FATIGUE STRENGTH REDUCTION MODEL: RANDOM3 and RANDOM4 USER MANUAL

 Prepared by:
 Lola Boyce, Ph.D., P.E.
 Thomas B. Lovelace

 APPENDIX 2
 of Annual Report
 of Project Entitled
 Development of Advanced Methodologies
 for Probabilistic Constitutive Relationships
 of Material Strength Models
 NASA Grant No. NAG 3-867

 Prepared for:
 NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
 Lewis Research Center
 Cleveland, OH 44135

 (NASA-CR-184940) FATIGUE STRENGTH REDUCTION MODEL: RANDOM3 AND RANDOM4 USER MANUAL,
 APPENDIX 2 Annual Report (Texas Univ.)
 CSCL 20K
 N89-23891

 The Division of Engineering
 The University of Texas at San Antonio
 San Antonio, TX 78285
 January, 1989
FATIGUE STRENGTH REDUCTION MODEL:
RANDOM3 and RANDOM4 USER MANUAL

Prepared by:
Lola Boyce, Ph.D., P.E.
Thomas B. Lovelace

APPENDIX 2
of Annual Report
of Project Entitled
Development of Advanced Methodologies
for Probabilistic Constitutive Relationships
of Material Strength Models

NASA Grant No. NAG 3-867

Prepared for:
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Lewis Research Center
Cleveland, OH 44135

The Division of Engineering
The University of Texas at San Antonio
San Antonio, TX 78285
January, 1989
<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2.0 Theoretical Background</td>
<td>2</td>
</tr>
<tr>
<td>3.0 Input Data</td>
<td>4</td>
</tr>
<tr>
<td>4.0 Sample Problem</td>
<td>7</td>
</tr>
<tr>
<td>5.0 References</td>
<td>11</td>
</tr>
<tr>
<td>6.0 Appendix A: Physical Quantities, Symbols, and Units</td>
<td>12</td>
</tr>
<tr>
<td>7.0 Appendix B: RANDOM3 Sample Problem: Source, Input and Output Files</td>
<td>13</td>
</tr>
<tr>
<td>8.0 Appendix C: RANDOM4 Sample Problem Source, Input and Output Files</td>
<td>37</td>
</tr>
<tr>
<td>9.0 Appendix D: IMSL Subroutine Calls from RANDOM3 and RANDOM4</td>
<td>76</td>
</tr>
<tr>
<td>10.0 Appendix E: SAS/GRAPH Program for RANDOM3 and RANDOM4</td>
<td>77</td>
</tr>
</tbody>
</table>
1.0 INTRODUCTION

This User Manual documents the FORTRAN programs RANDOM3 and RANDOM4. They are based on fatigue strength reduction, using a probabilistic constitutive model. They predict the random lifetime of an engine component to reach a given fatigue strength (see Section 2.0, Theoretical Background).

Included in this Manual are details regarding the theoretical backgrounds of RANDOM3 and RANDOM4, input data instructions and sample problems illustrating the use of RANDOM3 and RANDOM4. Appendix A gives information on the physical quantities, their symbols, FORTRAN names and both SI and U.S. Customary units. Appendix B and C include photocopies of the actual computer printout corresponding to the sample problems. Appendices D and E detail the IMSL, Version 10, subroutines and functions called by RANDOM3 and RANDOM4 and SAS/GRAPH programs that can be used to plot both the probability density functions (p.d.f.) and the cumulative distribution functions (c.d.f.).
2.0 THEORETICAL BACKGROUND

Fatigue strength data are usually presented as cycles to failure for each of several stress amplitudes, the familiar S-N diagram. Results indicate that for lower stress amplitudes the cycles (or time) to failure increases. Thus, a power curve fit through the data yields a monotonically decreasing curve. In general, this curve is represented as

\[ S = \left[ \frac{N}{C'} \right]^{-1/m'} \]  

(6)

where the primitive variables in this equation are as follows: \( S \) is the applied constant amplitude alternating stress at failure or fatigue strength, \( N \) is number of cycles, \( C' \) is a material parameter that varies from specimen to specimen and \( m' \) is a material constant.\(^3\) Equation (6) can be written in terms of "cycles to reach a given fatigue strength" as

\[ N = C'S^{-m'} \]  

(7)

Recently another fatigue strength reduction model has been proposed that takes into account the effect of temperature as well as other parameters that affect strength.\(^4\) The general form of the constitutive relationships for this model is applied to the constituents of high temperature composite materials. Specifically, it is applied herein for the case of a single material constituent. The mechanical property of interest is fatigue strength which is expressed in terms of primitive variables, including the general categories of temperature, mechanical cycles and mean stress. For these categories, the relationship becomes

\[ \frac{S}{S_O} = \left[ \frac{T_F - T}{T_F - T_O} \right]^n \left[ \frac{S_F - \sigma}{S_F - \sigma_o} \right]^m \frac{\log N_{MF} - \log N_{M}}{\log N_{MF} - \log N_{MO}} \]  

(8)

where \( S \) is the applied constant amplitude alternating stress at failure (fatigue strength) at current (or operating) temperature, \( T \), mean stress, \( \sigma \), and mechanical cycle, \( N_M \). \( S_O \) is fatigue strength at reference temperature, \( T_O \) (usually room temperature), reference mean stress (or residual stress), \( \sigma_o \), and reference mechanical cycle, \( N_{MO} \). Also, \( T_F \) is the final or melting temperature of the material, \( S_F \) is the final or tensile strength of the material, and \( N_{MF} \) is the final mechanical cycle or lifetime. Empirical parameters, \( n \), \( m \), and \( q \), are determined from available experimental data or estimated from anticipated behavior of the particular product term.\(^5\) Note that the term containing mechanical cycles is expressed in terms of the log of cycles rather than cycles. This formulation is attractive when \( N_M \) and \( N_{MO} \) are small compared to \( N_{MF} \). The equation may be solved for \( N_M \), or the "cycles to reach a given fatigue strength." The expression is

\[ N = 10 \exp \left[ \log N_{MF} - \left( \log N_{MF} - \log N_{MO} \right) \left( \frac{S}{S_O} \left[ \frac{T_F - T}{T_F - T_O} \right]^n \left[ \frac{S_F - \sigma}{S_F - \sigma_o} \right]^m \right) \right]^{1/q} \]  

(9)
For values typical of a cast nickel base-superalloy subjected to typical loads and temperatures, equation (9) indicates increasing life for decreasing temperature, decreasing tensile mean stress, and decreasing applied alternating stress. It indicates decreasing life for increasing temperature, decreasing compressive mean stress, and increasing applied alternating stress. Therefore, equation (9) predicts observed trends in general.

Probabilistic analysis, via simulation, yields the distribution of the dependent random variable, cycles, $N$. A probability density function (p.d.f.) of cycles is generated using the maximum penalized likelihood method for RANDOM3. For RANDOM4, a p.d.f. of cycles is generated using the maximum entropy method. Maximum entropy uses Jaynes' principle which says that "the minimally prejudiced distribution is that which maximizes the entropy subjected to the constraints supplied by the given information."
3.0 INPUT DATA

Data input for RANDOM3 and RANDOM4 is user friendly and easy to manipulate (see, for example, the file entitled NORMAL.INP, in Section 4.0). The first twelve lines of input have the same format, 2E12.4 and the last two lines differ. The last two lines of input have the formats 13,2X,13,2X,2E12.4,2X,I3 and I3, respectively. A brief, line by line description is given along with an example for each line (NOTE: the ruler is to aid the user in formatting and is not a part of the input). A table listing the physical quantities, their units and symbols is given in Appendix A.

1. Random Number Generator Seed, ISEED, and Sample Size, NTOT

EXAMPLE:

```
123456789012345678901234567890
1 40
```

2. Ultimate Tensile Strength, SF

EXAMPLE:

```
123456789012345678901234567890
900.0000 45.0000
```

3. Log of Final Cycle, NMF

EXAMPLE:

```
123456789012345678901234567890
8.0000 0.8000
```

4. Reference Fatigue Strength, SO

EXAMPLE:

```
123456789012345678901234567890
500.0000 25.0000
```

5. Log of Reference Cycle, NMO

EXAMPLE:

```
123456789012345678901234567890
7.0000 0.7000
```
6. Current Fatigue Strength, S
EXAMPLE:
123456789012345678901234567890
250.0000 12.0000

7. Residual Compressive Stress, SIGO
EXAMPLE:
123456789012345678901234567890
20.0000 1.0000

8. Current Mean Stress, SIG
EXAMPLE:
123456789012345678901234567890
150.0000 7.5000

9. Temperature Exponent, XXN, Stress Exponent, XXM, and Cycle Exponent, XXQ
EXAMPLE:
123456789012345678901234567890
0.5000 0.0150

10. Melting Temperature, TF
EXAMPLE:
123456789012345678901234567890
1500.0000 75.0000

11. Reference Temperature, TO
EXAMPLE:
123456789012345678901234567890
20.0000 0.6000
12. Current Temperature, T

EXAMPLE:

\[123456789012345678901234567890\]
\[850.0000\ \ \ 25.0000\]

13. The DESPL\(^1\) parameters are NODE, INIT, ALPHA, EPS, and MAXIT and are entered in that order as follows:

EXAMPLE:

\[123456789012345678901234567890\]
\[21\ \ 0\ \ 20.0000\ \ 1.0E-05\ \ 30\]

14. The DESPL parameter, IOPT, is entered as follows:

EXAMPLE:

\[1234567890\]
\[2\]
4.0 SAMPLE PROBLEMS FOR RANDOM3 AND RANDOM4

The objective of these programs is to predict the random lifetime to reach a given fatigue strength for an engine component. The theory is based on fatigue strength reduction, using a probabilistic constitutive model. The only difference between RANDOM3 and RANDOM4 is the method used to generate p.d.f. estimates. RANDOM3 uses maximum penalized likelihood, while RANDOM4 uses maximum entropy (see Section 2.0, Theoretical Background). RANDOM3 and RANDOM4 input parameters are given in Table A2.1.

TABLE A2.1 RANDOM3 and RANDOM4 input (SI units)

<table>
<thead>
<tr>
<th>FORTRAN Name</th>
<th>Distribution Type</th>
<th>Mean (Value)</th>
<th>Standard Deviation (Value)</th>
<th>Standard Deviation (% of Mean)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF</td>
<td>normal</td>
<td>900.0</td>
<td>45.0</td>
<td>(3%)</td>
</tr>
<tr>
<td>NMF</td>
<td>lognormal</td>
<td>8.0</td>
<td>0.8</td>
<td>(10%)</td>
</tr>
<tr>
<td>SO</td>
<td>lognormal</td>
<td>500.0</td>
<td>25.0</td>
<td>(5%)</td>
</tr>
<tr>
<td>NMO</td>
<td>lognormal</td>
<td>7.0</td>
<td>0.7</td>
<td>(10%)</td>
</tr>
<tr>
<td>S</td>
<td>lognormal</td>
<td>250.0</td>
<td>12.5</td>
<td>(5%)</td>
</tr>
<tr>
<td>SIGO</td>
<td>lognormal</td>
<td>-20.0</td>
<td>-1.0</td>
<td>(1%)</td>
</tr>
<tr>
<td>SIG</td>
<td>lognormal</td>
<td>150.0</td>
<td>7.5</td>
<td>(5%)</td>
</tr>
<tr>
<td>XXN</td>
<td>normal</td>
<td>0.5</td>
<td>0.015</td>
<td>(0.3%)</td>
</tr>
<tr>
<td>XXM</td>
<td>normal</td>
<td>0.5</td>
<td>0.015</td>
<td>(0.3%)</td>
</tr>
<tr>
<td>XXQ</td>
<td>normal</td>
<td>0.5</td>
<td>0.015</td>
<td>(0.3%)</td>
</tr>
<tr>
<td>TF</td>
<td>normal</td>
<td>1500.0</td>
<td>45.0</td>
<td>(3%)</td>
</tr>
<tr>
<td>TO</td>
<td>normal</td>
<td>20.0</td>
<td>0.6</td>
<td>(3%)</td>
</tr>
<tr>
<td>T</td>
<td>normal</td>
<td>850.0</td>
<td>25.5</td>
<td>(3%)</td>
</tr>
</tbody>
</table>
The input is entered in the following format in a file entitled NORMAL.INP.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>900.0000</td>
<td>45.0000</td>
</tr>
<tr>
<td>8.0000</td>
<td>0.8000</td>
</tr>
<tr>
<td>500.0000</td>
<td>25.0000</td>
</tr>
<tr>
<td>7.0000</td>
<td>0.7000</td>
</tr>
<tr>
<td>250.0000</td>
<td>12.5000</td>
</tr>
<tr>
<td>20.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>150.0000</td>
<td>7.5000</td>
</tr>
<tr>
<td>0.5000</td>
<td>0.0150</td>
</tr>
<tr>
<td>1500.0000</td>
<td>75.0000</td>
</tr>
<tr>
<td>20.0000</td>
<td>0.6000</td>
</tr>
<tr>
<td>850.0000</td>
<td>25.5000</td>
</tr>
</tbody>
</table>

21   0   20.00  1.0E-05  30

Execution of RANDOM3 and RANDOM4 (source code entitled NR3.FOR and NR4.FOR, respectively) produces files entitled RANDM33 and RANDM44. These give intermediate results (see Appendices B and C). Execution also produces plotfiles entitled PLOT1 and PLOT2 (see Appendices B and C). These files are used to plot the X and Y axes of the probability density function (p.d.f.) and the cumulative distribution function (c.d.f.), respectively, generated by RANDOM3 and RANDOM4. The plots are drawn from the plotfiles by the SAS/GRAPH graphing program (see Appendix D). These plots for the sample problem are shown Figures 1, 2, 3, and 4. This same sample problem has been reported in Boyce and Chamis. There, however, it utilized U.S. Customary units and older versions of RANDOM3 and RANDOM4 (using IMSL Version 9.2 subroutines).
Fig. A2.1  p.d.f. of log of mechanical cycles for fatigue strength reduction model, using maximum penalized likelihood method of p.d.f. generation.

Fig. A2.2  c.d.f. of log of mechanical cycles for fatigue strength reduction model, using maximum penalized likelihood method of p.d.f. generation.
Fig. A2.3  p.d.f. of log of mechanical cycles for fatigue strength reduction model, using maximum entropy method of p.d.f. generation.

Fig. A2.4  c.d.f. of log of mechanical cycles for fatigue strength reduction model, using maximum entropy method of p.d.f. generation.
5.0 REFERENCES

1 IMSL, "STAT/LIBRARY, FORTRAN Subroutines for Statistical Analysis", Houston, Texas


6.0 APPENDIX A

PHYSICAL QUANTITIES, SYMBOLS, AND UNITS

The physical quantities, their symbols and units for the fatigue crack growth model are given in the following table.

Table A2.2 Physical quantities, symbols, and units for fatigue crack growth model for RANDOM3 and RANDOM4.

<table>
<thead>
<tr>
<th>Physical Quantity</th>
<th>Theory Symbol</th>
<th>FORTRAN Name</th>
<th>SI</th>
<th>U.S.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ultimate Tensile Strength</td>
<td>SF</td>
<td>SF</td>
<td>MPa</td>
<td>ksi</td>
</tr>
<tr>
<td>Final Cycle (lifetime)</td>
<td>N_MF</td>
<td>NMF</td>
<td>dimensionless</td>
<td></td>
</tr>
<tr>
<td>Reference Fatigue Strength</td>
<td>SO</td>
<td>SO</td>
<td>MPa</td>
<td>ksi</td>
</tr>
<tr>
<td>Reference Cycles</td>
<td>N_MO</td>
<td>NMO</td>
<td>dimensionless</td>
<td></td>
</tr>
<tr>
<td>Current Fatigue Strengths</td>
<td>S</td>
<td>S</td>
<td>MPa</td>
<td>ksi</td>
</tr>
<tr>
<td>Residual Compressive Stress</td>
<td>( \sigma_0 )</td>
<td>SIGO</td>
<td>MPa</td>
<td>ksi</td>
</tr>
<tr>
<td>Current Mean Stress</td>
<td>( \sigma )</td>
<td>SIG</td>
<td>MPa</td>
<td>ksi</td>
</tr>
<tr>
<td>Empirical Material Parameters</td>
<td>( m )</td>
<td>XXN</td>
<td>dimensionless</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( q )</td>
<td>XXQ</td>
<td>dimensionless</td>
<td></td>
</tr>
<tr>
<td>Melting Temperature</td>
<td>TF</td>
<td>TF</td>
<td>°C</td>
<td>°F</td>
</tr>
<tr>
<td>Reference Temperature</td>
<td>TO</td>
<td>TO</td>
<td>°C</td>
<td>°F</td>
</tr>
<tr>
<td>Current Temperature</td>
<td>T</td>
<td>T</td>
<td>°C</td>
<td>°F</td>
</tr>
</tbody>
</table>
7.0 APPENDIX B

RANDOM3 SAMPLE PROBLEM: SOURCE, INPUT AND OUTPUT FILES
C CHAOS (MICROMECHANICS CONSTITUTIVE EQUATIONS)
C RANDOMIZED AND APPLIED TO FATIGUE STRENGTH
INTEGER NTOT, ISEEED, M, INITM, XMIS, MOD, NODE
REAL XM, YS, YM, TS, EPS, F, RWK3F, J312, ALPHM
COMMON / WORKSP/R WKSP
DIMENSION SF(1000), XLMNF(1000), SO(1000)
DIMENSION XLMH(1000), S(1000)
DIMENSION SIG(1000)
DIMENSION XM(1000), XM(1000), XX(1000)
DIMENSION XM(1000), BNDX(1000)
DIMENSION TF(1000), TBU(1000), T(1000)
DIMENSION STA(1000), DEN(1000), UTY(1000)
DIMENSION XXY(999), PFF(999)
DIMENSION XXXK(999), PFFFF(999)
DIMENSION BXX(999), FFF(999)
DIMENSION C(1000)
EXTERNAL RNLNL, RNM-, RNMOR, DESPL, IWKM,
1001 FORMAT(SE12,4)
1002 FORMAT(112,112)
1003 FORMAT(14,14)
1004 FORMAT(14,14)
1009 FORMAT(13,2X,13,2X,2E12,4,2X,13)
1010 FORMAT(13,13)
1011 FORMA(13,13)
55 C LOGNORMAL ULTIMATE TENSILE STRENGTH - SF
READ (3,1002) ISEEED, NTOT
WRITE (6,1003) ISEEED, NTOT
READ (3,1011) XM, XS
WRITE (6,1011) XM, XS
YS = SQRT(LOG(1,0(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2
CALL RNLNL(NTOT, YM, YS, SF)
WRITE (4,10) 2020
2020 FORMAT('LOGNORMAL SF')
WRITE (6,1011) (SF(I), I=1,1,NTOT)
55 C LOGNORMAL LOG OF FINAL CYCLE XLMNF
WRITE (6,1002) ISEEED, NTOT
READ (3,1011) XM, XS
WRITE (6,1011) XM, XS
YS = SQRT(LOG(1,0(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2
CALL RNLNL(NTOT, YM, YS, XLMNF)
WRITE (4,20) 2021
2021 FORMAT('LOGNORMAL XLMNF')
WRITE (6,1011) (XLMNF(I), I=1,1,NTOT)
55 C LOGNORMAL FATIGUE STRENGTH AT REFERENCE CONDITIONS, SO
WRITE (6,1001) ISEEED, NTOT
READ (3,1011) XM, XS
WRITE (6,1011) XM, XS
YS = LOG(XS/XM)**2
YM = LOG(XM) - 0.5*YS**2
CALL RNLNL(NTOT)
CALL RNLNW(NTOT, YM, YS, 30).

WRITE(*,2022).

2022 FORMAT(5,1X,'LOGNORMAL 30:')
WRITE(*,1001) (999(I=1, NTOT))

C LOGNORMAL LOG OF REFERENCE CYCLES, XLNM
WRITE(*,1002) (ISEED, NTOT)
READ(3,1011) XM, XS
WRITE(6,1011) XM, XS
YS = SQRT(LOG(1.0+(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLNW(NTOT, YM, YS, XLNM)
WRITE(*,2023)

C LOGNORMAL (XLNM)
WRITE(*,1001)(XLMHO(I), I=1, NTOT)

C LOGNORMAL FATIGUE STRENGTH AT CURRENT CONDITIONS, S
WRITE(*,1002) (ISEED, NTOT)
READ(3,1011) XM, XS
WRITE(6,1011) XM, XS
YS = SQRT(LOG(1.0+(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2.
CALL RNSET(ISEED)
CALL RNLNW(NTOT, YM+YS, S).
WRITE(*,2024)

C DEFINE RANDOM STRESSES
S
C LOGNORMAL REFERENCE STRESS, SIGO
WRITE(*,1002) (ISEED, NTOT)
READ(3,1011) XM, XS
WRITE(6,1011) XM, XS
YS = SQRT(LOG(1.0+(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLNW(NTOT, YM+YS, SIGO)

C CHANGE SIGO TO NEGATIVE VALUES FOR COMpressive
C RESIDUAL STRESSES
DO 201 I = 1, NTOT
SIGO(I) = -SIGO(I)
201 CONTINUE

WRITE(*,2036)

2036 FORMAT(5,1X,'LOGNORMAL SIGO:')
WRITE(*,1001)(SIGO(I), I=1, NTOT)

C LOGNORMAL CURRENT STRESS, SIG
WRITE(*,1002) (ISEED, NTOT)
READ(3,1011) YM, XS
WRITE(6,1011) YM, XS
YS = SQRT(LOG(1.0+(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLNW(NTOT, YM+YS, SIGO)
WRITE(*,2037)

2037 FORMAT(5,1X,'LOGNORMAL SIGO:')
WRITE(*,1002) (ISEED, NTOT)
READ(3,1011) YM, XS
WRITE(6,1011) YM, XS
CALL RNSET(ISEED)
CALL RNLNW(NTOT, XM+XSIGO)
DO 202 I = 1, NTOT
YS(I) = XLNM(I)+YM
202 CONTINUE

WRITE(*,2025)
CALCULATE CDF OF LOG OF CURRENT CYCLES

READ(3,1010)I0PT
WRITE(6,1010)I0PT
992 FORMAT('CDF PARAMETERS')
WRITE(6,1010)I0PT
X0=SNM1
DO 400 I=1-NODE
P=CDF(X0,I0PT,NODE,BNDS,DENS)
BNDSX(I)=X0
X0=P*HM
400 CONTINUE

994 FORMAT('CDF OF LOG OF CURRENT CYCLES, LOG XNM,
1X AXIS OF PDF, CDF PLOT')
WRITE(6,1001)(DISTX(I),I=1,NODE)
993 FORMAT('ORDERED LOG OF CURRENT CYCLES, LOG XNM,
1X AXIS OF PDF, CDF PLOT')
WRITE(6,1001)(BNDSX(I),I=1,NODE)

WRITE LOG OF CURRENT CYCLES AND CDF OF LOG OF CURRENT
TO THE PLOT FILES

WRITE(35,990)
WRITE(35,991)(BNDS(J),DISTX(J),J=1,NODE)
STOP
END

SUBROUTINE SORT(X,N)
DIMENSION Y(10000)
N1=N-1
DO 1 I=1,N1
1 U=X(I)
DO 2 J=I,N
2 IF (Y(J).LT.U) GO TO 2
TEMPI=I
Y(J)=U
Y(I)=TEMPI
1 CONTINUE
RETURN
END

IMSL Name: DISPL/DDISPL (Single/Double precision version)

Computer: IBM/SINGLE
Revised: November 1, 1990

Purpose: Nonparametric probability density function estimation
by the penalized likelihood method.

Usage: CALL DISPL (NOBS, Y, NODE, BNDS, WMIT, ALPHA, MAXIT, EPS,
DENS, STAT, HESS, LHESS, ILDHI, DENSEST, B,
IPV1, WK2)

Arguments:
NOBS - Number of observations. (Input)
SUBROUTINE D3SPL (NOBS, X, NODE, BDNS, INIT, ALPHA, MAXIT, EPS, DENS, STAT, HESS, LDHESS, ILOHI, DENEST, B, IPVT, WK2)

INTEGER NOBS, NODE, INIT, MAXIT, LDHESS, ILOHI(NODE),
REAL ALPHA, EPS, X(N), BDNS(2), DENS(N), STAT(N),
INTEGER I, INTR, IPTR, ITER, K, KN1, KM2, KPI, KP2, M, MLOD,
REAL BK, BKM1, BSMALL, CK, CKM1, CKM2, CKMCM1, CKPI, CKPI2,
INTEGER MINCR, SAVE

DOUBLE PRECISION SUM1, SUM2, SUM3

SPECIFICATIONS FOR ARGUMENTS
INTR, IPTR, ITER, K, KN1, KM2, KPI, KP2, M, MLOD,
REAL BK, BKM1, BSMALL, CK, CKM1, CKM2, CKMCM1, CKPI, CKPI2,
INTEGER MINCR, SAVE

SPECIFICATIONS FOR LOCAL VARIABLES

SPECIFICATIONS FOR INTRINSICS

intrinsic abs, max1, max2, min0, mod1, sort
CALL EIPSH ('D3SL')

NER = 1
IF (NODE .LE. 4) THEN
CALL EISTI (1, NODE)
CALL EIMES (5, 2, NODE) = Z(11) - 1. The number of mesh nodes; NODE must be an odd integer greater than 1
ELSE IF (MOD(NODE, 2) .EQ. 0) THEN
CALL EISTI (1, NODE)
CALL EIMES (5, 3, NODE) = Z(11) + 1 must be an odd integer greater than 1
ENDIF
IF (ALPHA .LE. 0.0) THEN
CALL EISTR (1, ALPHA)
CALL EIMES (5, 4, ALPHA) = Z(12) - 1. The penalty weight, the factor which controls smoothness; ALPHA must be greater than 0.
ENDIF
IF (MAXIT .LE. 0.0) THEN
CALL EISTI (1, MAXIT)
CALL EIMES (5, 5, MAXIT) = Z(11) - 1. The maximum number of iterations; MAXIT must be greater than 0.
ENDIF
IF (BND1 .GT. BND2) THEN
CALL EISTR (1, BND1)
CALL EISTR (2, BND2)
CALL EISTI (1, NODE) = Z(11) - 1 and BND2 = Z(12) - 1. The minimum value for X; BND1, must be less than or equal to the maximum value for X, BND2.
ENDIF
IF (INIT .NE. 0) THEN
IF (DENS1 .LE. 0.0 OR DENS(NODE) .LE. 0.0) THEN
CALL EISTR (1, DENS1)
CALL EISTR (2, DENS(NODE))
CALL EISTI (1, NODE)
CALL EIMES (5, 7, DENS1 = Z(11) and DENS(NODE) = Z(11)) = Z(12) - 1. The beginning and ending initial estimates of the density must be zero.
ENDIF
IF (DENS(ISHN) .GT. DENS1) OR DENS(NODE) .LT. 0) THEN
CALL EIMES (5, 8, 'The initial estimates of the density, DENS, must be greater than or equal to 0.')
END IF
4001 = 0
DO 10 XI=1, NQBS
   IF (X(XI).LT.BNDS1) OR. (X(XI).GT.BNDS2) THEN
      NQBI = NQBI + 1
   END IF
10 CONTINUE
IF (NQBI .EQ. NQBS) THEN
   CALL EIMES (5, 9, 'All elements in X lie outside the interval BNDS1 to BNDS2, at least one element of X must lie in this interval.' )
ENDIF
IF (EPS .LE. 0.0) THEN
   EPS1 = 1.0E-4
ELSE
   EPS1 = EPS
ENDIF
IF (MINCD .NE. 0) GO TO 2000
C
IMPTR = 0
C
IF (INIT .EQ. 0) THEN
   DENS(1) = 0.0
   DENS(2) = 2.0/(BNDS2-BNDS1)
   M = 3
ELSE
   M = NODE
ENDIF
C
20 IF (INIT .EQ. 0) THEN
   MOLD = M
   IMPTR = IMPTR + 1
   M = MIN(IMPR, MINCD(IMPR))
END IF
C
M = (BNDS2-BNDS1)/(M-1)
C
H3 = H2*M
C
IF (INIT .NE. 0) THEN
   CALL SSCALE (NODE, 1.0/(H3*SUM(NODE*DENS)), DENS, 1)
ENDIF
C
R(1) = BNDS1
DO 10 I = 2, M
   R(I) = R(I-1) + H
10 CONTINUE
C
IMPTR = 0
C
40 IF (X(IMPTR).LT. BNDS1) GO TO 40
   DO 50 K = 1, M - 1
      IL0M1(K+1) = IMPTR
      IL0M1(K+2) = IMPTR + 1
   END IF
50 CONTINUE
IF (IMPTR .LE. NQBS) THEN
   IF (X(IMPTR).LT. R(K1)) THEN
      IL0M1(K1) = IL0M1(K1+2) + 1
      IMPTR = IMPTR + 1
   END IF
ENDIF
60 CONTINUE
C \* FACTOR = 2.0*16*PI*H/3
C IF (INIT .EQ. 0) THEN
   CALL DESPT (K=2, B(2), 1, MOLD, BNDS, DENS, DENSEST, W1, WK, WK)
   TEMP = 1.0/(MARCH)
   DO 90 I=2, M - 1
       DENS(I) = MAX1(TEMP+SQRT(DENSEST(I-1,I)))
   90 CONTINUE
ELSE
   DO 90 I=2, M - 1
       DENS(I) = SQRT(DENS(I))
   90 CONTINUE
END IF
C DENS(M) = 0.0
C DO-140 ITER=1 MAXIT Maximize
C HESS(1,1) = 0.0
C HESS(1,2) = 0.0
C HESS(2,1) = 0.0
C BSMA = 0.0
C DO-120 K=2, M - 1 CK** are true estimates = FK**2
C K1 = K - 1
C K2 = K + 1
C F1 = DENS(K)
C F1M = DENS(KM1)
C F1M2 = DENS(KM2)
C CKM = DENS(KM2)
C CKM2 = DENS(KM2)**2
C CKM = ck M2 CKPM = DENS(KP1)**2
C CKP2 = DENS(KP2)**2
C BK = B(K)
C BKM = B(KM1)
C SUM = SUM + CK
C IF (M .GE. 4) HESS(1,KM1) = 4.0*FK*FKM2*FACTOR
C SUM1 = 0.ODO
C SUM2 = 0.ODO
C SUM3 = 0.ODO
C DO 100 I=10(HI(K1)), ILOHI(K2)
C TEMP = (K(K) - BK)/M
C CONS = (1.0 - TEMP)/(CK + (CKP1 - CK)*TEMP)
C SUM1 = SUM1 + CONS
C SUM2 = SUM2 + CONS*CONS
C SUM3 = SUM3 + CONS*(1.0 - CONS)/TEMP
C 100 CONTINUE
C CKM1 = CK - CKM
C DO 110 I=10(HI(KM1,1)), ILOHI(KM1,2)
C TEMP = (K(K) - BK)/M
C CONS = (K(K) - BK)/M
C SUM1 = SUM1 - CONS*CONS
C SUM2 = SUM2 + CONS*CONS
C SUM3 = SUM3 + CONS*(1.0 - CONS)/TEMP
C 110 CONTINUE
C TEMP = FACTORS*(CKM2 + CKP2 - 4.0*(CKM1 + CKP1) + 6.0*CK) + SUM1
C TEMP = 2.0*TEMP
C BSMA = BSMA + 2.0*CK*TEMP
HES(4+KM1) = TEMP + 1.0*CK*SUM**2

[F(K, NE. 2) HES(2+KM1) = 4.0*FK*FKM1*(-4.0*FACT)+SUM3)

DENEST(KM1, 1) = FK*TEMP
DENEST(KM1, 2) = -2.0*FK

120 CONTINUE
BSMALL = 1.0/H = SUM + BSMALL
C
CALL SCOPY (M-2, DENEST(1, 2) + 1, DENEST(1, 1) + 1)
 Finish with the hessian
CALL SADD (M-2, BSMALL/(2.0*SUM), HESS(3+1), LDBHESS)
C
CALL SCOPY (M-3, HESS(1, 3), LDBHESS, HESS(5+1), LDBHESS)
HESS(5+M-3) = 0
HESS(5+M-2) = 0
CALL SCOPY (M-3, HESS(2, 2), LDBHESS, HESS(4+1), LDBHESS)
HESS(4+M-2) = 0
C
CALL LSQR (M-2, HESS, LDBHESS, 2, 2, HESS, LDBHESS, INPUT, WK)
C
CALL LSQR (M-2, HESS, LDBHESS, 2, 2, INPUT, DENEST(1, 2), 1, DENEST(1, 1))
C
IF (NRCO(1) .NE. 0) GO TO 9000
C
CONS = DOT(M-2, DENEST(1, 3), 1, DENEST(1, 2), 1)
CONS = (1.0/H - SUM - DOT(M-2, DENEST(1, 3), 1, DENEST(1, 1) + 1))/CONS
C
CALL SOPY (M-2, CONS, DENEST(1, 2), L, DENEST(1, 1), L)
C
CALL SOPY (M-2, CONS, DENEST(1, 2), L, DENEST(1, 1), L)
C
TEMP = SNOSE(M-2, DENEST(1, 1))
IF (SNOSE(M-2, DENEST(1, 1)) .LT. EPSI*TEMP) GO TO 150
C
TEMP = TEMP + 1.0*4/SQRT(M-2.0)
DO 130 = 1 + 2, M - 1
DENS(I) = AMAXI(TEMP, DENS(I))
C
130 CONTINUE
140 CONTINUE
CALL HSRT (1, MAXI)
CALL EIMES (3, 1, 'The maximum number of iterations ')
1
150 CALL HSRT (M-2, DENS(2), 1, DENS(2), 1, DENS(2), 1)
C
SUMI = 0.0
C
DO 140 = 100, M
K1 = MAX0(KM1, 1)
SUM1 = SUM1 + (DENS(KM1) - 2.0*DENS(KM1 + DENEST(KM1)) + DENS(KM1 + 1))**2
C
160 CONTINUE

STAT(2) = -0.5*RFACTORSUM
C
170 IF I .GT. MBS

IF (X(I) .GE. BND(1) AND X(I) .LE. BND(2)) THEN

C
C
END IF
170 CONTINUE

STAT(I) = SUM2
C

Evaluate M.L.P.E. mean and variance
SUM1 = 0.0
SUM2 = 0.0
DO 130 K=1, M - 1
   FK = DENS(K)
   FKP1 = DENS(K+1)
   BK = B(K)
   CONS = FK + FKP1 - TEMP = CONS + FKP1
   SUM1 = SUM1 + H3*(TEMP/6.0 + 0.5*BK*CONS
   SUM2 = SUM2 + H3*(TEMP*FKP1)/12.0 + M*BK*TEMP/3.0 -
   0.5*H2*BK*CONS
130 CONTINUE
STAT(3) = SUM1
STAT(4) = SUM2 - SUM1*SUM1
C 9000 CALL EIPDP ('DISPL ')
   Exit section
RETURN
END
File _DBA01=./PLOT.11P; 1 (359.209.0), last revised on 23-NOV-1988 11:26, is a 2 block sequential file owned by UIC [11, 11]. The records are variable length with FORTRAN (FTN) carriage control. The longest record is 23 bytes.


started on printer _T1F7_ on 23-NOV-1988 11:26 from queue T1F7.
8.0 APPENDIX C

RANDOM4 SAMPLE PROBLEM: SOURCE, INPUT AND OUTPUT FILES
C CHAMS MICROMECHANICS CONSTITUTIVE EQUATIONS

DIMENSION NTO,t,IS,NSF,t,EPS,t,PKX3P(1000),ALPHA(2)
DIMENSION SF(10000),XLNMFI(10000),STAT(10000)
DIMENSION SIGU(10000),SIGD(10000)
DIMENSION XS(10000),XM(10000),X(10000)
DIMENSION XM(10000),BMDS(10000),BDMX(10000),BDMX(10000)
DIMENSION TFM(10000),NETW(10000),TOTW(10000)
DIMENSION SM(10),SM(10),XM(10),XM(10),SM(10),XM(10)
DIMENSION AL(12),CUM(12),CUM(12),CUM(12)
COMMON/MPI/PPRINT,TOL,MAXFN

C LOGNORMAL ULTIMATE TENSILE STRENGTH, SF
READ(5,1000)ISEED,NTO
WRITE(6,1005)ISEED,NTO
AM=900.
READ(5,1006)XS,XS
YN=SNORT(LD0.1,04(XS/XM)**2.1)
YM=LOG(YM)-DSYS**2.
CALL ANSET(ISEED)
CALL RNMLNL(NTO,YM,YS,SM)
WRITE(6,1020)
2020 FORMAT('LOGNORMAL SF')
WRITE(6,1020)SF(I),NTO
C LOGNORMAL LOG OF FINAL CYCLE, XLNMFI
READ(5,1005)ISEED,NTO
WRITE(6,1006)XS,XS
YN=0.
YS=SNORT(1.0,XS/XM)**2.
YM=LOG(XM)-0.5SYS**2
CALL ANSET(ISEED)
CALL RNMLNL(NTO,YS,SM)
WRITE(6,1020)
2021 FORMAT('LOGNORMAL XLNMFI')
WRITE(6,1020)XLNMFI(I),NTO
C CALCULATE PDF OF LOG OF CURRENT CYCLES; LOG XMM.
C USING THE MAXIMUM ENTROPY METHOD
C CALCULATE SAMPLE MOMENTS
C NUMBER OF MOMENTS: MMM
C MMM = CALL SHM(XMM,HMM,NMT;SM)
WRITE(39,1001)(SM(1)=HMM)
WRITE(6,2038)
C OBTAIN MAXIMUM ENTROPY DISTRIBUTION
KSTART=1
KDATA=1
C CALCULATE MAX AND MIN ORDINATES FOR PDF (AND CDF)
BNDG(1) = XMM(1) - 0.05*XMM(1)
BNDG(2) = XMM(NMT)+ 0.05*XMM(NMT)
WRITE(66,B372) BNDG(1),BNDG(2)
WRITE(66,B377) BNDG(1),BNDG(2)
C 8077 FORMAT (' BNDG(1),BNDG(2)=',E12.4,E12.4)
CALL MEP1(HMM+SM,3NDS(1),BNDG(1),BNDG(2),KSTART,KDATA,AL,L,CUM)
WRITE(31,1001)(AL(I),I=1,HMM+1)
WRITE(6,2039)
C 2039 FORMAT ('LAGRANGIAN MULTIPLIERS')
WRITE(6,1001)(AL(I),I=1,HMM+1)
C NUMBER OF ORDINATES USED
C CALCULATE WINDOW WIDTH, HH
HH = BNDG(2)-BNDG(1)/(NDUE-1)
C CALCULATE VALUES OF LOG OF CURRENT CYCLES AT WHICH PDF IS ESTIMATED:
C ALSO CALLED 'NODE' VALUES
DO 6001 I=HMM+1,NDUE-2
BNDG(I+2)=BNDG(I)+I*HH
6001 CONTINUE
WRITE(6,983)
C 983 FORMAT ('BNDG OF CURRENT CYCLES; LOG XMM')
WRITE(6,1001)(BNDG(I),I=1,NDUE)
C REORDER BNDG FOR PLOTTING
C SAVE1 = BNDG(I)
SAVE2 = BNDG(INODE)
BNDG(INODE)=BNDG(I)
BNDG(I)=SAVE1
DO 6002 I=1,INODE-2
BNDG(I+1)=BNDG(I+2)
6002 CONTINUE
BNDG(INODE-1)=SAVE2
BNDG(INODE)=SAVE1
C 984 FORMAT ('ORDERED LOG OF CURRENT CYCLES; LOG XMM')
WRITE(6,984)
C FOR 4 MOMENTS THERE ARE 3 LAGRANGIAN MULTIPLIERS
D3MNTS1=EXP(AL(1)+AL(2)+AL(3)+AL(4)+AL(5))*BNDG(I)**2
D3MNTS2=EXP(AL(1)+AL(2)+AL(3)+AL(4)+AL(5))*BNDG(I)**3
D3MNTS3=EXP(AL(1)+AL(2)+AL(3)+AL(4)+AL(5))*BNDG(I)**4
C WRITE LOG OF CURRENT CYCLES AND PDF OF LOG OF CURRENT CYCLES.
C - LOG-XNM TO PLOT FILES
WRITE(*,990)
990 FORMAT(' (E12.4,E12.4)
WRITE(*,991)(BNDX(I),I=1,NODE)
991 FORMAT(E12.4,E12.4)
C CALCULATE CDF OF LOG OF CURRENT CYCLES
IOPT=2
C READ(3,1004)IOPT
WRITE(6,992)
992 FORMAT(' 3CDF PARAMETERS')
WRITE(6,1004)IOPT
X0=BNDX(I)
DO 3003,J=1,NODE
P=SCDF(X0,IOPT,NODE,BNDX,DENS)
BNDX(J)=X0
3003 CONTINUE
WRITE(6,994)
994 FORMAT(' CDF OF LOG OF CURRENT CYCLES, LOG XNM, 
1Y AXIS OF PDF, CDF PLOT')
WRITE(6,1001)(DIISTX(I),I=1,NODE)
C WRITE(6,993)
993 FORMAT(' ORDERED LOG OF CURRENT CYCLES, LOG XNM, 
1X AXIS OF PDF, CDF PLOT')
WRITE(6,1001)(BNDX(I),I=1,NODE)
WRITE(6,1001)(BNDX(I),I=1,NODE)
C WRITE LOG OF CURRENT CYCLES AND CDF OF LOG OF CURRENT
C TO THE PLOT FILES
WRITE(*,990)
WRITE(*,991)(BNDX(I),DIISTX(J),J=1,NODE)
STOP
C SUBROUTINE SORT(Y,N)
DIMENSION Y(10000)
C Y IS THE ARRAY TO BE SORTED
C AT COMPLETION Y(1) IS SMALLEST VALUE
C AT COMPLETION Y(N) IS LARGEST VALUE
N1 = N - 1
DO 1 I=1,N1
J=I+1
DO 2 K=J,N
IF (Y(J).LT.Y(K)) GO TO 2
1 TEMP = Y(J)
Y(J) = Y(K)
Y(K) = TEMP
2 CONTINUE
RETURN
END
C SUBROUTINE SMOM(X,H,NSAMP,SM)
C CALCULATES SAMPLE CENTRAL MOMENTS
C X(I) = SAMPLE VALUES, DIMENSION NSAMP
C H = NUMBER OF MOMENTS DESIRED
C NSAMP = SAMPLE SIZE
C SM = VALUE OF MOMENTS, DIMENSION M
DIMENSION X(10000),SM(10)
C \text{CALCULATE MEAN} \\
\text{SUM}=0.0 \\
\text{DO 1 I=1,NSAMP} \\
1 \text{SUM}=\text{SUM}+X(I) \\
\text{SM(I)=SUM/FLOAT(NSAMP)} \\
\text{IF(M,LT,3)RETURN} \\
C \text{CALCULATE VARIANCE} \\
\text{SM}=0.0 \\
\text{DO 2 I=1,NSAMP} \\
2 \text{SUM}=\text{SUM}+(X(I)-\text{SM(I)})^2 \\
\text{IF(M,LT,3)RETURN} \\
C \text{CALCULATE HIGHER MOMENTS} \\
\text{DO 3 J=1,NSAMP} \\
3 \text{SUM}=\text{SUM}+(X(J)-\text{SM(I)})^{\text{J}} \\
\text{IF(M,LT,3)RETURN} \\
\text{END} \\
\text{SUBROUTINE MEP1(N,CM,XMIN,XMAX,NXP,XP,KSTART,KDATA,AL,CUM)} \\
\text{IMPLICIT REAL*8 (A-H,O-Z)} \\
\text{EXECUTIVE PROGRAM FOR USING MAXIMUM ENTROPY METHOD CONSTRUCTED BY} \\
\text{MOMENTS TO GENERATE A DENSITY FUNCTION} \\
\text{DIMENSION AL(4), CM(4), ETA(4), XP(*)}, \text{CUM(*)}, \text{C}(3), \text{K}(10) \\
\text{COMMON/FAIL/ NFAIL} \\
\text{COMMON/HELP/ } \text{XX(16,101)}, \text{C}(8), \text{M} \\
\text{A LINE DIFFERENT FROM TEXT} \\
\text{COMMON/HEP1/KPRINT,TOL,MAXFN} \\
\text{DATA KPRINT,TOL,MAXFN/} \\
\text{IF(4,E-11) KPRINT=1} \\
\text{WRITE THE INPUT DATA} \\
\text{IF(KDATA.EQ.0) GO TO 1} \\
\text{WRITE (6,24) KDATA} \\
\text{WRITE (6,25) KPRINT} \\
\text{WRITE (6,26) N} \\
\text{WRITE (6,29) XMAX} \\
\text{WRITE (6,30) XMIN} \\
\text{WRITE (6,31) (CM(I),I=1,4) \\
\text{WRITE (6,32) (CM(I),I=5,8) \\
\text{WRITE (6,33) NXP} \\
\text{CONTINUE} \\
\text{FAIL}=0 \\
\text{M=1} \\
\text{X2MIN}=0.0 \\
\text{X2MAX}=1.0 \\
\text{SAV=CA} \\
\text{DO 100 I=1,N} \\
\text{CC(I)=CM(I) \\
\text{CALCULATE THE MOMENTS AT THE MODIFIED LIMITS} \\
\text{CALL TRN1 (XMAX,XMIN,CC,X2MAX,X2MIN,N)} \\
\text{CALCULATE THE MOMENTS ABOUT THE ORIGIN FOR THE MODIFIED LIMITS} \\
\text{STORE THEM IN COMMON IN C}
CALL CONVER(CC,N)

GENERATE THE SIMPSON MULTIPLIERS AND STORE THEM IN HELP COMMON

CALL SIMSON

GENERATE THE X'S POWER FOR SUBROUTINE FUNCT, STORE THEM IN HELP COMMON ARRAY

CALL MULTI((XMAX-XMIN))

DEFINE THE INPUT DATA FOR SUBROUTINE MPOPT

ETA(1)=1.0-12
ETA(2)=70
ETA(3)=1.0-24
ETA(4)=1.0-24
MODE=I

WRITE THE INTERMEDIATE RESULTS YOU HAVE OBTAINED SO FAR

IF (KPRINT.EQ.0) GO TO 2

WRITE (6,34)
WRITE (6,35)
WRITE (6,36)
WRITE (6,37)
WRITE (6,38)
WRITE (6,39)
CONTINUE

FIND A STARTING POINT FOR SUBROUTINE MPOPT TO START THE OPTIMIZATION ALGORITHM

IF (KSTART.EQ.0) GO TO 16
IF (KSTART.EQ.1) WRITE (6,44)
CALL START (XMAX-XMIN,AL,KSTART,CC,N,KPRINT,UMIN,MOL,MAXFN,ETA)
IF (NFAIL.EQ.1) GO TO 9

PRINT THE STARTING VALUES

IF (KPRINT.EQ.0) GO TO 7
GO TO (3,4,5,6), KSTART

WRITE (6,40)
WRITE (6,41) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
WRITE (6,42)
WRITE (6,43)
GO TO 7

WRITE (6,44) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
GO TO 16

WRITE (6,45) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
GO TO 7
CONTINUE

RANGE=XMAX-XMIN

CHANGE STARTING VALUES TO 0-1 DOMAIN FOR KSTART=0
THE ALGORITHM IS SIMILAR TO TRANSFORM APPEARS TO GIVE BETTER
NUMERICAL RESULTS

IF (ABS(XMIN),LT.1.E-10) GO TO 19
DO 17 I=2:NPL
ALS(I)=0.0
17 CONTINUE
DO 19 J=1,N
ALS(J)=ALS(J)+FACTO(J)*XMIN**(J-I+1)*RANGE*I1*AL(J+1)/FACTO(I)
19 CONTINUE
GO TO 50
DO 50 I=2,NPL
ALS(I)=RANGE**I1*AL(I)
50 CONTINUE
PUT AL(I) IN PROPER LOCATIONS
AL(I)=ALS(I+1)
CONTINUE
IF (KPRINT.EQ.0) GO TO 9
WRITE (6,45)
CONTINUE
AL(N+1)=AL(N+2)=0.0
CALL MPMPL (AL,N,ETA,UMIN,MAXFN,MODE,KPRINT)
IF (NFAIL.EQ.0) GO TO 10
IF (KSTART.EQ.0) GO TO 9
KSTART=KSTART+1
K=K+1
GO TO 2
CONTINUE
WRITE (6,46)
RETURN
END
THE PROGRAM HAS FAILED SO FAR TRY ANOTHER STARTING POINT AND TRY
AGAIN
IF (KSTART.EQ.1.AND.N.LE.2) GO TO 9
GO TO 3
CONTINUE
WRITE (6,46)
RETURN
END
CALCULATE THE ZEROETH LAGRANGIAN MULTIPLIER

SUM=0.0
DO 12 I=1,N
SUM=SUM+31*EXP(SUM)
12 CONTINUE
NPL=N+1
DO 13 I=1,N
K=N+2
AL(I)=AL(K-1)
13 CONTINUE
DELTA=(X2MAX-X2MIN)/FLOAT(N-1)
AL(I)=AL(I)-DELTA*SUM
WRITE (6,101)SUM
101 FORMAT (24H SUM OF RESIDUALS SQUARED=,E12.5)
IF (KPRINT.EQ.0) GO TO 14
WRITE (6,147) (AL(I)+I1,NPL)
CONTINUE
C... RESET KSTART TO ZERO
CALCULATE THE LAGRANGIAN MULTIPLIERS FOR THE ORIGINAL LIMITS

CALL TNR2 (XMAX,XMIN,AL,X2MAX,X2MIN,N)

CALCULATE THE CUMULATIVE DISTRIBUTION FUNCTION VALUE AT THE GIVEN

POINT

IF(NXP.EQ.0)RETURN

DO 15 I = 1,NXP

CUM(I) = CDF(XMIN,XMAX,XP(I)-AL,NXP)

CONTINUE

RETURN

15 FORMAT (57X,E18.9,/)  

20 FORMAT (1X,E18.9,/)  

25 FORMAT (1X,E18.9,/)  

26 FORMAT (1X,E18.9,/)  

27 FORMAT (1X,E18.9,/)  

28 FORMAT (1X,E18.9,/)  

29 FORMAT (1X,E18.9,/)  

30 FORMAT (1X,E18.9,/)  

31 FORMAT (1X,E18.9,/)  

32 FORMAT (1X,E18.9,/)  

33 FORMAT (1X,E18.9,/)  

34 FORMAT (1X,E18.9,/)  

35 FORMAT (1X,E18.9,/)  

36 FORMAT (1X,E18.9,/)  

37 FORMAT (1X,E18.9,/)  

38 FORMAT (1X,E18.9,/)  

39 FORMAT (1X,E18.9,/)  

40 FORMAT (1X,E18.9,/)  

41 FORMAT (1X,E18.9,/)  

42 FORMAT (1X,E18.9,/)  

43 FORMAT (1X,E18.9,/)  

44 FORMAT (1X,E18.9,/)  

45 FORMAT (1X,E18.9,/)  

46 FORMAT (1X,E18.9,/)  

47 FORMAT (1X,E18.9,/)  

48 FORMAT (1X,E18.9,/)  

49 FORMAT (1X,E18.9,/)  

END
PRINT*,'GOT BY A15'
GO TO 29
DO 22 I=1,NDIM
GO=GG+G2(I)*G2(I)
PRINT*,'GOT BY A15'
CONTINUE
GO=SORT(GG)
KOUNT=KOUNT+1
MM=1
IF (IPRINT.EQ.0) GO TO 23
IF (MOD+NT=IPRINT).NE.0 GO TO 23
PRINT*,'GOT BY A15'
CALL OUTP (X2+F2,M,NDIM,GG,HUMF,RR)
PRINT*,'GOT BY A15'
CONTINUE
KTB=KT2+1
IF (MODE.EQ.2) GO TO 25
PRINT*,'GOT BY A15'
IF (M.GT.MAX) GO TO 30
PRINT*,'GOT BY A15'
NSOL=0
DO 24 I=1,NDIM
IF (ABS(RR(I)).GT.ETA(2)) .NSOL=NSOL+1
PRINT*,'GOT BY A15'
CONTINUE
IF (NSOL.EQ.0) GO TO 26
PRINT*,'GOT BY A15'
GO TO 29
PRINT*,'GOT BY A15'
IF (GO(2).GT.MAX) .OR.(H.GT.MAX)) .GO TO 26
PRINT*,'GOT BY A15'
GO TO 29
PRINT*,'GOT BY A15'
CONTINUE
PRINT*,'GOT BY A15'
IF (IPRINT.EQ.0) GO TO 27
PRINT*,'GOT BY A15'
WRITE (A,33)
PRINT*,'GOT BY A15'
CALL OUTP (X2+F2,M,NDIM,GG,HUMF,RR)
PRINT*,'GOT BY A15'
DO 28 I=1,NDIM
PRINT*,'GOT BY A15'
CONTINUE
IF NFAIL.GT.100
RETURN
CONTINUE
PRINT*,'KOUNT'
PRINT*,'GOT BY A15'
IF (KOUNT.LE.1) GO TO 11
PRINT*,'GOT BY A15'
IF (KOUNT.LE.10) GO TO 19
PRINT*,'GOT BY A15'
RETURN
KSTD=KSTD+1
IF (KSTD.GT.10) NFAIL=1
IF (NFAIL.EQ.1) RETURN
DO 32 I=1,NDIM
X(I)=X2(I)
G(I)=G2(I)
DO 32 I=1,NDIM
RETURN
IF (F-FY) 30,30,12
DALFA=0.
DO 11 I=1,N
DALFA=DALFA+G(I)*H(I)
11 CONTINUE
IF (DALFA)<24,27,27
IF (FY-FX) 30,30,13
IF (DX-DALFA) 30,30,06
FX=F
DY=DALFA
AMBEA=ALFA
GO TO 14
IF (FY-F) 29,29,29
IF (DY-DALFA) 29,30,29
FY=F
DY=DALFA
AMBEA=AMBEA-ALFA
AMBEA=AMBEA-ALFA
IF (I-FY) 34,34,34
RETURN
14 CONTINUE
IF (NRM/RM,LE.1,E-10) 00 TO 32
IF (NRM/RM,LE.EPS) IER=-3
IRE=1
WRITE(9,33)
FORMAT(7,1X,'THE PROGRAM HAS FAILED')
RETURN
END

SUBROUTINE FUNCT (N,AL,U,GRAD,RR)
IMPLICIT REAL*8 (A-M,0-Z)
THIS SUBROUTINE IS USED TO CALCULATE THE OPTIMIZATION AND THE
GRADIENT AT ANY GIVEN POINT FOR SUBROUTINE POPT
DIMENSION AL(N), GRAD(N), SUM(17), RR(N)
COMMON/FAIL,NFAIL,
COMMON/HDF/S(101),XX(101),C(8),M
C.... ABOVE LINE CHANGED FROM TEXT

N21=2*N+1
ZERO=0.0
DO 1 M=1,N21
SUM(I)=0.0
1 CONTINUE
PRINT*, 'GOT BY C1'
CONTINUE
IF (SZ=ZERO) 11,11,11
DO 3 K=1,N
SZ=SZ+AL(K)*XX(K,I)
3 CONTINUE
PRINT*, 'GOT BY C3'
CONTINUE
IF (SZ<ST,74) 80 TO 9
PRINT*, 'GOT BY C3'
S=S+XX(K,I)
SUM(I)=SUM(I)+SS
DO 4 J=1,N3
   SUM(J)=SUM(J)+XX(J-1)*I3
   PRINT*, 'GOT BY C4'
   CONTINUE
DO 3 I=1,N2
   SUM(I)=SUM(I)/SUM(1)
   PRINT*, 'GOT BY C5'
   CONTINUE
U=0.0
DO 5 I=1,N1
   RRI=(SUM(I-1)-C(I))/C(I)
   U=U+RRI*RR(I)
   PRINT*, 'GOT BY C6'
   CONTINUE
DO 3 J=1,N2
   GRAD(K)=0.0
   DO 7 J=1,N1
      GRAD(K)=GRAD(K)+(SUM(J+K)-SUM(J+1))*SUM(K+1)/C(J)
      PRINT*, 'GOT BY C7'
      CONTINUE
      PRINT*, 'GOT BY C8'
   CONTINUE
   PRINT*, 'GOT BY C9'
   RETURN
   PINT*, 'GOT BY C10'
   CONTINUE
   A=SQ-1.2
   ZERO=ZERO-AA
   GO TO 2
   PRINT*, 'GOT BY C11'
   END

SUBROUTINE START (XMAX, XMIN, ALAMDA, KSTART, CC, NL, IPRINT, UMIN, MODE, A
LAXF, ETA)
IMPLICIT REAL*8 (A-H, O-Z)
DIMENSION R(11), CC(*), ETA(*), ALAMDA(*), X(10), Y(10), W(10+10)
COMMON /FAIL/, NFAIL
GO TO (3, 15, 26), KSTART
1 CONTINUE
   NFAIL=0
   DO 2 I=1,N1
      ALAMDA(I)=0.0
2 CONTINUE
   RETURN
   CONTINUE
   NFAIL=0
   ALAMDA(I)=CC(I)/CC(2)
   ALAMDA(I)=CC(I)/CC(2)
   DO 4 I=3,N1
      ALAMDA(I)=0.0
4 CONTINUE
   RETURN
   CONTINUE
NFAIL=0
MNN=NL/2
MP1=MNN+1
DELTA=(XMAX-XMIN)/FLOAT(NL)
DO 6 I=1,MNN+1
   X(I)=XMIN+FLOAT(I-1)*DELTA
6 CONTINUE
IF (NFAIL.NE.NL) GO TO 19
W(I+1)=1.
DO 7 I=2,NL+2
   W(I)=W(I)/4.0
7 CONTINUE
IF (NL.EQ.2) GO TO 9
MNN=MNN-1
DO 8 I=3,MNN+1
   W(I)=W(I)+0.00002
8 CONTINUE
DO 10 I=1,MP1
   W(I)=0.00002
10 CONTINUE
W(1)=W(1)/W(I)
Y(I)=W(I)*X(I)
DO 11 I=1,MP1
   W(I+1)=W(I+1)*Y(I)
11 CONTINUE
CALL SOLVE (W+Y*XID+MP1,10)
DO 12 I=1,MP1
   W(I)=W(I)+0.00002
12 CONTINUE
DO 13 I=1,MP1
   Y(I)=ALOG(Y(I))
13 CONTINUE
DO 14 I=1,MP1
   Y(I)=Y(I) / Y(I+1)
14 CONTINUE
RETURN
R(I+2)=R(I+2)*R(I+1)
R(3)=R(3)/R(I+2)
IF (NL.EQ.3) GO TO 22
R(NL+1)=R(NL+1)/R(3)
R(I+2)=R(I+2)/R(3)
DO 20 I=2,NL+2
   R(I)=R(I)/R(I+2)
20 CONTINUE
NS=NL-1
DO 22 I=1,NS+2
   R(I)=R(I)+0.00002
22 CONTINUE
RETURN
CONTINUE
DO 23 I=1,NP1
W(I,I)=R(I)
23 CONTINUE
DO 24 J=1,NP1
W(I,J)=W(I-1,J)+X(J)
24 CONTINUE
Y(I)=1./DELTA
DO 25 I=1,N
Y(I+1)=Y(I)
25 CONTINUE
CALL SOLVE (W,Y,NP1,10)
GO TO 12
CONTINUE
N=2
ALAMDA(2)=-.5/CC(2)
ALAMDA(1)=CC(1)/CC(2)
NPAIL=0
CONTINUE
ALAMDA(N+1)=2.0
ALAMDA(N+2)=0.0
C PRINT$='GET BY A$
CALL MPOP (ALAMDA,N,STA,MIN+MAXF,MODE,IPRINT)
C PRINT$='GOT BY B$
IF (NPAIL.EQ.1) RETURN
ALAMDA(N+1)=0.0
N=N+1
GO TO 27
END

SUBROUTINE SOLVE (A,X,NP1,N,NA)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(NA,N), X(N)
N=0
DATA DIV/0.931471817/
DO 6 I=1,N
AA=0.
DO 1 J=1,N
AB=ABS(A(J,I))
IF (AB.LE.AA) GO TO 1
K=J
AA=AB
1 CONTINUE
B=0.4ALOG(AA)
IF (I.EQ.N) GO TO 7
IF (K.EQ.I) GO TO 3
DO 2 J=1,N
AB=ALOG(J)
A(J,J)=A(J,J)+AB
2 CONTINUE
AB=ALOG
X(I)=X(K)
X(K)=AB
3 II=I+1
DO 4 J=I+1,N
AA=A(J,J)/A(I,I)
A(J+I)=AA
4 CONTINUE
SUBROUTINE SIMSON
IMPLICIT REAL*8 (A-H,O-Z)
C
THIS SUBROUTINE IS TO CALCULATE THE SIMPSON MULTIPLIERS
COMMON/HELP/S(101),XX(16,101),C(8),M
C.... ABOVE LINE CHANGED FROM TEXT
S(1)=1.
M=M-1.
DO 1 I=2,N,2
S(I)=4.
1 CONTINUE
N=N-1.
DO 2 I=3,N,2
S(I)=2.
2 CONTINUE
RETURN.
END

SUBROUTINE MULTI (XMAX,XMIN,N)
IMPLICIT REAL*8 (A-H,O-Z)
C
THIS SUBROUTINE IS USED TO GENERATE THE X.S POWER FOR SUBROUTINE
FUNCTION
COMMON/HELP/S(101),XX(16,101),C(8),M
C.... ABOVE LINE CHANGED FROM TEXT
DELTA=(XMAX-XMIN)/FLOAT(M-1)
DO 1 I=1,M
XX(I,1)=XMIN+FLOAT(I-1)*DELTA
1 CONTINUE
RETURN.
END

SUBROUTINE CONVER (CM,NL)
IMPLICIT REAL*8 (A-H,O-Z)
C
THIS SUBROUTINE IS TO CALCULATE THE MOMENTS ABOUT THE ORIGIN
DIMENSION CM(*)

COMMON HELP: G(10)+X(10)+C(10)+H

C.... ABOVE LINE CHANGED FROM TEXT
C(1)=CH(1)
IF (NL.EQ.1) RETURN
DO 2 J=2,NL
C(J)=C(J)-C(1)*J*(-1)**J
M=M+1
DO 1 K=1,M
C(J)=C(J)-(-1)**K*FACTO(J)/(FACTO(K)*FACTO(J-K))*C(1)**K*C(J-K)
1 CONTINUE
RETURN
END

SUBROUTINE TRN1 (X1MAX,X1MIN,C,X2MAX,X2MIN,NL)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION C(1)
SCL=(X1MAX-X1MIN)/(X2MAX-X2MIN)
C(1)=C(1)/SCL-X1MIN/SCL+X2MIN
IF (NL.EQ.1) RETURN
DO 1 I=2,NL
C(1)=C(1)/SCL**(FLOAT(I))
1 CONTINUE
RETURN
END

SUBROUTINE TRN2 (X1MAX,X1MIN,X,X2MAX,X2MIN,N)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION X(1)
DXMAX=X1MAX
DXMIN=X1MIN
DX2MAX=X2MAX
DX2MIN=X2MIN
NP1=NP+1
DO 10 I=1,NP1
10 DX(I)=DXMAX-DXMIN/(DX2MAX-DX2MIN)
A=DX2MIN-DXMIN/S
DX(1)=DX(1)-ALOG(S)
DO 1 I=1,N
DX(I)=DX(I)+DX(I+1)**S
1 CONTINUE
IF(N.EQ.1)GO TO 6
DO 2 I=J,N
2 CONTINUE
FAC=1
DO 3 K=K+1,N
3 CONTINUE
FAC=FAC**DBLE(FLOAT(K))
DO 4 K=K+1,N
4 CONTINUE
DX(J)=DX(J)+FAC**DBLE(FACTO(J-1)**S**(N-J+1)*DX(I+1))
5 CONTINUE
DX(J)=DX(J)/S**(J-1)
CONTINUE

DX(N+1)=DX(N+1)/SN
DO 11 I=1,NP1
X(I)=0X(I)
RETURN
END

SUBROUTINE TRN2 (XMAX,XMIN,X,X2MAX,X2MIN,N)
IMPLICIT REAL (A-H,O-Z)
THIS SUBROUTINE IS USED TO CALCULATE THE LAGRANGIAN MULTIPLIERS AT THE ORIGINAL LIMITS
DIMENSION X(1)
S=XMAX-XMIN)/(X2MAX-X2MIN)
A=X2MIN-(XMIN/S)
X(1)=X(IN)-ALOG(S)
DO 1 I=1,N
X(I)=X(I)+X(I+1)*S
CONTINUE
IF (N.EQ.1) GO TO 5
DO 3 I=J,N
FAC=1.
K=I-2
DO 2 K=FLOAT(K)
FAC=FAC*X
CONTINUE
X(J)=X(J)+FAC/FACT(J-1)*X(J-1)
CONTINUE
X(N+1)=X(N+1)/SN
RETURN
END

FUNCTION CDF (XMIN,XMAX,XP,AL,N)
IMPLICIT REAL (A-H,O-Z)
THIS FUNCTION IS TO CALCULATE THE CUMULATIVE DISTRIBUTION FUNCTION AT A GIVEN POINT
INPUT
XMIN = LOWER BOUND
XMAX = UPPER BOUND
XP = SPECIFIED POINT
AL = ARRAY OF PARAMETERS, DIMENSION N
N = NUMBER OF PARAMETERS
DIMENSION AL(N)
IF (XP.LE.XMIN) GO TO 3
IF (XP.GE.XMAX) GO TO 4
RANGE=XMAX-XMIN
RANGE=XP-XMIN
SS=RANGE/RANGE*51.
JSS=SS
JSS=(JSS/2)*2+5
AREA=0.0
JSM1=JSS-1
DELTA=RANGE/FLOAT(JSM1)
DO 1 I=2,JSM1
1 AREA=AREA+.ENTRF(P(AL,N,X))
CONTINUE
JSM1=JSM1+1
DO 2 J=3,JSM1+1
X=XMIN+FLOAT(J-1)*DELTA
AREA=AREA+2.ENTRPALX
2 CONTINUE
AREA=AREA+ENTRPALX+ENTRPALXP
AREA=AREA*DELTA/3.
CDF=AREA
GO TO 5
5 CDF=0.0
GO TO 5
CDF=1
CONTINUE
RETURN
END

FUNCTION ENTRPALX.
IMPLICIT REAL*8 (A-H,O-Z)
FUNCTION TO EVALUATE THE ENTROPY DENSITY FUNCTION AT A GIVEN POINT
INPUT:
AL(I) = ARRAY CONTAINING PARAMETERS, DIMENSION NPL
NPL = NUMBER OF PARAMETERS
X = GIVEN VALUE

DIMENSION AL(I)
S=AL(1)
DO 1 I=2,NPL
S=S+AL(I)*X**(I-1)
1 CONTINUE
ENTRPALX=EXP(S)
RETURN
END

FUNCTION FACTO (M).
IMPLICIT REAL*8 (A-H,O-Z)
CALCULATES FACTORIAL OF M
FACTO=1
IF (M.EQ.0) RETURN
DO 1 I=1,M
FACTO=FACTO*FLOAT(I)
1 CONTINUE
RETURN
END
### INPUT DATA FOR SUBROUTINE MEP:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT DATA IS PRINTED OUT FOR KDATA = 1 ONLY</td>
<td>KDATA = 1</td>
</tr>
<tr>
<td>INTERMEDIATE OUTPUT EVERY KPRINT(TH) CYCLE</td>
<td>KPRINT = 1</td>
</tr>
<tr>
<td>NUMBER OF KNOWN FIRST MOMENTS</td>
<td>N = 4</td>
</tr>
<tr>
<td>HIGHER LIMIT</td>
<td>XMAX = 9.963779301E+01</td>
</tr>
<tr>
<td>LOWER LIMIT</td>
<td>XMIN = 0.972349819E+01</td>
</tr>
<tr>
<td>FIRST MOMENTS</td>
<td>CC(i) = 0.735481526E+01, 0.370334345E+00</td>
</tr>
<tr>
<td>THE ALLOWED TOLERANCE IN LAPRANGEAN EQUATIONS</td>
<td>TOL = 0.178109803E-06, 0.783791484E-06</td>
</tr>
<tr>
<td>THE CUMULATIVE DISTRIBUTION REQUIRED AT NXP POINTS</td>
<td>NXP = 0</td>
</tr>
</tbody>
</table>
## Intermediate Results for Subroutine MEP

<table>
<thead>
<tr>
<th>Number of Integration Station</th>
<th>M = 31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modified Maximum and Minimum Limits</td>
<td>X2MAX = X2MIN = 0.100000000E+00</td>
</tr>
<tr>
<td>Modified Moments About the Expected Value</td>
<td>C(i) = 0.41679124E+00</td>
</tr>
<tr>
<td>Modified Moments About the Origin</td>
<td>ETA(i) = 0.100000000E+00</td>
</tr>
</tbody>
</table>

### Normal Assumption Starting Method

### Starting Values

<table>
<thead>
<tr>
<th>CYC</th>
<th>M</th>
<th>NORMGRAD RESIDUALS</th>
<th>TOTAL X(1)</th>
<th>X(2)</th>
<th>X(3)</th>
<th>VARIABLES X(4)</th>
<th>R(1)</th>
<th>R(2)</th>
<th>RESIDUALE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>0.17606E-01</td>
<td>0.23715E-02</td>
<td>0.11222E-02</td>
<td>-0.13405E-02</td>
<td>0.23839E-01</td>
<td>0.19798E-01</td>
<td>0.224E-01</td>
<td>-0.417E-02</td>
</tr>
<tr>
<td>7</td>
<td>13</td>
<td>0.1059E-02</td>
<td>0.1059E-02</td>
<td>0.2292E-02</td>
<td>0.1340E-02</td>
<td>0.379E-02</td>
<td>0.291E-02</td>
<td>0.379E-02</td>
<td>0.379E-02</td>
</tr>
<tr>
<td>10</td>
<td>19</td>
<td>0.1214E-02</td>
<td>0.3609E-03</td>
<td>0.2701E-02</td>
<td>0.1059E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>0.34249E-03</td>
<td>0.1200E-03</td>
<td>0.2691E-02</td>
<td>0.1059E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
</tr>
<tr>
<td>13</td>
<td>23</td>
<td>0.1400E-03</td>
<td>0.7244E-03</td>
<td>0.2610E-02</td>
<td>0.1059E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
</tr>
<tr>
<td>18</td>
<td>32</td>
<td>0.99133E-04</td>
<td>0.3630E-02</td>
<td>0.1907E-02</td>
<td>0.1059E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
</tr>
<tr>
<td>7</td>
<td>19</td>
<td>0.34249E-03</td>
<td>0.1200E-03</td>
<td>0.2691E-02</td>
<td>0.1059E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
<td>0.291E-02</td>
</tr>
</tbody>
</table>

### Additional Details

- The table provides intermediate results for subroutine MEP, including integration station numbers, modified maximum and minimum limits, modified moments about the expected value, modified moments about the origin, and starting values.
- Normal assumption starting method is applied.
- The table lists cyclical values, numerical gradient residuals, total variables, and their respective residuals and variables for different iterations.
File EOF: 0 (36) last revised on 23-DEC-1988 11:21, is a 3 block sequential file owned by UIC "11111. The records are variable length with FORTRAN (FTM) carriage control. The longest record is 25 bytes.

Job PLOTZ (604) queued to SYSSPRINT on 23-DEC-1988 11:21 by user NETWORKPRIV, UIC "11111", under account 201000ADD at priority 100.

started on printer "JTP6" on 23-DEC-1988 11:21 from queue "JTP6".
9.0 APPENDIX D
IMSL SUBROUTINE CALLS FROM RANDOM3 AND RANDOM4

RANDOM3

1. RNSET - Initializes a random seed for use in the IMSL random number generators.

2. RNNOR - Generates pseudorandom numbers from a standard normal distribution using an inverse CDF method.

3. RNLNL - Generates pseudorandom numbers from a lognormal distribution.

4. DESPL - Performs nonparametric probability density function estimation by the penalized likelihood method.

5. GCDF - Evaluates a general continuous cumulative distribution function given the ordinates of the density.

RANDOM4

1. RNSET - Initializes a random seed for use in the IMSL random number generators.

2. RNNOR - Generates pseudorandom numbers from a standard normal distribution using an inverse CDF method.

3. RNLNL - Generates pseudorandom numbers from a lognormal distribution.
10.0 APPENDIX E
SAMPLE SAS/GRAPH PROGRAM FOR RANDOM3 AND RANDOM4

data a;
INFILE 'PLOT1.CPR' FIRSTOBS=2;input x y;
GOPTIONS DEVICE=HP7470;
proc gplot;
    axis1 label=(h=1 f=simplex 'LOG OF CYCLES')
        value=(h=1 f=simplex);
    axis2 value=(h=1 f=simplex) label=none;
    plot y*x / haxis=axis1 vaxis=axis2;
    TITLE H=1 A=90 F=SIMPLEX 'PROBABILITY DENSITY FUNCTION';
    symbol i=spline v=square;
data B;
INFILE 'PLOT2.CPR' FIRSTOBS=2;input x y;
proc gplot;
    axis1 label=(h=1 f=simplex 'LOG OF CYCLES')
        value=(h=1 f=simplex);
    axis2 value=(h=1 f=simplex) label=none;
    plot y*x / haxis=axis1 vaxis=axis2;
    TITLE H=1 A=90 F=SIMPLEX 'CUMULATIVE DISTRIBUTION FUNCTION';
    symbol i=spline v=square;