NAG Library Function Document

nag_complex_polygamma (s14afc)

1 Purpose

nag_complex_polygamma (s14afc) returns the value of the kth derivative of the psi function $\psi(z)$ for complex z and $k = 0, 1, \ldots, 4$.

2 Specification

```
#include <nag.h>
#include <nags.h>
Complex nag_complex_polygamma (Complex z, Integer k, NagError *fail)
```

3 Description

nag_complex_polygamma (s14afc) evaluates an approximation to the kth derivative of the psi function $\psi(z)$ given by

$$\psi^{(k)}(z) = \frac{d^k}{dz^k}\psi(z) = \frac{d^k}{dz^k}\left(\frac{d}{dz}\log_{\mathrm{e}}\Gamma(z)\right),$$

where $z = x + iy$ is complex provided $y \neq 0$ and $k = 0, 1, \ldots, 4$. If $y = 0$, z is real and thus $\psi^{(k)}(z)$ is singular when $z = 0, -1, -2, \ldots$.

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

nag_complex_polygamma (s14afc) is based on a modification of the method proposed by KÎlbig (1972).

To obtain the value of $\psi^{(k)}(z)$ when z is real, nag_real_polygamma (s14aec) can be used.

4 References

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

KÎlbig K S (1972) Programs for computing the logarithm of the gamma function, and the digamma function, for complex arguments *Comp. Phys. Comm.* **4** 221–226

5 Arguments

1: **z** – Complex *Input*

On entry: the argument z of the function.

Constraint: **z**.re must not be 'too close' (see Section 6) to a non-positive integer when **z**.$im = 0.0$.

2: **k** – Integer *Input*

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

Constraint: $0 \leq \mathbf{k} \leq 4$.

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_COMPLEX

On entry, $\mathbf{z}.re$ is 'too close' to a non-positive integer when $\mathbf{z}.im = 0.0$: $\mathbf{z}.re = \langle value \rangle$, $\text{nint}(\mathbf{z}.re) = \langle value \rangle$.

NE_INT

On entry, $\mathbf{k} = \langle value \rangle$.
Constraint: $\mathbf{k} \leq 4$.

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.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

NE_OVERFLOW_LIKELY

Evaluation abandoned due to likelihood of overflow.

7 Accuracy

Empirical tests have shown that the maximum relative error is a loss of approximately two decimal places of precision.

8 Parallelism and Performance

nag_complex_polygamma (s14afc) is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example evaluates the psi (trigamma) function $\psi^{(1)}(z)$ at $z = -1.5 + 2.5i$, and prints the results.

10.1 Program Text

```
/* nag_complex_polygamma (s14afc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * NAG C Library
 *
 * Mark 26, 2016.
 */

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

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

  INIT_FAIL(fail);

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
#endif
  printf("nag_complex_polygamma (s14afc) Example Program Results\n");
  printf("       z            k         (d^k/dz^k)psi(z)\n");
#ifdef _WIN32
  while (scanf_s(" (%lf,%lf)%" NAG_IFMT "%*[^\n] ", &z.re, &z.im, &k) != EOF)
#else
  while (scanf(" (%lf,%lf)%" NAG_IFMT "%*[^\n] ", &z.re, &z.im, &k) != EOF)
#endif
  {
    /* nag_complex_polygamma (s14afc).
     * Derivative of the psi function psi(z)
     */
    y = nag_complex_polygamma(z, k, &fail);
    if (fail.code == NE_NOERROR)
      printf("(%5.1f, %5.1f) %6" NAG_IFMT " (%13.4e, %13.4e)\n",
             z.re, z.im, k, y.re, y.im);
    else {
      printf("Error from nag_complex_polygamma (s14afc).\n%s\n",
             fail.message);
      exit_status = 1;
      goto END;
    }
  }
END:
  return exit_status;
}
```

10.2 Program Data

```
nag_complex_polygamma (s14afc) Example Program Data
(1.2,5.0)   0
(0.5,-0.2)  1
(-1.5,2.5)  1
(8.0,3.3)   3
(2.9,7.5)   4  : Values of z and k
```

10.3 Program Results

```
nag_complex_polygamma (s14afc) Example Program Results
       z            k         (d^k/dz^k)psi(z)
(  1.2,   5.0)      0 (   1.6176e+00,    1.4312e+00)
(  0.5,  -0.2)      1 (   3.4044e+00,    2.5394e+00)
( -1.5,   2.5)      1 (  -1.9737e-01,   -2.4271e-01)
(  8.0,   3.3)      3 (   1.1814e-03,   -3.4188e-03)
(  2.9,   7.5)      4 (  -5.0227e-04,   -1.4955e-03)
```