NAG Library Routine Document

S17AXF

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

S17AXF returns an array of values for the derivative of the Airy function $\mathrm{Bi}(x)$.

2 Specification

```
SUBROUTINE S17AXF (N, X, F, IVALID, IFAIL)

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

3 Description

S17AXF calculates an approximate value for the derivative of the Airy function $\mathrm{Bi}(x_i)$ for an array of arguments x_i, for $i = 1, 2, \dots, n$. It is based on a number of Chebyshev expansions.

For $x < -5$,

$$\mathrm{Bi}'(x) = \sqrt[4]{-x}\left[-a(t)\sin z + \frac{b(t)}{\zeta}\cos z\right],$$

where $z = \frac{\pi}{4} + \zeta$, $\zeta = \frac{2}{3}\sqrt{-x^3}$ and $a(t)$ and $b(t)$ are expansions in the variable $t = -2\left(\frac{5}{x}\right)^3 - 1$.

For $-5 \leq x \leq 0$,

$$\mathrm{Bi}'(x) = \sqrt{3}\left(x^2 f(t) + g(t)\right),$$

where f and g are expansions in $t = -2\left(\frac{x}{5}\right)^3 - 1$.

For $0 < x < 4.5$,

$$\mathrm{Bi}'(x) = e^{3x/2} y(t),$$

where $y(t)$ is an expansion in $t = 4x/9 - 1$.

For $4.5 \leq x < 9$,

$$\mathrm{Bi}'(x) = e^{21x/8} u(t),$$

where $u(t)$ is an expansion in $t = 4x/9 - 3$.

For $x \geq 9$,

$$\mathrm{Bi}'(x) = \sqrt[4]{x} e^z v(t),$$

where $z = \frac{2}{3}\sqrt{x^3}$ and $v(t)$ is an expansion in $t = 2\left(\frac{18}{z}\right) - 1$.

For $|x| <$ the square of the ***machine precision***, the result is set directly to $\mathrm{Bi}'(0)$. This saves time and avoids possible underflows in calculation.

For large negative arguments, it becomes impossible to calculate a result for the oscillating function with any accuracy so the routine must fail. This occurs for $x < -\left(\frac{\sqrt{\pi}}{\epsilon}\right)^{4/7}$, where ϵ is the ***machine precision***.

For large positive arguments, where Bi$'$ grows in an essentially exponential manner, there is a danger of overflow so the routine must fail.

4 References

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

5 Parameters

1: N – INTEGER *Input*

On entry: n, the number of points.

Constraint: $\mathrm{N} \geq 0$.

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

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

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

On exit: $\mathrm{Bi}'(x_i)$, the function values.

4: IVALID(N) – INTEGER array *Output*

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

IVALID$(i) = 0$
No error.

IVALID$(i) = 1$
x_i is too large and positive. F(i) contains zero. The threshold value is the same as for IFAIL $= 1$ in S17AKF, as defined in the Users' Note for your implementation.

IVALID$(i) = 2$
x_i is too large and negative. F(i) contains zero. The threshold value is the same as for IFAIL $= 2$ in S17AKF, 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 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 $= 1$

On entry, at least one value of X was invalid.
Check IVALID for more information.

IFAIL $= 2$

On entry, $\mathrm{N} = \langle \mathit{value} \rangle$.
Constraint: $\mathrm{N} \geq 0$.

7 Accuracy

For negative arguments the function is oscillatory and hence absolute error is appropriate. In the positive region the function has essentially exponential behaviour and hence relative error is needed. The absolute error, E, and the relative error ϵ, are related in principle to the relative error in the argument δ, by

$$E \simeq \left|x^2\,\mathrm{Bi}(x)\right|\delta \qquad \epsilon \simeq \left|\frac{x^2\,\mathrm{Bi}(x)}{\mathrm{Bi}'(x)}\right|\delta.$$

In practice, approximate equality is the best that can be expected. When δ, ϵ or E is of the order of the ***machine precision***, the errors in the result will be somewhat larger.

For small x, positive or negative, errors are strongly attenuated by the function and hence will effectively be bounded by the ***machine precision***.

For moderate to large negative x, the error is, like the function, oscillatory. However, the amplitude of the absolute error grows like $\frac{|x|^{7/4}}{\sqrt{\pi}}$. Therefore it becomes impossible to calculate the function with any accuracy if $|x|^{7/4} > \frac{\sqrt{\pi}}{\delta}$.

For large positive x, the relative error amplification is considerable: $\frac{\epsilon}{\delta} \sim \sqrt{x^3}$. However, very large arguments are not possible due to the danger of overflow. Thus in practice the actual amplification that occurs is limited.

8 Further Comments

None.

9 Example

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

9.1 Program Text

```
    Program s17axfe

!     S17AXF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, s17axf
!     .. 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,*) 'S17AXF 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 s17axf(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 s17axfe
```

9.2 Program Data

```
S17AXF Example Program Data

7

-10.0 -1.0 0.0 1.0 5.0 10.0 20.0
```

9.3 Program Results

```
 S17AXF Example Program Results

      X           F          IVALID

   -1.000E+01   1.194E-01    0
   -1.000E+00   5.924E-01    0
    0.000E+00   4.483E-01    0
    1.000E+00   9.324E-01    0
    5.000E+00   1.436E+03    0
    1.000E+01   1.429E+09    0
    2.000E+01   9.382E+25    0
```