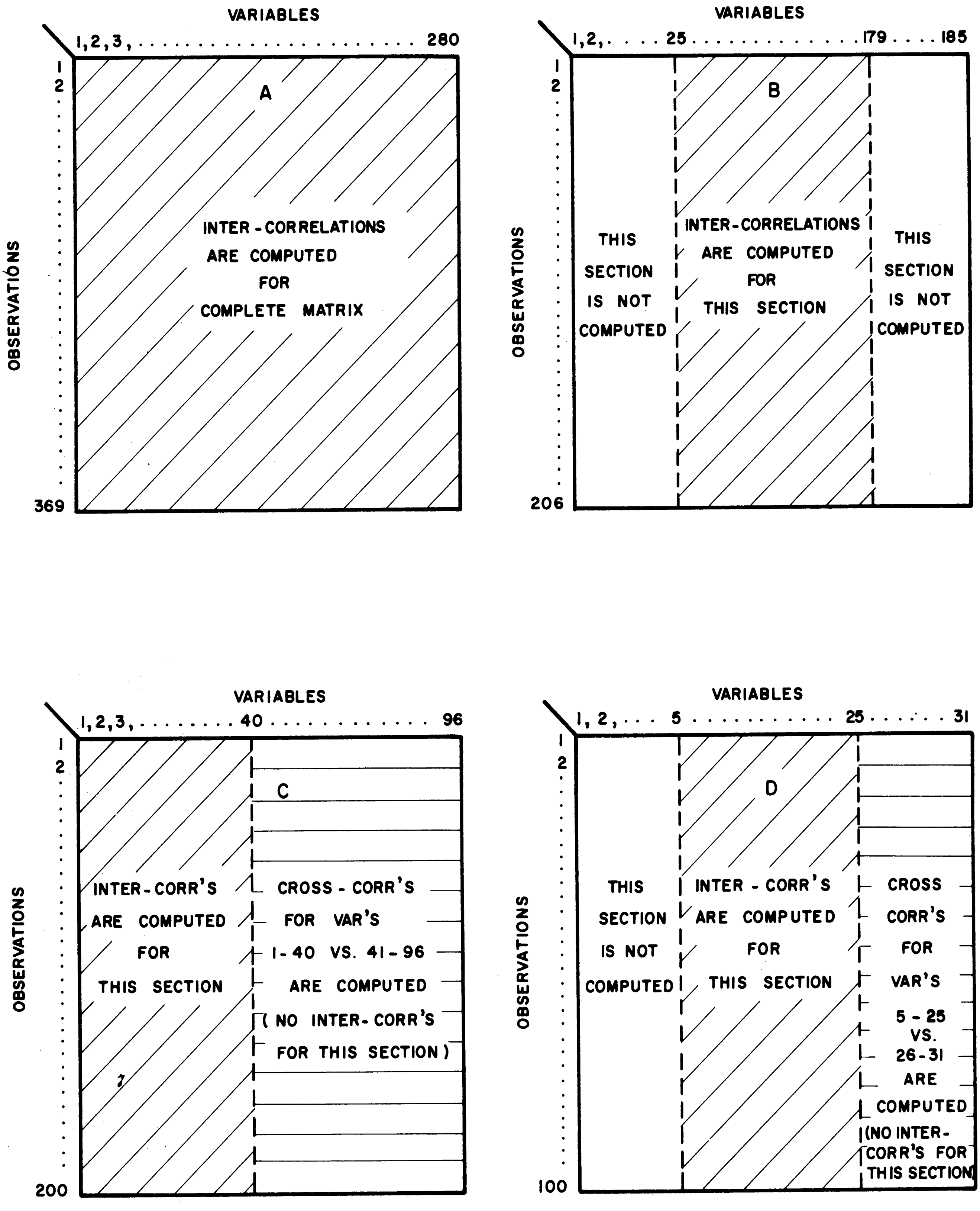


FIG. 4

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 I-D CARDS.



EXAMPLES : R 1A , R2A

INPUT MATRICES A , B , C , AND D

FIG. 5

```
*COMPILE FORTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP              R1A1553
      DIMENSION SUM(155),SOM(155),SQUARE(155),SQERE(155),PROD(155),
     1A(155,30),TOTAL(155),JA(155,30)
    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=30
      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=(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=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=I,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),
     1SQUARE(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
```

```
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),
  1SQUARE(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(I10)
   WRITE OUTPUT TAPE 6,53,NEW
53 FORMAT(38H1  CORRELATION  MATRIX    JOB NUMBER  I10   )
   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),
  1SQUARE(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))-(SOM(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) 71,71,72
71 SUM(L)=8.888
   GO TO 80
72 SUM(L)=(PROD(L)/(SQRT(1.-CC)))*(SQRT(TOTAL(L)))
80 CONTINUE
   WRITE OUTPUT TAPE 6,6,I,((JA(L,1),PROD(L),TOTAL(L),SUM(L)),L=L1,
  1NEND)
 6 FORMAT(1H0I7/(3(1H I10,F9.3,F9.0,F9.3)))
   IF (CARD) 83,90,81
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
   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
*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            R2A15033
      DIMENSION SUM(150),R(150,33),SIGMA(150),PROD(150),JA(150,33),
     1TEE(150)
      EQUIVALENCE (JA,R)
    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)
```

```
36 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 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  )
   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
   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
60 PROD(L)=9.999
   TEE(L)=9.999
   GO TO 80
65 PROD(L)=TOP/BELOW
   CC=PROD(L)**2
   IF (1.-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 6,6,I,DF,((JA(L,1),PROD(L),TEE(L)),L=L1,NEND)
 6 FORMAT(1H0I7,F8.0/(4(1H I8,F10.3,F10.4)))
   IF (CARD) 83,90,81
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(TEE(M))
   IF (CC-TVALUE) 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 6,6,I,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
```

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

Program Category	Maximum Number of Variables on a 8k memory	32k memory
R 1B	797*	3000*
R 2B	1100*	4200*

*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.

Program Category	NUMBER OF VARIABLES	NUMBER OF OBSERVATIONS	RUNNING TIME
R 1B	70 vs 115	200	35 min.
R 2B	70 vs 120	300	17 min.

Computation time for R 1B is .1 to .2 seconds per r, depending on the size of the matrix. The R 2B program is 3 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)

Example	Number of Variables		Total	Number of Observations	't' value	Card Output
	Sec-1	Sec-2				
A	80	200	280	250	yes	yes
B	60	120	180	350	yes	no
C	40	60	100	600	no	yes
D	20	30	50	95	no	no

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

Example	IBM CARD COLUMNS							
	1-6	7-12	13-18	19-24	25-30	31-34	35-36	37-44
	N	N BEGIN	N END	LY	COUNT	t VALUE	CARD	D
A	+00280	+00001	+00080	+00001	+00250	+196	+1	+0009999
B	+00173	+00005	+00050	+00001	+00350	+196	-1	+0009999
C	+00100	+00001	+00100	+00003	+00600	-196	+1	+0009999
D	+00050	+00005	+00050	+00002	+00095	-196	-1	+0009999

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

Example	IBM CARD COLUMNS			
	1-3	4-6	7-9	10-12...69-72
A	081			
B	061			
C	025	050	099	
D	018	026		

c. Control card no. 3

Example	IBM CARD COLUMNS			
	1-3	4-6	7-9	10-12...69-72
A	280			
B	173			
C	025	050	099	
D	018	026		

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 (±xxxxx, 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 stated = 173. Total count could also be stated = 180 (the full matrix). Control card 3 will anyway stop calculations beyond 173.

6. FORMAT STATEMENT

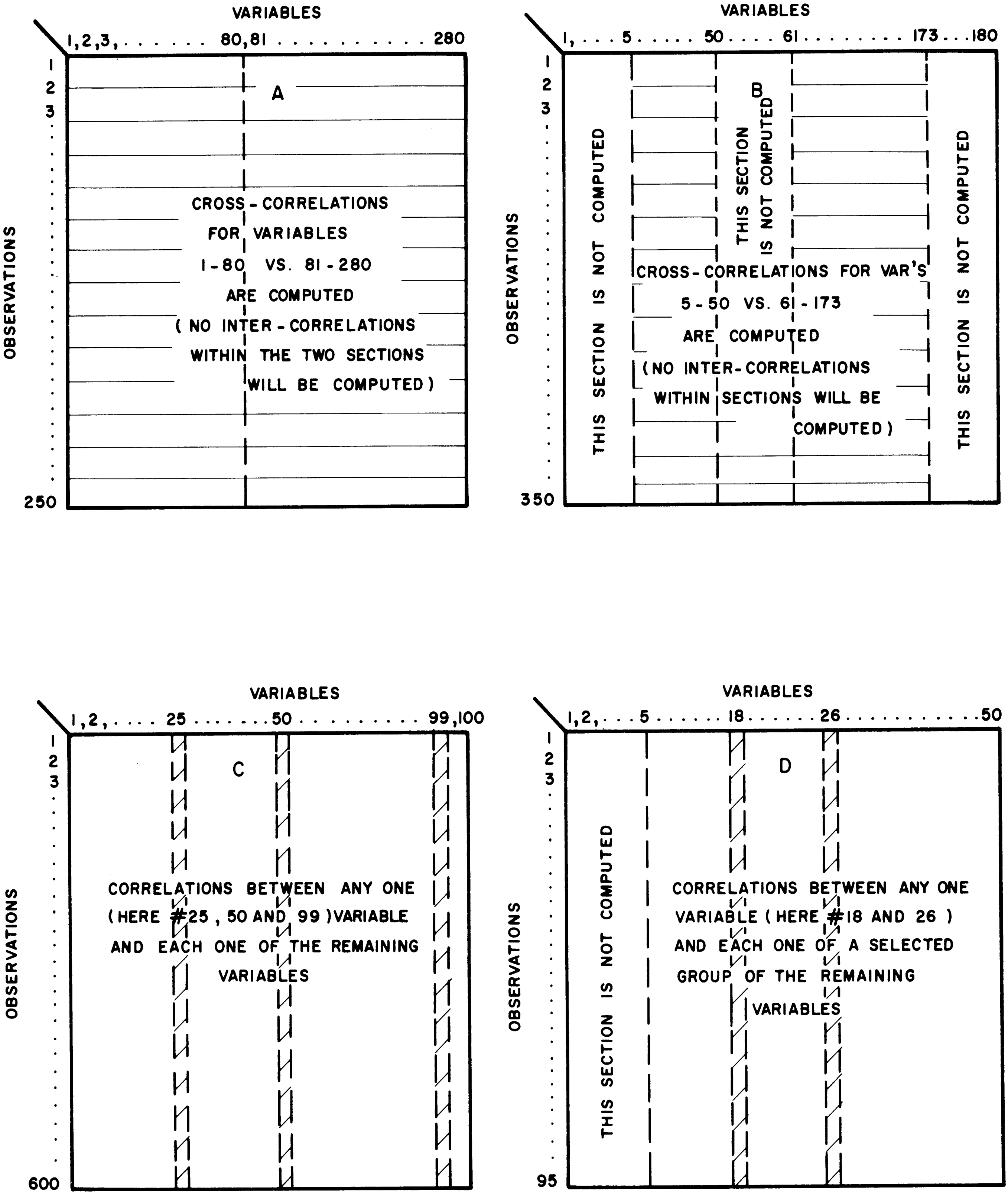
See Appendix A under means and standard-deviations program.

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.



EXAMPLES : R 1 B , R 2 B

INPUT MATRICES A , B , C , AND D

FIG. 6

```
*COMPILE FORTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP              R1B2500
      DIMENSION SUM(200),SOM(200),SQUARE(200),SQERE(200),PROD(200),
     1A(250,17),TOTAL(200),JA(250,17),NX(25),NY(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,LY)
    3 FORMAT(24I3)
      READ INPUT TAPE 7,3,(NY(I),I=1,LY)
      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
```

```
      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)
   36 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),
     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(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
      DO 90 LE=1,LY
      NFIRST=NX(LE)
      NLAST=NY(LE)
      DO 90 I=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))-(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))-(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,I,((JA(M,1),PROD(M),TOTAL(M),SUM(M)),M=1,NA)
    6 FORMAT(1H0I7/(3(1H I10,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,I,((JA(M,2),PROD(M),TOTAL(M),SUM(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
```

```
*COMPILE FORTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP           R2B15032
      DIMENSION SUM(150),R(150,32),SIGMA(150),PROD(150),JA(150,32),
     1TEE(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,(NX(I),I=1,LY)
    3 FORMAT(24I3)
      READ INPUT TAPE 7,3,(NY(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
      ASSIGN 36 TO JUMP
      NA=NEND-NBEGIN+1
      K=NBEGIN-1
      DO 50 LA=1,LG
      IF (LG-LA) 11,11,13
   11 M1=MLAST
   13 DO 14 K=K1,M1
      READ INPUT TAPE 7,4,(R(I,K),I=1,N)
   14 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
      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
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,M1
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=NFIRST,NEND
   CB=(COUNT*SIGMA(I))-(SUM(I)**2)
   SIGMA(I)=SQRT(CB/CA)
55 CONTINUE
   DO 90 LE=1,LY
   NFIRST=NX(LE)
   NLAST=NY(LE)
   DO 90 I=NFIRST,NLAST
   M=0
   READ TAPE JTAPE,(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.-CC) 71,71,72
71 TEE(M)=8.888
   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(1H0I7,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
      CC=ABSF(TEE(M))
      IF (CC-TVALUE) 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,((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 G)

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

EXAMPLE	NUMBER OF VARIABLES	DIAGONAL ELEMENT	INVERS	CARDS
A	280	1.0	no	yes
B	180	Max	no	yes
C	146	1.0	yes	no
D	96	1.0	no	yes
E	31	Max	no	yes

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

2. INVERS-yes calls for card out-put for the inversion program.

   CARDS-yes calls for card out-put for the factor analysis program.

3. The card out-put 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 ≤ 72 (using an 8k memory machine; for a 32k memory the corresponding numbers are >320 and ≤ 160). Otherwise, when these limits are:

8k memory, ≤ 144 and ≤ 72

32k memory, ≤ 320 and ≤ 160

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

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

3. 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

Example	IBM CARD COLUMNS				
	1-6	7-12	13-14	15-16	17-18
	N	NA	DIAG	INVERS	CARDS
A	+00280	+00280	+1	-1	+1
B	+00180	+00180	-1	-1	+1
C	+00072	+00146	+1	+1	-1
D	+00094	+00096	+1	-1	+1
E	+00027	+00031	-1	-1	+1

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.

CONTROL CARD NO.	VARIABLES NO.	1-3	4-6	7-9	10-12...52-54...61-63...70-72
2	1 thru 24	Blank......................999 ....Blank			
3	25 thru 48	Blank..			
4	49 thru 72	Blank..			
5	73 thru 96	Blank..............................999 ...Blank			

In each card 72 columns are used. Variable no. 93, in the 3-digit format, ends in column 3 x 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.

CONTROL CARD NO.	VARIABLE NO.	IBM CARD COLUMN 1-3 ...19-21...37-39...70-72
2	1 thru 24	Blank... 999 ... 999 ... 999
3	25 thru 31	Blank... 999 .... Blank

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).

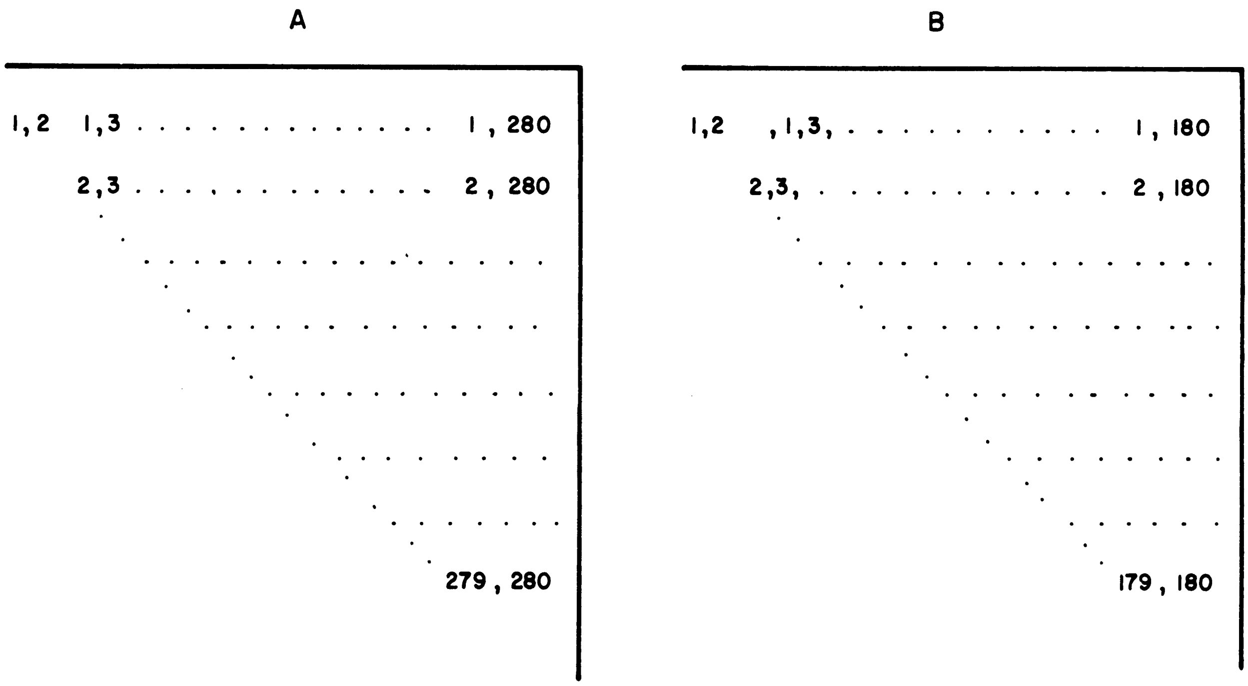
For example, a matrix of 34 x 34 should be punched the following way:

CARD NO.	IBM CARD COLUMNS 1-6	7-12	55-60 67-72	Row no. Card no. 73-80
1.	r_{1-2}*	r_{1-3}	r_{1-13}	00010001
2.	r_{1-14}	r_{1-15}	r_{1-25}	00010002
3	r_{1-26}	r_{1-27} .. r_{1-34}	Blank -------	00010003
4	r_{2-3}	r_{2-4}	r_{2-14}	00020001
5	r_{2-15}	r_{2-16}	r_{2-26}	00020002
..	.....	....		....
63	r_{33-34}	Blank ----------------------		00330001

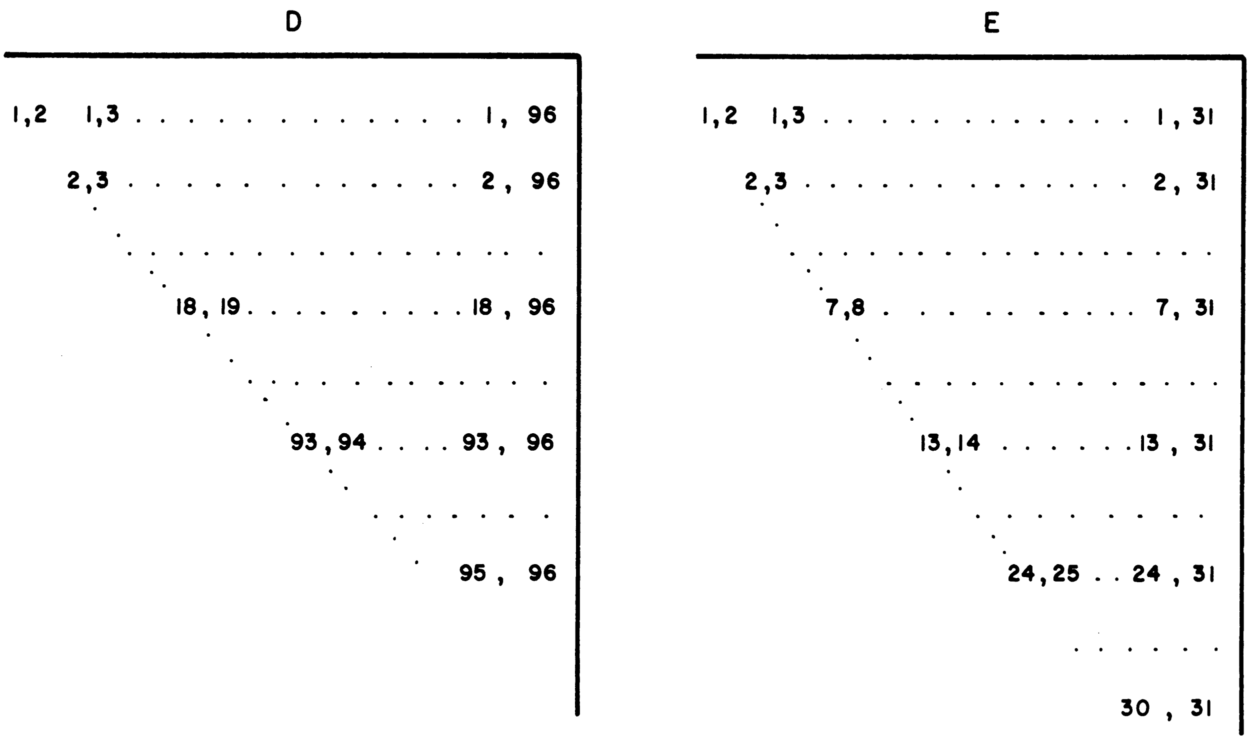
*r_{1-2} means the correlation for variable 1 with 2.

8. JOB NUMBER CARD

See Appendix A, means and standard-deviations program.



EXAMPLE C FOLLOWS THE PROCEDURE OF D AND E .



THE ABOVE MATRICES INDICATE THE ARRANGEMENT OF INPUT CARDS .

THE UPPER HALF OFF-DIAGONAL ELEMENTS ARE USED .

FIG. 7

```
*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(2I6,F2.0,I2,F2.0)
      READ INPUT TAPE 7,3,(JA(I),I=1,NA)
    3 FORMAT(24I3)
      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(I10)
      WRITE OUTPUT TAPE 6,20,NEW
   20 FORMAT(40H1       COMPLETED  CORRELATION  MATRIX   I10  )
      END FILE 2
      REWIND 2
      ASSIGN 32 TO NEW
      LG=N/MFIRST
```

```
      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
```

```
   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,LG
      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(12F6.3)
  122 WRITE OUTPUT TAPE 6,125,KA,(R(I,K),I=1,N)
  125 FORMAT(1H0,I4/(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
```

```
       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
```

VII. INVERSION OF SYMMETRIC MATRICES AND ESTIMATION OF COMMUNALITIES

A. DESCRIPTION OF THE PROGRAM

1. Program print out (See Appendix H).
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...N, where N is the order of the matrix.

a. Compute for temporary use the quantities

$$p_i = a_{in} \times a_{nn}^{-1} \text{ for } i<n \qquad (i = 1,2...N)$$

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

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

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

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

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 $E_{jj} = 1/a^{jj}$ where a^{jj} refers to the diagonal element of the inverted matrix, and E is a diagonal matrix (N x 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} refers to the input correlation matrix with 1'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:

Final Matrix Size	Option	Running Time minutes
18 x 18	R^2	.5
36 x 36	'P'-(Image-matrix)	1.7
50 x 50	'P'-(Image-matrix)	4.1

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}, $R_1R_1^{-1}$, $(R_1 + \text{diag } R_1^2)$

   b. R_1^{-1}, $R_1R_1^{-1}$, P

   c. R_1 with unities or row maxima in the diagonal

d. R_1^{-1}, R_1R^{-1}

2. Card output, without row identification, can be obtained (optional) for (R_1 + 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

Example	R_1	R_o	DIAG. ELEM. FIRST	MATRIX INVER-SION	DIAG. ELEM. FINAL	CARD OUT-PUT
A	70	110	1.0	no	1.0	yes
B	60	60	1.0	yes	R^2	yes
C	50	58	1.0	yes	R^{-1}	no
D	40	50	1.0	yes	P	yes
E	35	40	max	no	max	yes
F	30	30	max	no	max	no

Note: R_1 = Size of matrix after elimination of variables

R_o = 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

	IBM CARD COLUMNS					
Example	1-3	4-6	7-8	9-10	11-12	13-14
	N	NA	DIAG	SMCINV	SMC	CARDS
A	070	110	+1	-1	+0	+1
B	060	060	+1	+1	+1	+1
C	050	058	+1	+1	+0	-1
D	040	040	+1	+1	-1	+1
E	035	040	-1	+0	+0	+1
F	030	030	-1	+0	+0	-1

Note: N = R_1 in example problems.

NA = R_o in example problems.

DIAG: +1 inserts 1'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 out-put 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).

5. 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                 INV1000
      DIMENSION A(72,72),E(72),R(72),JA(144),F(144)
      EQUIVALENCE (E(1),F(1)),(R(1),F(73))
    1 READ INPUT TAPE 7,2,N,NA,DIAG,SMCINV,SMC,CARDS
    2 FORMAT(2I3,4F2.0)
      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
```

```
      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) 54,54,60
   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
  105 READ INPUT TAPE 7,82,I
   82 FORMAT(I10)
      WRITE OUTPUT TAPE 6,83,I
   83 FORMAT(44H1   INVERTED  MATRIX    JOB NUMBER           I10         )
      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(1H0I8/(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,I,(E(K),K=1,N)
  135 FORMAT(1H0I8/(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) 220,220,195
  195 PUNCH 200,(A(I,K),K=1,N)
  200 FORMAT(12F6.3)
  220 WRITE OUTPUT TAPE 6,94,I,(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. 363; 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; 5, p. 363; 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 (1/6) to one-third (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 (σ^2 per variable = 1.0) or from total starting communality (Σh^2_{est}). 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

$$d_{j1} = \sum_{k=1}^{n} r_{jk}\, a_{k1} \qquad (j = 1,2,\ldots n)$$

where

the first estimate for $a_{k1} = (1,1,\ldots,1)$ and n is the order of the matrix R.

2. Find

$$E = \text{Max}\ (d_{j1}) \qquad (j = 1,2,\ldots n)$$

3. Compute

$$a^*_{k1} = \frac{1}{E}(d_{k1}) \qquad (k = 1,2,\ldots n)$$

4. Compare

$$\text{Abs.}\ \left|(a_{k1}) - (a^*_{k1})\right| \leq \text{Epsilon}\ (k = 1,2,\ldots n)$$

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

5. Replace

a_{k1} by a^*_{k1}

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

$$C_k = E \times a_{k1} / \sqrt{\left(\sum_{j=1}^{n} a^2_{j1}\right)} \qquad (k = 1,2,\ldots n)$$

7. Compute the first residual matrix

$$R_1 = R - Q_1$$

where

$$Q_1 = C_1 C_1'$$

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

$$h_k^2 = \sum_{j=1}^{m} C_{kj}^2 \qquad (k = 1,2,\ldots,n)$$

10. Compute sum of latent roots and sum of communalities

$$\sum_{j=1}^{m} E_j \quad \text{and} \quad \sum_{k=1}^{n} h_k^2$$

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 3.5 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 12F6.3

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

Example	MATRIX ORDER	TOTAL VARIANCE	NO. OF FACTORS	LOWER BOUND	FIRST/ LAST ROOT	% VAR. EXTR.	EPSILON
	n	option 4a	option 1	option 2	option 3	option 4b	option 5
A	180	n	20	1.50	25	85	.001
B	100	Σh^2_{est}	15	1.00	20	95	.0001
C	60	n	12	0.50	30	100	.0001
D	30	Σh^2_{est}	7	0.10	20	90	.00001

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

Example	IBM CARD COLUMNS						
	1-6	7-8	9-11	12-15	16-18	19-22	23-28
	N	TOTCOM	NO	CHECK	END	PASENT	EPSLON
A	+00180	+1	+20	+150	+25	+085	+00100
B	+00100	-1	+15	+100	+20	+095	+00010
C	+00060	+1	+12	+050	+30	+100	+00010
D	+00030	-1	+07	+010	+20	+090	+00001

See example problems for explanation.

6. FORMAT CARD

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

```
*COMPILE FORTRAN,PRINT SAP,PUNCH OBJECT,EXECUTE,DUMP              FAC18024
      DIMENSION A(180),C(180),E(40),R(180,24)
    1 READ INPUT TAPE 7,2,N,TOTCOM,NO,CHECK,END,PASENT,EPSLON
    2 FORMAT(I6,F2.0,I3,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) 7,6,7
    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
      K=1
      K1=2
      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 CALL INPUT
      READ INPUT TAPE 7,15,(R(I,K),I=1,N)
   15 FORMAT(
     1                                  )
      DO 45 LA=1,LG
      IF (LG-LA) 25,25,30
   25 M1=MLAST
   30 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) 44,40,44
   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
 55 DO 58 I=1,N
 58 A(I)=1.
    SUM=0.0
 60 DO 65 J=1,N
 65 D(J)=0.0
    MA=0
    M1=MFIRST
    DO 92 LA=1,LG
    IF (LG-LA) 70,70,75
 70 M1=MLAST
 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=ABSF(D(I))
    DO 100 J=2,N
    CMIN=ABSF(D(J))
    IF (CMAX-CMIN) 95,100,100
 95 CMAX=CMIN
    I=J
100 CONTINUE
    DO 105 J=1,N
105 A(J)=D(J)/D(I)
    KB=1
    DO 120 J=1,N
    DIFF=ABSF(A(J)-C(J))
    IF (DIFF-EPSLON) 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 SOM=SOM+E(LC)
    ROOT=SQRT(CMAX)
    SUM=SQRT (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-PASENT) 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,LG
      IF (LG-LA) 170,170,175
170   M1=MLAST
174   GO TO LOAD,(175,176)
175   READ TAPE JTAPE,((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 LOAN,(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(40H0  DESIRED NUMBER OF FACTORS EXTRACTED   )
      GO TO 250
225   WRITE OUTPUT TAPE 6,226
226   FORMAT(55H0  LATENT ROOT LESS OR EQUAL TO FRACTION OF 1ST ROOT    )
      GO TO 250
230   WRITE OUTPUT TAPE 6,231
231   FORMAT(35H0  REQUIRED  PERCENTAGE  ACHIEVED  )
      GO TO 250
235   WRITE OUTPUT TAPE 6,236
236   FORMAT(55H0  LATENT ROOT LESS THAN OR EQUAL TO DESIRED VALUE      )
      GO TO 250
240   WRITE OUTPUT TAPE 6,241
241   FORMAT(30H0  FIRST NEGATIVE LATENT ROOT )
      LC=LC-1
250   END FILE 4
      REWIND 4
      REWIND JTAPE
      REWIND ITAPE
```

```
      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
  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,J,(D(I),I=1,LC)
  285 FORMAT(1H0I8/(1H010F11.3))
  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,I,(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(20H0      LATENT ROOTS  )
      WRITE OUTPUT TAPE 6,320,(E(J),J=1,LC)
  320 FORMAT(1H0(F16.3,12F8.3))
      WRITE OUTPUT TAPE 6,330
  330 FORMAT(30H0      SUM OF LATENT ROOTS       )
      WRITE OUTPUT TAPE 6,335,SOM
  335 FORMAT(1H0F20.3)
      WRITE OUTPUT TAPE 6,340
  340 FORMAT(20H0      COMMUNALITIES )
      WRITE OUTPUT TAPE 6,345,(C(I),I=1,N)
  345 FORMAT (1H010F11.3)
      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
       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
```

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).

4. Convergence.

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

---

*The procedure followed was originally given by Neuhaus, J., and Wrigley, C., The Quartimax method: An analytical approach to orthogonal simple structure, BJ Stat. Psych., 7 (1954), 81-91.

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 $f(b_{jr})$ 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

Notation:		Varimax method	Quartimax method
a_{jp}, a_{jq}	=	loadings of variable z_j on factors p and q ($j = 1,2,\ldots,n$)	Same
h_j^2	=	communality of variable z_j	Same

	Varimax method		Quartimax method
w	= angle of rotation		Same
x_j	= a_{jp} (raw) or a_{jp}/h_j (normalized)		Same as raw
y_j	= a_{jq} (raw) or a_{jq}/h_j (normalized)		Same as raw
X_j, Y_j	= rotated loadings		Same
u_j	= $x_j^2 - y_j^2$	$v_j = 2x_jy_j$	Same
A	= Σu_j	$B = \Sigma v_j$	Not calculated
C	= $\Sigma(u_j^2 - v_j^2)$	$D = 2\Sigma u_jv_j$	Same

The angle of rotation is given by

$$\tan 4w = \frac{nD-2AB}{nC-(A^2-B^2)} \qquad \tan 4w = \frac{D}{C}$$

$$(X_j \; Y_j) = (x_j \; y_j) \begin{pmatrix} \cos w & -\sin w \\ \sin w & \cos w \end{pmatrix}$$

X_j and Y_j are final for the Raw Varimax and the Quartimax.

h_jX_j and h_jY_j (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:

$$V_N = n \sum_{p=1}^{m} \sum_{j=1}^{n} (b_{jp}/h_j)^4 - \sum_{p=1}^{m} \sum_{j=1}^{n} b_{jp}^2/h_j^2 {}^2$$

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)*:

$$Q = \sum_{j=1}^{n} \sum_{p=1}^{m} b_{jp}^{4}$$

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)

Memory	Tests		Factors	Tests		Factors	Tests		Factors
8k	150	x	32	200	x	24	300	x	15
32k	300	x	75	500	x	50	700	x	35

*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 (1,2,...,m) corresponding to test z_j horizontally, with row identification (1,2,...,n)
3. Sum of squares of columns
4. Communalities (1,2,...,n)
5. Sum of communalities
6. Value of computed criterion.

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

Example	NUMBER OF TESTS	NUMBER OF FACTORS	TAN ANGLE	CRITERION ACCURACY	NO. OF ITER.	RAW VMAX NORM VMAX QUARTIMAX	CARD OUTPUT
A	200	24	.03490	.0000100	25	Normal	Yes
B	150	21	.01746	.0000010	20	Raw	No
C	100	15	.00870	.0000001	15	Q-Max	Yes

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:

tan 5° = .0875, tan 4° = .0699, tan 3° = .0524, tan 2° = .0349, tan 1° = .01746, tan 30' = .00873, tan 15' = .00436, tan 6' = .00175, tan 3' = .00087, tan 1' = .00029

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 program. 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 programs (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

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

5. CONTROL CARD

Example	IBM CARD COLUMNS						
	1-6	7-12	13-18	19-26	27-29	30-31	32-33
	N	LC	CHECK	CRIT	ITER	NOMVAR	CARD
A	+00200	+00024	+03490	+0000100	+25	+1	+1
B	+00150	+00021	+01746	+0000010	+20	-1	-1
C	+00100	+00015	+00873	+0000001	+15	+0	+1

See example problems for explanation.

6. FORMAT CARD

No format card is needed. Input format is 12F6.3. 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                    VAR2002
      DIMENSION C(200),R(200,20),E(40),H(200)
    1 READ INPUT TAPE 7,2,N,LC,CHECK,CRIT,ITER,NOMVAR,CARD
    2 FORMAT(2I6,F6.5,F8.7,I3,2I2)
      REWIND 3
      REWIND 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 6,8
    8 FORMAT(23H1      INPUT   FACTORS   )
      IF (N-MFIRST) 3,3,4
    3 ASSIGN 44 TO JUMP
    4 LG=N/MFIRST
      MLAST=MFIRST*LG
      IF (N-MLAST) 6,5,6
    5 MLAST=MFIRST
      GO TO 7
    6 LG=LG+1
      MLAST=N-MLAST
    7 DO 11 I=1,N
   11 H(I)=0.0
      DO 30 LA=1,LG
      IF (LG-LA) 12,12,15
   12 M1=MLAST
   15 DO 22 I=1,M1
      READ INPUT TAPE 7,20, (R(I,K),K=1,LC)
   20 FORMAT(12F6.3)
      WRITE OUTPUT TAPE 6,199,I,(R(I,K),K=1,LC)
   22 CONTINUE
      IF (NOMVAR) 28,28,23
   23 DO 27 I=1,M1
      L=L+1
      DO 25 K=1,LC
   25 H(L)=H(L)+(R(I,K)**2)
      H(L)=SQRT(H(L))
      DO 26 K=1,LC
   26 R(I,K)=R(I,K)/H(L)
   27 CONTINUE
   28 WRITE TAPE ITAPE,((R(I,K),I=1,M1),K=1,LC)
   30 CONTINUE
      END FILE ITAPE
      REWIND ITAPE
      READ INPUT TAPE 7,31,LOOP
   31 FORMAT(I10)
      IF (NOMVAR) 33,35,37
   33 WRITE OUTPUT TAPE 6,34,LOOP
   34 FORMAT(43H1      RAW VARIMAX PROGRAM  JOB NO.          I10 )
      GO TO 39
```

```
35 WRITE OUTPUT TAPE 6,36,LOOP
36 FORMAT(43H1      QUARTIMAX PROGRAM  JOB NO.              I10)
   GO TO 39
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
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,K1),I=1,M1),K1=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
   SUMXY=SUMXY+(2.*(X*Y))
   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
60 Z=0.25*Z
   IF (ABSF(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 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 LOOM,(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 FORMAT(55H0    NO CONVERGENCE EVEN AFTER DESIRED ITERATIONS          )
    XCRIT=SUMY
    GO TO 170
159 WRITE OUTPUT TAPE 6,160
160 FORMAT(55H0     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,LG
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(1H I5,5X,(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 (1H0F20.3)
    WRITE OUTPUT TAPE 6,240
240 FORMAT(20H0      COMMUNALITIES )
    WRITE OUTPUT TAPE 6,245,(C(I),I=1,N)
245 FORMAT (1H010F11.3)
    WRITE OUTPUT TAPE 6,250
250 FORMAT(30H0      SUM OF COMMUNALITIES     )
    WRITE OUTPUT TAPE 6,235,SUMY
    WRITE OUTPUT TAPE 6,255
255 FORMAT(30H0      CRITERION VALUE          )
    WRITE OUTPUT TAPE 6,235,XCRIT
    GO TO 1
*DATA
```

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:

Group	Number observations	Number missing data	N reduced
1	14	5	9
2	14	1	13
3	14	1	13
4	14	2	12
Total	56	9	47

TABLE I

T E S T D A T A

VAR. 1	VAR. 2	VAR. 3	VAR. 4	VAR. 5	VAR. 6	VAR. 7	VAR. 8	VAR. 9	GR	S	D
+0013000	+0004000	+0003000	+0014000	+0010000	+0005000	+0010000	+0007000	+0013000	01	001	001
+0011700	+0004300	+0006200	+0012100	+0014800	+0008000	+0011400	+0007200	+0008500	01	002	001
+0017200	+0005300	+0003000	+0012000	+0016000	+0006000	+0009700	+0006000	+0007400	01	003	001
+0013400	+0006600	+0003000	+0010500	+0010200	+0007200	+0010700	+0006000	+0006500	01	004	001
+0013000	+0007500	+0003000	+0010500	+0010600	+0007900	+0012800	+0007000	+0006000	01	005	001
+0015300	+0004700	+0004100	+0011800	+0012300	+0008100	+0011300	+0006700	+0008900	01	006	001
+0013200	+0008000	+0003000	+0011100	+0011700	+0008400	+0011400	+0007700	+0006000	01	007	001
+0017700	+0007700	+0005300	+0012000	+0012200	+0005000	+0009000	+0012400	+0012500	01	008	001
+0016000	+0006000	+0003500	+0011500	+0010500	+0007000	+0011500	+0006000	+0010000	01	009	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	01	010	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	01	011	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	01	012	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	01	013	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	01	014	001
+0015000	+0005000	+0004000	+0012000	+0011000	+0008000	+0011000	+0008000	+0011000	02	015	001
+0017000	+0004000	+0003000	+0013000	+0011000	+0013000	+0020000	+0006000	+0007000	02	016	001
+0015000	+0005000	+0003000	+0013000	+0013000	+0005000	+0010000	+0009000	+0009000	02	017	001
+0013000	+0004000	+0006000	+0009000	+0012000	+0007000	+0012000	+0006000	+0006000	02	018	001
+0020000	+0005000	+0004000	+0009000	+0012000	+0005000	+0011000	+0008000	+0007000	02	019	001
+0015000	+0004000	+0003000	+0012000	+0011000	+0005000	+0011000	+0006000	+0013000	02	020	001
+0011000	+0006000	+0003000	+0010000	+0012000	+0005000	+0011000	+0006000	+0006000	02	021	001
+0011000	+0007000	+0003000	+0014000	+0011000	+0005000	+0009000	+0007000	+0006000	02	022	001
+0015000	+0006000	+0008000	+0010000	+0016000	+0006000	+0011000	+0011000	+0009000	02	023	001
+0012000	+0004000	+0003000	+0012000	+0011000	+0005000	+0009000	+0006000	+0010000	02	024	001
+0012000	+0004000	+0003000	+0010000	+0013000	+0005000	+0009000	+0006000	+0007000	02	025	001
+0011000	+0007000	+0003000	+0009000	+0011000	+0005000	+0012000	+0007000	+0009000	02	026	001
+0014000	+0005000	+0007000	+0011000	+0010000	+0008000	+0018000	+0011000	+0009000	02	027	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	02	028	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	03	029	001
+0013000	+0006000	+0003000	+0012000	+0009000	+0009000	+0017000	+0008000	+0009000	03	030	001
+0017000	+0005000	+0011000	+0013000	+0016000	+0007000	+0013000	+0010000	+0010000	03	031	001
+0011000	+0007000	+0010000	+0013000	+0017000	+0010000	+0013000	+0016000	+0008000	03	032	001
+0013000	+0004000	+0003000	+0011000	+0009000	+0005000	+0009000	+0008000	+0009000	03	033	001
+0011000	+0004000	+0004000	+0010000	+0012000	+0006000	+0012000	+0009000	+0007000	03	034	001
+0019000	+0006000	+0005000	+0012000	+0021000	+0005000	+0012000	+0010000	+0010000	03	035	001
+0016000	+0005000	+0007000	+0012000	+0010000	+0006000	+0015000	+0009000	+0010000	03	036	001
+0010000	+0010000	+0003000	+0009000	+0009000	+0008000	+0015000	+0008000	+0005000	03	037	001
+0014000	+0004000	+0005000	+0013000	+0008000	+0009000	+0016000	+0006000	+0012000	03	038	001
+0021000	+0006000	+0003000	+0011000	+0021000	+0005000	+0013000	+0009000	+0009000	03	039	001
+0012000	+0004000	+0003000	+0013000	+0009000	+0005000	+0013000	+0006000	+0007000	03	040	001
+0020000	+0004000	+0003000	+0012000	+0015000	+0005000	+0009000	+0006000	+0013000	03	041	001
+0015000	+0008000	+0004000	+0009000	+0010000	+0007000	+0015000	+0013000	+0010000	03	042	001
+0014000	+0005000	+0009000	+0013000	+0009000	+0008000	+0014000	+0006000	+0009000	04	043	001
+0011000	+0008000	+0003000	+0011000	+0009000	+0010000	+0015000	+0006000	+0007000	04	044	001
+0013000	+0004000	+0003000	+0013000	+0013000	+0005000	+0010000	+0012000	+0015000	04	045	001
+0014000	+0004000	+0003000	+0013000	+0013000	+0005000	+0010000	+0007000	+0012000	04	046	001
+0010000	+0008000	+0012000	+0014000	+0016000	+0008000	+0013000	+0012000	+0006000	04	047	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	04	048	001
+0013000	+0005000	+0004000	+0015000	+0012000	+0007000	+0012000	+0012000	+0013000	04	049	001
+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	+0009999	04	050	001
+0022000	+0004000	+0003000	+0012000	+0017000	+0005000	+0009000	+0010000	+0013000	04	051	001
+0016000	+0004000	+0003000	+0011000	+0012000	+0005000	+0010000	+0008000	+0013000	04	052	001
+0010000	+0007000	+0006000	+0014000	+0009000	+0007000	+0013000	+0012000	+0012000	04	053	001
+0017000	+0004000	+0003000	+0014000	+0020000	+0007000	+0012000	+0009000	+0014000	04	054	001
+0013000	+0004000	+0008000	+0013000	+0022000	+0008000	+0015000	+0009000	+0010000	04	055	001
+0018000	+0004000	+0004000	+0013000	+0021000	+0006000	+0013000	+0009000	+0012000	04	056	001

FORMAT F8.3   GR= GROUP   S= SUBJECT   D= DECK   MISSING DATA= +0009999

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

Statistic	Program and Variable Identification					
	Missing Data			No Missing Data		
	i = 1, j = 2	i = 3, j = 4	Program Code	i = 1, j = 2	i = 3, j = 4	Program Code
$\bar{X}_i$	14.351	4.513	MSTD1A	+13.652	+ 5.394	MSTD2A
$\bar{X}_j$	5.406	11.819	MSTD1A	+ 6.144	+11.527	MSTD2A
σ_i	3.015	2.347	MSTD1A	+ 3.194	+ 2.957	MSTD2A
σ_j	1.546	1.547	MSTD1A	+ 2.213	+ 1.567	MSTD2A
N_i	47	47	MSTD1A	56	56	MSTD2A
N_j	47	47	MSTD1A	56	56	MSTD2A
$z_{10,i}$	+ 9.999*	+9.999*	Z1A	- 1.143	+ 1.557	Z2A
$z_{15,j}$	- 0.263	+0.117	Z1A	- 0.517	+ 0.302	Z2A
r_{ij}	- 0.310	+0.226	R1A	- 0.559	- 0.148	R2A
$t_{r_{ij}}$	- 2.184	+1.555	R1A	- 4.956	- 1.100	R2A
df_{ij}	45	45	R1A	54	54	R2A

*+9.999 = missing data.

TABLE III

TEST DATA RESULTS

Statistic	Missing Data: Group K = 2, L = 3, i = 4	Missing Data: Group K = 2, L = 3, i = 6	Missing Data: Program Code	No Missing Data: Group K = 2, L = 3, i = 4	No Missing Data: Group K = 2, L = 3, i = 6	No Missing Data: Program Code
N_i^K	13	13	M1A	14	14	M2A
N_i^L	13	13	and	14	14	and
$\overline{X}_i^K$	11.077	6.308	A1A	11.000	6.571	A2A
$\overline{X}_i^L$	11.538	6.692	"	11.428	6.928	"
σ_i^K	1.706	2.323	"	1.664	2.440	"
σ_i^L	1.450	1.797	"	1.453	1.940	"
df for $t_i^{K,L}$	24	24	M1A	26	26	M2A
df 1 for $F_i^{K,L}$	1	1	A1A	1	1	A2A
df 2 for $F_i^{K,L}$	24	24	"	26	26	"
$t_i^{K,L}$	-0.743	-0.472	M1A	-0.726	-0.429	M2A
$F_i^{K,L}$	0.552	0.223	A1A	0.527	0.184	A2A
	all 4 groups			all 4 groups		
$df1_i$	3	3	A1B	3	3	A2B
$df2_i$	43	43	"	52	52	"
F_i	12.639	0.739	"	9.986	3.963	"

TABLE IV

INVERSION AND ESTIMATION OF COMMUNALITIES PROGRAM

Row/Column	Matrix R^{-1}	Matrix P*	Row/Column	Matrix R^{-1}	Matrix P	Matrix R^2
1,2	0.067	-0.290	1,1	1.769	0.435	0.435
1,3	0.174	-0.077	3,3	1.760	0.432	0.432
2,5	0.318	-0.081	5,5	1.745	0.427	0.427

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

TABLE V

FACTOR ANALYSIS AND ROTATION PROGRAMS

Row i Col. j	FL_{ij}	h_i^2	L_j	FL_{ij}^*	$\sum FL_j^{*2}$	$\sum h^2$	$\sum L$
1,2	0.143	0.294	1.564	0.057	1.320		
2,1	0.552	0.500	2.126	0.646	1.600		
7,3	-0.357	0.607	0.812	-0.769	1.581		
				Sum:	4.501	4.501	4.501

FL = factor loading, h^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 ($\sum h^2_{est}$) = 4.585

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

1. Anderson, T. V. (1958), "An Introduction to Multivariate Statistical Analysis," New York, John Wiley and Sons.

2. Dwyer, P. S. (1939), "The Contribution of an Orthogonal Multiple Factor Solution to Multiple Correlation," Psychometrika, 4, 163-171.

3. Guttman, L. (1953), "Image Theory for the Structure of Quantitative Variates," Psychometrika, 18, 277-96.

4. Guttman, L. (1954), "Some Necessary Conditions for Common-Factor Analysis," Psychometrika, 19, 149-161.

5. Harman, H. H. (1960), "Modern Factor Analysis," The University of Chicago Press.

6. Hotelling, H. (1933), "Analysis of a Complex of Statistical Variables into Principal Components," Journal of Educational Psychology, 24, 417-41 and 498-520.

7. Hotelling, H. (1935b), "Simplified Calculation of Principal Components," Psychometrika, 1, 27-35.

8. Kaiser, H. F. (1957), "Lower Bound for the Number of Common Factors," Research Report 11, Contract No. AF 41(657)-76, University of California.

9. Kaiser, H. F. (1958), "The Best Approximation of a Common Factor Space," Unpublished manuscript, Bureau of Educational Research, University of Illinois.

10. Kaiser, H. F. (1958), "The Varimax Criterion for Analytic Rotation in Factor Analysis," Psychometrika, 23, 187-200.

11. Kaiser, H. F. (1960), "The Application of Electronic Computers to Factor Analysis," Educational and Psychological Measurement, 20, 141-151.

12. Poates, W. D. and Pierce, M. L. (1958), "Symmetrical Matrix Inversion," SHARE, PA No. 573 CF095.

13. Rao, C. R. (1952), "Advanced Statistical Methods in Biometric Research," New York, John Wiley and Sons.

14. Thomson, G. (1950), "The Factorial Analysis of Human Ability," University of London Press.

15. Wrigley, C. (1957), "The Distinction between Common and Specific Variance in Factor Theory," Brit. J. Stat. Psych., 10, 81-98.

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).

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          MSTD1A0
      DIMENSION SUM(24000),SQ(24000),COUNT(24000),R(24000),NX(24000),
     1NY(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,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
      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
```

```
      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
   41 FORMAT(I10)
      WRITE OUTPUT TAPE 6,42,IA
   42 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.))
      CB=(COUNT(ID)*SQ(IC))-(SUM(IB)**2)
      IF (CB) 45,45,56
   45 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
```

```
$COMPILE 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(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
      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 7,41,IA
   41 FORMAT(I10)
      WRITE OUTPUT TAPE 6,42,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 6,61
   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(1H0I14,F19.3,F20.3,F9.0)
   70 CONTINUE
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP         Z1A00
      DIMENSION SUM(22000),SQ(22000),COUNT(22000),R(22000),NX(22000),
     1NY(22000),FORM(13),Z(22000)
      EQUIVALENCE (R,SUM),(R,COUNT),(R,SQ),(R,NX),(R,NY),(R,Z)
    1 READ INPUT TAPE 7,2,N,LY,NTOTAL,NONE,NTWO,CARDS,D
    2 FORMAT(5I6,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
      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
      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      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) 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
      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
   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
      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(IC)-D) 86,88,86
   86 IF (R(IE)-D) 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+(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          Z2A00000
      DIMENSION SUM(22000),SQ(22000),R(22000),NX(22000),NY(22000),
     1Z(22000),FORM(13)
      EQUIVALENCE (R,SUM),(R,SQ),(R,NX),(R,NY),(R,Z)
    1 READ INPUT TAPE 7,2,N,LY,NTOTAL,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)
```

```
      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
      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   )
      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,L,SQ(IC),SUM(IB),COUNT
   66 FORMAT(1H0I14,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,LG
      IA=LY
      IF (LG-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 (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          M1A2000
      DIMENSION A(190,120),SUM(190),SQUARE(190),COUNT(190),NX(25),NY(25)
     1,XMEAN(190),DEVNY(190),YMEAN(190),DEVNX(190),LC(190),MLAST(190),
     1FORM(13)
      EQUIVALENCE (DEVNX,SQUARE),(DEVNY,LC),(MLAST,YMEAN)
    1 READ INPUT TAPE 7,2,N,LY,LX,D
    2 FORMAT(3I6,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)=(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)
      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.)
```

```
      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 7,33,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/
     1SUM(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),
```

```
     1SUM(I),YMEAN(I),DEVNY(I),TEE,DF
  55 FORMAT(1H0I5,I4,I6,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(200,100),SUM(200),SQUARE(200),COUNT(200),NX(25),NY(25)
     1,XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),LC(200),MLAST(200),
     1FORM(13)
      EQUIVALENCE (DEVNY,LC),(MLAST,YMEAN),(SUM,XMEAN),(SQUARE,DEVNX)
    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(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
      READ INPUT TAPE 7,8,(FORM(I),I=1,13)
    8 FORMAT(13A6)
      DO 32 LA=1,LX
      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=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
   16 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,((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   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)),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(1H0I5,I4,I6,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          A1A200
      DIMENSION A(200,100),SUM(200),SQUARE(200),COUNT(200),TOTAL(200),
     1XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),SOM(200),SQERE(200),
     1LC(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(3I6,F8.3)
      REWIND 3
      REWIND 4
      MFIRST=100
      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
      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
      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 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)
   WRITE TAPE 4,((SUM(I),SQUARE(I),COUNT(I),XMEAN(I),DEVNX(I)),I=NFIR
  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     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=NFIRS
  1T,NLAST)
35 CONTINUE
   DO 65 K=1,LX
   DO 36 L=1,LY
   NFIRST=NX(L)
   NLAST=NY(L)
   READ TAPE 4,((SOM(I),SQERE(I),TOTAL(I),YMEAN(I),DEVNY(I)),I=NFIRST
  1,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(1H0I9,I8,I4,43H    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
```

```
      SKIP=0.0
   45 WRITE OUTPUT TAPE 6,47,I,J,K,YMEAN(I),DEVNY(I)
   47 FORMAT(1H0I9,I8,I4,76H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
     1S                                 2(F10.3))
      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,I,J,K,XMEAN(I),DEVNX(I)
   53 FORMAT(1H0I9,I8,I4,52H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
     1S       F14.3,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)+SQERE(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(1H0I9,I8,I4,F13.3,F14.3,4F5.0,F9.3,F11.3,3F10.3)
   60 CONTINUE
   65 CONTINUE
      REWIND 4
   70 CONTINUE
      REWIND 3
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP           A2A200
      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),
     1SOM(200),SQERE(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(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
      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,((SUM(I),SQUARE(I),XMEAN(I),DEVNX(I)),I=NFIRST,NLAST)
   WRITE TAPE 4,((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(38HO  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,((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,((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,I,J,K
42 FORMAT(1HOI9,I8,I4,43H    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,I,J,K,YMEAN(I),DEVNY(I)
47 FORMAT(1HOI9,I8,I4,76H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
  1S                              2(F10.3))
   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,I,J,K,XMEAN(I),DEVNX(I)
53 FORMAT(1HOI9,I8,I4,52H    PLEASE CHECK YOUR DATA FOR THE TWO GROUP
  1S         F14.3,F10.3)
   GO TO 60
```

```
   54 TOP1=((SUM(I)**2)/COUNT(J))+((SOM(I)**2)/COUNT(K))
      DF2=COUNT(J)+COUNT(K)
      TOP2=((SUM(I)+SOM(I))**2)/(DF2)
      XBAR=TOP1-TOP2
      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
   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(J),COUNT(K),DF1,DF2,Z3,
     1XMEAN(I),DEVNX(I),YMEAN(I),DEVNY(I)
   59 FORMAT(1H0I9,I8,I4,F13.3,F14.3,4F5.0,F9.3,F11.3,3F10.3)
   60 CONTINUE
   65 CONTINUE
      REWIND 4
   70 CONTINUE
      REWIND 3
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          A1B200
      DIMENSION A(200,100),SUM(200),SQUARE(200),COUNT(200),TOTAL(200),
     1XMEAN(200),DEVNY(200),YMEAN(200),DEVNX(200),SOM(200),SQERE(200),
     1LC(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(3I6,F8.3)
      REWIND 3
      MFIRST=100
      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
      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
      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 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     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=NFIRS
     1T,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(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)
   70 CONTINUE
      REWIND 3
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          A2B200
      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),
     1SOM(200),SQERE(200),FORM(13)
      EQUIVALENCE (MLAST,YMEAN),(LC,DEVNY)
    1 READ INPUT TAPE 7,2,N,LY,LX
    2 FORMAT(3I6)
      REWIND 3
      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(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
   10 M1=MLAST(LA)
   11 DO12 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,((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  I10   )
   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
   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
   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)
   70 CONTINUE
      REWIND 3
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          R1A1000
      DIMENSION SUM(100),SOM(100),SQUARE(100),SQERE(100),PROD(100),
     1A(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 7,3,(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)
   11 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
      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),
     1SQUARE(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
   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),
     1SQUARE(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(I10)
      WRITE OUTPUT TAPE 6,53,NEW
   53 FORMAT(38H1  CORRELATION  MATRIX    JOB NUMBER  I10   )
      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),
     1SQUARE(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))-(SOM(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) 71,71,72
   71 SUM(L)=8.888
      GO TO 80
   72 SUM(L)=(PROD(L)/(SQRT(1.-CC)))*(SQRT(TOTAL(L)))
   80 CONTINUE
      WRITE OUTPUT TAPE 6,6,I,((JA(L,1),PROD(L),TOTAL(L),SUM(L)),L=L1,
     1NEND)
    6 FORMAT(1H0I7/(1H I10,F9.3,F9.0,F9.3,1H I10,F9.3,F9.0,F9.3,1H I10,F
     19.3,F9.0,F9.3))
      IF (CARD) 83,90,81
   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
```

```
   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          R2A1400
      DIMENSION SUM(140),R(140,140),SIGMA(140),PROD(140),JA(140,140),
     1TEE(140),FORM(13)
      EQUIVALENCE (JA,R)
    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=140
      M1=MFIRST
      READ INPUT TAPE 7,3,(FORM(I),I=1,13)
    3 FORMAT(13A6)
      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=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=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)
   36 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 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   )
      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
      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
   60 PROD(L)=9.999
      TEE(L)=9.999
      GO TO 80
   65 PROD(L)=TOP/BELOW
      CC=PROD(L)**2
      IF (1.-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 6,6,I,DF,((JA(L,1),PROD(L),TEE(L)),L=L1,NEND)
    6 FORMAT(1H0I7,F8.0/4(1H I8,F10.3,F10.3))
      IF (CARD) 83,90,81
   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(TEE(M))
      IF (CC-TVALUE) 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 6,6,I,DF,((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          R1B2000
      DIMENSION SUM(200),SOM(200),SQUARE(200),SQERE(200),PROD(200),
     1A(200,115),TOTAL(200),JA(200,115),NX(25),NY(25),FORM(13)
      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=115
      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) 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=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
      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
      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)
```

```
36 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),
  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(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
   DO 90 LE=1,LY
   NFIRST=NX(LE)
   NLAST=NY(LE)
   DO 90 I=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))-(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))-(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,I,((JA(M,1),PROD(M),TOTAL(M),SUM(M)),M=1,NA)
 6 FORMAT(1H0I7/(1H I10,F9.3,F9.0,F9.3,1H I10,F9.3,F9.0,F9.3,1H I10,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=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,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),
     1TEE(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)
      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,(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 M1=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
```

```
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,M1
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=NFIRST,NEND
   CB=(COUNT*SIGMA(I))-(SUM(I)**2)
   SIGMA(I)=SQRT(CB/CA)
55 CONTINUE
   DO 90 LE=1,LY
   NFIRST=NX(LE)
   NLAST=NY(LE)
   DO 90 I=NFIRST,NLAST
   M=0
   READ TAPE JTAPE,(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.-CC) 71,71,72
71 TEE(M)=8.888
   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(1H0I7,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
```

```
      CC=ABSF(TEE(M))
      IF (CC-TVALUE) 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,((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          RCOM200
      DIMENSION R(200,100),C(200),JA(200),FORM(13)
    1 READ INPUT TAPE 7,2,N,NA,DIAG,INVERS,CARDS
    2 FORMAT(2I6,F2.0,I2,F2.0)
      READ INPUT TAPE 7,3,(JA(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)
      DO 10 I=L,NA
      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 DO 17 K=1,M1
      L=L+1
      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(I10)
      WRITE OUTPUT TAPE 6,20,NEW
   20 FORMAT(40H1      COMPLETED  CORRELATION  MATRIX   I10  )
      END FILE 2
      REWIND 2
      ASSIGN 32 TO NEW
      LG=N/MFIRST
      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
   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,LG
      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(12F6.3)
  122 WRITE OUTPUT TAPE 6,125,KA,(R(I,K),I=1,N)
  125 FORMAT(1H0,I4/(1H 13F9.3))
  130 CONTINUE
  135 CONTINUE
      REWIND 4
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          INV1500
      DIMENSION A(150,150),E(150),R(150),JA(300),F(300)
      EQUIVALENCE (E(1),F(1)),(R(1),F(151))
    1 READ INPUT TAPE 7,2,N,NA,DIAG,SMCINV,SMC,CARDS
    2 FORMAT(2I3,4F2.0)
      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
```

```
      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) 54,54,60
   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
  105 READ INPUT TAPE 7,82,I
   82 FORMAT(I10)
      WRITE OUTPUT TAPE 6,83,I
   83 FORMAT(44H1  INVERTED  MATRIX    JOB NUMBER             I10          )
      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(1H0I8/(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,I,(E(K),K=1,N)
  135 FORMAT(1H0I8/(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) 220,220,195
  195 PUNCH 200,(A(I,K),K=1,N)
  200 FORMAT(12F6.3)
  220 WRITE OUTPUT TAPE 6,94,I,(A(I,K),K=1,N)
  250 CONTINUE
      GO TO 1
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          FAC1500
      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(I6,F2.0,I3,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)
   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) 44,40,44
   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
   55 DO 58 I=1,N
   58 A(I)=1.
      SUM=0.0
   60 DO 65 J=1,N
   65 D(J)=0.0
      MA=0
      M1=MFIRST
      DO 92 LA=1,LG
      IF (LG-LA) 70,70,75
   70 M1=MLAST
   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=ABSF(D(I))
      DO 100 J=2,N
      CMIN=ABSF(D(J))
      IF (CMAX-CMIN) 95,100,100
   95 CMAX=CMIN
      I=J
  100 CONTINUE
      DO 105 J=1,N
  105 A(J)=D(J)/D(I)
      KB=1
      DO 120 J=1,N
      DIFF=ABSF(A(J)-C(J))
      IF (DIFF-EPSLON) 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 SOM=SOM+E(LC)
      ROOT=SQRT(CMAX)
      SUM=SQRT (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-PASENT) 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,LG
      IF (LG-LA) 170,170,175
  170 M1=MLAST
  174 GO TO LOAD,(175,176)
  175 READ TAPE JTAPE,((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 LOAN,(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(40H0  DESIRED NUMBER OF FACTORS EXTRACTED   )
      GO TO 250
  225 WRITE OUTPUT TAPE 6,226
  226 FORMAT(55H0  LATENT ROOT LESS OR EQUAL TO FRACTION OF 1ST ROOT    )
      GO TO 250
  230 WRITE OUTPUT TAPE 6,231
  231 FORMAT(35H0  REQUIRED  PERCENTAGE  ACHIEVED  )
      GO TO 250
  235 WRITE OUTPUT TAPE 6,236
  236 FORMAT(55H0  LATENT ROOT LESS THAN OR EQUAL TO DESIRED VALUE      )
      GO TO 250
  240 WRITE OUTPUT TAPE 6,241
  241 FORMAT(30H0  FIRST NEGATIVE LATENT ROOT )
      LC=LC-1
  250 END FILE 4
      REWIND 4
      REWIND JTAPE
      REWIND ITAPE
      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
  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,J,(D(I),I=1,LC)
  285 FORMAT(1H0I8/(1H010F11.3))
  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,I,(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(20H0      LATENT ROOTS   )
      WRITE OUTPUT TAPE 6,320,(E(J),J=1,LC)
  320 FORMAT(1H0(F16.3,12F8.3))
      WRITE OUTPUT TAPE 6,330
  330 FORMAT(30H0      SUM OF LATENT ROOTS       )
      WRITE OUTPUT TAPE 6,335,SOM
  335 FORMAT(1H0F20.3)
      WRITE OUTPUT TAPE 6,340
  340 FORMAT(20H0      COMMUNALITIES )
      WRITE OUTPUT TAPE 6,345,(C(I),I=1,N)
  345 FORMAT (1H010F11.3)
      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
      END
$DATA
```

```
$COMPILE FORTRAN,PRINT OBJECT,PUNCH OBJECT,EXECUTE,DUMP          VAR
      DIMENSION C(300),R(300,50),E(50),H(300)
    1 READ INPUT TAPE 7,2,N,LC,CHECK,CRIT,ITER,NOMVAR,CARD
    2 FORMAT(2I6,F6.5,F8.7,I3,2I2)
      REWIND 3
      REWIND 4
      MFIRST=300
      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 6,8
    8 FORMAT(23H1      INPUT    FACTORS   )
      IF (N-MFIRST) 3,3,4
    3 ASSIGN 44 TO JUMP
    4 LG=N/MFIRST
      MLAST=MFIRST*LG
      IF (N-MLAST) 6,5,6
    5 MLAST=MFIRST
      GO TO 7
    6 LG=LG+1
      MLAST=N-MLAST
    7 DO 11 I=1,N
   11 H(I)=0.0
      DO 30 LA=1,LG
      IF (LG-LA) 12,12,15
   12 M1=MLAST
   15 DO 22 I=1,M1
      READ INPUT TAPE 7,20, (R(I,K),K=1,LC)
   20 FORMAT(12F6.3)
      WRITE OUTPUT TAPE 6,199,I,(R(I,K),K=1,LC)
   22 CONTINUE
      IF (NOMVAR) 28,28,23
   23 DO 27 I=1,M1
      L=L+1
      DO 25 K=1,LC
   25 H(L)=H(L)+(R(I,K)**2)
      H(L)=SQRT(H(L))
      DO 26 K=1,LC
   26 R(I,K)=R(I,K)/H(L)
   27 CONTINUE
   28 WRITE TAPE ITAPE,((R(I,K),I=1,M1),K=1,LC)
   30 CONTINUE
      END FILE ITAPE
      REWIND ITAPE
      READ INPUT TAPE 7,31,LOOP
   31 FORMAT(I10)
      IF (NOMVAR) 33,35,37
   33 WRITE OUTPUT TAPE 6,34,LOOP
   34 FORMAT(43H1      RAW VARIMAX PROGRAM  JOB NO.          I10 )
      GO TO 39
   35 WRITE OUTPUT TAPE 6,36,LOOP
```

```
36 FORMAT(43H1      QUARTIMAX PROGRAM  JOB NO.              I10)
   GO TO 39
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
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,K1),I=1,M1),K1=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
   SUMXY=SUMXY+(2.*(X*Y))
   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
60 Z=0.25*Z
   IF (ABSF(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 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 LOOM,(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 FORMAT(55H0   NO CONVERGENCE EVEN AFTER DESIRED ITERATIONS
```

```
      XCRIT=SUMY
      GO TO 170
  159 WRITE OUTPUT TAPE 6,160
  160 FORMAT(55H0    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,LG
  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(1H I6,5X,(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 (1H0F20.3)
      WRITE OUTPUT TAPE 6,240
  240 FORMAT(20H0     COMMUNALITIES )
      WRITE OUTPUT TAPE 6,245,(C(I),I=1,N)
  245 FORMAT (1H010F11.3)
      WRITE OUTPUT TAPE 6,250
  250 FORMAT(30H0     SUM OF COMMUNALITIES     )
      WRITE OUTPUT TAPE 6,235,SUMY
      WRITE OUTPUT TAPE 6,255
  255 FORMAT(30H0     CRITERION VALUE          )
      WRITE OUTPUT TAPE 6,235,XCRIT
      GO TO 1
      END
$DATA
```