NAG Library Routine Document

S19APF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of **bold italicised** terms and other implementation-dependent details.

1 Purpose

S19APF returns an array of values for the Kelvin function $\text{bei}\, x$.

2 Specification

```
SUBROUTINE S19APF (N, X, F, IVALID, IFAIL)

INTEGER          N, IVALID(N), IFAIL
REAL (KIND=nag_wp) X(N), F(N)
```

3 Description

S19APF evaluates an approximation to the Kelvin function $\text{bei}\, x_i$ for an array of arguments x_i, for $i = 1, 2, \ldots, n$.

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

The routine is based on several Chebyshev expansions:

For $0 \leq x \leq 5$,

$$\text{bei}\, x = \frac{x^2}{4} \sum_{r=0} a_r T_r(t), \qquad \text{with } t = 2\left(\frac{x}{5}\right)^4 - 1;$$

For $x > 5$,

$$\text{bei}\, x = \frac{e^{x/\sqrt{2}}}{\sqrt{2\pi x}}\left[\left(1 + \frac{1}{x}a(t)\right)\sin\alpha - \frac{1}{x}b(t)\cos\alpha\right]$$

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

where $\alpha = \dfrac{x}{\sqrt{2}} - \dfrac{\pi}{8}$, $\beta = \dfrac{x}{\sqrt{2}} + \dfrac{\pi}{8}$,

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

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

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 routine must fail.

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: $N \geq 0$.

2:     X(N) – REAL (KIND=nag_wp) array   *Input*

On entry: the argument x_i of the function, for $i = 1, 2, \ldots, N$.

3:     F(N) – REAL (KIND=nag_wp) array   *Output*

On exit: bei x_i, the function values.

4:     IVALID(N) – INTEGER array   *Output*

On exit: IVALID(i) contains the error code for x_i, for $i = 1, 2, \ldots, N$.

IVALID(i) = 0
       No error.

IVALID(i) = 1
       $\text{abs}(x_i)$ is too large for an accurate result to be returned. F(i) contains zero. The threshold
       value is the same as for IFAIL = 1 in S19ABF, as defined in the Users' Note for your
       implementation.

5:     IFAIL – INTEGER   *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should
refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is
detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then
the value 1 is recommended. Otherwise, if you are not familiar with this argument, the
recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of
IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see
Section 6).

6   Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit
(as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

       On entry, at least one value of X was invalid.
       Check IVALID for more information.

IFAIL = 2

       On entry, N = $\langle value \rangle$.
       Constraint: $N \geq 0$.

IFAIL = -99

       An unexpected error has been triggered by this routine. Please contact NAG.

       See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = −399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = −999

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

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 δ be the relative error in the argument. If δ is somewhat larger than the **machine precision**, then we have:

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

(provided E is within machine bounds).

For small x the error amplification is insignificant and thus the absolute error is effectively bounded by the **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

S19APF is not threaded in any implementation.

9 Further Comments

None.

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

```
    Program s19apfe

!     S19APF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!       .. Use Statements ..
      Use nag_library, Only: nag_wp, s19apf
!       .. Implicit None Statement ..
      Implicit None
!       .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!       .. Local Scalars ..
      Integer                           :: i, ifail, n
!       .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: f(:), x(:)
      Integer, Allocatable              :: ivalid(:)
```

```
!      .. Executable Statements ..
       Write (nout,*) 'S19APF Example Program Results'

!      Skip heading in data file
       Read (nin,*)

       Write (nout,*)
       Write (nout,*) '      X            F           IVALID'
       Write (nout,*)

       Read (nin,*) n

       Allocate (x(n),f(n),ivalid(n))

       Read (nin,*) x(1:n)

       ifail = 0
       Call s19apf(n,x,f,ivalid,ifail)

       Do i = 1, n
         Write (nout,99999) x(i), f(i), ivalid(i)
       End Do

99999 Format (1X,1P,2E12.3,I5)
     End Program s19apfe
```

10.2 Program Data

```
S19APF Example Program Data

7

0.1 1.0 2.5 5.0 10.0 15.0 -1.0
```

10.3 Program Results

```
 S19APF Example Program Results

     X            F           IVALID

   1.000E-01    2.500E-03     0
   1.000E+00    2.496E-01     0
   2.500E+00    1.457E+00     0
   5.000E+00    1.160E-01     0
   1.000E+01    5.637E+01     0
   1.500E+01   -2.953E+03     0
  -1.000E+00    2.496E-01     0
```