D01AXFP
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

D01AXFP is an integrator which calculates an approximation to the sine or the cosine transform of a function g over $[a, b]$:

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

(for a user-specified value of ω).

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 D01AXFP(ICNTXT, G, A, B, OMEGA, KEY, EPSABS, EPSREL,
     1                   RESULT, ABSERR, NFUN, WORK, LW, IWORK, LIW,
     2                   IFAIL)
      DOUBLE PRECISION   A, B, OMEGA, EPSABS, EPSREL, RESULT, ABSERR,
     1                   WORK(LW)
      INTEGER            ICNTXT, NFUN, LW, IWORK(LIW), LIW, IFAIL
      CHARACTER*1        KEY
      EXTERNAL           G
```

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, OMEGA, KEY, EPSABS, EPSREL, 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: G — SUBROUTINE, supplied by the user. *External Procedure*

G must return the values of the integrand g at a set of points.

Its specification is:

```
SUBROUTINE       G(X, GV, N)
DOUBLE PRECISION X(N), GV(N)
INTEGER          N
```

1: X(N) — DOUBLE PRECISION array *Local Input*

On entry: the points at which the integrand g must be evaluated.

2: GV(N) — DOUBLE PRECISION array *Local Output*

On exit: GV(j) must contain the value of g at the point X(j), for $j = 1, 2, \ldots$,N.

3: N — INTEGER *Local Input*

On entry: the number of points at which the integrand is to be evaluated. The actual value of N depends on the Kronrod rule (N = 15) or the modified Clenshaw-Curtis procedure (N = 24) being used.

G must be declared as EXTERNAL in the (sub)program from which D01AXFP 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: OMEGA — DOUBLE PRECISION *Global Input*

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

6: KEY — CHARACTER*1 *Global Input*

On entry: indicates which integral is to be computed:

if KEY = 'C', $w(x) = \cos(\omega x)$;
if KEY = 'S', $w(x) = \sin(\omega x)$.

Constraint: KEY = 'C' or 'S'.

7: EPSABS — DOUBLE PRECISION *Global Input*

On entry: the absolute accuracy required. If EPSABS is negative, the absolute value is used. See Section 6.3.

8: EPSREL — DOUBLE PRECISION *Global Input*

On entry: the relative accuracy required. If EPSREL is negative, the absolute value is used. See Section 6.3.

9: RESULT — DOUBLE PRECISION *Global Output*

On exit: the approximation to the integral I.

10: ABSERR — DOUBLE PRECISION *Global Output*

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

11: NFUN — INTEGER *Global Output*

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

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

13: LW — INTEGER *Global Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which D01AXFP 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 $\geq 4(p+1)$.

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

15: LIW — INTEGER *Global Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which D01AXFP is called. The number of sub-intervals into which the interval of integration may be divided cannot exceed LIW$/(p+1)$ on each processor (see Section 3.1 for the definition of p).

Suggested value: LIW = LW/4.

Constraint: LIW $\geq p+1$.

16: 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 = 6

On entry, KEY $\neq$ 'C' and KEY $\neq$ 'S'.

IFAIL = 7

On entry, LW $< 4(p+1)$,
or LIW $< p+1$ (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. 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. 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 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 the 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 in any value of IFAIL = 1,2,. . .,4.

6 Further Comments

6.1 Algorithmic Detail

D01AXFP is a modified version of the QUADPACK routine QFOUR (Piessens *et al.* [3]). It is an adaptive routine, designed to integrate a function of the form $w(x)g(x)$, where $w(x)$ is either $\sin(\omega x)$ or $\cos(\omega x)$. If a sub-interval has length

$$L = |b_i - a_i| 2^{-l},$$

where a_i and b_i are the limits of a sub-interval on a processor (which depends on the subdivision strategy), then the integration over this sub-interval is performed by means of a modified Clenshaw-Curtis procedure (Piessens and Branders [2]) if $L\omega > 4$ and $l \le 20$. 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 the 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.* [3], incorporates a global acceptance criterion (as defined in Malcolm and Simpson [1]) together with the ε-algorithm by Wynn [4] to perform extrapolation. The local error estimation is described in Piessens *et al.* [3].

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 QFOUR 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. The more expensive the integrand the better the performance of D01AXFP scales with the number of processors.

6.3 Accuracy

The routine 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-requested absolute and relative accuracy. Moreover it returns the quantity ABSERR which, in normal circumstances satisfies

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

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

[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 D01AXFP:

$$\int_0^5 \sin(10x) \sum_{k=1}^{5} \cos(1000k \cos(x)) \;\; dx.$$

8.1 Example Text

```
*     D01AXFP 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=LW/4)
*     .. Local Scalars ..
      DOUBLE PRECISION A, ABSERR, B, EPSABS, EPSREL, OMEGA, RESULT
      INTEGER          ICNTXT, IFAIL, MP, NFUN, NP
      LOGICAL          ROOT
      CHARACTER        KEY
*     .. Local Arrays ..
      DOUBLE PRECISION WORK(LW)
      INTEGER          IWORK(LIW)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         D01AXFP, G, Z01AAFP, Z01ABFP
*     .. Executable Statements ..
```

```
      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'D01AXFP Example Program Results'

      MP = 2
      NP = 2
      IFAIL = 0
*
*    Initialize Library Grid
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      A = 0.0D0
      B = 5.0D0
      KEY = 'S'
      OMEGA = 10.D0
      EPSABS = 0.0D0
      EPSREL = 1.0D-6
      IFAIL = -1
*
*     Integrate function sin(wx)g(x) : Set Key ='S' for sin
*
      CALL D01AXFP(ICNTXT,G,A,B,OMEGA,KEY,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 G(X,GV,N)
*     .. Scalar Arguments ..
      INTEGER      N
*     .. Array Arguments ..
      DOUBLE PRECISION GV(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)*COS(X(I)))
   20    CONTINUE
         GV(I) = SUM
   40 CONTINUE
*
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
D01AXFP 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            =      0.0098
Computed error is             =    0.10E-07
No. of function evaluations is =      101635
```