NAG Library Routine Document

S18ATF

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

S18ATF returns an array of values for the modified Bessel function $I_1(x)$.

2   Specification

```
SUBROUTINE S18ATF (N, X, F, IVALID, IFAIL)
INTEGER            N, IVALID(N), IFAIL
REAL (KIND=nag_wp) X(N), F(N)
```

3   Description

S18ATF evaluates an approximation to the modified Bessel function of the first kind $I_1(x_i)$ for an array of arguments x_i, for $i = 1, 2, \ldots, n$.

Note: $I_1(-x) = -I_1(x)$, so the approximation need only consider $x \geq 0$.

The routine is based on three Chebyshev expansions:

For $0 < x \leq 4$,

$$I_1(x) = x \sum_{r=0} a_r T_r(t), \qquad \text{where } t = 2\left(\frac{x}{4}\right)^2 - 1;$$

For $4 < x \leq 12$,

$$I_1(x) = e^x \sum_{r=0} b_r T_r(t), \qquad \text{where } t = \frac{x-8}{4};$$

For $x > 12$,

$$I_1(x) = \frac{e^x}{\sqrt{x}} \sum_{r=0} c_r T_r(t), \qquad \text{where } t = 2\left(\frac{12}{x}\right) - 1.$$

For small x, $I_1(x) \simeq x$. This approximation is used when x is sufficiently small for the result to be correct to **machine precision**.

For large x, the routine must fail because $I_1(x)$ cannot be represented without overflow.

4   References

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

5   Arguments

1:    N – INTEGER   *Input*

   On entry: n, the number of points.

   Constraint: $N \geq 0$.

2:     X(N) – REAL (KIND=nag_wp) array   *Input*

       On entry: the argument x_i of the function, for $i = 1, 2, \ldots, N$.

3:     F(N) – REAL (KIND=nag_wp) array   *Output*

       On exit: $I_1(x_i)$, the function values.

4:     IVALID(N) – INTEGER array   *Output*

       On exit: IVALID(i) contains the error code for x_i, for $i = 1, 2, \ldots, N$.

       IVALID$(i) = 0$
              No error.

       IVALID$(i) = 1$
              x_i is too large. F(i) contains the approximate value of $I_1(x_i)$ at the nearest valid argument. The threshold value is the same as for IFAIL $= 1$ in S18AFF, as defined in the Users' Note for your implementation.

5:     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, at least one value of X was invalid.
       Check IVALID for more information.

IFAIL $= 2$

       On entry, N $= \langle value \rangle$.
       Constraint: N ≥ 0.

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:

$$\epsilon \simeq \left| \frac{x I_0(x) - I_1(x)}{I_1(x)} \right| \delta.$$

Figure 1 shows the behaviour of the error amplification factor

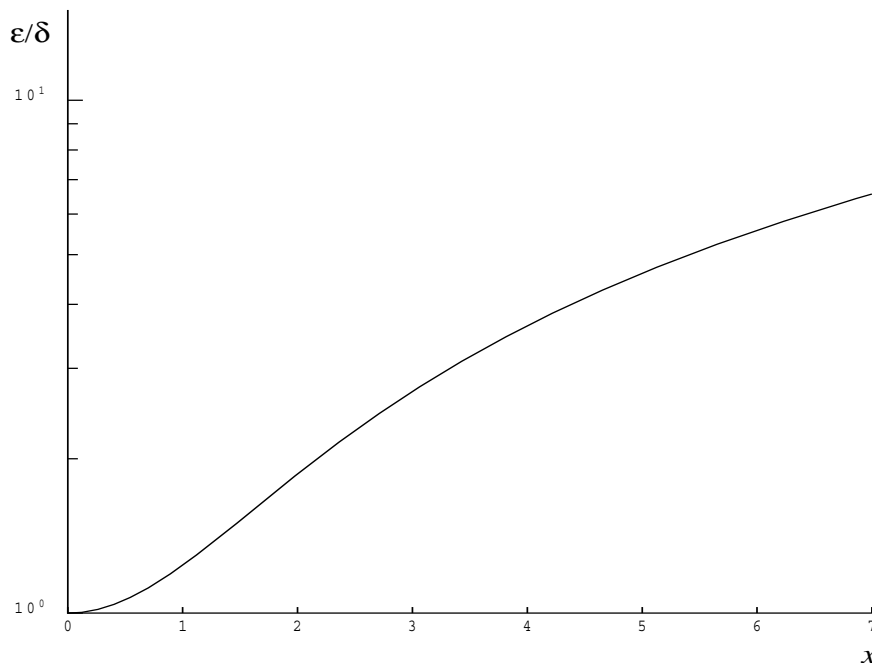$$\left| \frac{x I_0(x) - I_1(x)}{I_1(x)} \right|.$$

Figure 1

However, if δ is of the same order as ***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 errors. However, for quite moderate values of x ($x > \hat{x}$, the threshold value), the routine must fail because $I_1(x)$ would overflow; hence in practice the loss of accuracy for x close to $\hat{x}$ is not excessive and the errors will be dominated by those of the standard function exp.

8   Parallelism and Performance

S18ATF is not threaded in any implementation.

9   Further Comments

None.

10   Example

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

10.1 Program Text

```
    Program s18atfe

!     S18ATF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, s18atf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                 :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                            :: i, ifail, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: f(:), x(:)
      Integer, Allocatable             :: ivalid(:)
!     .. Executable Statements ..
      Write (nout,*) 'S18ATF Example Program Results'

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

      Write (nout,*)
      Write (nout,*) '     X             F           IVALID'
      Write (nout,*)

      Read (nin,*) n

      Allocate (x(n),f(n),ivalid(n))

      Read (nin,*) x(1:n)

      ifail = 0
      Call s18atf(n,x,f,ivalid,ifail)

      Do i = 1, n
        Write (nout,99999) x(i), f(i), ivalid(i)
      End Do

99999 Format (1X,1P,2E12.3,I5)
    End Program s18atfe
```

10.2 Program Data

```
S18ATF Example Program Data

10

0.0 0.5 1.0 3.0 6.0 8.0 10.0 15.0 20.0 -1.0
```

10.3 Program Results

```
 S18ATF Example Program Results

     X             F           IVALID

    0.000E+00   0.000E+00      0
    5.000E-01   2.579E-01      0
    1.000E+00   5.652E-01      0
    3.000E+00   3.953E+00      0
    6.000E+00   6.134E+01      0
```

```
8.000E+00   3.999E+02    0
1.000E+01   2.671E+03    0
1.500E+01   3.281E+05    0
2.000E+01   4.245E+07    0
-1.000E+00  -5.652E-01    0
```