NAG Library Routine Document

G01KKF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

G01KKF returns a number of values of the probability density function (PDF), or its logarithm, for the gamma distribution.

2 Specification

```
SUBROUTINE G01KKF (ILOG, LX, X, LA, A, LB, B, PDF, IVALID, IFAIL)
INTEGER            ILOG, LX, LA, LB, IVALID(*), IFAIL
REAL (KIND=nag_wp) X(LX), A(LA), B(LB), PDF(*)
```

3 Description

The gamma distribution with shape parameter α_i and scale parameter β_i has PDF

$$f(x_i,\alpha_i,\beta_i) = \frac{1}{{\beta_i}^{\alpha_i}\Gamma(\alpha_i)}{x_i}^{\alpha_i-1}e^{-x_i/\beta_i} \quad \text{if } x_i \geq 0; \quad \alpha_i,\beta_i > 0$$

$$f(x_i,\alpha_i,\beta_i) = 0 \qquad \text{otherwise.}$$

If $0.01 \leq x_i,\alpha_i,\beta_i \leq 100$ 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).

The input arrays to this routine are designed to allow maximum flexibility in the supply of vector parameters by re-using elements of any arrays that are shorter than the total number of evaluations required. See Section 2.6 in the G01 Chapter Introduction for further information.

4 References

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

5 Parameters

1: ILOG – INTEGER *Input*

On entry: the value of ILOG determines whether the logarithmic value is returned in PDF.

ILOG $= 0$
$f(x_i,\alpha_i,\beta_i)$, the probability density function is returned.

ILOG $= 1$
$\log(f(x_i,\alpha_i,\beta_i))$, the logarithm of the probability density function is returned.

Constraint: ILOG $= 0$ or 1.

2: LX – INTEGER *Input*

On entry: the length of the array X.

Constraint: LX > 0.

3: X(LX) – REAL (KIND=nag_wp) array *Input*

On entry: x_i, the values at which the PDF is to be evaluated with $x_i = \mathrm{X}(j)$, $j = ((i-1) \bmod \mathrm{LX}) + 1$, for $i = 1, 2, \ldots, \max(\mathrm{LX}, \mathrm{LA}, \mathrm{LB})$.

4: LA – INTEGER *Input*

On entry: the length of the array A.

Constraint: $\mathrm{LA} > 0$.

5: A(LA) – REAL (KIND=nag_wp) array *Input*

On entry: α_i, the shape parameter with $\alpha_i = \mathrm{A}(j)$, $j = ((i-1) \bmod \mathrm{LA}) + 1$.

Constraint: $\mathrm{A}(j) > 0.0$, for $j = 1, 2, \ldots, \mathrm{LA}$.

6: LB – INTEGER *Input*

On entry: the length of the array B.

Constraint: $\mathrm{LB} > 0$.

7: B(LB) – REAL (KIND=nag_wp) array *Input*

On entry: β_i, the scale parameter with $\beta_i = \mathrm{B}(j)$, $j = ((i-1) \bmod \mathrm{LB}) + 1$.

Constraint: $\mathrm{B}(j) > 0.0$, for $j = 1, 2, \ldots, \mathrm{LB}$.

8: PDF($*$) – REAL (KIND=nag_wp) array *Output*

Note: the dimension of the array PDF must be at least $\max(\mathrm{LX}, \mathrm{LA}, \mathrm{LB})$.

On exit: $f(x_i, \alpha_i, \beta_i)$ or $\log(f(x_i, \alpha_i, \beta_i))$.

9: IVALID($*$) – INTEGER array *Output*

Note: the dimension of the array IVALID must be at least $\max(\mathrm{LX}, \mathrm{LA}, \mathrm{LB})$.

On exit: IVALID(i) indicates any errors with the input arguments, with

IVALID(i) $= 0$
No error.

IVALID(i) $= 1$
$\alpha_i \leq 0.0$.

IVALID(i) $= 2$
$\beta_i \leq 0.0$.

IVALID(i) $= 3$
$\frac{x_i}{\beta_i}$ overflows, the value returned should be a reasonable approximation.

10: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, at least one value of X, A or B was invalid.
Check IVALID for more information.

IFAIL $= 2$

On entry, ILOG $= \langle value \rangle$.
Constraint: ILOG $= 0$ or 1.

IFAIL $= 3$

On entry, array size $= \langle value \rangle$.
Constraint: LX > 0.

IFAIL $= 4$

On entry, array size $= \langle value \rangle$.
Constraint: LA > 0.

IFAIL $= 5$

On entry, array size $= \langle value \rangle$.
Constraint: LB > 0.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

Not applicable.

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,

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

For large x and λ, $x\log\lambda$ and $\log\left(x!\right)$ 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 $x = 2\times10^6$ and $\lambda = 2\times10^6$, $\log\left(x!\right) \approx 2.7\times10^7$ and $\log\left(p(x;\lambda)\right) = -8.17326744645834$. But calculated with the method shown later we have $\log\left(p(x;\lambda)\right) = -8.1732674441334492$. 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,

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

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 $\log\left(x!\right)$ using the Stirling–De Moivre series,

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

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_i with differing α_i and β_i.

10.1 Program Text

```
    Program g01kkfe
!     G01KKF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: g01kkf, nag_wp
```

```
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, ilog, la, lb, lout, lx
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:), b(:), pdf(:), x(:)
      Integer, Allocatable             :: ivalid(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, mod, repeat
!     .. Executable Statements ..
      Write (nout,*) 'G01KKF Example Program Results'
      Write (nout,*)

!     Skip heading in data file
      Read (nin,*)

!     Read in flag indicating whether logs are required
      Read (nin,*) ilog

!     Read in the input vectors
      Read (nin,*) lx
      Allocate (x(lx))
      Read (nin,*) x(1:lx)

      Read (nin,*) la
      Allocate (a(la))
      Read (nin,*) a(1:la)

      Read (nin,*) lb
      Allocate (b(lb))
      Read (nin,*) b(1:lb)

!     Allocate memory for output
      lout = max(lx,la,lb)
      Allocate (pdf(lout),ivalid(lout))

!     Calculate the PDF
      ifail = -1
      Call g01kkf(ilog,lx,x,la,a,lb,b,pdf,ivalid,ifail)

      If (ifail==0 .Or. ifail==1) Then
!       Display titles
        Write (nout,*) '    X         A         B         PDF      IVALID'
        Write (nout,*) repeat('-',50)

!       Display results
        Do i = 1, lout
          Write (nout,99999) x(mod(i-1,lx)+1), a(mod(i-1,la)+1), &
            b(mod(i-1,lb)+1), pdf(i), ivalid(i)
        End Do
      End If

99999 Format (1X,3(F6.2,4X),E10.3,4X,I3)
    End Program g01kkfe
```

10.2 Program Data

```
G01KKF Example Program Data
0                        :: ILOG
5                        :: LX
0.1 3.0 6.0 4.0 9.0      :: X
5                        :: LA
3.0 10.0 5.0 10.0 9.0    :: A
5                        :: LB
2.0 11.0 1.0 0.1 0.5     :: B
```

10.3 Program Results

```
G01KKF Example Program Results

     X         A         B         PDF       IVALID
-------------------------------------------------------
   0.10      3.00      2.00      0.595E-03       0
   3.00     10.00     11.00      0.159E-11       0
   6.00      5.00      1.00      0.134E+00       0
   4.00     10.00      0.10      0.307E-07       0
   9.00      9.00      0.50      0.833E-02       0
```