DEPARTMENT OF MATHEMATICS
UNIVERSITY OF HOUSTON HOUSTON, TEXAS

NASA CR-
141623

(NASA-CR-141623) A SIMPLIFIED PACKAGE FOR CALCULATING WITH SPLINES (Houston Univ.) CSCL 12A

N75-17139

Unclas
G3/64 10221

A SIMPLIFIED PACKAGE FOR CALCULATING WITH SPLINES

BY PHILIP W. SMITH

REPORT #37 OCT. 1974

PRICES SUBJECT TO CHANGE

PREPARED FOR
EARTH OBSERVATION DIVISION, JSC
UNDER
CONTRACT NAS-9-12777

3801 CULLEN BLVD.
HOUSTON, TEXAS 77004

A Simplified Package For Calculating With Splines

By

Philip W. Smith

Mathematics Department

Texas A & M University

Consultant to

Mathematics Department

University of Houston

Houston, Texas

October, 1974

NAS-9-12777

Report #37

ABSTRACT

This package is designed to make all elementary calculations involved in evaluating

$$s^{(j)}(t) \equiv \left(\sum_{i=1}^{N} A_i N_{i,k}(t)\right)^{(j)} , \; j = 0,1,\ldots,$$

fairly easy where the $N_{i,k}$ are usual normalized B-splines. In addition interpolation and least squares subroutines are provided so that one may easily generate "meaningful" spline functions.

0. Basic Spline Package

This package is designed to solve some of the elementary problems associated with splines (eg. spline interpolation and least squares fit). I have attempted to keep the number of technical subroutines to a minimum while giving the user great flexibility in determining the "type" of spline desired. This flexibility occurs in the freedom the user has in his choice of knot sequence and/or his choice of piecewise degree.

The subroutines fall into three basic categories. The first category involves computations with a given spline and/or knot sequence. These routines deal with general properties of splines and are not tied down to any specific task. Included in this category is

BSPLVN

BVALUE

INTERV .

The second category involves routines which calculate the coefficients of splines which perform a specific task (such as interpolation or least squares fit). In this category are

EQUATE

INTERP .

Finally, in the third category is a banded equation solver BNDSLV which has nothing directly to do with splines, but which is needed to solve specific linear equations which arise in EQUATE and INTERP.

The reader interested in more sophisticated applications should consult [2], where BSPLVN, BVALUE, and INTERV were obtained and where certain program fragments of EQUATE and INTERP were sketched. Programs which, for example, solve collocation problems of differential equations are also located there.

The appendix contains a listing of the above subroutines.

1. Normalized B-splines

The normalized B-splines are used almost exclusively for calculating and evaluating piecewise polynomials. Here we give a very brief description of the normalized B-splines and some of their most important properties. For more information the reader should consult [1], and the references therein.

We set

$$(1.1) \qquad (s-t)_+ = \begin{cases} s-t & \text{if} \quad t \le s \\ 0 & \text{if} \quad t > s \end{cases} .$$

Let k be a positive integer which we will fix throughout this section. The biinfinite knot sequence $\{t_i\}_{i=-\infty}^{\infty}$ will satisfy the following properties:

$$(1.2) \qquad \begin{aligned} & t_i \ge t_{i-1} \\ & t_{i+k} > t_i \\ & a = \lim_{i \to -\infty} t_i \\ & b = \lim_{i \to \infty} t_i \qquad \text{(a or b may be infinite).} \end{aligned}$$

Although we will always be working on a <u>finite</u> interval with a <u>finite</u> number of knots, it is convenient, for the purpose of exposition, to work with the above biinfinite sequence.

The <u>normalized B-spline of order k determined by $\{t_i,\ldots,t_{i+k}\}$</u> is denoted by $N_{i,k}$ and is defined by

$$(1.3) \qquad N_{i,k}(t) = (t_{i+k}-t_i)\,[t_i,\ldots,t_{i+k}]\,(s-t)_+^{k-1} ,$$

where the k-th divided difference operator $[t_i,\ldots,t_{i+k}]$ is acting on the variable s. For example if k=1 we obtain

$$(1.4)\qquad [t_i,t_{i+1}]\ (s-t)_+^0 = \frac{(t_{i+1}-t)_+^0 - (t_i-t)_+^0}{t_{i+1}-t_i}\quad .$$

Since there might be some confusion as to the value of $N_{i,1}(t)$ at t_{i+1} or t_i we arbitrarily declare $N_{i,k}$ to be <u>right continuous</u>.

The following properties of the $N_{i,k}$ are very useful:

(1.5) i) $N_{i,k}(t) \geq 0$

ii) $\text{supp}\ (N_{i,k}) = [t_i,\ t_{i+k}]$

iii) $\sum_{i=-\infty}^{\infty} N_{i,k}(t) = 1\quad ,\ t\varepsilon(a,b)\quad .$

Let us suppose that $t_k < t_{k+1}$ and $t_N < t_{N+1}$, then the set

$$(1.6)\qquad \{N_{i,k}\}_{i=1}^{N}$$

is linearly independent on the interval $[t_k,t_{N+1}]$. Furthermore, the set $\{N_{i,k}\}_{i=1}^{N}$ forms a basis for the piecewise polynomials of order k (degree $\leq$ k-1) on the interval $[t_k,\ t_{N+1}]$ with knots at $\{t_{k+1},\ldots,t_N\}$ and continuity conditions as prescribed by those knots. That is, if

$$(1.7)\qquad t_k \leq t_j < t_{j+1} = \cdots = t_{j+m} < t_{j+m+1} \leq t_{N+1}$$

then the polynomial pieces must join at t_{j+1} in a C^{k-m-1} fashion. Thus if there are no knot multiplicities the $N_{i,k}$'s span the $C^{k-2}[t_k, t_{N+1}]$ piecewise polynomials of order k with knots at $\{t_{k+1}, \ldots, t_N\}$.

2. Evaluating Spline Expressions

In general, one has an expression of the form

$$(2.1) \qquad \sum_{i=1}^{N} A_i \, N_{i,k}(t) \equiv s(t)$$

to evaluate in the interval $[t_k, t_{N+1}]$ (recalling that $t_1,\ldots,t_{N+k}$ are from some given knot sequence). Due to the local support property of the $N_{i,k}$'s (1.5, ii) if $t_j \leq t < t_{j+1}$ then

$$(2.2) \qquad s(t) = \sum_{i=j-k+1}^{j} A_i N_{i,k}(t) \quad .$$

Thus in order to evaluate (2.1) efficiently the first step consists of finding the index j such that

$$(2.3) \qquad t_j \leq t < t_{j+1} \quad .$$

The subroutine INTERV accomplishes this.

In evaluating (2.1) and its derivatives the following formula from [1] is useful.

$$(2.4) \qquad s^{(j)}(t) = (k-1) \cdots (k-j) \sum_{i=1}^{N} A_i^{(j)} \, N_{i,k-j}(t)$$

where

$$(2.5) \qquad A_i^{o} = A_i \qquad \text{and}$$

$$A_i^{(j)} = (A_i^{(j-1)} - A_{i-1}^{(j-1)})/(t_{i+k-j} - t_i), \; j > 0 \quad .$$

Notice that evaluating the derivative of a spline written in terms of the normalized B-spline basis reduces to evaluating a lower order spline in a similar basis. These handy features are combined in the function routine BVALUE.

3. Computing the Coefficients of the $N_{i,k}$

In the previous section we indicated how to calculate the value of a spline or its derivative if one had the coefficients of the representation. In order to obtain the coefficients one generally has to solve a linear system of equations whose matrix entries depend on the values of the $N_{i,k}$ or its derivatives. For example, if the interpolating spline is desired for the function f, then one needs to solve the system

$$\sum_{i=1}^{N} A_i \, N_{i,k}(x_j) = f(x_j), \; j=1,\ldots,N. \tag{3.1}$$

Assuming that the x_j's are ordered then (3.1) will be a banded linear system. The matrix will have the form

$$\alpha_{ij} = N_{i,k}(x_j) \quad . \tag{3.2}$$

Thus it would be useful to have a subroutine which generates <u>simultaneously</u> the values of all the nonzero normalized B-splines. The subroutine BSPLVN accomplishes this task. Further, the subroutine BNDSLV is used to solve banded systems of linear equations.

4. Spline Interpolation

This section is a user's guide to the subroutine INTERP. The inputs to INTERP(T,A,N,K,TAU,F,IFLAG) are

T - (<u>ordered</u>) knot sequence

N - number of equations

K - order of the spline

TAU - (<u>ordered</u>) interpolation points

F - external user supplied function .

The returns are

A - N coefficients of the $N_{I,k}$

IFLAG - IFLAG = 0 or 1.

This routine solves the linear system of equations

$$\sum_{I=1}^{N} A(I)N_{I,k}\ (TAU(J)) = F(TAU(J)),\ J = 1,\dots,N.$$

If the linear system is singular then IFLAG is set to 1 otherwise it is zero. The user must supply $\{T(I) : I = 1,\dots,N+K\}$ and $\{TAU(I) : I = 1,\dots,N\}$, both non-decreasing sequences with $T(I) < T(I+K)$ and $TAU(I) < TAU(I+1)$.

It can be shown that the linear system is invertible if and only if $TAU(I) \in (T(I),T(I+K))$. If K is even then a fairly stable interpolation procedure is obtained by choosing

$$TAU(I) = T(K) + (I-1)\Delta_1 \text{ , } I = 1,\ldots,K/2$$

$$TAU(I) = T(K + (I-K/2)), \text{ } I = K/2 + 1,\ldots,N - K/2,$$

$$TAU(I) = T(N+1) - (N-I)\Delta_2 \text{ , } I = N - K/2 + 1,\ldots,N$$

where

$$\Delta_1 = \frac{T(K+1)-T(K)}{K/2} \quad \text{and } \Delta_2 = \frac{T(N+1) - T(N)}{K/2} \quad .$$

INTERP calls BSPLVN, INTERV, and BNDSLV. As the subroutines are dimensioned now it is assumed that $K \leq 10$.

5. Spline Least Squares

This section is a user's guide to the subroutine EQUATE. The inputs to EQUATE (T,N,K,LX,X,G,WEIGHT,A) are

T - (ordered) knot sequence

N - number of equations

K - order of the spline

LX - number of data points

X - (ordered) x-co-ordinate

G - Y-values

WEIGHT - (positive) vector .

The return is

A - N coefficients of the $N_{I,K}$.

The returned coefficients satisfy

$$\sum_{J=1}^{LX} \mathrm{WEIGHT}(J)\left(\sum_{I=1}^{N} A(I)N_{I,K}(x(J)) - G(J)\right)^2 \le \sum_{J=1}^{LX} \mathrm{WEIGHT}(J)\left(\sum_{I=1}^{N} b_I N_{I,K}(x(J)) - G(J)\right)^2$$

for any vector $b = (b_1,\ldots,b_n)$.

It is assumed that $\{X(J), J=1,\ldots,LX\}$ are ordered an further that $T(K) \le X(1) \le X(LX) \le T(N+1)$. This routine calls BSPLVN and BNDSLV. As the subroutines are dimensioned now it is assumed that $K \le 10$.

6. BVALUE

The function routine BVALUE will return the value of a spline or any one of its derivatives (depending on the calling parameters). The inputs for BVALUE(T,A,N,K,X,IDERIV) are

T - (<u>ordered</u>) knot sequence

A - coefficients of the spline

N - integer such that X ϵ [T(K), T(N+1)]

K - order of the spline

X - evaluation point

IDERIV - derivative wanted.

The return is

$$\text{BVALUE(T,A,N,K,X,IDERIV)} = S^{(\text{IDERIV})}(X)$$

where $S(\cdot) = \sum_{I=1}^{N} A(I)N_{I,K}(\cdot)$. This routine will return a value of zero if X $\notin$ [T(K), T(N+1)]. This routine calls INTERV.

This routine is based on formulae (2.2) and (2.5) along with the basic identity

$$(6.1)\quad N_{i,k}(t) = \frac{t-t_i}{t_{i+k}-t_i} N_{i,k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_i} N_{i+1,k-1}(t) \quad .$$

7. INTERV

The subroutine INTERV will return the subscript ILEFT which satisfies

$$T(ILEFT) \leq X < T(ILEFT + 1)$$

if possible. This routine is used for efficient evaluation of a spline as in (2.2). More precisely INTERV(XT,LXT,X,ILEFT,MFLAG) has inputs

XT - (ordered) vector

LXT - Index of the right hand end point XT(LXT).

X - evaluation point.

The returns are ILEFT and MFLAG where MFLAG = 0 means that $1 \leq ILEFT \leq LXT - 1$ and $XT(ILEFT) \leq X < XT(ILEFT + 1)$. This is the usual case. If MFLAG = -1 then $X < XT(1)$ and ILEFT = 1. If MFLAG = 1 then $XT(LXT) \leq X$ and ILEFT = LXT.

8. BSPLVN

The subroutine BSPLVN is used to produce (simultaneously) the values of the nonzero normalized B-spline $N_{i,k}$ at a given point X. This routine is used here exclusively to fill out the various coefficient matrices. Then inputs to BSPLVN(T,JHIGH,INDEX,X,ILEFT,VNIKX) are

T - (ordered) knot sequence

JHIGH - order of the spline

INDEX - either 1 or 2

X - evaluation point

ILEFT - where $T(ILEFT) \leq X < T(ILEFT + 1)$.

The return is

$$VNIKX(J) = N_{ILEFT-JHIGH+J,JHIGH}(X), \; J=1,\ldots,JHIGH$$

if INDEX = 1. This is the only option we have used in EQUATE and INTERP. If INDXE = 2, then

$$VNIKX(I) = N_{ILEFT+I-J',J'}(X)$$

where $J' = \max\{JHIGH,J+1\}$ and J is a local variable in the subroutine. This option is useful in generating simultaneously the values of some or all of the derivatives of the nonzero $N_{i,k}$ at X.

Note that before calling BSPLVN one must find ILEFT so that $T(ILEFT) \leq X < T(IEFT+1)$.

9. BNDSLV

The subroutine BNDSLV is designed to solve banded systems of linear equations. The input parameters for BNDSLV(A,B,X,N,IROWLN) are

A - Banded matrix

B - right hand side

N - number of equations

IROWLN - the row length (band width).

The return is X a vector which solves

$$MX = B ,$$

where (setting $R \equiv (IROWLN + 1)/2$)

$$M(K,J) = \begin{cases} A(K,R + J - K), & |J-K| \leq R - 1 \\ 0 & \text{otherwise} \end{cases} .$$

In short, the diagonal of M is stored in the R-th column of A and the rest of the entries are placed accordingly. The routine uses elimination with no provision for pivoting.

References

1. C. de Boor, On Calculating with B-spline, J. of Approximation Theory 6 (1972), 50-62.

2. C. de Boor, Package for Calculating with B-splines, MRC Technical Summary Report #1333, Oct. 73.

APPENDIX

SAMPLE MAIN Program

```
FORTRAN IV G LEVEL  21                    MAIN                DATE = 74249          09/27/40

                DIMENSION T(300),A(300)
                COMMON /DATA/ LX,X(201),G(201),WEIGHT (201)
                DATA T /3*0.,2.1,5.3,3*10./
                READ (5,600) N,K
          600   FORMAT (I3,I2)
                READ (5,510) (G(I),I=1,201)
          510   FORMAT (5E15.7)
                DO 2 I=1,201
                X(I)=FLOAT(I)/20-.05
            2   CONTINUE
                LX=200
                LXP1=201
                CALL EQUATE (T,N,K,LX,X,G,WEIGHT,A)
                WRITE (6,650)
          650   FORMAT ('0',12X,'COEFFICIENTS',/)
                DO 22 I=1,N
                WRITE (6,660) I,A(I)
          660   FORMAT (2G15.7)
           22   CONTINUE
                IDERIV=0
                TOTAL =0.
                WRITE (6,670)
          670   FORMAT ('0','EVALUATION OF LEAST SQUARES FIT AT X(I)',/)
                DO 33 I=1,LXP1
                Y=X(I)
                Z=EVALUE(T,A,N,K,Y,IDERIV)
                TOTAL = TOTAL +(Z-G(I))**2
                WRITE (6,630) Y,Z
          630   FORMAT (2G15.7)
           33   CONTINUE
                WRITE (6,680)
          680   FORMAT ('0','VARIANCE',/)
                WRITE (6,640) TOTAL
          640   FORMAT ('0',1G15.7)
                STOP
                END
```

```
      BLOCK DATA
      COMMON /DATA/ LX,X(201),G(201),WEIGHT (201)
      REAL WEIGHT/201*1./
      END
```

```
      SUBROUTINE EQUATE (T,N,K,LX,X,G,WEIGHT,A)
C     THIS VERSION OF EQUATE RETURNS THE DESIRED COEFFICIENTS
      KM1=K-1
      KPKM1=K+KM1
      DIMENSION T(300), A(300), B(300), Q(300,20),VNIKX(20)
      DIMENSION X(201),G(201),WEIGHT(201)
C     ***ZEROS OUT STORAGE ARRAYS
      DO 7 I=1,N
      B(I)=0.
      DO 7 J=1,KPKM1
      Q(I,J)=0
    7 CONTINUE
      ILEFT=K
      IMK=0
C     ***SEARCH FOR APPROPRIATE INTERVAL
      DO 20 L=1,LX
   10 IF (ILEFT .EQ. N) GO TO 15
      IF (X(L) .LT. T(ILEFT+1)) GO TO 15
      ILEFT=ILEFT+1
      IMK=ILEFT-K
      GO TO 10
C     ***SUBROUTINE TO CALCULATE THE VALUE OF THE B-SPLINE AT X
   15 CALL BSPLVN (T,K,1,X(L),ILEFT,VNIKX)
      DO 20 JJ=1,K
      DW=VNIKX(JJ)*WEIGHT(L)
      I=IMK+JJ
C     ***CALCULATION OF THE RIGHT HAND SIDE OF THE SYSTEM
      B(I)=DW*G(L) +B(I)
      J=K
      DO 20 M=JJ,K
C     ***CALCULATION OF THE UPPER RIGHT HAND PORTION OF THE MATRIX
      Q(I,J)=DW*VNIKX(M)+Q(I,J)
      J=J+1
   20 CONTINUE
      NM1=N-1
      DO 30 I=1,NM1
      DO 30 J=1,KM1
C     ***COMPLETION OF THE MATRIX BY SYMETRY
   30 Q(I+J,K-J)=Q(I,K+J)
      KPKM1=2*K-1
C     ***SUBROUTINE TO SOLVE THE BANDED SYSTEM OF EQUATIONS
      CALL BNDSLV (Q,B,A,N,KPKM1)
      RETURN
      END
```

```
      SUBROUTINE BNDSLV(A,B,X,N,IROWLN)
      DIMENSION A(300,IROWLN)
      DIMENSION B(N), X(N)
      IRWHLF = (IROWLN - 1)/2
      IRWMID = IRWHLF + 1
      IMIDP1 = IRWMID + 1
      NM1 = N - 1
      DO 150 K=1,NM1
      JSTART = IMIDP1
      JEND = IROWLN
      ISTART = K + 1
      IEND = K + IRWHLF
      IF(IEND.GT.N)  IEND = N
      DO 140 I=ISTART,IEND
      JSTART = JSTART - 1
      JEND = JEND - 1
      AMULT = A(I,JSTART - 1)/A(K,IRWMID)
      J1 = IRWMID
      DO 130 J=JSTART,JEND
      J1 = J1 + 1
      A(I,J) = A(I,J) - AMULT*A(K,J1)
  130 CONTINUE
      B(I) = B(I) - AMULT*B(K)
  140 CONTINUE
  150 CONTINUE
      X(N) = B(N)/A(N,IRWMID)
      K = N
      DO 210 I=1,NM1
      K = K - 1
      SUM = 0.
      J1 = K
      DO 200 J=IMIDP1,IROWLN
      J1 = J1 + 1
      IF(J1.GT.N)  GO TO 200
      SUM = SUM + A(K,J)*X(J1)
  200 CONTINUE
      X(K) = (B(K) - SUM)/A(K,IRWMID)
  210 CONTINUE
      RETURN
      END
```

```
                 C
0001                  SUBROUTINE BSPLVN ( T, JHIGH, INDEX, X, ILEFT, VNIKX )
                 CALCULATES THE VALUE OF ALL POSSIBLY NONZERO B-SPLINES AT *X* OF
                 C  ORDER MAX(JHIGH,(J+1)(INDEX-1)) ON *T*.
0002                  DIMENSION T(1),VNIKX(1)
0003                  DIMENSION DELTAM(20),DELTAP(20)
0004                  DATA J/1/
                 CONTENT OF J, DELTAM, DELTAP IS EXPECTED UNCHANGED BETWEEN CALLS.
0005                                                  GO TO (10,20),INDEX
0006               10 J = 1
0007                  VNIKX(1) = 1.
0008                  IF (J .GE. JHIGH)                GO TO 99
                 C
0009               20    IPJ = ILEFT+J
0010                     DELTAP(J) = T(IPJ) - X
0011                     IMJP1 = ILEFT-J+1
0012                     DELTAM(J) = X - T(IMJP1)
0013                     VMPREV = 0.
0014                     JP1 = J+1
0015                     DO 26 L=1,J
0016                        JP1ML = JP1-L
0017                        VM = VNIKX(L)/(DELTAP(L) + DELTAM(JP1ML))
0018                        VNIKX(L) = VM*DELTAP(L) + VMPREV
0019               26       VMPREV = VM*DELTAM(JP1ML)
0020                     VNIKX(JP1) = VMPREV
0021                     J = JP1
0022                     IF (J .LT. JHIGH)             GO TO 20
                 C
0023               99                                  RETURN
0024                  END
```

```
      C
            SUBROUTINE INTERV ( XT, LXT, X, ILEFT, MFLAG )
      COMPUTES LARGEST ILEFT IN (1,LXT) SUCH THAT XT(ILEFT) .LE. X
            DIMENSION XT(LXT)
            DATA ILO /1/
            IHI = ILO + 1
            IF (IHI .LT. LXT)                      GO TO 20
               IF (X .GE. XT(LXT))                 GO TO 110
               IF (LXT .LE. 1)                     GO TO 90
               ILO = LXT - 1
                                                   GO TO 21
         20 IF (X .GE. XT(IHI))                    GO TO 40
         21 IF (X .GE. XT(ILO))                    GO TO 100
      C**** NOW X .LT. XT(IHI) . FIND LOWER BOUND
         30 ISTEP = 1
         31 IHI = ILO
            ILO = IHI - ISTEP
            IF (ILO .LE. 1)                        GO TO 35
            IF (X .GE. XT(ILO))                    GO TO 50
            ISTEP = ISTEP*2
                                                   GO TO 31
         35 ILO = 1
            IF (X .LT. XT(1))                      GO TO 90
                                                   GO TO 50
      C**** NOW X .GE. XT(ILO) . FIND UPPER BOUND
         40 ISTEP = 1
         41 ILO = IHI
            IHI = ILO + ISTEP
            IF (IHI .GE. LXT)                      GO TO 45
            IF (X .LT. XT(IHI))                    GO TO 50
            ISTEP = ISTEP*2
                                                   GO TO 41
         45 IF (X .GE. XT(LXT))                    GO TO 110
            IHI = LXT
      C**** NOW XT(ILO) .LE. X .LT. XT(IHI) . NARROW THE INTERVAL
         50 MIDDLE = (ILO + IHI)/2
            IF (MIDDLE .EQ. ILO)                   GO TO 100
      C     NOTE. IT IS ASSUMED THAT MIDDLE = ILO IN CASE IHI = ILO+1
            IF (X .LT. XT(MIDDLE))                 GO TO 53
               ILO = MIDDLE
                                                   GO TO 50
         53    IHI = MIDDLE
                                                   GO TO 50
      C**** SET OUTPUT AND RETURN
         90 MFLAG = -1
```

23

```
      ILEFT = ILO
                                          RETURN
  110 MFLAG = 1
      ILEFT = LXT
                                          RETURN
      END
```

```
C
      FUNCTION BVALUE ( T, A, N, K, X, IDERIV )
CALCULATES VALUE AT *X* OF *IDERIV*-TH DERIVATIVE OF SPLINE FROM B-REPR.
      DIMENSION T(1),A(1)
      DIMENSION AJ(20),DP(20),DM(20)
      BVALUE = 0.
      KMIDER = K - IDERIV
      IF (KMIDER .LE. 0)                  GO TO 99
C
C  *** FIND *I* IN (K,N) SUCH THAT T(I) .LE. X .LT. T(I+1)
C       (OR, .LE. T(I+1) IF T(I) .LT. T(I+1) = T(N+1)).
      KMI = K-1
      CALL INTERV (T(K), N+1-KM1, X, I, MFLAG )
      I = I + KMI
      IF (MFLAG)                          99,20,9
    9 IF (X .GT. T(I))                    GO TO 99
   10 IF (I .EQ. K)                       GO TO 99
      I = I - 1
      IF (X .EQ. T(I))                    GO TO 10
C
C  *** DIFFERENCE THE COEFFICIENTS *IDERIV* TIMES.
   20 IMK = I-K
      DO 21 J=1,K
         IMKPJ = IMK + J
   21    AJ(J) = A(IMKPJ)
      IF (IDERIV .EQ. 0)                  GO TO 30
   22 DO 23 J=1,IDERIV
         KMJ = K-J
         FKMJ = FLOAT(KMJ)
         IHI = I
         DO 23 JJ=1,KMJ
            IHI = IHI+1
            IHMKMJ = IHI - KMJ
   23       AJ(JJ) = (AJ(JJ+1) - AJ(JJ))/(T(IHI) - T(IHMKMJ))*FKMJ
C
C  *** COMPUTE VALUE AT *X* IN (T(I),T(I+1)) OF IDERIV-TH DERIVATIVE,
C      GIVEN ITS RELEVANT B-SPLINE COEFF. IN AJ(1),...,AJ(K-IDERIV).
   30 IF (IDERIV .EQ. KMI)                GO TO 39
      IP1 = I+1
      DO 32 J=1,KMIDER
         IPJ = I + J
         DP(J) = T(IPJ) - X
         IP1MJ = IP1 - J
   32    DM(J) = X - T(IP1MJ)
      IDERP1 = IDERIV+1
      DO 33 J=IDERP1,KM1
```

```
0039          DO 33 JJ=1,KMJ
0040          AJ(JJ) = (AJ(JJ+1)*DM(ILO) + AJ(JJ)*DP(JJ))/(DM(ILO)+DP(JJ))
0041       33 ILO = ILO - 1
0042       39 BVALUE = AJ(1)
      C
0043       99                             RETURN
0044          END
```

96

```
              IMPLICIT REAL*8 (A-H,O-Z)
              DIMENSION T(20),TAU(20),A(20)
              EXTERNAL F
              READ (5,100) K,N
        100   FORMAT(2I4)
              J=N+K
              READ(5,110) (T(I),I=1,J)
              READ(5,110) (TAU(I),I=1,N)
        110   FORMAT(10D8.2)
              WRITE(6,115) K,N,(T(I),I=1,J),(TAU(I),I=1,N)
        115   FORMAT(1X,',K=',I4,'N=',I4,13(1X,D8.2))
              CALL INTERP(T,A,N,K,TAU,F,IFLAG)
              IF(IFLAG.EQ.1) GO TO 10
              WRITE(6,120)(A(I),I=1,N)
        120   FORMAT(1H 'COEF A',10(1X,D10.2))
              DO 5 J1=1,4
              DO 5 I=1,5
              J=J1-1
              X=I-3
              Y=F(X)
              WRITE(6,130) X,Y
        130   FORMAT(1H ,'ACTUAL VALUES',5X,'X=  ',D10.2,5X,'Y= ',D10.2)
              Y=BVALUE(T,A,N,K,X,J)
        5     WRITE(6,140) J,X,Y
        140   FORMAT(1H 'DER= ',I5,5X,'X= ',D10.2,5X,'Y= ',D10.2)
              STOP
        10    WRITE(6,150)
        150   FORMAT(1H ,'MATRIX SINGULAR')
              STOP
              END
```

```
      SUBROUTINE INTERP(T,A,N,K,TAU,F,IFLAG)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Q(20,20),T(20),A(20),TAU(20),DUMMY(20),B(20)
      KM1=K-1
      NP2MK=N+2-K
      KPKM1=K+K-1
      DO 30 I=1,N
           DO 13 J=1,KPKM1
   13      Q(I,J)=0.
      CALL INTERV(T(K),NP2MK,TAU(I),ILEFT,MFLAG)
      ILEFT=ILEFT+KM1
      IF(MFLAG) 99,15,14
   14 IF(I.LT.N) GO TO 99
      ILEFT=N
   15 CALL BSPLVN(T,K,1,TAU(I),ILEFT,DUMMY)
      L=ILEFT-I
      DO 16 J=1,K
      L=L+1
   16 Q(I,L)=DUMMY(J)
      IF(Q(I,K).EQ.0) GO TO 99
   30 B(I)=F(TAU(I))
      CALL BNDSLV(Q,B,A,N,KPKM1)
      IFLAG=0
      RETURN
   99 IFLAG=1
      RETURN
      END
```

```
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION T(20),TAU(20),A(20)
      EXTERNAL F
      READ (5,100) K,N
  100 FORMAT(2I4)
      J=N+K
      READ(5,110) (T(I),I=1,J)
      READ(5,110) (TAU(I),I=1,N)
  110 FORMAT(10D8.2)
      WRITE(6,115) K,N,(T(I),I=1,J),(TAU(I),I=1,N)
  115 FORMAT(1X,',K=',I4,'N=',I4,13(1X,D8.2))
      CALL INTERP(T,A,N,K,TAU,F,IFLAG)
      IF(IFLAG.EQ.1) GO TO 10
      WRITE(6,120)(A(I),I=1,N)
  120 FORMAT(1H 'COEF A',10(1X,D10.2))
      DO 5 J1=1,4
      DO 5 I=1,5
      J=J1-1
      X=I-3
      Y=F(X)
      WRITE(6,130) X,Y
  130 FORMAT(1H ,'ACTUAL VALUES',5X,'X=  ',D10.2,5X,'Y= ',D10.2)
      Y=BVALUE(T,A,N,K,X,J)
    5 WRITE(6,140) J,X,Y
  140 FORMAT(1H 'DER= ',I5,5X,'X= ',D10.2,5X,'Y= ',D10.2)
      STOP
   10 WRITE(6,150)
  150 FORMAT(1H ,'MATRIX SINGULAR')
      STOP
      END
```