

# NAG Library Function Document

## nag\_1d\_everett\_interp (e01abc)

### 1 Purpose

nag\_1d\_everett\_interp (e01abc) interpolates a function of one variable at a given point  $x$  from a table of function values evaluated at equidistant points, using Everett's formula.

### 2 Specification

```
#include <nag.h>
#include <nage01.h>

void nag_1d_everett_interp (Integer n, double p, double a[], double g[],
    NagError *fail)
```

### 3 Description

nag\_1d\_everett\_interp (e01abc) interpolates a function of one variable at a given point

$$x = x_0 + ph,$$

where  $-1 < p < 1$  and  $h$  is the interval of differencing, from a table of values  $x_m = x_0 + mh$  and  $y_m$  where  $m = -(n-1), -(n-2), \dots, -1, 0, 1, \dots, n$ . The formula used is that of Fr berg (1970), neglecting the remainder term:

$$y_p = \sum_{r=0}^{n-1} \left( \frac{1-p+r}{2r+1} \right) \delta^{2r} y_0 + \sum_{r=0}^{n-1} \left( \frac{p+r}{2r+1} \right) \delta^{2r} y_1.$$

The values of  $\delta^{2r} y_0$  and  $\delta^{2r} y_1$  are stored on exit from the function in addition to the interpolated function value  $y_p$ .

### 4 References

Fr berg C E (1970) *Introduction to Numerical Analysis* Addison–Wesley

### 5 Arguments

- 1: **n** – Integer *Input*  
*On entry:*  $n$ , half the number of points to be used in the interpolation.  
*Constraint:*  $n > 0$ .
- 2: **p** – double *Input*  
*On entry:* the point  $p$  at which the interpolated function value is required, i.e.,  $p = (x - x_0)/h$  with  $-1.0 < p < 1.0$ .  
*Constraint:*  $-1.0 < p < 1.0$ .
- 3: **a**[ $2 \times n$ ] – double *Input/Output*  
*On entry:* **a**[ $i-1$ ] must be set to the function value  $y_{i-n}$ , for  $i = 1, 2, \dots, 2n$ .  
*On exit:* the contents of **a** are unspecified.

- 4: **g**[ $2 \times n + 1$ ] – double *Output*  
*On exit:* the array contains  
 $y_0$  in **g**[0]  
 $y_1$  in **g**[1]  
 $\delta^{2r}y_0$  in **g**[ $2r$ ]  
 $\delta^{2r}y_1$  in **g**[ $2r + 1$ ], for  $r = 1, 2, \dots, n - 1$ .  
The interpolated function value  $y_p$  is stored in **g**[ $2n$ ].
- 5: **fail** – NagError \* *Input/Output*  
The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

### NE\_BAD\_PARAM

On entry, argument  $\langle value \rangle$  had an illegal value.

### NE\_INT

On entry, **n** =  $\langle value \rangle$ .

Constraint: **n** > 0.

### NE\_INTERNAL\_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please contact NAG for assistance.

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

See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

### NE\_NO\_LICENCE

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

See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

### NE\_REAL

On entry, **p** =  $\langle value \rangle$ .

Constraint: **p** < 1.0.

On entry, **p** =  $\langle value \rangle$ .

Constraint: **p** > -1.0.

## 7 Accuracy

In general, increasing  $n$  improves the accuracy of the result until full attainable accuracy is reached, after which it might deteriorate. If  $x$  lies in the central interval of the data (i.e.,  $0.0 \leq p < 1.0$ ), as is desirable, an upper bound on the contribution of the highest order differences (which is usually an upper bound on the error of the result) is given approximately in terms of the elements of the array **g** by  $a \times (|\mathbf{g}[2n - 2]| + |\mathbf{g}[2n - 1]|)$ , where  $a = 0.1, 0.02, 0.005, 0.001, 0.0002$  for  $n = 1, 2, 3, 4, 5$  respectively, thereafter decreasing roughly by a factor of 4 each time.

## 8 Parallelism and Performance

nag\_ld\_everett\_interp (e01abc) is not threaded in any implementation.

## 9 Further Comments

The computation time increases as the order of  $n$  increases.

## 10 Example

This example interpolates at the point  $x = 0.28$  from the function values

$$\begin{pmatrix} x_i & -1.00 & -0.50 & 0.00 & 0.50 & 1.00 & 1.50 \\ y_i & 0.00 & -0.53 & -1.00 & -0.46 & 2.00 & 11.09 \end{pmatrix}.$$

We take  $n = 3$  and  $p = 0.56$ .

### 10.1 Program Text

```
/* nag_ld_everett_interp (e01abc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nage01.h>

int main(void)
{
    /* Scalars */
    Integer exit_status = 0;
    Integer i, n, r;
    double p;
    NagError fail;
    /* Local Arrays */
    double *a = 0, *g = 0;

    INIT_FAIL(fail);

    printf("nag_ld_everett_interp (e01abc) Example Program Results\n");

    /* Skip heading in data file */
#ifdef _WIN32
    scanf_s("%*[^\\n] ");
#else
    scanf("%*[^\\n] ");
#endif
#ifdef _WIN32
    scanf_s("%" NAG_IFMT " ", &n);
#else
    scanf("%" NAG_IFMT " ", &n);
#endif
#ifdef _WIN32
    scanf_s("%lf", &p);
#else
    scanf("%lf", &p);
#endif
#ifdef _WIN32
    scanf_s("%*[^\\n] ");
#else
    scanf("%*[^\\n] ");
```

```

#endif

/* Allocate memory */
if (!(a = NAG_ALLOC((2 * n), double)) ||
    !(g = NAG_ALLOC((2 * n + 1), double)))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

for (i = 0; i < 2 * n; i++)
#ifdef _WIN32
    scanf_s("%lf", &a[i]);
#else
    scanf("%lf", &a[i]);
#endif
#ifdef _WIN32
    scanf_s("%*[^\\n] ");
#else
    scanf("%*[^\\n] ");
#endif

/* nag_ld_everett_interp (e01abc).
 * Interpolated values, Everett's formula, equally spaced data, one variable.
 */
nag_ld_everett_interp(n, p, a, g, &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_ld_everett_interp (e01abc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

printf("\n");
for (r = 0; r <= n - 1; r++) {
    printf("Central differences order %" NAG_IFMT " of Y0 = %12.5f\n",
           r, g[2 * r]);
    printf("                                Y1 = %12.5f\n", g[2 * r + 1]);
}
printf("\n");
printf("Function value at interpolation point = %12.5f\n", g[2 * n]);

END:
    NAG_FREE(a);
    NAG_FREE(g);

    return exit_status;
}

```

## 10.2 Program Data

nag\_ld\_everett\_interp (e01abc) Example Program Data

3	0.56					
0.00	-0.53	-1.00	-0.46	2.00	11.09	

## 10.3 Program Results

nag\_ld\_everett\_interp (e01abc) Example Program Results

Central differences order 0	of Y0 =	-1.00000
	Y1 =	-0.46000
Central differences order 1	of Y0 =	1.01000
	Y1 =	1.92000
Central differences order 2	of Y0 =	-0.04000
	Y1 =	3.80000
Function value at interpolation point =		-0.83591

---