NAG Library Function Document

nag_erfcx (s15agc)

1 Purpose

nag_erfcx (s15agc) returns the value of the scaled complementary error function $\mathrm{erfcx}(x)$.

2 Specification

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

3 Description

nag_erfcx (s15agc) calculates an approximate value for the scaled complementary error function

$$\mathrm{erfcx}(x) = e^{x^2}\,\mathrm{erfc}(x) = \frac{2}{\sqrt{\pi}} e^{x^2} \int_x^\infty e^{-t^2}\,dt = e^{x^2}(1 - \mathrm{erf}(x)).$$

Let $\hat{x}$ be the root of the equation $\mathrm{erfc}(x) - \mathrm{erf}(x) = 0$ (then $\hat{x} \approx 0.46875$). For $|x| \leq \hat{x}$ the value of $\mathrm{erfcx}(x)$ is based on the following rational Chebyshev expansion for $\mathrm{erf}(x)$:

$$\mathrm{erf}(x) \approx x R_{\ell,m}(x^2),$$

where $R_{\ell,m}$ denotes a rational function of degree ℓ in the numerator and m in the denominator.

For $|x| > \hat{x}$ the value of $\mathrm{erfcx}(x)$ is based on a rational Chebyshev expansion for $\mathrm{erfc}(x)$: for $\hat{x} < |x| \leq 4$ the value is based on the expansion

$$\mathrm{erfc}(x) \approx e^{x^2} R_{\ell,m}(x);$$

and for $|x| > 4$ it is based on the expansion

$$\mathrm{erfc}(x) \approx \frac{e^{x^2}}{x}\left(\frac{1}{\sqrt{\pi}} + \frac{1}{x^2} R_{\ell,m}(1/x^2)\right).$$

For each expansion, the specific values of ℓ and m are selected to be minimal such that the maximum relative error in the expansion is of the order 10^{-d}, where d is the maximum number of decimal digits that can be accurately represented for the particular implementation (see nag_decimal_digits (X02BEC)).

Asymptotically, $\mathrm{erfcx}(x) \sim 1/(\sqrt{\pi}|x|)$. There is a danger of setting underflow in $\mathrm{erfcx}(x)$ whenever $x \geq x_{\mathrm{hi}} = \min\left(x_{\mathrm{huge}}, 1/\left(\sqrt{\pi}x_{\mathrm{tiny}}\right)\right)$, where x_{huge} is the largest positive model number (see nag_real_largest_number (X02ALC)) and x_{tiny} is the smallest positive model number (see nag_real_smallest_number (X02AKC)). In this case nag_erfcx (s15agc) exits with **fail.code** = NW_HI and returns $\mathrm{erfcx}(x) = 0$. For x in the range $1/(2\sqrt{\epsilon}) \leq x < x_{\mathrm{hi}}$, where ϵ is the ***machine precision***, the asymptotic value $1/(\sqrt{\pi}|x|)$ is returned for $\mathrm{erfcx}(x)$ and nag_erfcx (s15agc) exits with **fail.code** = NW_REAL.

There is a danger of setting overflow in e^{x^2} whenever $x < x_{\mathrm{neg}} = -\sqrt{\log\left(x_{\mathrm{huge}}/2\right)}$. In this case nag_erfcx (s15agc) exits with **fail.code** = NW_NEG and returns $\mathrm{erfcx}(x) = x_{\mathrm{huge}}$.

The values of x_{hi}, $1/(2\sqrt{\epsilon})$ and x_{neg} are given in the Users' Note for your implementation.

4 References

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

Cody W J (1969) Rational Chebyshev approximations for the error function *Math.Comp.* **23** 631–637

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.

NW_HI

On entry, $\mathbf{x} = \langle value \rangle$ and the constant $x_{\mathrm{hi}} = \langle value \rangle$.
Constraint: $\mathbf{x} < x_{\mathrm{hi}}$.

NW_NEG

On entry, $\mathbf{x} = \langle value \rangle$ and the constant $x_{\mathrm{neg}} = \langle value \rangle$.
Constraint: $\mathbf{x} \geq x_{\mathrm{neg}}$.

NW_REAL

On entry, $|\mathbf{x}|$ was in the interval $[\langle value \rangle, \langle value \rangle)$ where $\mathrm{erfcx}(\mathbf{x})$ is approximately $1/(\sqrt{\pi} * |\mathbf{x}|)$: $\mathbf{x} = \langle value \rangle$.

7 Accuracy

The relative error in computing $\mathrm{erfcx}(x)$ may be estimated by evaluating

$$E = \frac{\mathrm{erfcx}(x) - e^{x^2} \displaystyle\sum_{n=1}^{\infty} I^n \,\mathrm{erfc}(x)}{\mathrm{erfcx}(x)},$$

where I^n denotes repeated integration. Empirical results suggest that on the interval $(\hat{x}, 2)$ the loss in base b significant digits for maximum relative error is around 3.3, while for root-mean-square relative error on that interval it is 1.2 (see nag_real_base (X02BHC) for the definition of the model parameter b). On the interval $(2, 20)$ the values are around 3.5 for maximum and 0.45 for root-mean-square relative errors; note that on these two intervals $\text{erfc}(x)$ is the primary computation. See also Section 7 in nag_erfc (s15adc).

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_erfcx (s15agc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 9, 2009.
 */
/* Pre-processor includes */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nags.h>
int main(void)
{
  /*Integer scalar and array declarations */
  Integer    exit_status = 0;
  /*Double scalar and array declarations */
  double     x, y;

  NagError   fail;
  const char *str_fail;

  INIT_FAIL(fail);

  printf("nag_erfcx (s15agc) Example Program Results\n");
  /* Skip heading in data file*/
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif
  printf("\n%s\n\n", "       x           erfcx(x)     fail");
#ifdef _WIN32
  while (scanf_s("%lf%*[^\n] ", &x) != EOF)
#else
  while (scanf("%lf%*[^\n] ", &x) != EOF)
#endif
    {
      /*
       * nag_erfcx (s15agc)
       * Scaled complement of error function, erfcx(x)
       */
      y = nag_erfcx(x, &fail);
      if (fail.code != NE_NOERROR)
        {
          if (fail.code == NW_HI || fail.code == NW_NEG ||
              fail.code == NW_REAL)
```

```
            {
              /* nag_code_to_error_name (x04ndc).
               * Converts NAG error code to its string value
               */
              str_fail = nag_code_to_error_name(fail.code);
              printf("%14.5e  %-14.5e %s\n", x, y, str_fail);
            }
          else
            {
              printf("Error from nag_erfcx (s15agc).\n%s\n",
                      fail.message);
              exit_status = 1;
              goto END;
            }
        }
      else
        {
          printf("%14.5e  %-14.5e\n", x, y);
        }
    }

 END:

  return exit_status;
}
```

10.2 Program Data

```
nag_erfcx (s15agc) Example Program Data
            -30.0
            -6.0
            -4.5
            -1.0
             1.0
             4.5
             6.0
             70000000.0
```

10.3 Program Results

```
nag_erfcx (s15agc) Example Program Results

       x          erfcx(x)     fail

 -3.00000e+01  1.79769e+308    NW_NEG
 -6.00000e+00  8.62246e+15
 -4.50000e+00  1.24593e+09
 -1.00000e+00  5.00898e+00
  1.00000e+00  4.27584e-01
  4.50000e+00  1.22485e-01
  6.00000e+00  9.27766e-02
  7.00000e+07  8.05985e-09     NW_REAL
```