D01AUFP

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

D01AUFP is an adaptive integrator, especially suited to oscillating, non-singular integrands, 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 D01AUFP(ICNTXT, F, A, B, EPSABS, EPSREL, KEY, RESULT,
     1                   ABSERR, NFUN, WORK, LW, IWORK, LIW, IFAIL)
      DOUBLE PRECISION   A, B, EPSABS, EPSREL, RESULT, ABSERR, WORK(LW)
      INTEGER            ICNTXT, KEY, NFUN, LW, IWORK(LIW), LIW, IFAIL
      EXTERNAL           F
```

3 Usage

3.1 Definitions

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

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

3.2 Global and Local Arguments

The following global **input** arguments must have the same value on entry to the routine on each processor and the global **output** arguments will have the same value on exit from the routine on each processor:

Global input arguments:	A, B, EPSABS, EPSREL, KEY, LW, LIW, IFAIL
Global output arguments:	RESULT, ABSERR, NFUN, IFAIL

The remaining arguments are local.

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP.

Note: the value of ICNTXT **must not** be changed.

2: F — SUBROUTINE, supplied by the user. *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 equal to the number of points in the Kronrod rule (see specification of KEY below).

F must be declared as EXTERNAL in the (sub)program from which D01AUFP 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: KEY — INTEGER *Global Input*

On entry: which integration rule is to be used:

if KEY = 1 for the Gauss 7-point and Kronrod 15-point rule,
if KEY = 2 for the Gauss 10-point and Kronrod 21-point rule,
if KEY = 3 for the Gauss 15-point and Kronrod 31-point rule,
if KEY = 4 for the Gauss 20-point and Kronrod 41-point rule,
if KEY = 5 for the Gauss 25-point and Kronrod 51-point rule,
if KEY = 6 for the Gauss 30-point and Kronrod 61-point rule.

Suggested value: KEY = 6.

Constraint: KEY = 1, 2, 3, 4, 5 or 6.

8: RESULT — DOUBLE PRECISION *Global Output*

On exit: the approximation to the integral I.

9: ABSERR — DOUBLE PRECISION *Global Output*

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

10: NFUN — INTEGER *Global Output*

On exit: the total number of evaluations of function $f(x)$ used in computing the integral.

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

12: LW — INTEGER *Global Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which D01AUFP 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 $\geq 4(p+1)$.

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

14: LIW — INTEGER *Global Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D01AUFP 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 $\geq p$.

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

The NAG Parallel Library provides a mechanism, via the routine Z02EAFP, to reduce the amount of parameter validation performed by this routine. For a full description refer to the Z02 Chapter Introduction.

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this argument (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 (or -9999 if reduced error checking is enabled) unless the routine detects an error (see Section 5).

5 Errors 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).

5.1 Full Error Checking Mode Only

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 = 4

On entry, KEY < 1,
or KEY > 6.

IFAIL = 5

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

5.2 Any Error Checking Mode

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. Probably 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 accuracy from being achieved on a sub-interval computed by a processor. 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.

6 Further Comments

6.1 Algorithmic Detail

D01AUFP is a modified version of the QUADPACK routine QAG (Piessens *et al.* [3]). It is an adaptive routine, offering a choice of six Gauss-Kronrod rules. A global acceptance criterion (as defined by Malcolm and Simpson [1]) is used. The local error estimation is described by Piessens *et al.* [3].

Because this routine is based on integration rules of high order, it is especially suitable for non-singular oscillating integrands.

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 QAG 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:

$$|I - \mathrm{RESULT}| \le tol$$

where

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

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

$$|I - \mathrm{RESULT}| \le \mathrm{ABSERR} \le tol.$$

7 References

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

[2] Piessens R (1973) An algorithm for automatic integration *Angew. Inf.* **15** 399–401

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

8 Example

The following integral is evaluated using D01AUFP:

$$I = \int_0^{10} \sin(10x) \sum_{k=1}^{10} \cos(1000k\cos(x)) \;\; dx.$$

8.1 Example Text

```
*     D01AUFP 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, KEY, MP, NFUN, NP
      LOGICAL          ROOT
*     .. Local Arrays ..
      DOUBLE PRECISION WORK(LW)
      INTEGER          IWORK(LIW)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         D01AUFP, F, Z01AAFP, Z01ABFP
*     .. Executable Statements ..

      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'D01AUFP Example Program Results'

      MP = 2
      NP = 2
      IFAIL = 0
*
*     Initialize Library Grid
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      A = 0.0D0
      B = 10.0D0
      KEY = 6
      EPSABS = 0.0D0
      EPSREL = 1.0D-6
      IFAIL = -1
```

```
*
*     Integrate the function F on Library Grid
*
      CALL D01AUFP(ICNTXT,F,A,B,EPSABS,EPSREL,KEY,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.3) 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, SIN
*     .. Executable Statements ..

      DO 40 I = 1, N
         SUM = 0.0D0
         DO 20 K = 1, 10
            SUM = SUM + COS(1000.D0*DBLE(K)*COS(X(I)))
   20    CONTINUE
```

```
         FV(I) = SIN(10.D0*X(I))*SUM
   40 CONTINUE
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
 D01AUFP Example Program Results

 A      - lower limit of integration  =      0.0000
 B      - upper limit of integration  =     10.0000
 EPSABS - absolute accuracy requested =    0.00E+00
 EPSREL - relative accuracy requested =    0.10E-05
 Number of tasks                      =           4

 Computed result is             =      0.0116
 Computed error is              =    0.12E-07
 No. of function evaluations is =      146705
```