nag_incomplete_gamma (s14bac)

1. Purpose

nag_incomplete_gamma (s14bac) computes values for the incomplete gamma functions $P(a,x)$ and $Q(a,x)$.

2. Specification

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

void nag_incomplete_gamma(double a, double x, double tol, double *p,
     double *q, NagError *fail)
```

3. Description

This function evaluates the incomplete gamma functions in the normalised form

$$\begin{aligned} P(a,x) &= \frac{1}{\Gamma(a)}\int_0^x t^{a-1}e^{-t}\,dt \\ Q(a,x) &= \frac{1}{\Gamma(a)}\int_x^\infty t^{a-1}e^{-t}\,dt, \end{aligned}$$

with $x \geq 0$ and $a > 0$, to a user-specified accuracy. With this normalisation, $P(a,x)+Q(a,x)=1$.

Several methods are used to evaluate the functions depending on the arguments a and x, the methods including Taylor expansion for $P(a,x)$, Legendre's continued fraction for $Q(a,x)$, and power series for $Q(a,x)$. When both a and x are large, and $a \simeq x$, the uniform asymptotic expansion of Temme (1987) is employed for greater efficiency – specifically, this expansion is used when $a \geq 20$ and $0.7a \leq x \leq 1.4a$.

Once either of P or Q is computed, the other is obtained by subtraction from 1. In order to avoid loss of relative precision in this subtraction, the smaller of P and Q is computed first.

This function is derived from subroutine GAM in Gautschi (1979b).

4. Parameters

a

Input: the argument a of the functions.
Constraint: $\mathbf{a} > 0.0$.

x

Input: the argument x of the functions.
Constraint: $\mathbf{x} \geq 0.0$.

tol

Input: the relative accuracy required by the user in the results. If nag_incomplete_gamma is entered with **tol** greater than 1.0 or less than ***machine precision***, then the value of ***machine precision*** is used instead.

p
q

Output: the values of the functions $P(a,x)$ and $Q(a,x)$ respectively.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

On error nag_incomplete_gamma returns with a value of 0.0 for **p** and **q**.

NE_REAL_ARG_LE

On entry, **a** must not be less than or equal to 0.0: **a** = ⟨*value*⟩.

NE_REAL_ARG_LT

On entry, **x** must not be less than 0.0: **x** = ⟨*value*⟩.

NE_ALG_NOT_CONV

The algorithm has failed to converge in ⟨*value*⟩ iterations.
Convergence of the Taylor series or Legendre continued fraction has failed within the specified number of iterations. This error is extremely unlikely to occur; if it does, contact NAG.

6. Further Comments

The time taken for a call of nag_incomplete_gamma depends on the precision requested through **tol**, and also varies slightly with the input arguments **a** and **x**.

6.1. Accuracy

There are rare occasions when the relative accuracy attained is somewhat less than that specified by parameter **tol**. However, the error should never exceed more than one or two decimal places. Note also that there is a limit of 18 decimal places on the achievable accuracy, because constants in the function are given to this precision.

6.2. References

Gautschi W (1979a) A Computational Procedure for Incomplete Gamma Functions *ACM Trans. Math. Software* **5** 466–481.

Gautschi W (1979b) Algorithm 542: Incomplete Gamma Functions *ACM Trans. Math. Software* **5** 482–489.

Temme N M (1987) On the computation of the incomplete gamma functions for large values of the parameters *Algorithms for Approximation* J C Mason and M G Cox (ed) Oxford University Press.

7. See Also

None.

8. Example

The following program reads values of the argument a and x from a file, evaluates the function and prints the results.

8.1. Program Text

```
/* nag_incomplete_gamma(s14bac) Example Program
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 2 revised, 1992.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nags.h>
#include <nagx02.h>

main()
{
  double a, p, q, tol, x;

  /*  Skip heading in data file */
  Vscanf("%*[^\n]");
```

```
  Vprintf("s14bac Example Program Results\n");
  tol = X02AJC;
  Vprintf("        a           x           p           q\n");
  while (scanf("%lf %lf", &a, &x) != EOF)
    {
      s14bac(a, x, tol, &p, &q, NAGERR_DEFAULT);
      Vprintf("%12.4f%12.4f%12.4f%12.4f\n", a, x, p, q);
    }
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

```
s14bac Example Program Data
  2.0   3.0
  7.0   1.0
  0.5  99.0
 20.0  21.0
 21.0  20.0
```

8.3. Program Results

```
s14bac Example Program Results
        a           x           p           q
      2.0000      3.0000      0.8009      0.1991
      7.0000      1.0000      0.0001      0.9999
      0.5000     99.0000      1.0000      0.0000
     20.0000     21.0000      0.6157      0.3843
     21.0000     20.0000      0.4409      0.5591
```