NAG Library Function Document

nag_polygamma_fun (s14acc)

1 Purpose

nag_polygamma_fun (s14acc) returns a value of the function $\psi(x) - \ln x$, where ψ is the psi function $\psi(x) = \dfrac{d}{dx}\ln\Gamma(x) = \dfrac{\Gamma'(x)}{\Gamma(x)}$.

2 Specification

```
#include <nag.h>
#include <nags.h>
double nag_polygamma_fun (double x, NagError *fail)
```

3 Description

nag_polygamma_fun (s14acc) returns a value of the function $\psi(x) - \ln x$. The psi function is computed without the logarithmic term so that when x is large, sums or differences of psi functions may be computed without unnecessary loss of precision, by analytically combining the logarithmic terms. For example, the difference $d = \psi\left(x+\frac{1}{2}\right) - \psi(x)$ has an asymptotic behaviour for large x given by $d \sim \ln\left(x+\frac{1}{2}\right) - \ln x + O\left(\dfrac{1}{x^2}\right) \sim \ln\left(1+\dfrac{1}{2x}\right) \sim \dfrac{1}{2x}$.

Computing d directly would amount to subtracting two large numbers which are close to $\ln\left(x+\frac{1}{2}\right)$ and $\ln x$ to produce a small number close to $\frac{1}{2x}$, resulting in a loss of significant digits. However, using this function to compute $f(x) = \psi(x) - \ln x$, we can compute $d = f\left(x+\frac{1}{2}\right) - f(x) + \ln\left(1+\frac{1}{2x}\right)$, and the dominant logarithmic term may be computed accurately from its power series when x is large. Thus we avoid the unnecessary loss of precision.

The function is derived from the function PSIFN in Amos (1983).

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** > 0.0.

2: **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_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

Computation halted due to likelihood of overflow. **x** may be too small. $\mathbf{x} = \langle value \rangle$.

NE_REAL

On entry, $\mathbf{x} = \langle value \rangle$.
Constraint: $\mathbf{x} > 0.0$.

NE_UNDERFLOW_LIKELY

Computation halted due to likelihood of underflow. **x** may be too large. $\mathbf{x} = \langle value \rangle$.

7 Accuracy

All constants in nag_polygamma_fun (s14acc) 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 $p = \min(t, 18)$.

With the above proviso, results returned by this function should be accurate almost to full precision, except at points close to the zero of $\psi(x)$, $x \simeq 1.461632$, where only absolute rather than relative accuracy can be obtained.

8 Parallelism and Performance

nag_polygamma_fun (s14acc) is not threaded in any implementation.

9 Further Comments

None.

10 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.

10.1 Program Text

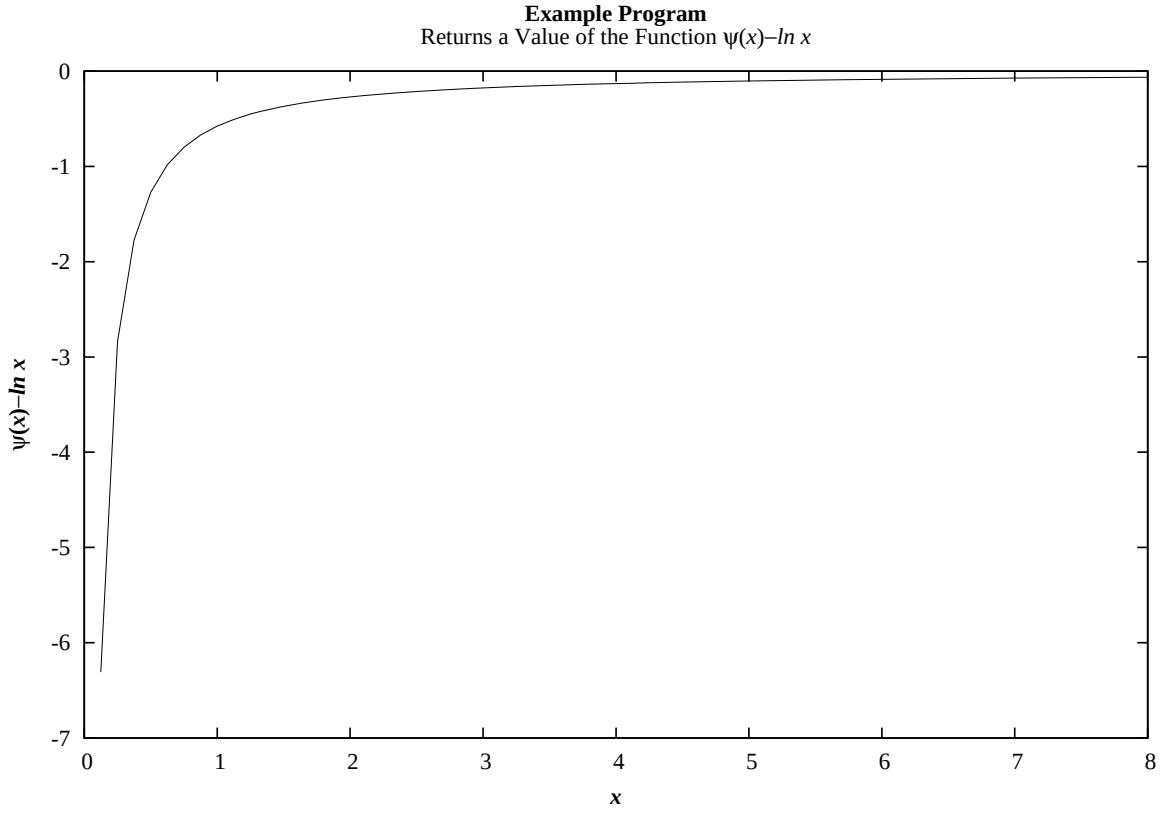
```
/* nag_polygamma_fun (s14acc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

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

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

  INIT_FAIL(fail);

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif
  printf("nag_polygamma_fun (s14acc) Example Program Results\n");
  printf("     x       psi(x)-log(x)\n");
#ifdef _WIN32
  while (scanf_s("%lf", &x) != EOF)
#else
  while (scanf("%lf", &x) != EOF)
#endif
  {
    /* nag_polygamma_fun (s14acc).
     * psi(x) - ln(x)
     */
    f = nag_polygamma_fun(x, &fail);
    if (fail.code != NE_NOERROR) {
      printf("Error from nag_polygamma_fun (s14acc).\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
    printf("%8.3f %14.4f\n", x, f);
  }

END:
  return exit_status;
}
```

10.2 Program Data

```
nag_polygamma_fun (s14acc) Example Program Data
0.1
0.5
3.6
8.0
```

10.3 Program Results

```
nag_polygamma_fun (s14acc) Example Program Results
     x       psi(x)-log(x)
   0.100        -8.1212
   0.500        -1.2704
   3.600        -0.1453
   8.000        -0.0638
```


Example Program
Returns a Value of the Function ψ(x)–ln x
0
-1
-2
-3
-4
-5
-6
-7
0
1
2
3
4
5
6
7
8
ψ(x)–ln x
x