NAG Toolbox

nag_quad_2d_fin (d01da)

1 Purpose

nag_quad_2d_fin (d01da) attempts to evaluate a double integral to a specified absolute accuracy by repeated applications of the method described by Patterson (1968) and Patterson (1969).

2 Syntax

```
[ans, npts, ifail] = nag_quad_2d_fin(ya, yb, phi1, phi2, f, absacc)
[ans, npts, ifail] = d01da(ya, yb, phi1, phi2, f, absacc)
```

3 Description

nag_quad_2d_fin (d01da) attempts to evaluate a definite integral of the form

$$I = \int_a^b \int_{\phi_1(y)}^{\phi_2(y)} f(x,y)\,dx\,dy$$

where a and b are constants and $\phi_1(y)$ and $\phi_2(y)$ are functions of the variable y.

The integral is evaluated by expressing it as

$$I = \int_a^b F(y)\,dy, \qquad \text{where} \qquad F(y) = \int_{\phi_1(y)}^{\phi_2(y)} f(x,y)\,dx.$$

Both the outer integral I and the inner integrals $F(y)$ are evaluated by the method, described by Patterson (1968) and Patterson (1969), of the optimum addition of points to Gauss quadrature formulae.

This method uses a family of interlacing common point formulae. Beginning with the 3-point Gauss rule, formulae using 7, 15, 31, 63, 127 and finally 255 points are derived. Each new formula contains all the points of the earlier formulae so that no function evaluations are wasted. Each integral is evaluated by applying these formulae successively until two results are obtained which differ by less than the specified absolute accuracy.

4 References

Patterson T N L (1968) On some Gauss and Lobatto based integration formulae *Math. Comput.* **22** 877–881

Patterson T N L (1969) The optimum addition of points to quadrature formulae, errata *Math. Comput.* **23** 892

5 Parameters

5.1 Compulsory Input Parameters

1: **ya** – REAL (KIND=nag_wp)

a, the lower limit of the integral.

2: **yb** – REAL (KIND=nag_wp)

b, the upper limit of the integral. It is not necessary that $a < b$.

3: **phi1** – REAL (KIND=nag_wp) FUNCTION, supplied by the user.

phi1 must return the lower limit of the inner integral for a given value of y.

```
[result] = phi1(y)
```

Input Parameters

1: **y** – REAL (KIND=nag_wp)

The value of y for which the lower limit must be evaluated.

Output Parameters

1: **result**

The lower limit of the inner integral for the given value of y.

4: **phi2** – REAL (KIND=nag_wp) FUNCTION, supplied by the user.

phi2 must return the upper limit of the inner integral for a given value of y.

```
[result] = phi2(y)
```

Input Parameters

1: **y** – REAL (KIND=nag_wp)

The value of y for which the upper limit must be evaluated.

Output Parameters

1: **result**

The upper limit of the inner integral for the given value of y.

5: **f** – REAL (KIND=nag_wp) FUNCTION, supplied by the user.

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

```
[result] = f(x, y)
```

Input Parameters

1: **x** – REAL (KIND=nag_wp)
2: **y** – REAL (KIND=nag_wp)

The coordinates of the point (x, y) at which the integrand f must be evaluated.

Output Parameters

1: **result**

The value of $f(x)$ evaluated at **x**.

6: **absacc** – REAL (KIND=nag_wp)

The absolute accuracy requested.

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **ans** – REAL (KIND=nag_wp)

The estimated value of the integral.

2: **npts** – INTEGER

The total number of function evaluations.

3: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Note: nag_quad_2d_fin (d01da) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

ifail $= 1$ (*warning*)

This indicates that 255 points have been used in the outer integral and convergence has not been obtained. All the inner integrals have, however, converged. In this case **ans** may still contain an approximate estimate of the integral.

ifail $= 10 \times n$ (*warning*)

This indicates that the outer integral has converged but n inner integrals have failed to converge with the use of 255 points. In this case **ans** may still contain an approximate estimate of the integral, but its reliability will decrease as n increases.

ifail $= 10 \times n + 1$ (*warning*)

This indicates that both the outer integral and n of the inner integrals have not converged. **ans** may still contain an approximate estimate of the integral, but its reliability will decrease as n increases.

ifail $= -99$

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

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The absolute accuracy is specified by the variable **absacc**. If, on exit, **ifail** $= 0$ then the result is most likely correct to this accuracy. Even if **ifail** is nonzero on exit, it is still possible that the calculated result could differ from the true value by less than the given accuracy.

8 Further Comments

The time taken by nag_quad_2d_fin (d01da) depends upon the complexity of the integrand and the accuracy requested.

With Patterson's method accidental convergence may occasionally occur, when two estimates of an integral agree to within the requested accuracy, but both estimates differ considerably from the true result. This could occur in either the outer integral or in one or more of the inner integrals.

If it occurs in the outer integral then apparent convergence is likely to be obtained with considerably fewer integrand evaluations than may be expected. If it occurs in an inner integral, the incorrect value could make the function $F(y)$ appear to be badly behaved, in which case a very large number of pivots may be needed for the overall evaluation of the integral. Thus both unexpectedly small and unexpectedly large numbers of integrand evaluations should be considered as indicating possible trouble. If accidental convergence is suspected, the integral may be recomputed, requesting better accuracy; if the new request is more stringent than the degree of accidental agreement (which is of course unknown), improved results should be obtained. This is only possible when the accidental agreement is not better than machine accuracy. It should be noted that the function requests the same accuracy for the inner integrals as for the outer integral. In practice it has been found that in the vast majority of cases this has proved to be adequate for the overall result of the double integral to be accurate to within the specified value.

The function is not well-suited to non-smooth integrands, i.e., integrands having some kind of analytic discontinuity (such as a discontinuous or infinite partial derivative of some low-order) in, on the boundary of, or near, the region of integration. **Warning**: such singularities may be induced by incautiously presenting an apparently smooth interval over the positive quadrant of the unit circle, R

$$I = \int_R (x+y)\,dx\,dy.$$

This may be presented to nag_quad_2d_fin (d01da) as

$$I = \int_0^1 dy \int_0^{\sqrt{1-y^2}} (x+y)\,dx = \int_0^1 \left(\tfrac{1}{2}(1-y^2) + y\sqrt{1-y^2}\right) dy$$

but here the outer integral has an induced square-root singularity stemming from the way the region has been presented to nag_quad_2d_fin (d01da). This situation should be avoided by re-casting the problem. For the example given, the use of polar coordinates would avoid the difficulty:

$$I = \int_0^1 dr \int_0^{\frac{\pi}{2}} r^2(\cos\upsilon + \sin\upsilon)\,d\upsilon.$$

9 Example

This example evaluates the integral discussed in Section 9, presenting it to nag_quad_2d_fin (d01da) first as

$$\int_0^1 \int_0^{\sqrt{1-y^2}} (x+y)\,dx\,dy$$

and then as

$$\int_0^1 \int_0^{\frac{\pi}{2}} r^2(\cos\upsilon + \sin\upsilon)\,d\upsilon\,dr.$$

Note the difference in the number of function evaluations.

9.1 Program Text

```
function d01da_example

fprintf('d01da example results\n\n');

ya = 0;
yb = 1;
absacc = 1e-06;
[ans1, npts1, ifail] = d01da(ya, yb, @phi1, @phi2a, @fa, absacc);
[ans2, npts2, ifail] = d01da(ya, yb, @phi1, @phi2b, @fb, absacc);
```

```
fprintf('First formulation:\n');
fprintf('Integral                        = %9.4f\n', ans1);
fprintf('Number of function evaluations = %5d\n\n', npts1);
fprintf('Second formulation:\n');
fprintf('Integral                        = %9.4f\n', ans2);
fprintf('Number of function evaluations = %5d\n', npts2);

function result = phi1(y)
  result=0;

function result = phi2a(y)
  result = sqrt(1-y^2);

function result = fa(x,y)
  result=x+y;

function result = phi2b(y)
  result = pi/2;

function result = fb(x,y)
  result=(y^2)*(cos(x)+sin(x));
```

9.2 Program Results

```
     d01da example results

First formulation:
Integral                        =    0.6667
Number of function evaluations =   189

Second formulation:
Integral                        =    0.6667
Number of function evaluations =    89
```