

NAG Library Routine Document

S18ADF

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

S18ADF returns the value of the modified Bessel function $K_1(x)$, via the function name.

2 Specification

```
FUNCTION S18ADF (X, IFAIL)
REAL (KIND=nag_wp) S18ADF
INTEGER IFAIL
REAL (KIND=nag_wp) X
```

3 Description

S18ADF evaluates an approximation to the modified Bessel function of the second kind $K_1(x)$.

Note: $K_1(x)$ is undefined for $x \leq 0$ and the routine will fail for such arguments.

The routine is based on five Chebyshev expansions:

For $0 < x \leq 1$,

$$K_1(x) = \frac{1}{x} + x \ln x \sum_{r=0} a_r T_r(t) - x \sum_{r=0} b_r T_r(t), \quad \text{where } t = 2x^2 - 1.$$

For $1 < x \leq 2$,

$$K_1(x) = e^{-x} \sum_{r=0} c_r T_r(t), \quad \text{where } t = 2x - 3.$$

For $2 < x \leq 4$,

$$K_1(x) = e^{-x} \sum_{r=0} d_r T_r(t), \quad \text{where } t = x - 3.$$

For $x > 4$,

$$K_1(x) = \frac{e^{-x}}{\sqrt{x}} \sum_{r=0} e_r T_r(t), \quad \text{where } t = \frac{9-x}{1+x}.$$

For x near zero, $K_1(x) \simeq \frac{1}{x}$. This approximation is used when x is sufficiently small for the result to be correct to ***machine precision***. For very small x on some machines, it is impossible to calculate $\frac{1}{x}$ without overflow and the routine must fail.

For large x , where there is a danger of underflow due to the smallness of K_1 , the result is set exactly to zero.

4 References

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

5 Arguments

- 1: X – REAL (KIND=nag_wp) *Input*
On entry: the argument x of the function.
Constraint: $X > 0.0$.
- 2: 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

$X \leq 0.0$, K_1 is undefined. On softfailure the routine returns zero.

IFAIL = 2

X is too small, there is a danger of overflow. On softfailure the routine returns approximately the largest representable value. (see the Users' Note for your implementation for details)

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

Let δ and ϵ be the relative errors in the argument and result respectively.

If δ is somewhat larger than the *machine precision* (i.e., if δ is due to data errors etc.), then ϵ and δ are approximately related by:

$$\epsilon \simeq \left| \frac{xK_0(x) - K_1(x)}{K_1(x)} \right| \delta.$$

Figure 1 shows the behaviour of the error amplification factor

$$\left| \frac{xK_0(x) - K_1(x)}{K_1(x)} \right|.$$

However if δ is of the same order as the *machine precision*, then rounding errors could make ϵ slightly larger than the above relation predicts.

For small x , $\epsilon \simeq \delta$ and there is no amplification of errors.

For large x , $\epsilon \simeq x\delta$ and we have strong amplification of the relative error. Eventually K_1 , which is asymptotically given by $\frac{e^{-x}}{\sqrt{x}}$, becomes so small that it cannot be calculated without underflow and hence the routine will return zero. Note that for large x the errors will be dominated by those of the standard function \exp .

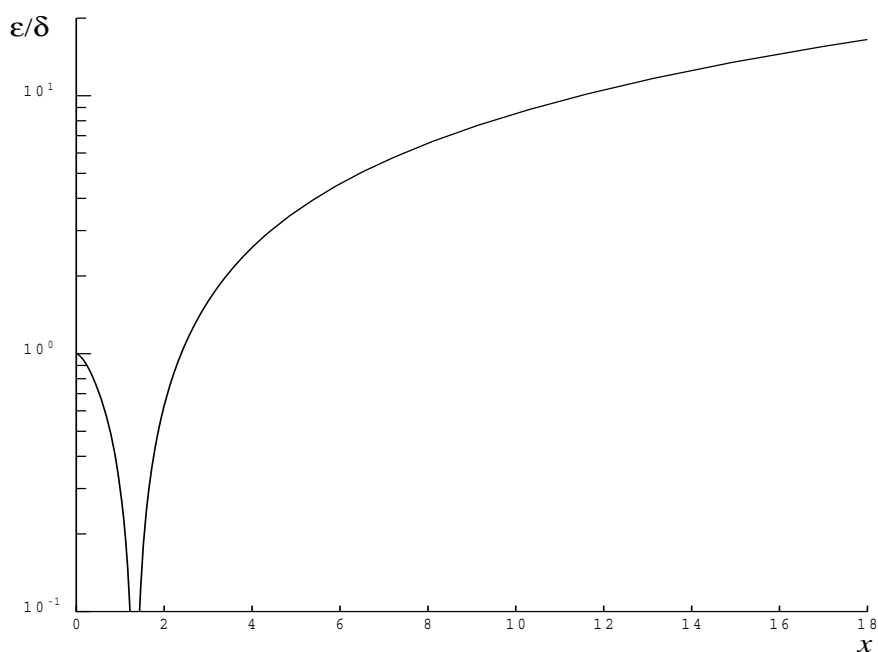


Figure 1

8 Parallelism and Performance

S18ADF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads values of the argument x from a file, evaluates the function at each value of x and prints the results.

10.1 Program Text

```

Program s18adfe

!      S18ADF Example Program Text
!
!      Mark 26 Release. NAG Copyright 2016.
!
!      .. Use Statements ..
```

```

      Use nag_library, Only: nag_wp, s18adf
!      .. Implicit None Statement ..
      Implicit None
!      .. Parameters ..
      Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
      Real (Kind=nag_wp)          :: x, y
      Integer                     :: ifail, ioerr
!      .. Executable Statements ..
      Write (nout,*) 'S18ADF Example Program Results'

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

      Write (nout,*)
      Write (nout,*) '      X      Y'
      Write (nout,*)

data: Do
      Read (nin,*,Iostat=ioerr) x

      If (ioerr<0) Then
        Exit data
      End If

      ifail = -1
      y = s18adf(x,ifail)

      If (ifail<0) Then
        Exit data
      End If

      Write (nout,99999) x, y
    End Do data

99999 Format (1X,1P,2E12.3)
End Program s18adfe

```

10.2 Program Data

S18ADF Example Program Data

```

0.4
0.6
1.4
1.6
2.5
3.5
6.0
8.0
10.0
1000.0

```

10.3 Program Results

S18ADF Example Program Results

X	Y
4.000E-01	2.184E+00
6.000E-01	1.303E+00
1.400E+00	3.208E-01
1.600E+00	2.406E-01
2.500E+00	7.389E-02
3.500E+00	2.224E-02
6.000E+00	1.344E-03
8.000E+00	1.554E-04
1.000E+01	1.865E-05
1.000E+03	0.000E+00

