NAG Toolbox for MATLAB

s14ad

1 Purpose

s14ad returns a sequence of values of scaled derivatives of the psi function $\psi(x)$.

2 Syntax

```
[ans, ifail] = s14ad(x, n, m)
```

3 Description

s14ad computes m values of the function

$$w(k,x) = \frac{(-1)^{k+1}\psi^{(k)}(x)}{k!},$$

for $x > 0$, $k = n,\ n+1,\ldots,n+m-1$, where ψ is the psi function

$$\psi(x) = \frac{d}{dx}\ln\Gamma(x) = \frac{\Gamma'(x)}{\Gamma(x)},$$

and $\psi^{(k)}$ denotes the kth derivative of ψ.

The function is derived from the function PSIFN in Amos 1983. The basic method of evaluation of $w(k,x)$ is the asymptotic series

$$w(k,x) \sim \epsilon(k,x) + \frac{1}{2x^{k+1}} + \frac{1}{x^k}\sum_{j=1}^{\infty} B_{2j}\frac{(2j+k-1)!}{(2j)!k!x^{2j}}$$

for large x greater than a machine-dependent value $x_{\min}$, followed by backward recurrence using

$$w(k,x) = w(k,x+1) + x^{-k-1}$$

for smaller values of x, where $\epsilon(k,x) = -\ln x$ when $k = 0$, $\epsilon(k,x) = \frac{1}{kx^k}$ when $k > 0$, and B_{2j}, $j = 1,2,\ldots,$ are the Bernoulli numbers.

When k is large, the above procedure may be inefficient, and the expansion

$$w(k,x) = \sum_{j=1}^{\infty}\frac{1}{(x+j)^{k+1}},$$

which converges rapidly for large k, is used instead.

4 References

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

Amos D E 1983 Algorithm 610: A portable FORTRAN subroutine for derivatives of the psi function *ACM Trans. Math. Software* **9** 494–502

5 Parameters

5.1 Compulsory Input Parameters

1: **x – double scalar**

The argument x of the function.

Constraint: $\mathbf{x} > 0.0$.

2: **n – int32 scalar**

The index of the first member n of the sequence of functions.

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

3: **m – int32 scalar**

the number of members m required in the sequence $w(k,x)$, for $k = n, n+1, \ldots, n+m-1$.

Constraint: $\mathbf{m} \geq 1$.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **ans(m) – double array**

The first m elements of **ans** contain the required values $w(k,x)$, for $k = n, n+1, \ldots, n+m-1$.

2: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, $\mathbf{x} \leq 0.0$.

ifail $= 2$

On entry, $\mathbf{n} < 0$.

ifail $= 3$

On entry, $\mathbf{m} < 1$.

ifail $= 4$

No results are returned because underflow is likely. Either **x** or $\mathbf{n} + \mathbf{m} - 1$ is too large. If possible, reduce the value of **m** and call s14ad again.

ifail $= 5$

No results are returned because overflow is likely. Either **x** is too small, or $\mathbf{n} + \mathbf{m} - 1$ is too large. If possible, reduce the value of **m** and call s14ad again.

ifail $= 6$

No results are returned because there is not enough internal workspace to continue computation. $\mathbf{n} + \mathbf{m} - 1$ may be too large. If possible, reduce the value of **m** and call s14ad again.

7 Accuracy

All constants in s14ad are given to approximately 18 digits of precision. Calling the number of digits of precision in the floating-point arithmetic being used t, then clearly the maximum number of correct digits in the results obtained is limited by $p = \min(t, 18)$. Empirical tests of s14ad, taking values of x in the range $0.0 < x < 50.0$, and n in the range $1 \leq n \leq 50$, have shown that the maximum relative error is a loss of approximately two decimal places of precision. Tests with $n = 0$, i.e., testing the function $-\psi(x)$, have shown somewhat better accuracy, except at points close to the zero of $\psi(x)$, $x \simeq 1.461632$, where only absolute accuracy can be obtained.

8 Further Comments

The time taken for a call of s14ad is approximately proportional to m, plus a constant. In general, it is much cheaper to call s14ad with m greater than 1 to evaluate the function $w(k, x)$, for $k = n, n + 1, \ldots, n + m - 1$, rather than to make m separate calls of s14ad.

9 Example

```
x = 0.1;
n = int32(0);
m = int32(4);
[ans, ifail] = s14ad(x, n, m)
```

```
ans =
   1.0e+04 *
    0.0010
    0.0101
    0.1001
    1.0001
ifail =
           0
```