NAG Library Function Document

nag_kelvin_ker_vector (s19aqc)

1 Purpose

nag_kelvin_ker_vector (s19aqc) returns an array of values for the Kelvin function $\ker x$.

2 Specification

```
#include <nag.h>
#include <nags.h>
void nag_kelvin_ker_vector (Integer n, const double x[], double f[],
     Integer ivalid[], NagError *fail)
```

3 Description

nag_kelvin_ker_vector (s19aqc) evaluates an approximation to the Kelvin function $\ker x_i$ for an array of arguments x_i, for $i = 1, 2, \ldots, n$.

Note: for $x < 0$ the function is undefined and at $x = 0$ it is infinite so we need only consider $x > 0$.

The function is based on several Chebyshev expansions:

For $0 < x \leq 1$,

$$\ker x = -f(t)\log(x) + \frac{\pi}{16}x^2 g(t) + y(t)$$

where $f(t)$, $g(t)$ and $y(t)$ are expansions in the variable $t = 2x^4 - 1$.

For $1 < x \leq 3$,

$$\ker x = \exp\left(-\frac{11}{16}x\right)q(t)$$

where $q(t)$ is an expansion in the variable $t = x - 2$.

For $x > 3$,

$$\ker x = \sqrt{\frac{\pi}{2x}}e^{-x/\sqrt{2}}\left[\left(1 + \frac{1}{x}c(t)\right)\cos\beta - \frac{1}{x}d(t)\sin\beta\right]$$

where $\beta = \dfrac{x}{\sqrt{2}} + \dfrac{\pi}{8}$, and $c(t)$ and $d(t)$ are expansions in the variable $t = \dfrac{6}{x} - 1$.

When x is sufficiently close to zero, the result is computed as

$$\ker x = -\gamma - \log\left(\frac{x}{2}\right) + \left(\pi - \frac{3}{8}x^2\right)\frac{x^2}{16}$$

and when x is even closer to zero, simply as $\ker x = -\gamma - \log\left(\dfrac{x}{2}\right)$.

For large x, $\ker x$ is asymptotically given by $\sqrt{\dfrac{\pi}{2x}}e^{-x/\sqrt{2}}$ and this becomes so small that it cannot be computed without underflow and the function fails.

4 References

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

5 Arguments

1: **n** – Integer *Input*

On entry: n, the number of points.

Constraint: $\mathbf{n} \geq 0$.

2: **x**[**n**] – const double *Input*

On entry: the argument x_i of the function, for $i = 1, 2, \ldots, \mathbf{n}$.

Constraint: $\mathbf{x}[i-1] > 0.0$, for $i = 1, 2, \ldots, \mathbf{n}$.

3: **f**[**n**] – double *Output*

On exit: $\ker x_i$, the function values.

4: **ivalid**[**n**] – Integer *Output*

On exit: **ivalid**$[i-1]$ contains the error code for x_i, for $i = 1, 2, \ldots, \mathbf{n}$.

ivalid$[i-1] = 0$
No error.

ivalid$[i-1] = 1$
x_i is too large, the result underflows. **f**$[i-1]$ contains zero. The threshold value is the same as for **fail.code** = NE_REAL_ARG_GT in nag_kelvin_ker (s19acc), as defined in the Users' Note for your implementation.

ivalid$[i-1] = 2$
$x_i \leq 0.0$, the function is undefined. **f**$[i-1]$ contains 0.0.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_BAD_PARAM

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

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: $\mathbf{n} \geq 0$.

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.

NW_IVALID

On entry, at least one value of **x** was invalid.
Check **ivalid** for more information.

7 Accuracy

Let E be the absolute error in the result, ϵ be the relative error in the result and δ be the relative error in the argument. If δ is somewhat larger than the ***machine precision***, then we have:

$$E \simeq \left| \frac{x}{\sqrt{2}} (\mathrm{ker}_1\, x + \mathrm{kei}_1\, x) \right| \delta,$$

$$\epsilon \simeq \left| \frac{x}{\sqrt{2}} \frac{\mathrm{ker}_1\, x + \mathrm{kei}_1\, x}{\mathrm{ker}\, x} \right| \delta.$$

For very small x, the relative error amplification factor is approximately given by $\dfrac{1}{|\log(x)|}$, which implies a strong attenuation of relative error. However, ϵ in general cannot be less than the ***machine precision***.

For small x, errors are damped by the function and hence are limited by the ***machine precision***.

For medium and large x, the error behaviour, like the function itself, is oscillatory, and hence only the absolute accuracy for the function can be maintained. For this range of x, the amplitude of the absolute error decays like $\sqrt{\dfrac{\pi x}{2}} e^{-x/\sqrt{2}}$ which implies a strong attenuation of error. Eventually, $\mathrm{ker}\, x$, which asymptotically behaves like $\sqrt{\dfrac{\pi}{2x}} e^{-x/\sqrt{2}}$, becomes so small that it cannot be calculated without causing underflow, and the function returns zero. Note that for large x the errors are dominated by those of the standard function exp.

8 Parallelism and Performance

Not applicable.

9 Further Comments

Underflow may occur for a few values of x close to the zeros of $\mathrm{ker}\, x$, below the limit which causes a failure with **fail.code** = NW_IVALID.

10 Example

This example reads values of **x** from a file, evaluates the function at each value of x_i and prints the results.

10.1 Program Text

```
/* nag_kelvin_ker_vector (s19aqc) Example Program.
 *
 * Copyright 2011, Numerical Algorithms Group.
 *
 * Mark 23 2011.
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>

int main(void)
{
  Integer  exit_status = 0;
  Integer  i, n;
  double   *f = 0, *x = 0;
  Integer  *ivalid = 0;
  NagError fail;
```

```
  INIT_FAIL(fail);

  /* Skip heading in data file */
  scanf("%*[^\n]");

  printf("nag_kelvin_ker_vector (s19aqc) Example Program Results\n");
  printf("\n");
  printf("     x           f          ivalid\n");
  printf("\n");
  scanf("%ld", &n);
  scanf("%*[^\n]");

  /* Allocate memory */
  if (!(x = NAG_ALLOC(n, double)) ||
      !(f = NAG_ALLOC(n, double)) ||
      !(ivalid = NAG_ALLOC(n, Integer)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  for (i=0; i<n; i++)
    scanf("%lf", &x[i]);
  scanf("%*[^\n]");

  /* nag_kelvin_ker_vector (s19aqc).
   * Kelvin Function ker x
   */
  nag_kelvin_ker_vector(n, x, f, ivalid, &fail);
  if (fail.code!=NE_NOERROR && fail.code!=NW_IVALID)
    {
      printf("Error from nag_kelvin_ker_vector (s19aqc).\n%s\n",
             fail.message);
      exit_status = 1;
      goto END;
    }

  for (i=0; i<n; i++)
    printf(" %11.3e %11.3e %4ld\n", x[i], f[i], ivalid[i]);

 END:
  NAG_FREE(f);
  NAG_FREE(x);
  NAG_FREE(ivalid);

  return exit_status;
}
```

10.2 Program Data

```
nag_kelvin_ker_vector (s19aqc) Example Program Data

6

0.1 1.0 2.5 5.0 10.0 15.0
```

10.3 Program Results

```
nag_kelvin_ker_vector (s19aqc) Example Program Results

     x           f          ivalid

   1.000e-01   2.420e+00    0
   1.000e+00   2.867e-01    0
   2.500e+00  -6.969e-02    0
   5.000e+00  -1.151e-02    0
   1.000e+01   1.295e-04    0
```

```
   1.500e+01  -1.514e-08    0
```