NAG Library Function Document

nag_monotonic_intg (e01bhc)

1 Purpose
nag_monotonic_intg (e01bhc) evaluates the definite integral of a piecewise cubic Hermite interpolant over the interval \([a, b]\).

2 Specification

```c
#include <nag.h>
#include <nage01.h>
void nag_monotonic_intg (Integer n, const double x[], const double f[],
                        const double d[], double a, double b, double *integral, NagError *fail)
```

3 Description

nag_monotonic_intg (e01bhc) evaluates the definite integral of a piecewise cubic Hermite interpolant, as computed by nag_monotonic_interpolant (e01bec), over the interval \([a, b]\).

If either \(a\) or \(b\) lies outside the interval from \(x[0]\) to \(x[n-1]\), computation of the integral involves extrapolation and a warning is returned.

The function is derived from routine PCHIA in Fritsch (1982).

4 References


5 Arguments

1: \(n\) – Integer
   
   \textit{Input}
   
   On entry: \(n\) must be unchanged from the previous call of nag_monotonic_interpolant (e01bec).

2: \(x[n]\) – const double
   
   \textit{Input}

3: \(f[n]\) – const double
   
   \textit{Input}

4: \(d[n]\) – const double
   
   \textit{Input}
   
   On entry: \(x\), \(f\) and \(d\) must be unchanged from the previous call of nag_monotonic_interpolant (e01bec).

5: \(a\) – double
   
   \textit{Input}

6: \(b\) – double
   
   \textit{Input}
   
   On entry: the interval \([a, b]\) over which integration is to be performed.

7: \(\text{integral}\) – double *
   
   \textit{Output}
   
   On exit: the value of the definite integral of the interpolant over the interval \([a, b]\).

8: \(\text{fail}\) – NagError *
   
   \textit{Input/Output}
   
   The NAG error argument (see Section 3.6 in the Essential Introduction).
6 Error Indicators and Warnings

NE_INT_ARG_LT
On entry, \( n = \text{value} \).
Constraint: \( n \geq 2 \).

NE_NOT_MONOTONIC
On entry, \( x[r-1] \geq x[r] \) for \( r = \langle \text{value} \rangle : x[r-1] = \langle \text{value} \rangle \), \( x[r] = \langle \text{value} \rangle \).
The values of \( x[r] \), for \( r = 0, 1, \ldots, n-1 \), are not in strictly increasing order.

NW_INTERVAL_EXTRAPOLATE
On entry, limits \( a, b \) must not be outside interval \([x[0], x[n-1]]\), \( a = \langle \text{value} \rangle \), \( b = \langle \text{value} \rangle \),
\( x[0] = \langle \text{value} \rangle \), \( x[n-1] = \langle \text{value} \rangle \). Extrapolation was performed to compute the integral. The
value returned is therefore unreliable.

7 Accuracy
The computational error in the value returned for \texttt{integral} should be negligible in most practical
situations.

8 Parallelism and Performance
Not applicable.

9 Further Comments
The time taken by \texttt{nag_monotonic_intg (e01bhc)} is approximately proportional to the number of data
points included within the interval \([a, b]\).

10 Example
This example program reads in values of \( n, x, f \) and \( d \). It then reads in pairs of values for \( a \) and \( b \), and
evaluates the definite integral of the interpolant over the interval \((a, b)\) until end-of-file is reached.

10.1 Program Text

```c
/* nag_monotonic_intg (e01bhc) Example Program. *
 * Copyright 2014 Numerical Algorithms Group. *
 * Mark 2, 1991. *
 * Mark 8 revised, 2004. *
 */
#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nage01.h>

int main(void)
{
    Integer exit_status = 0, n, r;
    NagError fail;
    double a, b, *d = 0, *f = 0, integral, *x = 0;

    INIT_FAIL(fail);
    printf("nag_monotonic_intg (e01bhc) Example Program Results\n");
    #ifdef _WIN32
    scanf_s("%*[\n]"); /* Skip heading in data file */
    ```
```c
#ifelse
  scanf("%*[\n]"); /* Skip heading in data file */
#endif
#ifdef _WIN32
  scanf_s("%"NAG_IFMT"", &n);
#else
  scanf("%"NAG_IFMT"", &n);
#endif
if (n >= 2)
{
  if (!(d = NAG_ALLOC(n, double)) ||
      !(f = NAG_ALLOC(n, double)) ||
      !(x = NAG_ALLOC(n, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }
}
else
{
  printf("Invalid n.\n");
  exit_status = 1;
  return exit_status;
}
for (r = 0; r < n; r++)
#ifdef _WIN32
  scanf_s("%lf%lf%lf", &x[r], &f[r], &d[r]);
#else
  scanf("%lf%lf%lf", &x[r], &f[r], &d[r]);
#endif
printf(" Integral
  a b over (a,b)
/* Read a, b pairs until end of file and compute
 * definite integrals.
 * nag_monotonic_intg (e01bhc).
 * Evaluation of interpolant computed by
 * nag_monotonic_interpolant (e01bec), definite integral */
  nag_monotonic_intg(n, x, f, d, a, b, &integral, &fail);
  if (fail.code != NE_NOERROR)
  {
    printf("Error from nag_monotonic_intg (e01bhc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }
  printf("%13.4f %13.4f %13.4f
", a, b, integral);
}
END:
NAG_FREE(d);
NAG_FREE(f);
NAG_FREE(x);
return exit_status;
}
```
### 10.2 Program Data

**nag_monotonic_intg (e01bhc) Example Program Data**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>integral value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.99</td>
<td>20.0</td>
<td>10.7648</td>
</tr>
<tr>
<td>10.00</td>
<td>12.00</td>
<td>1.9622</td>
</tr>
<tr>
<td>12.00</td>
<td>10.00</td>
<td>-1.9622</td>
</tr>
<tr>
<td>15.00</td>
<td>15.00</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

### 10.3 Program Results

**nag_monotonic_intg (e01bhc) Example Program Results**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>Integral over (a,b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.99</td>
<td>20.0000</td>
<td>10.7648</td>
</tr>
<tr>
<td>10.00</td>
<td>12.0000</td>
<td>1.9622</td>
</tr>
<tr>
<td>12.00</td>
<td>10.0000</td>
<td>-1.9622</td>
</tr>
<tr>
<td>15.00</td>
<td>15.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>