nag_1d_cheb_fit (e02adc)

1. Purpose

nag_1d_cheb_fit (e02adc) computes weighted least-squares polynomial approximations to an arbitrary set of data points.

2. Specification

```
#include <nag.h>
#include <nage02.h>

void nag_1d_cheb_fit(Integer m, Integer kplus1, Integer tda, double x[], double y[],
            double w[], double a[], double s[], NagError *fail)
```

3. Description

This routine determines least-squares polynomial approximations of degrees $0, 1, \ldots, k$ to the set of data points (x_r, y_r) with weights w_r, for $r = 1, 2, \ldots, m$.

The approximation of degree i has the property that it minimizes σ_i the sum of squares of the weighted residuals ϵ_r, where

$$\epsilon_r = w_r(y_r - f_r)$$

and f_r is the value of the polynomial of degree i at the rth data point.

Each polynomial is represented in Chebyshev-series form with normalised argument $\bar{x}$. This argument lies in the range -1 to $+1$ and is related to the original variable x by the linear transformation

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

Here $x_{\max}$ and $x_{\min}$ are respectively the largest and smallest values of x_r. The polynomial approximation of degree i is represented as

$$\tfrac{1}{2}a_{i+1,1}T_0(\bar{x}) + a_{i+1,2}T_1(\bar{x}) + a_{i+1,3}T_2(\bar{x}) + \ldots + a_{i+1,i+1}T_i(\bar{x}),$$

where $T_j(\bar{x})$ is the Chebyshev polynomial of the first kind of degree j with argument $\bar{x}$.

For $i = 0, 1, \ldots, k$, the routine produces the values of $a_{i+1,j+1}$, for $j = 0, 1, \ldots, i$, together with the value of the root mean square residual $s_i = \sqrt{\sigma_i/(m - i - 1)}$. In the case $m = i + 1$ the routine sets the value of s_i to zero.

The method employed is due to Forsythe (1957) and is based upon the generation of a set of polynomials orthogonal with respect to summation over the normalised data set. The extensions due to Clenshaw (1960) to represent these polynomials as well as the approximating polynomials in their Chebyshev-series forms are incorporated. The modifications suggested by Reinsch and Gentleman (Gentleman (1969)) to the method originally employed by Clenshaw for evaluating the orthogonal polynomials from their Chebyshev-series representations are used to give greater numerical stability.

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

Subsequent evaluation of the Chebyshev-series representations of the polynomial approximations should be carried out using nag_1d_cheb_eval (e02aec).

4. Parameters

m

Input: the number m of data points.

Constraint: $\mathbf{m} \geq \mathit{mdist} \geq 2$, where *mdist* is the number of distinct x values in the data.

kplus1
Input: $k+1$, where k is the maximum degree required.
Constraint: $0 <$ **kplus1** $\leq$ *mdist*, where *mdist* is the number of distinct x values in the data.

tda
Input: the second dimension of the array **a** as declared in the calling program.
Constraint: **tda** $\geq$ **kplus1**.

x[m]
Input: the values x_r of the independent variable, for $r = 1, 2, \ldots, m$.
Constraint: the values must be supplied in non-decreasing order with $\mathbf{x}[m-1] > \mathbf{x}[0]$.

y[m]
Input: the values y_r of the dependent variable, for $r = 1, 2, \ldots, m$.

w[m]
Input: the set of weights, w_r, for $r = 1, 2, \ldots, m$. For advice on the choice of weights, see the Chapter Introduction.
Constraint: $\mathbf{w}[r] > 0.0$, for $r = 0, 1, \ldots, \mathbf{m}-1$.

a[kplus1][tda]
Output: the coefficients of $\mathrm{T}_j(\bar{x})$ in the approximating polynomial of degree i. $\mathbf{a}[i][j]$ contains the coefficient $a_{i+1,j+1}$, for $i = 0, 1, \ldots, k$; $j = 0, 1, \ldots, i$.

s[kplus1]
Output: $\mathbf{s}[i]$ contains the root mean square residual s_i, for $i = 0, 1, \ldots, k$, as described in Section 3. For the interpretation of the values of the s_i and their use in selecting an appropriate degree, see the Chapter Introduction.

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, **kplus1** must not be less than 1: **kplus1** $= \langle$*value*$\rangle$.

NE_2_INT_ARG_LT
On entry, **tda** $= \langle$*value*$\rangle$ while **kplus1** $= \langle$*value*$\rangle$.
The parameters must satisfy **tda** $\geq$ **kplus1**.

NE_2_INT_ARG_GT
On entry, **kplus1** $= \langle$*value*$\rangle$ while the number of distinct x values, *mdist* $= \langle$*value*$\rangle$.
These parameters must satisfy **kplus1** $\leq$ *mdist*.

NE_WEIGHTS_NOT_POSITIVE
On entry, the weights are not strictly positive: $\mathbf{w}[\langle$*value*$\rangle] = \langle$*value*$\rangle$.

NE_NOT_NON_DECREASING
On entry, the sequence $\mathbf{x}[r]$, $r = 0, 1, \ldots, \mathbf{m}-1$ is not in non-decreasing order.

NE_NO_NORMALISATION
On entry, all the $\mathbf{x}[r]$ in the sequence $\mathbf{x}[r]$, $r = 0, 1, \ldots, \mathbf{m}-1$ are the same.

NE_ALLOC_FAIL
Memory allocation failed.

6. Further Comments

The time taken by the routine is approximately proportional to $m(k+1)(k+11)$.

The approximating polynomials may exhibit undesirable oscillations (particularly near the ends of the range) if the maximum degree k exceeds a critical value which depends on the number of data points m and their relative positions. As a rough guide, for equally spaced data, this critical value is about $2 \times \sqrt{m}$. For further details see page 60 of Hayes (1970).

6.1. Accuracy

No error analysis for the method has been published. Practical experience with the method, however, is generally extremely satisfactory.

6.2. References

Clenshaw C W (1960) Curve fitting with a digital computer *Comput. J.* **2** 170–173.

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 NAC 26* National Physical Laboratory.

Forsythe G E (1957) Generation and use of orthogonal polynomials for data fitting with a digital computer *J. Soc. Indust. Appl. Math.* **5** 74–88.

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

Hayes J G (ed.) (1970) Curve fitting by polynomials in one variable *Numerical Approximation to Functions and Data* Athlone Press, London.

7. See Also

nag_1d_cheb_eval (e02aec)

8. Example

Determine weighted least-squares polynomial approximations of degrees 0, 1, 2 and 3 to a set of 11 prescribed data points. For the approximation of degree 3, tabulate the data and the corresponding values of the approximating polynomial, together with the residual errors, and also the values of the approximating polynomial at points half-way between each pair of adjacent data points.

The example program supplied is written in a general form that will enable polynomial approximations of degrees $0, 1, \ldots, k$ to be obtained to m data points, with arbitrary positive weights, and the approximation of degree k to be tabulated. nag_1d_cheb_eval (e02aec) is used to evaluate the approximating polynomial. The program is self-starting in that any number of data sets can be supplied.

8.1. Program Text

```
/* nag_1d_cheb_fit(e02adc) Example Program
 *
 * Copyright 1997 Numerical Algorithms Group.
 *
 * Mark 5, 1997
 *
 */

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

main()
{
#define MMAX 200
#define KP1MAX 50
#define NROWS KP1MAX

  double d1;
  double xarg;
  double a[NROWS][KP1MAX];
  double s[KP1MAX], w[MMAX], x[MMAX], y[MMAX], xcapr;
  double  x1, ak[KP1MAX], xm, fit;

  Integer i, j, k, m, r;
  Integer iwght;
  Integer tdim;
```

```
  tdim = KP1MAX;
  Vprintf("e02adc Example Program Results \n");

  /*  Skip heading in data file */
  Vscanf("%*[^\n]");

  while ((scanf("%ld",&m)) != EOF)
    {
      if (m > 0 && m <= MMAX)
        {
          Vscanf("%ld",&k);
          Vscanf("%ld",&iwght);
          if (k + 1 <= KP1MAX)
            {
              for (r = 0; r < m; ++r)
                {
                  if (iwght != 1)
                    {
                      Vscanf("%lf",&x[r]);
                      Vscanf("%lf",&y[r]);
                      Vscanf("%lf",&w[r]);
                    }
                  else
                    {
                      Vscanf("%lf",&x[r]);
                      Vscanf("%lf",&y[r]);
                      w[r] = 1.0;
                    }
                }
              e02adc(m, k+1, tdim, x, y, w, (double *)a, s, NAGERR_DEFAULT);
              for (i = 0; i <= k; ++i)
                {
                  Vprintf("\n");
                  Vprintf(" %s%4ld%s%12.2e\n","Degree",i,"   R.M.S. residual =",s[i]);
                  Vprintf("\n   J  Chebyshev coeff A(J) \n");
                  for (j = 0; j < i+1; ++j)
                    Vprintf(" %3ld%15.4f\n",j+1,a[i][j]);
                }
              for (j = 0; j < k+1; ++j)
                ak[j] = a[k][j];
              x1 = x[0];
              xm = x[m-1];
              Vprintf("\n %s%4ld\n","Polynomial approximation and residuals for de-
gree",k);
              Vprintf("\n   R   Abscissa     Weight   Ordinate  Polynomial  Resid-
ual \n");
              for (r = 1; r <= m; ++r)
                {
                  xcapr = (x[r-1] - x1 - (xm - x[r-1])) / (xm - x1);
                  e02aec(k+1, ak, xcapr, &fit, NAGERR_DEFAULT);
                  d1 = fit - y[r-1];
                  Vprintf(" %3ld%11.4f%11.4f%11.4f%11.4f%11.2e\n",r,x[r-1],w[r-1],y[r-
1],fit,d1);
                  if (r < m)
                    {
                      xarg = (x[r-1]  + x[r]) * 0.5;
                      xcapr = (xarg - x1 - (xm - xarg)) / (xm - x1);
                      e02aec(k+1, ak, xcapr, &fit, NAGERR_DEFAULT);
                      Vprintf("    %11.4f                      %11.4f\n",xarg,fit);
                    }
                }
            }
        }
    }
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

```
e02adc Example Program Data
 11
  3   2
    1.00    10.40     1.00
    2.10     7.90     1.00
    3.10     4.70     1.00
    3.90     2.50     1.00
    4.90     1.20     1.00
    5.80     2.20     0.80
    6.50     5.10     0.80
    7.10     9.20     0.70
    7.80    16.10     0.50
    8.40    24.50     0.30
    9.00    35.30     0.20
```

8.3. Program Results

```
e02adc Example Program Results

 Degree   0   R.M.S. residual =    4.07e+00

   J  Chebyshev coeff A(J)
   1        12.1740

 Degree   1   R.M.S. residual =    4.28e+00

   J  Chebyshev coeff A(J)
   1        12.2954
   2         0.2740

 Degree   2   R.M.S. residual =    1.69e+00

   J  Chebyshev coeff A(J)
   1        20.7345
   2         6.2016
   3         8.1876

 Degree   3   R.M.S. residual =    6.82e-02

   J  Chebyshev coeff A(J)
   1        24.1429
   2         9.4065
   3        10.8400
   4         3.0589

 Polynomial approximation and residuals for degree   3

   R   Abscissa     Weight   Ordinate  Polynomial  Residual
   1     1.0000     1.0000    10.4000    10.4461   4.61e-02
         1.5500                           9.3106
   2     2.1000     1.0000     7.9000     7.7977  -1.02e-01
         2.6000                           6.2555
   3     3.1000     1.0000     4.7000     4.7025   2.52e-03
         3.5000                           3.5488
   4     3.9000     1.0000     2.5000     2.5533   5.33e-02
         4.4000                           1.6435
   5     4.9000     1.0000     1.2000     1.2390   3.90e-02
         5.3500                           1.4257
   6     5.8000     0.8000     2.2000     2.2425   4.25e-02
         6.1500                           3.3803
   7     6.5000     0.8000     5.1000     5.0116  -8.84e-02
         6.8000                           6.8400
   8     7.1000     0.7000     9.2000     9.0982  -1.02e-01
         7.4500                          12.3171
   9     7.8000     0.5000    16.1000    16.2123   1.12e-01
         8.1000                          20.1266
  10     8.4000     0.3000    24.5000    24.6048   1.05e-01
         8.7000                          29.6779
  11     9.0000     0.2000    35.3000    35.3769   7.69e-02
```