NAG Library Function Document

nag_specfun_1f1_real (s22bac)

1 Purpose

nag_specfun_1f1_real (s22bac) returns a value for the confluent hypergeometric function ${}_1F_1(a;b;x)$ with real parameters a and b, and real argument x. This function is sometimes also known as Kummer's function $M(a,b,x)$.

2 Specification

```
#include <nag.h>
#include <nags.h>
void nag_specfun_1f1_real (double a, double b, double x, double *m,
     NagError *fail)
```

3 Description

nag_specfun_1f1_real (s22bac) returns a value for the confluent hypergeometric function ${}_1F_1(a;b;x)$ with real parameters a and b, and real argument x. This function is unbounded or not uniquely defined for b equal to zero or a negative integer.

The associated function nag_specfun_1f1_real_scaled (s22bbc) performs the same operations, but returns M in the scaled form $M = m_f \times 2^{m_s}$ to allow calculations to be performed when M is not representable as a single working precision number. It also accepts the parameters a and b as summations of an integer and a decimal fraction, giving higher accuracy when a or b are close to an integer. In such cases, nag_specfun_1f1_real_scaled (s22bbc) should be used when high accuracy is required.

The confluent hypergeometric function is defined by the confluent series

$$ {}_1F_1(a;b;x) = M(a,b,x) = \sum_{s=0}^{\infty} \frac{(a)_s x^s}{(b)_s s!} = 1 + \frac{a}{b}x + \frac{a(a+1)}{b(b+1)2!}x^2 + \cdots $$

where $(a)_s = 1(a)(a+1)(a+2)\ldots(a+s-1)$ is the rising factorial of a. $M(a,b,x)$ is a solution to the second order ODE (Kummer's Equation):

$$ x\frac{d^2M}{dx^2} + (b-x)\frac{dM}{dx} - aM = 0. \tag{1} $$

Given the parameters and argument (a,b,x), this function determines a set of safe values $\{(\alpha_i,\beta_i,\zeta_i) \mid i \leq 2\}$ and selects an appropriate algorithm to accurately evaluate the functions $M_i(\alpha_i,\beta_i,\zeta_i)$. The result is then used to construct the solution to the original problem $M(a,b,x)$ using, where necessary, recurrence relations and/or continuation.

Additionally, an artificial bound, $arbnd$ is placed on the magnitudes of a, b and x to minimize the occurrence of overflow in internal calculations. $arbnd = 0.0001 \times I_{\max}$, where $I_{\max} =$ X02BBC. It should, however, not be assumed that this function will produce an accurate result for all values of a, b and x satisfying this criterion.

Please consult the NIST Digital Library of Mathematical Functions or the companion (2010) for a detailed discussion of the confluent hypergeoemtric function including special cases, transformations, relations and asymptotic approximations.

4 References

NIST Handbook of Mathematical Functions (2010) (eds F W J Olver, D W Lozier, R F Boisvert, C W Clark) Cambridge University Press

Pearson J (2009) Computation of hypergeometric functions *MSc Dissertation, Mathematical Institute, University of Oxford*

5 Arguments

1: **a** – double *Input*

On entry: the parameter a of the function.

Constraint: $|\mathbf{a}| \leq arbnd$.

2: **b** – double *Input*

On entry: the parameter b of the function.

Constraint: $|\mathbf{b}| \leq arbnd$.

3: **x** – double *Input*

On entry: the argument x of the function.

Constraint: $|\mathbf{x}| \leq arbnd$.

4: **m** – double * *Output*

On exit: the solution $M(a, b, x)$.

Note: if overflow occurs upon completion, as indicated by **fail**.**code** = NW_OVERFLOW_WARN , $|M(a, b, x)|$ may be assumed to be too large to be representable. **m** will be returned as $\pm R_{\max}$, where $R_{\max}$ is the largest representable real number (see nag_real_largest_number (X02ALC)). The sign of **m** should match the sign of $M(a, b, x)$. If overflow occurs during a subcalculation, as indicated by **fail**.**code** = NE_OVERFLOW, the sign may be incorrect, and the true value of $M(a, b, x)$ may or may not be greater than $R_{\max}$. In either case it is advisable to subsequently use nag_specfun_1f1_real_scaled (s22bbc).

5: **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_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

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_OVERFLOW

Overflow occurred in a subcalculation of $M(a, b, x)$.
The answer may be completely incorrect.

NE_REAL

On entry, **b** $= \langle value \rangle$.
$M(a, b, x)$ is undefined when b is zero or a negative integer.

NE_REAL_RANGE_CONS

On entry, **a** $= \langle value \rangle$.
Constraint: $|\mathbf{a}| \leq arbnd = \langle value \rangle$.

On entry, **b** $= \langle value \rangle$.
Constraint: $|\mathbf{b}| \leq arbnd = \langle value \rangle$.

On entry, **x** $= \langle value \rangle$.
Constraint: $|\mathbf{x}| \leq arbnd = \langle value \rangle$.

NE_TOTAL_PRECISION_LOSS

All approximations have completed, and the final residual estimate indicates no accuracy can be guaranteed.
Relative residual $= \langle value \rangle$.

NW_OVERFLOW_WARN

On completion, overflow occurred in the evaluation of $M(a, b, x)$.

NW_SOME_PRECISION_LOSS

All approximations have completed, and the final residual estimate indicates some precision may have been lost.
Relative residual $= \langle value \rangle$.

NW_UNDERFLOW_WARN

Underflow occurred during the evaluation of $M(a, b, x)$.
The returned value may be inaccurate.

7 Accuracy

In general, if **fail.code** = NE_NOERROR, the value of M may be assumed accurate, with the possible loss of one or two decimal places. Assuming the result does not under or overflow, an error estimate res is made internally using equation (1). If the magnitude of res is sufficiently large, a different **fail.code** will be returned. Specifically,

fail.code = NE_NOERROR	$res \leq 1000\epsilon$
fail.code = NW_SOME_PRECISION_LOSS	$1000\epsilon < res \leq 0.1$
fail.code = NE_TOTAL_PRECISION_LOSS	$res > 0.1$

where ϵ is the ***machine precision*** as returned by nag_machine_precision (X02AJC).

A further estimate of the residual can be constructed using equation (1), and the differential identity,

$$\frac{dM(a,b,x)}{dx} = \frac{a}{b}M(a+1,b+1,x),$$

$$\frac{d^2M(a,b,x)}{dx^2} = \frac{a(a+1)}{b(b+1)}M(a+2,b+2,x).$$

This estimate is however dependent upon the error involved in approximating $M(a+1,b+1,x)$ and $M(a+2,b+2,x)$.

Furthermore, the accuracy of the solution, and the error estimate, can be dependent upon the accuracy of the decimal fraction of the input parameters a and b. For example, if $b = b_i + b_r = 100 + 1.0\text{e}{-6}$, then on a machine with 16 decimal digits of precision, the internal calculation of b_r will only be accurate to 8 decimal places. This can subsequently pollute the final solution by several decimal places without affecting the residual estimate as greatly. Should you require higher accuracy in such regions, then you should use nag_specfun_1f1_real_scaled (s22bbc), which requires you to supply the correct decimal fraction.

8 Parallelism and Performance

nag_specfun_1f1_real (s22bac) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_specfun_1f1_real (s22bac) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

None.

10 Example

This example prints the results returned by nag_specfun_1f1_real (s22bac) called using parameters $a = 13.6$ and $b = 14.2$ with 11 differing values of argument x.

10.1 Program Text

```
/* nag_specfun_1f1_real (s22bac) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 24, 2013.
 */

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

void construct_table(double a, double b);

int main(void)
{
  /* Scalars */
  Integer  exit_status = 0;
  Integer  kx;
  double   a, b, m, x;
  /* Nag Types */
  NagError fail;
```

```
  INIT_FAIL(fail);

  printf("nag_specfun_1f1_real (s22bac) Example Program Results\n\n");

  a = 13.6;
  b = 14.2;

  construct_table(a, b);

  for (kx = -5; kx < 6; kx++)
    {
      x = (double) kx + 0.5;
      /* Evaluate Real confluent hypergeometric function M(a,b,x) using
       * nag_specfun_1f1_real (s22bac).
       */
      nag_specfun_1f1_real(a, b, x, &m, &fail);
      printf("%13.2f  ", x);
      switch (fail.code) {
      case NE_NOERROR:
      case NW_UNDERFLOW_WARN:
      case NW_SOME_PRECISION_LOSS:
        {
          printf("%13.5e\n", m);
          break;
        }
      default:
        {
          printf("No finite or sufficiently accurate result.\n");
          exit_status = 1;
          break;
        }
      }
    }
  return exit_status;
}

void construct_table(double a, double b)
{
  printf("          a              b\n");
  printf("+--------------+--------------+\n");
  printf("%13.2f  %13.2f\n", a, b);
  printf("+--------------+--------------+\n");
  printf("          x        M(a,b,x)\n");
  printf("+--------------+--------------+\n\n");
  return;
}
```

10.2 Program Data

None.

10.3 Program Results

```
nag_specfun_1f1_real (s22bac) Example Program Results

          a              b
+--------------+--------------+
         13.60          14.20
+--------------+--------------+
          x        M(a,b,x)
+--------------+--------------+

        -4.50    1.38786e-02
        -3.50    3.56741e-02
        -2.50    9.20723e-02
        -1.50    2.38486e-01
        -0.50    6.19691e-01
         0.50    1.61478e+00
```

```
        1.50    4.21840e+00
        2.50    1.10449e+01
        3.50    2.89776e+01
        4.50    7.61660e+01
        5.50    2.00533e+02
```