D01GCFP
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

D01GCFP computes an approximation to a definite integral,

$$I = \int_{c_1}^{d_1} dx_1, \ldots, \int_{c_n}^{d_n} dx_n \quad f(x_1, x_2, \ldots, x_n)$$

in up to 20 dimensions using the Korobov–Conroy number theoretic method. The region of integration defined in the above integral is such that generally c_i and d_i may be functions of $x_1, x_2, \ldots, x_{i-1}$, for $i = 2, 3, \ldots, n$, with c_1 and d_1 constants.

2 Specification

```
    SUBROUTINE D01GCFP(ICNTXT, N, F, REGION, NPTS, VK, NRAND, ITRANS,
   1                   RESULT, ERROR, WORK, IFAIL)
    DOUBLE PRECISION   F, VK(N), RESULT, ERROR, WORK(NRAND)
    INTEGER            ICNTXT, N, NPTS, NRAND, ITRANS, IFAIL
    EXTERNAL           F, REGION
```

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:	N, NPTS, NRAND, ITRANS, IFAIL
Global output arguments:	VK, RESULT, ERROR, 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: N — INTEGER *Global Input*

On entry: the number of dimensions of the integral, n.

Constraint: $1 \leq \text{N} \leq 20$.

3: F — DOUBLE PRECISION FUNCTION, supplied by the user. *External Procedure*

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

Its specification is:

```
DOUBLE PRECISION FUNCTION F(N, X)
DOUBLE PRECISION           X(N)
INTEGER                    N
```

1: N — INTEGER *Global Input*

On entry: the number of dimensions of the integral, n.

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

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

F must be declared as EXTERNAL in the (sub)program from which D01GCFP is called. Arguments denoted as *Input* must **not** be changed by this procedure.

4: REGION — SUBROUTINE, supplied by the user. *External Procedure*

REGION must evaluate the limits of integration in any dimension.

Its specification is:

```
SUBROUTINE REGION(N, X, J, C, D)
DOUBLE PRECISION  X(N), C, D
INTEGER           N, J
```

1: N — INTEGER *Global Input*

On entry: the number of dimensions of the integral, n.

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

On entry: X(1),...,X($j-1$) contain the current values of the first $(j-1)$ variables, which may be used if necessary in calculating c_j and d_j.

3: J — INTEGER *Local Input*

On entry: the index j for which the limits of the range of integration are required.

4: C — DOUBLE PRECISION *Local Output*

On exit: the lower limit c_j of the range of x_j.

5: D — DOUBLE PRECISION *Local Output*

On exit: the upper limit d_j of the range of x_j.

REGION must be declared as EXTERNAL in the (sub)program from which D01GCFP is called. Arguments denoted as *Input* must **not** be changed by this procedure.

5: NPTS — INTEGER *Global Input*

On entry: the Korobov rule to be used. There are six preset rules depending on the value of NPTS:

NPTS = 1 for 2129-point rule,
NPTS = 2 for 5003-point rule,
NPTS = 3 for 10007-point rule,
NPTS = 4 for 20011-point rule,
NPTS = 5 for 40009-point rule,
NPTS = 6 for 80021-point rule.

Constraint: $1 \leq \text{NPTS} \leq 6$.

6: VK(N) — DOUBLE PRECISION array *Global Output*

On exit: VK contains the n optimal coefficients used by the preset rule.

7: NRAND — INTEGER *Global Input*

On entry: the number of random samples to be generated in the error estimation (generally a small value, say 3 to 5, is sufficient). The total number of integrand evaluations will be NRAND × r, where r is the number of points selected by NPTS.

Constraint: NRAND ≥ 1.

8: ITRANS — INTEGER *Global Input*

On entry: indicates whether the periodising transformation is to be used:

if ITRANS $= 0$, the transformation is to be used;
if ITRANS $\neq 0$, the transformation is to be suppresssed (to cover cases where the integrand may already be periodic or where the user desires to specify a particular transformation in the definition of F).

Suggested value: ITRANS $= 0$.

9: RESULT — DOUBLE PRECISION *Global Output*

On exit: an estimate of the value of the integral.

10: ERROR — DOUBLE PRECISION *Global Output*

On exit: the standard error as computed from NRAND sample values. If NRAND $= 1$, then ERROR contains zero.

11: WORK(NRAND) — DOUBLE PRECISION *Local Workspace*

12: 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).

5.2 Any Error Checking Mode

IFAIL = 1

On entry, N < 1 or N > 20.

IFAIL = 2

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

IFAIL = 3

On entry, NRAND < 1.

6 Further Comments

The exact values of RESULT and ERROR returned by D01GCFP will depend (within statistical limits) on the sequence of random numbers generated within the routine by calls to G05AAFP. To ensure that the results returned by D01GCFP in separate runs are identical, users should call G05ABFP immediately before calling D01GCFP with the same seed and generator numbers; to ensure that they are different, call G05ABFP with different seed and/or generator numbers.

6.1 Algorithmic Detail

This routine calculates an approximation to the integral,

$$I = \int_{c_1}^{d_1} dx_1, \ldots, \int_{c_n}^{d_n} dx_n \quad f(x_1, x_2, \ldots, x_n) \tag{1}$$

using the Korobov–Conroy number theoretic method (Conroy [1], Cranley and Patterson [2], Korobov [3]). The region of integration defined in (1) is such that generally c_i and d_i may be functions of $x_1, x_2, \ldots, x_{i-1}$, for $i = 2, 3, \ldots, n$, with c_1 and d_1 constants. First the integral is transformed to an integral over the n-cube $[0,1]^n$ by the change of variables

$$x_i = c_i + (d_i - c_i) y_i, \quad i = 1, 2, \ldots, n.$$

The method then uses as its basis the number theoretic formula for the n-cube, $[0,1]^n$:

$$\int_0^1 dy_1 \ldots \int_0^1 dy_n g(y_1, y_2, \ldots, y_n) = \frac{1}{q} \sum_{k=1}^{q} g\left(\left\{k\frac{a_1}{q}\right\}, \ldots, \left\{k\frac{a_n}{q}\right\}\right) - E \tag{2}$$

where $\{y\}$ denotes the fractional part of y, and $a_1, a_2, \ldots, a_n$ are the so-called optimal coefficients, E is the error and q is a prime integer. (It is strictly only necessary that q be relatively prime to all $a_1, a_2, \ldots, a_n$ and is in fact chosen to be even for some cases in Conroy [1] .) The method makes use of properties of the Fourier expansion of $g(y_1, y_2, \ldots, y_n)$ which is assumed to have some degree of periodicity. Depending on the choice of $a_1, a_2, \ldots, a_n$ the contributions from certain groups of Fourier coefficients are eliminated from the error, E. Korobov shows that $a_1, a_2, \ldots, a_n$ can be chosen so that the error satisfies

$$E \le CKq^{-\alpha} \ln^{\alpha\beta} q, \tag{3}$$

where α and C are real numbers depending on the convergence rate of the Fourier series, β is a constant depending on n and K is a constant depending on α and n. There are a number of procedures for calculating these optimal coefficients. Korobov imposes the constraint that

$$\begin{aligned} a_1 &= 1 \\ a_i &= a^{i-1} (\mathrm{mod}\ \ q) \end{aligned} \tag{4}$$

and gives a procedure for calculating the parameter. This routine provides built-in sets of optimal coefficients, corresponding to six different values of q.

In this routine the periodisation is achieved by the simple transformation

$$y_i = z_i^2 (3 - 2z_i), \qquad i = 1, 2, \ldots, n.$$

More sophisticated periodisation procedures are available but in practice the degree of periodisation does not appear to be a critical requirement of the method.

An easily calculable error estimate is not available apart from repetition with an increasing sequence of values of q which can yield erratic results. The difficulties have been studied by Cranley and Patterson [2] who have proposed a Monte–Carlo error estimate arising from converting (2) into a stochastic integration rule by the inclusion of a random origin shift which leaves the form of the error (3) unchanged; i.e., in (2), $\left\{k\frac{a_i}{q}\right\}$ is replaced by $\left\{\alpha_i + k\frac{a_i}{q}\right\}$, for $i = 1, 2, \ldots, n$, where each α_i is uniformly distributed over $[0, 1]$. Computing the integral for each of a sequence of random vectors α allows a 'standard error' to be estimated.

6.2 Parallelism Detail

The computation of the summation in (2) is divided into p independent tasks. Hence, this algorithm is highly scalable.

6.3 Accuracy

An estimate of the absolute standard error is given, on exit, by the value of ERROR.

7 References

[1] Conroy H (1967) Molecular Shroedinger equation VIII. A new method for evaluting multi-dimensional integrals *J. Chem. Phys.* **47** 5307–5318

[2] Cranley R and Patterson T N L (1976) Randomisation of number theoretic methods for mulitple integration *SIAM J. Numer. Anal.* **13** 904–914

[3] Korobov N M (1957) The approximate calculation of multiple integrals using number theoretic methods *Dokl. Acad. Nauk SSSR* **115** 1062–1065

[4] Korobov N M (1963) *Number Theoretic Methods in Approximate Analysis* Fizmatgiz, Moscow

8 Example

This example calculates the integral

$$\int_0^1 \int_0^1 \int_0^1 \int_0^1 \cos(0.5 + 2(x_1 + x_2 + x_3 + x_4) - 4)\; dx_1 \;\; dx_2 \;\; dx_3 \;\; dx_4,$$

using the 20011-point rule. Four samples are generated for the error estimation.

8.1 Example Text

```
*     D01GCFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          NDIM
      PARAMETER        (NDIM=4)
*     .. Local Scalars ..
      DOUBLE PRECISION ERROR, RESULT
      INTEGER          ICNTXT, IFAIL, ITRANS, MP, NP, NPTS, NRAND
      LOGICAL          ROOT
*     .. Local Arrays ..
      DOUBLE PRECISION VK(NDIM), WORK(10)
*     .. External Functions ..
      DOUBLE PRECISION F
      LOGICAL          Z01ACFP
      EXTERNAL         F, Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         D01GCFP, REGION, Z01AAFP, Z01ABFP
*     .. Executable Statements ..

      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'D01GCFP Example Program Results'

      MP = 2
      NP = 2
      IFAIL = 0
*
*     Initialise Library Grid
*
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      NRAND = 4
      NPTS = 4
      ITRANS = 0
      IFAIL = 0
*
*     Compute definite integral of F
*
      CALL D01GCFP(ICNTXT,NDIM,F,REGION,NPTS,VK,NRAND,ITRANS,RESULT,
     +             ERROR,WORK,IFAIL)
*
      IF (ROOT) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Number of tasks             =', MP*NP
         WRITE (NOUT,99999) 'Number of samples           =', NRAND
         WRITE (NOUT,99998) 'Computed result is          =', RESULT
         WRITE (NOUT,99997) 'Computed error is           =', ERROR
         WRITE (NOUT,99999) 'No. of function evaluations =',
     +     20011*NRAND

      END IF

      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'No',IFAIL)

      STOP

99999 FORMAT (1X,A,I12)
99998 FORMAT (1X,A,F12.5)
99997 FORMAT (1X,A,E12.2)
      END

      DOUBLE PRECISION FUNCTION F(NDIM,X)
*
*     This function evaluates the function to be
*     integrated
*
*     .. Scalar Arguments ..
      INTEGER                    NDIM
*     .. Array Arguments ..
      DOUBLE PRECISION           X(NDIM)
*     .. Local Scalars ..
```

```
      DOUBLE PRECISION            SUM
      INTEGER                     J
*     .. Intrinsic Functions ..
      INTRINSIC                   COS, DBLE
*     .. Executable Statements ..

      SUM = 0.0D0
      DO 20 J = 1, NDIM
         SUM = SUM + X(J)
   20 CONTINUE
      F = COS(0.5D0+2.0D0*SUM-DBLE(NDIM))

      RETURN
      END

      SUBROUTINE REGION(N,X,J,A,B)
*
*     This routine evaluates the limits of integration
*     in any given dimension.
*
*     .. Scalar Arguments ..
      DOUBLE PRECISION  A, B
      INTEGER           J, N
*     .. Array Arguments ..
      DOUBLE PRECISION  X(N)
*     .. Executable Statements ..

      A = 0.0D0
      B = 1.0D0
      RETURN
      END
```

8.2 Example Data

None.

8.3 Example Results

```
D01GCFP Example Program Results

Number of tasks            =          4
Number of samples          =          4
Computed result is         =    0.43999
Computed error is          =   0.14E-06
No. of function evaluations =      80044
```