

NAG Toolbox

nag_quad_1d_inf (d01am)

1 Purpose

nag_quad_1d_inf (d01am) calculates an approximation to the integral of a function $f(x)$ over an infinite or semi-infinite interval $[a, b]$:

$$I = \int_a^b f(x) dx.$$

2 Syntax

```
[result, abserr, w, iw, ifail] = nag_quad_1d_inf(f, bound, inf, epsabs, epsrel, 'lw', lw, 'liw', liw)
```

```
[result, abserr, w, iw, ifail] = d01am(f, bound, inf, epsabs, epsrel, 'lw', lw, 'liw', liw)
```

3 Description

nag_quad_1d_inf (d01am) is based on the QUADPACK routine QAGI (see Piessens *et al.* (1983)). The entire infinite integration range is first transformed to $[0, 1]$ using one of the identities:

$$\int_{-\infty}^a f(x) dx = \int_0^1 f\left(a - \frac{1-t}{t}\right) \frac{1}{t^2} dt$$

$$\int_a^{\infty} f(x) dx = \int_0^1 f\left(a + \frac{1-t}{t}\right) \frac{1}{t^2} dt$$

$$\int_{-\infty}^{\infty} f(x) dx = \int_0^{\infty} (f(x) + f(-x)) dx = \int_0^1 \left[f\left(\frac{1-t}{t}\right) + f\left(\frac{-1+t}{t}\right) \right] \frac{1}{t^2} dt$$

where a represents a finite integration limit. An adaptive procedure, based on the Gauss 7-point and Kronrod 15-point rules, is then employed on the transformed integral. The algorithm, described in de Doncker (1978), incorporates a global acceptance criterion (as defined by Malcolm and Simpson (1976)) together with the ϵ -algorithm (see Wynn (1956)) to perform extrapolation. The local error estimation is described in Piessens *et al.* (1983).

4 References

de Doncker E (1978) An adaptive extrapolation algorithm for automatic integration *ACM SIGNUM Newsl.* **13**(2) 12–18

Malcolm M A and Simpson R B (1976) Local versus global strategies for adaptive quadrature *ACM Trans. Math. Software* **1** 129–146

Piessens R, de Doncker–Kapenga E, Überhuber C and Kahaner D (1983) *QUADPACK, A Subroutine Package for Automatic Integration* Springer–Verlag

Wynn P (1956) On a device for computing the $e_m(S_n)$ transformation *Math. Tables Aids Comput.* **10** 91–96

5 Parameters

5.1 Compulsory Input Parameters

- 1: **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)
```

Input Parameters

- 1: **x** – REAL (KIND=nag_wp)
 The point at which the integrand f must be evaluated.

Output Parameters

- 1: **result**
 The value of the integrand at **x**

- 2: **bound** – REAL (KIND=nag_wp)
 The finite limit of the integration range (if present). **bound** is not used if the interval is doubly infinite.
- 3: **inf** – INTEGER
 Indicates the kind of integration range.
inf = 1
 The range is [**bound**, $+\infty$).
inf = -1
 The range is $(-\infty, \mathbf{bound}]$.
inf = 2
 The range is $(-\infty, +\infty)$.
Constraint: **inf** = -1, 1 or 2.
- 4: **epsabs** – REAL (KIND=nag_wp)
 The absolute accuracy required. If **epsabs** is negative, the absolute value is used. See Section 7.
- 5: **epsrel** – REAL (KIND=nag_wp)
 The relative accuracy required. If **epsrel** is negative, the absolute value is used. See Section 7.

5.2 Optional Input Parameters

- 1: **lw** – INTEGER
Suggested value: **lw** = 800 to 2000 is adequate for most problems.
Default: 800
 The dimension of the array **w**. the value of **lw** (together with that of **liw**) imposes a bound on the number of sub-intervals into which the interval of integration may be divided by the function. The number of sub-intervals cannot exceed **lw**/4. The more difficult the integrand, the larger **lw** should be.
Constraint: **lw** \geq 4.

2: **liw** – INTEGER

Suggested value: **liw** = **lw**/4.

Default: **liw**/4

The dimension of the array **iw**, the number of sub-intervals into which the interval of integration may be divided cannot exceed **liw**.

Constraint: **liw** \geq 1.

5.3 Output Parameters

1: **result** – REAL (KIND=nag_wp)

The approximation to the integral I .

2: **abserr** – REAL (KIND=nag_wp)

An estimate of the modulus of the absolute error, which should be an upper bound for $|I - \mathbf{result}|$.

3: **w(liw)** – REAL (KIND=nag_wp) array

Details of the computation see Section 9 for more information.

4: **iw(liw)** – INTEGER array

iw(1) contains the actual number of sub-intervals used. The rest of the array is used as workspace.

5: **ifail** – INTEGER

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

6 Error Indicators and Warnings

Note: nag_quad_1d_inf (d01am) 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*)

The maximum number of subdivisions allowed with the given workspace has been reached without the accuracy requirements being achieved. Look at the integrand in order to determine the integration difficulties. If the position of a local difficulty within the interval can be determined (e.g., a singularity of the integrand or its derivative, a peak, a discontinuity, etc.) you will probably gain from splitting up the interval at this point and calling nag_quad_1d_inf (d01am) on the infinite subrange and an appropriate integrator on the finite subrange. Alternatively, consider relaxing the accuracy requirements specified by **epsabs** and **epsrel**, or increasing the amount of workspace.

ifail = 2 (*warning*)

Round-off error prevents the requested tolerance from being achieved. Consider requesting less accuracy.

ifail = 3 (*warning*)

Extremely bad local integrand behaviour causes a very strong subdivision around one (or more) points of the interval. The same advice applies as in the case of **ifail** = 1.

ifail = 4 (*warning*)

The requested tolerance cannot be achieved because the extrapolation does not increase the accuracy satisfactorily; the returned result is the best which can be obtained. The same advice applies as in the case of **ifail** = 1.

ifail = 5 (*warning*)

The integral is probably divergent, or slowly convergent. Please note that divergence can occur with any nonzero value of **ifail**.

ifail = 6

On entry, **lw** < 4,
or **liw** < 1,
or **inf** ≠ -1, 1 or 2.

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

nag_quad_1d_inf (d01am) cannot guarantee, but in practice usually achieves, the following accuracy:

$$|I - \mathbf{result}| \leq tol,$$

where

$$tol = \max\{|\mathbf{epsabs}|, |\mathbf{epsrel}| \times |I|\},$$

and **epsabs** and **epsrel** are user-specified absolute and relative error tolerances. Moreover, it returns the quantity **abserr** which, in normal circumstances, satisfies

$$|I - \mathbf{result}| \leq \mathbf{abserr} \leq tol.$$

8 Further Comments

The time taken by nag_quad_1d_inf (d01am) depends on the integrand and the accuracy required.

If **ifail** ≠ 0 on exit, then you may wish to examine the contents of the array **w**, which contains the end points of the sub-intervals used by nag_quad_1d_inf (d01am) along with the integral contributions and error estimates over these sub-intervals.

Specifically, for $i = 1, 2, \dots, n$, let r_i denote the approximation to the value of the integral over the sub-interval $[a_i, b_i]$ in the partition of $[a, b]$ and e_i be the corresponding absolute error estimate. Then,

$\int_{a_i}^{b_i} f(x) dx \simeq r_i$ and $\mathbf{result} = \sum_{i=1}^n r_i$, unless nag_quad_1d_inf (d01am) terminates while testing for divergence of the integral (see Section 3.4.3 of Piessens *et al.* (1983)). In this case, **result** (and **abserr**) are taken to be the values returned from the extrapolation process. The value of n is returned in **iw**(1), and the values a_i , b_i , e_i and r_i are stored consecutively in the array **w**, that is:

$$a_i = \mathbf{w}(i),$$

$$b_i = \mathbf{w}(n + i),$$

$$e_i = \mathbf{w}(2n + i) \text{ and}$$

$$r_i = \mathbf{w}(3n + i).$$

Note: this information applies to the integral transformed to $[0, 1]$ as described in Section 3, not to the original integral.

9 Example

This example computes

$$\int_0^{\infty} \frac{1}{(x+1)\sqrt{x}} dx.$$

The exact answer is π .

9.1 Program Text

```
function d01am_example
fprintf('d01am example results\n\n');

bound = 0;
inf = nag_int(1);
epsabs = 0;
epsrel = 1.0e-4;
f = @(x) 1/(x+1)/sqrt(x);

[result, abserr, w, iw, ifail] = ...
    d01am( ...
        f, bound, inf, epsabs, epsrel);

fprintf('Result = %13.4f, Standard error = %10.2e\n', result, abserr);
```

9.2 Program Results

```
d01am example results
Result =          3.1416, Standard error =    2.65e-05
```
