NAG Toolbox for MATLAB

d01fb

1 Purpose

d01fb computes an estimate of a multi-dimensional integral (from 1 to 20 dimensions), given the analytic form of the integrand and suitable Gaussian weights and abscissae.

2 Syntax

```
[result, ifail] = d01fb(nptvec, weight, abscis, fun, 'ndim', ndim, 'lwa', lwa)
```

3 Description

d01fb approximates a multi-dimensional integral by evaluating the summation

$$\sum_{i_1=1}^{l_1} w_{1,i_1} \sum_{i_2=1}^{l_2} w_{2,i_2} \cdots \sum_{i_n=1}^{l_n} w_{n,i_n} f\left(x_{1,i_1}, x_{2,i_2}, \ldots, x_{n,i_n}\right)$$

given the weights w_{j,i_j} and abscissae x_{j,i_j} for a multi-dimensional product integration rule (see Davis and Rabinowitz 1975). The number of dimensions may be anything from 1 to 20.

The weights and abscissae for each dimension must have been placed in successive segments of the arrays **weight** and **abscis**; for example, by calling d01bb or d01bc once for each dimension using a quadrature formula and number of abscissae appropriate to the range of each x_j and to the functional dependence of f on x_j.

If normal weights are used, the summation will approximate the integral

$$\int w_1(x_1) \int w_2(x_2) \cdots \int w_n(x_n) f(x_1, x_2, \ldots, x_n) dx_n \cdots dx_2 dx_1$$

where $w_j(x)$ is the weight function associated with the quadrature formula chosen for the jth dimension; while if adjusted weights are used, the summation will approximate the integral

$$\int \int \cdots \int f(x_1, x_2, \ldots, x_n) dx_n \cdots dx_2 dx_1.$$

You must supply a (sub)program to evaluate

$$f(x_1, x_2, \ldots, x_n)$$

at any values of $x_1, x_2, \ldots, x_n$ within the range of integration.

4 References

Davis P J and Rabinowitz P 1975 *Methods of Numerical Integration* Academic Press

5 Parameters

5.1 Compulsory Input Parameters

1: **nptvec**(**ndim**) – **int32 array**

nptvec(j) must specify the number of points in the jth dimension of the summation, for $j = 1, 2, \ldots, n$.

2: **weight**(**lwa**) – **double array**

Must contain in succession the weights for the various dimensions, i.e., **weight**(k) contains the ith weight for the jth dimension, with

$$k = \mathbf{nptvec}(1) + \mathbf{nptvec}(2) + \cdots + \mathbf{nptvec}(j-1) + i.$$

3: **abscis**(**lwa**) – **double array**

Must contain in succession the abscissae for the various dimensions, i.e., **abscis**(k) contains the ith abscissa for the jth dimension, with

$$k = \mathbf{nptvec}(1) + \mathbf{nptvec}(2) + \cdots + \mathbf{nptvec}(j-1) + i.$$

4: **fun – string containing name of m-file**

fun must return the value of the integrand f at a specified point.

Its specification is:

```
[result] = fun(ndim, x)
```

Input Parameters

1: **ndim – int32 scalar**

n, the number of dimensions of the integral.

2: **x**(**ndim**) – **double array**

The co-ordinates of the point at which the integrand f must be evaluated.

Output Parameters

1: **result – double scalar**

The result of the function.

5.2 Optional Input Parameters

1: **ndim – int32 scalar**

Default: The dimension of the array **nptvec**.

n, the number of dimensions of the integral.

Constraint: $1 \leq \mathbf{ndim} \leq 20$.

2: **lwa – int32 scalar**

Default: The dimension of the arrays **weight**, **abscis**. (An error is raised if these dimensions are not equal.)

Constraint: **lwa** $\geq \mathbf{nptvec}(1) + \mathbf{nptvec}(2) + \cdots + \mathbf{nptvec}(\mathbf{ndim})$.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **result – double scalar**

The result of the function.

2: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **ndim** < 1,
or **ndim** > 20,
or **lwa** $<$ **nptvec**$(1) +$ **nptvec**$(2) + \cdots +$ **nptvec**(**ndim**).

7 Accuracy

The accuracy of the computed multi-dimensional sum depends on the weights and the integrand values at the abscissae. If these numbers vary significantly in size and sign then considerable accuracy could be lost. If these numbers are all positive, then little accuracy will be lost in computing the sum.

8 Further Comments

The total time taken by d01fb will be proportional to

$$T \times \textbf{nptvec}(1) \times \textbf{nptvec}(2) \times \cdots \times \textbf{nptvec}(\textbf{ndim}),$$

where T is the time taken for one evaluation of user-supplied real function **fun**.

9 Example

d01fb_fun.m

```
function result = fun(ndim,x)
         result     =     (x(1)*x(2)*x(3))^6/(x(4)+2.0)^8*exp(-2.0*x(2)-
0.5*x(3)*x(3));
```

```
nptvec = [int32(4);
     int32(4);
     int32(4);
     int32(4)];
weight = [0.1739274225687269;
     0.326072577431273;
     0.326072577431273;
     0.1739274225687269;
     0.4163695619189446;
     1.024051219227148;
     1.815573152910759;
     3.243572542203831;
     1.753944171790685;
     1.49901614734385;
     1.49901614734385;
     1.753944171790685;
     0.6025497939505366;
     2.179205635439835;
     8.982199653682851;
     108.2360449169268];
abscis = [1.930568155797026;
     1.669990521792428;
     1.330009478207572;
     1.069431844202974;
     0.1612738448096961;
     0.8728805505791732;
```

```
     2.268310148460564;
     4.697535456150566;
     2.334414218338977;
     0.7419637843027259;
     -0.7419637843027259;
     -2.334414218338977;
     1.223836944463802;
     2.477675284083257;
     7.090647990761759;
     41.20783978069118];
[result, ifail] = d01fb(nptvec, weight, abscis, 'd01fb_fun')
```

```
result =
    0.2506
ifail =
           0
```