

# NAG Library Routine Document

## S17DLF

**Note:** before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details.

### 1 Purpose

S17DLF returns a sequence of values for the Hankel functions  $H_{\nu+n}^{(1)}(z)$  or  $H_{\nu+n}^{(2)}(z)$  for complex  $z$ , non-negative  $\nu$  and  $n = 0, 1, \dots, N-1$ , with an option for exponential scaling.

### 2 Specification

```
SUBROUTINE S17DLF (M, FNU, Z, N, SCAL, CY, NZ, IFAIL)
INTEGER          M, N, NZ, IFAIL
REAL (KIND=nag_wp) FNU
COMPLEX (KIND=nag_wp) Z, CY(N)
CHARACTER(1)     SCAL
```

### 3 Description

S17DLF evaluates a sequence of values for the Hankel function  $H_{\nu}^{(1)}(z)$  or  $H_{\nu}^{(2)}(z)$ , where  $z$  is complex,  $-\pi < \arg z \leq \pi$ , and  $\nu$  is the real, non-negative order. The  $N$ -member sequence is generated for orders  $\nu, \nu+1, \dots, \nu+N-1$ . Optionally, the sequence is scaled by the factor  $e^{-iz}$  if the function is  $H_{\nu}^{(1)}(z)$  or by the factor  $e^{iz}$  if the function is  $H_{\nu}^{(2)}(z)$ .

**Note:** although the routine may not be called with  $\nu$  less than zero, for negative orders the formulae  $H_{-\nu}^{(1)}(z) = e^{\nu\pi i} H_{\nu}^{(1)}(z)$ , and  $H_{-\nu}^{(2)}(z) = e^{-\nu\pi i} H_{\nu}^{(2)}(z)$  may be used.

The routine is derived from the routine CBESH in Amos (1986). It is based on the relation

$$H_{\nu}^{(m)}(z) = \frac{1}{p} e^{-p\nu} K_{\nu}(ze^{-p}),$$

where  $p = \frac{i\pi}{2}$  if  $m = 1$  and  $p = -\frac{i\pi}{2}$  if  $m = 2$ , and the Bessel function  $K_{\nu}(z)$  is computed in the right half-plane only. Continuation of  $K_{\nu}(z)$  to the left half-plane is computed in terms of the Bessel function  $I_{\nu}(z)$ . These functions are evaluated using a variety of different techniques, depending on the region under consideration.

When  $N$  is greater than 1, extra values of  $H_{\nu}^{(m)}(z)$  are computed using recurrence relations.

For very large  $|z|$  or  $(\nu + N - 1)$ , argument reduction will cause total loss of accuracy, and so no computation is performed. For slightly smaller  $|z|$  or  $(\nu + N - 1)$ , the computation is performed but results are accurate to less than half of *machine precision*. If  $|z|$  is very small, near the machine underflow threshold, or  $(\nu + N - 1)$  is too large, there is a risk of overflow and so no computation is performed. In all the above cases, a warning is given by the routine.

### 4 References

Abramowitz M and Stegun I A (1972) *Handbook of Mathematical Functions* (3rd Edition) Dover Publications

Amos D E (1986) Algorithm 644: A portable package for Bessel functions of a complex argument and non-negative order *ACM Trans. Math. Software* **12** 265–273

## 5 Arguments

- 1: M – INTEGER *Input*  
*On entry:* the kind of functions required.  
M = 1  
The functions are  $H_{\nu}^{(1)}(z)$ .  
M = 2  
The functions are  $H_{\nu}^{(2)}(z)$ .  
*Constraint:* M = 1 or 2.
- 2: FNU – REAL (KIND=nag\_wp) *Input*  
*On entry:*  $\nu$ , the order of the first member of the sequence of functions.  
*Constraint:* FNU  $\geq$  0.0.
- 3: Z – COMPLEX (KIND=nag\_wp) *Input*  
*On entry:* the argument  $z$  of the functions.  
*Constraint:* Z  $\neq$  (0.0,0.0).
- 4: N – INTEGER *Input*  
*On entry:* N, the number of members required in the sequence  $H_{\nu}^{(M)}(z), H_{\nu+1}^{(M)}(z), \dots, H_{\nu+N-1}^{(M)}(z)$ .  
*Constraint:* N  $\geq$  1.
- 5: SCAL – CHARACTER(1) *Input*  
*On entry:* the scaling option.  
SCAL = 'U'  
The results are returned unscaled.  
SCAL = 'S'  
The results are returned scaled by the factor  $e^{-iz}$  when M = 1, or by the factor  $e^{iz}$  when M = 2.  
*Constraint:* SCAL = 'U' or 'S'.
- 6: CY(N) – COMPLEX (KIND=nag\_wp) array *Output*  
*On exit:* the N required function values: CY(*i*) contains  $H_{\nu+i-1}^{(M)}(z)$ , for  $i = 1, 2, \dots, N$ .
- 7: NZ – INTEGER *Output*  
*On exit:* the number of components of CY that are set to zero due to underflow. If NZ > 0, then if  $\text{Im}(z) > 0.0$  and M = 1, or  $\text{Im}(z) < 0.0$  and M = 2, elements CY(1), CY(2), ..., CY(NZ) are set to zero. In the complementary half-planes, NZ simply states the number of underflows, and not which elements they are.
- 8: IFAIL – INTEGER *Input/Output*  
*On entry:* IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.  
For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

*On exit:* IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

## 6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry,  $M \neq 1$  and  $M \neq 2$ ,  
 or  $FNU < 0.0$ ,  
 or  $Z = (0.0, 0.0)$ ,  
 or  $N < 1$ ,  
 or  $SCAL \neq 'U'$  or  $'S'$ .

IFAIL = 2

No computation has been performed due to the likelihood of overflow, because  $\text{abs}(Z)$  is less than a machine-dependent threshold value (given in the Users' Note for your implementation).

IFAIL = 3

No computation has been performed due to the likelihood of overflow, because  $FNU + N - 1$  is too large – how large depends on  $Z$  and the overflow threshold of the machine.

IFAIL = 4

The computation has been performed, but the errors due to argument reduction in elementary functions make it likely that the results returned by S17DLF are accurate to less than half of *machine precision*. This error exit may occur if either  $\text{abs}(Z)$  or  $FNU + N - 1$  is greater than a machine-dependent threshold value (given in the Users' Note for your implementation).

IFAIL = 5

No computation has been performed because the errors due to argument reduction in elementary functions mean that all precision in results returned by S17DLF would be lost. This error exit may occur when either of  $\text{abs}(Z)$  or  $FNU + N - 1$  is greater than a machine-dependent threshold value (given in the Users' Note for your implementation).

IFAIL = 6

No results are returned because the algorithm termination condition has not been met. This may occur because the arguments supplied to S17DLF would have caused overflow or underflow.

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -999

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

## 7 Accuracy

All constants in S17DLF are given to approximately 18 digits of precision. Calling the number of digits of precision in the floating-point arithmetic being used  $t$ , then clearly the maximum number of correct digits in the results obtained is limited by  $p = \min(t, 18)$ . Because of errors in argument reduction when computing elementary functions inside S17DLF, the actual number of correct digits is limited, in general, by  $p - s$ , where  $s \approx \max(1, |\log_{10} |z||, |\log_{10} \nu|)$  represents the number of digits lost due to the argument reduction. Thus the larger the values of  $|z|$  and  $\nu$ , the less the precision in the result. If S17DLF is called with  $N > 1$ , then computation of function values via recurrence may lead to some further small loss of accuracy.

If function values which should nominally be identical are computed by calls to S17DLF with different base values of  $\nu$  and different  $N$ , the computed values may not agree exactly. Empirical tests with modest values of  $\nu$  and  $z$  have shown that the discrepancy is limited to the least significant 3 – 4 digits of precision.

## 8 Parallelism and Performance

S17DLF is not threaded in any implementation.

## 9 Further Comments

The time taken for a call of S17DLF is approximately proportional to the value of  $N$ , plus a constant. In general it is much cheaper to call S17DLF with  $N$  greater than 1, rather than to make  $N$  separate calls to S17DLF.

Paradoxically, for some values of  $z$  and  $\nu$ , it is cheaper to call S17DLF with a larger value of  $N$  than is required, and then discard the extra function values returned. However, it is not possible to state the precise circumstances in which this is likely to occur. It is due to the fact that the base value used to start recurrence may be calculated in different regions for different  $N$ , and the costs in each region may differ greatly.

## 10 Example

This example prints a caption and then proceeds to read sets of data from the input data stream. The first datum is a value for the kind of function,  $M$ , the second is a value for the order  $FNU$ , the third is a complex value for the argument,  $Z$ , and the fourth is a character value to set the argument  $SCAL$ . The program calls the routine with  $N = 2$  to evaluate the function for orders  $FNU$  and  $FNU + 1$ , and it prints the results. The process is repeated until the end of the input data stream is encountered.

### 10.1 Program Text

```

Program s17dlfe

!      S17DLF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
Use nag_library, Only: nag_wp, s17dlf
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: n = 2, nin = 5, nout = 6
!      .. Local Scalars ..
Complex (Kind=nag_wp)      :: z
Real (Kind=nag_wp)         :: fnu
Integer                     :: ifail, ioerr, m, nz
Character (1)               :: scal
!      .. Local Arrays ..
Complex (Kind=nag_wp)      :: cy(n)
!      .. Executable Statements ..
Write (nout,*) 'S17DLF Example Program Results'

```

```

!      Skip heading in data file
      Read (nin,*)

      Write (nout,*)
      Write (nout,99999) 'Calling with N =', n
      Write (nout,*)
      Write (nout,*) 'M      FNU          Z          SCAL          CY(1) ' //      &
      '      CY(2)          NZ'
      Write (nout,*)

data: Do
      Read (nin,*,Iostat=ioerr) m, fnu, z, scal

      If (ioerr<0) Then
        Exit data
      End If

      ifail = 0
      Call s17dlf(m,fnu,z,n,scal,cy,nz,ifail)

      Write (nout,99998) m, fnu, z, scal, cy(1), cy(2), nz
      End Do data

99999 Format (1X,A,I2)
99998 Format (1X,I1,1X,F7.4,' (' ,F7.3,',',',F7.3,') ' ,A,      &
      2(' (' ,F7.3,',',',F7.3,')'),I4)
      End Program s17dlfe

```

## 10.2 Program Data

```

S17DLF Example Program Data
1   0.00   ( 0.3, 0.4)   'U'
1   2.30   ( 2.0, 0.0)   'U'
1   2.12   (-1.0, 0.0)   'U'
2   6.00   ( 3.1, -1.6)  'U'
2   6.00   ( 3.1, -1.6)  'S'

```

## 10.3 Program Results

S17DLF Example Program Results

Calling with N = 2

M	FNU	Z	SCAL	CY(1)	CY(2)	NZ
1	0.0000	( 0.300, 0.400)	U	( 0.347, -0.559)	( -0.791, -0.818)	0
1	2.3000	( 2.000, 0.000)	U	( 0.272, -0.740)	( 0.089, -1.412)	0
1	2.1200	( -1.000, 0.000)	U	( -0.772, -1.693)	( 2.601, 6.527)	0
2	6.0000	( 3.100, -1.600)	U	( -1.371, -1.280)	( -1.491, -5.993)	0
2	6.0000	( 3.100, -1.600)	S	( 7.050, 6.052)	( 8.614, 29.352)	0