NAG Library Routine Document

D01UAF

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

D01UAF computes an estimate of the definite integral of a function of known analytical form, using a Gaussian quadrature formula with a specified number of abscissae. Formulae are provided for a finite interval (Gauss–Legendre), a semi-infinite interval (Gauss–Laguerre, rational Gauss), and an infinite interval (Gauss–Hermite).

2 Specification

```
SUBROUTINE D01UAF (KEY, A, B, N, F, DINEST, IUSER, RUSER, IFAIL)
INTEGER             KEY, N, IUSER(*), IFAIL
REAL (KIND=nag_wp) A, B, DINEST, RUSER(*)
EXTERNAL            F
```

3 Description

3.1 General

D01UAF evaluates an estimate of the definite integral of a function $f(x)$, over a finite or infinite range, by n-point Gaussian quadrature (see Davis and Rabinowitz (1975), FrÎberg (1970), Ralston (1965) or Stroud and Secrest (1966)). The integral is approximated by a summation of the product of a set of weights and a set of function evaluations at a corresponding set of abscissae x_i. For adjusted weights, the function values correspond to the values of the integrand f, and hence the sum will be

$$\sum_{i=1}^{n} w_i f(x_i)$$

where the w_i are called the weights, and the x_i the abscissae. A selection of values of n is available. (See Section 5.)

Where applicable, normal weights may instead be used, in which case the corresponding weight function ω is factored out of the integrand as $f(x) = \omega(x)g(x)$ and hence the sum will be

$$\sum_{i=-1}^{n} \bar{w}_i g(x),$$

where the normal weights $\bar{w}_i = w_i\omega(x_i)$ are computed internally.

D01UAF uses a vectorized F to evaluate the integrand or normalized integrand at a set of abscissae, x_i, for $i = 1, 2, \ldots, n_x$. If adjusted weights are used, the integrand $f(x_i)$ must be evaluated otherwise the normalized integrand $g(x_i)$ must be evaluated.

3.2 Both Limits Finite

$$\int_a^b f(x)\,dx.$$

The Gauss–Legendre weights and abscissae are used, and the formula is exact for any function of the form:

$$f(x) = \sum_{i=0}^{2n-1} c_i x^i.$$

The formula is appropriate for functions which can be well approximated by such a polynomial over $[a, b]$. It is inappropriate for functions with algebraic singularities at one or both ends of the interval, such as $(1+x)^{-1/2}$ on $[-1, 1]$.

3.3 One Limit Infinite

$$\int_a^{\infty} f(x)\,dx \quad \text{or} \quad \int_{-\infty}^{a} f(x)\,dx.$$

Two quadrature formulae are available for these integrals.

(a) The Gauss–Laguerre formula is exact for any function of the form:

$$f(x) = e^{-bx} \sum_{i=0}^{2n-1} c_i x^i.$$

This formula is appropriate for functions decaying exponentially at infinity; the argument b should be chosen if possible to match the decay rate of the function.

If the adjusted weights are selected, the complete integrand $f(x)$ should be provided through F.

If the normal form is selected, the contribution of e^{-bx} is accounted for internally, and F should only return $g(x)$, where $f(x) = e^{-bx} g(x)$.

If $b < 0$ is supplied, the interval of integration will be $[a, \infty)$. Otherwise if $b > 0$ is supplied, the interval of integration will be $(-\infty, a]$.

(b) The rational Gauss formula is exact for any function of the form:

$$f(x) = \sum_{i=2}^{2n+1} \frac{c_i}{(x+b)^i} = \frac{\sum_{i=0}^{2n-1} c_{2n+1-i} (x+b)^i}{(x+b)^{2n+1}}.$$

This formula is likely to be more accurate for functions having only an inverse power rate of decay for large x. Here the choice of a suitable value of b may be more difficult; unfortunately a poor choice of b can make a large difference to the accuracy of the computed integral.

Only the adjusted form of the rational Gauss formula is available, and as such, the complete integrand $f(x)$ must be supplied in F.

If $a + b < 0$, the interval of integration will be $[a, \infty)$. Otherwise if $a + b > 0$, the interval of integration will be $(-\infty, a]$.

3.4 Both Limits Infinite

$$\int_{-\infty}^{+\infty} f(x)\,dx.$$

The Gauss–Hermite weights and abscissae are used, and the formula is exact for any function of the form:

$$f(x) = e^{-b(x-a)^2} \sum_{i=0}^{2n-1} c_i x^i,$$

where $b > 0$. Again, for general functions not of this exact form, the argument b should be chosen to match if possible the decay rate at $\pm\infty$.

If the adjusted weights are selected, the complete integrand $f(x)$ should be provided through F.

If the normal form is selected, the contribution of $e^{-b(x-a)^2}$ is accounted for internally, and F should only return $g(x)$, where $f(x) = e^{-b(x-a)^2}g(x)$.

4 References

Davis P J and Rabinowitz P (1975) *Methods of Numerical Integration* Academic Press

FrÎberg C E (1970) *Introduction to Numerical Analysis* Addison–Wesley

Ralston A (1965) *A First Course in Numerical Analysis* pp. 87–90 McGraw–Hill

Stroud A H and Secrest D (1966) *Gaussian Quadrature Formulas* Prentice–Hall

5 Arguments

1: KEY – INTEGER *Input*

On entry: indicates the quadrature formula.

KEY = 0
Gauss–Legendre quadrature on a finite interval, using normal weights.

KEY = 3
Gauss–Laguerre quadrature on a semi-infinite interval, using normal weights.

KEY = −3
Gauss–Laguerre quadrature on a semi-infinite interval, using adjusted weights.

KEY = 4
Gauss–Hermite quadrature on an infinite interval, using normal weights.

KEY = −4
Gauss–Hermite quadrature on an infinite interval, using adjusted weights.

KEY = −5
Rational Gauss quadrature on a semi-infinite interval, using adjusted weights.

Constraint: KEY = 0, 3, −3, 4, −4 or −5.

2: A – REAL (KIND=nag_wp) *Input*

3: B – REAL (KIND=nag_wp) *Input*

On entry: the quantities a and b as described in the appropriate subsection of Section 3.

Constraints:

Rational Gauss: A + B ≠ 0.0;
Gauss–Laguerre: B ≠ 0.0;
Gauss–Hermite: B > 0.

4: N – INTEGER *Input*

On entry: n, the number of abscissae to be used.

Constraint: N = 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 16, 20, 24, 32, 48 or 64.

If the soft fail option is used, the answer is evaluated for the largest valid value of N less than the requested value.

5: F – SUBROUTINE, supplied by the user. *External Procedure*

F must return the value of the integrand f, or the normalized integrand g, at a specified point.

The specification of F is:

```
SUBROUTINE F (X, NX, FV, IFLAG, IUSER, RUSER)
INTEGER             NX, IFLAG, IUSER(*)
REAL (KIND=nag_wp) X(NX), FV(NX), RUSER(*)
```

1: X(NX) – REAL (KIND=nag_wp) array *Input*

On entry: the abscissae, x_i, for $i = 1, 2, \ldots, n_x$ at which function values are required.

2: NX – INTEGER *Input*

On entry: n_x, the number of abscissae.

3: FV(NX) – REAL (KIND=nag_wp) array *Output*

On exit: if adjusted weights are used, the values of the integrand f. FV$(i) = f(x_i)$, for $i = 1, 2, \ldots, n_x$.

Otherwise the values of the normalized integrand g. FV$(i) = g(x_i)$, for $i = 1, 2, \ldots, n_x$.

4: IFLAG – INTEGER *Input/Output*

On entry: IFLAG $= 0$.

On exit: set IFLAG < 0 if you wish to force an immediate exit from D01UAF with IFAIL $= -1$.

5: IUSER$(*)$ – INTEGER array *User Workspace*

6: RUSER$(*)$ – REAL (KIND=nag_wp) array *User Workspace*

F is called with the arguments IUSER and RUSER as supplied to D01UAF. You should use the arrays IUSER and RUSER to supply information to F.

F must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which D01UAF is called. Arguments denoted as *Input* must **not** be changed by this procedure.

Some points to bear in mind when coding F are mentioned in Section 7.

6: DINEST – REAL (KIND=nag_wp) *Output*

On exit: the estimate of the definite integral.

7: IUSER$(*)$ – INTEGER array *User Workspace*

8: RUSER$(*)$ – REAL (KIND=nag_wp) array *User Workspace*

IUSER and RUSER are not used by D01UAF, but are passed directly to F and should be used to pass information to this routine.

9: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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, because for this routine the values of the output arguments may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: 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).

Note: D01UAF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL $= 1$

The n-point rule is not among those stored.
On entry: N $= \langle value \rangle$.
n-point rule used: N $= \langle value \rangle$.

IFAIL $= 2$

Underflow occurred in calculation of normal weights.
Reduce N or use adjusted weights: N $= \langle value \rangle$.

IFAIL $= 3$

No nonzero weights were generated for the provided parameters.

IFAIL $= 11$

On entry, KEY $= \langle value \rangle$.
Constraint: KEY $= 0$, 3, -3, 4, -4 or -5.

IFAIL $= 12$

The value of A and/or B is invalid for the chosen KEY. Either:

The value of A and/or B is invalid.
On entry, KEY $= \langle value \rangle$.
On entry, A $= \langle value \rangle$ and B $= \langle value \rangle$.
Constraint: $|\mathrm{A} + \mathrm{B}| > 0.0$.

The value of A and/or B is invalid.
On entry, KEY $= \langle value \rangle$.
On entry, A $= \langle value \rangle$ and B $= \langle value \rangle$.
Constraint: $|\mathrm{B}| > 0.0$.

The value of A and/or B is invalid.
On entry, KEY $= \langle value \rangle$.
On entry, A $= \langle value \rangle$ and B $= \langle value \rangle$.
Constraint: B > 0.0.

IFAIL $= 14$

On entry, N $= \langle value \rangle$.
Constraint: N > 0.

IFAIL $= -1$

Exit requested from F with IFLAG $= \langle value \rangle$.

IFAIL $= -99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The accuracy depends on the behaviour of the integrand, and on the number of abscissae used. No tests are carried out in D01UAF to estimate the accuracy of the result. If such an estimate is required, the routine may be called more than once, with a different number of abscissae each time, and the answers compared. It is to be expected that for sufficiently smooth functions a larger number of abscissae will give improved accuracy.

Alternatively, the range of integration may be subdivided, the integral estimated separately for each sub-interval, and the sum of these estimates compared with the estimate over the whole range.

The coding of F may also have a bearing on the accuracy. For example, if a high-order Gauss–Laguerre formula is used, and the integrand is of the form

$$f(x) = e^{-bx}g(x)$$

it is possible that the exponential term may underflow for some large abscissae. Depending on the machine, this may produce an error, or simply be assumed to be zero. In any case, it would be better to evaluate the expression with

$$f(x) = \text{sgn}(g(x)) \times \exp(-bx + \ln|g(x)|)$$

Another situation requiring care is exemplified by

$$\int_{-\infty}^{+\infty} e^{-x^2}x^m\,dx = 0, \quad m \text{ odd}.$$

The integrand here assumes very large values; for example, when $m = 63$, the peak value exceeds 3×10^{33}. Now, if the machine holds floating-point numbers to an accuracy of k significant decimal digits, we could not expect such terms to cancel in the summation leaving an answer of much less than 10^{33-k} (the weights being of order unity); that is, instead of zero we obtain a rather large answer through rounding error. Such situations are characterised by great variability in the answers returned by formulae with different values of n.

In general, you should be aware of the order of magnitude of the integrand, and should judge the answer in that light.

8 Parallelism and Performance

D01UAF is currently neither directly nor indirectly threaded. In particular, the user-supplied argument F is not called from within a parallel region initialized inside D01UAF.

The user-supplied argument F uses a vectorized interface, allowing for the required vector of function values to be evaluated in parallel; for example by placing appropriate OpenMP directives in the code for the user-supplied argument F.

9 Further Comments

The time taken by D01UAF depends on the complexity of the expression for the integrand and on the number of abscissae required.

10 Example

This example evaluates the integrals

$$\int_0^1 \frac{4}{1+x^2}\,dx = \pi$$

by Gauss–Legendre quadrature;

$$\int_2^\infty \frac{1}{x^2 \ln x}\,dx = 0.378671$$

by rational Gauss quadrature with $b = 0$;

$$\int_2^\infty \frac{e^{-x}}{x}\,dx = 0.048901$$

by Gauss–Laguerre quadrature with $b = 1$; and

$$\int_{-\infty}^{+\infty} e^{-3x^2-4x-1}\,dx = \int_{-\infty}^{+\infty} e^{-3(x+1)^2} e^{2x+2} dx = 1.428167$$

by Gauss–Hermite quadrature with $a = -1$ and $b = 3$.

The formulae with $n = 2, 4, 8, 16, 32$ and 64 are used in each case. Both adjusted and normal weights are used for Gauss–Laguerre and Gauss–Hermite quadrature.

10.1 Program Text

```
!   D01UAF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module d01uafe_mod

!     D01UAF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
      Public                             :: d01uaf_f
!     .. Parameters ..
      Integer, Parameter, Public         :: i_funid = 1
      Integer, Parameter, Public         :: liuser = i_funid
      Integer, Parameter, Public         :: lruser = 1
    Contains
      Subroutine d01uaf_f(x,nx,fv,iflag,iuser,ruser)

!       .. Implicit None Statement ..
        Implicit None
!       .. Scalar Arguments ..
        Integer, Intent (Inout)          :: iflag
        Integer, Intent (In)             :: nx
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out) :: fv(nx)
        Real (Kind=nag_wp), Intent (Inout) :: ruser(*)
        Real (Kind=nag_wp), Intent (In) :: x(nx)
        Integer, Intent (Inout)          :: iuser(*)
!       .. Intrinsic Procedures ..
        Intrinsic                        :: exp, log
!       .. Executable Statements ..
        Select Case (iuser(i_funid))
        Case (1)
          fv = 4.0E0_nag_wp/(1.0E0_nag_wp+x*x)
        Case (2)
          fv = 1.0E0_nag_wp/(x*x*log(x))
```

```
        Case (3)
          fv = exp(-x)/x
        Case (4)
          fv = 1.0E0_nag_wp/x
        Case (5)
          fv = exp(-3.0E0_nag_wp*x*x-4.0E0_nag_wp*x-1.0E0_nag_wp)
        Case (6)
          fv = exp(2.0E0_nag_wp*x+2.0E0_nag_wp)
        Case Default
          iflag = -1
        End Select
      End Subroutine d01uaf_f
    End Module d01uafe_mod
    Program d01uafe

!     D01UAF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: d01uaf, nag_wp
      Use d01uafe_mod, Only: d01uaf_f, i_funid, liuser, lruser
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: a, b, dinest
      Integer                          :: funid, i, ifail, key, nstor
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: ruser(lruser)
      Integer                          :: iuser(liuser)
!     .. Executable Statements ..
      Write (nout,*) 'D01UAF Example Program Results'

cases: Do funid = 1, 6
        Write (nout,*)
        Select Case (funid)
        Case (1)
          Write (nout,*) 'Gauss-Legendre example'
          a = 0.0_nag_wp
          b = 1.0_nag_wp
          key = 0
        Case (2)
          Write (nout,*) 'Rational Gauss example'
          a = 2.0_nag_wp
          b = 0.0_nag_wp
          key = -5
        Case (3)
          Write (nout,*) 'Gauss-Laguerre example (adjusted weights)'
          a = 2.0_nag_wp
          b = 1.0_nag_wp
          key = -3
        Case (4)
          Write (nout,*) 'Gauss-Laguerre example (normal weights)'
          a = 2.0_nag_wp
          b = 1.0_nag_wp
          key = 3
        Case (5)
          Write (nout,*) 'Gauss-Hermite example (adjusted weights)'
          a = -1.0_nag_wp
          b = 3.0_nag_wp
          key = -4
        Case (6)
          Write (nout,*) 'Gauss-Hermite example (normal weights)'
          a = -1.0_nag_wp
          b = 3.0_nag_wp
          key = 4
        End Select
        iuser(i_funid) = funid

        Do i = 1, 6
          nstor = 2**(i)
```

```
          ifail = -1
          Call d01uaf(key,a,b,nstor,d01uaf_f,dinest,iuser,ruser,ifail)
          Select Case (ifail)
          Case (:-1)
!           Error flag returned by d01uaf_f
            Exit cases
          Case (0,1)
!           The definite integral has been estimated.
            Write (nout,99999) nstor, dinest
          Case Default
!           Illegal parameters on entry to d01uaf
            Exit cases
          End Select
        End Do
        Write (nout,*)

      End Do cases

99999 Format (1X,I5,' Points     Answer = ',F10.5)
    End Program d01uafe
```

10.2 Program Data

None.

10.3 Program Results

```
 D01UAF Example Program Results

 Gauss-Legendre example
     2 Points     Answer =    3.14754
     4 Points     Answer =    3.14161
     8 Points     Answer =    3.14159
    16 Points     Answer =    3.14159
    32 Points     Answer =    3.14159
    64 Points     Answer =    3.14159


 Rational Gauss example
     2 Points     Answer =    0.37989
     4 Points     Answer =    0.37910
     8 Points     Answer =    0.37876
    16 Points     Answer =    0.37869
    32 Points     Answer =    0.37867
    64 Points     Answer =    0.37867


 Gauss-Laguerre example (adjusted weights)
     2 Points     Answer =    0.04833
     4 Points     Answer =    0.04887
     8 Points     Answer =    0.04890
    16 Points     Answer =    0.04890
    32 Points     Answer =    0.04890
    64 Points     Answer =    0.04890


 Gauss-Laguerre example (normal weights)
     2 Points     Answer =    0.04833
     4 Points     Answer =    0.04887
     8 Points     Answer =    0.04890
    16 Points     Answer =    0.04890
    32 Points     Answer =    0.04890
    64 Points     Answer =    0.04890


 Gauss-Hermite example (adjusted weights)
     2 Points     Answer =    1.38381
     4 Points     Answer =    1.42803
     8 Points     Answer =    1.42817
    16 Points     Answer =    1.42817
```

```
    32 Points     Answer =    1.42817
    64 Points     Answer =    1.42817


 Gauss-Hermite example (normal weights)
     2 Points     Answer =    1.38381
     4 Points     Answer =    1.42803
     8 Points     Answer =    1.42817
    16 Points     Answer =    1.42817
    32 Points     Answer =    1.42817
    64 Points     Answer =    1.42817
```