NAG Library Routine Document

S18AFF

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

S18AFF returns a value for the modified Bessel function $I_1(x)$, via the function name.

2 Specification

```
FUNCTION S18AFF (X, IFAIL)
REAL (KIND=nag_wp) S18AFF
INTEGER            IFAIL
REAL (KIND=nag_wp) X
```

3 Description

S18AFF evaluates an approximation to the modified Bessel function of the first kind $I_1(x)$.

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 Parameters

1: X – REAL (KIND=nag_wp) *Input*

On entry: the argument x of the function.

2: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction 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 parameter, 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 is too large. On softfailure the routine returns the approximate value of $I_1(x)$ at the nearest valid argument. See also the Users' Note for your implementation.

IFAIL $= -99$

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

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

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

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction 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:

$$\epsilon \simeq \left| \frac{xI_0(x) - I_1(x)}{I_1(x)} \right| \delta.$$

Figure 1 shows the behaviour of the error amplification factor

$$\left| \frac{xI_0(x) - I_1(x)}{I_1(x)} \right|.$$

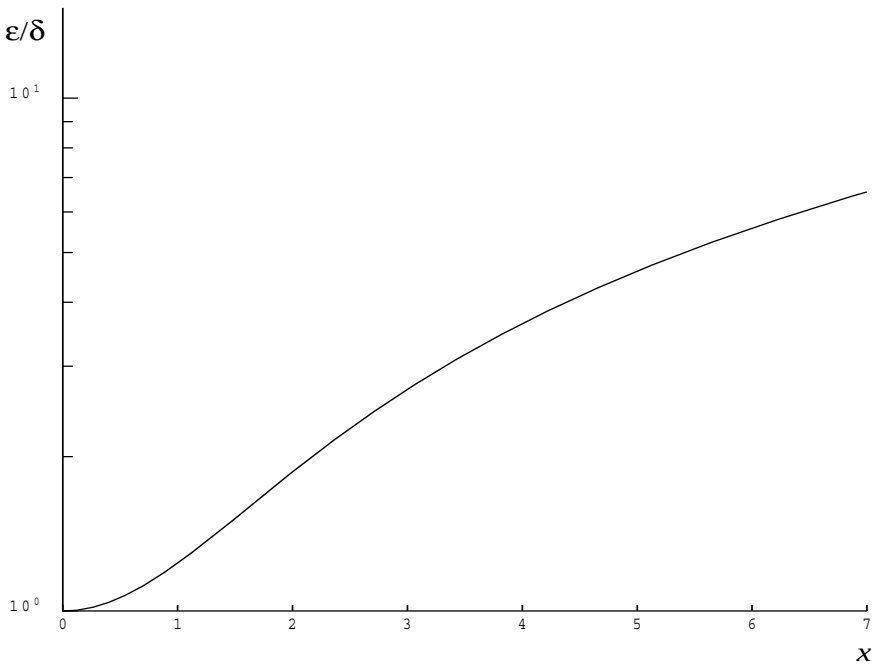


Figure 1

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 the routine must fail for quite moderate values of x because $I_1(x)$ would overflow; hence in practice the loss of accuracy for large x is not excessive. Note that for large x, the errors will be dominated by those of the standard function exp.

8 Parallelism and Performance

Not applicable.

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 s18affe

!     S18AFF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, s18aff
!     .. 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,*) 'S18AFF 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 = s18aff(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 s18affe
```

10.2 Program Data

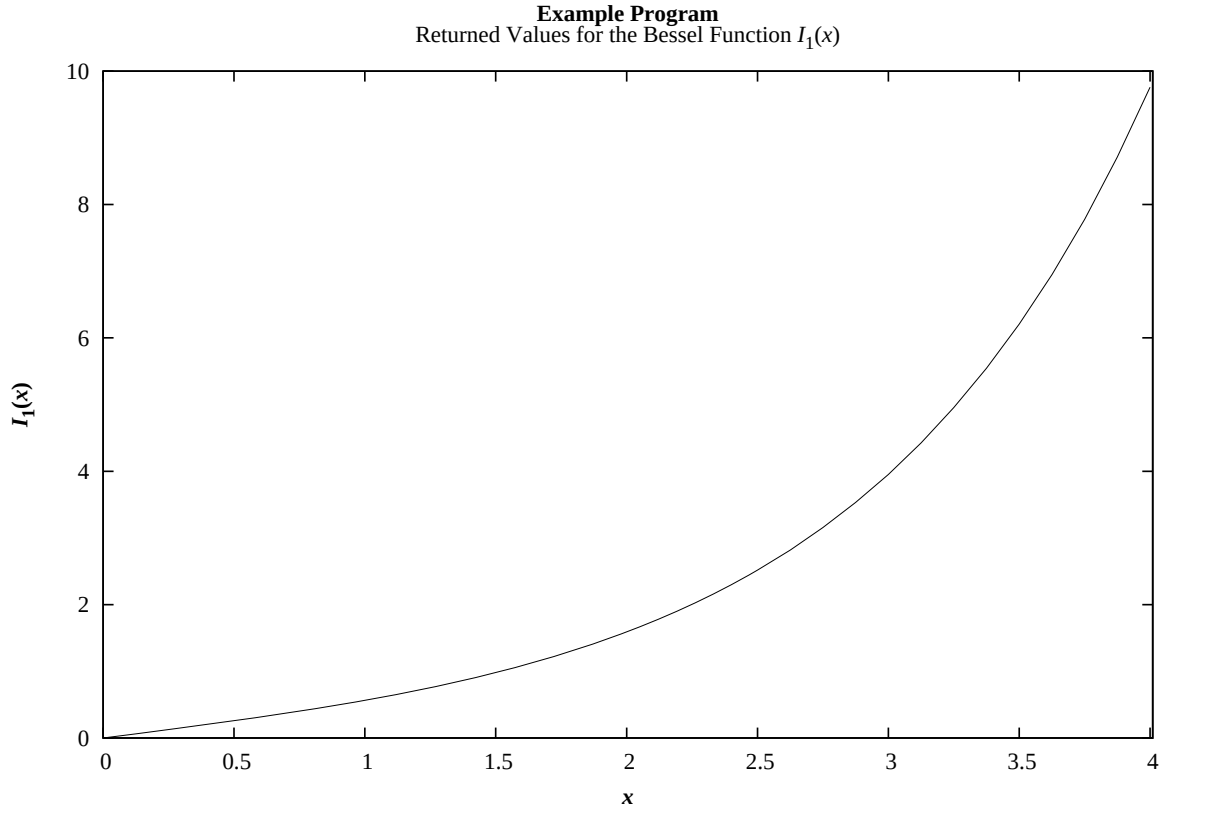
```
S18AFF Example Program Data
             0.0
             0.5
             1.0
             3.0
             6.0
             8.0
             10.0
             15.0
             20.0
             -1.0
```

10.3 Program Results

```
 S18AFF Example Program Results

      X           Y

    0.000E+00   0.000E+00
    5.000E-01   2.579E-01
    1.000E+00   5.652E-01
    3.000E+00   3.953E+00
    6.000E+00   6.134E+01
    8.000E+00   3.999E+02
    1.000E+01   2.671E+03
    1.500E+01   3.281E+05
    2.000E+01   4.245E+07
   -1.000E+00  -5.652E-01
```


Example Program
Returned Values for the Bessel Function $I_1(x)$
10
8
6
4
2
0
$I_1(x)$
0
0.5
1
1.5
2
2.5
3
3.5
4
x