NAG Library Function Document

nag_real_polygamma (s14aec)

1 Purpose

nag_real_polygamma (s14aec) returns the value of the kth derivative of the psi function $\psi(x)$ for real x and $k = 0, 1, \ldots, 6$.

2 Specification

```
#include <nag.h>
#include <nags.h>
```

```
double nag_real_polygamma (double x, Integer k, NagError *fail)
```

3 Description

nag_real_polygamma (s14aec) evaluates an approximation to the kth derivative of the psi function $\psi(x)$ given by

$$\psi^{(k)}(x) = \frac{d^k}{dx^k}\psi(x) = \frac{d^k}{dx^k}\left(\frac{d}{dx}\log_e \Gamma(x)\right),$$

where x is real with $x \neq 0, -1, -2, \ldots$ and $k = 0, 1, \ldots, 6$. For negative noninteger values of x, the recurrence relationship

$$\psi^{(k)}(x+1) = \psi^{(k)}(x) + \frac{d^k}{dx^k}\left(\frac{1}{x}\right)$$

is used. The value of $\frac{(-1)^{k+1}\psi^{(k)}(x)}{k!}$ is obtained by a call to nag_polygamma_deriv (s14adc), which is based on the function PSIFN in Amos (1983).

Note that $\psi^{(k)}(x)$ is also known as the *polygamma* function. Specifically, $\psi^{(0)}(x)$ is often referred to as the *digamma* function and $\psi^{(1)}(x)$ as the *trigamma* function in the literature. Further details can be found in Abramowitz and Stegun (1972).

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 Arguments

1: **x** – double *Input*

On entry: the argument x of the function.

Constraint: **x** must not be 'too close' (see Section 6) to a non-positive integer.

2: **k** – Integer *Input*

On entry: the function $\psi^{(k)}(x)$ to be evaluated.

Constraint: $0 \leq \mathbf{k} \leq 6$.

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

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

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_INT

On entry, $\mathbf{k} = \langle value \rangle$.
Constraint: $\mathbf{k} \leq 6$.

On entry, $\mathbf{k} = \langle value \rangle$.
Constraint: $\mathbf{k} \geq 0$.

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.

NE_OVERFLOW_LIKELY

Evaluation abandoned due to likelihood of overflow.

NE_REAL

On entry, **x** is 'too close' to a non-positive integer: $\mathbf{x} = \langle value \rangle$ and $\text{nint}(\mathbf{x}) = \langle value \rangle$.

NE_UNDERFLOW_LIKELY

Evaluation abandoned due to likelihood of underflow.

7 Accuracy

All constants in nag_polygamma_deriv (s14adc) are given to approximately 18 digits of precision. If t denotes the number of digits of precision in the floating-point arithmetic being used, then clearly the maximum number in the results obtained is limited by $p = \min(t, 18)$. Empirical tests by Amos (1983) have shown that the maximum relative error is a loss of approximately two decimal places of precision. Further tests with the function $-\psi^{(0)}(x)$ have shown somewhat improved accuracy, except at points near the positive zero of $\psi^{(0)}(x)$ at $x = 1.46\ldots$, where only absolute accuracy can be obtained.

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

This example evaluates $\psi^{(2)}(x)$ at $x = 2.5$, and prints the results.

10.1 Program Text

```
/* nag_real_polygamma (s14aec) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * NAG C Library
 *
 * Mark 6, 2000.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nags.h>

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

  INIT_FAIL(fail);

  /* Skip heading in data file */
  scanf("%*[^\n]");
  printf("nag_real_polygamma (s14aec) Example Program Results\n\n");
  printf("   x      k    (d^k/dx^k)psi(x)\n");
  while (scanf("%lf %ld%*[^\n]", &x, &k) != EOF)
    {
      /* nag_real_polygamma (s14aec).
       * Derivative of the psi function psi(x)
       */
      y = nag_real_polygamma(x, k, &fail);
      if (fail.code == NE_NOERROR)
        printf("%5.1f %5ld     %13.4e\n", x, k, y);
      else
        {
          printf("Error from nag_real_polygamma (s14aec).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }
    }
 END:
  return exit_status;
}
```

10.2 Program Data

```
nag_real_polygamma (s14aec) Example Program Data
 1.0   0
 0.5   1
-3.6   2
 8.0   3
 2.9   4
-4.7   5
-5.4   6  : Values of x and k
```

10.3 Program Results

```
nag_real_polygamma (s14aec) Example Program Results

   x      k    (d^k/dx^k)psi(x)
  1.0     0      -5.7722e-01
  0.5     1       4.9348e+00
 -3.6     2      -2.2335e+01
  8.0     3       4.6992e-03
  2.9     4      -1.5897e-01
 -4.7     5       1.6566e+05
 -5.4     6       4.1378e+05
```