

NAG Library Routine Document

S17AHF

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

S17AHF returns a value of the Airy function, $\text{Bi}(x)$, via the routine name.

2 Specification

double precision FUNCTION S17AHF(X, IFAIL)
 INTEGER IFAIL
double precision X

3 Description

S17AHF evaluates an approximation to the Airy function $\text{Bi}(x)$. It is based on a number of Chebyshev expansions.

For $x < -5$,

$$\text{Bi}(x) = \frac{a(t) \cos z + b(t) \sin z}{(-x)^{1/4}},$$

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

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

$$\text{Bi}(x) = \sqrt{3}(f(t) + xg(t)),$$

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

For $0 < x < 4.5$,

$$\text{Bi}(x) = e^{11x/8}y(t),$$

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

For $4.5 \leq x \leq 9$,

$$\text{Bi}(x) = e^{5x/2}v(t),$$

where v is an expansion in $t = 4x/9 - 3$.

For $x \geq 9$,

$$\text{Bi}(x) = \frac{e^z u(t)}{x^{1/4}},$$

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

For $|x| < \textbf{machine precision}$, the result is set directly to $\text{Bi}(0)$. This both saves time and avoids possible intermediate underflows.

For large negative arguments, it becomes impossible to calculate the phase of the oscillating function with any accuracy so the routine must fail. This occurs if $x < -\left(\frac{3}{2\epsilon}\right)^{2/3}$, where ϵ is the ***machine precision***.

For large positive arguments, there is a danger of causing overflow since Bi grows in an essentially exponential manner, 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: X – *double precision* *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.
On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

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.**

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 and positive. On soft failure, the routine returns zero. (see the Users' Note for your implementation for details)

IFAIL = 2

X is too large and negative. On soft failure, the routine returns zero. (see the Users' Note for your implementation for details)

7 Accuracy

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

$$E \simeq |xBi'(x)|\delta, \epsilon \simeq \left| \frac{xBi'(x)}{Bi(x)} \right| \delta.$$

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 , errors are strongly damped and hence will be bounded essentially by the *machine precision*.

For moderate to large negative x , the error behaviour is clearly oscillatory but the amplitude of the error grows like amplitude $\left(\frac{E}{\delta} \right) \sim \frac{|x|^{5/4}}{\sqrt{\pi}}$.

However the phase error will be growing roughly as $\frac{2}{3}\sqrt{|x|^3}$ and hence all accuracy will be lost for large negative arguments. This is due to the impossibility of calculating sin and cos to any accuracy if $\frac{2}{3}\sqrt{|x|^3} > \frac{1}{\delta}$.

For large positive arguments, the relative error amplification is considerable:

$$\frac{\epsilon}{\delta} \sim \sqrt{x^3}.$$

This means a loss of roughly two decimal places accuracy for arguments in the region of 20. However very large arguments are not possible due to the danger of causing overflow and errors are therefore limited in practice.

8 Further Comments

None.

9 Example

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

9.1 Program Text

```
*      S17AHF Example Program Text
*      Mark 14 Revised. NAG Copyright 1989.
*      .. Parameters ..
          INTEGER          NIN, NOUT
          PARAMETER        (NIN=5,NOUT=6)
*      .. Local Scalars ..
          DOUBLE PRECISION X, Y
          INTEGER          IFAIL
*      .. External Functions ..
          DOUBLE PRECISION S17AHF
          EXTERNAL         S17AHF
*      .. Executable Statements ..
          WRITE (NOUT,*) 'S17AHF Example Program Results'
*      Skip heading in data file
          READ (NIN,*)
          WRITE (NOUT,*)
          WRITE (NOUT,*) '      X              Y              IFAIL'
          WRITE (NOUT,*)
20      READ (NIN,*,END=40) X
          IFAIL = 1
*
          Y = S17AHF(X,IFAIL)
*
          IF (IFAIL.GE.0) THEN
              WRITE (NOUT,99999) X, Y, IFAIL
              GO TO 20
          ELSE
              WRITE (NOUT,99998) IFAIL
          END IF
          40 CONTINUE
*
99999 FORMAT (1X,1P,2E12.3,I7)
99998 FORMAT (1X,' ** S17AHF returned with IFAIL = ',I5)
          END
```

9.2 Program Data

S17AHF Example Program Data

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

9.3 Program Results

S17AHF Example Program Results

X	Y	IFAIL
-1.000E+01	-3.147E-01	0
-1.000E+00	1.040E-01	0
0.000E+00	6.149E-01	0
1.000E+00	1.207E+00	0
5.000E+00	6.578E+02	0
1.000E+01	4.556E+08	0
2.000E+01	2.104E+25	0
