JOURNAL OF RESEARCH of the National Bureau of Standards — B. Mathematical Sciences
Vol. 77B, Nos. 3&4, July–December 1973

Certification of an Algorithm for Bessel Functions of Real Argument

David J. Sookne*

Institute for Basic Standards, National Bureau of Standards, Washington, D.C. 20234

(June 4, 1973)

The accuracy of routine BESLRI is certified.

Key words: Bessel functions; bit comparison; error bounds.

1. Introduction

Bessel function values $I_n(x)$ and $J_n(x)$ (x real), generated by BESLRI [1],[1] were compared with check values. For $|x| < 64$, these check values were calculated via the ascending series

$$J_n(x) = \left(\frac{x}{2}\right)^n \sum_{k=0}^{\infty} \frac{(-x^2/4)^k}{k!(n+k)!}, \qquad I_n(x) = \left(\frac{x}{2}\right)^n \sum_{k=0}^{\infty} \frac{(x^2/4)^k}{k!(n+k)!},$$

using multiprecision arithmetic [2]. For $x \geqslant 64$, the asymptotic formulas for large argument were used:

$$I_n(x) \sim \frac{e^x}{\sqrt{2\pi x}} \left(1 - \frac{(\mu-1)}{8x} + \frac{(\mu-1)(\mu-9)}{2!(8x)^2} - \frac{(\mu-1)(\mu-9)(\mu-25)}{3!(8x)^3} + \cdots \right)$$

where $\mu = 4n^2$, and error is controlled by 7.16 of [3], and

$$J_n(x) = \sqrt{\frac{2}{\pi x}} \{P(n, x) \cos \chi - Q(n, x) \sin \chi\}$$

where $\chi = x - \left(\frac{n}{2} + \frac{1}{4}\right)\pi$, and

$$P(n, x) \sim 1 - \frac{(\mu-1)(\mu-9)}{2!(8x)^2} + \frac{(\mu-1)(\mu-9)(\mu-25)(\mu-49)}{4!(8x)^4} + \cdots$$

$$Q(n, x) \sim \frac{\mu-1}{8x} - \frac{(\mu-1)(\mu-9)(\mu-25)}{3!(8x)^3} + \cdots,$$

AMS Subject Classification: 68A10.

*Present address: W.U.J.S. Institute, Arad, Israel.

[1] Figures in brackets indicate the literature references at the end of this paper.

with errors bounded by the first neglected term [4, 9.2.9 and 9.2.10]. These were programmed in double precision. For $x \leq -64$, $-x$ was used as the argument, and the sign of the function value was determined from the equations

$$J_n(-x) = (-1)^n J_n(x), \qquad I_n(-x) = (-1)^n I_n(x).$$

The method used was that of bit comparison; the difference between the 60-bit mantissas of test and check values was expressed as a multiple, m, say, of the last bit. Such a bit error corresponds to a relative error between $m \cdot 2^{-60}$ and $m \cdot 2^{-59}$.

This test is too strict near a zero of $J_n(x)$, where absolute error is a more realistic measure. Thus for $n < |x|$, both mantissas were right-shifted so that the bit error described above was a multiple of the 60th binary place. These cases are given above the dashed line in the computer printout; statistics for this test are compiled separately.

A bit comparison test was also used to check the section of the code involving the two-term ascending series for small $|x|$.

The error return feature was tested exactly as in [5].

2. The Bit Comparison Test

To test the function $I_n(x)$, ten arguments were chosen in each interval

$$A_j = \{x : 2^{j-1} \leq |x| < 2^j\}, \qquad j = -13(1)10,$$

the 60-bit mantissas of each x being produced by a pseudorandom number generator [5]. The orders used in all cases were $n = 0(1)15$. The only modification occurred in the set A_{10}, for which $|x|$ was limited to 700 to avoid computer overflow in the final answers. Summary statistics for the 60-bit test on I_n are:

373 results correct to 60 bits
583 results in error by 1 multiple of the last bit
383 results in error by 2 multiples of the last bit
288 results in error by 3 multiples of the last bit
232 results in error by 4 multiples of the last bit
196 results in error by 5 multiples of the last bit
176 results in error by 6 multiples of the last bit
130 results in error by 7 multiples of the last bit
1479 results in error by more than 7 multiples of the last bit

3840 (total)

The largest bit error was $(36)_8$, implying that at least 55 bits are correct in each case.

For the function $J_n(x)$, the set of intervals A_j was expanded to $j = -13(1)16$. Again, ten pseudorandom values were used in each interval, and the orders used were 0(1)15. Summaries for the tests are:

Test on 60 Significant Bits $|x| < 64$

403 results correct to 60 significant bits
643 results in error by 1 multiple of the last bit
453 results in error by 2 multiples of the last bit
331 results in error by 3 multiples of the last bit

232 results in error by 4 multiples of the last bit
171 results in error by 5 multiples of the last bit
109 results in error by 6 multiples of the last bit
68 results in error by 7 multiples of the last bit
82 results in error by more than 7 multiples of the last bit

2492 (total)

The largest bit error was $(15)_8$, showing that at least 56 bits are correct in each case.

Test on 60 Binary Places $|x| < 64$

161 results correct to 60 places
200 results in error by 1 multiple of the last place
100 results in error by 2 multiples of the last place
85 results in error by 3 multiples of the last place
56 results in error by 4 multiples of the last place
46 results in error by 5 multiples of the last place
45 results in error by 6 multiples of the last place
14 results in error by 7 multiples of the last place
1 result in error by more than 7 multiples of the last place.

708 (total)

The largest bit error was $(10)_8$, showing that 56 binary places are correct in each case, and 57 places in all but one case.

In the test on 60 binary places for $|x| \geq 64$, the largest error was $(422)_8$. This occurred in $J_5(x)$ and $J_7(x)$ for $x = 59766.6 \ldots$. Bearing in mind that about 60,000 back-recursion steps were executed, that each step included two multiplications and one addition, and also that the computer uses truncation in floating-point arithmetic rather than roundoff, we see that the error of $(422)_8 = 274$ is quite reasonable. The other errors can be explained similarly.

For the test of the two-term ascending series, arguments of 2^{-16} and 2^{-100} were used; the former is just barely small enough for the truncation error to be neglected. Both $I_n(x)$ and $J_n(x)$ were tested with $n = 0(1)15$, and the largest error was 6 multiples of the last bit.

3. Testing of Error Return

The method used is the same as in [5]. Arguments used were $x = 2^k$ where $k = -13(1)9$ for I's, and $k = -13(1)13$ for J's. In the test on Bessel function values of order NCALC–1, the largest error was 6 multiples of the last bit; most were zero.

In the test on order $[x]$, the largest error was $(46)_8$, which occurred in $J_{8192}(8192)$. This test value involved about 1600 more back-recursion steps than were required for the check value, so this error is quite reasonable. The other errors were $(35)_8$ for $x = 4096$, zero for $x = 2048$, $(12)_8$ for $x = 1024$; for $x \leq 512$, the errors did not exceed 5.

4. Summary and Conclusions

For $n = 0(1)15$ and the values of x tested, BESLRI yielded $J_n(x)$ and $I_n(x)$ with either a relative error or an absolute error bounded by 10^{-14}. The latter assessment applied to $J_n(x)$ when $n < |x|$. For $|x| < 64$, the bound was 10^{-16}. The error return and ascending series features were tested and found to be executed accurately in all cases.

For other values of n and x, similar accuracy can be expected, since the tests we have used include large and small values of $|x|$ and $\frac{|x|}{n}$, the former as large as 2^{16}.

Calculations were performed on a UNIVAC 1108 under EXEC 2. All test values were calculated with double precision mantissa of 60 bits, corresponding to just over 18 figures. Subroutine executions took approximately 0.2 N milliseconds, where

$$N = \max\ (|x|,\ \mathrm{NB}) + 10.$$

5. Appendix: Algorithm BESLRI

```
@ ELT BESLCI,1,730112, 51748

      SUBRØUTINE BESLCI(X,Y,NB,IZE,BR,BI,NCALC)
C THIS RØUTINE CALCULATES BESSEL FUNCTIØNS I AND J ØF
C CØMPLEX ARGUMENT AND INTEGER ØRDER.
C
C
C       EXPLANATIØN ØF VARIABLES IN THE CALLING SEQUENCE
C
C X      DØUBLE PRECISIØN REAL PART ØF THE CØMPLEX ARGUMENT
C        FØR WHICH I*S ØR J*S ARE TØ BE CALCULATED.  IF I*S
C        ARE TØ BE CALCULATED, ABS(X) MUST NØT EXCEED EXPARG
C        (WHICH SEE BELØW).
C Y      IMAGINARY PART ØF THE ARGUMENT.  IF J*S ARE TØ BE
C        CALCULATED, ABS(Y) MUST NØT EXCEED EXPARG.
C NB     INTEGER TYPE.  1 + HIGHEST ØRDER TØ BE CALCULATED.
C        IT MUST BE PØSITIVE.
C IZE    INTEGER TYPE.  ZERØ IF J*S ARE TØ BE CALCULATED, 1
C        IF I*S ARE TØ BE CALCULATED.
C BR     DØUBLE PRECISIØN VECTØR ØF LENGTH NB, NEED NØT BE
C        INITIALIZED BY USER.  IF THE RØUTINE TERMINATES
C        NØRMALLY, (NCALC=NB), IT RETURNS THE REAL PART ØF
C        J(ØR I)-SUB-ZERØ THRØUGH J(ØR I)-SUB-NB-MINUS-ØNE
C        ØF Z IN THIS VECTØR.
C BI     IMAGINARY ANALØG ØF BR.
C NCALC INTEGER TYPE, NEED NØT BE INITIALIZED BY USER.
C        BEFØRE USING THE RESULTS, THE USER SHØULD CHECK THAT
C        NCALC=NB, I.E. ALL ØRDERS HAVE BEEN CALCULATED TØ
C        THE DESIRED ACCURACY.  SEE ERRØR RETURNS BELØW.
C
C
C        EXPLANATIØN ØF MACHINE-DEPENDENT CØNSTANTS
C
C NSIG   DECIMAL SIGNIFICANCE DESIRED.  SHØULD BE SET TØ
C        IFIX(ALØG10(2)*NBIT+1), WHERE NBIT IS THE NUMBER ØF
C        BITS IN THE MANTISSA ØF A DØUBLE PRECISIØN VARIABLE.
C        SETTING NSIG HIGHER WILL INCREASE CPU TIME WITHØUT
C        INCREASING ACCURACY, WHILE SETTING NSIG LØWER WILL
C        DECREASE ACCURACY.  IF ØNLY SINGLE-PRECISIØN
C        ACCURACY IS DESIRED, REPLACE NBIT BY THE NUMBER ØF
C        BITS IN THE MANTISSA ØF A SINGLE-PRECISIØN VARIABLE.
C        THE RELATIVE TRUNCATIØN ERRØR IS LIMITED TØ T=.5*10
C        **-NSIG FØR ØRDER GREATER THAN ABS(Z), AND FØR ØRDER
C        LESS THAN ABS(Z) (GENERAL TEST), THE RELATIVE ERRØR
C        IS LIMITED TØ T FØR FUNCTIØN VALUES ØF MAGNITUDE AT
C        LEAST 1, AND THE ABSØLUTE ERRØR IS LIMITED TØ T FØR
C        SMALLER VALUES.
C NTEN   LARGEST INTEGER K SUCH THAT 10**K IS MACHINE-
C        REPRESENTABLE IN DØUBLE PRECISIØN.
C LARGEZ UPPER LIMIT ØN THE MAGNITUDE ØF Z.  BEAR IN MIND
C        THAT IF ABS(Z)=N, THEN AT LEAST N ITERATIØNS ØF THE
C        BACKWARD RECURSIØN WILL BE EXECUTED.
C EXPARG LARGEST DØUBLE PRECISIØN ARGUMENT THAT THE LIBRARY
C        DEXP RØUTINE CAN HANDLE.
C
C
```

J.Res. 77B3&4-386

```
C                            ERRØR RETURNS
C
C       LET G DENØTE EITHER I ØR J.
C       IN CASE ØF AN ERRØR, NCALC.NE.NB, AND NØT ALL G*S
C  ARE CALCULATED TØ THE DESIRED ACCURACY.
C       IF NCALC.LT.0, AN ARGUMENT IS ØUT ØF RANGE.  NB.LE.0
C  ØR IZE IS NEITHER 0 NØR 1 ØR IZE=0 AND ABS(Y).GT.EXPARG,
C  ØR IZE=1 AND ABS(X).GT.EXPARG.  IN THIS CASE, THE VECTØRS
C  BR AND BI ARE NØT CALCULATED, AND NCALC IS SET TØ
C  MINO(NB,0)-1 SØ NCALC.NE.NB.
C       NB.GT.NCALC.GT.0 WILL ØCCUR IF NB.GT.MAGZ AND ABS(G-
C  SUB-NB-ØF-Z/G-SUB-MAGX-ØF-Z).LT.10.**(NTEN/2), I. E. NB
C  IS MUCH GREATER THAN MAGZ.  IN THIS CASE, BR(N) AND BI(N)
C  ARE CALCULATED TØ THE DESIRED ACCURACY FØR N.LE.NCALC,
C  BUT FØR NCALC.LT.N.LE.NB, PRECISIØN IS LØST.  IF N.GT.
C  NCALC AND ABS(G(NCALC-1)/G(N-1)).EQ.10**-K, THEN THE LAST
C  K SIGNIFICANT FIGURES ØF G(N-1) (=BR(N)+I*BI(N)) ARE
C  ERRØNEØUS.  IF THE USER WISHES TØ CALCULATE G(N-1) TØ
C  HIGHER ACCURACY, HE SHØULD USE AN ASYMPTØTIC FØRMULA FØR
C  LARGE ØRDER.
C
      DØUBLE PRECISIØN
     1 X,Y,BR,BI,PR,PI,PLASTR,PLASTI,PØLDR,PØLDI,PSAVER,
     2 PSAVEI,EXPARG,TEST,TØVER,TEMPAR,TEMPAI,TEMPBR,TEMPBI,
     3 TEMPCR,TEMPCI,SIGN,SUMR,SUMI,ZINVR,ZINVI
      DIMENSIØN BR(NB),BI(NB)
      DATA NSIG,NTEN,LARGEZ,EXPARG/19,307,10000,7.D2/
      TEMPAR=DSQRT(X*X+Y*Y)
      MAGZ=IFIX(SNGL(TEMPAR))
      IF(NB.GT.0.AND.MAGZ.LE.LARGEZ.AND.((IZE.EQ.0.AND.
     1 DABS(Y).LE.EXPARG).ØR.(IZE.EQ.1.AND.DABS(X).LE.
     2 EXPARG))) GØ TØ 1
C ERRØR RETURN -- Z, NB, ØR IZE IS ØUT ØF RANGE
      NCALC=MINO(NB,0)-1
      RETURN
    1 SIGN=DBLE(FLØAT(1-2*IZE))
      NCALC=NB
C USE 2-TERM ASCENDING SERIES FØR SMALL Z
      IF(TEMPAR**4.LT..1D0**NSIG) GØ TØ 50
C INITIALIZE THE CALCULATIØN ØF THE P*S
      NBMZ=NB-MAGZ
      N=MAGZ+1
      IF(DABS(X).LT.DABS(Y)) GØ TØ 2
      ZINVR=1.D0/(X+Y*Y/X)
      ZINVI=-Y*ZINVR/X
      GØ TØ 3
    2 ZINVI=-1.D0/(Y+X*X/Y)
      ZINVR=-X*ZINVI/Y
    3 PLASTR=1.D0
      PLASTI=0.D0
      PR=SIGN*DBLE(FLØAT(2*N))*ZINVR
      PI=SIGN*DBLE(FLØAT(2*N))*ZINVI
      TEST=2.D0*1.D1**NSIG
      M=0
      IF(NBMZ.LT.3) GØ TØ 6
C CALCULATE P*S UNTIL N=NB-1.  CHECK FØR PØSSIBLE ØVERFLØW.
      TØVER=1.D1**(NTEN-NSIG)
      NSTART=MAGZ+2
```

```
      NEND=NB-1
      DØ 5 N=NSTART,NEND
        PØLDR=PLASTR
        PØLDI=PLASTI
        PLASTR=PR
        PLASTI=PI
        PR=SIGN*(DBLE(FLØAT(2*N))*(PLASTR*ZINVR-PLASTI*ZINVI
     1  )-PØLDR)
        PI=SIGN*(DBLE(FLØAT(2*N))*(PLASTI*ZINVR+PLASTR*ZINVI
     1  )-PØLDI)
        IF((PR/TØVER)**2+(PI/TØVER)**2-1.D0) 5,5,7
    5   CØNTINUE
      N=NEND
C CALCULATE SPECIAL SIGNIFICANCE TEST FØR NBMZ.GT.2.
      TEMPBI=DMAX1(DABS(PR),DABS(PI))
      TEMPBI=TEMPBI*DSQRT(2.D0*1.D1**NSIG*DSQRT(((PR/TEMPBI)
     1**2+(PI/TEMPBI)**2)*((PLASTR/TEMPBI)**2+(PLASTI/
     2TEMPBI)**2)))
      TEST=DMAX1(TEST,TEMPBI)
C CALCULATE P*S UNTIL SIGNIFICANCE TEST IS PASSED.
    6 N=N+1
      PØLDR=PLASTR
      PØLDI=PLASTI
      PLASTR=PR
      PLASTI=PI
      PR=SIGN*(DBLE(FLØAT(2*N))*(PLASTR*ZINVR-PLASTI*ZINVI)
     1 -PØLDR)
      PI=SIGN*(DBLE(FLØAT(2*N))*(PLASTI*ZINVR+PLASTR*ZINVI)
     1 -PØLDI)
      IF((PR/TEST)**2+(PI/TEST)**2.LT.1.D0) GØ TØ 6
      IF(M.EQ.1) GØ TØ 12
C CALCULATE STRICT VARIANT ØF SIGNIFICANCE TEST, AND
C CALCULATE P*S UNTIL THIS TEST IS PASSED.
      M=1
      TEMPBI=DMAX1(DABS(PR),DABS(PI))
      TEMPBR=DSQRT(((PR/TEMPBI)**2+(PI/TEMPBI)**2)/
     1 ((PLASTR/TEMPBI)**2+(PLASTI/TEMPBI)**2))
      TEMPBI=DBLE(FLØAT(N+1))/TEMPAR
      IF(TEMPBR+1.D0/TEMPBR.GT.2.D0*TEMPBI) TEMPBR=TEMPBI
     1 +DSQRT(TEMPBI**2-1.D0)
      TEST=TEST/DSQRT(TEMPBR-1.D0/TEMPBR)
      IF((PR/TEST)**2+(PI/TEST)**2-1.D0) 6,12,12
    7 NSTART=N+1
C TØ AVØID ØVERFLØW, NØRMALIZE P*S BY DIVIDING BY TØVER.
C CALCULATE P*S UNTIL UNNØRMALIZED P WØULD ØVERFLØW.
      PR=PR/TØVER
      PI=PI/TØVER
      PLASTR=PLASTR/TØVER
      PLASTI=PLASTI/TØVER
      PSAVER=PR
      PSAVEI=PI
      TEMPCR=PLASTR
      TEMPCI=PLASTI
      TEST=1.D1**(2*NSIG)
    8 N=N+1
      PØLDR=PLASTR
      PØLDI=PLASTI
      PLASTR=PR
```

```
      PLASTI=PI
      PR=SIGN*(DBLE(FLØAT(2*N))*(PLASTR*ZINVR-PLASTI*ZINVI)
     1 -PØLDR)
      PI=SIGN*(DBLE(FLØAT(2*N))*(PLASTI*ZINVR+PLASTR*ZINVI)
     1 -PØLDI)
      IF(PR**2+PI**2.LE.TEST) GØ TØ 8
C CALCULATE BACKWARD TEST, AND FIND NCALC, THE HIGHEST N
C SUCH THAT THE TEST IS PASSED.
      TEMPBR=DSQRT((PLASTR**2+PLASTI**2)/(PØLDR**2+PØLDI**2)
     1)
      TEMPBI=DBLE(FLØAT(N))/TEMPAR
      IF(TEMPBR+1.D0/TEMPBR.GT.2.D0*TEMPBI) TEMPBR=TEMPBI+
     1 DSQRT(TEMPBI**2-1.D0)
      TEST=.5D0*(1.D0-1.D0/TEMPBR**2)/1.D1**NSIG
      TEST=((PLASTR**2+PLASTI**2)*TEST)*((PØLDR**2+PØLDI**2)
     1 *TEST)
      PR=PLASTR*TØVER
      PI=PLASTI*TØVER
      N=N-1
      NEND=MIN0(NB,N)
      DØ 9 NCALC=NSTART,NEND
        PØLDR=TEMPCR
        PØLDI=TEMPCI
        TEMPCR=PSAVER
        TEMPCI=PSAVEI
        PSAVER=SIGN*(DBLE(FLØAT(2*N))*(TEMPCR*ZINVR-TEMPCI*
     1  ZINVI)-PØLDR)
        PSAVEI=SIGN*(DBLE(FLØAT(2*N))*(TEMPCI*ZINVR+TEMPCR*
     1  ZINVI)-PØLDI)
        IF((PSAVER**2+PSAVEI**2)*(TEMPCR**2+TEMPCI**2)-TEST)
     1  9,9,10
    9   CØNTINUE
      NCALC=NEND+1
   10 NCALC=NCALC-1
C THE CØEFFICIENT ØF B(N) IN THE NØRMALIZATIØN SUM IS
C M*SQRT(-1)**IMAG, WHERE M=-2,0, ØR 2, AND IMAG IS 0 ØR 1.
C CALCULATE RECURSIØN RULES FØR M AND IMAG, AND INITIALIZE
C THEM.
   12 N=N+1
      TEMPBR=DBLE(FLØAT(IZE))*X+DBLE(FLØAT(1-IZE))*Y
      IPØS=0
      IF(TEMPBR) 13,14,13
   13 IPØS=IFIX(SNGL(1.1D0*TEMPBR/DABS(TEMPBR)))
   14 MRECUR=4*((2+IZE+IPØS)/2)-3-2*(IZE+IPØS)
      K=2+IPØS+2*IZE-IPØS**2-IZE
      L=N-4*(N/4)
      MLAST=2+8*((K*L)/4)-4*((K*L)/2)
      IF(IPØS.EQ.0.AND.(L.EQ.1.ØR.L.EQ.3)) MLAST=0
      L=L+3-4*((L+3)/4)
      M    =2+8*((K*L)/4)-4*((K*L)/2)
      IF(IPØS.EQ.0.AND.(L.EQ.1.ØR.L.EQ.3)) M=0
      IMRECR=(1-IZE)*IPØS**2
      IMAG=IMRECR*(L-2*(L/2))
C INITIALIZE THE BACKWARD RECURSIØN AND THE NØRMALIZATIØN
C SUM.
      TEMPBR=0.D0
      TEMPBI=0.D0
      IF(DABS(PI).GT.DABS(PR)) GØ TØ 15
```


507-714 O - 73 - 4

```
      TEMPAR=1.D0/(PR+PI*(PI/PR))
      TEMPAI=-(PI*TEMPAR)/PR
      GØ TØ 16
   15 TEMPAI=-1.D0/(PI+PR*(PR/PI))
      TEMPAR=-(PR*TEMPAI)/PI
   16 IF(IMAG.NE.0) GØ TØ 17
      SUMR=DBLE(FLØAT(M))*TEMPAR
      SUMI=DBLE(FLØAT(M))*TEMPAI
      GØ TØ 18
   17 SUMR=-DBLE(FLØAT(M))*TEMPAI
      SUMI=DBLE(FLØAT(M))*TEMPAR
   18 NEND=N-NB
      IF(NEND) 26,22,19
C RECUR BACKWARD VIA DIFFERENCE EQUATIØN CALCULATING (BUT
C NØT STØRING) BR(N) AND BI(N) UNTIL N=NB.
   19 DØ 21 L=1,NEND
        N=N-1
        TEMPCR=TEMPBR
        TEMPCI=TEMPBI
        TEMPBR=TEMPAR
        TEMPBI=TEMPAI
        PR=DBLE(FLØAT(2*N))*ZINVR
        PI=DBLE(FLØAT(2*N))*ZINVI
        TEMPAR=PR*TEMPBR-PI*TEMPBI-SIGN*TEMPCR
        TEMPAI=PR*TEMPBI+PI*TEMPBR-SIGN*TEMPCI
        IMAG=(1-IMAG)*IMRECR
        K=MLAST
        MLAST=M
        M=K*MRECUR
        IF(IMAG.NE.0) GØ TØ 20
        SUMR=SUMR+DBLE(FLØAT(M))*TEMPAR
        SUMI=SUMI+DBLE(FLØAT(M))*TEMPAI
        GØ TØ 21
   20   SUMR=SUMR-DBLE(FLØAT(M))*TEMPAI
        SUMI=SUMI+DBLE(FLØAT(M))*TEMPAR
   21   CØNTINUE
C STØRE BR(NB), BI(NB)
   22 BR(N)=TEMPAR
      BI(N)=TEMPAI
      IF(N.GT.1) GØ TØ 23
C NB=1.  SINCE 2*TEMPAR AND 2*TEMPAI WERE ADDED TØ SUMR AND
C SUMI RESPECTIVELY, WE MUST SUBTRACT TEMPAR AND TEMPAI
      SUMR=SUMR-TEMPAR
      SUMI=SUMI-TEMPAI
      GØ TØ 35
C CALCULATE AND STØRE BR(NB-1),BI(NB-1)
   23 N=N-1
      PR=DBLE(FLØAT(2*N))*ZINVR
      PI=DBLE(FLØAT(2*N))*ZINVI
      BR(N)=PR*TEMPAR-PI*TEMPAI-SIGN*TEMPBR
      BI(N)=PR*TEMPAI+PI*TEMPAR-SIGN*TEMPBI
      IF(N.EQ.1) GØ TØ 34
      IMAG=(1-IMAG)*IMRECR
      K=MLAST
      MLAST=M
      M=K*MRECUR
      IF(IMAG.NE.0) GØ TØ 24
      SUMR=SUMR+DBLE(FLØAT(M))*BR(N)
```

```
              SUMI=SUMI+DBLE(FLØAT(M))*BI(N)
              GØ TØ 30
           24 SUMR=SUMR-DBLE(FLØAT(M))*BI(N)
              SUMI=SUMI+DBLE(FLØAT(M))*BR(N)
              GØ TØ 30
C N.LT.NB, SØ STØRE BR(N), BI(N), AND SET HIGHER ØRDERS ZERØ
           26 BR(N)=TEMPAR
              BI(N)=TEMPAI
              NEND=-NEND
              DØ 27 L=1,NEND
                BR(N+L)=0.D0
           27   BI(N+L)=0.D0
           30 NEND=N-2
              IF(NEND.EQ.0) GØ TØ 33
C CALCULATE VIA DIFFERENCE EQUATIØN AND STØRE BR(N),BI(N),
C UNTIL N=2
              DØ 32 L=1,NEND
                N=N-1
                PR=DBLE(FLØAT(2*N))*ZINVR
                PI=DBLE(FLØAT(2*N))*ZINVI
                BR(N)=PR*BR(N+1)-PI*BI(N+1)-SIGN*BR(N+2)
                BI(N)=PR*BI(N+1)+PI*BR(N+1)-SIGN*BI(N+2)
                IMAG=(1-IMAG)*IMRECR
                K=MLAST
                MLAST=M
                M=K*MRECUR
                IF(IMAG.NE.0) GØ TØ 31
                SUMR=SUMR+DBLE(FLØAT(M))*BR(N)
                SUMI=SUMI+DBLE(FLØAT(M))*BI(N)
                GØ TØ 32
           31   SUMR=SUMR-DBLE(FLØAT(M))*BI(N)
                SUMI=SUMI+DBLE(FLØAT(M))*BR(N)
           32   CØNTINUE
C CALCULATE AND STØRE BR(1), BI(1)
           33 BR(1)=2.D0*(BR(2)*ZINVR-BI(2)*ZINVI)-SIGN*BR(3)
              BI(1)=2.D0*(BR(2)*ZINVI+BI(2)*ZINVR)-SIGN*BI(3)
           34 SUMR=SUMR+BR(1)
              SUMI=SUMI+BI(1)
C CALCULATE NØRMALIZATIØN FACTØR, TEMPAR +I*TEMPAI
           35 IF(IZE.EQ.1) GØ TØ 36
              TEMPCR=DBLE(FLØAT(IPØS))*Y
              TEMPCI=DBLE(FLØAT(-IPØS))*X
              GØ TØ 37
           36 TEMPCR=DBLE(FLØAT(IPØS))*X
              TEMPCI=DBLE(FLØAT(IPØS))*Y
           37 TEMPCR=DEXP(TEMPCR)
              TEMPBR=DCØS(TEMPCI)
              TEMPBI=DSIN(TEMPCI)
              IF(DABS(SUMR).LT.DABS(SUMI)) GØ TØ 38
              TEMPCI=SUMI/SUMR
              TEMPCR=(TEMPCR/SUMR)/(1.D0+TEMPCI*TEMPCI)
              TEMPAR=TEMPCR*(TEMPBR+TEMPBI*TEMPCI)
              TEMPAI=TEMPCR*(TEMPBI-TEMPBR*TEMPCI)
              GØ TØ 39
           38 TEMPCI=SUMR/SUMI
              TEMPCR=(TEMPCR/SUMI)/(1.D0+TEMPCI*TEMPCI)
              TEMPAR=TEMPCR*(TEMPBR*TEMPCI+TEMPBI)
              TEMPAI=TEMPCR*(TEMPBI*TEMPCI-TEMPBR)
C NØRMALIZE
           39 DØ 40 N=1,NB
                TEMPBR=BR(N)*TEMPAR-BI(N)*TEMPAI
                BI(N)=BR(N)*TEMPAI+BI(N)*TEMPAR
           40   BR(N)=TEMPBR
              RETURN
C TWØ-TERM ASCENDING SERIES FØR SMALL Z
           50 TEMPAR=1.D0
              TEMPAI=0.D0
              TEMPCR=.25D0*(X*X-Y*Y)
              TEMPCI=.5D0*X*Y
              BR(1)=1.D0-SIGN*TEMPCR
              BI(1)=-SIGN*TEMPCI
              IF(NB.EQ.1) GØ TØ 52
              DØ 51 N=2,NB
                TEMPBR=(TEMPAR*X-TEMPAI*Y)/DBLE(FLØAT(2*N-2))
                TEMPAI=(TEMPAR*Y+TEMPAI*X)/DBLE(FLØAT(2*N-2))
                TEMPAR=TEMPBR
                TEMPBR=DBLE(FLØAT(N))
                BR(N)=TEMPAR*(1.D0-SIGN*TEMPCR/TEMPBR)
     1          +TEMPAI*TEMPCI/TEMPBR
           51   BI(N)=TEMPAI*(1.D0-SIGN*TEMPCR/TEMPBR)
     1          -TEMPAR*TEMPCI/TEMPBR
           52 RETURN
              END
```

END CUR LCC 1102-0038 L8

6. References

[1] Algorithm BESLRI, see previous paper.

[2] Maximon, Leonard C., FORTRAN Program for Arbitrary Precision Arithmetic, unpublished data.

[3] Olver, Frank W. J., Error Bounds for Asymptotic Expansion with an Application to Cylinder Functions of Large Argument, in Asymptotic Solutions of Differential Equations and Their Applications, Calvin H. Wilcox, Editor (Wiley & Sons, 1964).

[4] National Bureau of Standards, Handbook of Mathematical Functions, Nat. Bur. Stand. (U.S.), Appl. Math. Ser. 55 (1964).

[5] Certification of an algorithm for Bessel functions of complex argument, J. Res. Nat. Bur. Stand. (U.S.), in press.

(Paper 77B3&4–386)