

NAG Library Routine Document

D01GDF

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

D01GDF calculates an approximation to a definite integral in up to 20 dimensions, using the Korobov–Conroy number theoretic method. This routine is designed to be particularly efficient on vector processors.

2 Specification

```

SUBROUTINE D01GDF (NDIM, VECFUN, VECREG, NPTS, VK, NRAND, ITRANS, RES,
1                ERR, IFAIL)
    INTEGER          NDIM, NPTS, NRAND, ITRANS, IFAIL
    double precision VK(NDIM), RES, ERR
    EXTERNAL         VECFUN, VECREG

```

3 Description

D01GDF calculates an approximation to the integral

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

using the Korobov–Conroy number theoretic method (see Korobov (1957), Korobov (1963) and Conroy (1967)). The region of integration defined in (1) is such that generally c_i and d_i may be functions of $x_1, x_2, \dots, x_{i-1}$, for $i = 2, 3, \dots, n$, with c_1 and d_1 constants. The integral is first of all transformed to an integral over the n -cube $[0, 1]^n$ by the change of variables

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

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

$$\int_0^1 \cdots \int_0^1 g(x_1, \dots, x_n) dx_n \cdots dx_1 = \frac{1}{p} \sum_{k=1}^p g\left(\left\{k \frac{a_1}{p}\right\}, \dots, \left\{k \frac{a_n}{p}\right\}\right) - E \quad (2)$$

where $\{x\}$ denotes the fractional part of x , $a_1, \dots, a_n$ are the so-called optimal coefficients, E is the error, and p is a prime integer. (It is strictly only necessary that p be relatively prime to all $a_1, \dots, a_n$ and is in fact chosen to be even for some cases in Conroy (1967).) The method makes use of properties of the Fourier expansion of $g(x_1, \dots, x_n)$ which is assumed to have some degree of periodicity. Depending on the choice of $a_1, \dots, a_n$ the contributions from certain groups of Fourier coefficients are eliminated from the error, E . Korobov shows that $a_1, \dots, a_n$ can be chosen so that the error satisfies

$$E \leq CKp^{-\alpha} \ln^{\alpha\beta} p \quad (3)$$

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

$$a_1 = 1 \quad \text{and} \quad a_i = a^{i-1} \pmod{p} \quad (4)$$

and gives a procedure for calculating the parameter, a , to satisfy the optimal conditions.

In this routine the periodisation is achieved by the simple transformation

$$x_i = y_i^2(3 - 2y_i), \quad i = 1, 2, \dots, n.$$

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 p which can yield erratic results. The difficulties have been studied by Cranley and Patterson (1976) 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 the formula (2), $\left\{k\frac{a_i}{p}\right\}$ is replaced by $\left\{\alpha_i + k\frac{a_i}{p}\right\}$, for $i = 1, 2, \dots, n$, 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.

This routine provides built-in sets of optimal coefficients, corresponding to six different values of p . Alternatively, the optimal coefficients may be supplied by you. Routines D01GYF and D01GZF compute the optimal coefficients for the cases where p is a prime number or p is a product of two primes, respectively.

This routine is designed to be particularly efficient on vector processors, although it is very important that you also code VECFUN and VECREG efficiently.

4 References

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

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

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

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

5 Parameters

1: NDIM – INTEGER *Input*

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

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

2: VECFUN – SUBROUTINE, supplied by the user. *External Procedure*

VECFUN must evaluate the integrand at a specified set of points.

The specification of VECFUN is:

```
SUBROUTINE VECFUN(NDIM, X, FV, M)
  INTEGER          NDIM, M
  double precision X(M,NDIM), FV(M)
```

1: NDIM – INTEGER *Input*

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

2: X(M,NDIM) – **double precision** array *Input*

On entry: the co-ordinates of the m points at which the integrand must be evaluated. $X(i, j)$ contains the j th co-ordinate of the i th point.

3: FV(M) – **double precision** array *Output*

On exit: $FV(i)$ must contain the value of the integrand of the i th point, i.e., $FV(i) = f(X(i, 1), X(i, 2), \dots, X(i, \text{NDIM}))$, for $i = 1, 2, \dots, M$.

4:	M – INTEGER	<i>Input</i>
<i>On entry:</i> the number of points m at which the integrand is to be evaluated.		

VECFUN must be declared as EXTERNAL in the (sub)program from which D01GDF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

3: VECREG – SUBROUTINE, supplied by the user. *External Procedure*

VECREG must evaluate the limits of integration in any dimension for a set of points.

The specification of VECREG is: <pre style="margin: 0;">SUBROUTINE VECREG(NDIM, X, J, C, D, M) INTEGER NDIM, J, M double precision X(M,NDIM), C(M), D(M)</pre>		
1:	NDIM – INTEGER	<i>Input</i>
<i>On entry:</i> n , the number of dimensions of the integral.		
2:	X(M,NDIM) – double precision array	<i>Input</i>
<i>On entry:</i> for $i = 1, 2, \dots, m$, $X(i, 1), X(i, 2), \dots, X(i, j - 1)$ contain the current values of the first $(j - 1)$ co-ordinates of the i th point, which may be used if necessary in calculating the m values of c_j and d_j .		
3:	J – INTEGER	<i>Input</i>
<i>On entry:</i> the index j for which the limits of the range of integration are required.		
4:	C(M) – double precision array	<i>Output</i>
<i>On exit:</i> $C(i)$ must be set to the lower limit of the range for $X(i, j)$, for $i = 1, 2, \dots, m$.		
5:	D(M) – double precision array	<i>Output</i>
<i>On exit:</i> $D(i)$ must be set to the upper limit of the range for $X(i, j)$, for $i = 1, 2, \dots, m$.		
6:	M – INTEGER	<i>Input</i>
<i>On entry:</i> the number of points m at which the limits of integration must be specified.		

VECREG must be declared as EXTERNAL in the (sub)program from which D01GDF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

4: NPTS – INTEGER *Input*

On entry: the Korobov rule to be used. There are two alternatives depending on the value of NPTS.

(i) $1 \leq \text{NPTS} \leq 6$.

In this case one of six preset rules is chosen using 2129, 5003, 10007, 20011, 40009 or 80021 points depending on the respective value of NPTS being 1, 2, 3, 4, 5 or 6.

(ii) $\text{NPTS} > 6$.

NPTS is the number of actual points to be used with corresponding optimal coefficients supplied in the array VK.

Constraint: $\text{NPTS} \geq 1$.

5: VK(NDIM) – **double precision** array *Input/Output*

On entry: if $\text{NPTS} > 6$, VK must contain the n optimal coefficients (which may be calculated using D01GYF or D01GZF).

If $NPTS \leq 6$, VK need not be set.

On exit: if $NPTS > 6$, VK is unchanged.

If $NPTS \leq 6$, VK contains the n optimal coefficients used by the preset rule.

6: NRAND – INTEGER

Input

On entry: the number of random samples to be generated (generally a small value, say 3 to 5, is sufficient). The estimate, RES, of the value of the integral returned by the routine is then the average of NRAND calculations with different random origin shifts. If $NPTS > 6$, the total number of integrand evaluations will be $NRAND \times NPTS$. If $1 \leq NPTS \leq 6$, then the number of integrand evaluations will be $NRAND \times p$, where p is the number of points corresponding to the six preset rules. For reasons of efficiency, these values are calculated a number at a time in VECFUN.

Constraint: $NRAND \geq 1$.

7: ITRANS – INTEGER

Input

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

$ITRANS = 0$

The transformation is to be used.

$ITRANS \neq 0$

The transformation is to be suppressed (to cover cases where the integrand may already be periodic or where you want to specify a particular transformation in the definition of VECFUN).

Suggested value: $ITRANS = 0$.

8: RES – *double precision*

Output

On exit: the approximation to the integral I .

9: ERR – *double precision*

Output

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

10: 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, if you are not familiar with this parameter, the recommended value is 0 . **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).

Errors or warnings detected by the routine:

$IFAIL = 1$

On entry, $NDIM < 1$,
or $NDIM > 20$.

IFAIL = 2

On entry, NPTS < 1.

IFAIL = 3

On entry, NRAND < 1.

7 Accuracy

If NRAND > 1, an estimate of the absolute standard error is given by the value, on exit, of ERR.

8 Further Comments

D01GDF performs the same computation as D01GCF. However, the interface has been modified so that it can perform more efficiently on machines with vector processing capabilities. In particular, VECFUN and VECREG must calculate the integrand and limits of integration at a *set* of points. For some problems the amount of time spent in these two subroutines, which must be supplied by you, may account for a significant part of the total computation time. For this reason it is vital that you consider the possibilities for vectorization in the code supplied for these two subroutines.

The time taken will be approximately proportional to NRAND × *p*, where *p* is the number of points used, but may depend significantly on the efficiency of the code provided by you in VECFUN and VECREG.

The exact values of RES and ERR on return will depend (within statistical limits) on the sequence of random numbers generated within D01GDF by calls to G05SAF. Separate runs will produce identical answers.

9 Example

This example calculates the integral

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

9.1 Program Text

```
*      D01GDF Example Program Text
*      Mark 14 Release. NAG Copyright 1989.
*      .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          NDIM
      PARAMETER        (NDIM=4)
*      .. Local Scalars ..
      DOUBLE PRECISION ERR, RES
      INTEGER          IFAIL, ITRANS, NPTS, NRAND
*      .. Local Arrays ..
      DOUBLE PRECISION VK(NDIM)
*      .. External Subroutines ..
      EXTERNAL         D01GDF, VECFUN, VECREG
*      .. Executable Statements ..
      WRITE (NOUT,*) 'D01GDF Example Program Results'
      WRITE (NOUT,*)
      NPTS = 2
      ITRANS = 0
      NRAND = 4
      IFAIL = 1

*
      CALL D01GDF(NDIM,VECFUN,VECREG,NPTS,VK,NRAND,ITRANS,RES,ERR,IFAIL)
*
      IF (IFAIL.EQ.0) THEN
        WRITE (NOUT,99999) 'Result = ', RES, ', standard error = ', ERR
      ELSE
        WRITE (NOUT,99998) ' ** D01GDF returned with IFAIL = ', IFAIL
      END IF
```

```

*
99999 FORMAT (1X,A,F13.5,A,E10.2)
99998 FORMAT (1X,A,I5)
      END
*
      SUBROUTINE VECFUN(NDIM,X,FV,M)
*      .. Scalar Arguments ..
      INTEGER          M, NDIM
*      .. Array Arguments ..
      DOUBLE PRECISION FV(M), X(M,NDIM)
*      .. Local Scalars ..
      INTEGER          I, J
*      .. Intrinsic Functions ..
      INTRINSIC        COS, DBLE
*      .. Executable Statements ..
      DO 20 I = 1, M
          FV(I) = 0.0D0
20  CONTINUE
      DO 60 J = 1, NDIM
          DO 40 I = 1, M
              FV(I) = FV(I) + X(I,J)
40      CONTINUE
60  CONTINUE
      DO 80 I = 1, M
          FV(I) = COS(0.5D0+2.0D0*FV(I)-DBLE(NDIM))
80  CONTINUE
      RETURN
      END
*
      SUBROUTINE VECREG(NDIM,X,J,C,D,M)
*      .. Scalar Arguments ..
      INTEGER          J, M, NDIM
*      .. Array Arguments ..
      DOUBLE PRECISION C(M), D(M), X(M,NDIM)
*      .. Local Scalars ..
      INTEGER          I
*      .. Executable Statements ..
      DO 20 I = 1, M
          C(I) = 0.0D0
          D(I) = 1.0D0
20  CONTINUE
      RETURN
      END

```

9.2 Program Data

None.

9.3 Program Results

D01GDF Example Program Results

Result = 0.43999, standard error = 0.19E-05
