NAG Library Function Document

nag_airy_ai_deriv (s17ajc)

1   Purpose

nag_airy_ai_deriv (s17ajc) returns a value of the derivative of the Airy function $\text{Ai}(x)$.

2   Specification

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

3   Description

nag_airy_ai_deriv (s17ajc) evaluates an approximation to the derivative of the Airy function $\text{Ai}(x)$. It is based on a number of Chebyshev expansions.

For $x < -5$,

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

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

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

$$\text{Ai}'\,(x) = x^2 f(t) - g(t),$$

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

For $0 < x < 4.5$,

$$\text{Ai}'\,(x) = e^{-11x/8}y(t),$$

where $y(t)$ is an expansion in $t = 4\left(\frac{x}{9}\right) - 1$.

For $4.5 \le x < 9$,

$$\text{Ai}'\,(x) = e^{-5x/2}v(t),$$

where $v(t)$ is an expansion in $t = 4\left(\frac{x}{9}\right) - 3$.

For $x \ge 9$,

$$\text{Ai}'\,(x) = \sqrt[4]{x}e^{-z}u(t),$$

where $z = \frac{2}{3}\sqrt{x^3}$ and $u(t)$ 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 $\text{Ai}'\,(0)$. This both saves time and avoids possible intermediate underflows.

For large negative arguments, it becomes impossible to calculate a result for the oscillating function with any accuracy and so the function 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 Ai$'$ 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 3.6 in the Essential Introduction).

6   Error Indicators and Warnings

NE_ALLOC_FAIL

   Dynamic memory allocation failed.
   See Section 3.2.1.2 in the Essential Introduction 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 3.6.6 in the Essential Introduction for further information.

NE_NO_LICENCE

   Your licence key may have expired or may not have been installed correctly.
   See Section 3.6.5 in the Essential Introduction for further information.

NE_REAL_ARG_GT

   On entry, **x** = $\langle value \rangle$.
   Constraint: **x** $\leq \langle value \rangle$.
   x is too large and positive. The function returns zero.

NE_REAL_ARG_LT

   On entry, **x** = $\langle value \rangle$.
   Constraint: **x** $\geq \langle value \rangle$.
   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 in character and here 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

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

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 be roughly bounded by the **machine precision**.

For moderate to large negative x, the error, like the function, is oscillatory; however the amplitude of the error grows like

$$\frac{|x|^{7/4}}{\sqrt{\pi}}.$$

Therefore it becomes impossible to calculate the function with any accuracy if $|x|^{7/4} > \dfrac{\sqrt{\pi}}{\delta}$.

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

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

However, very large arguments are not possible due to the danger of underflow. Thus in practice error amplification is limited.

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

```
/* nag_airy_ai_deriv (s17ajc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 2 revised, 1992.
 */

#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_deriv (s17ajc) 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_deriv (s17ajc).
       * Derivative of the Airy function Ai(x)
       */
      y = nag_airy_ai_deriv(x, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_airy_ai_deriv (s17ajc).\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_deriv (s17ajc) Example Program Data
            -10.0
            -1.0
            0.0
            1.0
            5.0
            10.0
            20.0
```

10.3 Program Results

```
nag_airy_ai_deriv (s17ajc) Example Program Results
     x            y
  -1.000e+01    9.963e-01
  -1.000e+00   -1.016e-02
   0.000e+00   -2.588e-01
   1.000e+00   -1.591e-01
   5.000e+00   -2.474e-04
   1.000e+01   -3.521e-10
   2.000e+01   -7.586e-27
```

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