NAG Library Function Document

nag_1d_cheb_eval (e02aec)

1 Purpose

nag_1d_cheb_eval (e02aec) evaluates a polynomial from its Chebyshev series representation.

2 Specification

```
#include <nag.h>
#include <nage02.h>
void nag_1d_cheb_eval (Integer nplus1, const double a[], double xcap,
     double *p, NagError *fail)
```

3 Description

nag_1d_cheb_eval (e02aec) evaluates the polynomial

$$\frac{1}{2}a_1T_0(\bar{x}) + a_2T_1(\bar{x}) + a_3T_2(\bar{x}) + \cdots + a_{n+1}T_n(\bar{x})$$

for any value of $\bar{x}$ satisfying $-1 \leq \bar{x} \leq 1$. Here $T_j(\bar{x})$ denotes the Chebyshev polynomial of the first kind of degree j with argument $\bar{x}$. The value of n is prescribed by you.

In practice, the variable $\bar{x}$ will usually have been obtained from an original variable x, where $x_{\text{min}} \leq x \leq x_{\text{max}}$ and

$$\bar{x} = \frac{((x - x_{\text{min}}) - (x_{\text{max}} - x))}{(x_{\text{max}} - x_{\text{min}})}$$

Note that this form of the transformation should be used computationally rather than the mathematical equivalent

$$\bar{x} = \frac{(2x - x_{\text{min}} - x_{\text{max}})}{(x_{\text{max}} - x_{\text{min}})}$$

since the former guarantees that the computed value of $\bar{x}$ differs from its true value by at most 4ϵ, where ϵ is the ***machine precision***, whereas the latter has no such guarantee.

The method employed is based upon the three-term recurrence relation due to Clenshaw (1955), with modifications to give greater numerical stability due to Reinsch and Gentleman (see Gentleman (1969)).

For further details of the algorithm and its use see Cox (1974), Cox and Hayes (1973).

4 References

Clenshaw C W (1955) A note on the summation of Chebyshev series *Math. Tables Aids Comput.* **9** 118–120

Cox M G (1974) A data-fitting package for the non-specialist user *Software for Numerical Mathematics* (ed D J Evans) Academic Press

Cox M G and Hayes J G (1973) Curve fitting: a guide and suite of algorithms for the non-specialist user *NPL Report NAC26* National Physical Laboratory

Gentleman W M (1969) An error analysis of Goertzel's (Watt's) method for computing Fourier coefficients *Comput. J.* **12** 160–165

5 Arguments

1: **nplus1** – Integer *Input*

On entry: the number $n+1$ of terms in the series (i.e., one greater than the degree of the polynomial).

Constraint: **nplus1** ≥ 1.

2: **a**[**nplus1**] – const double *Input*

On entry: **a**$[i-1]$ must be set to the value of the ith coefficient in the series, for $i = 1, 2, \ldots, n+1$.

3: **xcap** – double *Input*

On entry: $\bar{x}$, the argument at which the polynomial is to be evaluated. It should lie in the range -1 to $+1$, but a value just outside this range is permitted (see Section 9) to allow for possible rounding errors committed in the transformation from x to $\bar{x}$ discussed in Section 3. Provided the recommended form of the transformation is used, a successful exit is thus assured whenever the value of x lies in the range x_{min} to x_{max}.

4: **p** – double * *Output*

On exit: the value of the polynomial.

5: **fail** – NagError * *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, **nplus1** must not be less than 1: **nplus1** $= \langle value \rangle$.

NE_INVALID_XCAP

On entry, abs(**xcap**) $> 1.0 + 4\epsilon$, where ϵ is the ***machine precision***.
In this case the value of **p** is set arbitrarily to zero.

7 Accuracy

The rounding errors committed are such that the computed value of the polynomial is exact for a slightly perturbed set of coefficients $a_i + \delta a_i$. The ratio of the sum of the absolute values of the δa_i to the sum of the absolute values of the a_i is less than a small multiple of $(n+1)\times$ ***machine precision***.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by nag_1d_cheb_eval (e02aec) is approximately proportional to $n+1$.

It is expected that a common use of nag_1d_cheb_eval (e02aec) will be the evaluation of the polynomial approximations produced by nag_1d_cheb_fit (e02adc) and nag_1d_cheb_interp_fit (e02afc).

10 Example

Evaluate at 11 equally-spaced points in the interval $-1 \leq \bar{x} \leq 1$ the polynomial of degree 4 with Chebyshev coefficients, 2.0, 0.5, 0.25, 0.125, 0.0625.

The example program is written in a general form that will enable a polynomial of degree n in its Chebyshev series form to be evaluated at m equally-spaced points in the interval $-1 \leq \bar{x} \leq 1$. The program is self-starting in that any number of datasets can be supplied.

10.1 Program Text

```
/* nag_1d_cheb_eval (e02aec) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 5, 1998.
 * Mark 8 revised, 2004.
 *
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nage02.h>

int main(void)
{
  Integer  exit_status = 0, i, m, n, r;
  NagError fail;
  double   *a = 0, p, xcap;

  INIT_FAIL(fail);

  printf("nag_1d_cheb_eval (e02aec) Example Program Results \n");

  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif
#ifdef _WIN32
  while ((scanf_s("%"NAG_IFMT"", &m) != EOF))
#else
  while ((scanf("%"NAG_IFMT"", &m) != EOF))
#endif
    {
      if (m > 0)
        {
#ifdef _WIN32
          scanf_s("%"NAG_IFMT"", &n);
#else
          scanf("%"NAG_IFMT"", &n);
#endif
          if (n >= 0)
            {
              if (!(a = NAG_ALLOC(n+1, double)))
                {
                  printf("Allocation failure\n");
                  exit_status = -1;
                  goto END;
                }
            }
          else
            {
              printf("Invalid n.\n");
              exit_status = 1;
              return exit_status;
            }
```

```
          for (i = 0; i < n+1; ++i)
#ifdef _WIN32
            scanf_s("%lf", &a[i]);
#else
            scanf("%lf", &a[i]);
#endif
          printf("\n   R       Argument       Value of polynomial \n");
          for (r = 1; r <= m; ++r)
            {
              xcap = (double)(2*r - m - 1) / (double)(m - 1);
              /* nag_1d_cheb_eval (e02aec).
               * Evaluates the coefficients of a Chebyshev series
               * polynomial
               */
              nag_1d_cheb_eval(n+1, a, xcap, &p, &fail);
              if (fail.code != NE_NOERROR)
                {
                  printf("Error from nag_1d_cheb_eval (e02aec).\n%s\n",
                          fail.message);
                  exit_status = 1;
                  goto END;
                }
              printf(" %3"NAG_IFMT"%14.4f    %14.4f\n", r, xcap, p);
            }
        }
 END:
      NAG_FREE(a);
    }
  return exit_status;
}
```

10.2 Program Data

```
nag_1d_cheb_eval (e02aec) Example Program Data
  11
   4
     2.0000
     0.5000
     0.2500
     0.1250
     0.0625
```

10.3 Program Results

```
nag_1d_cheb_eval (e02aec) Example Program Results

   R       Argument       Value of polynomial
   1       -1.0000            0.6875
   2       -0.8000            0.6613
   3       -0.6000            0.6943
   4       -0.4000            0.7433
   5       -0.2000            0.7843
   6        0.0000            0.8125
   7        0.2000            0.8423
   8        0.4000            0.9073
   9        0.6000            1.0603
  10        0.8000            1.3733
  11        1.0000            1.9375
```