NAG Library Routine Document

D01ANF

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

D01ANF calculates an approximation to the sine or the cosine transform of a function g over $[a, b]$:

$$I = \int_a^b g(x)\sin(\omega x)\,dx \qquad \text{or} \qquad I = \int_a^b g(x)\cos(\omega x)\,dx$$

(for a user-specified value of ω).

2 Specification

```
SUBROUTINE D01ANF (G, A, B, OMEGA, KEY, EPSABS, EPSREL, RESULT, ABSERR, W,      &
                   LW, IW, LIW, IFAIL)

INTEGER            KEY, LW, IW(LIW), LIW, IFAIL
REAL (KIND=nag_wp) G, A, B, OMEGA, EPSABS, EPSREL, RESULT, ABSERR, W(LW)
EXTERNAL           G
```

3 Description

D01ANF is based on the QUADPACK routine QFOUR (see Piessens *et al.* (1983)). It is an adaptive routine, designed to integrate a function of the form $g(x)w(x)$, where $w(x)$ is either $\sin(\omega x)$ or $\cos(\omega x)$. If a sub-interval has length

$$L = |b - a|2^{-l}$$

then the integration over this sub-interval is performed by means of a modified Clenshaw–Curtis procedure (see Piessens and Branders (1975)) if $L\omega > 4$ and $l \leq 20$. In this case a Chebyshev series approximation of degree 24 is used to approximate $g(x)$, while an error estimate is computed from this approximation together with that obtained using Chebyshev series of degree 12. If the above conditions do not hold then Gauss 7-point and Kronrod 15-point rules are used. The algorithm, described in Piessens *et al.* (1983), incorporates a global acceptance criterion (as defined in Malcolm and Simpson (1976)) together with the ϵ-algorithm (see Wynn (1956)) to perform extrapolation. The local error estimation is described in Piessens *et al.* (1983).

4 References

Malcolm M A and Simpson R B (1976) Local versus global strategies for adaptive quadrature *ACM Trans. Math. Software* **1** 129–146

Piessens R and Branders M (1975) Algorithm 002: computation of oscillating integrals *J. Comput. Appl. Math.* **1** 153–164

Piessens R, de Doncker–Kapenga E, Überhuber C and Kahaner D (1983) *QUADPACK, A Subroutine Package for Automatic Integration* Springer–Verlag

Wynn P (1956) On a device for computing the $e_m(S_n)$ transformation *Math. Tables Aids Comput.* **10** 91–96

5 Parameters

1: G – REAL (KIND=nag_wp) FUNCTION, supplied by the user. *External Procedure*

G must return the value of the function g at a given point X.

The specification of G is:

```
FUNCTION G (X)
REAL (KIND=nag_wp) G
REAL (KIND=nag_wp) X
```

1: X – REAL (KIND=nag_wp) *Input*

On entry: the point at which the function g must be evaluated.

G must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub)program from which D01ANF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

2: A – REAL (KIND=nag_wp) *Input*

On entry: a, the lower limit of integration.

3: B – REAL (KIND=nag_wp) *Input*

On entry: b, the upper limit of integration. It is not necessary that $a < b$.

4: OMEGA – REAL (KIND=nag_wp) *Input*

On entry: the parameter ω in the weight function of the transform.

5: KEY – INTEGER *Input*

On entry: indicates which integral is to be computed.

KEY $= 1$
: $w(x) = \cos(\omega x)$.

KEY $= 2$
: $w(x) = \sin(\omega x)$.

Constraint: KEY $= 1$ or 2.

6: EPSABS – REAL (KIND=nag_wp) *Input*

On entry: the absolute accuracy required. If EPSABS is negative, the absolute value is used. See Section 7.

7: EPSREL – REAL (KIND=nag_wp) *Input*

On entry: the relative accuracy required. If EPSREL is negative, the absolute value is used. See Section 7.

8: RESULT – REAL (KIND=nag_wp) *Output*

On exit: the approximation to the integral I.

9: ABSERR – REAL (KIND=nag_wp) *Output*

On exit: an estimate of the modulus of the absolute error, which should be an upper bound for $|I - \text{RESULT}|$.

10: W(LW) – REAL (KIND=nag_wp) array *Output*

On exit: details of the computation see Section 8 for more information.

11: LW – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which D01ANF is called. The value of LW (together with that of LIW) imposes a bound on the number of sub-intervals into which the interval of integration may be divided by the routine. The number of sub-intervals cannot exceed LW/4. The more difficult the integrand, the larger LW should be.

Suggested value: LW = 800 to 2000 is adequate for most problems.

Constraint: LW $\geq$ 4.

12: IW(LIW) – INTEGER array *Output*

On exit: IW(1) contains the actual number of sub-intervals used. The rest of the array is used as workspace.

13: LIW – INTEGER *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which D01ANF is called. The number of sub-intervals into which the interval of integration may be divided cannot exceed LIW/2.

Suggested value: LIW = LW/2.

Constraint: LIW $\geq$ 2.

14: 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, because for this routine the values of the output parameters 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: D01ANF 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 maximum number of subdivisions allowed with the given workspace has been reached without the accuracy requirements being achieved. Look at the integrand in order to determine the integration difficulties. If the position of a local difficulty within the interval can be determined (e.g., a singularity of the integrand or its derivative, a peak, a discontinuity, etc.) you will probably gain from splitting up the interval at this point and calling the integrator on the subranges. If necessary, another integrator, which is designed for handling the type of difficulty involved, must be used. Alternatively, consider relaxing the accuracy requirements specified by EPSABS and EPSREL, or increasing the amount of workspace.

IFAIL $= 2$

Round-off error prevents the requested tolerance from being achieved. Consider requesting less accuracy.

IFAIL $= 3$

Extremely bad local behaviour of $g(x)$ causes a very strong subdivision around one (or more) points of the interval. The same advice applies as in the case of IFAIL $= 1$.

IFAIL $= 4$

The requested tolerance cannot be achieved because the extrapolation does not increase the accuracy satisfactorily; the returned result is the best which can be obtained. The same advice applies as in the case of IFAIL $= 1$.

IFAIL $= 5$

The integral is probably divergent, or slowly convergent. Please note that divergence can occur with any nonzero value of IFAIL.

IFAIL $= 6$

On entry, KEY $\neq 1$ or 2.

IFAIL $= 7$

On entry, LW < 4,
or LIW < 2.

7 Accuracy

D01ANF cannot guarantee, but in practice usually achieves, the following accuracy:

$$|I - \text{RESULT}| \leq tol,$$

where

$$tol = \max\{|\text{EPSABS}|, |\text{EPSREL}| \times |I|\},$$

and EPSABS and EPSREL are user-specified absolute and relative tolerances. Moreover, it returns the quantity ABSERR which in normal circumstances, satisfies

$$|I - \text{RESULT}| \leq \text{ABSERR} \leq tol.$$

8 Further Comments

The time taken by D01ANF depends on the integrand and the accuracy required.

If IFAIL $\neq 0$ on exit, then you may wish to examine the contents of the array W, which contains the end points of the sub-intervals used by D01ANF along with the integral contributions and error estimates over these sub-intervals.

Specifically, for $i = 1, 2, \ldots, n$, let r_i denote the approximation to the value of the integral over the sub-interval $[a_i, b_i]$ in the partition of $[a, b]$ and e_i be the corresponding absolute error estimate. Then, $\int_{a_i}^{b_i} g(x)w(x)\,dx \simeq r_i$ and RESULT $= \sum_{i=1}^{n} r_i$ unless D01ANF terminates while testing for divergence of the integral (see Section 3.4.3 of Piessens *et al.* (1983)). In this case, RESULT (and ABSERR) are taken to be the values returned from the extrapolation process. The value of n is returned in IW(1), and the values a_i, b_i, e_i and r_i are stored consecutively in the array W, that is:

$a_i = \text{W}(i)$,

$b_i = \text{W}(n + i)$,

$e_i = \mathrm{W}(2n + i)$ and

$r_i = \mathrm{W}(3n + i)$.

9 Example

This example computes

$$\int_0^1 \ln x \sin(10\pi x)\, dx.$$

9.1 Program Text

```
!   D01ANF Example Program Text
!   Mark 24 Release. NAG Copyright 2012.

    Module d01anfe_mod

!     D01ANF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                   :: lw = 800, nout = 6
      Integer, Parameter                   :: liw = lw/2
    Contains
      Function g(x)

!       .. Function Return Value ..
        Real (Kind=nag_wp)                   :: g
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (In)      :: x
!       .. Intrinsic Procedures ..
        Intrinsic                            :: log
!       .. Executable Statements ..
        If (x>0.0E0_nag_wp) Then
          g = log(x)
        Else
          g = 0.0E0_nag_wp
        End If

        Return

      End Function g
    End Module d01anfe_mod
    Program d01anfe

!     D01ANF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: d01anf, nag_wp, x01aaf
      Use d01anfe_mod, Only: g, liw, lw, nout
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)                   :: a, abserr, b, epsabs, epsrel,    &
                                              omega, pi, result
      Integer                              :: ifail, key
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable      :: w(:)
      Integer, Allocatable                 :: iw(:)
!     .. Executable Statements ..
      Write (nout,*) 'D01ANF Example Program Results'

      Allocate (w(lw),iw(liw))
```

```
      epsrel = 1.0E-04_nag_wp
      epsabs = 0.0E+00_nag_wp
      a = 0.0E0_nag_wp
      b = 1.0E0_nag_wp
      omega = 10.0E0_nag_wp*x01aaf(pi)
      key = 2

      ifail = -1
      Call d01anf(g,a,b,omega,key,epsabs,epsrel,result,abserr,w,lw,iw,liw, &
        ifail)

      If (ifail>=0) Then
        Write (nout,*)
        Write (nout,99999) 'A     ', 'lower limit of integration', a
        Write (nout,99999) 'B     ', 'upper limit of integration', b
        Write (nout,99998) 'EPSABS', 'absolute accuracy requested', epsabs
        Write (nout,99998) 'EPSREL', 'relative accuracy requested', epsrel
      End If

      If (ifail>=0 .And. ifail<=5) Then
        Write (nout,*)
        Write (nout,99997) 'RESULT', 'approximation to the integral', result
        Write (nout,99998) 'ABSERR', 'estimate of the absolute error', abserr
        Write (nout,99996) 'IW(1) ', 'number of subintervals used', iw(1)
      End If

99999 Format (1X,A6,' - ',A32,' = ',F10.4)
99998 Format (1X,A6,' - ',A32,' = ',E9.2)
99997 Format (1X,A6,' - ',A32,' = ',F9.5)
99996 Format (1X,A6,' - ',A32,' = ',I4)
    End Program d01anfe
```

9.2 Program Data

None.

9.3 Program Results

```
 D01ANF Example Program Results

 A      -       lower limit of integration =     0.0000
 B      -       upper limit of integration =     1.0000
 EPSABS -      absolute accuracy requested =  0.00E+00
 EPSREL -      relative accuracy requested =  0.10E-03

 RESULT -    approximation to the integral =  -0.12814
 ABSERR -   estimate of the absolute error =  0.36E-05
 IW(1)  -      number of subintervals used =    8
```