An algorithm for the interpolation of functions using quintic splines

ALGORITHM 005

E. H. Mund, P. Hallet and J. P. Hennart (*)

ABSTRACT

A method is described for the interpolation of N arbitrarily given data points using fifth degree polynomial spline functions. The interpolating spline is built from a set of basis functions belonging to the fifth degree smooth Hermite space. The resulting algebraic system is symmetric and bloc-tridiagonal. Its solution is calculated using a direct inversion method, namely a block-gaussian elimination without pivoting. Various boundary conditions are provided for independently at each end point. The stability of the algorithm is examined and some examples are given of experimental convergence rates for the interpolation of elementary analytical functions. A listing is given of the two FORTRAN subroutines INSPL5 and SPLIN5 which form the algorithm.

1. INTRODUCTION

This paper is concerned with the interpolation of numerical data points using polynomial spline functions. Various algorithms have already been published on this subject which are characterized either by their choice of a set of basis functions [1, 2], or by the numerical method used to solve the resulting system of algebraic equations [3, 4]. Frequent use is made for instance, of the so-called B-splines [5] and iterative methods appear in SPÄTH's work to invert a block-tridiagonal matrix.
In this work, a set of piecewise Hermite polynomials of fifth degree is used in which the unknown parameters may be determined by specifying continuity conditions at each node. A system of algebraic equations arises which is always symmetric and block-tridiagonal. The same boundary conditions as in SPÄTH's work [4] are provided, but *independently* at each boundary. After balancing the system of equations [6] a direct method adapted from Cholesky's inversion technique [7] is used, allowing an evaluation of the interpolating spline function at any point of interest.

2. GENERAL FORMALISM

Following GREVILLE [5], a quintic spline function (QSF) $S(x)$ with N knots $(x_1 < \ldots < x_N)$ may be defined as a function with the following properties :

a/ in each subinterval $I_i \equiv [x_i, x_{i+1}]$, $i = 1, \ldots, N-1$, $S(x)$ is given by some polynomial of degree five or less.

b/ $S(x) \in C^4[x_1, x_N]$

For given x_i, the QSF-S form a $(N+4)$-dimensional vector space. An interpolating QSF (i.e. with given $S(x_i)$), may thus be made unique when four well-chosen boundary conditions are added. In this algorithm the following conditions are provided :

1. Type I : S^{I} and S^{II} are given
2. Type II : $S^{III} = S^{IV} = 0$
3. Type III : S^{II} given, $S^{IV} = 0$

If boundary conditions of the same type are used at both extremities, one gets the same set of QSF-S as considered by SPÄTH in his algorithm [3]. But here, the choice is made independently at each end-

(*) E. H. Mund, Service de Métrologie Nucléaire, Université Libre de Bruxelles, B-1050 Bruxelles (Belgium) and Faculté des Sciences Appliquées, Université Catholique de Louvain, B-1348 Louvain-La-Neuve (Belgium), Chargé de Recherches au FNRS, Bruxelles (Belgium). P. Hallet, Centre d'Informatique Générale - KDL, BP 297 Lubumbashi, Republic of Zaïre. J. P. Hennart, Departamento de Ingenieria Nuclear, Instituto Politecnico Nacional, Unidad de Zacatenco, Mexico 14. D. F. (Mexico).

point. By the way, condition of type I at each boundary would give KERSHAW's D12 QSF [8], whereas condition of type II at x_1 and x_N corresponds to the standard natural QSF [5].
Consider a subinterval I_i with $i = 1, \ldots, N-1$. Let $Q_i(x)$ be the unique Hermite-type interpolation polynomial whose value, first and second derivatives at each end-point of I_i, are respectively $y_i^{(j)}$, $y_{i+1}^{(j)}$, $j = 0, 1, 2$.

$$Q_i(x) = \sum_{j=0}^{2} h_i^j [y_i^{(j)} s_j(\xi) + (-1)^j y_{i+1}^{(j)} s_j(1-\xi)] \tag{1}$$

with $h_i = x_{i+1} - x_i$

$$\xi = (x - x_i)/h_i$$

and $S_0(\xi) = (1-\xi)^3 (1 + 3\xi + 6\xi^2)$

$$S_1(\xi) = \xi (1-\xi)^3 (1 + 3\xi)$$

$$S_2(\xi) = \xi^2 (1-\xi)^3 / 2$$

Now consider the set of polynomials $Q_i(x)$, $i = 1, \ldots, N-1$ with given y_i and yet unknown y_i' and y_i''. They obviously form a piecewise polynomial function $Q(x)$ of class $C^2[x_1, x_N]$. Q will moreover be a QSF if both third and fourth derivatives are continuous across each inner knot. These $2(N-2)$ conditions, along with four extra boundary conditions chosen among types I, II or III given hereabove, define uniquely the unknowns y_i' and y_i'' as the solutions of an algebraic system of linear equations.
In the general case, the system may be written

$$M \underline{Z} = \underline{k} \tag{2}$$

with

$$M = \begin{bmatrix} A_1 & -B_1 & & & 0 \\ -B_1^T & A_2 & -B_2 & & \\ & \ddots & \ddots & \ddots & \\ & & -B_{N-2}^T & A_{N-1} & -B_{N-1} \\ 0 & & & -B_{N-1}^T & A_N \end{bmatrix}$$

$$\underline{Z} = (\underline{Z}_1, \underline{Z}_2, \ldots, \underline{Z}_N)^T$$

and

$$\underline{k} = (\underline{k}_1, \underline{k}_2, \ldots, \underline{k}_N)^T$$

In these notations, A_i, B_i represent the following 2×2 matrices :

$$A_i = \begin{bmatrix} 64(g_{i-1}^3 + g_i^3) & 12\alpha(g_{i-1}^2 - g_i^2) g_i \\ 12\alpha(g_{i-1}^2 - g_i^2) g_i & 3\alpha^2 (g_{i-1} + g_i) g_i^2 \end{bmatrix}$$

$$B_i = \begin{bmatrix} -56\, g_i^3 & -8\alpha\, g_i^2 g_{i+1} \\ 8\alpha\, g_i^3 & \alpha^2\, g_i^2\, g_{i+1} \end{bmatrix}$$

with $g_i = h_i^{-1}$

$$g_0 = g_N = 0 .$$

Whereas $\underline{Z}_i$ and $\underline{k}_i$ are 2-vectors given by

$$\underline{Z}_i = (-y_i',\ y_i''/\alpha\, g_i)^T$$

$$\underline{k}_i = (-120\, g_i^4 (y_{i+1} - y_i) - 120 g_{i-1}^4 (y_i - y_{i-1}),$$

$$20\alpha g_i^4 (y_{i+1} - y_i) - 20\alpha g_{i-1}^3 g_i (y_i - y_{i-1}))^T$$

First of all, let us point out that by performing a slight modification of the original unknowns, all the submatrices A_i, B_i may be rendered homogeneous in the power of the partition size. Moreover a parameter α has been introduced and chosen in such a way as to minimize $\|A_i^{-1}\| (\|B_{i-1}\| + \|B_i\|)$ where $\| \cdot \|$ is any matrix norm, e. g. the spectral norm. According to VARAH [6], this should improve the numerical stability of the algorithm chosen, namely the following extension of Cholesky's method [7]:

1. Forward elimination :

$$H_m = (A_m - B_{m-1}^T H_{m-1})^{-1} B_m$$

$$\underline{P}_m = (A_m - B_{m-1}^T H_{m-1})^{-1} (\underline{k}_m + B_{m-1}^T \underline{P}_{m-1})$$

$$H_1 = A_1^{-1} B_1$$

$$\underline{P}_1 = A_1^{-1} \underline{k}_1$$

2. Backward substitution :

$$\underline{Z}_N = \underline{P}_N$$

$$\underline{Z}_m = \underline{P}_m + H_m \underline{Z}_{m+1} \qquad m = N-1, N-2, \ldots, 1$$

In the particular case of an homogeneous partition, the best α turned out to be $(64/3)^{1/2}$: this value

was introduced into the algorithm for the general case.
If a boundary condition of type I is chosen at x_1 and (or) x_N, the corresponding "unknowns" are automatically skipped and a system of reduced order is solved along similar lines. For a boundary condition of type III, for instance at x_1, A_1, B_1 and $\underline{k}_1$, instead of being of the general form explicited hereabove, become :

$$A_1 = \begin{bmatrix} 64\, g_1^3 & -12\, \alpha\, g_1^3 \\ 0 & 3\, \alpha^2 g_1^3 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} -56\, g_1^3 & -8\, \alpha\, g_1^2\, g_2 \\ 0 & 0 \end{bmatrix}$$

$$\underline{k}_1 = (-120\, g_1^4 (y_2 - y_1),\ 3\alpha\, g_1^2\, y_1'')$$

Similar expressions would be obtained at x_N.

At this point, it is worthwhile to mention that in case of natural spline boundary conditions, the same general set of algebraic equations (2) would have been obtained using the extremal properties of the spline functions [9]. Indeed, it is easy to show that by introduction of $Q_i(x)$ $i = 1, \ldots, N$ into the quadratic functional

$$I = \int_{x_1}^{x_N} (\sigma^{(3)}(x))^2\, dx \qquad \sigma(x) \in H^3[X_1, X_N]$$

and minimization of I, with respect to the unknown first and second derivatives in $Q_i(x)$, one obtains exactly the same symmetric, blocktridiagonal linear system. $H^3[x_1, x_N]$ is the set of functions which, on $[x_1, x_N]$ have an absolutely continuous second derivative and whose third derivative is square integrable.
Finally, having solved the linear system, values of the spline function and its successive derivatives may be determined with the aid of expression (1).

3. DESCRIPTION OF THE ROUTINES

The package is made of two FORTRAN subroutines: INSPL5 and SPLIN5. The subroutine INSPL5 computes the elements of the matrix M and the vector $\underline{k}$ and solves the system of equations (2). The evaluation of the elements and the forward elimination are performed simultaneously to reduce core storage and improve coding efficiency.

The user provides :

N : Number of data points
X(N) : vector giving their abscissa : $X(I) = x_i \quad i = 1, \ldots, N$
IND1, INDN : boundary condition switches at x_1 and x_N respectively. For both switches, one has the correspondence:
Type I : +1
Type II : 0
Type III : -1
H : auxiliary vector with at least 6N - 3 components
CF(3, N) : coefficients matrix. The first row *must* contain the given ordinates : $CF(1, I) = y_i \quad i = 1, \ldots, N$

The subroutine returns the values of the first two derivatives of the interpolating spline into the other rows of CF :

$CF(2, I) = y_i' \qquad i = 1, \ldots, N$
$CF(3, I) = y_i'' \qquad i = 1, \ldots, N$

When boundary conditions of type I or III are used, at one or both end-points, the corresponding values of the derivatives must, of course, be put into CF(2, 1), CF(3, 1), CF (2, N) or CF(3, N).
The subroutine SPLIN5 provides values of the interpolating spline and of its derivatives at any point.
The user provides :

XX : abscissa where values of the function and its derivatives are required
CF(3, N) : coefficients of the matrix described above
N : number of points in the partition

The subroutine returns :

F(3) : vector with three components which contains the required values of the function and its derivatives.

4. NUMERICAL TESTS AND RESULTS

The algorithm has been applied to various elementary analytical functions (e.g. e^x, sin x, $1/(1+x^2)$...) on the interval [0,0.98] with 2^n $(n = 1, \ldots, 6)$ subintervals. Analytical simulations of sharp, multilevel resonance cross sections such as those encountered in nuclear physics, were also considered in these tests. Three values of δ, the ratio of largest to smallest subinterval in the partition of [0,0.98], were chosen in order to include some cases with strongly non-homogeneous partition sizes, namely : $\delta = 1$, 19 and 199. An extensive study of convergence orders was

made for both natural and D12 spline functions. As predicted by theory [10], the absolute error for the i^{th} derivative ($i = 0, 1, 2$) decreased as the $(3-i)^{th}$ power of h (largest subinterval width) for natural splines, and as its $(6-i)^{th}$ power for D12 splines. This was fairly well corroborated by our numerical tests, a sample of which has been displayed in Table 1.

A double precision version of our algorithm was run in the same conditions as hereabove, but with natural splines only. The single precision version was then applied, with the same knots to the resulting spline, in order to isolate the "finite accuracy errors" due to machine limitations. Indeed, in this case, no interpolation error appears since both the interpolating and the interpolated functions ought to be equal. For $\delta = 1$, the interpolation accuracy, as measured by the maximum value of the absolute error, was the machine accuracy (i. e. 14 digits on a CDC 6500 computer). This error grew with increasing values of δ, but in no systematic way.

ACKNOWLEDGEMENTS

The authors would like to thank Professor P. J. LAURENT (Université Scientifique et Médicale de Grenoble) for many fruitful discussions about this work.

REFERENCES

1. C. Carasso, P. J. Laurent,
"On the numerical construction and the practical use of interpolating spline functions",
Proceedings IFIP Congress (1968) pp. 86-89 Edinburgh.

2. J. Herriot, C. Reinsch,
"Algol 60 procedures for the calculation of interpolating natural spline functions",
Comp. Science Depart. Stanford Univ. (1971) STAN-CS-200.

3. H. Späth,
"Die numerische Berechnung von interpolierende Spline Funktionen mit Blockunterrelaxation",
KFK 1132 (1972) Karlsruhe.

4. H. Späth,
"The numerical calculation of quintic splines by block-underrelaxation",
Computing 7 (1971) 75-82.

5. T. N. E. Greville,
"Introduction to spline functions",
in 'Theory and applications of spline functions' (1969) T. N. E. Greville Ed. Academic Press, London.

6. J. M. Varah,
"On the solution of block-tridiagonal systems arising from certain finite-difference equations",
Maths. of Comput. 26 (1972) 859-868.

7. E. L. Wachspress,
"Iterative solution of elliptic systems",
(1966) Prentice Hall Inc., Englewood Cliffs.

8. D. Kershaw,
"A note on the convergence of interpolating cubic splines",
SIAM J. Numer. Anal. 8 (1971) 67-74.

9. P. J. Laurent,
"Approximation et optimisation",
(1972) Hermann Paris.

10. C. A. Hall,
"On error bounds for spline interpolation",
J. of Approx. Theory 1 (1968) 209-218.

11. H. Späth,
"Spline-Algorithmen zur Konstruktion glatter Kurven und Flächen",
R. Oldenbourg Verlag, München (1973).

TABLE I. Experimental Convergence Orders of natural and D12 Quintic Functions for some elementary Analytical Functions

Nat. S		$y(x) = \exp(x)$		$y'(x)$		$y''(x)$	
δ	N	$\|\epsilon\|_{L^\infty}$	α	$\|\epsilon\|_{L^\infty}$	α	$\|\epsilon\|_{L^\infty}$	α
1	3	1.34 (-2)		1.50 (-1)		9.99 (-1)	
	5	1.06 (-3)	3.66	2.69 (-2)	2.48	4.13 (-1)	1.28
	9	1.31 (-4)	3.02	6.70 (-3)	2.00	2.08 (-1)	0.99
	17	1.67 (-5)	2.97	1.71 (-3)	1.97	1.06 (-1)	0.98
	33	2.11 (-6)	2.98	4.32 (-4)	1.98	5.32 (-2)	0.99
	65	2.66 (-7)	2.99	1.08 (-4)	1.99	2.67 (-2)	0.99
199	3	3.55 (-2)		2.67 (-1)		1.24 (0)	
	5	3.26 (-3)	3.45	5.49 (-2)	2.28	5.71 (-1)	1.11
	9	4.21 (-4)	2.95	1.43 (-2)	1.95	2.96 (-1)	0.95
	17	5.47 (-5)	2.94	3.68 (-3)	1.95	1.52 (-1)	0.96
	33	6.98 (-6)	2.97	9.36 (-4)	1.98	7.68 (-2)	0.98
	65	8.80 (-7)	2.99	2.36 (-4)	1.99	3.86 (-2)	0.99

D12 S		$y(x) = 1/(1+x^2)$		$y'(x)$		$y''(x)$	
δ	N	$\|\epsilon\|_{L^\infty}$	α	$\|\epsilon\|_{L^\infty}$	α	$\|\epsilon\|_{L^\infty}$	α
1	3	7.16 (-5)		5.73 (-4)		7.31 (-3)	
	5	2.10 (-5)	1.77	2.63 (-4)	1.12	4.18 (-3)	0.81
	9	1.48 (-7)	7.15	4.31 (-6)	5.93	1.45 (-4)	4.85
	17	3.16 (-9)	5.55	1.48 (-7)	4.87	1.27 (-5)	3.51
	33	5.31 (-11)	5.89	5.02 (-9)	4.88	8.94 (-7)	3.83
	65	8.46 (-13)	5.97	1.60 (-10)	4.97	5.74 (-8)	3.96
199	3	4.18 (-3)		1.51 (-2)		1.11 (-1)	
	5	3.99 (-5)	6.71	3.06 (-4)	5.62	5.62 (-3)	4.31
	9	2.68 (-6)	3.90	3.70 (-5)	3.05	1.03 (-3)	2.44
	17	6.87 (-8)	5.29	1.84 (-6)	4.33	9.81 (-5)	3.40
	33	1.66 (-9)	5.37	8.65 (-8)	4.41	9.57 (-6)	3.36
	65	3.27 (-11)	5.67	3.37 (-9)	4.68	7.50 (-7)	3.67

N : number of points in the partition of [0,0.98]

δ : $\max_i h_i / \min_i h_i$; $i = 1, \ldots, N-1$

$\|\epsilon\|_{L^\infty}$: discrete L^∞ norm of the interpolation error, evaluated with 10^3 points covering [0,0.98]

$\alpha^{(i)}$: $\log(\|\epsilon\|^{(i)}_{L^\infty} / \|\epsilon\|^{(i-1)}_{L^\infty}) / \log(h^{(i)}/h^{(i-1)})$ at any level (i) with $2^i + 1$ interpolation points, $i = 2, ..$

```
      SUBROUTINE INSPL5(N,X,IND1,INDN,CF,H)
C
C   CALCULATION OF THE PARAMETERS OF AN INTERPOLATING
C   QUINTIC SPLINE
C
C   INPUT
C    N                  NUMBER OF INTERPOLATION POINTS
C    X(N)               VECTOR OF ABSCISSAE
C    CF(1,N)            ORDINATES
C    IND1,INDN          BOUNDARY CONDITION SWITCHES AT X1 AND XN
C                       FOR BOTH SWITCHES ONE HAS THE CORRESPONDENCE
C                        = 1    TYPE 1
C                        = 0    TYPE 2      (  CFR. TEXT FOR MORE DETAILS
C                        =-1    TYPE 3
C   OUTPUT
C    CF(2,N)            FIRST DERIVATIVES
C    CF(3,N)            SECOND DERIVATIVES
C
C    H(6*N-3)           AUXILIARY VECTOR
C
      DIMENSION X(1),CF(3,1),H(1)
C
      AS=64./3.
      A=SQRT(AS)
      P1=0.0
      P2=0.0
      P3=0.0
      Q1=0.0
      Q2=0.0
      SF1=0.0
      SF2=0.0
      IM=0
C
C   FORWARD ELIMINATION PROCEDURE FOR CHOLESKYS ALGORITHM
C   APPLIED TO BLOCK-TRIDIAGONAL SYSTEMS OF ALGEBRAIC
C                    EQUATIONS.
C
      I=1
        GO TO 15
    5   DI1=DS1
        DI2=DS3
        DI3=DS2
        DI4=DS4
   10   SF1=FP4
        SF2=FP3*Q1*A
   15   DP1=+64.0*P3
        DP2= 12.0*P2*Q1*A
        DP3=     DP2
        DP4=  3.0*P1*Q2*AS
        IF (I.LT.N) GO TO 30
C
C    BOUNDARY CONDITION AT  XN
C
        IF (INDN) 20,50,25
   20   DP2=0.0
        DP4=1.0
        DI2=0.0
```

```
      DI4=0.0
      SF2=CF(3,N)*(X(N)-X(N-1))/A
      GO TO 50
   25 DP1=1.0
      DP2=0.0
      DP3=0.0
      DP4=1.0
      DI1=0.0
      DI2=0.0
      DI3=0.0
      DI4=0.0
      SF1=-CF(2,N)
      SF2= CF(3,N)*(X(N)-X(N-1))/A
      GO TO 50
C
   30 P1=1.0/(X(I+1)-X(I))
      P2=P1*P1
      P3=P1*P2
      Q1=P1
      Q2=P2
      IF (I.GE.(N-1)) GO TO 35
      Q1=1.0/(X(I+2)-X(I+1))
      Q2=Q1*Q1
   35 FP =CF(1,I)-CF(1,I+1)
      FP3=20.0*P3*FP
      FP4= 6.0*P1*FP3
      DP1=DP1+64.0*P3
      DP2=DP2-12.0*P3*A
      DP3=       DP2
      DP4=DP4+3.0*P3*AS
      DS1=-56.0*P3
      DS2=  8.0*P3*A
      DS3= -8.0*P2*Q1*A
      DS4=      P2*Q1*AS
      SF1=SF1+FP4
      SF2=SF2-FP3*P1*A
      IF (I.GT.1) GO TO 50
C
      DI1=DS1
      DI2=DS3
      DI3=DS2
      DI4=DS4
C
C   BOUNDARY CONDITION AT  X1
C
      IF (IND1) 40,55,45
   40 DP2=0.0
      DP4=1.0
      DS2=0.0
      DS4=0.0
      SF2=CF(3,1)*(X(2)-X(1))/A
      GO TO 55
   45 DP1=1.0
      DP2=0.0
      DP3=0.0
      DP4=1.0
      DS1=0.0
```

```
      DS2=0.0
      DS3=0.0
      DS4=0.0
      SF1=-CF(2,1)
      SF2= CF(3,1)*(X(2)-X(1))/A
      GO TO 55
C
   50 DP1=DP1-DI1*H(IM-3)-DI3*H(IM-2)
      DP2=DP2-DI2*H(IM-3)-DI4*H(IM-2)
      DP3=DP3-DI1*H(IM-1)-DI3*H(IM  )
      DP4=DP4-DI2*H(IM-1)-DI4*H(IM  )
   55 DET=DP1*DP4-DP2*DP3
      IF (I.EQ.1) GO TO 60
      SF1=SF1+DI1*H(IM-5)+DI3*H(IM-4)
      SF2=SF2+DI2*H(IM-5)+DI4*H(IM-4)
   60 H(IM+1)=(+DP4*SF1-DP3*SF2)/DET
      H(IM+2)=(-DP2*SF1+DP1*SF2)/DET
      IF (I.GE.N) GO TO 65
      H(IM+3)=(+DP4*DS1-DP3*DS2)/DET
      H(IM+4)=(-DP2*DS1+DP1*DS2)/DET
      H(IM+5)=(+DP4*DS3-DP3*DS4)/DET
      H(IM+6)=(-DP2*DS3+DP1*DS4)/DET
      IM=IM+6
      I=I+1
    IF (I-2) 5,10,5
C
C  BACKWARD SUBSTITUTION AND SOLUTION OF THE ALGEBRAIC SYSTEM
C
   65 CF(2,N)=-H(IM+1)
      DPD1   = H(IM+2)*A
      CF(3,N)= DPD1/(X(N)-X(N-1))
      IM=IM-6
C
      DO 70 I=2,N
        K=N+1-I
        CF(2,K)=-H(IM+1)+H(IM+3)*CF(2,K+1)-H(IM+5)*DPD1/A
        DPD2   = H(IM+2)*A-H(IM+4)*CF(2,K+1)*A+H(IM+6)*DPD1
        CF(3,K)= DPD2/(X(K+1)-X(K))
        DPD1   = DPD2
        IM=IM-6
   70 CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE SPLIN5 (N,X,CF,XX,F)
C
C  CALCULATION OF THE VALUES OF AN INTERPOLATING QUINTIC
C  SPLINE AND OF ITS FIRST AND SECOND DERIVATIVES AT
C  ANY POINT XX
C
C  INPUT
C   N              NUMBER OF INTERPOLATION POINTS
C   X  (N)         ABSCISSAE
C   CF(3,N)        ORDINATES AND FIRST TWO DERIVATIVES
C   XX             ABSCISSA WHERE VALUES OF THE FUNCTION AND
C                  ITS DERIVATIVES ARE COMPUTED
C  OUTPUT
C   F(1)           VALUE OF THE INTERPOLATING FUNCTION
C   F(2)           VALUE OF ITS FIRST DERIVATIVE
C   F(3)           VALUE OF ITS SECOND DERIVATIVE
C
      DIMENSION F(3),X(1),CF(3,1)
C
      I=1
      XN=XX-X(1)
      IF (XN) 5,20,10
    5 I=2
      GO TO 25
C
   10 DO 15 I=2,N
        XN=XX-X(I)
        IF (XN) 25,20,15
   15 CONTINUE
      I=N
      GO TO 25
C
   20 F(1)=CF(1,I)
      F(2)=CF(2,I)
      F(3)=CF(3,I)
      GO TO 30
C
   25 CC=CF(1,I-1)-CF(1,I)
      H=X(I)-X(I-1)
      XN=XX-X(I-1)
      XR1=XN/H
      XR2=XR1*XR1
      XR3=XR2*XR1
      Y=CF(1,I-1)+CC*XR3*(-10.+XR1*(+15.- 6.*XR1))
      W=CF(2,I-1)*XR1
     1     +CF(2,I-1)*XR3*(- 6.+XR1*(+ 8.- 3.*XR1))
     2     -CF(2,I  )*XR3*(+ 4.+XR1*(- 7.+ 3.*XR1))
      U=CF(3,I-1)*XR2
     1     +CF(3,I-1)*XR3*(- 3.+XR1*(+ 3.-    XR1))
     2     +CF(3,I  )*XR3*(+ 1.+XR1*(- 2.+    XR1))
      F(1)=Y+H*(W+H*U/2.)
      Y=       30.*CC*XR2*(- 1.+XR1*(+ 2.-    XR1))
      W=CF(2,I-1)
     1     +CF(2,I-1)*XR2*(-18.+XR1*(+32.-15.*XR1))
     2     -CF(2,I  )*XR2*(+12.+XR1*(-28.+15.*XR1))
      U=CF(3,I-1)*XR1*2.
     1     +CF(3,I-1)*XR2*(- 9.+XR1*(+12.- 5.*XR1))
```

```
     2     +CF(3,I  )*XR2*(+ 3.+XR1*(- 8.+ 5.*XR1))
      F(2)=Y/H+W+H*U/2.
      XN=10.*XR1
      Y=              CC*XN*(- 1.+XR1*(+ 3.-2.*XR1))
      W= CF(2,I-1)*(-6.+XR1*(+16.-XN))
     1  -CF(2,I  )*(+4.+XR1*(-14.+XN))
      U=CF(3,I-1)
     1  +CF(3,I-1)*(-9.+XR1*(+18.-XN))*XR1
     2  +CF(3,I  )*(+3.+XR1*(-12.+XN))*XR1
      F(3)=(Y/H+XR1*W)*6./H+U
   30 RETURN
C
      END
```