PB-229 616

ALGOL 60 PROCEDURES FOR THE CALCULATION OF INTERPOLATING NATURAL QUINTIC SPLINE FUNCTIONS

John G. Herriot, et al

Stanford University

Prepared for:

National Science Foundation

January 1974

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151

Algol 60 Procedures for the Calculation of Interpolating Natural Quintic Spline Functions

by

John G. Herriot and Christian H. Reinsch */

Abstract

Three Algol 60 procedures are described for finding interpolating natural quintic spline functions. The first procedure treats the case of an arbitrary set of knots and the second procedure handles the case of equidistant knots. The third procedure finds the quintic natural spline of deficiency 2 when the values of both the function and its first derivative are given at the knots. These procedures are much faster than more general procedures, which find interpolating natural splines of degree $2m-1$, when used with $m = 3$.

*/ Leibnitz-Rechenzentrum der Bayerischen Akademie der Wissenschaften, 8 München 2, Germany.

This work was supported in part by the National Science Foundation under grant number GJ 29988X.

BIBLIOGRAPHIC DATA SHEET	1. Report No. STAN-CS-74-402	2.	PB 229 616
4. Title and Subtitle ALGOL 60 PROCEDURES FOR THE CALCULATION OF INTERPOLATING QUINTIC SPLINE FUNCTIONS.			5. Report Date JANUARY 1974 6.
7. Author(s) John G. Herriot and Christian H. Reinsch			8. Performing Organization Rept. No. STAN-CS-74-402
9. Performing Organization Name and Address Stanford University Computer Science Department Stanford, California 94305			10. Project/Task/Work Unit No. 11. Contract/Grant No. GJ 29988X
12. Sponsoring Organization Name and Address National Science Foundation 1800 G Street, N.W. Washington, D. C. 20550			13. Type of Report & Period Covered technical, 14.
15. Supplementary Notes			

16. Abstracts

Three Algol 60 procedures are described for finding interpolating natural quintic spline functions. The first procedure treats the case of an arbitrary set of knots and the second procedure handles the case of equidistant knots. The third procedure finds the quintic natural spline of deficiency 2 when the values of both the function and its first derivative are given at the knots. These procedures are . . much faster than more general procedures, which find interpolating natural splines of degree 2m-1, when used with m=3.

17. Key Words and Document Analysis. 17a. Descriptors

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 44
Approved for public release; distribution unlimited	20. Security Class (This Page) UNCLASSIFIED	22. Price 3.25/1.45

FORM NTIS-35 (REV. 3-72) USCOMM-DC 14952-P72

Table of Contents

1. Introduction

Algorithm 472 [5] provided a set of Algol 60 procedures for the calculation of interpolating natural spline functions of degree $2m-1$. Since the case of a cubic natural spline is of frequent occurrence, a procedure for this special case was also included. The special procedure is very much faster than the general procedure when used with $m = 2$ to produce the same results.

The next most useful case is that of the quintic natural spline which can, of course, be obtained by using the general procedures of Algorithm 472 with $m = 3$. However, the calculations can be greatly simplified by considering this special case as described below. The procedure, QUINAT, which is given here, takes advantage of these simplifications and is much faster than the general procedure with $m = 3$. An even faster procedure QUINEQ treats the case of equidistant knots. Also included in the present set of procedures is the procedure QUINDF which treats the case in which the first derivative as well as the functional value is given at each of the knots.

2. Formulation of the Problem and Description of the Procedures

Let (x_i, y_i), $i = N1, N1+1, \ldots, N2$ be a set of data points where it is assumed that $x_{N1} < x_{N1+1} < \cdots < x_{N2}$. The interpolating quintic natural spline function $S(x)$ with the knots $x_{N1}, \ldots, x_{N2}$ has the following properties: (i) $S(x)$ is a polynomial of degree 5 in each interval (x_i, x_{i+1}), $i = N1, \ldots, N2-1$. (ii) $S(x)$ and its derivatives $S'(x)$, $S''(x)$, $S'''(x)$ and $S''''(x)$ are continuous in (x_{N1}, x_{N2}). (iii) $S'''(x_{N1}) = S'''(x_{N2}) = S''''(x_{N1}) = S''''(x_{N2}) = 0$. (iv) $S(x_i) = y_i$,

$i = N1,\ldots,N2$. It is known that if $N2 > N1+1$, then there is a unique quintic natural spline function which has the properties (i) - (iv). (See e.g. Greville [3,4].) This spline function can be represented in the form

$$(2.1) \qquad S(x) = y_i + B_i t + C_i t^2 + D_i t^3 + E_i t^4 + F_i t^5$$

with $t = x - x_i$ for $x_i \le x < x_{i+1}$, $i = N1,\ldots,N2-1$.

If at one or more of the knots x_i , one also specifies the derivative y_i' , thus requiring $S'(x_i) = y_i'$ then the condition that $S''''(x)$ be continuous at the knot x_i need not hold. If the second derivative y_i'' is also specified thus requiring $S''(x_i) = y_i''$ then $S'''(x)$ also need not be continuous at x_i . If the values of the derivative y_i' are specified at all the knots x_i then $S''''(x)$ need not be continuous at the knots and also $S''''(x_{N1})$ and $S''''(x_{N2})$ need not be zero. Such a spline is said to be of deficiency 2 . It is not of interest to specify the first and second derivatives at each knot because in this case the quintic polynomial is completely determined in each interval independently of all the other intervals.

The procedure QUINAT computes the coefficients B_i , C_i , D_i , F_i , F_i of the quintic natural spline represented as in equation (2.1) for an arbitrary set of data points (x_i,y_i) as specified above. The procedure QUINEQ treats the case of equidistant knots x_i . If the knots are known to be equidistant QUINEQ should be used as it is much faster than QUINAT. In this case it is not necessary to specify the values of x_i . The representation (2.1) is still used but now $t = (x-x_i)/h$ where $h = x_{i+1} - x_i$, the constant spacing of the knots.

QUINAT can also be used for the case in which the first and second derivatives are specified at an arbitrary set of the knots. To specify the value of the first derivative y'_j at x_j one increases the number of knots by one, setting $x_{j+1} = x_j$ (and renumbering the knots and values to the right). Then one chooses $y_{j+1} = y'_j$. Then the spline function computed by QUINAT will have the property $S(x_j) = y_j$, $S'(x_j) = y_{j+1}$. To specify also the second derivative, note that if $x_j = x_{j+1} = x_{j+2}$ then $S(x_j) = y_j$, $S'(x_j) = y_{j+1}$, $S''(x_j) = y_{j+2}$. For further details see Section 3.2.

The procedure QUINDF computes the coefficients of the quintic natural spline of deficiency 2 when the values of the function y_i and the values of the first derivative y'_i are given at each knot. QUINDF is much faster than QUINAT.

3. Procedure QUINAT

As in the general case of Algorithm 472 [5] the calculation of the coefficients of the spline function is carried out in a numerically stable manner following a method described by Anselone and Laurent [1]. The basic ideas on which the method is based were given earlier by Schoenberg [6]. The method is specialized to the case of the quintic natural spline and uses minimum support B-splines [2,4] of degree 2 to form a basis for the class of third derivatives of the quintic natural splines. Instead of specializing the formulas of Algorithm 472 [5] by setting $m = 3$, we derive the necessary formulas directly and indeed choose a different numbering and a different normalization for the B-splines.

3.1 Distinct Knots

We assume that the knots are strictly monotone increasing. In order to simplify the notation we shall choose $N1 = 0$ and let $N2 = n$ so that the data points are denoted by (x_i, y_i), $i = 0,1,\ldots,n$. This is merely a translation of the subscripts and involves no loss of generality.

We denote the B-spline of degree 2 by $M_i(x)$ and require that it vanish outside the interval (x_{i-1}, x_{i+2}). $M_i(x)$ and $M_i'(x)$ must be continuous at each of the knots. Let $h_i = x_{i+1} - x_i$, $t = x - x_{i-1}$, $u = x - x_i$, $v = x - x_{i+1}$. Then we must have

$$
\begin{aligned}
M_i(x) &= At^2 && x_{i-1} \le x < x_i \\
&= B + Cu - Du^2 && x_i \le x < x_{i+1} \\
&= E(v - h_{i+1})^2 && x_{i+1} \le x < x_{i+2} \quad .
\end{aligned}
\tag{3.1}
$$

Hence also

$$
\begin{aligned}
M_i'(x) &= 2At & M_i''(x) &= 2A && x_{i-1} \le x < x_i \\
&= C - 2Du & &= -2D && x_i \le x < x_{i+1} \\
&= 2E(v - h_{i+1}) & &= 2E && x_{i+1} \le x < x_{i+2} \quad .
\end{aligned}
\tag{3.2}
$$

Imposing the continuity requirements at x_i, x_{i+1} yields

$$
\begin{aligned}
Ah_{i-1}^2 &= B & B + Ch_i - Dh_i^2 &= Eh_{i+1}^2 \\
2Ah_{i-1} &= C & C - 2Dh_i &= -2Eh_{i+1} \quad .
\end{aligned}
$$

Hence up to a common factor

$$
\begin{aligned}
A &= \frac{1}{h_{i-1}(h_{i-1} + h_i)} \qquad B = \frac{h_{i-1}}{h_{i-1} + h_i} \qquad C = \frac{2}{h_{i-1} + h_i} \\
D &= \frac{h_{i-1} + 2h_i + h_{i+1}}{(h_{i-1} + h_i)h_i(h_i + h_{i+1})} \qquad E = \frac{1}{h_{i+1}(h_i + h_{i+1})} \quad .
\end{aligned}
\tag{3.3}
$$

Choosing these values of the coefficients we find that

$$\int_{-\infty}^{\infty} M_i(x) f'''(x)dx = \int_{-\infty}^{\infty} M_i''(x) f'(x)dx$$

(3.4) $$= 2A \int_{x_{i-1}}^{x_i} f'(x)dx - 2D \int_{x_i}^{x_{i+1}} f'(x)dx + 2E \int_{x_{i+1}}^{x_{i+2}} f'(x)dx$$

$$= 2(f(x_i, x_{i+1}, x_{i+2}) - f(x_{i-1}, x_i, x_{i+1}))$$

using the usual notation for divided differences. This is a very convenient choice of normalization of the $M_i(x)$.

Next we need the inner products of the basis B-splines. Since each $M_i(x)$ is different from zero in only 3 consecutive intervals it is clear that

(3.5) $$\int_{-\infty}^{\infty} M_i(x) M_j(x)dx = 0 \quad \text{if} \quad |i-j| > 2 \quad .$$

If we use the representations of $M_i(x)$ in (3.1) we obtain

$$30 \int_{-\infty}^{\infty} [M_i(x)]^2 dx = 30 \int_0^{h_{i-1}} (At^2)^2 dt + 30 \int_0^{h_i} (B + Cu - Du^2)^2 du + 30 \int_0^{h_{i+1}} E(v - h_{i+1})^2 dv \ .$$

If we substitute the constants from (3.3) and carry out the integrations we obtain

(3.6) $$30 \int_{-\infty}^{\infty} [M_i(x)]^2 dx = T_1 + T_2 + T_3$$

where

$$T_1 = \frac{6h_{i-1}^3}{(h_{i-1} + h_i)^2} \qquad T_3 = \frac{6h_{i+1}^3}{(h_i + h_{i+1})^2}$$

T_2 =

$$h_i \frac{30h_{i-1}^2 h_{i+1}^2 + (h_{i-1} + h_{i+1}) h_i (40h_{i-1} h_{i+1} + 14h_i^2) + h_i^2 [16(h_{i-1}^2 + h_{i+1}^2) + 42h_{i-1} h_{i+1} + 4h_i^2]}{(h_{i-1} + h_i)^2 (h_i + h_{i+1})^2} \ .$$

In the same way we find that

$$(3.7) \qquad 30 \int_{-\infty}^{\infty} M_i(x) M_{i+1}(x) dx = T_4 + T_5$$

where

$$T_4 = h_i^2 \frac{h_{i-1}(h_i + h_{i+1}) + 3(h_{i-1} + h_i)(h_i + 3h_{i+1})}{(h_{i-1} + h_i)(h_i + h_{i+1})^2}$$

$$T_5 = h_{i+1}^2 \frac{h_{i+2}(h_i + h_{i+1}) + 3(h_{i+1} + h_{i+2})(3h_i + h_{i+1})}{(h_i + h_{i+1})^2 (h_{i+1} + h_{i+2})}$$

and

$$(3.8) \qquad 30 \int_{-\infty}^{\infty} M_i(x) M_{i+2}(x) dx = \frac{h_{i+1}^3}{(h_i + h_{i+1})(h_{i+1} + h_{i+2})} \quad .$$

Note that all terms in these expressions are positive and consequently no cancellations can occur.

Now the third derivative $S'''(x)$ will vanish outside the interval (x_0, x_n) and it can be expressed in terms of the basis functions:

$$(3.9) \qquad S'''(x) = \sum_{j=1}^{n-2} 60\gamma_j M_j(x) \quad .$$

If we multiply equation (3.9) by $\frac{1}{2} M_i(x)$, $i = 1,2,\ldots,n-2$ and integrate, we obtain a well-conditioned system of linear equations for the determination of the γ_j :

$$(3.10) \qquad \sum_{j=1}^{n-2} \left(30 \int_{-\infty}^{\infty} M_i(x) M_j(x) dx\right) \gamma_j = \frac{1}{2} \int_{-\infty}^{\infty} M_i(x) S'''(x) dx \ , \quad i = 1,2,\ldots,n-2 \ .$$

If we use (3.4) and (3.5) we see that (3.10) is a pentadiagonal system of linear equations and can be written in the form

$$
(3.11)\quad
\begin{bmatrix}
d_1 & e_1 & f_1 & & & & \\
e_1 & d_2 & e_2 & f_2 & & \bigcirc & \\
f_1 & e_2 & d_3 & e_3 & f_3 & & \\
 & f_2 & e_3 & d_4 & e_4 & f_4 & \\
 & & \ddots & \ddots & \ddots & \ddots & \ddots \\
 & \bigcirc & & f_{n-5} & e_{n-4} & d_{n-3} & e_{n-3} \\
 & & & & f_{n-4} & e_{n-3} & d_{n-2}
\end{bmatrix}
\begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \gamma_4 \\ \vdots \\ \gamma_{n-3} \\ \gamma_{n-2} \end{bmatrix}
=
\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ \vdots \\ c_{n-3} \\ c_{n-2} \end{bmatrix}
$$

where

$$d_i = 30 \int_{-\infty}^{\infty} [M_i(x)]^2 dx \quad , \qquad i = 1,2,\ldots,n-2$$

$$e_i = 30 \int_{-\infty}^{\infty} M_i(x) M_{i+1}(x) dx \quad , \qquad i = 1,2,\ldots,n-3$$

$$f_i = 30 \int_{-\infty}^{\infty} M_i(x) M_{i+2}(x) dx \quad , \qquad i = 1,2,\ldots,n-4$$

$$c_i = y_{i,i+1,i+2} - y_{i-1,i,i+1} \quad , \qquad i = 1,2,\ldots,n-2 \quad .$$

The values of d_i , e_i , f_i are obtained from (3.6), (3.7), and (3.8).

This system of equations can be solved for the γ_j by using Gaussian elimination to reduce the matrix of coefficients to upper triangular form. This is conveniently done by annihilating the elements below the principal diagonal a row at a time. Note that the f_i are never changed. We use d_i' , e_i' and d_i'' , e_i'' to denote the new values of d_i , e_i as they are formed by the annihilation of f_{i-2} and e_{i-1}' in the i-th row. Before the elimination process operates on the i-th row, the form of the system is:

$$
\begin{bmatrix}
d''_1 & e''_1 & f_1 & & & & & \bigcirc \\
 & \ddots & \ddots & \ddots & & & & \\
 & & d''_{i-2} & e''_{i-2} & f_{i-2} & & & \\
 & & 0 & d''_{i-1} & e''_{i-1} & f_{i-1} & & \\
 & & f_{i-2} & e_{i-1} & d_i & e_i & f_i & \\
 \bigcirc & & & \ddots & \ddots & \ddots & \ddots & \ddots
\end{bmatrix}
\begin{bmatrix}
\gamma_1 \\ \vdots \\ \gamma_{i-2} \\ \gamma_{i-1} \\ \gamma_i \\ \vdots
\end{bmatrix}
=
\begin{bmatrix}
c''_1 \\ \vdots \\ c''_{i-2} \\ c''_{i-1} \\ c_i \\ \vdots
\end{bmatrix}
$$

The equations for the annihilation of f_{i-2} are

$$p_i = f_{i-2} / d''_{i-2}$$

$$e'_{i-1} = e_{i-1} - p_i e''_{i-2}$$

$$d'_i = d_i - p_i f_{i-2}$$

$$c'_i = c_i - p_i c''_{i-2}$$

and those for the annihilation of e_{i-1} are

$$q_i = e'_{i-1} / d''_{i-1}$$

$$d''_i = d'_i - q_i e''_{i-1}$$

$$e''_i = e_i - q_i f_{i-1}$$

$$c''_i = c'_i - q_i c''_{i-1} \quad .$$

Note that e'_{i-1} need not be calculated because $e'_{i-1} = e''_{i-1}$. This follows by induction because if $e'_{i-1} = e''_{i-1}$ then

$$e'_i = e_i - \frac{f_{i-1}}{d''_{i-1}} e''_{i-1} = e''_i \quad .$$

Thus $q_i = e''_{i-1} / d''_{i-1}$. For the first step of the elimination (operating on the 2-nd row) the above formulas are valid if we choose $p_2 = 0$.

When the coefficients γ_j have been found, $S'''(x)$ is given by the equation (3.9). We want to find the coefficients of $S(x)$ as expressed in equation (2.1). Clearly

$$(3.12) \qquad D_i = S'''(x_i+0)/6 \ , \ E_i = S''''(x_i+0)/24 \ , \ F_i = S^{V}(x_i+0)/120 \ ,$$

$$i = 0,1,\ldots,n-1 \ .$$

But because $M_j(x)$ vanishes outside the interval (x_{j-1},x_{j+2}) , $S'''(x)$ can be represented in the interval $[x_i,x_{i+1})$ in the very simple form

$$(3.13) \qquad \frac{1}{60} S'''(x_i+t) = \gamma_{i-1}M_{i-1}(x_i+t) + \gamma_i M_i(x_i+t) + \gamma_{i+1}M_{i+1}(x_i+t)$$

with $0 \leq t < h_i$, $i = 2,3,\ldots,n-3$.

Then also

$$(3.14) \qquad \frac{1}{60} S''''(x_i+t) = \gamma_{i-1}M'_{i-1}(x_i+t) + \gamma_i M'_i(x_i+t) + \gamma_{i+1}M'_{i+1}(x_i+t)$$

$$(3.15) \qquad \frac{1}{60} S^{V}(x_i+t) = \gamma_{i-1}M''_{i-1}(x_i+t) + \gamma_i M''_i(x_i+t) + \gamma_{i+1}M''_{i+1}(x_i+t) \ .$$

On the right hand sides of equations (3.13) - (3.15) we insert the values from equations (3.1) - (3.3). If we make use of (3.12) then we find that

$$\left.\begin{aligned}
&(3.16) \qquad \frac{D_i}{10} = \frac{\gamma_{i-1}h_i + \gamma_i h_{i-1}}{h_{i-1} + h_i} \\
&(3.17) \qquad \frac{E_i}{5} = \frac{\gamma_i - \gamma_{i-1}}{h_{i-1} + h_i} \\
&(3.18) \qquad F_i = \frac{1}{h_i}\left(\frac{\gamma_{i+1} - \gamma_i}{h_i + h_{i+1}} - \frac{\gamma_i - \gamma_{i-1}}{h_{i-1} + h_i}\right)
\end{aligned}\right\} \quad i = 2,3,\ldots,n-3 \ .$$

These formulas can also be used for $i = 1, n-2, n-1$ by adding the convention that $\gamma_0 = \gamma_{n-1} = \gamma_n = 0$. Also (3.18) can be used for $i = 0$ by setting $\gamma_{-1} = 0$. ((3.16) and (3.17) also yield the correct required values $D_0 = E_0 = 0$ with these conventions.)

Next we want to find the values of B_i and C_i. Remembering that $S(x)$ and its first four derivatives are continuous at x_i but that $S^v(x)$ need not be continuous we can write polynomial expressions for $S(x)$ valid for the intervals on either side of x_i:

$$S(x) = y_i + B_i t + C_i t^2 + D_i t^3 + E_i t^4 + F_i t^5 \quad , \qquad x_i \le x \le x_{i+1}$$

$$S(x) = y_i + B_i t + C_i t^2 + D_i t^3 + E_i t^4 + F_{i-1} t^5 \quad , \qquad x_{i-1} \le x \le x_i$$

with $t = x - x_i$ in both cases, $i = 1, 2, \ldots, n-1$. Then

$$S(x_{i+1}) = y_{i+1} = y_i + B_i h_i + C_i h_i^2 + D_i h_i^3 + E_i h_i^4 + F_i h_i^5$$

and

$$S(x_{i-1}) = y_{i-1} = y_i - B_i h_{i-1} + C_i h_{i-1}^2 - D_i h_{i-1}^3 + E_i h_{i-1}^4 - F_{i-1} h_{i-1}^5 \quad .$$

Since D_i, E_i, F_i and F_{i-1} are already known we can solve these two equations for B_i and C_i obtaining

$$(3.19) \qquad B_i = \frac{h_{i-1}}{h_{i-1}+h_i} \, \frac{y_{i+1}-y_i}{h_i} + \frac{h_i}{h_{i-1}+h_i} \, \frac{y_i - y_{i-1}}{h_{i-1}} - D_i h_{i-1} h_i$$

$$+ E_i h_{i-1} h_i (h_{i-1} - h_i) - \frac{h_{i-1} h_i}{h_{i-1}+h_i} (F_{i-1} h_{i-1}^3 + F_i h_i^3)$$

and

$$(3.20) \qquad C_i = \frac{1}{h_{i-1}+h_i}\left(\frac{y_{i+1}-y_i}{h_i} - \frac{y_i-y_{i-1}}{h_{i-1}}\right) + D_i(h_{i-1}-h_i)$$

$$- E_i\,\frac{h_{i-1}^3+h_i^3}{h_{i-1}+h_i} + \frac{1}{h_{i-1}+h_i}\,(F_{i-1}h_{i-1}^4 - F_i h_i^4) \quad .$$

These formulas are valid for $i = 1,2,\ldots,n-1$.

Finally we have to find the coefficients at the end points x_0 , x_n . In the interval (x_0,x_1) we have (since $D_0 = E_0 = 0$)

$$S(x) = y_0 + B_0 t + C_0 t^2 + 0 + 0 + F_0 t^5$$

with $t = x-x_0$. Hence also

$$S''(x) = 2C_0 + 20F_0 t^3 \quad .$$

Since $S(x)$ and $S''(x)$ must be continuous at $x = x_1$ we have

$$y_0 + B_0h_0 + C_0h_0^2 + F_0h_0^2 = y_1$$

$$2C_0 + 20F_0h_0^3 = 2C_1 \quad .$$

Therefore

$$(3.21) \qquad C_0 = C_1 - 10F_0h_0^3$$

$$(3.22) \qquad B_0 = \frac{y_1-y_0}{h_0} - C_0h_0 - F_0h_0^4 \quad .$$

In the interval (x_{n-1},x_n) we have (since $S'''(x_n) = S''''(x_n) = 0$)

$$S(x) = y_n + B_n t + C_n t^2 + 0 + 0 + F_{n-1} t^5$$

with $t = x-x_n$. (Here $B_n = S'(x_n)$, $C_n = S''(x_n)/2$.) Hence also

$$S''(x) = 2C_n + 20F_{n-1} t^3 \quad .$$

Since $S(x)$ and $S''(x)$ must be continuous at $x = x_{n-1}$ we have

$$y_n - B_n h_{n-1} + C_n h_{n-1}^2 - F_{n-1} h_{n-1}^5 = y_{n-1}$$

$$2C_n - 20F_{n-1} h_{n-1}^3 = 2C_{n-1} \quad .$$

Therefore

(3.23) $$C_n = C_{n-1} + 10F_{n-1} h_{n-1}^3$$

(3.24) $$B_n = \frac{y_n - y_{n-1}}{h_{n-1}} + C_n h_{n-1} - F_{n-1} h_{n-1}^4 \quad .$$

3.2 Coincident Knots

By relaxing the condition that the set of knots x_i be strictly monotone increasing and allowing two consecutive knots x_j, x_{j+1} to be equal we can use the procedure QUINAT to find a spline function for which the first derivatives are specified at an arbitrary number of knots in addition to the specification of the function values at the knots. In order to understand this situation consider two knots x_j and x_{j+1} which are close together. Let $x_{j+1} - x_j = \epsilon$ where ϵ is small and positive. Instead of specifying $S(x_j) = y_j$ and $S(x_{j+1}) = y_{j+1}$ we may instead specify $S(x_j) = y_j$ and the first divided difference $S(x_j, x_{j+1}) = (y_{j+1} - y_j)/\epsilon = y_{j,j+1}$. The two data specifications are completely equivalent. Now when ϵ is small, $S(x_j, x_{j+1})$ is very close to $S'(x_j)$ and indeed $S(x_j, x_{j+1}) \to S'(x_j)$ as $\epsilon \to 0$. Hence if $x_{j+1} = x_j$ it is entirely reasonable to adopt the convention of specifying $S(x_j)$ as y_j and $S'(x_j)$ as y_{j+1} . The procedure QUINAT has been written making use of this convention. For the spline function produced by QUINAT in this case, the fourth derivative $S''''(x)$ and the fifth derivative $S^V(x)$ have jump discontinuities at x_j . However $S(x)$, $S'(x)$, $S''(x)$ and $S'''(x)$ are all continuous at x_j . Table 1 shows the input and output values of QUINAT corresponding to the subscripts j and j+1 .

$y_j = S(x_j)$	$y_{j+1} = S'(x_j)$
$B_j = S'(x_j)$	$= B_{j+1}$
$C_j = S''(x_j)/2$	$= C_{j+1}$
$D_j = S'''(x_j)/6$	$= D_{j+1}$
$E_j = S''''(x_j-0)/24$	$E_{j+1} = S''''(x_j+0)/24$
$F_j = S^V(x_j-0)/120$	$F_{j+1} = S^V(x_j+0)/120$

Table 1. Double Knot $x_j = x_{j+1}$

When one uses the equation (2.1) to calculate $S(x)$ in the interval $[x_{j+1},x_{j+2})$ one should remember that y_{j+1} appearing there is in fact $S(x_{j+1}) = S(x_j) = y_j$ and not the y_{j+1} of the input data. Rather B_{j+1} has the value of the input y_{j+1} .

In the same way one may choose $x_j = x_{j+1} = x_{j+2}$ in QUINAT and specify $S(x_j)$ as y_j , $S'(x_j)$ as y_{j+1} and $S''(x_j)$ as y_{j+2} . For the spline function produced by QUINAT in this case, the derivatives $S'''(x)$, $S''''(x)$ and $S^V(x)$ all have jump discontinuities at x_j .

However $S(x)$, $S'(x)$ and $S''(x)$ are continuous at x_j . Table 2 shows the input and output values of QUINAT corresponding to the subscripts j , $j+1$ and $j+2$.

$y_j = S(x_j)$	$y_{j+1} = S'(x_j)$	$y_{j+2} = S''(x_j)$
$B_j = S'(x_j)$	$= B_{j+1} = B_{j+2}$	
$C_j = S''(x_j)/2$	$= C_{j+1} = C_{j+2}$	
$D_j = S'''(x_j-0)/6$	$D_{j+1} = 0$	$D_{j+2} = S'''(x_j+0)/6$
$E_j = S''''(x_j-0)/24$	$E_{j+1} = 0$	$E_{j+2} = S''''(x_j+0)/24$
$F_j = S^{V}(x_j-0)/120$	$F_{j+1} = 0$	$F_{j+2} = S^{V}(x_j+0)/120$

Table 2. Triple Knot $x_j = x_{j+1} = x_{j+2}$

When one uses the equation (2.1) to calculate $S(x)$ in the interval $[x_{j+2}, x_{j+3})$ one should remember that y_{j+2} appearing there is in fact $S(x_{j+2}) = S(x_{j+1}) = S(x_j) = y_j$ and not the y_{j+2} of the input data. Rather B_{j+2} has the value of the input y_{j+1} and $2C_{j+2}$ has the value of the input y_{j+2} .

4. Procedure QUINEQ

The calculation of the coefficients in QUINEQ for the case of equidistant knots is carried out in the same manner as in QUINAT for the general case. However, there are a number of simplifications which result in considerable economy of computational effort. It is not necessary to specify x_i . Hence we can assume $x_i = i$. Then $h_i = 1$ for all i and the coefficients of M_i are independent of i as are also the inner products. Thus equations (3.1) reduce to

$$
\begin{aligned}
M_i(x) &= \frac{1}{2} t^2 & & i-1 \le x < i \\
&= \frac{1}{2} + u - u^2 & & i \le x < i+1 \qquad (4.1) \\
&= \frac{1}{2} (v-1)^2 & & i+1 \le x < i+2
\end{aligned}
$$

with $t = x-(i-1)$, $u = x-i$, $v = x-(i+1)$.

The inner products become

$$(4.2) \qquad 30 \int_{-\infty}^{\infty} [M_i(x)]^2 dx = 66/4$$

$$(4.3) \qquad 30 \int_{-\infty}^{\infty} M_i(x) M_{i+1}(x) = 26/4$$

$$(4.4) \qquad 30 \int_{-\infty}^{\infty} M_i(x) M_{i+2}(x) dx = 1/4 \quad .$$

The divided differences become ordinary differences so that equation (3.4) becomes

$$
\begin{aligned}
(4.5) \qquad \int_{-\infty}^{\infty} M_i(x) f'''(x) dx &= f(x_{i+2}) - 3f(x_{i+1}) + 3f(x_i) - f(x_{i-1}) \\
&= \Delta^3 f(x_{i-1}) \quad .
\end{aligned}
$$

If instead of equation (3.9) we take

$$(4.6) \qquad S'''(x) = \sum_{j=0}^{n-3} 120 \gamma_j M_{j+1}(x)$$

then the system of linear equations for the calculation of γ_j can be written in the form

$$
(4.7) \qquad
\begin{bmatrix}
66 & 26 & 1 & & & & & \\
26 & 66 & 26 & 1 & & & & \\
1 & 26 & 66 & 26 & 1 & & & \\
 & 1 & 26 & 66 & 26 & 1 & & \\
 & & \ddots & \ddots & \ddots & \ddots & \ddots & \\
 & & & & 1 & 26 & 66 & 26 \\
 & & & & & 1 & 26 & 66
\end{bmatrix}
\begin{bmatrix}
\gamma_0 \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \\ \gamma_{n-4} \\ \gamma_{n-3}
\end{bmatrix}
=
\begin{bmatrix}
\Delta^3 y_0 \\ \Delta^3 y_1 \\ \Delta^3 y_2 \\ \Delta^3 y_3 \\ \vdots \\ \Delta^3 y_{n-4} \\ \Delta^3 y_{n-3}
\end{bmatrix}
$$

The solution of this system of equations is somewhat simplified because the matrix of coefficients is a set of constants.

The equations for the determination of the spline function coefficients then become

$$(4.8) \qquad \frac{D_i}{10} = \gamma_{i-2} + \gamma_{i-1}$$

$$(4.9) \qquad \frac{E_i}{5} = \gamma_{i-1} - \gamma_{i-2}$$

$$(4.10) \qquad F_i = \gamma_i - \gamma_{i-1} - \gamma_{i-1} + \gamma_{i-2}$$

$$(4.11) \qquad B_i = \frac{1}{2}(y_{i+1} - y_{i-1} - F_{i-1} - F_i) - D_i$$

$$(4.12) \qquad C_i = \frac{1}{2}(y_{i+1} + y_{i-1} + F_{i-1} - F_i) - y_i - E_i \quad .$$

These formulas are valid for $i = 1,2,\ldots,n-1$ with the convention that $\gamma_{-1} = \gamma_{n-2} = \gamma_{n-1} = 0$. Also equation (4.10) can be used for $i = 0$ by setting $\gamma_{-2} = 0$. (Equations (4.8) and (4.9) also yield the correct values $D_0 = E_0 = 0$ with these conventions.)

Finally the coefficients at the end points are given by

$$C_0 = C_1 - 10F_0$$

$$B_0 = y_1 - y_0 - C_0 - F_0$$

$$C_n = C_{n-1} + 10F_{n-1}$$

$$B_n = y_n - y_{n-1} + C_n - F_{n-1} \ .$$

5. Procedure QUINDF

We now assume that $S(x_i) = y_i$ and $S'(x_i) = y_i'$ are specified at each of the knots. We must exclude the possibility that $x_i = x_{i+1}$ as this would imply a multiplicity of four which is not feasible for quintic splines.

As in Section 3 we shall use minimum support B-splines of degree 2 to form a basis for the class of third derivatives of the quintic natural splines. Since the spline we are seeking is of deficiency two because the derivatives are specified, so also our B-splines must be of deficiency two. Specifying a derivative at a knot is equivalent to considering a knot to be a double knot as we saw in Section 3.2. Hence the desired deficient splines of degree 2 can be derived from those used in Section 3.1 by permitting two knots to become coincident. However we prefer to derive these B-splines directly. Two sets of such deficient splines are possible.

As in Section 3.1 we assume that the knots are strictly monotone increasing and we again choose $N1 = 0$, $N2 = n$. The specified data are denoted by (x_i, y_i, y_i') , $i = 0,1,\ldots,n$.

We denote a B-spline of one set by $M_i(x)$ and require that it vanish outside the interval (x_{i-1}, x_{i+1}) . $M_i(x)$ and $M_i'(x)$ should be continuous at x_{i-1} and x_{i+1} but continuity is required at x_i only for $M_i(x)$ and not for $M_i'(x)$. As usual, let $h_i = x_{i+1} - x_i$, $t = x - x_{i-1}$, $u = x - x_i$. Then

$$
\begin{aligned}
(5.1) \qquad M_i(x) &= At^2 && x_{i-1} \leq x < x_i \\
&= B(u-h_i)^2 && x_i \leq x < x_{i+1}
\end{aligned}
$$

Hence also

$$
\begin{array}{llll}
(5.2) \qquad M_i'(x) = 2At & \Big| \; M_i''(x) = 2A & & x_{i-1} \leq x < x_i \\
\qquad\qquad\quad\;\; = 2B(u-h_i) & \Big| \; \qquad\quad\; = 2B & & x_i \leq x < x_{i+1} \quad .
\end{array}
$$

Imposing the continuity requirement at x_i yields

$$
Ah_{i-1}^2 = Bh_i^2 \quad .
$$

Hence up to a common factor

$$
(5.3) \qquad A = \frac{1}{h_{i-1}^2} \qquad\qquad B = \frac{1}{h_i^2} \quad .
$$

We denote a B-spline of the other set by $N_i(x)$ and require that it vanish outside the interval (x_i, x_{i+1}) . $N_i(x)$ should be continuous at x_i , x_{i+1} but no continuity is required for the derivative. We can clearly choose

$$
(5.4) \qquad N_i(x) = \frac{2}{h_i^2} u(h_i - u) \qquad\qquad x_i \leq x < x_{i+1}
$$

with $u = x - x_i$. Then also

$$
(5.5) \qquad N_i'(x) = \frac{2}{h_i^2}(h_i - 2u) \qquad N_i''(x) = -\frac{4}{h_i^2} \qquad x_i \leq x < x_{i+1}
$$

Our choice of coefficients implies the following two relations

$$(5.6)' \qquad \int_{-\infty}^{\infty} M_i(x) f'''(x)\, dx = -\int_{-\infty}^{\infty} M_i'(x) f''(x) dx$$

$$= 2\left\{\frac{f(x_i) - f(x_{i-1})}{h_{i-1}^2} - \left(\frac{1}{h_{i-1}} + \frac{1}{h_i}\right) f'(x_i) + \frac{f(x_{i+1}) - f(x_i)}{h_i^2}\right\}$$

$$(5.7) \qquad \int_{-\infty}^{\infty} N_i(x) f'''(x) dx = -\int_{-\infty}^{\infty} N_i'(x) f''(x)$$

$$= 2\left\{\frac{f'(x_i)}{h_i^2} - \frac{2}{h_i^2}\left(f(x_{i+1}) - f(x_i)\right) + \frac{f'(x_{i+1})}{h_i^2}\right\} \quad .$$

The basis for the third derivatives of the deficient quintic natural splines is the set of B-splines

$$\{N_0(x), M_1(x), N_1(x), M_2(x), N_2(x), \ldots, M_{n-1}(x), N_{n-1}(x)\} \quad .$$

We need the inner products of these basis functions. We easily find that

$$(5.8) \qquad 30\int_{-\infty}^{\infty} [M_i(x)]^2 dx = 6(h_{i-1} + h_i) \quad , \qquad i = 1,2,\ldots,n-1$$

$$(5.9) \qquad 30\int_{-\infty}^{\infty} [N_i(x)]^2 dx = 4h_i \quad , \qquad i = 0,1,\ldots,n-1$$

$$(5.10) \qquad 30\int_{-\infty}^{\infty} N_i(x) M_{i+1}(x) dx = 3h_i \quad , \qquad i = 0,1,\ldots,n-2$$

$$(5.11) \qquad 30\int_{-\infty}^{\infty} N_i(x) M_i(x) dx = 3h_i \quad , \qquad i = 1,2,\ldots,n-1$$

$$(5.12) \qquad 30\int_{-\infty}^{\infty} M_i(x) M_{i+1}(x) dx = h_i \quad , \qquad i = 1,2,\ldots,n-2 \; .$$

All the other inner products are zero.

Now the third derivative $S'''(x)$ can be expressed in terms of the basis functions:

$$(5.13) \qquad S'''(x) = 60 \left\{ \sum_{j=1}^{n-1} \beta_j M_j(x) + \sum_{j=0}^{n-1} \gamma_j N_j(x) \right\} .$$

If we multiply equation (5.13) by $\frac{1}{2} M_i(x)$ and $\frac{1}{2} N_i(x)$ respectively and integrate we obtain the well-conditioned system of linear equations for the determination of β_j and γ_j :

$$(5.14) \quad \begin{cases} \sum_{j=1}^{n-1} \left(30 \int_{-\infty}^{\infty} M_i(x) M_j(x) dx\right) \beta_j + \sum_{j=0}^{n-1} \left(30 \int_{-\infty}^{\infty} M_i(x) N_j(x) dx\right) \gamma_j \\ \qquad = \frac{1}{2} \int_{-\infty}^{\infty} M_i(x) S'''(x) dx \quad , \quad i = 1,2,\ldots,n-1 \\ \\ \sum_{j=1}^{n-1} \left(30 \int_{-\infty}^{\infty} N_i(x) M_j(x) dx\right) \beta_j + \sum_{j=0}^{n-1} \left(30 \int_{-\infty}^{\infty} N_i(x) N_j(x) dx\right) \gamma_j \\ \qquad = \frac{1}{2} \int_{-\infty}^{\infty} N_i(x) S'''(x) dx \quad , \quad i = 0,1,\ldots,n-1 \quad . \end{cases}$$

This is a positive definite pentadiagonal system of linear equations. The values of the non-zero coefficients are given by equations (5.8)-(5.12) and the right hand sides by equations (5.6) and (5.7). The system can be written in the form

$$
\begin{bmatrix}
e_0 & f_0 & 0 & & & & & & & & \\
f_0 & d_1 & g_1 & h_1 & & & & & \mathbf{O} & & \\
0 & g_1 & e_1 & f_1 & 0 & & & & & & \\
 & h_1 & f_1 & d_2 & g_2 & h_2 & & & & & \\
 & & 0 & g_2 & e_2 & f_2 & 0 & & & & \\
 & & & h_2 & f_2 & d_3 & g_3 & h_3 & & & \\
 & & & & 0 & g_3 & e_3 & f_3 & 0 & & \\
 & & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \\
 & \mathbf{O} & & & & & \ddots & \ddots & \ddots & \ddots & \ddots \\
 & & & & & & & h_{n-2} & f_{n-2} & d_{n-1} & g_{n-1} \\
 & & & & & & & & 0 & g_{n-1} & e_{n-1}
\end{bmatrix}
\begin{bmatrix}
\gamma_0 \\ \beta_1 \\ \gamma_1 \\ \beta_2 \\ \gamma_2 \\ \beta_3 \\ \gamma_3 \\ \vdots \\ \beta_{n-1} \\ \gamma_{n-1}
\end{bmatrix}
=
\begin{bmatrix}
c_0 \\ b_1 \\ c_1 \\ b_2 \\ c_2 \\ b_3 \\ c_3 \\ \vdots \\ b_{n-1} \\ c_{n-1}
\end{bmatrix}
$$

where

$$d_i = 6(h_{i-1} + h_i) \quad , \qquad i = 1,2,\ldots,n-1$$

$$e_i = 4h_i \quad , \qquad i = 0,1,\ldots,n-1$$

$$f_i = 3h_i \quad , \qquad i = 0,1,\ldots,n-2$$

$$g_i = 3h_i \quad , \qquad i = 1,2,\ldots,n-1 \quad .$$

$$b_i = \frac{y_i - y_{i-1}}{h_{i-1}^2} - \left(\frac{1}{h_{i-1}} + \frac{1}{h_i}\right) y_i' + \frac{y_{i+1} - y_i}{h_i^2} \quad , \qquad i = 1,2,\ldots,n-1$$

$$c_i = \frac{y_i'}{h_i^2} - \frac{2}{h_i^2}(y_{i+1} - y_i) + \frac{y_{i+1}'}{h_i^2} \quad , \qquad i = 0,1,\ldots,n-1 \quad .$$

This system of equations can be solved for the β_j and γ_j by using Gaussian elimination to reduce the matrix of coefficients to upper triangular form. This is conveniently done by annihilating the elements below the principal diagonal in two rows at a time. Note that the g_i and h_i are never changed. We use d_i' , e_i', f_i' and d_i'' , e_i'' , f_i'' to denote the new values of d_i , e_i , f_i as they are formed by the annihilation of h_{i-1} , f_{i-1}' and g_i in the i-th pair of rows. Before the elimination process operates on the i-th pair of rows the form of the system is

$$
\begin{bmatrix}
e_0'' & f_0'' & 0 & & & & & & \\
 & \ddots & \ddots & \ddots & & & \bigcirc & & \\
 & & d_{i-1}'' & g_{i-1} & h_{i-1} & & & & \\
 & & 0 & e_{i-1}'' & f_{i-1}'' & & & & \\
 & & h_{i-1} & f_{i-1} & d_i & g_i & h_i & & \\
 & \bigcirc & & 0 & g_i & e_i & f_i & 0 & \\
 & & & & & \ddots & \ddots & \ddots &
\end{bmatrix}
\begin{bmatrix}
\gamma_0 \\ \vdots \\ \beta_{i-1} \\ \gamma_{i-1} \\ \beta_i \\ \gamma_i \\ \vdots
\end{bmatrix}
=
\begin{bmatrix}
c_0'' \\ \vdots \\ b_{i-1}'' \\ c_{i-1}'' \\ b_i \\ c_i \\ \vdots
\end{bmatrix}
$$

The equations for the annihilation of h_{i-1} are

$$u_i = h_{i-1} / d_{i-1}''$$

$$f_{i-1}' = f_{i-1} - u_i g_{i-1}$$

$$d_i' = d_i - u_i h_{i-1}$$

$$b_i' = b_i - u_i b_{i-1}'' \quad .$$

Those for the annihilation of f'_{i-1} are

$$v_i = f'_{i-1} / e''_{i-1}$$

$$d''_i = d'_i - v_i f''_{i-1}$$

$$b''_i = b'_i - v_i c''_{i-1}$$

and those for the annihilation of g_i are

$$w_i = g_i / d''_i$$

$$e''_i = e_i - w_i g_i$$

$$f''_i = f_i - w_i h_i$$

$$c''_i = c_i - w_i b_i \quad .$$

Note that f'_{i-1} need not be calculated because

$$f'_{i-1} = f_{i-1} - (h_{i-1} / d''_{i-1}) g_{i-1} = f_{i-1} - (g_{i-1} / d''_{i-1}) h_{i-1} = f''_{i-1} \quad .$$

For the first step of the elimination (operating on the 2nd and 3rd rows) the above formulas are valid if we choose $u_1 = 0$.

When the coefficients β_j and γ_j have been found, $S'''(x)$ is given by equation (5.13). We want to find the coefficients of $S(x)$ as expressed in equation (2.1) with $B_i = y'_i$. Clearly

$$(5.15) \qquad D_i = S'''(x_i+0)/6 \ , \ E_i = S''''(x_i+0)/24 \ , \ F_i = S^V(x_i+0)/120 \ , \qquad i = 0,1,\ldots,n-1 \quad .$$

But because $M_j(x)$ vanishes outside the interval (x_{j-1},x_{j+1}) and $N_j(x)$ vanishes outside the interval (x_j,x_{j+1}) , $S'''(x)$ can be represented in the interval $[x_i,x_{i+1})$ in the very simple form

$$(5.16)\qquad \frac{1}{60} S'''(x_i+t) = \beta_i M_i(x_i+t) + \beta_{i+1} M_{i+1}(x_{i+t}) + \gamma_i N_i(x_i+t) \quad ,$$

$$0 < t < h_i \ , \quad i = 1,2,\ldots,n-2 \quad .$$

Also

$$(5.17)\qquad \frac{1}{60} S''''(x_i+t) = \beta_i M_i'(x_i+t) + \beta_{i+1} M_{i+1}'(x_i+t) + \gamma_i N_i'(x_i+t) \quad .$$

$$(5.18)\qquad \frac{1}{60} S^{V}(x_i+t) = \beta_i M_i''(x_i+t) + \beta_{i+1} M_{i+1}''(x_i+t) + \gamma_i N_i''(x_i+t) \quad .$$

On the right-hand sides of equations (5.16)-(5.18) we insert the values from equations (5.1)-(5.3). If we make use of (5.15) we find that

$$\left.\begin{aligned} &(5.19)\qquad \frac{D_i}{10} = \beta_i \\ &(5.20)\qquad \frac{E_i}{5} = \frac{\gamma_i - \beta_i}{h_i} \\ &(5.21)\qquad F_i = \frac{\beta_i - 2\gamma_i + \beta_{i+1}}{h_i^2} \end{aligned}\right\} \qquad i = 1,2,\ldots,n-2 \quad .$$

These formulas can also be used for $i = 0 \, , n-1$ by adding the convention that $\beta_0 = \beta_n = 0$.

Next we want to find the values of the C_i . We can write $S(x)$ in the form (2.1) for $x_i \le x < x_{i+1}$. Then we can use either $S(x_{i+1}) = y_{i+1}$ or $S'(x_{i+1}) = y_{i+1}'$. We prefer the latter because the resulting formula has less danger of cancellation. We have at once

$$2C_i = \frac{y_{i+1}' - y_i'}{h_i} - 3D_i h_i - 4E_i h_i^2 - 5F_i h_i^3 \quad .$$

If we substitute from equations (5.19)-(5.21) this becomes

$$(5.22) \qquad 2C_i = \frac{y'_{i+1} - y'_i}{h_i} - h_i(15\beta_i + 10\gamma_i + 5\beta_{i+1}) \quad .$$

This formula can be used for $i = 0,1,\ldots,n-1$ but not for $i = n$.

In order to get C_n we could write the polynomial for $S(x)$ in (x_{n-1},x_n) in powers of $x-x_n$ and then use $S'(x_{n-1}) = y'_{n-1}$. However, it is more convenient to obtain another formula for C_i valid for $i = 1,2,\ldots,n$ by writing the polynomial for $S(x)$ in $(x_{i-1},x_i]$ in powers of $x-x_i$ and then using $S'(x_{i-1}) = y'_{i-1}$. We have

$$S(x) = y_i + y'_i t + C_i^* t^2 + D_i^* t^3 + E_i^* t^4 + F_i^* t^5 \quad , \qquad x_{i-1} < x \leq x_i$$

with $t = x-x_i$. Then

$$D_i^* = S'''(x_i-0)/6 \quad , \quad E_i^* = S''''(x_i-0)/24 \quad , \quad F_i^* = S^{V}(x_i-0)/120 \quad ,$$

$$i = 1,2,\ldots,n \quad .$$

Proceeding as in the derivation of (5.19)-(5.21) we find that

$$\left.\begin{aligned} \frac{D_i^*}{10} &= \beta_i \\ \frac{E_i^*}{5} &= \frac{\beta_i - \gamma_{i-1}}{h_{i-1}} \\ F_i^* &= \frac{\beta_{i-1} - 2\gamma_{i-1} + \beta_i}{h_{i-1}^2} \end{aligned}\right\} \quad i = 2,3,\ldots,n-1 \quad .$$

These formulas can also be used for $i = 1, n$ by adding the convention $\beta_0 = \beta_n = 0$.

If we now use the relation $S'(x_{i-1}) = y'_{i-1}$, we find

$$2C_i^* = \frac{y'_i - y'_{i-1}}{h_{i-1}} + 3D_i^* h_{i-1} - 4E_i^* h_{i-1}^2 + 5F_i^* h_{i-1}^3 \quad .$$

Substituting the above values for D_i^* , E_i^* , F_i^* yields

$$(5.23) \qquad 2C_i^* = \frac{y'_i - y'_{i-1}}{h_{i-1}} + h_{i-1}(5\beta_{i-1} + 10\gamma_{i-1} + 15\beta_i) \quad .$$

This formula can be used for $i = 1,2,\ldots,n$ but not $i = 0$.

Since $S''(x)$ is continuous for $x = x_i$, $i = 1,2,\ldots,n-1$, formulas (5.22) and (5.23) must yield the same values for these values of i . We use (5.23) only for $i = n$.

6. Tests

These procedures have been tested in Algol 60 on the Telefunken TR-440 computer at the Leibniz Rechenzentrum of the Bavarian Academy of Sciences, Munich, and in Algol W on the IBM 360/67 at the Stanford Center for Information Processing. The latter tests included timing tests of the procedures with the number of knots $N = N2 - N1 + 1$ ranging up to 1000 . The time was found to be approximately proportional to the number N of knots. The time T in seconds for the execution of the procedure QUINAT was found to be approximately $T = .00212N$ whereas for the procedure NATSPLINE of Algorithm 472 [5] with $m = 3$ it was found that $T = .01324N$ or over six times as great. For the procedure QUINEQ the time was approximately $T = .00073N$ whereas for the procedure NATSPLINEEQ of Algorithm 472 with $m = 3$ it was $T = .00410N$ or nearly six times as great. For the procedure QUINDF the time was approximately $T = .00116N$ whereas for the procedure QUINAT with $2N$ knots, consecutive knots being equal in pairs, the time was $T = .00360N$ or over three times as great. Moreover, in order to compute the same results the procedure QUINAT requires approximately 75 per cent more storage for the arrays used than does the procedure QUINDF. Note also that from the above formula for the time required by the procedure QUINAT, the time for $2N$ distinct knots would be $T = .00424N$ which can be compared with $T = .00360N$ given above for N pairs of equal knots. The reduction for the case of double knots probably occurs because some calculations are omitted when knots are coincident.

These timing comparisons show that it is definitely advantageous to have these special procedures for the quintic natural spline instead of using the general cases given in Algorithm 472 with $m = 3$.

Tests of the accuracy and correctness of the coefficients computed by the procedures QUINAT, QUINEQ and QUINDF were carried out as described in Algorithm 472 [5]. Table 3 shows the results of a typical run using QUINAT for 5 equidistant points. The first line of each box gives the tabulated quantities at the given value of x which is the left-hand endpoint of the subinterval and the second line of the box gives the tabulated quantities at the right-hand endpoint of the same subinterval. The close agreement of the quantities $S(x)$, $S'(x)$, $S''(x)/2$, $S'''(x)/6$ and $S''''(x)/24$ shows that the quintic spline function and its derivatives satisfy the required continuity conditions. This is a good indication of the correctness of the results. Almost identical results were obtained from the same data using QUINEQ. The

x	S(x)	S'(x)	S"(x)/2	S"'(x)/6	S""(x)/24	$S^V(x)/120$
1.000000	1.000000 0	-3.199998 0.8999977	2.299998 1.299998	0 -0.9999990	0 -0.4999995	-0.09999990 -0.09999990
2.000000	0 1.000000	0.8999997 1.907349'-06	1.299999 -1.699997	-0.9999996 4.768372'-07	-0.4999995 0.9999990	0.2999997 0.2999997
3.000000	1.000000 5.364418'-07	5.662441'-07 -0.9000010	-1.699999 1.299996	-5.960464'-07 0.9999983	0.9999990 -0.4999992	-0.2999997 -0.2999997
4.000000	0 0.9999982	-0.8999985 3.199994	1.299998 2.299995	0.9999985 0	-0.4999992 0	0.09999985 0.09999985
5.000000	1.000000					

Table 3. Quintic Spline. 5 Equidistant Knots. Coefficients calculated by QUINAT. (Machine precision approximately 7 decimal digits.)

procedures NATSPLINE and NATSPLINEEQ of Algorithm 472 also produced essentially the same results.

Table 4 shows the results of a typical run using QUINDF for 5 nonequidistant points. The values of the function and its first derivatives were specified and the results are given in the same format as in Table 3. Note that the fourth and fifth derivatives are now discontinuous. Essentially the same results were obtained by using QUINAT with 10 knots, equal in pairs.

x	S(x)	S'(x)	S"(x)/2	S"'(x)/6	S""(x)/24	$S^{V}(x)/120$
-3.000000	7.000000 11.00002	2.000000 15.00002	-6.108372 7.674892	0 -4.933491	2.956281 -4.189653	-0.7145936 -0.7145936
-1.000000	11.00000 26.00000	15.00000 10.00001	7.674872 -1.908858	-4.933500 16.59850	-8.157616 18.92361	5.416246 5.416246
0	26.00000 55.99988	10.00000 -27.00012	-1.908856 -5.264510	16.59848 20.03847	-9.059000 9.632335	1.246089 1.246089
3.000000	56.00000 29.00002	-27.00000 -29.99995	-5.264445 -7.754762	20.03851 4.005432'-05	-21.28369 11.26445	6.509629 6.509629
4.000000	29.00000	-30.00000				

Table 4. Quintic Spline. 5 nonequidistant knots. Values and first derivatives specified. Coefficients calculated by QUINDF. (Machine precision approximately 7 decimal digits.)

References

[1] Anselone, P. M. and Laurent, P. J., "A general method for the construction of interpolating and smoothing spline functions," Numer. Math. 12 (1968), 66-82.

[2] Curry, H. B. and Schoenberg, I. J., "On Pólya frequency functions. IV. The fundamental spline functions and their limits," J. Analyse Math. 17 (1966), 71-107.

[3] Greville, T. N. E., "Spline functions, interpolation and numerical quadrature," in Mathematical Methods for Digital Computers, vol. II. A. Ralston and H. S. Wilf (Eds.), Wiley, New York, 1967.

[4] Greville, T. N. E., "Introduction to spline functions," in Theory and Applications of Spline Functions. T. N. E. Greville (Ed.), Academic Press, New York, 1969, 1-35. (Pub. No. 22, Mathematics Research Center, U. S. Army, University of Wisconsin.)

[5] Herriot, John G. and Reinsch, Christian H., "Algorithm 472. Procedures for natural spline interpolation," Comm. ACM 16 (1973), 763-768.

[6] Schoenberg, I. J., "On interpolation by spline functions and its minimal properties," in On Approximation Theory. Proceedings of the Conference at Oberwolfach, 1963. P. L. Butzer and J. Korevaar (Eds.), Birkhaüser Verlag, Basel, 1964, 109-129.

APPENDIX I

Algol 60 procedure QUINAT

```
procedure QUINAT(N1,N2) data:(x,y) result:(B,C,D,E,F);
  value N1,N2; integer N1,N2; array x,y,B,C,D,E,F;
comment  QUINAT computes the coefficients of a quintic natural spline
  S(x) interpolating the ordinates y[i] at points x[i], i = N1 through
  N2.  For xx in [x[i],x[i+1]) the value of the spline function S(xx)
  is given by the fifth degree polynomial:
  S(xx) = ((((F[i]xt + E[i])xt + D[i])xt + C[i])xt + B[i])xt + y[i]
  with t = xx - x[i].
  Input:
    N1,N2 subscript of first and last data point respectively, it is
          required that N2 > N1 + 1,
    x,y[N1:N2] arrays with x[i] as abscissa and y[i] as ordinate of
          i-th data point.  The elements of the array x must be strictly
          monotone increasing (but see below for exceptions to this).
  Output:
    B,C,D,E,F[N1:N2] arrays collecting the coefficients of the quintic
          natural spline S(xx) as described above.  Specifically
          B[i] = S'(x[i]), C[i] = S"(x[i])/2, D[i] = S"'(x[i])/6,
          E[i] = S""(x[i])/24, F[i] = S^V(x[i]+0)/120.  F[N2] is neither
          used nor altered.  The arrays B,C,D,E,F must always be distinct.
  Options:
    1.  The requirement that the elements of the array x be strictly
        monotone increasing can be relaxed to allow two or three consecutive
        abscissas to be equal and then specifying values of the first and
        second derivatives of the spline function at some of the
        interpolating points.  Specifically
        if x[j] = x[j+1] then S(x[j]) = y[j] and S'(x[j]) = y[j+1],
        if x[j] = x[j+1] = x[j+2] then in addition S"(x[j]) = y[j+2].
        Note that S""(x) is discontinuous at a double knot and in
        addition S"'(x) is discontinuous at a triple knot.  At a double
        knot, x[j] = x[j+1], the output coefficients have the following
        values:
```

```
    B[j] = S'(x[j])         = B[j+1]
    C[j] = S"(x[j])/2       = C[j+1]
    D[j] = S"'(x[j])/6      = D[j+1]
    E[j] = S""(x[j]-0)/24   E[j+1] = S""(x[j]+0)/24
    F[j] = S^V(x[j]-0)/120  F[j+1] = S^V(x[j]+0)/120
  The representation of S(xx) remains valid in all intervals
  provided the redefinition y[j+1] := y[j] is made immediately
  after the call of the procedure QUINAT.  At a triple knot,
  x[j] = x[j+1] = x[j+2], the output coefficients have the
  following values:
    B[j] = S'(x[j])         = B[j+1] = B[j+2]
    C[j] = S"(x[j])/2       = C[j+1] = C[j+2]
    D[j] = S"'(x[j]-0)/6    D[j+1] = 0        D[j+2] = S"'(x[j]+0)/6
    E[j] = S""(x[j]-0)/24   E[j+1] = 0        E[j+2] = S""(x[j]+0)/24
    F[j] = S^V(x[j]-0)/120  F[j+1] = 0        F[j+2] = S^V(x[j]+0)/120
  The representation of S(xx) remains valid in all intervals
  provided the redefinition y[j+2] := y[j+1] := y[j] is made
  immediately after the call of the procedure QUINAT.
 2. The array x may be monotone decreasing instead of increasing;
if N2 > N1 + 1 then
begin
  integer i,m;
  real b1,p,pq,pqqr,pr,p2,p3,q,qr,q2,q3,r,r2,s,t,u,v;
  comment  Coefficients of a positive definite, pentadiagonal matrix
    stored in D,E,F[N1+1:N2-2];
  m := N2 - 2;
  q := x[N1+1] - x[N1]; r := x[N1+2] - x[N1+1];
  q2 := q×q; r2 := r×r; qr := q + r;
  D[N1] := E[N1] := 0.0;
  D[N1+1] := if q = 0 then 0.0 else 6.0×q×q2/(qr×qr);
  for i := N1 + 1 step 1 until m do
  begin
    p := q; q := r; r := x[i+2] - x[i+1];
    p2 := q2; q2 := r2; r2 := r×r; pq := qr; qr := q + r;
    if q = 0 then D[i+1] := E[i] := F[i-1] := 0.0
    else
```

```
  begin
    q3 := q2xq; pr := pxr; pqqr := pqxqr;
    D[i+1] := 6.0xq3/(qrxqr);
    D[i] := D[i] + (q+q)x(15.0xprxpr + (p+r)xqx(20.0xpr + 7.0xq2)
            + q2x(8.0x(p2 + r2) + 21.0xpr + q2 + q2))/(pqqrxpqqr);
    D[i-1] := D[i-1] + 6.0xq3/(pqxpq);
    E[i] := q2x(pxqr + 3.0xpqx(qr+r+r))/(pqqrxqr);
    E[i-1] := E[i-1] + q2x(rxpq + 3.0xqrx(pq+p+p))/(pqqrxpq);
    F[i-1] := q3/pqqr
  end q < > 0
end i;
if r ≠ 0.0 then D[m] := D[m] + 6.0xrxr2/(qrxqr);
comment  First and second order divided differences of the given
  function values stored in B[N1+1:N2] and C[N1+2:N2] respectively.
  Take care of double and triple knots;
s := y[N1];
for i := N1 + 1 step 1 until N2 do
  if x[i] = x[i-1] then B[i] := y[i]
  else
  begin
    B[i] := (y[i] -s)/(x[i] - x[i-1]);
    s := y[i]
  end i;
for i := N1 + 2 step 1 until N2 do
  if x[i] = x[i-2] then
  begin C[i] := y[i]x0.5; B[i] := B[i-1] end
  else C[i] := (B[i] - B[i-1])/(x[i] - x[i-2]);
comment  Solve the linear system with C[i+2] - C[i+1] as right-hand side;
if m > N1 then
begin
  p := C[N1] := E[m] := F[N1] := F[m-1] := F[m] := 0.0;
  C[N1+1] := C[N1+3] - C[N1+2]; D[N1+1] := 1.0/D[N1+1]
end m > N1;
for i := N1 + 2 step 1 until m do
```

```
begin
  q := D[i-1]xE[i-1];
  D[i] := 1.0/(D[i] - pxF[i-2] - qxE[i-1]);
  E[i] := E[i] - qxF[i-1];
  C[i] := C[i+2] - C[i+1] - pxC[i-2] - qxC[i-1];
  p := D[i-1]xF[i-1]
end i;
m := N1 + 1; C[N2-1] := C[N2] := 0.0;
for i := N2 - 2 step -1 until m do
  C[i] := (C[i] - E[i]xC[i+1] - F[i]xC[i+2])xD[i];
comment Integrate the third derivative of S(x);
m := N2 - 1;
q := x[N1+1] - x[N1]; r := x[N1+2] - x[N1+1]; b1 := B[N1+1];
q3 := qxqxq; qr := q + r;
v := t := if qr = 0.0 then 0.0 else C[N1+1]/qr;
F[N1] := if q = 0.0 then 0.0 else v/q;
for i := N1 + 1 step 1 until m do
begin
  p := q; q := r;
  r := if i = N2 - 1 then 0.0 else x[i+2] - x[i+1];
  p3 := q3; q3 := qxqxq; pq := qr; qr := q + r;
  s := t; t := if qr = 0.0 then 0.0 else (C[i+1] - C[i])/qr;
  u := v; v := t - s;
  if pq = 0.0 then
  begin C[i] := 0.5xy[i+1]; D[i] := E[i] := F[i] := 0.0 end
  else
  begin
    F[i] := if q = 0.0 then F[i-1] else v/q;
    E[i] := 5.0xs;
    D[i] := 10.0x(C[i] - qxs);
    C[i] := D[i]x(p - q) + (B[i+1] - B[i] + (u - E[i])xp3
            - (v + E[i])xq3)/pq;
    B[i] := (px(B[i+1] - vxq3) + qx(B[i] - uxp3))/pq
            - pxqx(D[i] + E[i]x(q - p))
  end pq < > 0
end i;
```

```
  comment  End points x[N1] and x[N2];
  p := x[N1+1] - x[N1]; s := F[N1]×p×p×p;
  E[N1] := D[N1] := 0.0;
  C[N1] := C[N1+1] - 10.0×s;
  B[N1] := b1 - (C[N1] + s)×p;
  q := x[N2] - x[N2-1]; t := F[N2-1]×q×q×q;
  E[N2] := D[N2] := 0.0;
  C[N2] := C[N2-1] + 10.0×t;
  B[N2] := B[N2] + (C[N2] - t)×q
end QUINAT;
```

APPENDIX II

Algol 60 procedure QUINEQ

```
procedure QUINEQ(N1,N2) data:(y) result:(B,C,D,E,F);
  value N1,N2; integer N1,N2; array y,B,C,D,E,F;
comment  QUINEQ computes the coefficients of a quintic natural spline
  S(x) interpolating the ordinates y[i] at equidistant points x[i],
  i = N1 through N2.  For xx in [x[i],x[i+1]) the value of the spline
  function S(xx) is given by the fifth degree polynomial:
  S(xx) = ((((F[i]×t + E[i])×t + D[i])×t + C[i])×t + B[i])×t + y[i]
  with t = (xx - x[i])/(x[i+1] - x[i]).
  Input:
    N1,N2 subscript of first and last data point respectively, it is
          required that N2 > N1 + 1,
    y[N1:N2] the given function values (ordinates).
  Output:
    B,C,D,E,F[N1:N2] arrays collecting the coefficients of the quintic
          natural spline S(xx) as described above.  Specifically
          B[i] = S'(x[i]), C[i] = S"(x[i])/2, D[i] = S"'(x[i])/6,
          E[i] = S""(x[i])/24, F[i] = S^V(x[i]+0)/120.  F[N2] is neither
          used nor altered.  The arrays y,B,C,D must always be distinct.
          If E and F are not wanted, the call QUINEQ(N1,N2,y,B,C,D,D,D)
          may be used to save storage locations;
if N2 > N1 + 1 then
begin
  integer i,n;
  real p,q,r,s,t,u,v;
  n := N2 - 3; p := q := r := s := t := 0.0;
  for i := N1 step 1 until n do
  begin
    u := p×r; B[i] := 1.0/(66.0 - u×r - q);
    C[i] := r := 26.0 - u;
    D[i] := y[i+3] - 3.0×(y[i+2] - y[i+1]) - y[i] - u×s - q×t;
    q := p; p := B[i]; t := s; s := D[i]
  end i;
```

```
D[N1+1] := D[N1+2] := 0.0;
for i := n step -1 until N1 do
  D[i] := (D[i] - C[i]xD[i+1] - D[i+2])xB[i];
n := N2 - 1; q := 0.0; r := t := v := D[N1];
for i := N1 + 1 step 1 until n do
begin
  p := q; q := r; r := D[i]; s := t;
  F[i] := t := p - q - q + r;
  E[i] := u := 5.0x(-p + q);
  D[i] := 10.0x(p + q);
  C[i] := 0.5x(y[i+1] + y[i-1] + s - t) - y[i] - u;
  B[i] := 0.5x(y[i+1] - y[i-1] - s - t) - D[i]
end i;
  F[N1] := v; E[N1] := E[N2] := D[N1] := D[N2] := 0.0;
  C[N1] := C[N1+1] - 10.0xv; C[N2] := C[N2-1] + 10.0xt;
  B[N1] := y[N1+1] - y[N1] - C[N1] - v;
  B[N2] := y[N2] - y[N2-1] + C[N2] - t
end QUINEQ;
```

APPENDIX III

Algol 60 procedure QUINDF

```
procedure QUINDF(N1,N2) data:(x,y,yp) result:(C,D,E,F);
  value N1,N2; integer N1,N2; array x,y,yp,C,D,E,F;
comment QUINDF computes the coefficients of a quintic natural spline
  S(x) for which the ordinates y[i] and the first derivatives yp[i]
  are specified at points x[i], i = N1 through N2.  For xx in
  [x[i],x[i+1]) the value of the spline function S(xx) is given by
  the fifth degree polynomial:
  S(xx) = ((((F[i]×t + E[i])×t + D[i])×t + C[i])×t + yp[i])×t + y[i]
  with t = xx - x[i].
  Input:
    N1,N2 subscript of first and last data point respectively, it is
          required that N2 > N1,
    x,y,yp[N1:N2] arrays with x[i] as abscissa, y[i] as ordinate and
          yp[i] as first derivative at the i-th data point.  The
          elements of the array x must be strictly monotone increasing
          or decreasing.
  Output:
    C,D,E,F[N1:N2] arrays collecting the coefficients of the quintic
          natural spline S(xx) as described above.  E[N2] and F[N2] are
          neither used nor altered.  The arrays C,D,E,F must always be
          distinct;
if N2 > N1 then
begin
  integer i,m1,m2;
  real g,h,hh,p,pp,q,qq,t,u,v,w;
  array B[N1:N2];
  m1 := N1 + 1; m2 := N2 - 1;
  h := x[m1] - x[N1]; E[N1] := 4.0×h; F[N1] := 3.0×h;
  u := 0.0; v := 0.75;
  p := (y[m1] - y[N1])/(h×h); q := yp[m1]/h;
  B[N1] := 0.0; C[N1] := q - p - p + yp[N1]/h;
  for i := m1 step 1 until m2 do
```

```
  begin
    hh := h; h := x[i+1] - x[i];
    pp := p; p := (y[i+1] - y[i])/(hxh);
    qq := q; q := yp[i+1]/h; t := yp[i]/h;
    D[i] := 6.0x(hh + h) - uxhh - vxF[i-1];
    B[i] := p - t - qq + pp - uxB[i-1] - vxC[i-1];
    g := 3.0xh; w := g/D[i];
    E[i] := 4.0xh - wxg; F[i] := g - wxh;
    C[i] := q - p - p + t - wxB[i];
    u := h/D[i]; v := F[i]/E[i]
  end i;
  B[N2] := 0.0; t := C[m2] := C[m2]/E[m2];
  for i := m2 step -1 until m1 do
  begin
    B[i] := (B[i] - (3.0xC[i] + B[i+1])x(x[i+1] - x[i]))/D[i];
    C[i-1] := (C[i-1] - F[i-1]xB[i])/E[i-1]
  end i;
  for i := N1 step 1 until m2 do
  begin
    h := x[i+1] - x[i];
    F[i] := (B[i+1] - C[i] - C[i] + B[i])/(hxh);
    E[i] := 5.0x(C[i] - B[i])/h;
    D[i] := 10.0xB[i];
    C[i] := 0.5x(yp[i+1] - yp[i])/h - (7.5xB[i] + 5.0xC[i] + 2.5xB[i+1])xh
  end i;
  D[N2] := 0.0;
  C[N2] := 0.5x(yp[N2] - yp[m2])/h + 2.5x(B[m2] + t + t)xh
end QUINDF;
```