

# NAG Library Function Document

## nag\_airy\_ai (s17agc)

### 1 Purpose

nag\_airy\_ai (s17agc) returns a value for the Airy function,  $\text{Ai}(x)$ .

### 2 Specification

```
#include <nag.h>
#include <nags.h>
double nag_airy_ai (double x, NagError *fail)
```

### 3 Description

nag\_airy\_ai (s17agc) evaluates an approximation to the Airy function,  $\text{Ai}(x)$ . It is based on a number of Chebyshev expansions:

For  $x < -5$ ,

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

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  $-5 \leq x \leq 0$ ,

$$\text{Ai}(x) = f(t) - xg(t),$$

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

For  $0 < x < 4.5$ ,

$$\text{Ai}(x) = e^{-3x/2}y(t),$$

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

For  $4.5 \leq x < 9$ ,

$$\text{Ai}(x) = e^{-5x/2}u(t),$$

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

For  $x \geq 9$ ,

$$\text{Ai}(x) = \frac{e^{-z}v(t)}{x^{1/4}},$$

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

For  $|x| < \mathbf{machine\ precision}$ , the result is set directly to  $\text{Ai}(0)$ . This both saves time and guards against underflow in intermediate calculations.

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

For large positive arguments, where  $A_i$  decays in an essentially exponential manner, there is a danger of underflow so the function must fail.

## 4 References

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

## 5 Arguments

- 1: **x** – double *Input*  
*On entry:* the argument  $x$  of the function.
- 2: **fail** – NagError \* *Input/Output*  
 The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

### NE\_INTERNAL\_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

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

See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

### NE\_NO\_LICENCE

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

See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

### NE\_REAL\_ARG\_GT

On entry,  $x = \langle \text{value} \rangle$ .

Constraint:  $x \leq \langle \text{value} \rangle$ .

$x$  is too large and positive. The function returns zero.

### NE\_REAL\_ARG\_LT

On entry,  $x = \langle \text{value} \rangle$ .

Constraint:  $x \geq \langle \text{value} \rangle$ .

$x$  is too large and negative. The function returns zero.

## 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,  $\epsilon$ , are related in principle to the relative error in the argument,  $\delta$ , by

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

In practice, approximate equality is the best that can be expected. When  $\delta$ ,  $\epsilon$  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 by the function and hence will be bounded by the *machine precision*.

For moderate negative  $x$ , the error behaviour is oscillatory but the amplitude of the error grows like

$$\text{amplitude}\left(\frac{E}{\delta}\right) \sim \frac{|x|^{5/4}}{\sqrt{\pi}}.$$

However the phase error will be growing roughly like  $\frac{2}{3}\sqrt{|x|^3}$  and hence all accuracy will be lost for large negative arguments 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 setting underflow and so the errors are limited in practice.

## 8 Parallelism and Performance

nag\_airy\_ai (s17agc) 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

```
/* nag_airy_ai (s17agc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
    Integer exit_status = 0;
    double x, y;
    NagError fail;

    INIT_FAIL(fail);

    /* Skip heading in data file */
#ifdef _WIN32
```

```

scanf_s("%*[\n]");
#else
scanf("%*[\n]");
#endif
printf("nag_airy_ai (s17agc) Example Program Results\n");
printf("      x              y\n");
#ifdef _WIN32
while (scanf_s("%lf", &x) != EOF)
#else
while (scanf("%lf", &x) != EOF)
#endif
{
/* nag_airy_ai (s17agc).
 * Airy function Ai(x)
 */
y = nag_airy_ai(x, &fail);
if (fail.code != NE_NOERROR) {
printf("Error from nag_airy_ai (s17agc).\n%s\n", fail.message);
exit_status = 1;
goto END;
}
printf("%12.3e%12.3e\n", x, y);
}
END:
return exit_status;
}

```

## 10.2 Program Data

```

nag_airy_ai (s17agc) Example Program Data
      -10.0
       -1.0
        0.0
         1.0
         5.0
        10.0
        20.0

```

## 10.3 Program Results

```

nag_airy_ai (s17agc) Example Program Results
      x              y
-1.000e+01  4.024e-02
-1.000e+00  5.356e-01
 0.000e+00  3.550e-01
 1.000e+00  1.353e-01
 5.000e+00  1.083e-04
 1.000e+01  1.105e-10
 2.000e+01  1.692e-27

```

**Example Program**  
Returns a Value for the Airy Function  $Ai(x)$

