NAG Fortran Library Routine Document

D01FDF

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

D01FDF calculates an approximation to a definite integral in up to 30 dimensions, using the method of Sag and Szekeres. The region of integration is an n-sphere, or by built-in transformation via the unit n-cube, any product region.

2 Specification

```
      SUBROUTINE D01FDF(NDIM, FUNCTN, SIGMA, REGION, LIMIT, R0, U, RESULT,
     1                  NCALLS, IFAIL)
      INTEGER           NDIM, LIMIT, NCALLS, IFAIL
      real              FUNCTN, SIGMA, R0, U, RESULT
      EXTERNAL          FUNCTN, REGION
```

3 Description

This subroutine calculates an approximation to

$$\int_{n-\text{sphere of radius }\sigma} f(x_1, x_2, ..., x_n)\ dx_1\ dx_2\ \cdots\ dx_n \tag{1}$$

or, more generally,

$$\int_{c_1}^{d_1} dx_1 \cdots \int_{c_n}^{d_n} dx_n\ f(x_1, ..., x_n) \tag{2}$$

where each c_i and d_i may be functions of x_j $(j < i)$.

The routine uses the method of Sag and Szekeres (1964), which exploits a property of the shifted p-point trapezoidal rule, namely, that it integrates exactly all polynomials of degree $< p$ (Krylov (1962)). An attempt is made to induce periodicity in the integrand by making a parameterised transformation to the unit n-sphere. The Jacobian of the transformation and all its direct derivatives vanish rapidly towards the surface of the unit n-sphere, so that, except for functions which have strong singularities on the boundary, the resulting integrand will be pseudo-periodic. In addition, the variation in the integrand can be considerably reduced, causing the trapezoidal rule to perform well.

Integrals of the form (1) are transformed to the unit n-sphere by the change of variables:

$$x_i = y_i \frac{\sigma}{r} \tanh\left(\frac{ur}{1 - r^2}\right)$$

where $r^2 = \sum_{i=1}^{n} y_i^2$ and u is an adjustable parameter.

Integrals of the form (2) are first of all transformed to the n-cube $[-1, 1]^n$ by a linear change of variables

$$x_i = ((d_i + c_i) + (d_i - c_i)y_i)/2$$

and then to the unit sphere by a further change of variables

$$y_i = \tanh\left(\frac{uz_i}{1 - r}\right)$$

where $r^2 = \sum_{i=1}^{n} z_i^2$ and u is again an adjustable parameter.

The parameter u in these transformations determines how the transformed integrand is distributed between the origin and the surface of the unit n-sphere. A typical value of u is 1.5. For larger u, the integrand is concentrated toward the centre of the unit n-sphere, while for smaller u it is concentrated toward the perimeter.

In performing the integration over the unit n-sphere by the trapezoidal rule, a displaced equidistant grid of size h is constructed. The points of the mesh lie on concentric layers of radius

$$r_i = \frac{h}{4}\sqrt{n+8(i-1)}, \quad \text{for} \quad i = 1,2,3,\ldots.$$

The routine requires the user to specify an approximate maximum number of points to be used, and then computes the largest number of whole layers to be used, subject to an upper limit of 400 layers.

In practice, the rapidly-decreasing Jacobian makes it unnecessary to include the whole unit n-sphere and the integration region is limited by a user-specified cut-off radius $r_0 < 1$. The grid-spacing h is determined by r_0 and the number of layers to be used. A typical value of r_0 is 0.8.

Some experimentation may be required with the choice of r_0 (which determines how much of the unit n-sphere is included) and u (which determines how the transformed integrand is distributed between the origin and surface of the unit n-sphere), to obtain best results for particular families of integrals. This matter is discussed further in Section 8.

4 References

Sag T W and Szekeres G (1964) Numerical evaluation of high-dimensional integrals *Math. Comput.* **18** 245–253

Krylov V I (1962) *Approximate Calculation of Integrals* (trans A H Stroud) Macmillan

5 Parameters

1: NDIM – INTEGER *Input*

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

Constraint: $1 \leq \text{NDIM} \leq 30$.

2: FUNCTN – ***real*** FUNCTION, supplied by the user. *External Procedure*

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

Its specification is:

```
real FUNCTION FUNCTN(NDIM, X)
INTEGER             NDIM
real                X(NDIM)
```

1: NDIM – INTEGER *Input*

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

2: X(NDIM) – ***real*** array *Input*

On entry: the co-ordinates of the point at which the integrand must be evaluated.

FUNCTN must be declared as EXTERNAL in the (sub)program from which D01FDF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

3: SIGMA – ***real*** *Input*

On entry: SIGMA indicates the region of integration:

if SIGMA $\geq$ 0.0, the integration is carried out over the n-sphere of radius SIGMA, centred at the origin;

if SIGMA $<$ 0.0, the integration is carried out over the product region described by the user-specified subroutine REGION.

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

If SIGMA $<$ 0.0, REGION must evaluate the limits of integration in any dimension.

Its specification is:

```
SUBROUTINE REGION(NDIM, X, J, C, D)
INTEGER          NDIM, J
real             X(NDIM), C, D
```

1: NDIM – INTEGER *Input*

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

2: X(NDIM) – ***real*** array *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 *Input*

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

4: C – ***real*** *Output*

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

5: D – ***real*** *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 D01FDF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

If SIGMA $\geq$ 0.0, REGION is not called by D01FDF, but a dummy routine must be supplied (NAG Fortran Library auxiliary routine D01FDV may be used).

5: LIMIT – INTEGER *Input*

On entry: the approximate maximum number of integrand evaluations to be used.

Constraint: LIMIT $\geq$ 100.

6: R0 – ***real*** *Input*

On entry: the cut-off radius on the unit n-sphere, which may be regarded as an adjustable parameter of the method.

Suggested value: a typical value is R0 = 0.8. (See also Section 8.)

Constraint: 0.0 $<$ R0 $<$ 1.0.

7: U – ***real*** *Input*

On entry: U must specify an adjustable parameter of the transformation to the unit n-sphere.

Suggested value: a typical value is U = 1.5. (See also Section 8.)

Constraint: U > 0.0.

8: RESULT – ***real*** *Output*

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

9: NCALLS – INTEGER *Output*

On exit: the actual number of integrand evaluations used. (See also Section 8.)

10: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 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, for users 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 > 30.

IFAIL = 2

On entry, LIMIT < 100.

IFAIL = 3

On entry, R0 $\leq$ 0.0,
or R0 $\geq$ 1.0.

IFAIL = 4

On entry, U $\leq$ 0.0.

7 Accuracy

No error estimate is returned, but results may be verified by repeating with an increased value of LIMIT (provided that this causes an increase in the returned value of NCALLS).

8 Further Comments

The time taken by the routine will be approximately proportional to the returned value of NCALLS, which, except in the circumstances outlined in (b) below, will be close to the given value of LIMIT.

(a) Choice of R0 and U

If the chosen combination of r_0 and u is too large in relation to the machine accuracy it is possible that some of the points generated in the original region of integration may transform into points in the unit n-sphere which lie too close to the boundary surface to be distinguished from it to machine accuracy (despite the fact that $r_0 < 1$). To be specific, the combination of r_0 and u is too large if

$$\frac{ur_0}{1-r_0^2} > 0.3465(t-1), \quad \text{if SIGMA} \geq 0.0,$$

or

$$\frac{ur_0}{1-r_0} > 0.3465(t-1), \quad \text{if SIGMA} < 0.0,$$

where t is the number of bits in the mantissa of a ***real*** number.

The contribution of such points to the integral is neglected. This may be justified by appeal to the fact that the Jacobian of the transformation rapidly approaches zero towards the surface. Neglect of these points avoids the occurrence of overflow with integrands which are infinite on the boundary.

(b) Values of LIMIT and NCALLS

LIMIT is an approximate upper limit to the number of integrand evaluations, and may not be chosen less than 100. There are two circumstances when the returned value of NCALLS (the actual number of evaluations used) may be significantly less than LIMIT.

Firstly, as explained in Section 8(a), an unsuitably large combination of R0 and U may result in some of the points being unusable. Such points are not included in the returned value of NCALLS.

Secondly, no more than 400 layers will ever be used, no matter how high LIMIT is set. This places an effective upper limit on NCALLS as follows:

$n = 1$:	56
$n = 2$:	1252
$n = 3$:	23690
$n = 4$:	394528
$n = 5$:	5956906

9 Example

This example program calculates the integral

$$\int\int_s\int \frac{dx_1\,dx_2\,dx_3}{\sqrt{\sigma^2 - r^2}} = 22.2066$$

where s is the 3-sphere of radius σ, $r^2 = x_1^2 + x_2^2 + x_3^2$ and $\sigma = 1.5$. Both sphere-to-sphere and general product region transformations are used. For the former, we use $r_0 = 0.9$ and $u = 1.5$; for the latter, $r_0 = 0.8$ and $u = 1.5$.

9.1 Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     D01FDF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Local Scalars ..
```

```
      real              R0, RESULT, SIGMA, U
      INTEGER           IFAIL, LIMIT, NCALLS, NDIM
*     .. External Functions ..
      real              FUNCTN
      EXTERNAL          FUNCTN
*     .. External Subroutines ..
      EXTERNAL          D01FDF, D01FDV, REGION
*     .. Executable Statements ..
      WRITE (NOUT,*) 'D01FDF Example Program Results'
      NDIM = 3
      LIMIT = 8000
      U = 1.5e0
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Sphere-to-sphere transformation'
      SIGMA = 1.5e0
      R0 = 0.9e0
      IFAIL = 0
*
      CALL D01FDF(NDIM,FUNCTN,SIGMA,D01FDV,LIMIT,R0,U,RESULT,NCALLS,
     +            IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Estimated value of the integral =', RESULT
      WRITE (NOUT,99998) 'Number of integrand evaluations =', NCALLS
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Product region transformation'
      SIGMA = -1.0e0
      R0 = 0.8e0
      IFAIL = 0
*
      CALL D01FDF(NDIM,FUNCTN,SIGMA,REGION,LIMIT,R0,U,RESULT,NCALLS,
     +            IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Estimated value of the integral =', RESULT
      WRITE (NOUT,99998) 'Number of integrand evaluations =', NCALLS
      STOP
*
99999 FORMAT (1X,A,F9.3)
99998 FORMAT (1X,A,I4)
      END
*
      real FUNCTION FUNCTN(NDIM,X)
*     .. Scalar Arguments ..
      INTEGER               NDIM
*     .. Array Arguments ..
      real                  X(NDIM)
*     .. Local Scalars ..
      INTEGER               I
*     .. Intrinsic Functions ..
      INTRINSIC             ABS, SQRT
*     .. Executable Statements ..
      FUNCTN = 2.25e0
      DO 20 I = 1, NDIM
         FUNCTN = FUNCTN - X(I)*X(I)
   20 CONTINUE
      FUNCTN = 1.0e0/SQRT(ABS(FUNCTN))
      RETURN
      END
*
      SUBROUTINE REGION(NDIM,X,J,C,D)
*     .. Scalar Arguments ..
      real              C, D
      INTEGER           J, NDIM
*     .. Array Arguments ..
      real              X(NDIM)
*     .. Local Scalars ..
      real              SUM
      INTEGER           I, J1
*     .. Intrinsic Functions ..
      INTRINSIC         ABS, SQRT
```

```
*     .. Executable Statements ..
      C = -1.5e0
      D = 1.5e0
      IF (J.GT.1) THEN
         SUM = 2.25e0
         J1 = J - 1
         DO 20 I = 1, J1
            SUM = SUM - X(I)*X(I)
   20    CONTINUE
         D = SQRT(ABS(SUM))
         C = -D
      END IF
      RETURN
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 D01FDF Example Program Results

 Sphere-to-sphere transformation

 Estimated value of the integral =   22.168
 Number of integrand evaluations =8026

 Product region transformation

 Estimated value of the integral =   22.137
 Number of integrand evaluations =8026
```