NAG Library Function Document

nag_gamma_pdf (g01kfc)

1 Purpose

nag_gamma_pdf (g01kfc) returns the value of the probability density function (PDF) for the gamma distribution with shape argument α and scale argument β at a point x.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
double nag_gamma_pdf (double x, double a, double b, NagError *fail)
```

3 Description

The gamma distribution has PDF

$$f(x) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)}x^{\alpha-1}e^{-x/\beta} \quad \text{if } x \geq 0; \quad \alpha, \beta > 0$$

$$f(x) = 0 \qquad \text{otherwise.}$$

If $0.01 \leq x, \alpha, \beta \leq 100$ then an algorithm based directly on the gamma distribution's PDF is used. For values outside this range, the function is calculated via the Poisson distribution's PDF as described in Loader (2000) (see Section 9).

4 References

Loader C (2000) Fast and accurate computation of binomial probabilities **(not yet published)**

5 Arguments

1: **x** – double *Input*

On entry: x, the value at which the PDF is to be evaluated.

2: **a** – double *Input*

On entry: α, the shape argument of the gamma distribution.

Constraint: **a** > 0.0.

3: **b** – double *Input*

On entry: β, the scale argument of the gamma distribution.

Constraints:

b > 0.0;
$\frac{\mathbf{x}}{\mathbf{b}} < \frac{1}{\text{nag\_real\_safe\_small\_number}()}$.

4: **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 3.2.1.2 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 3.6.6 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 3.6.5 in How to Use the NAG Library and its Documentation for further information.

NE_OVERFLOW

Computation abandoned owing to overflow due to extreme parameter values.

NE_REAL

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

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

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_gamma_pdf (g01kfc) is not threaded in any implementation.

9 Further Comments

Due to the lack of a stable link to Loader (2000) paper, we give a brief overview of the method, as applied to the Poisson distribution. The Poisson distribution has a continuous mass function given by,

$$p(x;\lambda) = \frac{\lambda^x}{x!}e^{-\lambda}. \tag{1}$$

The usual way of computing this quantity would be to take the logarithm and calculate,

$$\log(x;\lambda) = x\log\lambda - \log(x!) - \lambda.$$

For large x and λ, $x\log\lambda$ and $\log(x!)$ are very large, of the same order of magnitude and when calculated have rounding errors. The subtraction of these two terms can therefore result in a number, many orders of magnitude smaller and hence we lose accuracy due to subtraction errors. For example for $x = 2 \times 10^6$ and $\lambda = 2 \times 10^6$, $\log(x!) \approx 2.7 \times 10^7$ and $\log(p(x;\lambda)) = -8.17326744645834$. But calculated with the method shown later we have $\log(p(x;\lambda)) = -8.1732674441334492$. The difference between these two results suggests a loss of about 7 significant figures of precision.

Loader introduces an alternative way of expressing (1) based on the saddle point expansion,

$$\log(p(x;\lambda)) = \log(p(x;x)) - D(x;\lambda), \tag{2}$$

where $D(x;\lambda)$, the deviance for the Poisson distribution is given by,

$$\begin{aligned} D(x;\lambda) &= \log\left(p(x;x)\right) - \log\left(p(x;\lambda)\right), \\ &= \lambda D_0\left(\tfrac{x}{\lambda}\right), \end{aligned} \tag{3}$$

and

$$D_0(\epsilon) = \epsilon\log\epsilon + 1 - \epsilon.$$

For ϵ close to 1, $D_0(\epsilon)$ can be evaluated through the series expansion

$$\lambda D_0\left(\frac{x}{\lambda}\right) = \frac{(x-\lambda)^2}{x+\lambda} + 2x\sum_{j=1}^{\infty}\frac{v^{2j+1}}{2j+1}, \qquad \text{where } v = \frac{x-\lambda}{x+\lambda},$$

otherwise $D_0(\epsilon)$ can be evaluated directly. In addition, Loader suggests evaluating $\log(x!)$ using the Stirling–De Moivre series,

$$\log(x!) = \frac{1}{2}\log(2\pi x) + x\log(x) - x + \delta(x), \tag{4}$$

where the error $\delta(x)$ is given by

$$\delta(x) = \frac{1}{12x} - \frac{1}{360x^3} + \frac{1}{1260x^5} + \mathcal{O}\left(x^{-7}\right).$$

Finally $\log\left(p(x;\lambda)\right)$ can be evaluated by combining equations (1)–(4) to get,

$$p(x;\lambda) = \frac{1}{\sqrt{2\pi x}}e^{-\delta(x)-\lambda D_0(x/\lambda)}.$$

10 Example

This example prints the value of the gamma distribution PDF at six different points **x** with differing **a** and **b**.

10.1 Program Text

```
/* nag_gamma_pdf (g01kfc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

/* Pre-processor includes */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg01.h>

int main(void)
{
  /*Integer scalar and array declarations */
  Integer exit_status = 0;
  Integer i, ndata;
  /*Double scalar and array declarations */
  double a, b, f, x;
  /* Nag Types */
  NagError fail;

  INIT_FAIL(fail);

  printf("nag_gamma_pdf (g01kfc) Example Program Results\n\n");
#ifdef _WIN32
  scanf_s("%*[^\n] ");
#else
  scanf("%*[^\n] ");
```

```
#endif
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%*[^\n] ", &ndata);
#else
  scanf("%" NAG_IFMT "%*[^\n] ", &ndata);
#endif
  printf("%14s%16s%16s%16s\n\n", "X", "A", "B", "RESULT");
  for (i = 0; i < ndata; i++) {
#ifdef _WIN32
    scanf_s("%lf%lf%lf%*[^\n] ", &x, &a, &b);
#else
    scanf("%lf%lf%lf%*[^\n] ", &x, &a, &b);
#endif
    /*
     * nag_gamma_pdf (g01kfc)
     * Calculates the value for the probability density function of
     * the gamma distribution at a chosen point.
     */
    f = nag_gamma_pdf(x, a, b, &fail);
    if (fail.code != NE_NOERROR) {
      printf("Error from nag_gamma_pdf (g01kfc).\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
    printf("%18.4e%16.4e%16.4e%16.4e\n", x, a, b, f);
  }

END:

  return exit_status;
}
```

10.2 Program Data

```
nag_gamma_pdf (g01kfc) Example Program Data
6                   : ndata
1.0E-1 3.0E0 2.0E0
3.0E0 1.0E1 1.1E1
6.0E0 5.0E0 1.0E0
4.0E0 1.0E1 1.0E-1
9.0E0 9.0E0 5.0E-1
1.6E1 3.5E0 2.5E0   : x, a, b
```

10.3 Program Results

```
nag_gamma_pdf (g01kfc) Example Program Results

             X               A               B          RESULT

        1.0000e-01      3.0000e+00      2.0000e+00      5.9452e-04
        3.0000e+00      1.0000e+01      1.1000e+01      1.5921e-12
        6.0000e+00      5.0000e+00      1.0000e+00      1.3385e-01
        4.0000e+00      1.0000e+01      1.0000e-01      3.0690e-08
        9.0000e+00      9.0000e+00      5.0000e-01      8.3251e-03
        1.6000e+01      3.5000e+00      2.5000e+00      2.0723e-02
```

Example Program
Plots of the Gamma Distribution