D01ATFP

NAG Parallel Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details. You are advised to enclose any calls to NAG Parallel Library routines between calls to Z01AAFP and Z01ABFP.

1 Description

D01ATFP is a general-purpose integrator which calculates an approximation to the integral of a function $f(x)$ over a finite interval $[a, b]$:

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

The routine requires a user-supplied subroutine to evaluate the integrand at an array of different points and is therefore particularly efficient when the evaluation can be performed in vector mode on a vector-processing machine.

2 Specification

```
SUBROUTINE D01ATFP(ICNTXT, F, A, B, EPSABS, EPSREL, RESULT,
1                  ABSERR, NFUN, WORK, LW, IWORK, LIW, IFAIL)
DOUBLE PRECISION   A, B, EPSABS, EPSREL, RESULT, ABSERR, WORK(LW)
INTEGER            ICNTXT, NFUN, LW, IWORK(LIW), LIW, IFAIL
EXTERNAL           F
```

3 Data Distribution

3.1 Definitions

The following definitions are used in describing the data distribution within this document:

m_p	–	the number of rows in the logical processor grid,
n_p	–	the number of columns in the logical processor grid,
p	–	$m_p \times n_p$, the total number of processors in the logical processor grid.

3.2 Global and Local Arguments

The input arguments A, B, EPSABS, EPSREL, LW, LIW and IFAIL are global and so must have the same value on entry to the routine on each processor. The output arguments RESULT, ABSERR, NFUN and IFAIL are global, and so will have the same value on exit from the routine on each processor. The external procedure F is global. The remaining arguments are local.

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the BLACS context used by the communication mechanism, usually returned by a call to Z01AAFP.

2: F — SUBROUTINE, supplied by the user. *Global External Procedure*

F must return the values of the integrand f at a set of points.

Its specification is:

```
SUBROUTINE       F(X, FV, N)
DOUBLE PRECISION X(N), FV(N)
INTEGER          N
```

1: X(N) — DOUBLE PRECISION array *Local Input*
On entry: the points at which the integrand f must be evaluated.

2: FV(N) — DOUBLE PRECISION array *Local Output*
On exit: FV(j) must contain the value of f at the point X(j), for $j = 1, 2, \ldots,$N.

3: N — INTEGER *Global Input*
On entry: the number of points at which the integrand is to be evaluated. The actual value of N is always 21 which is the number of points in the Kronrod rule being used.

F must be declared as EXTERNAL in the (sub)program from which D01ATFP is called. Arguments denoted as *Input* must **not** be changed by this procedure.

3: A — DOUBLE PRECISION *Global Input*
On entry: the lower limit of integration, a.

4: B — DOUBLE PRECISION *Global Input*
On entry: the upper limit of integration, b. It is not necessary that $a < b$.

5: EPSABS — DOUBLE PRECISION *Global Input*
On entry: the absolute accuracy required. If EPSABS is negative, the absolute value is used. See Section 6.3.

6: EPSREL — DOUBLE PRECISION *Global Input*
On entry: the relative accuracy required. If EPSREL is negative, the absolute value is used. See Section 6.3.

7: RESULT — DOUBLE PRECISION *Global Output*
On exit: the approximation to the integral I.

8: ABSERR — DOUBLE PRECISION *Global Output*
On exit: an estimate of the modulus of the absolute error, which should be an upper bound for $|I - \text{RESULT}|$.

9: NFUN — INTEGER *Global Output*
On exit: the total number of evaluations of function $f(x)$ used in computing the integral.

10: WORK(LW) — DOUBLE PRECISION array *Local Workspace*

11: LW — INTEGER *Global Input*
On entry: the dimension of the array WORK as declared in the (sub)program from which D01ATFP is called. The value of LW (together with that of LIW below) imposes a bound on the number of sub-intervals into which the interval of integration may be divided by the routine on each processor (see Section 6.2). The number of sub-intervals on each processor cannot exceed LW/$(4p + 4)$ (see Section 3.1 for the definition of p). The more difficult the integrand, the larger LW should be.

Suggested value: a value in the range $400(p+1)$ to $800(p+1)$ should be adequate for most problems.

Constraint: LW $\geq 4(p + 1)$.

12: IWORK(LIW) — INTEGER array *Local Workspace*

13: LIW — INTEGER *Global Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D01ATFP is called. The number of sub-intervals into which the interval of integration may be divided cannot exceed LIW/p on each processor (see Section 3.1 for the definition of p).

Suggested value: LIW = (LW$\times$ p)/$(4p + 4)$.

Constraint: LIW $\geq p$.

14: IFAIL — INTEGER *Global Input/Global Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in the Essential Introduction) the recommended values are:

IFAIL = 0, if multigridding is **not** employed;
IFAIL = -1, if multigridding is employed.

On exit: IFAIL = 0 unless the routine detects an error (see Section 5).

5 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output from the root processor (or processor {0,0} when the root processor is not available) on the current error message unit (as defined by X04AAF).

Errors or warnings specified by the routine:

IFAIL = -2000

The routine has been called with an invalid value of ICNTXT on one or more processors.

IFAIL = -1000

The logical processor grid and library mechanism (Library Grid) have not been correctly defined, see Z01AAFP.

IFAIL = $-i$

On entry, the ith (global) argument did not have the same value on all logical processors (see Section 3.2).

IFAIL = 1

The maximum number of subdivisions allowed with the given workspace has been reached on one of the processors 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 discontinuity, etc.) you will probably gain from splitting up the interval at this point and calling the integrator on the sub-intervals. 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 accuracy from being achieved on a sub-interval computed by a processsor. Consider requesting less accuracy.

IFAIL = 3

Extremely bad local integrand behaviour causes a very strong subdivision around one (or more) points of an interval processed by one of the processors. The same advice applies as in the case of IFAIL = 1.

IFAIL = 4

The requested accuracy cannot be achieved because the extrapolation does not increase the accuracy satisfactorily on a sub-interval evaluated by one of the processors; 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 on a sub-interval evaluated by one of the processors. Note that divergence can result for any value of IFAIL = 1,. . .,4.

IFAIL = 6

On entry, LW $< 4(p+1)$,
or LIW $< p$ (see Section 3.1 for the definition of p).

6 Further Comments

6.1 Algorithmic Detail

D01ATFP is a modified version of the QUADPACK routine QAGS (Piessens *et al.* [3]). It is an adaptive routine, using the Gauss 10-point and Kronrod 21-point rules. The algorithm, described by de Doncker [1], incorporates a global acceptance criterion (as defined by Malcolm and Simpson [2]) together with the ϵ-algorithm (Wynn [4]) to perform extrapolation. The local error estimation is described by Piessens *et al.* [3].

The routine is suitable as a general purpose integrator, and can be used when the integrand has singularities, especially when these are of algebraic or logarithmic type.

6.2 Parallelism Detail

The routine initially subdivides the interval of integration into p sub-intervals of equal length. Then a modified version of the QUADPACK routine QAGS is applied to each sub-interval. If the required accuracy is achieved then the process is terminated. Otherwise, if convergence is achieved only on some processors then the other processors are interrupted and are marked as unfinished. Under certain criteria, some local sub-intervals associated with the unfinished processors are collected and then redistributed across all the processors. This procedure is repeated until the required accuracy is achieved.

6.3 Accuracy

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

$$|I - \texttt{RESULT}| \leq tol$$

where

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

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

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

6.4 Processor Limit

This routine is currently restricted to a total of no more than 2048 logical processors.

7 References

[1] De Doncker E (1978) An adaptive extrapolation algorithm for automatic integration *SIGNUM Newsl.* **13 (2)** 12–18

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

[3] Piessens R, De Doncker-Kapenga E, Überhuber C and Kahaner D (1983) *QUADPACK, A Subroutine Package for Automatic Integration* Springer-Verlag

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

8 Example

The following integral is evaluated using D01ATFP:

$$\int_0^5 x^{-0.8} \sum_{k=1}^{5} \cos(1000kx)\, dx.$$

8.1 Example Text

```
*     D01ATFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          MAXNP, MAXMP, MAXSUB, LW, LIW
      PARAMETER        (MAXNP=4,MAXMP=4,MAXSUB=400,LW=4*(MAXMP*MAXNP+1)
     +                 *MAXSUB,LIW=MAXMP*MAXNP*MAXSUB)
*     .. Local Scalars ..
      DOUBLE PRECISION A, ABSERR, B, EPSABS, EPSREL, RESULT
      INTEGER          ICNTXT, IFAIL, MP, NFUN, NP
      LOGICAL          ROOT
*     .. Local Arrays ..
      DOUBLE PRECISION WORK(LW)
      INTEGER          IWORK(LIW)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         D01ATFP, F, Z01AAFP, Z01ABFP
*     .. Executable Statements ..

      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'D01ATFP Example Program Results'
*
      MP = 2
      NP = 2
      IFAIL = 0
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      A = 0.0D0
      B = 5.0D0
      EPSABS = 0.0D0
      EPSREL = 1.0D-6
      IFAIL = -1
*
      CALL D01ATFP(ICNTXT,F,A,B,EPSABS,EPSREL,RESULT,ABSERR,NFUN,WORK,
     +             LW,IWORK,LIW,IFAIL)
*
      IF (ROOT) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'A      - lower limit of integration  ='
     +     , A
         WRITE (NOUT,99998) 'B      - upper limit of integration  ='
     +     , B
         WRITE (NOUT,99997) 'EPSABS - absolute accuracy requested ='
     +     , EPSABS
         WRITE (NOUT,99997) 'EPSREL - relative accuracy requested ='
```

```
     +     , EPSREL
           WRITE (NOUT,99999) 'Number of tasks                      ='
     +     , NP*MP
           WRITE (NOUT,*)
           IF (IFAIL.NE.0) WRITE (NOUT,99999) 'IFAIL is =', IFAIL
           IF (IFAIL.LE.5) THEN
              WRITE (NOUT,99998) 'Computed result is             =',
     +        RESULT
              WRITE (NOUT,99997) 'Computed error is              =',
     +        ABSERR
              WRITE (NOUT,99999) 'No. of function evaluations is =',
     +        NFUN
           END IF
         END IF
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'No',IFAIL)
*
      STOP
*
99999 FORMAT (1X,A,I12)
99998 FORMAT (1X,A,F12.4)
99997 FORMAT (1X,A,E12.2)
      END

      SUBROUTINE F(X,FV,N)
*     .. Scalar Arguments ..
      INTEGER      N
*     .. Array Arguments ..
      DOUBLE PRECISION FV(N), X(N)
*     .. Local Scalars ..
      DOUBLE PRECISION SUM
      INTEGER      I, K
*     .. Intrinsic Functions ..
      INTRINSIC    COS, DBLE
*     .. Executable Statements ..
*
      DO 40 I = 1, N
         SUM = 0.0D0
         DO 20 K = 1, 5
            SUM = SUM + COS(1000.D0*DBLE(K)*X(I))
   20    CONTINUE
         FV(I) = X(I)**(-0.8D0)*SUM
   40 CONTINUE
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
D01ATFP Example Program Results

A      - lower limit of integration  =      0.0000
B      - upper limit of integration  =      5.0000
EPSABS - absolute accuracy requested =    0.00E+00
```

```
EPSREL - relative accuracy requested =    0.10E-05
Number of tasks                      =           4

Computed result is             =      4.5577
Computed error is              =    0.81E-08
No. of function evaluations is =       85953
```
