nag_1d_quad_vals (d01gac)

1. Purpose

nag_1d_quad_vals (d01gac) integrates a function which is specified numerically at four or more points, over the whole of its specified interval using third-order finite-difference formulae with error estimates, according to a method due to Gill and Miller.

2. Specification

```
#include <nag.h>
#include <nagd01.h>

void nag_1d_quad_vals(Integer n, double x[], double y[], double *ans,
                double *er, NagError *fail)
```

3. Description

This routine evaluates the definite integral

$$I = \int_{x_1}^{x_n} y(x)\, dx$$

where the function y is specified at the n-points $x_1, x_2, \ldots, x_n$, which should be all distinct, and in either ascending or descending order. The integral between successive points is calculated by a four-point finite-difference formula centred on the interval concerned, except in the case of the first and last intervals, where four-point forward and backward difference formulae respectively are employed. If n is less than 4, the routine fails. An approximation to the truncation error is integrated and added to the result. It is also returned separately to give an estimate of the uncertainty in the result. The method is due to Gill and Miller (1972).

4. Parameters

n

Input: the number of points, n.
Constraint: $\mathbf{n} \geq 4$.

x[n]

Input: the values of the independent variable, i.e., $x_1, x_2, \ldots, x_n$.
Constraint: either $x_1 < x_2 < \ldots < x_n$ or $x_1 > x_2 > \ldots > x_n$.

y[n]

Input: the values of the dependent variable y_i at the points x_i, for $i = 1, 2, \ldots, n$.

ans

Output: the estimate of the integral.

er

Output: an estimate of the uncertainty in **ans**.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_INT_ARG_LT

On entry, **n** must not be less than 4: **n** = ⟨*value*⟩.

NE_NOT_STRICTLY_INCREASING

The sequence **x** is not strictly increasing: **x** [⟨*value*⟩] = ⟨*value*⟩, **x** [⟨*value*⟩] = ⟨*value*⟩.

NE_NOT_STRICTLY_DECREASING

The sequence **x** is not strictly decreasing: **x** [⟨*value*⟩] = ⟨*value*⟩, **x** [⟨*value*⟩] = ⟨*value*⟩.

NE_QUAD_FIRST_TWO_PTS_EQL
The sequence **x** has first two points equal: **x**[0] = ⟨*value*⟩, **x**[1] = ⟨*value*⟩.

No error is reported arising from the relative magnitudes of **ans** and **er** on return, due to the difficulty when the true answer is zero.

6. Further Comments

The time taken by the routine depends on the number of points supplied, n.

In their paper, Gill and Miller (1972) do not add the quantity **er** to **ans** before return. However, extensive tests have shown that a dramatic reduction in the error often results from such addition. In other cases, it does not make an improvement, but these tend to be cases of low accuracy in which the modified answer is not significantly inferior to the unmodified one. The user has the option of recovering the Gill-Miller answer by subtracting **er** from **ans** on return from the routine.

6.1. Accuracy

No accuracy level is specified by the user before calling the routine but on return |**er**| is an approximation to, but not necessarily a bound for, $|I-$**ans**$|$. If on exit **fail.code** is not equal to **NE_NOERROR**, both **ans** and **er** are returned as zero.

6.2. References

Gill P E and Miller G F (1972) An Algorithm for the Integration of Unequally Spaced Data *Comput. J.* **15** 80–83.

7. See Also

None.

8. Example

The following program evaluates the integral

$$\int_0^1 \frac{4}{1+x^2}dx = \pi$$

reading in the function values at 21 unequally-spaced points.

8.1. Program Text

```
/* nag_1d_quad_vals(d01gac) Example Program
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagd01.h>

#define NMAX 21
main()
{

  Integer i, n;
  static NagError fail;
  double x[NMAX], y[NMAX], error, ans;

  Vprintf("d01gac Example Program Results\n");
  Vscanf("%*[^\n]");    /* Skip heading in data file */
  Vscanf("%ld",&n);
  if (n <= NMAX)
    {
      for (i=0; i < n; ++i)
        Vscanf("%lf%lf", &x[i], &y[i]);
      d01gac(n, x, y, &ans, &error, &fail);
      if (fail.code == NE_NOERROR)
        {
          Vprintf("Integral = %7.4f\n", ans);
          Vprintf("Estimated error = %7.4f\n", error);
          exit(EXIT_SUCCESS);
        }
      else
        {
          Vprintf("%s\n", fail.message);
          exit(EXIT_FAILURE);
        }
    }
  else
    {
      Vprintf("More than NMAX data points\n");
      exit(EXIT_FAILURE);
    }
}                               /* main */
```

8.2. Program Data

```
d01gac Example Program Data
   21
   0.00   4.0000
   0.04   3.9936
   0.08   3.9746
   0.12   3.9432
   0.22   3.8153
   0.26   3.7467
   0.30   3.6697
   0.38   3.4943
   0.39   3.4719
   0.42   3.4002
   0.45   3.3264
   0.46   3.3014
   0.60   2.9412
   0.68   2.7352
   0.72   2.6344
   0.73   2.6094
   0.83   2.3684
   0.85   2.3222
   0.88   2.2543
   0.90   2.2099
   1.00   2.0000
```

8.3. Program Results

```
d01gac Example Program Results
Integral =  3.1414
Estimated error = -0.0001
```