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
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 OF CONTENTS

<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

\[ \frac{S}{S_o} = \left[ \frac{T_F - T}{T_F - T_o} \right]^n \left[ \frac{S_F - \sigma}{S_F - \sigma_o} \right]^m \left\{ \frac{\log N_{MF} - \log N_{MO}}{\log N_{MF} - \log N_{MO}} \right\}^q \]  

(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

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

(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 I3,2X,I3,2X,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:

\begin{align*}
\text{123456789012345678901234567890} \\
250.0000 & \quad 12.0000
\end{align*}

7. Residual Compressive Stress, SIGO

EXAMPLE:

\begin{align*}
\text{123456789012345678901234567890} \\
20.0000 & \quad 1.0000
\end{align*}

8. Current Mean Stress, SIG

EXAMPLE:

\begin{align*}
\text{123456789012345678901234567890} \\
150.0000 & \quad 7.5000
\end{align*}

9. Temperature Exponent, XXN, Stress Exponent, XXM, and Cycle Exponent, XXQ

EXAMPLE:

\begin{align*}
\text{123456789012345678901234567890} \\
0.5000 & \quad 0.0150
\end{align*}

10. Melting Temperature, TF

EXAMPLE:

\begin{align*}
\text{123456789012345678901234567890} \\
1500.0000 & \quad 75.0000
\end{align*}

11. Reference Temperature, TO

EXAMPLE:

\begin{align*}
\text{123456789012345678901234567890} \\
20.0000 & \quad 0.6000
\end{align*}
12. Current Temperature, $T$

**EXAMPLE:**

\[ 123456789012345678901234567890 \]

850.0000 25.0000

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

**EXAMPLE:**

\[ 12345678901234567890123456789012345678901234567890 \]

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</th>
<th>Standard Deviation (Value)</th>
<th>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>SIG</td>
<td>lognormal</td>
<td>-20.0</td>
<td>-1.0</td>
<td>(1%)</td>
</tr>
<tr>
<td>SIGO</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.

<table>
<thead>
<tr>
<th>1</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>900.000</td>
<td>45.000</td>
</tr>
<tr>
<td>8.0000</td>
<td>0.8000</td>
</tr>
<tr>
<td>500.000</td>
<td>25.0000</td>
</tr>
<tr>
<td>7.0000</td>
<td>0.7000</td>
</tr>
<tr>
<td>250.000</td>
<td>12.5000</td>
</tr>
<tr>
<td>20.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>150.000</td>
<td>7.5000</td>
</tr>
<tr>
<td>0.5000</td>
<td>0.0150</td>
</tr>
<tr>
<td>1500.000</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>
<tr>
<td>21</td>
<td>0</td>
</tr>
</tbody>
</table>

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>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ultimate Tensile Strength</td>
<td>SF</td>
<td>SF</td>
<td>MPa, ksi</td>
</tr>
<tr>
<td>Final Cycle (lifetime)</td>
<td>N_{MF}</td>
<td>NMF</td>
<td>dimensionless</td>
</tr>
<tr>
<td>Reference Fatigue Strength</td>
<td>SO</td>
<td>SO</td>
<td>MPa, ksi</td>
</tr>
<tr>
<td>Reference Cycles</td>
<td>N_{MO}</td>
<td>NMO</td>
<td>dimensionless</td>
</tr>
<tr>
<td>Current Fatigue Strengths</td>
<td>S</td>
<td>S</td>
<td>MPa, ksi</td>
</tr>
<tr>
<td>Residual Compressive Stress</td>
<td>σ₀</td>
<td>SIGO</td>
<td>MPa, ksi</td>
</tr>
<tr>
<td>Current Mean Stress</td>
<td>σ</td>
<td>SIG</td>
<td>MPa, ksi</td>
</tr>
<tr>
<td>Empirical Material Parameters</td>
<td>m</td>
<td>XXM</td>
<td>dimensionless</td>
</tr>
<tr>
<td></td>
<td>q</td>
<td>XXQ</td>
<td>dimensionless</td>
</tr>
<tr>
<td>Melting Temperature</td>
<td>TF</td>
<td>TF</td>
<td>°C, °F</td>
</tr>
<tr>
<td>Reference Temperature</td>
<td>TO</td>
<td>TO</td>
<td>°C, °F</td>
</tr>
<tr>
<td>Current Temperature</td>
<td>T</td>
<td>T</td>
<td>°C, °F</td>
</tr>
</tbody>
</table>
7.0 APPENDIX B

RANDOM3 SAMPLE PROBLEM: SOURCE, INPUT AND OUTPUT FILES
CALL RNLM(LMOT,IM,IS,30)
WRITE(6,202)

2022 FORMAT('LOGNORMAL 3D')
WRITE(6,1001) SO((I=1,NTOT))
C LOGNORMAL, LOG OF REFERENCE CYCLES, XLMO
WRITE(6,1002) ISEED, NTOT
READ(3,1011) XH, XS
WRITE(6,1011) XM, XS
YS = SQRT(L(1.0+(XS/XM)**2))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLM(LMOT,IM,YS,XLMO)
WRITE(6,2033)

2023 FORMAT('LOGNORMAL XLMO')
WRITE(6,1001) XLMO((I=1,NTOT))
C LOGNORMAL, FATIGUE STRENGTH AT CURRENT CONDITIONS, S
WRITE(6,1002) ISEED, NTOT
READ(3,1011) XM, XS
WRITE(6,1011) XM, XS
YS = SQRT(L(1.0+(XS/XM)**2,1))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLM(LMOT,IM,YS,XS)
WRITE(6,204)

2024 FORMAT('LOGNORMAL S')
WRITE(6,1001) (S(I=1,NTOT))
C DEFINE RANDOM STRESS, SIG0
WRITE(6,1002) ISEED, NTOT
READ(3,1011) XM, XS
WRITE(6,1011) XM, XS
YS = SQRT(L(1.0+(XS[XM]**2,1))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLM(LMOT,IM,YS,SIG0)

15 CHANGE SIG0 TO NEGATIVE VALUES FOR COMPRESSIVE
C RESIDUAL STRESSES
DO 201 I = 1, NTOT
SIG0(I) = SIGO(I)
201 CONTINUE
WRITE(6,2036)

2036 FORMAT('LOGNORMAL SIG0')
WRITE(6,1002) ISEED, NTOT
C LOGNORMAL, CURRENT STRESS, SIG
WRITE(6,1002) ISEED, NTOT
READ(3,1011) XM, XS
WRITE(6,1011) XM, XS
YS = SQRT(L(1.0+(XS/XM)**2,1))
YM = LOG(XM) - 0.5*YS**2
CALL RNSET(ISEED)
CALL RNLM(LMOT,IM,YS,SIG)
WRITE(6,2037)

2037 FORMAT('LOGNORMAL SIG')
WRITE(6,1002) ISEED, NTOT
C NORMAL EXPONENTS, XNL, XM, XXQ
WRITE(6,1002) ISEED, NTOT
READ(3,1011) YM, YS
WRITE(6,1011) YM, YS
CALL RNMOR(LMOT,XXN,YM)
CALL RNMG(NTOT,XXQ)
DO 202 I = 1, NTOT
XXN(I) = YM+XXQ(I)+YM
202 CONTINUE
WRITE(6,2025)
102 CONTINUE
WRITE(6,9028)
2028 FORMAT(' LOG OF CYCLES TO REACH MEAN FATIGUE STR = ')
WRITE(6,9010)(XNM(I),I=1,NTOT)
- C SORT LOG OF CYCLES
CALL SORT(XNM,NTOT)
WRITE(6,9029)
2029 FORMAT(' SORTED LOG OF CYCLES')
WRITE(6,9010)(XNM(I),I=1,NTOT)
- C CALCULATE PDF OF LOG OF CURRENT CYCLES LOG XNM
READ(3,1009)(NODE,INIT,ALPHA EPS,MAXIT)
WRITE(6,995)
985 FORMAT(' DESPL PARAMETERS')
WRITE(6,1009)(NODE,INIT,ALPHA EPS,MAXIT)
BND5(I)=XNM(I) - 0.05*XNM(I)
BND2(I)=XNM(I) + 0.05*XNM(I)
WRITE(6,995)BND5(I),BND2(I)
- C FORMAT(' PDF OF LOG OF CURRENT CYCLES LOG Y AXIS OF PDF PLOT')
WRITE(6,9010)(DENSI(I),I=1,NODE)
WRITE(6,995)
979 FORMAT(' BND5(I),BND2(I) @ (E12.4,IX,E12.4)
CALL DESPI(NODENODE,INIT,ALPHA MAXIT EPS DENSI STAT)',
979)
WRITE(6,9020)
980 FORMAT(' OUTPUT STATISTICS')
WRITE(6,9010)(STAT(I),I=1,NODE)
WRITE(6,995)
981 FORMAT(' NUMBER OF MISSING VALUES')
WRITE(6,9010)MISS
WRITE(6,995)
982 FORMAT(' Window Width, HH
HH=(BND2(I)-BND5(I))/(NODE-1)')
CALCULATE WINDOW WIDTH, HH
HH=(BND2(I)-BND5(I))/(NODE-1)
- CALCIUATE VALUES OF LOG OF CURRENT CYCLES AT WHICH PDF IS ESTIMATED
ALSO CALLED "NODE" VALUES
DO 6001 I=1,NODE-2
BND5(I)=BND5(I)+(I*XNM)
6002 CONTINUE
WRITE(6,995)
983 FORMAT(' LOG OF CURRENT CYCLES LOG XNM')
WRITE(6,9010)(BND5(I),I=1,NODE)
- C REORDER BND5 FOR PLOTTING
SAVE = BND5(I)
SAVE2 = BND5(I)
DO 6002 I=1,NODE-2
BND5(I)=BND5(I-1)
BND5(I-1)=SAVE2
SAVE = SAVE2
6002 CONTINUE
WRITE(6,995)
984 FORMAT(' ORDERED LOG OF CURRENT CYCLES LOG XNM
11 AX IN PDF, CON PLOT')
WRITE(6,9010)(BND5(I),I=1,NODE)
- C WRITE LOG OF CURRENT CYCLES AND PDF OF LOG OF CURRENT CYCLES
C LOG XNM TO PLOT FILES
WRITE(4,995)
990 FORMAT(' (E12.4,IX,E12.4)')
WRITE(3,991)(BNDS(j), DENS(j), j=1, NODE)
991 FORMAT(E12.4, 1X, E12.4)

CALCULATE PDF OF LOG OF CURRENT CYCLES
READ(3,1010)IOPT
WRITE(*,992)
992 FORMAT(' PDF PARAMETERS')
WRITE(6,1010)IOPT
XD = BNDS(1)
DO A003=1,NODE
P = GCDF(XD, IOPT, NODE, BNDS, DENS)
BNDSX(I) = XD
XO = XD+MM
DISTX(I) = P
1003 CONTINUE
WRITE(3, *)
994 FORMAT(' PDF OF LOG OF CURRENT CYCLES, LOG XNM, 1X AXIS OF PDF, PDF PLOT')
WRITE(6,1001)(DISTX(I), I=1, NODE)

CALCULATE CDF OF LOG OF CURRENT CYCLES
WRITE(6,993)
993 FORMAT(' CDF OF LOG OF CURRENT CYCLES, LOG XNM, 1X AXIS OF PDF, PDF PLOT')
WRITE(6,1001)(BNDSX(I), I=1, NODE)
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(Y, N)
DIMENSION Y(10000)
M = N
DO 1 = 1, N
J = 1
DO 2 = J, N
IF (Y(J) .LT. Y(K)) GO TO 2
Y(K) = Y(J)
Y(J) = TEMP
2 CONTINUE
1 CONTINUE
RETURN
END

IMSL Name: B3SP/LD3SP (Single/Double precision version)

Computer: IBM/SINGLE

Revised: November 1, 1985

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

Usage: CALL B3SP (NOBS, I, NODE, BNDS, INIT, ALPHA, MAXIT, EPS,
           DENS, STAT, HESS, LOGH, ILOG, DENEST, B,
           IPVT, WK2)

Arguments:

NOBS - Number of observations. (Input)
**SUBROUTINE D3PL** (NOBS, NODE, BNDS, INIT, ALPHA, MAXIT, EPS, DENS, STAT, HESS, LDHESS, ILOHI, DENEST, B, IPVT, WK2)

**INTEGER** NOBS, NODE, INIT, MAXIT, EPS, DENS, STAT, HESS, LDHESS, ILOHI, DENEST, B, IPVT, WK2

**REAL** ALPHA, EPS, X(K), BNDS(2), DENS(2), STAT(4), HESS(LDHESS+4), DENEST(NODE+3), B(K), WK2(K)

**INTEGER** I, IMPTR, IPTR, ITER, K, KM1, KM2, KPI, KP2, M, N, MOLD, MINCR(B)

**REAL** BK, BRK1, BSML1, CK, CKM1, CKM2, CKMC1, CKP1, CKP2, CONS, EPS1, FACTOR, FK, FKM1, FKM2, FKPI, FKPI1, H, M2, M3, SUM, TEMP, WK(4)

**DOUBLE PRECISION** SUM1, SUM2, SUM3

**INTEGER** MINCR(B)

**SAVE** MINCR

**SPECIFICATIONS FOR ARGUMENTS**

**SPECIFICATIONS FOR LOCAL VARIABLES**

**SPECIFICATIONS FOR SAVE VARIABLES**

**SPECIFICATIONS FOR INTRINSICS**
INTEGRAL, LOG, A, MAX, MIN, MOD, SORT

INTEGER MAX, MIN, MOD, SORT
REAL A, MAX, MOD

EXTERNAL EIMES, E1POP, E1PSH, E1S, EISTR, SADD, SATH, SCOPY, SPROD, SSUM, DSPT, LSTR, LF3R

EXTERNAL ISMIN, NRC, SDOT, SNULL, SSUM

DATA HINCR/5, 7, 17, 33, 65, 129, 253, 100001/

CALL E1PSH ('DSPL')

ERROR CHECKS

IF (NOBS LT 1) THEN
  PRINT ('After removing all missing (NaN; not a) number values from X there are no valid observations. At least one valid observation is necessary.')
END IF

IF (NODE LE 4) THEN
  CALL EISTR (1, NODE)
  CALL EIMES (5, 1, NODE = X(1)). The number of mesh nodes, NODE, must be an odd integer greater than 4.)
ELSE IF (.MOD(NODE-2, 2), EQ. 0) THEN
  CALL EISTR (1, NODE)
  CALL EIMES (5, 3, NODE = X(1)) must be an odd integer greater than 4.
END IF

IF (ALPHA LE 0.0) THEN
  CALL EISTR (1, ALPHAT)
  CALL EIMES (5, 4, ALPHA = X(R)). The penalty weightings, ALPHA, must be greater than 0.
END IF

IF (MAXIT LE 0.0) THEN
  CALL EISTR (1, MAXIT)
  CALL EIMES (5, 5, MAXIT = X(1)). The maximum number of iterations, MAXIT, must be greater than 0.
END IF

IF (BND(S1), .GT. BVNS(2)) THEN
  CALL EISTR (1, BND(S1))
  CALL EIMES (5, 2, X(R), BVNS(1), BVNS(2) = X(R)). The minimum value for X, BVNS(1), must be less than or equal to the maximum value for X, BVNS(2).
END IF

IF (.INIT NE 0) THEN
  IF (DENS(S1), .NE. 0 .OR. DENS(NODE), .NE. 0) THEN
    CALL EISTR (1, DENS(S1))
    CALL EIMES (5, 7, 'DENS(1) = X(R1) and DENS(NODE) = X(1)').
  ELSE
    END IF
  ELSE
    EIMES (5, 7, 'The initial estimates of the density must be zero.'
END IF

END IF
END IF
40 NOB1 = 0
DO 10 X = 1, NOBS
IF (X(I)).LT.BNDS(1) .OR. X(I).GT.BNDS(2) THEN
   NOB1 = NOB1 + 1
END IF
10 CONTINUE
IF (NOB1 .EQ. NOBS) THEN
   CALL ELMES (S, 9, 'All elements in X lie outside the '//'interval BNDS(1) to BNDS(2). At least one '//'element of X must lie in this interval.' )
END IF
IF (EPS .LE. 0.0) THEN
   EPS1 = 1.0E-4
ELSE
   EPS1 = EPS
END IF
IF (NRCO .NE. 0) GO TO 2000
C Initialization
IMPTR = 0
IF (INIT .EQ. 0) THEN
   DENS(1) = 0.0
   DENS(2) = 2.0/(BNDS(2) - BNDS(1))
   M = 3
ELSE
   H = NODE
   Refine mesh
C20 IF (INIT .EQ. 0) THEN
   MOLD = M
   IMPTR = IMPTR + 1
   M = MINO+NODE-MINCR(IMPTR)
END IF
C Get mesh interval width
M = (BNDS(2) - BNDS(1))/(M-1)
H2 = M/2
H3 = M/3
C Make initial DENS integrate to 1
IF (INIT .NE. 0) THEN
   S = SCAL (NODE, 1.0/SCALE(NODE, DENS(1)), DENS(1))
END IF
C Set mesh nodes
B(1) = BNDS(1)
DO 30, I = 2, M
   B(I) = B(I-1) + H
30 CONTINUE
C Set B indices for interpolating X
IPTR = 0
DO 40 X = 1, NOBS
IF (XI(IPTR)).LT.BNDS(1) GO TO 40
DO K = 1, M - 1
   ILOM(K+1) = IPTR
   ILOM(K+2) = IPTR + 1
   IF (IPTR .LE. NOBS) THEN
      IF (XI(IPTR)).LT.B(K+1) THEN
         ILOM(K+3) = -1
         IPTR = IPTR + 1
      END IF
      IF (IPTR .LE. NOBS) GO TO 50
50 CONTINUE
60 CONTINUE
I

C IF (INIT .EQ. 0) THEN  Via DESPT
C CALL DESPT (N-2, B(2), 1, MOLD, BNDS, DENS, DENSEST, JK, WK, 
C TEMP = 1.0/(MAMSH)
C DO 80 IZ2 = M - 1
C DENS(I) = AMAX1(TEMP, SORT(DENSEST(I-1, I)))
C 80 CONTINUE
C ELSE
C DO 90 IZ2 = M - 1
C DENS(I) = SORT(DENS(I))
C 90 CONTINUE
C END IF
C DENS(A) = 0.0
C 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 BSMALL = 0.0
C SUM = 0.0
C C DO 120 KZ2 = K - 1  CKx are true estimates = FKx2
C KM1 = K - 1
C KP2 = MAXO(1, K-2)
C KP2 = MINO(K, KP2)
C FK = DENS(K1)
C FKM1 = DENS(KM1)
C FKM2 = DENS(KM2)
C CF1 = FKM1**2
C CK = FK**2
C CPF1 = DENS(KP1)**2
C CPF2 = DENS(KP2)**2
C BK = B(K)
C BM1 = B(KM1)
C CM = CK + CK
C IF (KZ2 .GE. 1) HESS(1+K1) = 4.0*FK*KMN2*FACTOR
C SUM1 = 0.000
C SUM2 = 0.000
C SUM3 = 0.000
C DO 100 K=1 ILOM(K) = ILOM(K) + 1
C CONS = (1.0-TEMP)/ NCK*(CP1-CK)*TEMP
C SUM1 = SUM1 + CONS
C SUM2 = SUM2 + CONS*CONS
C 100 CONTINUE
C C C K = CM1
C DO 110 I = ILOM(KM1+1), ILOM(KM1, 2)
C CONS = (XI-I-1)*BM1
C TEMP = CKM1 + CKMCM1*CONS
C SUM1 = SUM1 + CONS/TEMP
C TEMP = TEMP*TEMP
C SUM2 = SUM2 + CONS*CONS/TEMP
C C 110 CONTINUE
C TEMP = FACTOR*(CM2+CKP2-4.0*(CM1+CKP1)+6.0*CK) + SUM1
C TEMP = 2.0*TEMP
C BSMALL = BSMALL + 2.0*CK*TEMP
HESS+3+KMI = TEMP + 0.0*STAT + 0.0*FACTOR + SUM2.

IF (K .NE. 2) HESS(2,KMI) = 4.0*FK*FKM1(-4)*FACT+3+SUM3
DENEST(KMI,1) = FK*TEMP
DENEST(KMI,2) = -2.0*FK

120 CONTINUE
BSMALL = 1.0/H - SUM + BSMALL
Save portion of DENEST
CALL SCOPY (M-2, DENEST(1,2), 1, DENEST(1,3), 1)
Finish with the hessian
CALL SADD (M-2, BSMALL/(2.0*SUM), HESS(1,1), LDHESS)
CALL SCOPY (M-4, HESS(1,3), LDHESS, HESS(2,2), LDHESS)
CALL SCOPY (M-3, HESS(1,4), LDHESS, HESS(3,3), LDHESS)
CALL SCOPY (M-2, HESS(4,3), LDHESS, HESS(4,4), LDHESS)

3 Solve symmetric band linear system
CALL LSQR (M-2, HESS, LDHESS, 2, 2, HESS, LDHESS, IPUT, W(2),
DENEST(1,2), 1,

6 IF (NIRC(1) .NE. 0) GO TO 3000
Compute the constant
CONS = SDOT(M-2, DENEST(1,3), 1, DENEST(1,2), 1)
CONS = (1.0/HR - SDOT(M-2, DENEST(1,3), 1, DENEST(1,2), 1))/CONS
Update the gradient
CALL SAXPY (M-2, CONS, DENEST(1,2), 1, DENEST(1,1), 1)
CALL SAXPY (M-2, 1.0, DENEST(1,1), 1, DENEST(1,1), 1)
Check the convergence criterion
TEMPC(SNRM2(M-2, DENS(2), 1)
IF (SNRM2(M-2, DENS(1) .LT. EPS*TEMPC) GO TO 150
Ad hoc projection to plus quadrant

DO 120 DEN(I) = AMAX1(TEMP, DENS(1))

130 CONTINUE
CALL EISS(I, 1, MAXIT)
CALL EINES (3, 1, 'The maximum number of iterations' '
150 IF (M .NE. NODE) GO TO 20
Replace DENS(2) with squares
IF (M .NE. NODE) GO TO 20

150 CALL SHPROD (M-2, DENS(2), 1, DENS(2), 1, DENS(2), 1)
Penalty
Evaluate log likelihood and penalty
SUM1 = 0.0

DO 160 K1 = 1, M
KMI = MAX0(K-1,1)
KP1 = 0
SUM1 = HESS(KMI+2,0)*DENS(K)+DENS(KP1))**2

160 CONTINUE
STAT(2) = -0.5*FACTOR*SUM1
SUM2 = 0.0

170 CONTINUE
180 DO 170 I1=1, M
DO 170 IF (X(I).GE.BNDS(1), .AND. X(I).LE.BNDS(2) THEN
CALL D2SPT (1, X(I), 1, NODE, BNDS, DENS, DEWEST, WK, WK,

30 SUM2 = SUM2 + ALOG(DEWEST(1,1))

END IF

170 CONTINUE

C Evaluate M.L.P.E. mean and variance
ORIGINAL PAGE IS OF POOR QUALITY
<table>
<thead>
<tr>
<th>OF CYCLES TO REACH MEAN FATIGUE STR</th>
<th>30 HPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESPL. PARAMETERS</td>
<td></td>
</tr>
<tr>
<td>B puppies (1), BNDs (2) = 0.573E+01</td>
<td>0.933E-01</td>
</tr>
<tr>
<td>PDF OF LOG OF CURRENT CYCLES, LOG XNM</td>
<td>7 AXIS OF PDF PLOT</td>
</tr>
<tr>
<td>0.000E+00</td>
<td>0.373E-01</td>
</tr>
<tr>
<td>0.114E+00</td>
<td>0.433E-01</td>
</tr>
<tr>
<td>0.490E+00</td>
<td>0.433E-01</td>
</tr>
<tr>
<td>0.254E+00</td>
<td>0.433E-01</td>
</tr>
<tr>
<td>0.000E+00</td>
<td>0.433E-01</td>
</tr>
<tr>
<td>OUTPUT STATISTICS</td>
<td></td>
</tr>
<tr>
<td>0.434E+02</td>
<td>0.130E+02</td>
</tr>
<tr>
<td>NUMBER OF MISSING VALUES</td>
<td></td>
</tr>
<tr>
<td>LOG OF CURRENT CYCLES, LOG XNM</td>
<td></td>
</tr>
<tr>
<td>0.572E+01</td>
<td>0.633E+01</td>
</tr>
<tr>
<td>0.728E+01</td>
<td>0.784E+01</td>
</tr>
<tr>
<td>0.844E+01</td>
<td>0.844E+01</td>
</tr>
<tr>
<td>0.944E+01</td>
<td>0.944E+01</td>
</tr>
<tr>
<td>ORDERED LOG OF CURRENT CYCLES, LOG XNM, Y AXIS PDF, CDF PLOT</td>
<td></td>
</tr>
</tbody>
</table>
File _DBAO: [IPLT2.CPR]1 (363, 178, 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 25 bytes.

8.0 APPENDIX C

RANDOM4 SAMPLE PROBLEM: SOURCE, INPUT AND OUTPUTFILES
LOGNORMAL FA TURE STRENGTH AT REFERENCE CONDITIONS: SIGO

WRITE(6,10051)ISEED,NTOT
READ(5,1006)X*X
WRITE(6,1006)X*X
X = 300.
Y = LOG(1.0+((X/XM)**2))
Y = LOG(YM) - 0.5*YS**2
CALL RNSET(ISEEED)
CALL RNRLN(NTOT,YM,YS,SIGO)
WRITE(6,1001)(SIGO(I),I=1,NTOT)
WRITE(6,1002)

2022 FORMAT('LOGNORMAL SIGO')
WRITE(6,1001)SIGO(I),I=1,NTOT
C LOGNORMAL LOG OF REFERENCE CYCLES >XLMN0
WRITE(6,1005)ISEED,NTOT
WRITE(6,1006)X*X
X = 7.
YS = SORT(LOG(1.0+(X/Y)**2))
Y = LOG(YM) - 0.5*YS**2
CALL RNSET(ISEEED)
CALL RNRLN(NTOT,YM,YS,XLMN0)
WRITE(6,1001)(XLMN0(I),I=1,NTOT)
WRITE(6,1003)

2033 FORMAT('LOGNORMAL XLMN0')
WRITE(6,1001)XLMN0(I),I=1,NTOT
C LOGNORMAL FA TURE STRENGTH AT CURRENT CONDITIONS: SIG
WRITE(6,1005)ISEED,NTOT
WRITE(6,1006)X*X
X = 250.
YS = SORT(LOG(1.0+(X/Y)**2))
Y = LOG(YM) - 0.5*YS**2.
CALL RNSET(ISEEED)
CALL RNRLN(NTOT,YM,YS,SIG)
WRITE(6,1001)(SIG(I),I=1,NTOT)
WRITE(6,1004)

2044 FORMAT('LOGNORMAL SIG')
WRITE(6,1001)SIG(I),I=1,NTOT
C DEFINE RANDOM STRESSES
C LOGNORMAL REFERENCE STRESSES: SIGO
WRITE(6,1005)ISEED,NTOT
WRITE(6,1006)X*X
X = 20.
YS = SORT(LOG(1.0+(X/Y)**2))
Y = LOG(YM) - 0.5*YS**2.
CALL RNSET(ISEEED)
CALL RNRLN(NTOT,YM,YS,SIG0)
C CHANGE SIGO TO NEGATIVE VALUES FOR COMPRESSIVE RESI DUAL STRESSES
DO 401 I = 1,NTOT
SIG0(I) = -SIG0(I)
401 CONTINUE
WRITE(6,1001)(SIG0(I),I=1,NTOT)
WRITE(6,1004)
C LOGNORMAL CURRENT STRESSES: SIG
WRITE(6,1005)ISEED,NTOT
I1

C CALCULATE PDF OF LOG OF CURRENT CYCLES, LOG XNM.
C USING THE MAXIMUM ENTROPY METHOD
C CALCULATE SAMPLE MOMENTS SM
C NUMBER OF MOMENTS, MM
3MM
C CALL SMOM(XNM,MM,NTOT,SM)
WRITE(30,1001)(SM(I),I=1,MM)
WRITE(6,2038)
C OBTAIN MAXIMUM ENTROPY DISTRIBUTION
KSTART=1
KDATA=1
C CALCULATE MAX AND MIN ORDINATES FOR PDF (AND CDF)
BNDS(1) = XNM(1) - 0.05*XNM(1)
BNDS(2) = XNM(NTOT) + 0.05*XNM(NTOT)
WRITE(6,8877) BNDS(1),BNDS(2)
WRITE(6,8877) BNDS(1),BNDS(2)
CALL ME1(MMM,SM,BNDS(1),BNDS(2),0,DPS,START,KDATA,AL,CUM)
WRITE(6,2039)
C CALCULATE VALUES OF ORDINATES FOR PDF (AND CDF)
C NUMBER OF ORDINATES USED
C CALCULATE WINDOW WIDTH, HH
HH=(BNDS(2)-BNDS(1))/(NODE-1)
C CALCULATE VALUES OF LOG OF CURRENT CYCLES AT WHICH PDF IS ESTIMATED;
C ALSO CALLED 'NODE' VALUES
DO 6001:I=1,NODE-2
BNDS(I)=BNDS(1)+(I*HH)
6001 CONTINUE
WRITE(6,983)
983 FORMAT('LOG OF CURRENT CYCLES, LOG XNM')
WRITE(6,1001)(BNDS(I),I=1,NODE)
C REORDER BNDS FOR PLOTTING
SAVE1=BNDS(2)
SAVE2=BNDS(NODE)
BNDS(NODE)=BNDS(2)
DO 6002:I=1,NODE-2
BNDS(I)=BNDS(I+2)
6002 CONTINUE
BNDS(NODE-1)=SAVE2
BNDS(NODE)=SAVE1
C FORMAT ORDERED LOG OF CURRENT CYCLES, LOG XNM, HH
1X axis PDF, CDF PLOT
DO 1001:I=1,NODE
C CALCULATE VALUES OF THE PDF AT EACH ORDNATE
C FOR 4 MOMENTS THERE ARE 3 LA GRANGIAN MULTIPLIERS
Ddens(I)=EXP(AL(I)+AL(2)*BNDS(I)+AL(3)*BNDS(I)**2)
DO 1002:I=1,NODE
1002 CONTINUE
C WRITE LOG OF CURRENT CYCLES AND PDF OF LOG OF CURRENT CYCLES.
CALCULATE CDF OF LOG OF CURRENT CYCLES
I OPT = 2

READ(3,1004) IOPT
WRITE(3,992)

992 FORMAT(' 3CDF PARAMETERS')
WRITE(6,1004) IOPT
10 = BMDS(1)
DO 5005, I = 1, NODE
F = BMDF(X0, IOPT, NODE, BMDS, DENS)
BMDS(I) = X0
X0 = X0 + FH
DISTX(I) = P.

5003 CONTINUE
WRITE(6,994)

994 FORMAT('CDF OF LOG OF CURRENT CYCLES: LOG XNM. 
1Y AXIS OF PDF, CDF PLOT')
WRITE(6,1001)(DISTX(I), I = 1, NODE)

WRITE (6,993)

993 FORMAT('ORDERED LOG OF CURRENT CYCLES: LOG XNM. 
1Y AXIS OF PDF, CDF PLOT')
WRITE(6,1001)(BMDS(I), I = 1, NODE)
WRITE(6,1001)(BMDS(I), I = 1, NODE)

5005 WRITE LOG OF CURRENT CYCLES AND CDF OF LOG OF CURRENT TO THE PLOT FILES

5006 WRITE(35,990)
WRITE(35,991)(BMDS(J), DISTX(J), J = 1, NODE)
STOP

END

SUBROUTINE SORT (Y,N)
DIMENSION Y(10000)

Y IS THE ARRAY TO BE SORTED

AT COMPLETION Y(I) IS SMALLEST VALUE

AT COMPLETION Y(N) IS LARGEST VALUE

N  = N - 1
DO 1 I = 1, N
1 Y(I+1) = Y(I)
DO 2 K = 1, N
IF (Y(I) LT Y(K)) GO TO 2
2 Y(I) = Y(K)
1 CONTINUE
2 CONTINUE
RETURN

SUBROUTINE SHD(X,M,NSAMP,SM)

CALCULATES SAMPLE CENTRAL MOMENTS

X(I) = SAMPLE VALUES; DIMENSION NSAMP

M = NUMBER OF MOMENTS DESIRED

NSAMP = SAMPLE SIZE

SM = VALUE OF MOMENTS; DIMENSION M

DIMENSION X(10000), SM(10)
SUBROUTINE MEPI(H,CM,XMIN,XMAX,NXP,KSTART,KDATA,AL,CUM)

EXECUTIVE PROGRAM FOR USING MAXIMUM ENTROPY METHOD CONstrained BY
MOMENTS TO GENERATE A DENSITY FUNCTION

DIMENSION AL(*), CM(*), ETA(*), XP(*), CUM(*), CC(3), ALC(10)
COMMON /FAIL/, NFAIL
COMMON/HELP/, KPRINT, TOL, MAXF, DATA, KPRINT, TOL, MAXF, N1, E-6, 70/
IF (N.EQ.1) KSTART=2

WRITE THE INPUT DATA
IF (KDATA.EQ.0) GO TO 1
WRITE (6,23) KDATA
WRITE (6,23) KDATA
WRITE (6,26) KPRINT
WRITE (6,28) N
WRITE (6,30) XMAX
WRITE (6,31) XMIN
WRITE (6,31) CM(I), I=1, N
IF (N.GT.4) WRITE (6,31) (CM(I), I=5,N)
WRITE (6,32) TOL
WRITE (6,33) NXP
1 CONTINUE
NFAIL=0
H=31
XMIN=0.0
XMAX=1.0
XMIN=0.0
WRITE CM
1 CONTINUE

CALL TRL1 (XMAX,XMIN,CC,XMAX,XMIN,CM)

CALCULATE THE MOMENTS ABOUT THE ORIGIN FOR THE MODIFIED LIMITS
STORE THEM IN COMMON IN C
CALL CONVER(CC,N)

GENERATE THE SIMPSON MULTIPLIERS AND STORE THEM IN HELP COMM

CALL SIMON

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

CALL MULTI-(XZMAX+XZMIN+N)

DEFINE THE INPUT DATA FOR SUBROUTINE MPOPT

ETA(1)=1.0-12
ETA(2)=10L
ETA(3)=1.0-24
ETA(4)=1.0-24
MODE=1

UMIN=0.0

WRITE THE INTERMEDIATE RESULTS YOU HAVE OBTAINED SOFar

IF (KPRINT.EQ.0) GO TO 2
WRITE (6,34)
WRITE (6,35) N
WRITE (6,36) XZMAX+XZMIN
WRITE (6,37) (CC(I),I=1,4)
WRITE (6,38) (CT(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (CC(I),I=5,N)
WRITE (6,39) (ETA(I),I=1,4)
CONTINUE

FIND A STARTING POINT FOR SUBROUTINE MPOPT TO START THE OPTIMIZAT-
ION ALGORITHM

IF (KSTART.EQ.0) GO TO 16
CALL START (XZMAX+XZMIN,AL,KSTART,CC,N,KPRINT,UMIN,MODE,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)
GO TO 3
WRITE (6,42)
WRITE (6,43) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
GO TO 4
WRITE (6,44)
WRITE (6,41) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
GO TO 5
WRITE (6,45)
WRITE (6,41) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
GO TO 6
WRITE (6,46)
WRITE (6,41) (AL(I),I=1,4)
IF (N.GT.4) WRITE (6,22) (AL(I),I=5,N)
GO TO 7
CONTINUE

CHANGE MAX-XMIN

C.... CHANGE STARTING VALUES TO 0-1 DOMAIN FOR KSTART=0
This algorithm is similar to the one put forward to give better numerical results.

1. IF A(j)(X(j)) < = XMIN LT. 1.E-10 GO TO 18
   DO 17 I = 2, NPL
   JS(I) = 0.0
   JS(I) = JS(I) + FACTO(J)*XMIN*(J-I)*RANGE*AL(J)*AL(I)*FACTO(J-I)
18 CONTINUE
17 CONTINUE
   GO TO 30
   DO 20 I = 2, NPL
   JS(I) = JS(I) + FACTO(J-I)*AL(I)
   JS = JS(I) IN PROPER LOCATIONS
50 IF JS = JS(I) THEN 70
   JS(I) = JS(I+1)
70 CONTINUE
   WRITE (8, 45)
8 CONTINUE
   AL(N+1) = JS(I)
   AL(N+2) = JS(I)
   CALL NROPL(AL, N+2, UMIN, MAXN, MOD, KPRINT)
   IF (NFAIL.EQ.0) GO TO 19
   IF (KSTART.EQ.1) GO TO 30
   THE PROGRAM HAS FAILED SO FAR, TRY ANOTHER STARTING POINT AND TRY AGAIN
KSTART = KSTART + 1
   IF (KSTART.EQ.1.AND. N.LE.2) GO TO 9
9 CONTINUE
   WRITE (6, 46)
10 CONTINUE
   CALL EXIT
11 CONTINUE
   CALL CALCULATE THE ZERO TH LAGRANGIAN MULTIPLIER
   SUM = 0.0
   DO 12 I = 1, N
     SUM = SUM + EXP(SZ)
   12 CONTINUE
NPL = N+1
   DO 13 I = 1, NPL
     KS = 0.0
     DO 11 K = 1, N
       KS = KS + AL(K)*XX(K, I)
     11 CONTINUE
     CONTINUE
   SUM = SUM + KS*EXP(SZ)
12 CONTINUE
   DELTA = (XMAX - XMIN)/FLOAT(N-1)
   AL(I) = AL(I) + DELTA*SUM
90 CONTINUE
   FORMAT(24H SUM OF RESIDUALS SQUARED = , E12.5)
   WRITE (6, 47) (AL(I), I = 1, NPL)
14 CONTINUE
C.... RESET KSTART TO ZERO
CALL TRN2 (XMAX,XMIN+AL,XMAX,XMIN+N)

CALCULATE THE CUMULATIVE DISTRIBUTION FUNCTION VALUE AT THE GIVEN POINT
IF(NXP.EQ.0)RETURN DO 15 I=1,NXP

15 CONTINUE
RETURN
C 01 FORMAT (5F7.4,E18.9) C 02 FORM FOR UDATA FOR SUBRUTINE MEP/-/20X,33(/17/)
C 03 FORMAT (' -INPUT DATA IS PRINTED OUT FOR KDATA =1 ONLY ----- KDATA =
C 04 1'.'X19/')
C 05 FORM (' INTERMEDIATE OUTPUT EVERY KPRINT(TH) CYCLE ------ KPRINT =
C 06 1'.'X19/')
C 07 FORM (' NUMBER OF KNOWN FIRST MOMENTS -------- N =
C 08 1'.'X19/')
C 09 FORM (' HIGHER LIMIT -------------- XMAX =
C 10 1'.'X19/')
C 11 FORM (' LOWER LIMIT -------------- XMIN =
C 12 FORM (' FIRST MOMENTS -------------- .CC(I) =
C 13 FORM (' THE ALLOWED TOLERANCE IN LAGRANIAN EQUATIONS ----.TOL =
C 14 1'.'X19/')
C 15 FORM (' THE CUMULATIVE DISTRIBUTION REQUIRED AT NXP POINTS,NXP =
C 16 1'.'X19/')
C 17 FORMAT (' INTERMEDIATE RESULTS FOR SUBRUTINE MEP/-/20X,33(/17/)
C 18 FORM (' NUMBER OF INTEGRATION STATION -------- M =
C 19 1'.'X19/')
C 20 FORM (' MODIFIED MAXIMUM AND MINIMUM LIMITS ------ X2MAX , X2MIN =
C 21 1'.'X19/')
C 22 FORM (' MODIFIED MOMENTS ABOUT THE EXPECTED VALUE .CC(I) =
C 23 1'.'X19/')
C 24 FORM (' MODIFIED MOMENTS ABOUT THE ORIGIN .C(I) =
C 25 1'.'X19/')
C 26 FORM (' SUBRUTINE MPOR TOLERANCES ----------- ETA(I) =
C 27 1'.'X19/')
C 28 FORM (' NORMAL ASSUMPTION STARTING METHOD'/34(1-)/)
C 29 FORM (' STARTING VALUES ------------------- .AL(I) =
C 30 1'.'X19/')
C 31 FORM (' UNIFORM ASSUMPTION STARTING METHOD'/35(1-)/)
C 32 FORM (' L-POINTS STARTING METHOD'/29(1-)/)
C 33 FORM (' STEP BY STEP STARTING METOD'/29(1-)/)
C 34 FORMAT (' CYC NMF NORMGRAD TOTAL =24X,'VARIABLES',40
C 35 X,'RESIDUALS',/ NO.,10X,'RESIDUALS X(1) X(2)
C 36 (2) R(3) R(4) R(5) R(6)
C 37 FORMAT (' THE PROGRAM HAS FAILED')
C 38 WRITE(6,40)
C 39 FORMAT (' WARNING MEAN IS NEARLY ZERO AND MPM WILL NOT WORK/12
C 40 IN TRANSFORM X')
END
SUBROUTINE OUTP (XNEW,F0,KOUNT,N1,GG,NUMF,R)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION XNEW(*), R(*)
WRITE (6,7) KOUNT,NUMF,GG,F0,(XNEW(I),I=1,4),(R(I),I=1,4)
IF (N1.LT.4) RETURN
5
RETURN.
WRITE (6,7) XNEW(5),R(5)
RETURN.
WRITE (6,8) (XNEW(I),I=1,6),R(I),I=1,6)
RETURN.
WRITE (6,9) (XNEW(I),I=1,7),R(I),I=1,7)
RETURN.
WRITE (6,10) (XNEW(I),I=1,8),R(I),I=1,8)
RETURN.
FORMAT (1X,I3:14.6E14.5,4E10.3)
FORMAT (36X,3E14.5,28X,2E10.3)
FORMAT (36X,3E14.5,14X,3E10.3)
FORMAT (36X,4E14.5,4E10.3)
END
SUBROUTINE LINES (FUNCT,X,H,AMBDA,N,F,G,NUMF,IER,EPS,EST,RR)
REAL*8 Z,DY,DX
COMMON /FAIL/ NFAIL
DIMENSION H(*), X(*), G(*), RR(*)
IER=0.
DY=0.
HNRM=0.
QNR=0.
DO 1 J=1,N
HNRM=HNRM+ABS(H(J))
QNR=QNR+ABS(G(J))
1 CONTINUE
IF (DY) 2,31,31
PRINT*, 'GOT BY B1'
IF (QNR) 3,1,31
PRINT*, 'GOT BY B2'
IF (HNRM) 3,1,31
PRINT*, 'GOT BY B3'
3 F=F
ALFA=2.* (EST-F)/DY
IF (X(N+1),G0.0) ALFA=X(N+1)*ALFA/2.
PRINT*, 'GOT BY B4'
CONTINUE
F=F
X1(N2)=X(N2)
X1(N1)=X(N1)
Y(N2)=X(N2)
Y(N1)=X(N1)
GO TO 3
AMBDAA = ALFA*4
IF (ALFA) 5,6,7
PRINT*, 'GOT BY B5'
IF (ALFA=AMBDAA) 5,6,7
PRINT*, 'GOT BY B6'
AMBDAA = ALFA
DO 8 I=1,N
X(I) = X(I) + AMBDAA*W(I)
PRINT*, 'GOT BY B7'
CONTINUE
7 F = F + D*O
PRINT*, 'GOT BY B8'
CALL FUNCT (N*X+F*G+RR)
PRINT*, 'GOT BY B9'
IF (NFAIL.EQ.1) RETURN
PRINT*, 'GOT BY B10'
NUM = NUM + 1
IF (F < LT) RETURN
PRINT*, 'GOT BY B11'
F = F + D*O
DO 9 I=1,N
DY(I) = W[I]*AMBDAA
PRINT*, 'GOT BY B12'
CONTINUE
8 PRINT*, 'GOT BY B13'
IF (DY) 10,30,13
PRINT*, 'GOT BY B14'
10 IF (FY-FX) 11,11,13
PRINT*, 'GOT BY B15'
AMBDAA = AMBDAA + ALFA
ALFA = AMBDAA
IF (NINH=AMBDAA-1.E10) 7,7,12
C PRINT*, 'GOT BY B16'
12 IER = 1
GO TO 31
C PRINT*, 'GOT BY B17'
13 T = 0
14 IF (AMBDAA) 15,90,15
PRINT*, 'GOT BY B18'
Z = Z + FX*W[I]/AMBDAA + D*O
ALFA = AMBDAA + W*O + DX + O
ALFA = W[O]*ALFA/ALFA
ALFA = W[O]*ALFA/DX + ALFA*O/ALFA
IF (D[O]) 11,20,6
C PRINT*, 'GOT BY B19'
IF (ALFA) 12,90,17
C PRINT*, 'GOT BY B20'
ALFA = W[I]*ALFA/ALFA
ALFA = W[I]*ALFA/ALFA
ALFA = W[I]*ALFA/ALFA
ALFA = W[I]*ALFA/ALFA
DO 20 I = 1,N
X(I) = X(I) + ALFA*W(I)
CONTINUE
20 CALL FUNCT (N*X+F*G+RR)
IF (NFAIL.EQ.1) RETURN
NUM = NUM + 1
IF (F < LT) GO TO 30
DO 10 J=1,N
SUM(J)=SUM(J)+XX(J-1)*S
PRINT*, 'GOT BY C4'
CONTINUE
DO 5 I=1,N
SUM(I)=SUM(I)/SUM(J)
PRINT*, 'GOT BY C5'
CONTINUE
U=0.0
DO 8 I=1,N
U=U+RR(I)/C(I)
PRINT*, 'GOT BY C6'
CONTINUE
DO 8 K=1,N
GRAD(K)=0.0
DO 7 J=1,N
GRAD(K)=GRAD(K)+(SUM(J+1)-SUM(J+1)*SUM(K+1)*RR(J)/C(J))
PRINT*, 'GOT BY C7'
CONTINUE
GRAD(K)=GRAD(K)*2,
PRINT*, 'GOT BY C8'
CONTINUE
PRINT*, 'GOT BY C9'
RETURN
PRINT*, 'GOT BY C10'
CONTINUE
M=92-2
ZERO=ZERO-AAA
GO TO 2
PRINT*, 'GOT BY C11'
END

SUBROUTINE START (XMAX,XMIN,ALAMDA,KSTART,CC,NL,IPRINT,UMIN,MODE+M,IAS,ETA)
IMPLICIT REAL*8 (A-H,O-Z)

THIS SUBROUTINE IS USED TO FIND A REASONABLE STARTING POINT FOR
SUBROUTINE HPMOPT

DIMENSION CC(10), ETA(10), ALAMDA(10), X(10), Y(10), W(10+10)

COMMON /FAIL/ MFAIL

1 CONTINUE
DO 2 I=1,N
2 CONTINUE RETURN
3 CONTINUE MFAIL=M
4 CONTINUE RETURN
5 CONTINUE
NF=10
NNN=NL/2
NNN=NNN*2
NP1=NL+1
DELTA=(XMAX-XMIN)/FLOAT(NL)
DO 6 I=1,NP1
7 CONTINUE
6 CONTINUE IF (NNN,NE,NL) GO TO 19
W(1,NP1)=1.
DO 7 J=2,NL,2
W(J,1)=W(J-1,1)*DELTA
7 CONTINUE IF (NL.EQ.2) GO TO 9
NNM=NL-1
DO 8 J=3,NNM,2
W(J,1)=W(J-1,1)*DELTA
8 CONTINUE DO 12 J=1,NP1
10 CONTINUE DO 10 I=1,NP1
11 CONTINUE M(I,1)=0.
12 CONTINUE Y(I)=0.
13 CONTINUE IF (Y(I),LE,0.0) Y(I)=.0001
14 CONTINUE DO 15 J=1,NP1
15 CONTINUE Y(I)=ALOG(Y(I))
16 CONTINUE DO 16 I=1,NP1
17 CONTINUE W(I,1)=W(I,1-1)*DELTA
18 CONTINUE ALAMBDA(I)=Y(I+1)
19 CONTINUE RETURN
20 CONTINUE R(4)=3./8.
21 CONTINUE R(3)=9./8.
22 CONTINUE R(3)=9./8.
23 CONTINUE IF (NL.EQ.3) GO TO 22
24 CONTINUE R(NL+1)=1./3.
25 CONTINUE R(4)=R(4)*1./3.
26 CONTINUE DO 20 I=2,NL
27 CONTINUE R(I)=R(I-1)*3.
28 CONTINUE IF (NL.EQ.5) GO TO 22
29 CONTINUE NS=NL-1
30 CONTINUE DO 21 I=6,NS+2
31 CONTINUE R(I)=R(I-2)/3.
CONTINUE
DO 23 I=1, NP1
W(I,I)=X(I)
CONTINUE
DO 24 J=1, NP1
DO 25 I=1, NP1
W(I,J)=W(I,J)*X(J)
Y(I,J)=Y(I,J)+X(J)
DO 25 J=1, NP1
CONTINUE
CALL SOLVE (W, Y, XID, NP1 * 10)
GO TO 12
CONTINUE
N=2
ALAMBA(2)=-.5/CC(2)
ALAMBA(1)=CC(1)/CC(2)
CONTINUE
ALAMBA(N+1)=2.0
ALAMBA(N+2)=0.0
C PRINT*,'GOTO BY 3'
CALL MPORT (ALAMBA, N, STA, UMIN, MAXFN, MODE, IPRINT)
C PRINT*,'GOTO BY 2'
IF (NFAIL.EQ.1) RETURN
ALAMBA(N+1)=0.0
N=N+1
GO TO 27
END
SUBROUTINE SOLVE (A, X, XID, N, NA)
IMPLICIT REAL*8 (A-H, Q-Z)
DIMENSION A(NA, X), X(N)
D=0.
DATA DIV, 693147181/0.
DO 4 I=1,N
AA=0.
DO 3 J=I,N
AB=AABS(A(I,J))
IF (AB.LE.AA) GO TO 1
AA=AB
CONTINUE
AA=AABS(AA)
IF (I.EQ.N) GO TO 7
IF (K.EQ.I) GO TO 3
DO 2 J=I,N
AB=AABS(A(I,J))
A(I,J)=AB
CONTINUE
AB=X(I)
X(I)=X(K)
X(K)=AB
CONTINUE
DO 3 J=1,N
AA=AABS(A(J,I))
A(J,I)=0.
CONTINUE
A(J,K)=AB(I,K)+AABS(A(I,K))
SUBROUTINE SIMSON
 IMPLICIT REAL*8 (A-H,O-Z)

 THIS SUBROUTINE IS TO CALCULATE THE SIMPSON MULTIPLIERS

COMMON/HELP/S(101),XX(16,101),C(8)*M

ABOVE LINE CHANGED FROM TEXT

S(1)=1.
N=M-1
DO 1 I=2,N+2
S(I)=4.
CONTINUE
N=N-1
DO 1 I=3,N+2
CONTINUE
END

SUBROUTINE MULTI (XMAX,XMIN,N)
 IMPLICIT REAL*8 (A-H,O-Z)

 THIS SUBROUTINE IS USED TO GENERATE THE X'S POWER FOR SUBROUTINE

COMMON/HELP/S(101),XX(16,101),C(8)*M

ABOVE LINE CHANGED FROM TEXT

DELTA=(XMAX-XMIN)/FLOAT(N-1)
DO 1 I=1,M
XX(I,1)=XMIN+FLOAT(I-1)*DELTA
CONTINUE
END

SUBROUTINE CMWHER (CM,NL)
 IMPLICIT REAL*8 (A-H,O-Z)

 THIS SUBROUTINE IS TO CALCULATE THE MOMENTS ABOUT THE ORIGIN

DIMENSION CM(*),
SUBROUTINE TRM1 (XMAX, XMIN, C, X2MAX, X2MIN, NL)
IMPLICIT REAL*8 (A-H, O-Z)

THIS SUBROUTINE IS USED TO CALCULATE THE MOMENTS FOR THE MODIFIED LIMITS

DIMENSION C(1), S
S = (XMAX - XMIN) / (X2MAX - X2MIN)
C(1) = C(1) / S
IF (NL .EQ. 1) RETURN
DO 1 K = 1, N
1 CONTINUE
RETURN
END

SUBROUTINE TRM2 (XMAX, XMIN, X, X2MAX, X2MIN, N)

DOUBLE PRECISION VERSION

DIMENSION X(N)

DO 3 I = 1, N
3 CONTINUE

IF (W .EQ. 1) GO TO 6
DO 5 J = 2, N
5 CONTINUE

DO 2 K = 1, N
2 CONTINUE

DO 1 I = 1, N
1 CONTINUE

DO 0 J = 1, N
0 CONTINUE

RETURN
END
CONTINUE
DO I=I1,IN
X(I)=OX(I)
CONTINUE
SUBROUTINE TRN2 (X1MAX,X1MIN,X2MAX,X2MIN,N)
IMPLICIT REAL*8 (A-H,O-Z)
THIS SUBROUTINE IS USED TO CALCULATE THE LAGRANGIAN MULTIPLIERS AT THE ORIGINAL LIMITS
DIMENSION X(N)
S=(X1MAX-X1MIN)/(X2MAX-X2MIN)
A=X1MIN-X1MIN/S
X(J)=X(J)-ALOG(S)
DO I=1,N
X(I)=X(I)**(1+X(I))**3
CONTINUE
IF (N.EQ.1) GO TO 5
DO 5 J=2,N
fact=1
K=L+2
DO 2 K=KK,1,-1
fact=fact**K
2 CONTINUE
X(J)=X(J)+FACT/J
CONTINUE
X(J)=X(J)**(J-1)
CONTINUE
X(N+1)=X(N+1)/S**N
RETURN
END
FUNCTION CDF (XMIN,XMAX,XPI,AL,N)
IMPLICIT REAL*8 (A-H,O-Z)
THIS FUNCTION SUBROUTINE IS TO CALCULATE THE CUMULATIVE DISTRIBUTION FUNCTION AT A GIVEN POINT
PARAMETER XMIN = LOWER QUADRANT XMAX = UPPER QUADRANT XP = SPECIFIED POINT AL(1) = ARRAY OF PARAMETERS, DIMENSION N N = NUMBER OF PARAMETERS
DIMENSION AL(N)
IF (XPI.LE.XMIN) GO TO 3
IF (XPI.GE.XMAX) GO TO 4
RANGE=XMAX-XMIN
RANGE=RANGE/51.
SS/=SS
SS=SS/2+X
AREA=0.0
JSM=SS-1
DELTA=RANGE/FLOAT(JSM1)
DO 3 J=1,JSM1-2
X=XMIN+DELT*(J-1)+DELTA
AREA=AREA+4.*ENTRF(AL,N,X)
3 CONTINUE
4 END
```
CONTINUE
JSM1=JSM1-1
DO 2 I=1,JSM1,Z
Z=XMIN+PAINT(I-1)*DELTA
AREA=AREA+2.*ENTRPF(AL+N,X)
2 CONTINUE
AREA=AREA+ENTRPF(AL+N,XMIN)+ENTRPF(AL+N,X)
CDF=AREA/CDF=1
GO TO 5
5 CONTINUE
RETURN
END

FUNCTION ENTRPF(AL,NPL,Z)
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(*)
DO 1 I=1,NPL
13=3+AL(I)*X
CONTINUE
ENTERF=ENTERF+EXP(3)
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
```
ORIGINAL PAGE IS OF POOR QUALITY
ORIGINAL PAGE IS
OF POOR QUALITY
### INPUT DATA FOR SUBROUTINE MEP:

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT DATA IS PRINTED OUT FOR A_DATA = 1 ONLY</td>
<td>A_DATA = 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 = 0.96377930E+01</td>
</tr>
<tr>
<td>LOWER LIMIT</td>
<td>XMIN = 0.572349819E+01</td>
</tr>
<tr>
<td>FIRST MOMENTS</td>
<td>C(i) = 0.7054816285E+01</td>
</tr>
<tr>
<td>THE ALLOWED TOLERANCE IN LAGRANGIAN EQUATIONS</td>
<td>TOL = 1.781e-03</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>CYC NUMF</th>
<th>NORMGRAD RESIDUALS</th>
<th>TOTAL X(1)</th>
<th>X(2)</th>
<th>VARIABLES X(3)</th>
<th>X(4)</th>
<th>R(1)</th>
<th>R(2)</th>
<th>RESIDUALS R(3)</th>
<th>R(4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>0.1760E+01</td>
<td>0.2371E-02</td>
<td>0.1122E+02</td>
<td>0.1240E+02</td>
<td>0.2382E+01</td>
<td>0.1976E+01</td>
<td>0.224E-01</td>
<td>0.210E-01</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0.3297E-02</td>
<td>0.2206E-02</td>
<td>0.1126E+02</td>
<td>0.1345E+02</td>
<td>0.3384E-01</td>
<td>0.1976E+01</td>
<td>0.234E-01</td>
<td>0.256E-01</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>0.3189E-02</td>
<td>0.9700E-04</td>
<td>0.8079E+01</td>
<td>0.1019E+02</td>
<td>0.7794E-01</td>
<td>0.1976E+01</td>
<td>0.234E-01</td>
<td>0.256E-01</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>0.2277E+02</td>
<td>0.3206E-02</td>
<td>0.1729E+02</td>
<td>0.4737E+02</td>
<td>0.4944E-02</td>
<td>0.1976E+01</td>
<td>0.234E-01</td>
<td>0.256E-01</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>0.1869E-04</td>
<td>0.2004E-02</td>
<td>0.5662E-02</td>
<td>0.5903E-02</td>
<td>0.3462E+02</td>
<td>0.200E-01</td>
<td>0.304E-01</td>
<td>0.304E-01</td>
</tr>
</tbody>
</table>

**NUMBER OF INTEGRATION STATION** M = 31
**MODIFIED MAXIMUM AND MINIMUM LIMITS** XMAX, XMIN = 0.100000000E+01 0.000000000E+00
**MODIFIED MOMENTS ABOUT THE EXPECTED VALUE** CC(1) = 0.41679124E+00 0.37229390E-01 0.3170796a1E-02 0.514753a5E-12
**MODIFIED MOMENTS ABOUT THE ORIGIN** C(1) = 0.41679124E+00 0.21091219E+00 0.12189717E+00 0.77157013E-11
**SUBROUTINE IMPORT TOLERANCES** ETA(1) = 0.100000000E-11 0.100000000E-05 0.100000000E-25 0.100000000E-23

**NORMAL ASSUMPTION STARTING METHOD**

**STARTING VALUES** AL(1) = 0.11199940E+02 -0.13402212E+02 0.100000000E+00 0.100000000E+00
File DBAG:C\JPLTZ.CPR:1 (JAG, 177.0), last revised on 29-NOV-1989 11:21, 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 25 bytes.


started on printer _JTPA_ on 29-NOV-1989 11:21 from queue _TPA_.

Original page is of poor quality.
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;