1 SUMMARY

Given the n points $x_1 \le x_2 \le ... \le x_n$ and a value of x, this subroutine **computes the values of the k B-splines of degree $k-1$**

$$M_{k,i}(x) \quad 1 \le i \le n-k,$$

which have knots at the points $x_i, x_{i+1}, ..., x_{i+k}$ and which cover the point x, and it also computes the values of the corresponding integrals

$$\int_{x_i}^{x} M_{k,i}(\xi)\, d\xi, \qquad 1 \le i \le n-k.$$

The subroutine uses the recurrence relation due to De Boor, 'On Calculating with B-splines', J. Approx. Theory, **6**, (1972), to evaluate the B-spline. The integrals are evaluated using the method due to Gaffney, AERE CSS.10, (1974).

ATTRIBUTES — Version: 1.0.0. **Types:** `TG04A`, `TG04AD`. **Original date:** December 1976. **Origin:** P.W.Gaffney*, Harwell.

2 HOW TO USE THE PACKAGE

2.1 The argument list and calling sequence

The single precision version:

```
CALL TG04A(N,X,K,NORM,XVALUE,JINT,V,VINT)
```

The double precision version:

```
CALL TG04AD(N,X,K,NORM,XVALUE,JINT,V,VINT)
```

`N` is an `INTEGER` variable which must be set by the user to n the number of points x_i, i=1,2,...,n. This argument is not altered by the subroutine. **Restriction:** $n > 0$.

`X` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length at least n, which must be set by the user to the values x_i, i=1,2,...,n. This argument is not altered by the subroutine. **Restriction:** these must be ordered so that $x_1 \le x_2 \le ... \le x_n$ and $x_i < x_{i+k}$, i=1,2,...,$n-k$.

`K` is an `INTEGER` variable which must be set by the user to the value k, where k is the order of the B-spline $M_{k,i}(x)$. This argument is not altered by the subroutine. **Restriction:** the value of k must be in the range $1 \le k \le n-1$.

`NORM` is an `INTEGER` variable which must be set by the user to the value 1 or 2 to indicate which normalization is to be used when computing the B-spline values.

When `NORM=1` the subroutine evaluates B-splines $M_{k,i}(x)$, $1 \le i \le n-k$, which are normalized so that

$$\int_{x_i}^{x_{i+k}} M_{k,i}(\xi)\, d\xi = k^{-1}$$

When `NORM=2` the subroutine evaluates B-splines $N_{k,i}(x)$, $1 \le i \le n-k$, which are normalized so that $\sum_i N_{k,i}(x) = 1$. **N.B.**

$$N_{k,i}(x) = (x_{i+k} - x_i) M_{k,i}(x)$$

If the value of `NORM` on entry to the subroutine is different from 1 or 2, a diagnostic message is printed, `NORM` is set to 1 and the calculation is continued.

`XVALUE` is a `REAL` (`DOUBLE PRECISION` in the D version) variable which is set by the user to the value of the argument x at which the B-splines are to be evaluated. This argument is not altered by the subroutine.

`JINT` is an `INTEGER` variable which is set by the subroutine to indicate the knot interval containing x and is set in the following way.

(i) For $x < x_n$, `JINT` is the unique integer such that

$$x_{JINT} \le x < x_{JINT+1}.$$

(ii) For $x = x_n$, `JINT` is set to n if $x_{n-1} < x_n$, otherwise `JINT` is set so that

$$x_{JINT} < x \le x_{JINT+1} = x_n.$$

(iii) If $x < x_1$ then `JINT` is set to 1, and if $x > x_n$ then `JINT` is set to n.

`V` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length at least k which is set by the subroutine so that on return `V(J), J=1,K` contains the values of $M_{k,JINT-k+j}(x)$, j=1,2,...,k, in the case when $1 \le JINT-k+j \le n-k$, and is set to zero otherwise. All the values of the other B-splines at the point x are zero.

`VINT` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length at least k which is set by the subroutine to the values of the integrals.

When x is in the interval $x_1 \le x \le x_n$, `VINT(J), J=1,K` is set to the value of

$$\int_{x_{JINT-k+j}}^{x} M_{k,JINT-k+j}(\xi)\, d\xi \qquad j=1,2,...,k$$

in the case when $1 \le JINT-k+j \le n-k$ and in all other cases is set to zero.

If $x < x_1$ then the contents of `VINT` are set to zero. If $x \ge x_n$ the integral

$$\int_{x_{n-k}}^{x} M_{k,n-k}(\xi)\, d\xi$$

has the value k^{-1} and this is stored in `VINT(1),` with `VINT(J)=0, J=2,K.`

2.2 Checks on the input parameters

The restrictions imposed on the input parameters are checked by the subroutine. For example, the value of `K` is checked to see if it satisfies the inequality `1≤K≤N-1.` If the restrictions are not satisfied an error diagnostic is printed, an error return flag, `IFAIL,` is set (see §2.3), and a return is made to the calling program.

2.3 The Common area and diagnostic messages

The subroutine uses a Common area which the user may also reference. To do this the calling program should include a Common statement of the form

The single precision version:

```
COMMON/TG04B/ LP,IFAIL
```

The double precision version:

```
COMMON/TG04BD/ LP,IFAIL
```

`LP` is an `INTEGER` variable which specifies the Fortran stream number to be used for the error messages. The default value is 6 (line printer). To suppress the printing of error messages set `LP` to zero.

`IFAIL` is an `INTEGER` variable which is always set by the subroutine to indicate success or failure. On exit from the subroutine `IFAIL` will take one of the following values.

0 successful entry
1 $n<k+1$
2 $k<1$
3 the data points are not in ascending order
4 more than k data points coalesce
5 `NORM` was not equal to 1 or 2 (it was reset to 1)

3 GENERAL INFORMATION

Use of Common: the subroutine uses a Common area `TG04B/BD`, see §2.3.

Workspace: none.

Other subroutines: none.

Input/Output: In the event of errors diagnostic messages are printed. The output stream for these may be changed or the messages suppressed by altering the Common variable `LP`, see §2.3.

System dependence: none.

Restrictions:

$n>0$,

$1 \le k \le n-1$,

$x_1 \le x_2 \le ... \le x_n$,

$x_i < x_{i+k}$, $i=1,2,...,n-k$.

4 METHOD

To evaluate the B-splines $M_{k,i}(x)$, $1 \le i \le n-k$, at the point x we need only compute k numbers. To do this we employ the recurrence relation of De Boor (1972) and Cox (1972),

$$M_{k,i}(x) = \frac{(x-x_i)M_{k-1,i}(x) + (x_{i+k}-x)M_{k-1,i+1}(x)}{x_{i+k}-x_i}$$

where

$$M_{1,i}(x) = \begin{cases} (x_{i+1}-x_i)^{-1} & x_i \le x < x_{i+1} \\ 0 & \text{otherwise,} \end{cases}$$

in the scheme proposed by Cox (1975).

In order to compute the corresponding integrals we use the formula

$$\int_{x_i}^{x} M_{k,i}(\xi)\,d\xi = \begin{cases} k^{-1}\sum_{r=0}^{k-1}(x-x_{i+r})M_{k-r,i+r}(x), & x_i \le x < x_{i+k} \\ k^{-1} & x = x_{i+k}, \end{cases}$$

which is due to Gaffney (1974).

References

Cox,M.G.(1972) ‘‘The Numerical Evaluation of B-splines’’, J.Inst Maths.Applics. **10**, 134-149.

Cox,M.G.(1975) ‘‘An Algorithm for Spline Interpolation’’, J.Inst Maths.Applics. **15**, 95-108.

De Boor,C.(1972) ‘‘On Calculating with B-splines’’, J.Approx.Theory **6**, 50-62.

Gaffney,P.W.(1974) ‘‘The Calculation of Indefinite Integrals of B-splines’’, Harwell report CSS 10.

5 EXAMPLE OF USE

5.1 First example

Suppose we are given the ten data points

$$\begin{aligned} x_1 = x_2 = x_3 &= 0.0 \\ x_4 = x_5 &= 1.0 \\ x_6 &= 3.0 \\ x_7 &= 4.0 \\ x_8 = x_9 = x_{10} &= 6.0 \end{aligned}$$

and we require the elements in the matrix

$$\mathbf{N} = \left\{ \begin{matrix} N_{3,1}(t_1) & N_{3,2}(t_1) & N_{3,3}(t_1) & N_{3,4}(t_1) \\ N_{3,1}(t_2) & N_{3,2}(t_2) & N_{3,3}(t_2) & N_{3,4}(t_2) \\ : & : & : & : \\ N_{3,1}(t_7) & N_{3,2}(t_7) & N_{3,3}(t_7) & N_{3,4}(t_7) \end{matrix} \right\}$$

where $N_{3,j}(x)$ is a normalized B-spline of degree 2 with knots at $x_j, x_{j+1},.., x_{j+k}$, and

$$t_i = \frac{i-1}{4}, \quad i=1,2,..,7$$

The Fortran code to generate the matrix **N** might be as follows

```
      REAL X(10),NA(7,4),V(3),VINT(3)
      DATA X/0.0,0.0,0.0,1.0,1.0,3.0,4.0,6.0,6.0,6.0/
        …
        …
      N=10                                   number of points x_i  i=1,2,...,n
      NORM=2                                 set to compute normalized B-splines, i.e. N_k,i(x)
      K=3                                    the order k of the B-spline
      DO 10 I=1,7                            initialize the elements of N …
      DO 10 J=1,4                            … to zero so that in the next loop …
   10 NA(I,J)=0.                             … we need only set the non-zero ones
      DO 30 I=1,7
      TI=FLOAT(I-1)/4.                       set the argument t_i
      CALL TG04A(N,X,K,NORM,TI,JINT,V,VINT)  compute N_3,j(t_i)
      DO 20 J=1,K                            set the values of …
      J1=JINT-K+J                            … N_3,JINT-3+j(t_i) …
   20 NA(I,J1)=V(J)                          … in the ith row of N
   30 CONTINUE
        …
        …
```

5.2 The second example

Suppose we are given the data points of the first example and we wish to compute the value of the B-spline $M_{3,2}(x)$ and the corresponding integral

$$\int_{x_2}^{x} M_{3,2}(\xi)\, d\xi$$

at the values of $x=(i-1)/100$, $i=1,2,...,101$. The Fortran code for generating these values might be as follows

`      REAL X(10),VALUE(101),VALINT(101),V(3),VINT(3)`	
`      DATA X/0.0,0.0,0.0,1.0,1.0,3.0,4.0,6.0,6.0,6.0/`	
`        …`	
`        …`	
`      NORM=1`	set to compute $M_{k,i}(x)$
`      K=3`	the order k of the B-spline
`      N1=K+1`	the number of knots
`      DO 20 I=1,101`	
`      XVALUE=.01*FLOAT(I-1)`	set the argument x
`      CALL TG04A(N1,X(2),K,NORM,XVALUE,JINT,V,VINT)`	compute value of …
	… $M_{3,2}(x)$ at x=***XVALUE***
`      J=MAX0(1,N1-JINT)`	select required elements of ***V*** and ***VINT*** †
`      VALUE(I)=V(J)`	put the B-spline value and its integral value …
`   20 VALINT(I)=VINT(J)`	… into the arrays ***VALUE*** and ***VALINT***
`        …`	
`        …`	

Figure 2. The Fortran code for the second example.

† $M_{3,2}(x)$ and $\int_{x_2}^{x} M_{3,2}(\xi)\, d\xi$ are stored in `V(1+K-JINT)` and `VINT(1+K-JINT)`, when `1+K-JINT` ≥ 1 and in `V(1)` and in `VINT(1)` otherwise.