National Bureau of Standards
Statistical Engineering Division
Gaithersburg, MD 20899

N-86-4
November 1986

An Algorithm for Computing the Doubly Noncentral F C.D.F. to a Specified Accuracy

Charles P. Reeve

Let X_1 and X_2 be independent noncentral chi-squared random variables with degrees of freedom ν_1 and ν_2 (both >0) and noncentrality parameters λ_1 and λ_2 (both $\geq$0) respectively. Then the random variable

$$Y = (X_1/\nu_1)/(X_2/\nu_2) \qquad (1)$$

is said to have the doubly noncentral F distribution, indicated by $Y \sim F''(\nu_1,\nu_2,\lambda_1,\lambda_2)$. This distribution has been used in the evaluation of the power function of analysis of variance tests in which interaction or bias effects occur, as in Scheffe´ [11]. Numerical examples of this usage are given in Bulgren [2] and Tiku [13]. It has also been used in engineering problems in the context of information theory as discussed in Price [8].

Exact formulas for the F" cumulative distribution function (c.d.f.) are given in Tiku [12] and [2] using the beta c.d.f., in Tiku [14] using Laguerre polynomials, and in [8] for special cases of ν_1 and ν_2. Approximations are given in Johnson and Kotz [4] and Tiku [12,13] using the central F c.d.f., and in Mudholkar, Chaubey, and Lin [5] using Edgeworth series expansions.

The author has been unable to find any published algorithms for computing exact values of the F" c.d.f. although computer programs have obviously been used in generating published tables in [2,5,12,14]. The purpose of this note is to present an efficient algorithm for computing the F" c.d.f. to a specified accuracy using exact formulas.

The algorithm uses the series representation in eq. (2.2) of [2] which, unfortunately, contains typographical errors. With these corrected the c.d.f. of Y can be re-written

$$F_Y(x) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} A_i B_j I(u,\nu_1/2+i,\nu_2/2+j) \qquad (2)$$

where $A_i = (\lambda_1/2)^i e^{-\lambda_1/2}/\Gamma(i+1)$, $B_j = (\lambda_2/2)^j e^{-\lambda_2/2}/\Gamma(j+1)$, $u = \nu_1 x/(\nu_1 x+\nu_2)$, and $x\geq0$. The A_i and B_j are Poisson probabilities, and

$I(u,a,b) = \int_0^u t^{a-1}(1-t)^{b-1}dt/B(a,b)$ is the c.d.f. of the beta distribution (also called the incomplete beta ratio) where $0\leq u\leq 1$, a>0, b>0, and

$B(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}dt$. For computational purposes the two infinite series must be truncated, thus (2) is re-expressed as

$$F_Y(x) = \sum_{i=I'}^{I''} \sum_{j=J'}^{J''} A_i B_j I(u,\nu_1/2+i,\nu_2/2+j) + R \qquad (3)$$

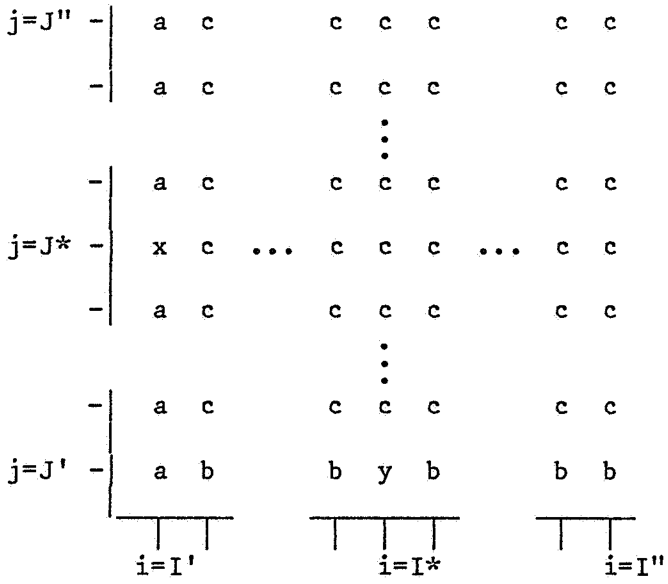
where I', I", J', and J" are non-negative integers and R is the remainder. If the beta c.d.f.'s are computed without error, it can easily be shown that choosing I', I", J', and J" such that $\sum_{i=I'}^{I''} A_i > 1-\varepsilon/2$ and $\sum_{j=J'}^{J''} B_j > 1-\varepsilon/2$ yields $R<\varepsilon$ provided $\varepsilon>0$. Therefore, ε serves as an absolute error bound on $F_Y(x)$. For maximum computational efficiency, the number of terms in each sum is minimized by indexing i and j over the largest of the Poisson probabilities A_i and B_j respectively. The final task is to compute the $(I''-I'+1)(J''-J'+1)$ beta c.d.f.'s and the double summation. An efficient procedure for doing this is to compute only $I(u,\nu_1/2+I',\nu_2/2+J^*)$ and $I(u,\nu_1/2+I^*,\nu_2/2+J')$ directly, indicated by the symbols "x" and "y" in figure 1. The remaining beta c.d.f's are computed using the recurrence relations

$$I(x,a,b) = I(x,a,b+1) - x^a(1-x)^b/[bB(a,b)], \qquad (4a)$$

$$I(x,a,b) = I(x,a+1,b) + x^a(1-x)^b/[aB(a,b)], \text{ and} \qquad (4b)$$

$$I(x,a,b) = xI(x,a-1,b) + (1-x)I(x,a,b-1) \qquad (4c)$$

as found in Abramowitz and Stegun [1]. Subject to the restrictions $J'<J^*<J''$



<u>Figure 1</u>

Computation of the $I(u,\nu_1/2+i,\nu_2/2+j)$

and $I'<I^*<I''$, J^* and I^* are chosen to maximize the magnitudes of the rightmost terms in (4a) and (4b) respectively. In applying each of these two recurrence relations only one direct evaluation of $B(a,b)$ is necessary, computed by

$B(a,b) = e^{\ln\Gamma(a) + \ln\Gamma(b) - \ln\Gamma(a+b)}$ where $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1}e^{-t}dt$. Further values are easily computed from the identities $B(a+1,b) = aB(a,b)/(a+b)$ and $B(a,b+1) = bB(a,b)/(a+b)$.

In figure 1 the symbols "a", "b", and "c" indicate which of the recurrence relations (4) is used in computing each beta c.d.f., with those indicated by "c" being done last.

The above algorithm has been incorporated into the FORTRAN subroutine CDFDNF. If $x<0$ then 0 is returned. External routines for computing the beta c.d.f. and the double precision log of the gamma function are required. In the current version of CDFNDF these routines are the subroutine CDFBET and the function GAMLOG as described in Reeve [9,10]. Other routines which can be substituted for CDFBET and GAMLOG are those in [3,6,7]. For all practical purposes the absolute error criterion ε will be met if the beta c.d.f. routine is accurate to two or three digits beyond ε.

The recursive method of computing the beta c.d.f.'s requires a little extra computer programming and storage, but results in a tremendous savings in computing time as λ_1 and λ_2 become large. This is illustrated in table 1 where, for selected parameter values, the CPU time for this method is compared with that of the "brute force" approach in which each beta c.d.f. is directly evaluated by a subroutine call. The computing time and storage also increase as ε becomes small, but are unaffected by ν_1 and ν_2. The computations in table 1 were done on the CDC Cyber 180/855 computer at NBS.

The current dimension limits in CDFDNF allow values of both λ_1 and λ_2 up to about 10,000 with ε as small as 10^{-8}, but these limits could easily be increased by the user. The limiting factor in using CDFDNF is more likely to be execution time than storage.

Steps were taken to eliminate underflow situations, minimize the effects of roundoff error, and minimize storage requirements. Only those c.d.f. values indicated by "x", "y", "a", or "b" in figure 1 are actually stored. The Poisson probabilities $A_{I'}, \ldots, A_{I''}$ and $B_{J'}, \ldots, B_{j''}$ are also stored.

If $\lambda_2=0$ then the doubly noncentral F reduces to the (singly) noncentral F, and if $\lambda_1=\lambda_2=0$ it reduces to the central F. In either case, CDFDNF will run almost as efficiently as routines designed for those specific cases.

Portions of tables in [2,5,14] were reproduced by CDFDNF and agreed to within roundoff error in each case.

A listing of CDFDNF is an appendix to this note. It is invoked by

```
CALL CDFDNF(X,DF1,DF2,ALAMB1,ALAMB2,EPS,IFLAG,CDFX)
```

where the arguments are defined in the program documentation. The returned value of CDFX is valid only if IFLAG=0 on return. In passing ε (variable name EPS) to CDFDNF the user should realize that accuracy is limited by the number of digits carried in a single precision variable, and that roundoff error may affect the last one or two of these digits.

Table 1

Computing times on the CDC Cyber 180/855 for the c.d.f. of $F''(\nu_1,\nu_2,\lambda_1,\lambda_2)$ using both CDFDNF and the "brute force" method (described in the text), for selected parameter values.

		$\varepsilon=10^{-6}$				CPU seconds		No. beta
ν_1	ν_2	λ_1	λ_2	x	$P\{F''<x\}$	CDFDNF	"Brute Force"	c.d.f. values
3	3	5	5	2.0	0.757918	<0.01	0.13	196
3	3	5	25	2.0	0.997561	<0.01	0.30	476
3	3	25	5	2.0	0.190910	<0.01	0.31	476
3	3	25	25	2.0	0.897835	<0.01	0.69	1,156
3	10	5	5	2.0	0.593795	<0.01	0.14	196
3	10	5	25	2.0	0.943093	<0.01	0.30	476
3	10	25	5	2.0	0.026209	<0.01	0.28	476
3	10	25	25	2.0	0.289601	<0.01	0.73	1,156
10	3	5	5	2.0	0.898330	<0.01	0.12	196
10	3	5	25	2.0	0.999879	<0.01	0.26	476
10	3	25	5	2.0	0.657879	<0.01	0.29	476
10	3	25	25	2.0	0.997703	<0.01	0.66	1,156
10	10	5	5	2.0	0.868071	<0.01	0.14	196
10	10	5	25	2.0	0.998234	<0.01	0.30	476
10	10	25	5	2.0	0.367101	<0.01	0.32	476
10	10	25	25	2.0	0.934321	<0.01	0.71	1,156
14	15	80	80	1.1	0.552328	0.03	2.55	3,969
14	15	400	400	1.1	0.582507	0.12	16.28	20,164
14	15	2000	2000	1.1	0.664981	0.59	108.10	101,124
14	15	10000	10000	1.1	0.825080	2.68	≈800	505,521
† 14	15	50000	50000	1.1	0.981351	13.76	-	2,528,100

† required a doubling of current dimension limits

References

1. Abramowitz, Milton and Stegun, Irene A., Handbook of Mathematical Functions, NBS Special Publication 55, 1970, p. 944.

2. Bulgren, W.G., "On Representations of the Doubly Non-Central F Distribution", Journal of the American Statistical Association, Vol. 66, No. 333, March 1971, pp. 184-6.

3. IMSL, Inc., Houston, TX. [MDBETA, DLGAMA]

4. Johnson, Norman L. and Kotz, Samuel, Continuous Univariate Distributions-2, Houghton Mifflin Company, 1970, pp. 189-200.

5. Mudholkar, Govind S., Chaubey, Yogendra P., and Lin, Ching-Chuong, "Approximations for the Doubly Noncentral-F Distribution", Communications in Statistics - Theory and Methods, Vol. A5, No. 1, 1976, pp.49-63.

6. NBS Core Math Library (CMLIB). [BETAI, DLNGAM]

7. Numerical Algorithms Group (NAG), Downers Grove, IL. [G01BDE, S14ABF]

8. Price, Robert, "Some Noncentral F Distributions Expressed in Closed Form", Biometrika, Vol. 51, 1964, pp. 107-22.

9. Reeve, Charles P., "An Algorithm for Computing the Beta C.D.F. to a Specified Accuracy", SED Note 86-3, October 1986.

10. Reeve, Charles P., "Accurate Computation of the Log of the Gamma Function", SED Note 86-1, October 1986.

11. Scheffe´, Henry, The Analysis of Variance, John Wiley & Sons, Inc., 1959, pp. 134-6.

12. Tiku, M.L., "Series Expansions of the doubly Noncentral F Distributions", Australian Journal of Statistics, Vol. 7, 1965, pp. 78-89.

13. Tiku, M.L., "A Note on Approximating to the Doubly Noncentral F Distribution", Australian Journal of Statistics, Vol. 14, 1972, pp. 37-40.

14. Tiku, M.L., "Doubly Noncentral F Distribution - Tables and Applications", Selected Tables in Mathematical Statistics, Vol. II, 1974, pp. 139-49.

```
*CDFDNF
      SUBROUTINE CDFDNF (X,DF1,DF2,ALAMB1,ALAMB2,EPS,IFLAG,CDFX)
C
C-----------------------------------------------------------------------
C   CDFDNF    WRITTEN BY CHARLES P. REEVE, STATISTICAL ENGINEERING
C             DIVISION, NATIONAL BUREAU OF STANDARDS, GAITHERSBURG,
C             MARYLAND  20899
C
C   FOR: COMPUTING THE CUMULATIVE DISTRIBUTION FUNCTION OF THE DOUBLY
C        NONCENTRAL F DISTRIBUTION TO A SPECIFIED ACCURACY (TRUNCATION
C        ERROR IN THE INFINITE SERIES REPRESENTATION GIVEN BY EQUATION
C        2.2 IN REFERENCE 1 BELOW).  THE BETA C.D.F. ROUTINE IS CALLED
C        AT MOST TWO TIMES.  FURTHER VALUES OF THE BETA C.D.F. ARE
C        OBTAINED FROM RECURRENCE RELATIONS GIVEN IN REFERENCE 2.
C        REFERENCE 3 GIVES A DETAILED DESCRIPTION OF THE ALGORITHM
C        HEREIN.
C
C        THIS PROGRAM MAY ALSO BE EFFICIENTLY USED TO COMPUTE THE
C        CUMULATIVE DISTRIBUTION FUNCTIONS OF THE SINGLY NONCENTRAL
C        AND CENTRAL F DISTRIBUTIONS BY SETTING THE APPROPRIATE
C        NONCENTRALITY PARAMETERS EQUAL TO ZERO.
C
C        CHECKS ARE MADE TO ASSURE THAT ALL PASSED PARAMETERS ARE
C        WITHIN VALID RANGES AS GIVEN BELOW.  NO UPPER LIMIT IS SET
C        FOR THE NONCENTRALITY PARAMETERS, BUT VALUES UP TO ABOUT
C        10,000 CAN BE HANDLED WITH THE CURRENT DIMENSION LIMITS.  THE
C        COMPUTED VALUE CDFX IS VALID ONLY IF IFLAG=0 ON RETURN.
C
C   NOTE: IN EQUATION 2.2 OF REFERENCE 1 THE AUTHOR HAS MISTAKENLY
C         REVERSED THE ARGUMENTS OF THE INCOMPLETE BETA FUNCTION.
C         THEY SHOULD READ [(M/2)+R,(N/2+S)] WHERE M AND N ARE THE
C         DEGREES OF FREEDOM ASSOCIATED WITH THE NUMERATOR AND
C         DENOMINATOR RESPECTIVELY OF THE F STATISTIC.  TO FURTHER
C         CONFUSE THE ISSUE, THE AUTHOR HAS REVERSED THE USAGE OF
C         M AND N IN SECTION 1 OF THE PAPER.
C
C   NOTE: IN SUBROUTINE EDGEF THE DOUBLE PRECISION CONSTANT DEUFLO IS
C         THE EXPONENTIAL UNDERFLOW LIMIT WHOSE CURRENT VALUE IS SET
C         AT -69D0.  ON A COMPUTER WHERE DEXP(-69D0) CAUSES UNDERFLOW
C         THIS LIMIT SHOULD BE CHANGED.
C
C   SUBPROGRAMS CALLED: CDFBET (BETA C.D.F.)
C                       GAMLOG (DOUBLE PRECISION LOG OF GAMMA FUNCTION)
C                       POISSF, EDGEF (ATTACHED)
C
C   CURRENT VERSION COMPLETED SEPTEMBER 29, 1988
C
C   REFERENCES:
C
C   1. BULGREN, W.G., 'ON REPRESENTATIONS OF THE DOUBLY NONCENTRAL F
C      DISTRIBUTION', JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION,
C      MARCH 1971, VOLUME 66, NO. 333, PP. 184-186.
C
C   2. ABRAMOWITZ, MILTON, AND STEGUN, IRENE A., 'HANDBOOK OF
C      MATHEMATICAL FUNCTIONS', NATIONAL BUREAU OF STANDARDS APPLIED
C      MATHEMATICS SERIES 55, NOVEMBER 1970, P. 944.
C
C   3. REEVE, CHARLES P., 'AN ALGORITHM FOR COMPUTING THE DOUBLY
C      NONCENTRAL F C.D.F. TO A SPECIFIED ACCURACY', STATISTICAL
C      ENGINEERING DIVISION NOTE 86-4, NOVEMBER 1986.
C-----------------------------------------------------------------------
C   DEFINITION OF PASSED PARAMETERS:
C
C        * X = VALUE (>=0) AT WHICH THE C.D.F. IS TO BE COMPUTED (REAL)
C
C      * DF1 = DEGREES OF FREEDOM (>0) IN THE NUMERATOR (REAL)
C
C      * DF2 = DEGREES OF FREEDOM (>0) IN THE DENOMINATOR (REAL)
C
C   * ALAMB1 = THE NONCENTRALITY PARAMETER (>=0) FOR THE NUMERATOR
C              (REAL) [EQUAL TO ZERO FOR THE CENTRAL F DISTRIBUTION]
C
C   * ALAMB2 = THE NONCENTRALITY PARAMETER (>=0) FOR THE DENOMINATOR
C              (REAL) [EQUAL TO ZERO FOR THE SINGLY NONCENTRAL F AND
C              CENTRAL F DISTRIBUTIONS]
C
C      * EPS = THE DESIRED ABSOLUTE ACCURACY OF THE C.D.F. (REAL)
C              [1 >= EPS >= 10**(-10)]
C
C      IFLAG = ERROR INDICATOR ON OUTPUT (INTEGER)   INTERPRETATION:
```

```
C                 0 -> NO ERRORS DETECTED
C               1,2 -> ERROR FLAGS FROM SUBROUTINE CDFBET
C                 3 -> EITHER ALAMB1 OR ALAMB2 IS < 0
C                 4 -> EITHER DF1 OR DF2 IS <= 0
C                 5 -> EPS IS OUTSIDE THE RANGE [10**(-10),1]
C                 6 -> VECTOR DIMENSIONS ARE TOO SMALL - INCREASE NX
C
C        CDFX = THE DOUBLY NONCENTRAL F C.D.F. EVALUATED AT X (REAL)
C
C    * INDICATES PARAMETERS REQUIRING INPUT VALUES
C-----------------------------------------------------------------------
C
      PARAMETER (NX=1000)
      DIMENSION BFI(NX),BFJ(NX),POI(NX),POJ(NX)
      CDFX = 0.0
C
C---- CHECK VALIDITY OF ARGUMENTS
C
      IF (ALAMB1.LT.0.0.OR.ALAMB2.LT.0.0) THEN
         IFLAG = 3
         RETURN
      ENDIF
      IF (DF1.LE.0.0.OR.DF2.LE.0.0) THEN
         IFLAG = 4
         RETURN
      ENDIF
      IF (EPS.GT.1.0.OR.EPS.LT.1.0E-10) THEN
         IFLAG = 5
         RETURN
      ENDIF
      IFLAG = 0
C
C---- SET ERROR CRITERION FOR THE BETA C.D.F. (PECULIAR TO CDFBET)
C
      EPS3 = 0.001*EPS
C
      FA = 0.5*ALAMB1
      GA = 0.5*ALAMB2
      FB = 0.5*DF1
      GB = 0.5*DF2
      YY = DF2/(DF2+DF1*X)
      IF (YY.GE.1.0) RETURN
      XX = 1.0-YY
      IF (XX.GE.1.0) THEN
         CDFX = 1.0
         RETURN
      ENDIF
C
C---- COMPUTE POISSON PROBABILITIES IN VECTORS POI AND POJ
C
      CALL POISSF (FA,EPS,IMIN,NI,POI,NX,IFLAG)
      IF (IFLAG.NE.0) RETURN
      FC = FB+REAL(IMIN)
      CALL POISSF (GA,EPS,JMIN,NJ,POJ,NX,IFLAG)
      IF (IFLAG.NE.0) RETURN
      GC = GB+REAL(JMIN)
C
C---- COMPUTE BETA C.D.F. BY RECURRENCE WHEN I=IMIN AND J=JMIN TO JMAX
C
      CALL EDGEF (NJ,GC,FC,YY,XX,BFJ,CDFX,POJ,POI,EPS3,IFLAG,1)
      IF (NI.LE.1.OR.IFLAG.NE.0) RETURN
C
C---- COMPUTE BETA C.D.F. BY RECURRENCE WHEN J=JMIN AND I=IMIN TO IMAX
C
      BFI(1) = BFJ(1)
      CALL EDGEF (NI,FC,GC,XX,YY,BFI,CDFX,POI,POJ,EPS3,IFLAG,2)
      IF (NJ.LE.1.OR.IFLAG.NE.0) RETURN
C
C---- COMPUTE BETA C.D.F. BY RECURRENCE WHEN I>IMIN AND J>JMIN
C
      DO 20 I = 2, NI
         BFJ(1) = BFI(I)
         DO 10 J = 2, NJ
            BFJ(J) = XX*BFJ(J)+YY*BFJ(J-1)
            CDFX = CDFX+POI(I)*POJ(J)*BFJ(J)
   10    CONTINUE
   20 CONTINUE
      RETURN
      END
C
```

```
      SUBROUTINE POISSF (ALAMB,EPS,L,NSPAN,V,NV,IFLAG)
C
C---- COMPUTE THE POISSON(ALAMB) PROBABILITIES OVER THE RANGE [L,K]
C---- WHERE THE TOTAL TAIL PROBABILITY IS LESS THAN EPS/2, SUM THE
C---- PROBABILITIES IN DOUBLE PRECISION, AND SHIFT THEM TO THE
C---- BEGINNING OF VECTOR V.
C
      DIMENSION V(*)
      DOUBLE PRECISION DAL,DK,DLIMIT,DSUM,GAMLOG
      DLIMIT = 1.0D0-0.5D0*DBLE(EPS)
      K = INT(ALAMB)
      L = K+1
      IF (ALAMB.EQ.0.0) THEN
         PL = 1.0
      ELSE
         DAL = DBLE(ALAMB)
         DK = DBLE(K)
         PL = SNGL(DEXP(DK*DLOG(DAL)-DAL-GAMLOG(REAL(K+1))))
      ENDIF
      PK = ALAMB*PL/REAL(L)
      NK = NV/2
      NL = NK+1
      DSUM = 0.0
   10 IF (PL.LT.PK) THEN
         NK = NK+1
         IF (NK.GT.NV) THEN
            IFLAG = 6
            RETURN
         ENDIF
         V(NK) = PK
         DSUM = DSUM+DBLE(PK)
         K = K+1
         IF (DSUM.GE.DLIMIT) GO TO 20
         PK = ALAMB*PK/REAL(K+1)
      ELSE
         NL = NL-1
         V(NL) = PL
         DSUM = DSUM+DBLE(PL)
         L = L-1
         IF (DSUM.GE.DLIMIT) GO TO 20
         PL = REAL(L)*PL/ALAMB
      ENDIF
      GO TO 10
   20 INC = NL-1
      DO 30 I = NL, NK
         V(I-INC) = V(I)
   30 CONTINUE
      NSPAN = NK-INC
      RETURN
      END
C
      SUBROUTINE EDGEF (NK,FC,GC,XX,YY,BFK,CDFX,POI,POJ,EPS3,IFLAG,L)
C
C---- COMPUTE THE BETA C.D.F.'S BY A RECURRENCE RELATION ALONG THE EDGES
C---- I = IMIN AND J = JMIN OF A GRID.  THE CORRESPONDING COMPONENTS OF
C---- THE F" C.D.F. ARE INCLUDED IN THE SUMMATION.  TERMS WHICH MIGHT
C---- CAUSE UNDERFLOW ARE SET TO ZERO.
C
      DIMENSION BFK(*),POI(*),POJ(*)
      DOUBLE PRECISION DARG,DEUFLO,GAMLOG
      DATA DEUFLO / -69.0D0 /
      FD = FC-1.0
      K = MAX0(L,MIN0(NK,INT((GC-1.0)*XX/YY-FD)))
      FK = FD+REAL(K)
      CALL CDFBET (XX,FK,GC,EPS3,IFLAG,BFK(K))
      IF (IFLAG.NE.0) RETURN
      IF (L.EQ.1) BFK(K) = 1.0-BFK(K)
      IF (NK.EQ.1) GO TO 40
      DARG = DBLE(FK)*DLOG(DBLE(XX))+DBLE(GC)*DLOG(DBLE(YY))-
     *   DLOG(DBLE(FK))+GAMLOG(FK+GC)-GAMLOG(FK)-GAMLOG(GC)
      IF (DARG.LT.DEUFLO) THEN
         DK = 0.0
      ELSE
         DK = SNGL(DEXP(DARG))*(-1.0)**L
      ENDIF
      IF (K.GE.NK) GO TO 20
      BFK(K+1) = BFK(K)-DK
      DI = DK
      KFLAG = 1
      DO 10 I = K+1, NK-1
```

```
         IF (KFLAG.EQ.1) THEN
            DI = DI*(FD+GC+REAL(I-1))*XX/(FD+REAL(I))
            IF (DK+DI.EQ.DK) THEN
               KFLAG = 0
               DI = 0.0
            ENDIF
         ENDIF
         BFK(I+1) = BFK(I)-DI
   10 CONTINUE
   20 DI = DK
      KFLAG = 1
      DO 30 I = K-1, L, -1
         IF (KFLAG.EQ.1) THEN
            DI = DI*(FC+REAL(I))/((FD+GC+REAL(I))*XX)
            IF (DK+DI.EQ.DK) THEN
               KFLAG = 0
               DI = 0.0
            ENDIF
         ENDIF
         BFK(I) = BFK(I+1)+DI
   30 CONTINUE
   40 DO 50 I = L, NK
         CDFX = CDFX+POI(I)*POJ(1)*BFK(I)
   50 CONTINUE
      RETURN
      END
```