General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.
A Joint Program for Agriculture and Resources Inventory Surveys Through Aerospace Remote Sensing
February 1982

FORTRAN IMPLEMENTATION OF FRIEDMAN'S TEST FOR SEVERAL RELATED SAMPLES

S. A. Davidson

Lockheed Engineering and Management Services Company, Inc.
1830 NASA Road 1, Houston, Texas 77258

Lyndon B. Johnson Space Center
Houston, Texas 77058
Fortran Implementation of Friedman's Test for Several Related Samples

S. A. Davidson

Lockheed Engineering and Management Services Company, Inc.
1330 NASA Road 1
Houston, Texas 77058

National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Houston, Texas 77058

The FRIEDMAN program is a FORTRAN-coded implementation of Friedman's nonparametric test for several related samples with one observation per treatment/block combination, or as it is sometimes called, the two-way analysis of variance by ranks. A detailed description of the FRIEDMAN program is given in this document. A test data set and its results are also presented to aid the potential users of this program.
FORTRAN IMPLEMENTATION OF FRIEDMAN'S TEST
FOR SEVERAL RELATED SAMPLES

Job Order 71-324

This report describes activities of the Soil Moisture project of the AgRISTARS program.

PREPARED BY
S. A. Davidson

APPROVED BY

J. G. Carnes, Supervisor
Radiation Characterization Section

T. C. Minter, Jr., Manager
Development and Evaluation Department

LOCKHEED ENGINEERING AND MANAGEMENT SERVICES COMPANY, INC.
Under Contract NAS 9-15800
For
Earth Resources Research Division
Space and Life Sciences Directorate
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS
February 1982
PREFACE

The Agriculture and Resources Inventory Surveys Through Aerospace Remote Sensing is a multiyear program of research, development, evaluation, and application of aerospace remote sensing for agricultural resources, which began in fiscal year 1980. This program is a cooperative effort of the U.S. Department of Agriculture, the National Administration (U.S. Department of Commerce), the Agency for International Development (U.S. Department of State), and the U.S. Department of the Interior.

The work which is the subject of this document was performed by the Earth Resources Applications Division, Space and Life Sciences Directorate, Lyndon B. Johnson Space Center, National Aeronautics and Space Administration and Lockheed Engineering and Management Services Company, Inc. The tasks performed by Lockheed Engineering and Management Services Company, Inc., were accomplished under Contract NAS 9-15800.
1. INTRODUCTION

The FRIEDMAN program is a FORTRAN-coded implementation of Friedman's nonparametric test for several related samples with one observation per treatment/block combination, or as it is sometimes called, the two-way analysis of variance by ranks. This test is particularly useful when the assumptions of normal theory fail to hold for a randomized, complete block design. In fact, for this test, the only assumptions made are that the blocks are mutually independent and that within each block the data may be ranked from smallest to largest. A test statistic is then formed based upon the ranks, not the actual data. This statistic is then used to test the null hypothesis, as follows:

\[ H_0: \text{ Treatments have identical effects.} \]

\[ H_1: \text{ At least one treatment tends to yield larger observations than at least one other treatment.} \]

For further details, see reference 1.

A detailed description of the FRIEDMAN program is given in this document. A test data set and its results are also presented to aid the potential users of this program.
2. PROGRAM DESCRIPTION

2.1 GENERAL OVERVIEW

The FRIEDMAN program calculates Friedman's T-statistic for data sets with a maximum of 500 blocks and 30 treatments, allowing for only one observation per treatment/block combination. This is done via the main calling program, MAIN, and five subroutines: INPUT, SORT, XRANK, TSTAT, and OUTPUT. The FRIEDMAN program utilizes an execution time format, providing for a great deal of flexibility in the configuration of the experimental data set. The program is written in FORTRAN IV-H and is currently implemented on the AS/3000 computer in the Earth Observations Data Laboratory, NASA Johnson Space Center.
3. MAIN

The MAIN program is the main routine of FRIEDMAN. This component reads from the terminal the number of card images per block, the number of treatments, and the execution time format. Subsequently, it calls the subroutines that actually calculate the T-statistic.

CALLING PROCEDURE: Not applicable.

INPUT PARAMETERS: NC, NT, FMT

OUTPUT PARAMETERS: M

REFERENCED BY: Not applicable.

SUBPROGRAMS REFERENCED: INPUT, SORT, XRANK, TSTAT, OUTPUT

INPUT/OUTPUT DEVICES: Unit 16 (terminal), Unit 6 (disk, LRECL = 133, BLKSIZE = 133)

LOCAL VARIABLES: The following abbreviations are used in tables throughout this document.
A = alphanumeric
I = integer
R = real

Local variables for the MAIN program are as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XINPUT</td>
<td>R</td>
<td>500 x 30</td>
<td>Input data set.</td>
</tr>
<tr>
<td>RANK</td>
<td>R</td>
<td>500 x 30</td>
<td>Array of ranked data.</td>
</tr>
<tr>
<td>R</td>
<td>R</td>
<td>1 x 30</td>
<td>Sum of ranks for each treatment.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Dimension</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>DEV</td>
<td>R</td>
<td>$1 \times 30$</td>
<td>Array of squared deviations from the expected value of the treatment sum of ranks.</td>
</tr>
<tr>
<td>WORK</td>
<td>R</td>
<td>30</td>
<td>Working storage.</td>
</tr>
<tr>
<td>NC</td>
<td>I</td>
<td>1</td>
<td>Number of card images per block.</td>
</tr>
<tr>
<td>NT</td>
<td>I</td>
<td>1</td>
<td>Number of treatments.</td>
</tr>
<tr>
<td>M</td>
<td>I</td>
<td>1</td>
<td>Number of blocks.</td>
</tr>
<tr>
<td>FMT</td>
<td>A</td>
<td>80</td>
<td>Format of XINPUT.</td>
</tr>
</tbody>
</table>
3.1 INPUT

The INPUT subroutine reads the experimental data set from the disk by reading in all treatments for a fixed block. That is, treatments are arranged horizontally and blocks are arranged vertically. (See figure 1 for data configuration.)

CALLING PROCEDURE: CALL INPUT(XINPUT, FMT, M, NT)

INPUT PARAMETERS: FMT, M, NT

OUTPUT PARAMETERS: XINPUT

REFERENCED BY: The INPUT subroutine is referenced by the driver routine MAIN.

SUBPROGRAM REFERENCED: None.

INPUT/OUTPUT DEVICES: Unit 5, the disk containing the experimental data set

LOCAL VARIABLES:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XINPUT</td>
<td>R</td>
<td>M x NT</td>
<td>Input data set.</td>
</tr>
<tr>
<td>M</td>
<td>I</td>
<td>1</td>
<td>Number of blocks.</td>
</tr>
<tr>
<td>NT</td>
<td>I</td>
<td>1</td>
<td>Number of treatments.</td>
</tr>
<tr>
<td>FMT</td>
<td>A</td>
<td>80</td>
<td>Format of XINPUT.</td>
</tr>
</tbody>
</table>
\[
\begin{array}{cccc}
  x_{1,1} & x_{1,2} & x_{1,3} & \cdots & x_{1,k_1} \\
  x_{1,k_1+1} & \cdots & \cdots & \cdots & x_{1,k_2} \\
  \vdots & & & & \vdots \\
  \vdots & & & & \vdots \\
  x_{2,1} & x_{2,2} & x_{2,3} & \cdots & x_{2,k_1} \\
  x_{2,k_1+1} & \cdots & \cdots & \cdots & x_{2,k_2} \\
  \vdots & & & & \vdots \\
  \vdots & & & & \vdots \\
  \vdots & & & & \vdots \\
  x_{m,1} & x_{m,2} & x_{m,3} & \cdots & x_{m,k_1} \\
  x_{m,k_1+1} & \cdots & \cdots & \cdots & x_{m_1,k_2} \\
  \vdots & & & & \vdots \\
  \vdots & & & & \vdots \\
  \vdots & & & & \vdots \\
  x_{n,1} & x_{n,2} & x_{n,3} & \cdots & x_{n,n} \\
\end{array}
\]

Index $I = \text{block}$; index $J = \text{treatment}$

Figure 1.- Input data configuration.
3.2 SORT

The SORT subroutine sorts the treatment values for each block in ascending order. The algorithm used is a Shell sort (ref. 2).

CALLING PROCEDURE: CALL SORT(XINPUT, INDEX, I, M, NT)

INPUT PARAMETERS: XINPUT, I, M, NT

OUTPUT PARAMETERS: XINPUT, INDEX

REFERENCED BY: The SORT subprogram is referenced by the driver routine MAIN.

SUBPROGRAMS REFERENCED: None.

INPUT/OUTPUT DEVICES: None.

LOCAL VARIABLES:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XINPUT</td>
<td>R</td>
<td>M × NT</td>
<td>On input, the raw experimental data; on output, the sorted experimental data.</td>
</tr>
<tr>
<td>INDEX</td>
<td>R</td>
<td>NT</td>
<td>Array recording treatment exchanges that occur during the sort.</td>
</tr>
<tr>
<td>I</td>
<td>I</td>
<td>1</td>
<td>Number of current block being sorted.</td>
</tr>
<tr>
<td>M</td>
<td>I</td>
<td>1</td>
<td>Number of blocks.</td>
</tr>
<tr>
<td>NT</td>
<td>I</td>
<td>1</td>
<td>Number of treatments.</td>
</tr>
<tr>
<td>NDELTA</td>
<td>I</td>
<td>1</td>
<td>Variable used to partition each block for sorting.</td>
</tr>
<tr>
<td>NSWITCH</td>
<td>I</td>
<td>1</td>
<td>0 if no switch is made; 1 if a switch is made.</td>
</tr>
<tr>
<td>MAX</td>
<td>I</td>
<td>1</td>
<td>NT - NDELTA.</td>
</tr>
<tr>
<td>TEMP</td>
<td>R</td>
<td>1</td>
<td>Temporary storage used when interchanging of treatment values occurs.</td>
</tr>
</tbody>
</table>
3.3 XRANK

The XRANK subroutine, using the sorted experimental data, checks for ties and then assigns ranks. Using the array INDEX, XRANK positions treatment I in column I of array RANK.

CALLING PROCEDURE: CALL XRANK(XINPUT,RANK,INDEX,I,M,NT,WORK)

INPUT PARAMETERS: XINPUT, INDEX, I, M, NT

OUTPUT PARAMETERS: RANK

REFERENCED BY: The XRANK subroutine is referenced by the driver routine MAIN.

SUBPROGRAM REFERENCED: None.

INPUT/OUTPUT DEVICES: None.

LOCAL VARIABLES:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XINPUT</td>
<td>R</td>
<td>M x NT</td>
<td>Array of sorted experimental data.</td>
</tr>
<tr>
<td>RANK</td>
<td>R</td>
<td>M x NT</td>
<td>Array of ranks.</td>
</tr>
<tr>
<td>INDEX</td>
<td>R</td>
<td>NT</td>
<td>Array of treatment exchanges that occur during sort.</td>
</tr>
<tr>
<td>I</td>
<td>I</td>
<td>1</td>
<td>Block number.</td>
</tr>
<tr>
<td>M</td>
<td>I</td>
<td>1</td>
<td>Number of blocks.</td>
</tr>
<tr>
<td>NT</td>
<td>I</td>
<td>1</td>
<td>Number of treatments.</td>
</tr>
<tr>
<td>WORK</td>
<td>R</td>
<td>NT</td>
<td>Working storage.</td>
</tr>
<tr>
<td>NEQ</td>
<td>I</td>
<td>1</td>
<td>Number of treatment values making up a tie.</td>
</tr>
<tr>
<td>NFIRST</td>
<td>I</td>
<td>1</td>
<td>Column number of first of a sequence of equal values.</td>
</tr>
<tr>
<td>LAST</td>
<td>I</td>
<td>1</td>
<td>Column number of the last of a sequence of equal values.</td>
</tr>
</tbody>
</table>
3.4 TSTAT

The TSTAT subroutine calculates Friedman's T-statistic based upon the contents of the array RANK.

**CALLING PROCEDURE:** CALL TSTAT(RANK,R,DEV,M,NT,T)

**INPUT PARAMETERS:** RANK, M, NT

**OUTPUT PARAMETERS:** R, T

**REFERENCED BY:** The TSTAT subroutine is referenced by the driver routine MAIN.

**SUBPROGRAMS REFERENCED:** None.

**INPUT/OUTPUT DEVICES:** None.

**LOCAL VARIABLES:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANK</td>
<td>R</td>
<td>M x NT</td>
<td>Array of the ranks of the experimental data.</td>
</tr>
<tr>
<td>R</td>
<td>R</td>
<td>1 x NT</td>
<td>Rank totals for treatments.</td>
</tr>
<tr>
<td>Z</td>
<td>R</td>
<td>1</td>
<td>Expected value of R(1,J), J = 1, ..., NT.</td>
</tr>
<tr>
<td>DEV</td>
<td>R</td>
<td>NT</td>
<td>([R(1,J) - Z]^2)</td>
</tr>
<tr>
<td>S</td>
<td>R</td>
<td>1</td>
<td>(\sum_j [R(1,J) - Z]^2)</td>
</tr>
<tr>
<td>T</td>
<td>R</td>
<td>1</td>
<td>Friedman's T-statistic.</td>
</tr>
</tbody>
</table>
3.5 OUTPUT

The OUTPUT subroutine prints an arbitrary $M \times N$ array in the following format:

\[
\begin{array}{cccc}
  x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\
  x_{2,1} & x_{2,2} & \cdots & x_{2,N} \\
  \vdots & \vdots & \ddots & \vdots \\
  x_{m,1} & x_{m,2} & \cdots & x_{m,N} \\
\end{array}
\]

where $q = \text{MOD}(NT,8) - 1$.

**CALLING PROCEDURE:** CALL OUTPUT(S,NO,NV)

**INPUT PARAMETERS:** S, NO, NV

**OUTPUT PARAMETERS:** None.

**REFERENCED BY:** The OUTPUT subroutine is referenced by driver routine MAIN.

**SUBPROGRAMS REFERENCED:** None.

**INPUT/OUTPUT DEVICES:** UNIT 6, disk with LRECL = 133, BLKSIZE = 133
### LOCAL VARIABLES:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>R</td>
<td>NO \times NV</td>
<td>Array to be printed.</td>
</tr>
<tr>
<td>NO</td>
<td>I</td>
<td>1</td>
<td>Number of rows.</td>
</tr>
<tr>
<td>NV</td>
<td>I</td>
<td>1</td>
<td>Number of columns.</td>
</tr>
<tr>
<td>H</td>
<td>I</td>
<td>1</td>
<td>NV + 8.</td>
</tr>
<tr>
<td>M</td>
<td>I</td>
<td>1</td>
<td>MOD(NV,8).</td>
</tr>
<tr>
<td>K</td>
<td>I</td>
<td>1</td>
<td>Lower limit of each print iteration.</td>
</tr>
<tr>
<td>L</td>
<td>I</td>
<td>1</td>
<td>Upper limit of each print iteration.</td>
</tr>
</tbody>
</table>
4. EXAMPLE

Assuming the FRIEDMAN program has been compiled, the following EXEC routine may be used to run FRIEDMAN:

&CONTROL OFF
FILEDEF 5 DISK &1 &2 &3 (PERM
FILEDEF 6 DISK FRIEDMAN OUTPUT D
   (PERM LRECL 133 BLKSIZE 13)
FILEDEF 16 TERM (PERM
LOAD (CLEAR NOMAP
START
SPOOL E NOH
PR FRIEDMAN OUTPUT D (CC
SPOOL E HOLD
&EXIT

Suppose the data set to be used is as shown in figure 2. When the above EXEC routine is executed, the following prompt will appear:

ENTER NUMBER OF CARDS PER BLOCK IN I2 FORMAT.

The user will respond with "SPACE 2." The space is necessary because of the I2 format. Upon carriage return, the prompt

ENTER NUMBER OF TREATMENTS IN I2 FORMAT

will appear. The user should respond with "SPACE 7." Upon carriage return, the final prompt will appear:

ENTER INPUT FORMAT.

The user should respond with

(4F3.0,/,3F3.0).

The output will be spooled to the printer no-hold and should appear as shown in figure 3.
Figure 2.- Data set example.
INPUT DATA

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.0</td>
<td>17.0</td>
<td>19.0</td>
<td>14.0</td>
<td>10.0</td>
<td>21.0</td>
<td>9.0</td>
</tr>
<tr>
<td>2</td>
<td>13.0</td>
<td>14.0</td>
<td>15.0</td>
<td>14.0</td>
<td>13.0</td>
<td>13.0</td>
<td>11.0</td>
</tr>
<tr>
<td>3</td>
<td>21.0</td>
<td>9.0</td>
<td>19.0</td>
<td>17.0</td>
<td>9.0</td>
<td>7.0</td>
<td>19.0</td>
</tr>
</tbody>
</table>

a. Input data.

MATRIX OF RANKS

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.5</td>
<td>5.0</td>
<td>6.0</td>
<td>3.5</td>
<td>2.0</td>
<td>7.0</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>3.0</td>
<td>7.0</td>
<td>6.0</td>
<td>5.0</td>
<td>3.0</td>
<td>3.0</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>7.0</td>
<td>2.5</td>
<td>5.5</td>
<td>4.0</td>
<td>2.5</td>
<td>1.0</td>
<td>5.5</td>
</tr>
</tbody>
</table>

b. Ranks and rank totals.

FRIEDMAN T

\[ T = 5.750 \]

c. Friedman's T.

Figure 3.- Output example.
5. REFERENCES


APPENDIX

FRIEDMAN FORTRAN CODE
FILE: FRIEDMAN FORTRAN A  
EUDL / JOHNSON SPACE CENTER

FRIEDMAN'S TEST FOR SEVERAL RELATED SAMPLES

DIMENSION FMT(8), RANK(500,30), REAL XINPUT(500,30), R(1,30), W(30)
INTEGER IN(30)

SET UP PROMPTS AT TERMINAL

WRITE(6,90) FORMAT(I), ENTER NUMBER OF CARDS PER BLOCK IN I2 FORMAT,
READ(5,U93) NC
40 FORMAT(I), ENTER NUMBER OF TREATMENTS IN I2 FORMAT,
READ(6,105) NT
105 FORMAT(I2)

COUNT NUMBER OF CARD IMAGES & DETERMINE NUMBER OF BLOCKS

M=0
120 READ(5,U25)-format(5)
129 FORMAT(I), COUNT NUMBER OF BLOCKS
M=M+1
GO TO 120
9 M=M/NC

READ DATA

CALL INPUT(XINPUT,FMT,M,NT)
WRITE(6,130) I, INPUT DATA

WRITE INPUT DATA

CALL OUTPUT(XINPUT,M,NT)

SORT TREATMENTS & ASSIGN RANKS

DO 20 I=1,M
CALL SORT(XINPUT,INDEX,1,M,NT)
CALL X(1,M,INDEX,1,M,NT,WORK)
20 CONTINUE
WRITE(6,135)
135 FORMAT(I, MATRIX OF RANKS)

WRITE MATRIX OF RANKS

CALL OUTPUT(RANK,M,NT)
WRITE(6,140)
140 FORMAT(150,, RANK TOTALS)

CALCULATE FRIEDMAN'S T STATISTIC

CALL TSTAT(RANK,W,DEV,M,NT,T)

WRITE RANK TOTALS
FILE: FRIEDMAN FORTRAN A EUCL JENNISON SPACE CENTER

CALL OUTPUT(R,1,NT)
WRITE(6,145)
145 FORMAT(1H1, A, F7.3)
WHITE FRIEDMAN'S T STATISTIC
WRITE(6,150) T
150 FORMAT(1H0, /, ' T = ', F7.3)
STOP

SUBROUTINE INPUT(XINPUT,FMT,M,NT)

SUBROUTINE INPUT(XINPUT,FMT,M,NT)
REAL XINPUT(M,NT),FMT(M)
REWIND 5
DU 10 I=1,14
10 READ(5,FMT) (XINPUT(I,J),J=1,NT)
RETURN
END

SUBROUTINE SORT(XINPUT,INDEX,1,14,NT)

SUBROUTINE SORT(XINPUT,INDEX,1,14,NT)
DIMENSION XINPUT(14,NT), INDEX(NT)

BEGIN SORT
20 NDELTAM=NDELTAM/2
IF (NDELTAM .LT. 1) GO TO 50
30 NSWITC=0
MAX=NDELTAM
DO 40 J=1,MAX
40 IF (XINPUT(I,J) .LE. XINPUT(I,J+NDELTAM)) GO TO 40
TEMP=XINPUT(I,J)
INDEX(J)=INDEX(J+NDELTAM)
INDEX(J+NDELTAM)=TEMP
NSWITC=NSWITC+1
CONTINUE
IF (NSWITC .EQ. 0) GO TO 20
50 RETURN
END

SUBROUTINE XRANK(XINPUT,RANK,INDEX,1,14,NT,WORK)

SUBROUTINE XRANK(XINPUT,RANK,INDEX,1,14,NT,WORK)
REAL XINPUT(14,NT), RANK(1,NT), WORK(NT)
INTEGER INDEX(NT)

ASSIGN INITIAL RANKS & STORE IN ARRAY WORK
DU 10 J=1,NT
10 WORK(J)=FLOAT(J)
N=NT-1
NEQUAL=1

A-2
FILE: FRIEDMAN FORTRAN A  EODL / JOHNSON SPACE CENTER

CHECK FOR TIES & FIND MEAN RANK WHEN DETECTED

DO 20 K=1,N
   IF (XINPUT(I,K) .NE. XINPUT(I,K+1)) GO TO 20
   NEQUAL=NEQUAL+1
   GO TO 20
20 IF (NEQUAL .GT. 1) GO TO 40
40 NFIRST=K-NEQUAL+1
   INC=NEQUAL+1
   LAST=NFIRST+INC
   SUM=0.
   DO 50 L=NFIRST,LAST
      SUM=SUM+WORK(L)
   50 WORK(L)=SUM/NEQUAL
20 CONTINUE

ASSIGN FINAL RANKS TO TREATMENTS

DO 70 MM=1,NT
   RANK(M,index(MM))=WORK(MM)
70 END

SUBROUTINE OUTPUT(S,NO,NV)

SUBROUTINE OUTPUT(S,NO,NV)
   INTEGER H
   REAL S(NO,NV)
   IF (M .EQ. 0) H=1
   M=MOD(NV,H)
   K=1
   DO 10 KK=1,M
      WRITE(6,105)
105 FORMAT(1X,/)  
      L=MOD(NV-1,H)
      WRITE(6,110) (N+K=L)
110 FORMAT(1X,1X,0X,1X,1X,1X)
      WRITE(6,120)
120 FORMAT(1X,/)  
      DO 30 I=1,NO
      WRITE(6,130) I,(S(I,J),J=K,L)
      130 FORMAT(1X,H0,I3,H0,F12.5))
      K=L+1
   10 CONTINUE
   IF (M .EQ. 0 .OR. NV .LE. 8) GO TO 60
   L=K+M-1
   WRITE(6,140) (I,I=K,L)
140 FORMAT(1X,/)  
   WRITE(6,150)
150 FORMAT(1X,/)  
   DO 50 I=1,NO
      WRITE(6,130) I,(S(I,J),J=K,L)
      50 RETURN
END

SUBROUTINE TSTAT(RANK,H,DEV,M,NT,T)

SUBROUTINE TSTAT(RANK,H,DEV,M,NT,T)
   REAL RANK(H,NT),R(I,NT)
   REAL DEV(I,NT)
   FIND RANK TOTALS FOR EACH TREATMENT

DO 30 J=1,NT
30 CONTINUE
R(I,J)=0.
        DO 40 I=1,M
            H(I,J)=H(I,J)+RANK(I,J)
        30 CONTINUE

CALCULATE EXPECTATION OF RANK TOTALS
        Z=FLOAT(M*(NT+1))/2.

CALCULATE SQUARED DEVIATIONS FROM MEAN
        DO 50 LL=1,NT
            DEV(I,LL)=(H(I,LL)-Z)**2
            S=S+DEV(I,LL)
        50 CONTINUE
        DO 60 IJ=1,NT
            U=FLOAT(M*NT*(NT+1))
            S=S+DEV(IJ)
        60 CONTINUE

CALCULATE T
        T=(12./U)*S

RETURN
END