NAG Library Routine Document

D01ALF

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

D01ALF is a general purpose integrator which calculates an approximation to the integral of a function $f(x)$ over a finite interval $[a, b]$:

$$I = \int_a^b f(x)\,dx$$

where the integrand may have local singular behaviour at a finite number of points within the integration interval.

2 Specification

```
 SUBROUTINE D01ALF(F, A, B, NPTS, POINTS, EPSABS, EPSREL, RESULT, ABSERR,
1                  W, LW, IW, LIW, IFAIL)
 INTEGER           NPTS, LW, IW(LIW), LIW, IFAIL
 double precision  F, A, B, POINTS(*), EPSABS, EPSREL, RESULT, ABSERR,
1                  W(LW)
 EXTERNAL          F
```

3 Description

D01ALF is based on the QUADPACK routine QAGP (see Piessens *et al.* (1983)). It is very similar to D01AJF, but allows you to supply 'break points', points at which the integrand is known to be difficult. It employs an adaptive algorithm, using the Gauss 10-point and Kronrod 21-point rules. The algorithm, described in de Doncker (1978), incorporates a global acceptance criterion (as defined by Malcolm and Simpson (1976)) together with the ϵ-algorithm (see Wynn (1956)) to perform extrapolation. The user-supplied 'break points' always occur as the end points of some sub-interval during the adaptive process. The local error estimation is described in Piessens *et al.* (1983).

4 References

de Doncker E (1978) An adaptive extrapolation algorithm for automatic integration *ACM SIGNUM Newsl.* **13 (2)** 12–18

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

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: F – ***double precision*** FUNCTION, supplied by the user. *External Procedure*

F must return the value of the integrand f at a given point.

The specification of F is:

double precision `FUNCTION F(X)`
double precision `X`

1: X – ***double precision*** *Input*

On entry: the point at which the integrand f must be evaluated.

F must be declared as EXTERNAL in the (sub)program from which D01ALF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

2: A – ***double precision*** *Input*

On entry: a, the lower limit of integration.

3: B – ***double precision*** *Input*

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

4: NPTS – INTEGER *Input*

On entry: the number of user-supplied break points within the integration interval.

Constraint: NPTS ≥ 0.

5: POINTS$(*)$ – ***double precision*** array *Input*

Note: the dimension of the array POINTS must be at least $\max(1, \text{NPTS})$.

On entry: the user-specified break points.

Constraint: the break points must all lie within the interval of integration (but may be supplied in any order).

6: EPSABS – ***double precision*** *Input*

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

7: EPSREL – ***double precision*** *Input*

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

8: RESULT – ***double precision*** *Output*

On exit: the approximation to the integral I.

9: ABSERR – ***double precision*** *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) – ***double precision*** array *Output*

On exit: details of the computation, as described in Section 8.

11: LW – INTEGER *Input*

On entry: the dimension of the array W as declared in the (sub)program from which D01ALF 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 $(\text{LW} - 2 \times \text{NPTS} - 4)/4$. The more difficult the integrand, the larger LW should be.

Suggested value: a value in the range 800 to 2000 is adequate for most problems.

Constraint: LW $\geq 2 \times$ NPTS $+ 8$.

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 D01ALF is called. The number of sub-intervals into which the interval of integration may be divided cannot exceed $(\text{LIW} - \text{NPTS} - 2)/2$.

Suggested value: $\text{LIW} = \text{LW}/2$.

Constraint: LIW $\geq$ NPTS $+ 4$.

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.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 6).

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.**

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: D01ALF 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.) it should be supplied to the routine as an element of the vector POINTS. 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 integrand behaviour 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 $= 6$

The input is invalid: break points are specified outside the integration range, NPTS $>$ LIMIT or NPTS < 0. RESULT and ABSERR are set to zero.

IFAIL $= 7$

On entry, LW $< 2 \times$ NPTS $+ 8$,
or LIW $<$ NPTS $+ 4$.

7 Accuracy

D01ALF 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 error 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 D01ALF 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 D01ALF 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, $\displaystyle\int_{a_i}^{b_i} f(x)\,dx \simeq r_i$ and RESULT $= \displaystyle\sum_{i=1}^{n} r_i$ unless D01ALF 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:

$a_i =$ W(i),

$b_i =$ W$(n + i)$,

$e_i =$ W$(2n + i)$ and

$r_i =$ W$(3n + i)$.

9 Example

This example computes

$$\int_0^1 \frac{1}{\sqrt{|x-1/7|}}\,dx.$$

A break point is specified at $x = 1/7$, at which point the integrand is infinite. (For definiteness the function FST returns the value 0.0 at this point.)

9.1 Program Text

```
*     D01ALF Example Program Text
*     Mark 14 Revised. NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPTS, LW, LIW
      PARAMETER        (NPTS=1,LW=800,LIW=LW/2)
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Scalars in Common ..
      INTEGER          KOUNT
*     .. Local Scalars ..
      DOUBLE PRECISION A, ABSERR, B, EPSABS, EPSREL, RESULT
      INTEGER          IFAIL
*     .. Local Arrays ..
      DOUBLE PRECISION POINTS(NPTS), W(LW)
      INTEGER          IW(LIW)
*     .. External Functions ..
      DOUBLE PRECISION F
      EXTERNAL         F
*     .. External Subroutines ..
      EXTERNAL         D01ALF
*     .. Common blocks ..
      COMMON           /TELNUM/KOUNT
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D01ALF Example Program Results'
      EPSABS = 0.0D0
      EPSREL = 1.0D-03
      A = 0.0D0
      B = 1.0D0
      POINTS(1) = 1.0D0/7.0D0
      KOUNT = 0
      IFAIL = 1
*
      CALL D01ALF(F,A,B,NPTS,POINTS,EPSABS,EPSREL,RESULT,ABSERR,W,LW,IW,
     +            LIW,IFAIL)
*
      IF (IFAIL.GE.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
         WRITE (NOUT,99999) 'POINTS(1) - given break-point = ',
     +     POINTS(1)
         WRITE (NOUT,*)
         IF (IFAIL.NE.0) WRITE (NOUT,99996) 'IFAIL = ', IFAIL
         IF (IFAIL.LE.5) THEN
            WRITE (NOUT,99997)
     +        '  RESULT - approximation to the integral = ', RESULT
            WRITE (NOUT,99998)
     +        '  ABSERR - estimate of the absolute error = ', ABSERR
            WRITE (NOUT,99996)
     +        '  KOUNT  - number of function evaluations = ', KOUNT
            WRITE (NOUT,99996) 'IW(1)  - number of subintervals used = '
     +        , IW(1)
         END IF
```

```
      ELSE
         WRITE (NOUT,*)
         WRITE (NOUT,99995) ' ** D01ALF returned with IFAIL = ', IFAIL
      END IF
*
99999 FORMAT (1X,A,F10.4)
99998 FORMAT (1X,A,E9.2)
99997 FORMAT (1X,A,F9.5)
99996 FORMAT (1X,A,I4)
99995 FORMAT (1X,A,I5)
      END
*
      DOUBLE PRECISION FUNCTION F(X)
*     .. Scalar Arguments ..
      DOUBLE PRECISION X
*     .. Scalars in Common ..
      INTEGER          KOUNT
*     .. Local Scalars ..
      DOUBLE PRECISION A
*     .. Intrinsic Functions ..
      INTRINSIC        ABS
*     .. Common blocks ..
      COMMON           /TELNUM/KOUNT
*     .. Executable Statements ..
      KOUNT = KOUNT + 1
      A = ABS(X-1.0D0/7.0D0)
      F = 0.0D0
      IF (A.NE.0.0D0) F = A**(-0.5D0)
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D01ALF 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-02
 POINTS(1) - given break-point =     0.1429

   RESULT - approximation to the integral =   2.60757
   ABSERR - estimate of the absolute error =  0.62E-13
   KOUNT  - number of function evaluations =  462
 IW(1)  - number of subintervals used =   12
```