S14AAF – NAG Fortran Library Routine Document

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

S14AAF returns the value of the Gamma function $\Gamma(x)$, via the routine name.

2 Specification

```
real FUNCTION S14AAF(X, IFAIL)
INTEGER              IFAIL
real                 X
```

3 Description

This routine evaluates an approximation to the Gamma function $\Gamma(x)$. The routine is based on the Chebyshev expansion:

$$\Gamma(1+u) = {\sum_{r=0}}' a_r T_r(t) \ , \text{ where } 0 \leq u < 1 \ , \ t = 2u - 1,$$

and uses the property $\Gamma(1+x) = x\Gamma(x)$. If $x = N + 1 + u$ where N is integral and $0 \leq u < 1$ then it follows that:

for $N > 0$ $\Gamma(x) = (x-1)(x-2)\ldots(x-N)\Gamma(1+u)$,
for $N = 0$ $\Gamma(x) = \Gamma(1+u)$,
for $N < 0$ $\Gamma(x) = \frac{\Gamma(1+u)}{x(x+1)(x+2)\ldots(x-N-1)}$.

There are four possible failures for this routine:

(i) if x is too large, there is a danger of overflow since $\Gamma(x)$ could become too large to be represented in the machine;
(ii) if x is too large and negative, there is a danger of underflow;
(iii) if x is equal to a negative integer, $\Gamma(x)$ would overflow since it has poles at such points;
(iv) if x is too near zero, there is again the danger of overflow on some machines. For small x, $\Gamma(x) \simeq \frac{1}{x}$, and on some machines there exists a range of non-zero but small values of x for which $1/x$ is larger than the greatest representable value.

4 References

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

5 Parameters

1: X — ***real*** *Input*

On entry: the argument x of the function.

Constraint: X must not be zero or a negative integer.

2: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

The argument is too large. On soft failure the routine returns the approximate value of $\Gamma(x)$ at the nearest valid argument.

IFAIL = 2

The argument is too large and negative. On soft failure the routine returns zero.

IFAIL = 3

The argument is too close to zero. On soft failure the routine returns the approximate value of $\Gamma(x)$ at the nearest valid argument.

IFAIL = 4

The argument is a negative integer, at which value $\Gamma(x)$ is infinite. On soft failure the routine returns a large positive value.

7 Accuracy

Let δ and ϵ be the relative errors in the argument and the result respectively. If δ is somewhat larger than the ***machine precision*** (i.e., is due to data errors etc.), then ϵ and δ are approximately related by:

$$\epsilon \simeq |x\Psi(x)|\delta$$

(provided ϵ is also greater than the representation error). Here $\Psi(x)$ is the digamma function $\frac{\Gamma'(x)}{\Gamma(x)}$. Figure 1 shows the behaviour of the error amplification factor $|x\Psi(x)|$.

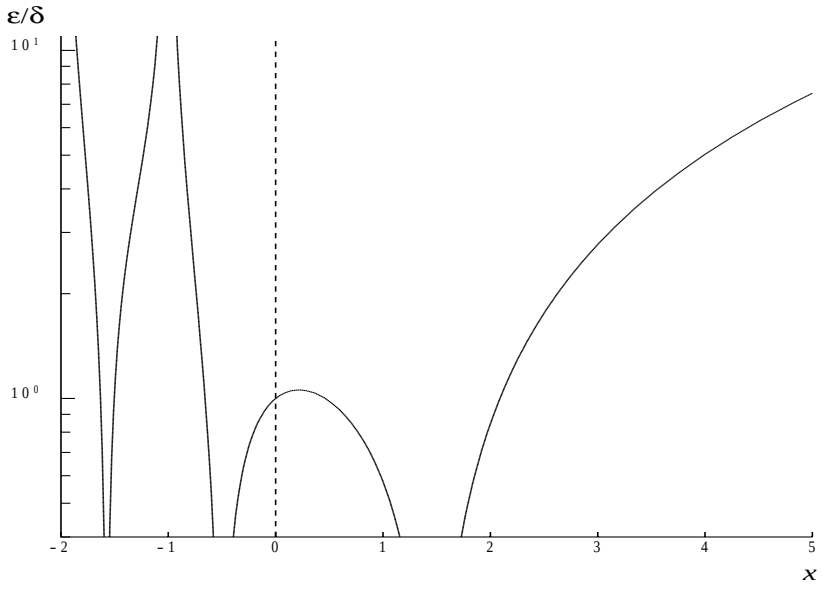

Figure 1

If δ is of the same order as ***machine precision***, then rounding errors could make ϵ slightly larger than the above relation predicts.

There is clearly a severe, but unavoidable, loss of accuracy for arguments close to the poles of $\Gamma(x)$ at negative integers. However relative accuracy is preserved near the pole at $x = 0$ right up to the point of failure arising from the danger of overflow.

Also accuracy will necessarily be lost as x becomes large since in this region

$$\epsilon \simeq \delta x \ln x.$$

However since $\Gamma(x)$ increases rapidly with x, the routine must fail due to the danger of overflow before this loss of accuracy is too great. (e.g., for $x = 20$, the amplification factor $\simeq 60$.)

8 Further Comments

None.

9 Example

The example program reads values of the argument x from a file, evaluates the function at each value of x and prints the results.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     S14AAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             X, Y
      INTEGER          IFAIL
*     .. External Functions ..
      real             S14AAF
      EXTERNAL         S14AAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'S14AAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      WRITE (NOUT,*)
      WRITE (NOUT,*) '     X           Y        IFAIL'
      WRITE (NOUT,*)
   20 READ (NIN,*,END=40) X
      IFAIL = 1
*
      Y = S14AAF(X,IFAIL)
*
      WRITE (NOUT,99999) X, Y, IFAIL
      GO TO 20
   40 STOP
*
99999 FORMAT (1X,1P,2e12.3,I7)
      END
```

9.2 Program Data

```
S14AAF Example Program Data
             1.0
             1.25
             1.5
             1.75
             2.0
             5.0
             10.0
             -1.5
```

9.3 Program Results

```
S14AAF Example Program Results

     X           Y        IFAIL

  1.000E+00   1.000E+00      0
  1.250E+00   9.064E-01      0
  1.500E+00   8.862E-01      0
  1.750E+00   9.191E-01      0
  2.000E+00   1.000E+00      0
  5.000E+00   2.400E+01      0
  1.000E+01   3.629E+05      0
 -1.500E+00   2.363E+00      0
```