LOWER CONFIDENCE LIMITS FOR PROPORTION OF CONFORMANCE

Teresa Lam
Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, MI 48109-2117

C. Ming Wang
National Institute of Standards & Technology
Boulder, CO 80303

Technical Report 93-19

August 1993

Lower Confidence Limits for Proportion of Conformance

C. Teresa Lam

University of Michigan, Ann Arbor, MI 48109

C. Ming Wang

National Institute of Standards and Technology, Boulder, CO 80303

Abstract

A program that evaluates the uniformly minimum variance unbiased estimator, the maximum likelihood estimator, and an approximate lower confidence limit for the proportion of conformance is presented. Proportion of conformance is defined as the proportion of products with quality characteristic within the specification limits. In the case when the nominal value of the quality characteristic is not centered in the middle of the specification range, the program also computes the maximum likelihood estimator and the lower confidence limit for a modified proportion of conformance.

Dr. Lam is an Assistant Professor in the Industrial and Operations Engineering Department. She is a member of ASQC.

Dr. Wang is a Mathematical Statistician in the Statistical Engineering Division. He is a member of ASQC.

Introduction

The quality of a product can usually be quantified by certain observable characteristics of the product or the manufacturing process which produces the product. The performance of a product quality characteristic is specified by a nominal value T, a lower specification limit L and an upper specification limit U. To measure how well the product quality characteristic meets the specifications, proportion of conformance is commonly used. Proportion of conformance is defined as the proportion of products with quality characteristic inside the specification limits.

Point and interval estimation of the proportion of conformance were studied by Lam and Wang (1993). Specifically, under the assumption that a quality characteristic is normally distributed, they presented five different point estimators and compared their root mean squared errors. They recommended the use of the uniformly minimum variance unbiased estimator (UMVUE) or the maximum likelihood estimator (MLE) in practice. They also proposed two approximate lower confidence limits for the proportion of conformance for a normally distributed quality characteristic. In this paper, we present a computer program that evaluates the UMVUE, the MLE, and a lower confidence limit for the proportion of conformance. The confidence limit is based on equation (6) in Lam and Wang (1993).

As discussed in Lam and Wang (1993), proportion of conformance is maximized when the process mean of the normally distributed quality characteristic is in the middle of the specification range. However, in many assembly-fit processes, the nominal value of the quality characteristic may be off-center, indicating that a deviation in one direction is less acceptable than a deviation in the other. In this case, they suggested the use of a modified proportion of conformance (see also, Littig and Lam (1993)). The new measure is defined such that it is maximized when the process mean is at the nominal value. Other desirable properties are discussed in Lam and Wang (1993). The computer program presented in this paper evaluates the ML estimator and a lower confidence limit for the modified proportion of conformance.

Program Description

The program uses subroutine ZEROIN from Forsythe, Malcolm and Moler (1977) to solve for p_1^m and p_2^m (equations (8) and (9)) in Lam and Wang (1993). This routine employs an algorithm that combines the bisection with the secant and inverse quadratic interpolation methods to search for a zero of a nonlinear function. Subroutine TNC in Lenth (1989) is called to evaluate the noncentral t distribution.

As illustrated in Output Listing 1, the program first prompts the user for the sample size n, values of $K_1 = (\bar{X}-L)/S$, $K_2 = (U-\bar{X})/S$, the relative location $\rho = (U-T)/(T-L)$, and the desired confidence level γ, where $\bar{X}$ and S are the sample mean and sample standard deviation respectively. The program then prints the UMVU and ML estimates and the approximate lower $100\gamma\%$ confidence limit for p_C. In Output Listing 2, a nonsymmetric tolerance region ($\rho = 0.75$) was specified, the program prints the ML estimates (for both p_C and p_C^m) and the lower confidence limit for p_C^m.

Acknowledgment

Dr. Lam was visiting the National Institute of Standards and Technology under a joint ASA/NSF/NIST fellowship program while this research was carried out.

References

[1] Forsythe, G. E., Malcolm, M. A., and Moler, C. B. (1977). *Computer Methods for Mathematical Computations.* Prentice-Hall, Englewood Cliffs, NJ.

[2] Lam, C. T., and Wang, C. M. (1993). "On Estimation of Proportion of Conformance", submitted to *Journal of Quality Technology.*

[3] Lenth, R. V. (1989). "Cumulative Distribution Function of the Non-central t Distribution", Applied Statistics, 38, pp. 185-189.

[4] Littig, S. J., and Lam, C. T. (1993). "Case Studies in Process Capability Measurement", *ASQC: 47th Annual Quality Congress*, pp. 569-575.

Output Listing 1. Point estimates and confidence limit for p_C

```
 This program evaluates the uniformly minimum
 variance unbiased (UMVU) estimator,  maximum
 likelihhod (ML) estimator and an approximate
 lower limit for the (modified) proportion of
 conformance (PC).

Enter sample size n
30

Enter value of K1
2.4

Enter value of K2
3.0

Enter value of RHO (relative location of T)
1

Enter confidence level (e.g., 0.95)
0.95

UMVU estimate for PC          0.9935
ML estimate for PC            0.9915
Lower confidence limit        0.9519
```

Output Listing 2. Point estimates and confidence limit for p_C^m

```
Enter sample size n
30

Enter value of K1
2.4

Enter value of K2
3.0

Enter value of RHO (relative location of T)
0.75

Enter confidence level (e.g., 0.95)
0.95

ML estimate for PC          0.9915
ML estimate for Modified PC 0.9644
Lower confidence limit      0.8954
```

Program Listing

```
      PROGRAM CIMPC
C
C     Computes  MUVUE,  MLE and  one-sided lower
C     confidence limit for (modified) proportion
C     of conformance
C
      INTEGER NSIZE
      DOUBLE PRECISION XK1, XK2, ALPHA, RHO, LCLP
      DOUBLE PRECISION PUE, PML, PMML, PTMP, P1, P2
      DOUBLE PRECISION AR1, AR2, BIGP
      DOUBLE PRECISION SRN, RN2, RHOP1, RHOM1
      DOUBLE PRECISION ALNORM, TDF
C
      WRITE (*, 100)
      WRITE (*, '(A)') 'Enter sample size n'
      READ (*, *) NSIZE
      WRITE (*, *)
C
      WRITE (*, '(A)') 'Enter value of K1'
      READ (*, *) XK1
      WRITE (*, *)
      WRITE (*, '(A)') 'Enter value of K2'
      READ (*, *) XK2
      WRITE (*, *)
C
      WRITE (*, '(A)') 'Enter value of RHO'//
     &          ' (relative location of T)'
      READ (*, *) RHO
      WRITE (*, *)
C
      WRITE (*, '(A)') 'Enter confidence level'//
     &          ' (e.g., 0.95)'
      READ (*, *) ALPHA
      WRITE (*, *)
C
      RN2 = FLOAT(NSIZE - 2)
      SRN = SQRT(FLOAT(NSIZE))
      RHOP1 = 1.0D0 + RHO
      RHOM1 = 1.0D0 - RHO
C
C     MLE for modified p_C
C
      IF (RHO .NE. 1.0D0) THEN
         AR1 = SQRT((RN2 + 1.0D0)/(RN2 + 2.0))
         AR2 = (XK2 - RHO*XK1)/(RHOP1*AR1)
         IF (AR2 .GE. 0.0D0) THEN
            BIGP = MAX(1.0D0, 1.0D0/RHO)
            P1 = ALNORM(-XK1/(BIGP*AR1), .FALSE.)
            P2 = (2.0*XK2 + RHOM1*XK1)/RHOP1
            P2 = ALNORM(-P2/(BIGP*AR1), .FALSE.)
         ELSE
            BIGP = MAX(1.0D0, RHO)
            P1 = SRN*(2.0*RHO*XK1 - RHOM1*XK2)/RHOP1
            P1 = ALNORM(-P1/(BIGP*AR1), .FALSE.)
            P2 = ALNORM(-XK2/(BIGP*AR1), .FALSE.)
         END IF
         PMML = 1.0D0 - P1 - P2
      END IF
C
C     UMVUE and MLE for p_C
```

```
C
      AR1 = SQRT((RN2 + 1.0D0)/(RN2 + 2.0))
      PML = ALNORM(XK2/AR1, .FALSE.) -
     &      ALNORM(-XK1/AR1, .FALSE.)
      AR1 = (RN2 + 1.0D0)/SQRT(RN2 + 2.0)
      PUE = 1.0
      IF (ABS(XK2) .LE. AR1) THEN
          AR2 = XK2/AR1
          PUE = SQRT(RN2)*AR2/SQRT(1.0D0 - AR2**2)
          PUE = TDF(PUE, NSIZE-2)
      ELSE
          IF (XK2 .LT. -AR1) THEN
              PUE = 0.0
          END IF
      END IF
C
      PTMP = 1.0
      IF (ABS(XK1) .LE. AR1) THEN
          AR2 = -XK1/AR1
          PTMP = SQRT(RN2)*AR2/SQRT(1.0D0 - AR2**2)
          PTMP = TDF(PTMP, NSIZE-2)
      ELSE
          IF (-XK1 .LT. -AR1) THEN
              PTMP = 0.0
          END IF
      END IF
      PUE = PUE - PTMP
C
C     Lower confidence limit for modified p_C
C
      IF ((XK2 - RHO*XK1)/RHOP1 .GE. 0.0) THEN
           BIGP = MAX(1.0D0, 1.0D0/RHO)
           AR1 = SRN*XK1
           CALL GETP(NSIZE, AR1, ALPHA, P1)
           P1 = ALNORM(-P1/(BIGP*SRN), .FALSE.)
           AR2 = SRN*(2.0*XK2 + RHOM1*XK1)/RHOP1
           CALL GETP(NSIZE, AR2, ALPHA, P2)
           P2 = ALNORM(-P2/(BIGP*SRN), .FALSE.)
      ELSE
           BIGP = MAX(1.0D0, RHO)
           AR1 = SRN*(2.0*RHO*XK1 - RHOM1*XK2)/RHOP1
           CALL GETP(NSIZE, AR1, ALPHA, P1)
           P1 = ALNORM(-P1/(BIGP*SRN), .FALSE.)
           AR2 = SRN*XK2
           CALL GETP(NSIZE, AR2, ALPHA, P2)
           P2 = ALNORM(-P2/(BIGP*SRN), .FALSE.)
      END IF
C
      LCLP = MAX(1.0D0 - P1 - P2, 0.0D0)
C
      IF (.NOT. (RHO .NE. 1.0D0)) WRITE (*, 200) PUE
      WRITE (*, 300) PML
      IF (RHO .NE. 1.0D0) WRITE (*, 400) PMML
C
      WRITE (*, 500) LCLP
C
  100 FORMAT(//' This program evaluates the unifor',
     &'mly minimum'/' variance unbiased (UMVU)',
     &' estimator,  maximum'/' likelihhod (ML)',
     &' estimator and an approximate'/' lower',
     &' limit for the (modified) proportion of'/
     &' conformance (PC).'/)
  200 FORMAT ('UMVU estimate for PC', 7X, F7.4)
```

```
  300 FORMAT ('ML estimate for PC', 9X, F7.4)
  400 FORMAT ('ML estimate for Modified PC', F7.4)
  500 FORMAT ('Lower confidence limit', 5X, F7.4)
C
      STOP
      END
C
      SUBROUTINE GETP(N, ARG, PROB, POUT)
C
C     Solves for p_1 and p_2
C
      INTEGER N
      DOUBLE PRECISION ARG, PROB, POUT
C
      DOUBLE PRECISION XX1, XX2, XX3
      COMMON /CMN1/ XX1, XX2, XX3
C
      EXTERNAL FINT
      DOUBLE PRECISION ZEROIN, FINT
C
      DOUBLE PRECISION A, B, ABSERR
C
      XX1 = FLOAT(N - 1)
      XX2 = ARG
      XX3 = PROB
C
      A = -20000.0
      B =  20000.0
      ABSERR = 1.0E-5
C
      POUT = ZEROIN(A, B, FINT, ABSERR)
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION FINT(Z)
C
      DOUBLE PRECISION Z
C
      DOUBLE PRECISION XX1, XX2, XX3
      COMMON /CMN1/ XX1, XX2, XX3
C
      INTEGER IFAULT
      DOUBLE PRECISION TNC
C
      FINT = TNC(XX2, XX1, Z, IFAULT) - XX3
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION ALNORM(X, UPPER)
C
C     Evaluates the  tail area of the  standardized
C     normal curve  from X to infinity if  UPPER is
C     .TRUE.  or from minus  infinity to X if UPPER
C     is .FALSE.  Adapt from Algorithm AS66 Applied
C     Statist. (1973) VOL22 NO.3
C
      DOUBLE PRECISION ZERO, ONE, HALF
      DOUBLE PRECISION CON, Z, Y, X
      DOUBLE PRECISION P, Q, R, A1, A2, A3, B1, B2
      DOUBLE PRECISION C1, C2, C3, C4, C5, C6
      DOUBLE PRECISION D1, D2, D3, D4, D5
```

```
      DOUBLE PRECISION LTONE, UTZERO
      LOGICAL UPPER, UP
C
      DATA ZERO/0.0D0/, ONE/1.0D0/, HALF/0.5D0/
      DATA LTONE/7.0D0/, UTZERO/18.66D0/, CON/1.28D0/
      DATA P/0.398942280444D0/,   Q/0.39990348504D0/
      DATA R/0.398942280385D0/,   A1/5.75885480458D0/
      DATA A2/2.62433121679D0/,   A3/5.92885724438D0/
      DATA B1/-29.8213557807D0/,  B2/48.6959930692D0/
      DATA C1/-3.8052D-8/,        C2/3.98064794D-4/
      DATA C3/-0.151679116635D0/, C4/4.8385912808D0/
      DATA C5/0.742380924027D0/,  C6/3.99019417011D0/
      DATA D1/1.00000615302D0/,   D2/1.98615381364D0/
      DATA D3/5.29330324926D0/,   D4/-15.1508972451D0/
      DATA D5/30.789933034D0/
C
      UP = UPPER
      Z = X
      IF (Z. GE. ZERO) GOTO 100
      UP= .NOT. UP
      Z = -Z
  100 IF (Z .LE. LTONE .OR. UP .AND.
     &    Z .LE. UTZERO) GOTO 200
C
      ALNORM = ZERO
      GOTO 400
C
  200 Y = HALF*Z*Z
      IF (Z .GT. CON) GOTO 300
C
      ALNORM = HALF - Z*(P-Q*Y/(Y+A1+B1/
     &         (Y+A2+B2/(Y+A3))))
      GOTO 400
C
  300 ALNORM = R*EXP(-Y)/(Z+C1+D1/(Z+C2+D2/(Z+C3+D3/
     &         (Z+C4+D4/(Z+C5+D5/(Z+C6))))))
  400 IF( .NOT. UP) ALNORM = ONE - ALNORM
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION TNC(T, DF, DELTA, IFT)
C
C     Cumulative probability at T of the noncentral
C     t-distribution  with  DF degrees  of  freedom
C     (may be fractional) and non-centrality DELTA.
C     Adapt from  Algorithm AS 243 Applied Statist.
C     (1989), VOL.38, NO. 1
C
      INTEGER IFT
      DOUBLE PRECISION T, DF, DELTA
C
      DOUBLE PRECISION A, ALBETA, ALNRPI, B, DEL, EN
      DOUBLE PRECISION ERRBD, ERRMAX, GEVEN, GODD
      DOUBLE PRECISION HALF, ITRMAX, LAMBDA, ONE
      DOUBLE PRECISION P, Q, R2PI, RXB, S, TT
      DOUBLE PRECISION TWO, X, XEVEN, XODD, ZERO
      DOUBLE PRECISION ALNORM, BETAIN, DLNGAM
      LOGICAL NEGDEL
C
      DATA ITRMAX/10000.1/, ERRMAX/1.0D-06/
      DATA ZERO/0.0/, HALF/0.5/, ONE/1.0/, TWO/2.0/
      DATA R2PI/0.797884560803/
```

```
      DATA ALNRPI/0.572364942925/
C
      TNC = ZERO
      IFT = 2
      IF (DF .LE. ZERO) RETURN
      IFT = 0
C
      TT = T
      DEL = DELTA
      NEGDEL = .FALSE.
      IF (T .GE. ZERO) GO TO 100
      NEGDEL = .TRUE.
      TT = -TT
      DEL = -DEL
  100 CONTINUE
C
      EN = ONE
      X = T*T/(T*T + DF)
      IF (X .LE. ZERO) GO TO 300
      LAMBDA = DEL*DEL
      P = HALF*EXP(-HALF*LAMBDA)
      Q = R2PI*P*DEL
      S = HALF - P
      A = HALF
      B = HALF*DF
      RXB = (ONE - X)**B
      ALBETA = ALNRPI + DLNGAM(B, IFT) -
     &         DLNGAM(A + B, IFT)
      XODD = BETAIN(X, A, B, ALBETA, IFT)
      GODD = TWO*RXB*EXP(A*LOG(X) - ALBETA)
      XEVEN = ONE - RXB
      GEVEN = B*X*RXB
      TNC = P*XODD + Q*XEVEN
C
C     Repeat until convergence
C
  200 A = A + ONE
      XODD = XODD - GODD
      XEVEN = XEVEN - GEVEN
      GODD = GODD*X*(A + B - ONE)/A
      GEVEN = GEVEN*X*(A + B - HALF)/(A + HALF)
      P = P*LAMBDA/(TWO * EN)
      Q = Q*LAMBDA/(TWO * EN + ONE)
      S = S - P
      EN = EN + ONE
      TNC = TNC + P*XODD + Q*XEVEN
      ERRBD = TWO*S*(XODD - GODD)
      IF (ERRBD .GT. ERRMAX .AND. EN .LE. ITRMAX)
     &    GO TO 200
C
  300 IFT = 1
      IF (EN .GT. ITRMAX) RETURN
      IFT = 0
      TNC = TNC + ALNORM(DEL, .TRUE.)
      IF (NEGDEL) TNC = ONE - TNC
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION DLNGAM(XVALUE, IFAULT)
C
C     Calculation of the  logarithm  of  the gamma
C     function. Adapt from Algorithm AS245 Applied
```

```
C     Statist. (1989) VOL. 38, NO. 2
C
      INTEGER IFAULT
      DOUBLE PRECISION XVALUE
      DOUBLE PRECISION ALR2PI, FOUR, HALF, ONE, ONEP5
      DOUBLE PRECISION R1(9), R2(9), R3(9), R4(5)
      DOUBLE PRECISION TWELVE, X, X1, X2, XLGE, XLGST
      DOUBLE PRECISION Y, ZERO
C
C     Coefficients of rational functions
C
      DATA R1/-2.66685511495D0, -2.44387534237D1,
     &        -2.19698958928D1,  1.11667541262D1,
     &        3.13060547623D0,  6.07771387771D-1,
     &        1.19400905721D1,  3.14690115749D1,
     &        1.52346874070D1/
      DATA R2/-7.83359299449D1, -1.42046296688D2,
     &         1.37519416416D2,  7.86994924154D1,
     &         4.16438922228D0,  4.70668766060D1,
     &         3.13399215894D2,  2.63505074721D2,
     &         4.33400022514D1/
      DATA R3/-2.12159572323D5,  2.30661510616D5,
     &         2.74647644705D4, -4.02621119975D4,
     &        -2.29660729780D3, -1.16328495004D5,
     &        -1.46025937511D5, -2.42357409629D4,
     &        -5.70691009324D2/
      DATA R4/2.7919531791853D-1, 4.917317610506D-1,
     &        6.92910599291889D-2, 3.350343815022304,
     &        6.012459259764103D0/
C
C     Fixed constants
C
      DATA ALR2PI/9.18938533204673D-1/, FOUR/4.D0/
      DATA HALF/0.5D0/, ONE/1.D0/, ONEP5/1.5D0/
      DATA TWELVE/12.D0/, ZERO/0.D0/
C
      DATA XLGE/5.10D6/, XLGST/1.D+30/
C
      X = XVALUE
      DLNGAM = ZERO
C
C     Test for valid function argument
C
      IFAULT = 2
      IF (X .GE. XLGST) RETURN
      IFAULT = 1
      IF (X .LE. ZERO) RETURN
      IFAULT = 0
C
      IF (X .LT. ONEP5) THEN
         IF (X .LT. HALF) THEN
            DLNGAM = -LOG(X)
            Y = X + ONE
C
C     Test whether X < machine epsilon
C
            IF (Y .EQ. ONE) RETURN
         ELSE
            DLNGAM = ZERO
            Y = X
            X = (X - HALF) - HALF
         END IF
         DLNGAM = DLNGAM + X*((((R1(5)*Y + R1(4))*Y +
```

```
     &             R1(3))*Y + R1(2))*Y + R1(1))/((((Y +
     &             R1(9))*Y + R1(8))*Y + R1(7))*Y +
     &             R1(6))
         RETURN
      END IF
C
C     Calculation for 1.5 <= X < 4.0
C
      IF (X .LT. FOUR) THEN
        Y = (X - ONE) - ONE
        DLNGAM = Y*((((R2(5)*X + R2(4))*X + R2(3))*X +
     &           R2(2))*X + R2(1))/((((X + R2(9))*X +
     &           R2(8))*X + R2(7))*X + R2(6))
        RETURN
      END IF
C
C     Calculation for 4.0 <= X < 12.0
C
      IF (X .LT. TWELVE) THEN
        DLNGAM = ((((R3(5)*X + R3(4))*X + R3(3))*X +
     &           R3(2))*X + R3(1))/((((X + R3(9))*X +
     &           R3(8))*X + R3(7))*X + R3(6))
        RETURN
      END IF
C
C     Calculation for X >= 12.0
C
      Y = LOG(X)
      DLNGAM = X * (Y - ONE) - HALF * Y + ALR2PI
      IF (X .GT. XLGE) RETURN
      X1 = ONE / X
      X2 = X1 * X1
      DLNGAM = DLNGAM + X1 * ((R4(3)*X2 + R4(2))*X2 +
     &         R4(1))/((X2 + R4(5))*X2 + R4(4))
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION BETAIN(X, P, Q, BETA,
     &                                 IFAULT)
C
C     Computes  incomplete beta function ratio for
C     arguments  X  between zero and one,  P and Q
C     positive. Adapt from Algorithm AS 63 Applied
C     Statist. (1973), VOL.22, NO.3.
C
      INTEGER IFAULT
      DOUBLE PRECISION X, P, Q, BETA
C
      INTEGER NS
      DOUBLE PRECISION ZERO, ONE, ACU, PSQ, CX, XX
      DOUBLE PRECISION PP, QQ, TERM, AI, RX, TEMP
      LOGICAL INDX
C
      DATA ZERO/0.0D0/, ONE/1.0D0/, ACU/0.1D - 14/
C
      BETAIN = X
C
      IFAULT = 1
      IF (P .LE. ZERO .OR. Q .LE. ZERO) RETURN
      IFAULT = 2
      IF (X .LT. ZERO .OR. X .GT. ONE) RETURN
      IFAULT = 0
```

```
      IF (X .EQ. ZERO .OR. X .EQ. ONE) RETURN
C
C     change tail if necessary
C
      PSQ = P + Q
      CX = ONE - X
      IF (P .GE. PSQ*X) GOTO 100
      XX = CX
      CX = X
      PP = Q
      QQ = P
      INDX = .TRUE.
      GOTO 200
C
  100 XX = X
      PP = P
      QQ = Q
      INDX = .FALSE.
  200 TERM = ONE
      AI = ONE
      BETAIN = ONE
      NS = QQ + CX*PSQ
C
      RX = XX/CX
  300 TEMP = QQ - AI
      IF (NS .EQ. 0) RX = XX
  400 TERM = TERM*TEMP*RX/(PP + AI)
      BETAIN = BETAIN + TERM
      TEMP = ABS(TERM)
      IF (TEMP .LE. ACU .AND. TEMP .LE. ACU*BETAIN)
     &    GOTO 500
      AI = AI + ONE
      NS = NS - 1
      IF (NS .GE. 0) GOTO 300
      TEMP = PSQ
      PSQ = PSQ + ONE
      GOTO 400
C
  500 BETAIN = BETAIN*EXP(PP*LOG(XX) +
     &         (QQ - ONE)*LOG(CX) - BETA)/PP
      IF (INDX) BETAIN = ONE - BETAIN
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION TDF(X, NU)
C
C     Cumulative probability at X of the Student t
C     with NU degrees of freedom.
C
      INTEGER NU, NUCUT, I, IMAX, IMIN, IEVODD
      DOUBLE PRECISION X, DX, DNU, PI, C, CSQ, S, SUM
      DOUBLE PRECISION AI, TERM, DCONST, TERM1,TERM2
      DOUBLE PRECISION TERM3, DCDFN, DCDF, B11, B21
      DOUBLE PRECISION B22, B23, B24, B25, B31, B32
      DOUBLE PRECISION B33, B34, B35, B36, B37
      DOUBLE PRECISION D1, D3, D5, D7, D9, D11
      DOUBLE PRECISION ANU, SD, Z
      DOUBLE PRECISION ALNORM
C
      DATA NUCUT/1000/
      DATA PI/3.14159265358979D0/
      DATA DCONST/0.3989422804D0/
```

```
      DATA B11/0.25D0/
      DATA B21/0.01041666666667D0/
      DATA B22, B23/ 3.0D0, -7.0D0/
      DATA B24, B25/-5.0D0, -3.0D0/
      DATA B31/0.00260416666667D0/
      DATA B32, B33/1.0D0, -11.0D0/
      DATA B34, B35/14.0D0, 6.0D0/
      DATA B36, B37/-3.0D0, -15.0D0/
C
      IF (NU .LE. 0) THEN
         TDF = 0.0
         RETURN
      END IF
C
      DX = X
      ANU = NU
      DNU = NU
C
      IF (NU .LE. 2) GOTO 300
      SD = SQRT(ANU/(ANU - 2.0))
      Z = X/SD
      IF (NU .LT. 10 .AND. Z .LT. -3000.0D0) GOTO 100
      IF (NU .GE. 10 .AND. Z .LT. -150.0D0) GOTO 100
      IF (NU .LT. 10 .AND. Z .GT. 3000.0D0) GOTO 200
      IF (NU .GE. 10 .AND. Z .GT. 150.0D0) GOTO 200
      GOTO 300
C
  100 TDF = 0.0
      RETURN
  200 TDF = 1.0
      RETURN
  300 CONTINUE
C
      IF (NU .LT. NUCUT) GOTO 400
      GOTO 1000
C
  400 CONTINUE
      C = SQRT(DNU/(DX*DX + DNU))
      CSQ = DNU/(DX*DX + DNU)
      S = DX/SQRT(DX*DX + DNU)
      IMAX = NU - 2
      IEVODD = NU - 2*(NU/2)
      IF (IEVODD .EQ. 0) GOTO 500
C
      SUM = C
      IF (NU .EQ. 1) SUM = 0.0D0
      TERM = C
      IMIN = 3
      GOTO 600
C
  500 SUM = 1.0D0
      TERM = 1.0D0
      IMIN = 2
C
  600 IF (IMIN .GT. IMAX) GOTO 800
      DO 700 I = IMIN, IMAX, 2
         AI = I
         TERM = TERM*((AI - 1.0D0)/AI)*CSQ
         SUM = SUM + TERM
  700 CONTINUE
C
  800 SUM = SUM*S
      IF (IEVODD .EQ. 0) GOTO 900
```

```
      SUM = (2.0D0/PI)*(ATAN(DX/SQRT(DNU)) + SUM)
  900 TDF = 0.5D0 + SUM/2.0D0
      RETURN
C
 1000 CONTINUE
      DCDFN = ALNORM(X, .FALSE.)
      D1 = DX
      D3 = DX**3
      D5 = DX**5
      D7 = DX**7
      D9 = DX**9
      D11 = DX**11
      TERM1 = B11*(D3 + D1)/DNU
      TERM2 = B21*(B22*D7 + B23*D5 + B24*D3 +
     &        B25*D1)/(DNU**2)
      TERM3 = B31*(B32*D11 + B33*D9 + B34*D7 +
     &        B35*D5 + B36*D3 + B37*D1)/(DNU**3)
      DCDF = TERM1 + TERM2 + TERM3
      DCDF = DCDFN - (DCONST*(EXP(-DX*DX/2.0D0)))*DCDF
      TDF = DCDF
C
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION ZEROIN(AX, BX, F, TOL)
C
C     Computes a zero of the function F(X) in the
C     interval (AX, BX) with  TOL  as the desired
C     length of the  interval  of  uncertainty of
C     the final result.  Adapt from  "Computer
C     Methods for Mathematical Computations"
C
      DOUBLE PRECISION AX, BX, F, TOL
      DOUBLE PRECISION A, B, C, D, E, EPS, FA, FB, FC
      DOUBLE PRECISION TOL1, XM, P, Q, R, S
C
C     Compute EPS, the relative machine precision
C
      EPS = 1.0D0
  100 EPS = EPS/2.0D0
      TOL1 = 1.0D0 + EPS
      IF (TOL1 .GT. 1.0D0) GO TO 100
C
      A = AX
      B = BX
      FA = F(A)
      FB = F(B)
C
  200 C = A
      FC = FA
      D = B - A
      E = D
  300 IF (ABS(FC) .GE. ABS(FB)) GO TO 400
      A = B
      B = C
      C = A
      FA = FB
      FB = FC
      FC = FA
C
C     Convergence test
C
  400 TOL1 = 2.0D0*EPS*ABS(B) + 0.5D0*TOL
```

```
      XM = .5*(C - B)
      IF (ABS(XM) .LE. TOL1) GO TO 900
      IF (FB .EQ. 0.0D0) GO TO 900
C
C     is bisection necessary
C
      IF (ABS(E) .LT. TOL1) GO TO 700
      IF (ABS(FA) .LE. ABS(FB)) GO TO 700
C
C     Is quardratic interpolation possible
C
      IF (A .NE. C) GO TO 500
C
C     linear interpolation
C
      S = FB/FA
      P = 2.0D0*XM*S
      Q = 1.0D0 - S
      GO TO 600
C
C     Inverse quadratic interpolation
C
  500 Q = FA/FC
      R = FB/FC
      S = FB/FA
      P = S*(2.0D0*XM*Q*(Q - R) - (B - A)*(R - 1.0D0))
      Q = (Q - 1.0D0)*(R - 1.0D0)*(S - 1.0D0)
C
C     adjust sign
C
  600 IF (P .GT. 0.0D0) Q = -Q
      P = ABS(P)
C
C     is interpolation acceptable
C
      IF ((2.0D0*P) .GE. (3.0D0*XM*Q - ABS(TOL1*Q)))
     &     GO TO 700
      IF (P .GE. ABS(0.5D0*E*Q)) GO TO 700
      E = D
      D = P/Q
      GO TO 800
C
C     Bisection
C
  700 D = XM
      E = D
C
C     Complete step
C
  800 A = B
      FA = FB
      IF (ABS(D) .GT. TOL1) B = B + D
      IF (ABS(D) .LE. TOL1) B = B + SIGN(TOL1, XM)
      FB = F(B)
      IF ((FB*(FC/ABS(FC))) .GT. 0.0D0) GO TO 200
      GO TO 300
C
C     Done
C
  900 ZEROIN = B
C
      RETURN
      END
```