NAG Toolbox

nag_fit_1dspline_deriv (e02bc)

1 Purpose

nag_fit_1dspline_deriv (e02bc) evaluates a cubic spline and its first three derivatives from its B-spline representation.

2 Syntax

```
[s, ifail] = nag_fit_1dspline_deriv(lamda, c, x, left, 'ncap7', ncap7)
[s, ifail] = e02bc(lamda, c, x, left, 'ncap7', ncap7)
```

3 Description

nag_fit_1dspline_deriv (e02bc) evaluates the cubic spline $s(x)$ and its first three derivatives at a prescribed argument x. It is assumed that $s(x)$ is represented in terms of its B-spline coefficients c_i, for $i = 1, 2, \ldots, \bar{n} + 3$ and (augmented) ordered knot set λ_i, for $i = 1, 2, \ldots, \bar{n} + 7$, (see nag_fit_1dspline_knots (e02ba)), i.e.,

$$s(x) = \sum_{i=1}^{q} c_i N_i(x).$$

Here $q = \bar{n} + 3$, $\bar{n}$ is the number of intervals of the spline and $N_i(x)$ denotes the normalized B-spline of degree 3 (order 4) defined upon the knots $\lambda_i, \lambda_{i+1}, \ldots, \lambda_{i+4}$. The prescribed argument x must satisfy

$$\lambda_4 \leq x \leq \lambda_{\bar{n}+4}.$$

At a simple knot λ_i (i.e., one satisfying $\lambda_{i-1} < \lambda_i < \lambda_{i+1}$), the third derivative of the spline is in general discontinuous. At a multiple knot (i.e., two or more knots with the same value), lower derivatives, and even the spline itself, may be discontinuous. Specifically, at a point $x = u$ where (exactly) r knots coincide (such a point is termed a knot of multiplicity r), the values of the derivatives of order $4 - j$, for $j = 1, 2, \ldots, r$, are in general discontinuous. (Here $1 \leq r \leq 4$; $r > 4$ is not meaningful.) You must specify whether the value at such a point is required to be the left- or right-hand derivative.

The method employed is based upon:

(i) carrying out a binary search for the knot interval containing the argument x (see Cox (1978)),

(ii) evaluating the nonzero B-splines of orders 1, 2, 3 and 4 by recurrence (see Cox (1972) and Cox (1978)),

(iii) computing all derivatives of the B-splines of order 4 by applying a second recurrence to these computed B-spline values (see de Boor (1972)),

(iv) multiplying the fourth-order B-spline values and their derivative by the appropriate B-spline coefficients, and summing, to yield the values of $s(x)$ and its derivatives.

nag_fit_1dspline_deriv (e02bc) can be used to compute the values and derivatives of cubic spline fits and interpolants produced by nag_fit_1dspline_knots (e02ba).

If only values and not derivatives are required, nag_fit_1dspline_eval (e02bb) may be used instead of nag_fit_1dspline_deriv (e02bc), which takes about 50% longer than nag_fit_1dspline_eval (e02bb).

4 References

Cox M G (1972) The numerical evaluation of B-splines *J. Inst. Math. Appl.* **10** 134–149

Cox M G (1978) The numerical evaluation of a spline from its B-spline representation *J. Inst. Math. Appl.* **21** 135–143

de Boor C (1972) On calculating with B-splines *J. Approx. Theory* **6** 50–62

5 Parameters

5.1 Compulsory Input Parameters

1: **lamda**(**ncap7**) – REAL (KIND=nag_wp) array

lamda(j) must be set to the value of the jth member of the complete set of knots, λ_j, for $j = 1, 2, \ldots, \bar{n} + 7$.

Constraint: the **lamda**(j) must be in nondecreasing order with **lamda**(**ncap7** $- 3$) $>$ **lamda**(4).

2: **c**(**ncap7**) – REAL (KIND=nag_wp) array

The coefficient c_i of the B-spline $N_i(x)$, for $i = 1, 2, \ldots, \bar{n} + 3$. The remaining elements of the array are not referenced.

3: **x** – REAL (KIND=nag_wp)

The argument x at which the cubic spline and its derivatives are to be evaluated.

Constraint: **lamda**(4) $\leq$ **x** $\leq$ **lamda**(**ncap7** $- 3$).

4: **left** – INTEGER

Specifies whether left- or right-hand values of the spline and its derivatives are to be computed (see Section 3). Left- or right-hand values are formed according to whether **left** is equal or not equal to 1.

If x does not coincide with a knot, the value of **left** is immaterial.

If $x =$ **lamda**(4), right-hand values are computed.

If $x =$ **lamda**(**ncap7** $- 3$), left-hand values are formed, regardless of the value of **left**.

5.2 Optional Input Parameters

1: **ncap7** – INTEGER

Default: the dimension of the arrays **lamda**, **c**. (An error is raised if these dimensions are not equal.)

$\bar{n} + 7$, where $\bar{n}$ is the number of intervals of the spline (which is one greater than the number of interior knots, i.e., the knots strictly within the range λ_4 to $\lambda_{\bar{n}+4}$ over which the spline is defined).

Constraint: **ncap7** ≥ 8.

5.3 Output Parameters

1: **s**(**4**) – REAL (KIND=nag_wp) array

s(j) contains the value of the $(j-1)$th derivative of the spline at the argument x, for $j = 1, 2, 3, 4$. Note that **s**(1) contains the value of the spline.

2: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

ncap7 < 8, i.e., the number of intervals is not positive.

ifail $= 2$

Either **lamda**$(4) \geq$ **lamda**(**ncap7** $- 3$), i.e., the range over which $s(x)$ is defined is null or negative in length, or **x** is an invalid argument, i.e., **x** $<$ **lamda**(4) or **x** $>$ **lamda**(**ncap7** $- 3$).

ifail $= -99$

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

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The computed value of $s(x)$ has negligible error in most practical situations. Specifically, this value has an **absolute** error bounded in modulus by $18 \times c_{\max} \times$ ***machine precision***, where $c_{\max}$ is the largest in modulus of c_j, c_{j+1}, c_{j+2} and c_{j+3}, and j is an integer such that $\lambda_{j+3} \leq x \leq \lambda_{j+4}$. If c_j, c_{j+1}, c_{j+2} and c_{j+3} are all of the same sign, then the computed value of $s(x)$ has **relative** error bounded by $20 \times$ ***machine precision***. For full details see Cox (1978).

No complete error analysis is available for the computation of the derivatives of $s(x)$. However, for most practical purposes the absolute errors in the computed derivatives should be small.

8 Further Comments

The time taken is approximately linear in $\log(\bar{n} + 7)$.

Note: the function does not test all the conditions on the knots given in the description of **lamda** in Section 5, since to do this would result in a computation time approximately linear in $\bar{n} + 7$ instead of $\log(\bar{n} + 7)$. All the conditions are tested in nag_fit_1dspline_knots (e02ba), however.

9 Example

Compute, at the 7 arguments $x = 0$, 1, 2, 3, 4, 5, 6, the left- and right-hand values and first 3 derivatives of the cubic spline defined over the interval $0 \leq x \leq 6$ having the 6 interior knots $x = 1$, 3, 3, 3, 4, 4, the 8 additional knots 0, 0, 0, 0, 6, 6, 6, 6, and the 10 B-spline coefficients 10, 12, 13, 15, 22, 26, 24, 18, 14, 12.

The input data items (using the notation of Section 5) comprise the following values in the order indicated:

$\bar{n}$	m
lamda(j),	for $j = 1, 2, \ldots,$ **ncap7**
c(j),	for $j = 1, 2, \ldots,$ **ncap7** $- 4$
x(i),	for $i = 1, 2, \ldots, m$

This example program is written in a general form that will enable the values and derivatives of a cubic spline having an arbitrary number of knots to be evaluated at a set of arbitrary points. Any number of

datasets may be supplied. The only changes required to the program relate to the dimensions of the arrays **lamda** and **c**.

9.1 Program Text

```
function e02bc_example

fprintf('e02bc example results\n\n');

knots = [ 1  3  3  3  4  4];
ncap = size(knots,2) + 1;
ncap7 = ncap + 7;

lamda = zeros(ncap7,1);
lamda(5:ncap+3) = knots;
lamda(ncap+4:ncap7) = 6;

% B-spline coefficients
c = zeros(ncap7,1);
c(1:ncap+3) = [10  12  13  15  22  26  24  18  14  12];

% Evaluate spline at values in lamda range
left  = nag_int(1);
right = nag_int(2);
k = 0;
for x = 0:0.2:6;
  k = k+1;
  [sl(:,k), ifail] = e02bc( ...
                            lamda, c, x, left);
  [sr(:,k), ifail] = e02bc( ...
                            lamda, c, x, right);
end
x = 0:0.2:6;
fprintf('Left hand values and derivatives\n');
fprintf('%5s%12s%12s%11s%11s\n', 'x', 'spline', '1st deriv', ...
        '2nd deriv', '3rd deriv');
sol = [ x; sl];
fprintf('%7.2f%11.4f%11.4f%11.4f%11.4f\n',sol(:,1:5:end));

fprintf('\nRight hand values and derivatives\n');
fprintf('%5s%12s%12s%11s%11s\n', 'x', 'spline', '1st deriv', ...
        '2nd deriv', '3rd deriv');
sol = [ x; sr];
fprintf('%7.2f%11.4f%11.4f%11.4f%11.4f\n', sol(:,1:5:end));

fig1 = figure;
plot(x,sl(1,:),x,sl(2,:),x,sl(3,:),x,sl(4,:));
xlabel('x');
title('Evaluation of Left-hand cubic spline and derivatives');
legend('cubic spline', '1st derivative', '2nd derivative', ...
       '3rd derivative', 'Location', 'NorthWest');
fig2 = figure;
plot(x,sr(1,:),x,sr(2,:),x,sr(3,:),x,sr(4,:));
xlabel('x');
title('Evaluation of Right-hand cubic spline and derivatives');
legend('cubic spline', '1st derivative', '2nd derivative', ...
       '3rd derivative', 'Location', 'NorthWest');
```

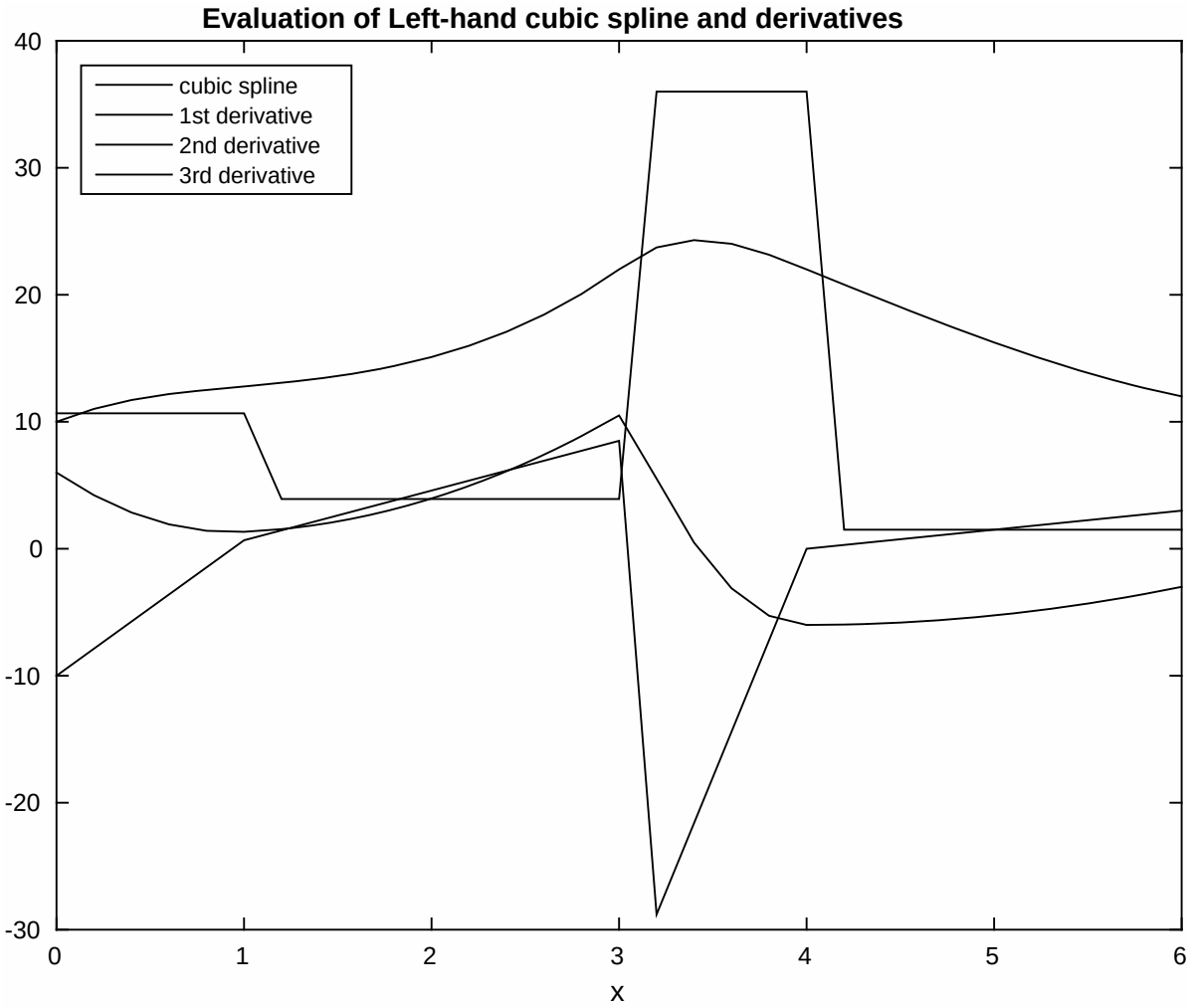
9.2 Program Results

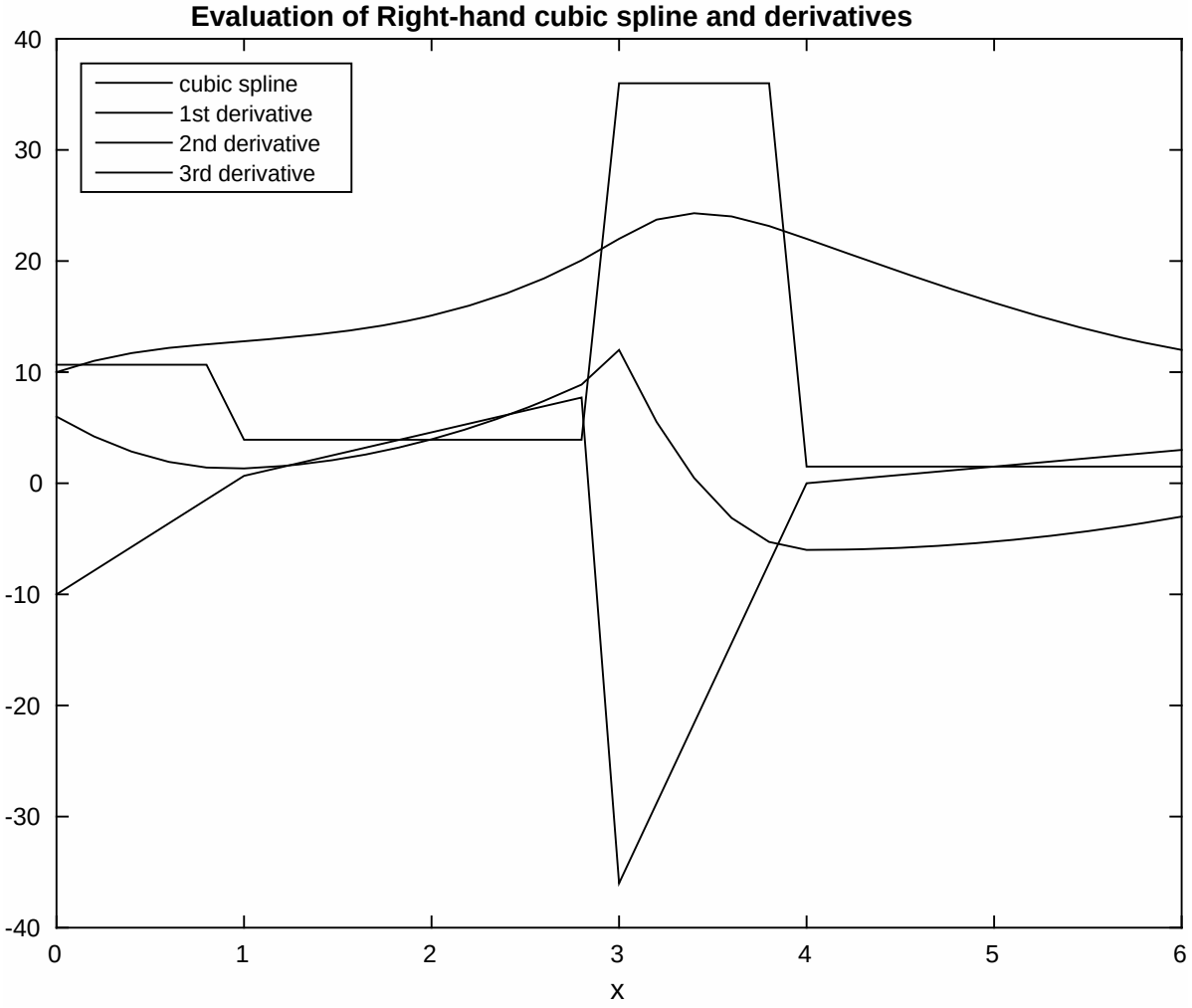
```
e02bc example results

Left hand values and derivatives
    x      spline   1st deriv  2nd deriv  3rd deriv
   0.00    10.0000     6.0000   -10.0000    10.6667
   1.00    12.7778     1.3333     0.6667    10.6667
   2.00    15.0972     3.9583     4.5833     3.9167
   3.00    22.0000    10.5000     8.5000     3.9167
   4.00    22.0000    -6.0000     0.0000    36.0000
```

```
   5.00    16.2500    -5.2500     1.5000     1.5000
   6.00    12.0000    -3.0000     3.0000     1.5000

Right hand values and derivatives
    x      spline   1st deriv  2nd deriv  3rd deriv
   0.00    10.0000     6.0000   -10.0000    10.6667
   1.00    12.7778     1.3333     0.6667     3.9167
   2.00    15.0972     3.9583     4.5833     3.9167
   3.00    22.0000    12.0000   -36.0000    36.0000
   4.00    22.0000    -6.0000     0.0000     1.5000
   5.00    16.2500    -5.2500     1.5000     1.5000
   6.00    12.0000    -3.0000     3.0000     1.5000
```




Evaluation of Right-hand cubic spline and derivatives
cubic spline
1st derivative
2nd derivative
3rd derivative
40
30
20
10
0
-10
-20
-30
-40
0
1
2
3
4
5
6
x