NAG C Library Function Document

nag_bessel_k_nu (s18ecf)

1 Purpose

nag_bessel_k_nu (s18ecf) returns the value of the modified Bessel function $K_{\nu/4}(x)$ for real $x > 0$.

2 Specification

double nag_bessel_k_nu (double x, Integer nu, NagError *fail)

3 Description

This routine evaluates an approximation to the modified Bessel function of the second kind $K_{\nu/4}(x)$, where the order $\nu = -3, -2, -1, 1, 2$ or 3 and $x$ is real and positive. For negative orders the formula

$$K_{-\nu/4}(x) = K_{\nu/4}(x)$$

is used.

4 Parameters

1: \hspace{10pt} x – double \hspace{10pt} \text{Input}

On entry: the argument $x$ of the function.

Constraint: $x > 0.0$.

2: \hspace{10pt} nu – Integer \hspace{10pt} \text{Input}

On entry: the argument $\nu$ of the function.

Constraint: $1 \leq \text{abs(nu)} \leq 3$.

3: \hspace{10pt} fail – NagError * \hspace{10pt} \text{Input/Output}

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

\textbf{NE_REAL}

On entry, $x = \text{<value>}$.

Constraint: $x > 0.0$.

\textbf{NE_INT}

On entry, $\text{nu} = \text{<value>}$.

Constraint: $1 \leq \text{abs(nu)} \leq 3$.

\textbf{NE_OVERFLOWLIKELY}

The evaluation has been abandoned due to the likelihood of overflow. The result is returned as zero.

\textbf{NW SOME PRECISION LOSS}

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

The evaluation has been abandoned due to total loss of precision. The result is returned as zero.

NE_TERMINATION_FAILURE

The evaluation has been abandoned due to failure to satisfy the termination condition. The result is returned as zero.

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


7 See Also

None.

8 Example

The example program reads values of the arguments $x$ and $\nu$ from a file, evaluates the function and prints the results.

8.1 Program Text

/* nag_bessel_k_nu (s18fc) Example Program.

Copyright 2000 Numerical Algorithms Group.

NAG C Library

Mark 6, 2000.
*/

#include <stdio.h>
#include <nag.h>
#include <nag_stdbib.h>
#include <nags.h>

int main(void)
{
    double x;
    double y;
}
Integer exit_status=0;
Integer nu;
NagError fail;

INIT_FAIL(fail);
Vprintf("sl8efc Example Program Results\n\n");
/* Skip heading in data file */
Vscanf("%*[\n]");
Vprintf("\n x     nu     y\n\n");
while (scanf("%lf %ld%[\n]", &x, &nu) != EOF)
{
    y = sl8efc (x, nu, &fail);
    if (fail.code == NE_NOERROR)
    Vprintf("%4.1f %lld %12.4e\n", x, nu, y);
    else
    {
        Vprintf("Error from sl8efc.\n%s\n", fail.message);
        exit_status = 1;
        goto END;
    }
}
END:
return exit_status;
}

8.2 Program Data

sl8efc Example Program Data
  3.9  -3
  1.4  -2
  8.2  -1
  6.7   1
  0.5   2
  2.3   3 : Values of x and nu

8.3 Program Results

sl8efc Example Program Results

<table>
<thead>
<tr>
<th>x</th>
<th>nu</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.9</td>
<td>-3</td>
<td>1.3315e-02</td>
</tr>
<tr>
<td>1.4</td>
<td>-2</td>
<td>2.6121e-01</td>
</tr>
<tr>
<td>8.2</td>
<td>-1</td>
<td>1.1892e-04</td>
</tr>
<tr>
<td>6.7</td>
<td>1</td>
<td>5.8826e-04</td>
</tr>
<tr>
<td>0.5</td>
<td>2</td>
<td>1.0750e+00</td>
</tr>
<tr>
<td>2.3</td>
<td>3</td>
<td>8.7724e-02</td>
</tr>
</tbody>
</table>