

## NAG 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  $n = 0, 1, \dots, N$ .

#### 2 Specification

```
#include <nag.h>
#include <nags.h>

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

#### 3 Description

nag\_bessel\_k\_alpha\_scaled (s18ehc) 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  $\alpha, \alpha+1, \dots, \alpha+N$ .

#### 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.

*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  $(\mathbf{ia}, \mathbf{ja}) = (0, 1), (1, 2), (1, 3), (1, 4), (2, 3)$  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:*  $0 \leq \mathbf{nl} \leq 100$ .
- 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  $n = 0, 1, \dots, N$ .
- 6: **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\_INT

On entry, **nl** =  $\langle value \rangle$ .  
Constraint:  $0 \leq \mathbf{nl} \leq 100$ .

### NE\_INT\_2

On entry, **ia** =  $\langle value \rangle$ , **ja** =  $\langle value \rangle$ .  
Constraint: **ia** and **ja** must constitute a valid pair (**ia,ja**).

### 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.

### NE\_OVERFLOW\_LIKELY

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

### NE\_REAL

On entry, **x** =  $\langle value \rangle$ .  
Constraint: **x** > 0.0.

### NE\_TERMINATION\_FAILURE

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

### NE\_TOTAL\_PRECISION\_LOSS

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

### NW\_SOME\_PRECISION\_LOSS

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

## 7 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.

## 8 Parallelism and Performance

nag\_bessel\_k\_alpha\_scaled (s18ehc) is not threaded in any implementation.

## 9 Further Comments

None.

## 10 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.

### 10.1 Program Text

```

/* nag_bessel_k_alpha_scaled (s18ehc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * NAG C Library
 *
 * Mark 26, 2016.
 */

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

int main(void)
{
    Integer exit_status = 0, i, ia, ja, nl;
    NagError fail;
    double alpha, *b = 0, x;

    INIT_FAIL(fail);

    /* Skip heading in data file */
#ifdef _WIN32
    scanf_s("%*[\n]");
#else
    scanf("%*[\n]");
#endif
    printf("nag_bessel_k_alpha_scaled (s18ehc) Example Program Results\n");
    if (!(b = NAG_ALLOC(101, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

#ifdef _WIN32
    while (scanf_s
        ("%lf %" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "%*[\n]", &x, &ia,
         &ja, &nl) != EOF) {
#else
    while (scanf
        ("%lf %" NAG_IFMT " %" NAG_IFMT " %" NAG_IFMT "%*[\n]", &x, &ia,
         &ja, &nl) != EOF) {
#endif
}

```

```

printf("  x      ia      ja      nl\n");
printf("%4.1f%6" NAG_IFMT "%6" NAG_IFMT "%6" NAG_IFMT "\n\n", x, ia, ja,
      nl);
/* nag_bessel_k_alpha_scaled (s18ehc).
 * Scaled modified Bessel functions K_(alpha+n)(x) for
 * real x > 0, selected values of alpha >= 0 and
 * n = 0,1,...,N
 */
nag_bessel_k_alpha_scaled(x, ia, ja, nl, b, &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_bessel_k_alpha_scaled (s18ehc).\n%s\n",
          fail.message);
    exit_status = 1;
    goto END;
}
printf(" Requested values of exp(x)*K_alpha(x)\n\n");
alpha = (double) ia / (double) ja;
printf("      alpha      exp(x)*K_alpha(x)\n");
for (i = 0; i <= nl; ++i) {
    printf(" %13.4e      %13.4e\n", alpha, b[i]);
    alpha += 1.0;
}
}
END:
    NAG_FREE(b);
    return exit_status;
}

```

## 10.2 Program Data

nag\_bessel\_k\_alpha\_scaled (s18ehc) Example Program Data  
0.5 0 1 3 : Values of x, ia, ja and nl

## 10.3 Program Results

nag\_bessel\_k\_alpha\_scaled (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

---