NAG C Library Function Document

nag_bessel_k_alpha_scaled (s18ehc)

1 Purpose

nag_bessel_k_alpha_scaled (s18ehc) returns a sequence of values for the scaled modified Bessel functions $e^x K_{\alpha+n}(x)$ for real $x > 0$, selected values of $\alpha \geq 0$ and $n = 0, 1, \ldots, N$.

2 Specification

```
void nag_bessel_k_alpha_scaled (double x, Integer ia, Integer ja, Integer nl,
      double b[], NagError *fail)
```

3 Description

This routine evaluates a sequence of values for the scaled modified Bessel function of the second kind $e^x K_\alpha(x)$, where x is real and non-negative and $\alpha \in \{0, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{2}{3}, \frac{3}{4}\}$ is the order. The $(N+1)$-member sequence is generated for orders $\alpha, \alpha+1, \ldots, \alpha+N$.

4 Parameters

1: **x** – double *Input*

On entry: the argument x of the function.

Constraint: **x** $>$ 0.0.

2: **ia** – Integer *Input*

3: **ja** – Integer *Input*

On entry: the numerator i and denominator j, respectively, of the order $\alpha = i/j$ of the first member in the required sequence of function values. Only the following combinations of pairs of values of i and j are allowed:

$i = 0$ and $j = 1$ corresponds to $\alpha = 0$;

$i = 1$ and $j = 2$ corresponds to $\alpha = \frac{1}{2}$;

$i = 1$ and $j = 3$ corresponds to $\alpha = \frac{1}{3}$;

$i = 1$ and $j = 4$ corresponds to $\alpha = \frac{1}{4}$;

$i = 2$ and $j = 3$ corresponds to $\alpha = \frac{2}{3}$;

$i = 3$ and $j = 4$ corresponds to $\alpha = \frac{3}{4}$.

Constraint: **ia** and **ja** must constitute a valid pair (**ia**,**ja**) = (0,1), (1,2), (1,3), (1,4), (2,3) or (3,4).

4: **nl** – Integer *Input*

On entry: the value of N. Note that the order of the last member in the required sequence of function values is given by $\alpha + N$.

Constraint: $0 \leq$ **nl** ≤ 100.

5: **b[nl+1]** – double *Output*

On exit: with **fail.code** = **NE_NOERROR** or **fail.code** = **NW_SOME_PRECISION_LOSS**, the required sequence of function values: **b**(n) contains $K_{\alpha+n}(x)$ for $n = 0, 1, \ldots, N$.

6: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

NE_REAL

On entry, **x** = <*value*>.
Constraint: **x** > 0.0.

NE_INT

On entry, **nl** = <*value*>.
Constraint: 0 ≤ **nl** ≤ 100.

NE_INT_2

On entry, **ia** = <*value*>, **ja** = <*value*>.
Constraint: **ia** and **ja** must constitute a valid pair (**ia**,**ja**).

NE_OVERFLOW_LIKELY

The evaluation has been abandoned due to the likelihood of overflow.

NW_SOME_PRECISION_LOSS

The evaluation has been completed but some precision has been lost.

NE_TOTAL_PRECISION_LOSS

The evaluation has been abandoned due to total loss of precision.

NE_TERMINATION_FAILURE

The evaluation has been abandoned due to failure to satisfy the termination condition.

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 consult NAG for assistance.

6 Further Comments

6.1 Accuracy

All constants in the underlying function are specified to approximately 18 digits of precision. If t denotes the number of digits of precision in the floating-point arithmetic being used, then clearly the maximum number of correct digits in the results obtained is limited by $p = \min(t, 18)$. Because of errors in argument reduction when computing elementary functions inside the underlying function, the actual number of correct digits is limited, in general, by $p - s$, where $s \approx \max(1, |\log_{10} x|)$ represents the number of digits lost due to the argument reduction. Thus the larger the value of x, the less the precision in the result.

6.2 References

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

7 See Also

None.

8 Example

The example program evaluates $e^x K_0(x), e^x K_1(x), e^x K_2(x)$ and $e^x K_3(x)$ at $x = 0.5$, and prints the results.

8.1 Program Text

```
/* nag_bessel_k_alpha_scaled (s18ehc) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * NAG C Library
 *
 * Mark 6, 2000.
 */

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


int main(void)
{
  double alpha;
  double b[101];
  double x;
  Integer i;
  Integer ia;
  Integer exit_status=0;
  Integer ja;
  Integer nl;

  NagError fail;

  INIT_FAIL(fail);
  Vprintf("s18ehc Example Program Results\n\n");
  /*     Skip heading in data file */
  Vscanf("%*[^\n]");
  while (scanf("%lf %ld %ld %ld%*[^\n]", &x, &ia, &ja, &nl) != EOF)
    {
      Vprintf("\n  x     ia    ja    nl\n\n");
      Vprintf("%4.1f%6ld%6ld%6ld\n\n", x, ia, ja, nl);
      s18ehc  (x, ia, ja, nl, b, &fail);
      if (fail.code == NE_NOERROR)
        {
          Vprintf("\n Requested values of exp(X)*K_alpha(X)\n\n");
          alpha = (double) ia / (double) ja;
          Vprintf("     alpha      exp(X)*K_alpha(X)\n");
          for (i = 0; i <= nl; ++i)
            {
              Vprintf(" %12.4e      %12.4e\n", alpha, b[i]);
              alpha += 1.;
            }
        }
      else
        {
          Vprintf("Error from s18ehc.\n%s\n", fail.message);
```

```
          exit_status = 1;
          goto END;
        }
    }
 END:
  return exit_status;
}
```

8.2 Program Data

```
s18ehc Example Program Data
 0.5   0   1   3 : Values of x, ia, ja and nl
```

8.3 Program Results

```
s18ehc Example Program Results


  x     ia    ja    nl

 0.5     0     1     3


 Requested values of exp(X)*K_alpha(X)

     alpha      exp(X)*K_alpha(X)
   0.0000e+00        1.5241e+00
   1.0000e+00        2.7310e+00
   2.0000e+00        1.2448e+01
   3.0000e+00        1.0232e+02
```