NAG Library Function Document

nag_kelvin_bei (s19abc)

1 Purpose
nag_kelvin_bei (s19abc) returns a value for the Kelvin function bei\(x\).

2 Specification
#include <nag.h>
#include <nags.h>

double nag_kelvin_bei (double x, NagError *fail)

3 Description
nag_kelvin_bei (s19abc) evaluates an approximation to the Kelvin function bei\(x\).

Note: bei\((-x) = bei\(x\), so the approximation need only consider \(x \geq 0.0\).

The function is based on several Chebyshev expansions:

For \(0 \leq x \leq 5\),
\[
bei(x) = \frac{x^2}{4} \sum_{r=0}^{t} a_r T_r(t), \quad \text{with } t = 2 \left(\frac{x}{5}\right)^4 - 1;
\]

For \(x > 5\),
\[
bei(x) = \frac{e^{x/\sqrt{2}}}{\sqrt{2\pi x}} \left[\left(1 + \frac{1}{x} a(t)\right) \sin \alpha - \frac{1}{x} b(t) \cos \alpha\right]
\]
\[
+ \frac{e^{x/\sqrt{2}}}{\sqrt{2\pi x}} \left[\left(1 + \frac{1}{x} c(t)\right) \cos \beta - \frac{1}{x} d(t) \sin \beta\right]
\]

where \(\alpha = \frac{x}{\sqrt{2}} - \frac{\pi}{8}\), \(\beta = \frac{x}{\sqrt{2}} + \frac{\pi}{8}\),

and \(a(t), b(t), c(t), \text{ and } d(t)\) are expansions in the variable \(t = \frac{10}{x} - 1\).

When \(x\) is sufficiently close to zero, the result is computed as bei\(x = \frac{x^2}{4}\). If this result would underflow, the result returned is bei\(x = 0.0\).

For large \(x\), there is a danger of the result being totally inaccurate, as the error amplification factor grows in an essentially exponential manner; therefore the function must fail.

4 References
5 Arguments

1: x – double
   \textit{Input}
   \textit{On entry}: the argument $x$ of the function.

2: fail – NagError *
   \textit{Input/Output}
   The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

\textbf{NE_ALLOC_FAIL}

Dynamic memory allocation failed.
See Section 3.2.1.2 in the Essential Introduction for further information.

\textbf{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.

\textbf{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.

\textbf{NE_REAL_ARG_GT}

On entry, $x = \langle\text{value}\rangle$.
Constraint: $|x| \leq \langle\text{value}\rangle$.
$|x|$ is too large for an accurate result to be returned and the function returns zero.

7 Accuracy

Since the function is oscillatory, the absolute error rather than the relative error is important. Let $E$ be
the absolute error in the function, and $\delta$ be the relative error in the argument. If $\delta$ is somewhat larger
than the \textit{machine precision}, then we have:

$$ E \simeq \frac{x}{\sqrt{2}} (\text{ber}_1 x + \text{bei}_1 x) \left|\delta\right| $$

(provided $E$ is within machine bounds).
For small $x$ the error amplification is insignificant and thus the absolute error is effectively bounded by
the \textit{machine precision}.

For medium and large $x$, the error behaviour is oscillatory and its amplitude grows like $\sqrt{\frac{x}{2\pi}} e^{x/\sqrt{2}}$.
Therefore it is impossible to calculate the functions with any accuracy when $\sqrt{x} e^{x/\sqrt{2}} > \frac{\sqrt{2\pi}}{\delta}$. Note that
this value of $x$ is much smaller than the minimum value of $x$ for which the function overflows.

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_kelvin_bei (s19abc) Example Program.
   * Copyright 2014 Numerical Algorithms Group.
   * Mark 4, 1996.
   */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>

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

    /* Skip heading in data file */
    #ifdef _WIN32
    scanf_s("%*[\n"]);
    #else
    scanf("%*[\n"]);
    #endif
    printf("nag_kelvin_bei (s19abc) Example Program Results
");
    printf(" x y
");
    #ifdef _WIN32
    while (scanf_s("%lf", &x) != EOF)
    #else
    while (scanf("%lf", &x) != EOF)
    #endif
    {
        /* nag_kelvin_bei (s19abc).
           * Kelvin function bei x
           */
        y = nag_kelvin_bei(x, &fail);
        if (fail.code != NE_NOERROR) {
            printf("Error from nag_kelvin_bei (s19abc).\n", fail.message);
            exit_status = 1;
            goto END;
        }
        printf("%12.3e%12.3e
", x, y);
    }
    END:
    return exit_status;
}
10.2 Program Data

nag_kelvin_bei (s19abc) Example Program Data
0.1
1.0
2.5
5.0
10.0
15.0
-1.0

10.3 Program Results

nag_kelvin_bei (s19abc) Example Program Results
x        y
1.000e-01 2.500e-03
1.000e+00 2.496e-01
2.500e+00 1.457e+00
5.000e+00 1.160e-01
1.000e+01 5.637e+01
1.500e+01 -2.953e+03
-1.000e+00 2.496e-01