NAG Library Routine Document

D01JAF

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

D01JAF attempts to evaluate an integral over an n-dimensional sphere ($n = 2$, 3, or 4), to a user-specified absolute or relative accuracy, by means of a modified Sag–Szekeres method. The routine can handle singularities on the surface or at the centre of the sphere, and returns an error estimate.

2 Specification

```
SUBROUTINE D01JAF (F, NDIM, RADIUS, EPSA, EPSR, METHOD, ICOORD, RESULT,    &
                   ESTERR, NEVALS, IFAIL)

INTEGER             NDIM, METHOD, ICOORD, NEVALS, IFAIL
REAL (KIND=nag_wp) F, RADIUS, EPSA, EPSR, RESULT, ESTERR
EXTERNAL            F
```

3 Description

D01JAF calculates an approximation to the n-dimensional integral

$$I = \int \cdots \int_S F(x_1, \ldots, x_n)\, dx_1 \cdots dx_n, \quad 2 \leq n \leq 4,$$

where S is the hypersphere

$$\sqrt{\left(x_1^2 + \cdots + x_n^2\right)} \leq \alpha < \infty$$

(the integrand function may also be defined in spherical coordinates). The algorithm is based on the Sag–Szekeres method (see Sag and Szekeres (1964)), applying the product trapezoidal formula after a suitable radial transformation. An improved transformation technique is developed: depending on the behaviour of the function and on the required accuracy, different transformations can be used, some of which are 'double exponential', as defined by Takahasi and Mori (1974). The resulting technique allows the routine to deal with integrand singularities on the surface or at the centre of the sphere. When the estimated error of the approximation with mesh size h is larger than the tolerated error, the trapezoidal formula with mesh size $h/2$ is calculated. A drawback of this method is the exponential growth of the number of function evaluations in the successive approximations (this number grows with a factor $\approx 2^n$). This introduces the restriction $n \leq 4$. Because the convergence rate of the successive approximations is normally better than linear, the error estimate is based on the linear extrapolation of the difference between the successive approximations (see Robinson and de Doncker (1981) and Roose and de Doncker (1981)). For further details of the algorithm, see Roose and de Doncker (1981).

4 References

Robinson I and de Doncker E (1981) Automatic computation of improper integrals over a bounded or unbounded planar region *Computing* **27** 89–284

Roose D and de Doncker E (1981) Automatic integration over a sphere *J. Comput. Appl. Math.* **7** 203–224

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

Takahasi H and Mori M (1974) *Double Exponential Formulas for Numerical Integration* **9** Publ. RIMS, Kyoto University 721–741

5 Arguments

1: F – REAL (KIND=nag_wp) FUNCTION, supplied by the user. *External Procedure*

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

The specification of F is:

```
FUNCTION F (NDIM, X)
REAL (KIND=nag_wp) F
INTEGER             NDIM
REAL (KIND=nag_wp) X(NDIM)
```

1: NDIM – INTEGER *Input*

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

2: X(NDIM) – REAL (KIND=nag_wp) array *Input*

On entry: the coordinates of the point at which the integrand f must be evaluated. These coordinates are given in Cartesian or spherical polar form according to the value of ICOORD.

F must either be a module subprogram USEd by, or declared as EXTERNAL in, the (sub) program from which D01JAF is called. Arguments denoted as *Input* must **not** be changed by this procedure.

See also Section 9.

2: NDIM – INTEGER *Input*

On entry: n, the dimension of the sphere.

Constraint: $2 \leq \text{NDIM} \leq 4$.

3: RADIUS – REAL (KIND=nag_wp) *Input*

On entry: α, the radius of the sphere.

Constraint: $\text{RADIUS} \geq 0.0$.

4: EPSA – REAL (KIND=nag_wp) *Input*

On entry: the requested absolute tolerance. If $\text{EPSA} < 0.0$, its absolute value is used. See Section 7.

5: EPSR – REAL (KIND=nag_wp) *Input*

On entry: the requested relative tolerance.

$\text{EPSR} < 0.0$
Its absolute value is used.

$\text{EPSR} < 10 \times$ (***machine precision***)
The latter value is used as EPSR by the routine. See Section 7.

6: METHOD – INTEGER *Input*

On entry: must specify the transformation to be used by the routine. The choice depends on the behaviour of the integrand and on the required accuracy.

For well-behaved functions and functions with mild singularities on the surface of the sphere only:

$\text{METHOD} = 1$
Low accuracy required.

METHOD = 2
High accuracy required.

For functions with severe singularities on the surface of the sphere only:

METHOD = 3
Low accuracy required.

METHOD = 4
High accuracy required.

(in this case ICOORD must be set to ICOORD = 2, and the function defined in special spherical coordinates).

For functions with a singularity at the centre of the sphere (and possibly with singularities on the surface as well):

METHOD = 5
Low accuracy required.

METHOD = 6
High accuracy required.

METHOD = 0 can be used as a default value and is equivalent to:

METHOD = 1 if EPSR $> 10^{-6}$, and

METHOD = 2 if EPSR $\leq 10^{-6}$.

The distinction between low and high required accuracies, as mentioned above, depends also on the behaviour of the function. Roughly one may assume the critical value of EPSA and EPSR to be 10^{-6}, but the critical value will be smaller for a well-behaved integrand and larger for an integrand with severe singularities.

Suggested value: METHOD = 0.

Constraint: METHOD = 0, 1, 2, 3, 4, 5 or 6.

If ICOORD = 2, METHOD = 3 or 4

7: ICOORD – INTEGER *Input*

On entry: must specify which kind of coordinates are used in F.

ICOORD = 0
Cartesian coordinates x_i, for $i = 1, 2, \ldots, n$.

ICOORD = 1
Spherical coordinates (see Section 9.2): X(1) $= \rho$; X(i) $= \theta_{i-1}$, for $i = 2, 3, \ldots, n$.

ICOORD = 2,
Special spherical polar coordinates (see Section 9.3), with the additional transformation $\rho = \alpha - \lambda$: X(1) $= \lambda = \alpha - \rho$; X($i$) $= \theta_{i-1}$, for $i = 2, 3, \ldots, n$.

Constraint: ICOORD = 0, 1 or 2.

If METHOD = 3 or 4, ICOORD = 2

8: RESULT – REAL (KIND=nag_wp) *Output*

On exit: the approximation to the integral I.

9: ESTERR – REAL (KIND=nag_wp) *Output*

On exit: an estimate of the modulus of the absolute error.

10: NEVALS – INTEGER *Output*

On exit: the number of function evaluations used.

11: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

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, because for this routine the values of the output arguments may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

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).

Note: D01JAF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL $= 1$

The required accuracy cannot be achieved within a limiting number of function evaluations (which is set by the routine).

IFAIL $= 2$

The required accuracy cannot be achieved because of round-off error.

IFAIL $= 3$

The required accuracy cannot be achieved because the maximum accuracy with respect to the machine constants X02AJF and X02AMF has been attained. If this maximum accuracy is rather low (compared with X02AJF), the cause of the problem is a severe singularity on the boundary or at the centre of the sphere. If METHOD $= 0$, 1 or 2, then setting METHOD $= 3$ or 4 may help.

IFAIL $= 4$

On entry, NDIM < 2 or NDIM > 4,
or RADIUS < 0.0,
or METHOD $\neq 0$, 1, 2, 3, 4, 5 or 6,
or ICOORD $\neq 0$, 1 or 2,
or ICOORD $= 2$ and METHOD $\neq 3$ or 4,
or METHOD $= 3$ or 4 and ICOORD $\neq 2$.

No calculations have been performed. RESULT and ESTERR are set to 0.0.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

You can specify an absolute and/or a relative tolerance, setting EPSA and EPSR. The routine attempts to calculate an approximation RESULT such that

$$|I - \text{RESULT}| \leq \max\{\text{EPSA}, \text{EPSR} \times |I|\}.$$

If $0 \leq \text{IFAIL} \leq 3$, ESTERR returns an estimate of, but not necessarily a bound for, $|I - \text{RESULT}|$.

8 Parallelism and Performance

D01JAF is not threaded in any implementation.

9 Further Comments

9.1 Timing

Timing depends on the integrand and the accuracy required.

9.2 Spherical Polar Coordinates

Cartesian coordinates are related to the spherical polar coordinates by:

$$\begin{aligned} x_1 &= \rho.\sin\theta_1 \cdots \sin\theta_{n-2}.\sin\theta_{n-1} \\ x_2 &= \rho.\sin\theta_1 \cdots \sin\theta_{n-2}.\cos\theta_{n-1} \\ x_3 &= \rho.\sin\theta_1 \cdots \cos\theta_{n-2} \\ &\vdots \\ x_n &= \rho.\cos\theta_1 \end{aligned}$$

where $0 < \theta_i < \pi$, for $i = 1, 2, \ldots, n-2$ and $0 < \theta_{n-1} < 2\pi$.

9.3 Machine Dependencies

As a consequence of the transformation technique, the severity of the singularities which can be handled by D01JAF depends on the precision and range of real numbers on the machine. METHOD $= 3$ or 4 must be used when the singularity on the surface is 'severe' in view of the requested accuracy and ***machine precision***. In practice one has to set METHOD $= 3$ or 4 if D01JAF terminates with IFAIL $= 3$ when called with METHOD $= 0$, 1 or 2.

When integrating a function with a severe singular behaviour on the surface of the sphere, the additional transformation $\rho = \alpha - \lambda$ helps to avoid the loss of significant figures due to round-off error in the calculation of the integration nodes which are very close to the surface. For these points, the value of λ can be computed more accurately than the value of ρ. Naturally, care must be taken that the function subprogram does not contain expressions of the form $\alpha - \lambda$, which could cause a large round-off error in the calculation of the integrand at the boundary of the sphere.

Care should be taken to avoid underflow and/or overflow problems in the function subprogram, because some of the integration nodes used by D01JAF may be very close to the surface or to the centre of the sphere.

Example:

suppose the function

$$f(\rho) = \left(1 - \rho^2\right)^{-0.7}$$

is to be integrated over the unit sphere, with METHOD = 3 or 4. Then ICOORD should be set to 2; the transformation $\rho = 1 - \lambda$ gives $f(\rho) = \left(2\lambda - \lambda^2\right)^{-0.7}$; and F could be coded thus:

```
F = 1.0
 A = X(1)
 IF (A.GT.0.0) F = 1.0/(A*(2.0-A))**0.7
 RETURN
```

Note that D01JAF ensures that $\lambda = \mathrm{X}(1) > \mathrm{X02AMF}$, but underflow could occur in the computation of λ^2.

10 Example

This example evaluates the integrals

$$\int \cdots \int_S \frac{1}{\sqrt{1-\rho^2}} dx_1 \cdots dx_n$$

where $\rho = \sqrt{\sum_{i=1}^{n} x_i^2}$, and S is the unit sphere of dimension $n = 2$ or 4.

The exact values (to 12 decimal places) are 6.28318530718 and 13.1594725348.

10.1 Program Text

```
!   D01JAF Example Program Text
!   Mark 26 Release. NAG Copyright 2016.

    Module d01jafe_mod

!     D01JAF Example Program Module:
!            Parameters and User-defined Routines

!     .. Use Statements ..
      Use nag_library, Only: nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Accessibility Statements ..
      Private
      Public                             :: f
!     .. Parameters ..
      Integer, Parameter, Public         :: nout = 6
    Contains
      Function f(ndim,x)

!       .. Function Return Value ..
        Real (Kind=nag_wp)               :: f
!       .. Scalar Arguments ..
        Integer, Intent (In)             :: ndim
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (In) :: x(ndim)
!       .. Local Scalars ..
        Real (Kind=nag_wp)               :: a, rho
!       .. Intrinsic Procedures ..
        Intrinsic                        :: sqrt
!       .. Executable Statements ..
        rho = x(1)
        a = (1.0E0_nag_wp-rho)*(1.0E0_nag_wp+rho)

        If (a/=0.0E0_nag_wp) Then
          f = 1.0E0_nag_wp/sqrt(a)
        Else
          f = 0.0E0_nag_wp
        End If

        Return
```

```
      End Function f
    End Module d01jafe_mod
    Program d01jafe

!     D01JAF Example Main Program

!     .. Use Statements ..
      Use nag_library, Only: d01jaf, nag_wp
      Use d01jafe_mod, Only: f, nout
!     .. Implicit None Statement ..
      Implicit None
!     .. Local Scalars ..
      Real (Kind=nag_wp)              :: epsa, epsr, esterr, radius, relest,  &
                                         result
      Integer                         :: icoord, ifail, method, ndim, nevals
!     .. Executable Statements ..
      Write (nout,*) 'D01JAF Example Program Results'

      radius = 1.0E0_nag_wp
      method = 0
      icoord = 1
      epsa = 0.0E0_nag_wp
      epsr = 0.5E-4_nag_wp

test: Do ndim = 2, 4, 2

        ifail = -1
        Call d01jaf(f,ndim,radius,epsa,epsr,method,icoord,result,esterr,       &
          nevals,ifail)

        Select Case (ifail)
        Case (:-1)
          Exit test
        Case (0:3)
          relest = esterr/result
          Write (nout,*)
          Write (nout,99999) 'Dimension of the sphere       =', ndim
          Write (nout,99998) 'Requested relative tolerance  =', epsr
          Write (nout,99997) 'Approximation to the integral =', result
          Write (nout,99999) 'No. of function evaluations   =', nevals
          Write (nout,99998) 'Estimated relative error      =', relest
        End Select

      End Do test

99999 Format (1X,A,I5)
99998 Format (1X,A,E9.2)
99997 Format (1X,A,F9.5)
    End Program d01jafe
```

10.2 Program Data

None.

10.3 Program Results

```
 D01JAF Example Program Results

 Dimension of the sphere       =    2
 Requested relative tolerance  = 0.50E-04
 Approximation to the integral =  6.28319
 No. of function evaluations   =  193
 Estimated relative error      = 0.31E-04
```

```
 Dimension of the sphere       =    4
 Requested relative tolerance  = 0.50E-04
 Approximation to the integral = 13.16004
 No. of function evaluations   = 2873
 Estimated relative error      = 0.40E-04
```