NAG Toolbox for MATLAB

e02ad

1 Purpose

e02ad computes weighted least-squares polynomial approximations to an arbitrary set of data points.

2 Syntax

```
[a, s, ifail] = e02ad(kplus1, x, y, w, 'm', m)
```

3 Description

e02ad 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 normalized argument $\bar{x}$. This argument lies in the range -1 to $+1$ and is related to the original variable x by the linear transformation

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

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

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

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 function 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 function sets the value of s_i to zero.

The method employed is due to Forsythe 1957 and is based on the generation of a set of polynomials orthogonal with respect to summation over the normalized 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 (see 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 and Cox and Hayes 1973.

Subsequent evaluation of the Chebyshev-series representations of the polynomial approximations should be carried out using e02ae.

4 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 NAC26* 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 *Numerical Approximation to Functions and Data* Athlone Press, London

5 Parameters

5.1 Compulsory Input Parameters

1: **kplus1 – int32 scalar**

$k+1$, where k is the maximum degree required.

Constraint: $0 < \mathbf{kplus1} \leq mdist$, where $mdist$ is the number of distinct x values in the data.

2: **x(m) – double array**

The values x_r of the independent variable, for $r = 1, 2, \ldots, m$.

Constraint: the values must be supplied in nondecreasing order with $\mathbf{x}(\mathbf{m}) > \mathbf{x}(1)$.

3: **y(m) – double array**

The values y_r of the dependent variable, for $r = 1, 2, \ldots, m$.

4: **w(m) – double array**

The set of weights, w_r, for $r = 1, 2, \ldots, m$. For advice on the choice of weights, see Section 2.1.2 in the E02 Chapter Introduction.

Constraint: $\mathbf{w}(r) > 0.0$, for $r = 1, 2, \ldots, m$.

5.2 Optional Input Parameters

1: **m – int32 scalar**

Default: The dimension of the arrays **x**, **y**, **w**. (An error is raised if these dimensions are not equal.)

the number m of data points.

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

5.3 Input Parameters Omitted from the MATLAB Interface

lda, work1, work2

5.4 Output Parameters

1: **a(lda,kplus1) – double array**

The coefficients of $T_j(\bar{x})$ in the approximating polynomial of degree i. $\mathbf{a}(i+1, j+1)$ contains the coefficient $a_{i+1,j+1}$, for $i = 0, 1, \ldots, k$ and $j = 0, 1, \ldots, i$.

2: **s(kplus1) – double array**

$\mathbf{s}(i+1)$ 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 Section 3.1 in the E02 Chapter Introduction.

3: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

The weights are not all strictly positive.

ifail $= 2$

The values of $\mathbf{x}(r)$, for $r = 1, 2, \ldots, \mathbf{m}$, are not in nondecreasing order.

ifail $= 3$

All $\mathbf{x}(r)$ have the same value: thus the normalization of $\mathbf{x}$ is not possible.

ifail $= 4$

On entry, **kplus1** < 1 (so the maximum degree required is negative)
or **kplus1** $> mdist$, where $mdist$ is the number of distinct x values in the data (so there cannot be a unique solution for degree $k = \mathbf{kplus1} - 1$).

ifail $= 5$

lda $<$ **kplus1**.

7 Accuracy

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

8 Further Comments

The time taken 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.

9 Example

```
kplus1 = int32(4);
x = [1;
     2.1;
     3.1;
     3.9;
     4.9;
     5.8;
     6.5;
     7.1;
     7.8;
     8.4;
     9];
y = [10.4;
     7.9;
     4.7;
     2.5;
     1.2;
     2.2;
     5.1;
     9.199999999999999;
```

```
     16.1;
     24.5;
     35.3];
w = [1;
     1;
     1;
     1;
     1;
     0.8;
     0.8;
     0.7;
     0.5;
     0.3;
     0.2];
[a, s, ifail] = e02ad(kplus1, x, y, w)
```

```
a =
   12.1740         0         0         0
   12.2954    0.2740         0         0
   20.7345    6.2016    8.1876         0
   24.1429    9.4065   10.8400    3.0589
s =
    4.0659
    4.2840
    1.6865
    0.0682
ifail =
           0
```