AD-767 970

NONLINEAR SPLINE FUNCTIONS

Michael A. Malcolm

Stanford University

Prepared for:

Office of Naval Research
National Science Foundation

June 1973

DISTRIBUTED BY:

NTIS

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

AD 762720

NONLINEAR SPLINE FUNCTIONS

by

Michael A. Malcolm

STAN-CS-73-372
June 1973

Approved for public release;
Distribution Unlimited

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
US Department of Commerce
Springfield, VA. 22151

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY (Corporate author) Stanford University Computer Science Department Stanford, California 94305	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3 REPORT TITLE

NONLINEAR SPLINE FUNCTIONS

4 DESCRIPTIVE NOTES (Type of report and inclusive dates)

technical report, June 1973

5 AUTHOR(S) (First name, middle initial, last name)

Michael A. Malcolm

6 REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1973	60	19

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-67-A-0112-0029 b. PROJECT NO. c. d.	STAN-CS-73-372 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10 DISTRIBUTION STATEMENT

Releasable without limitations on dissemination

11 SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13 ABSTRACT

A mathematical characterization of nonlinear interpolating spline curves is developed through a variational calculus approach, based on the Euler-Bernoulli large-deflection theory for the bending of thin beams or elastica. Algorithms previously used for computing discrete approximations of nonlinear interpolating splines are discussed and compared. The discrete natural cubic interpolating spline is discussed. An algorithm for computing discrete natural cubic splines is given and analyzed for discretization error and computational difficulty. Finally, a new algorithm together with its Fortran implementation is given for computing discrete nonlinear spline functions.

DD FORM 1 NOV 68 1473 (PAGE 1)

S/N 0101-807-6801

UNCLASSIFIED

Security Classification

NONLINEAR SPLINE FUNCTIONS

Michael A. Malcolm

June 1973

This work was supported by the Office of Naval Research, Contract N00014-67-A-0112-0029, and NSF Contract GJ29988X.

ABSTRACT

A mathematical characterization of nonlinear interpolating spline curves is developed through a variational calculus approach, based on the Euler-Bernoulli large-deflection theory for the bending of thin beams or elastica. Algorithms previously used for computing discrete approximations of nonlinear interpolating splines are discussed and compared. The discrete natural cubic interpolating spline is discussed. An algorithm for computing discrete natural cubic splines is given and analyzed for discretization error and computational difficulty. Finally, a new algorithm together with its Fortran implementation is given for computing discrete nonlinear spline functions.

NONLINEAR SPLINE FUNCTIONS

Contents

ACKNOWLEDGEMENTS

This work is part of a Ph.D. thesis which was directed by the late Professor George E. Forsythe. The author is deeply indebted to George Forsythe for help and encouragement throughout much of the research reported herein.

The author would also like to thank the following people who have helped in various ways: Richard Underwood, Gene Golub, David Stoutemyer, E. H. Lee, Maria Aurora Morgana, Linda Kaufman, Paul Concus, J. G. Herriot, Harold Stone and Victor Pereyra.

Special thanks are due Miss Mary Bodley for her skillful typing on both the thesis and this report.

1. Introduction

For the past 25 years and particularly the past decade, the subject of spline functions has been an area of great mathematical interest. The name spline comes from a very old technique in drafting in which a long thin strip of wood, called a draftsman's spline, is used to pass a smooth curve through a set of points in the euclidean plane. The points of interpolation are called knots and the spline is secured at the knots by means of lead weights called ducks. The wooden (and now plastic) splines and lead ducks are still manufactured; however, less expensive modern drafting tools are generally used today.

The mathematical model of a spline is a special case of the elastic line, or elastica, the first problem of any importance treated by the theory of elasticity. Its treatment began with James Bernoulli in 1705, Daniel Bernoulli (1742), Euler (1744), Kirchhoff, and many others. The history and theory are summarized by A.E.H. Love (1927); however, research papers on the elastica have been published more recently.

Perhaps the simplest way to characterize a spline mathematically is with the fact that a spline assumes a shape which minimizes its elastic strain energy. Daniel Bernoulli (1742) first suggested that the strain energy is proportional to the integral of the square of the curvature taken along the curve. He suggested in a letter to Euler that the differential equation of the elastica could be found by making the integral a minimum. Euler (1744), acting on this suggestion, was able to find the differential equation using techniques now known as Calculus of Variations and Lagrange multipliers. (It is interesting that Euler did this work when Lagrange was a small child.)

Since a modern development of Bernoulli's discovery could not be found in the literature, one is presented here. The reader is referred to Y. C. Fung (1965) for discussions of the basic concepts of solid mechanics used in the following discussion.

Assume that the spline is made from a linearly elastic material with coefficient of elasticity E. Let the spline be initially straight and assume that every cross section of the spline which is initially perpendicular to the major axis remains plane and perpendicular to the principal axis at all times. Let ds be the differential arc length along the neutral axis. When the beam is bent into a curve of radius R, the length of a filament, initially of length ds and parallel to the neutral axis, is changed by a factor of $1 + \eta/R$, where η is the distance from the neutral axis to the filament measured in a direction away from the center of curvature, as shown in Figure 1.1.

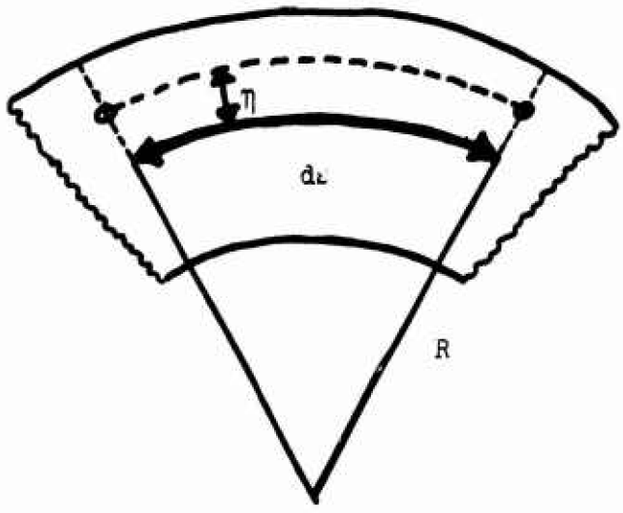


Figure 1.1. A spline element

Thus, the strain is η/R, or $\eta\kappa$, where κ is the curvature. If dA represents the cross-sectional area of the filament, $E\eta\kappa dA$ is the force (tension) acting on the filament. The resultant moment

of forces acting on the spline at the cross section is

$$M = \int_A \eta \cdot E\eta\kappa dA = E\kappa \int_A \eta^2 dA \ .$$

By definition, the moment of area of the cross section is

$$I = \int_A \eta^2 dA \ .$$

Therefore,

$$M = EI\kappa \ .$$

The angle through which the cross section rotates is κds . Hence, the work required to bend a segment ds of the spline to a curvature κ is given by

$$W_k = (EI/2)\kappa^2 ds \ . \tag{1.1}$$

There is a longitudinal force acting on the ends of the element. However, the strain energy resulting from this force is negligible. Hence, integrating throughout the spline, we have the strain energy

$$U = \tfrac{1}{2} \int_0^{\ell} EI\kappa^2 ds \ .$$

If the material is homogeneous, and the spline is of uniform cross section, then

$$U = \frac{EI}{2} \int_0^{\ell} \kappa^2 ds \ . \tag{1.2}$$

Now let $\mathcal{V}$ denote the potential energy of the system. In the general case

$$\mathcal{V} = \int_V W dv - \int_V \underset{\sim}{f}_i \underset{\sim}{u}_i dv - \int_S \underset{\sim}{t}_i \underset{\sim}{u}_i dS$$

where W is the strain energy function, $\underset{\sim}{f}_i$ are the body forces per unit volume, $\underset{\sim}{t}_i$ are the surface traction forces per unit area, and $\underset{\sim}{u}_j$ are the displacements. The repeated subscript is the usual tensor notation for summation over all possible values of the subscript. We are assuming that the $\underset{\sim}{f}_i = \underset{\sim}{0}$. Now, for this case, the $\underset{\sim}{t}_i$ correspond to the forces acting upon the spline at the nodes. But the spline is constrained at each node, so that either $\underset{\sim}{u}_i = \underset{\sim}{0}$, or $\underset{\sim}{u}_i$ is orthogonal to $\underset{\sim}{t}_i$. Thus, since the nodes cannot apply torques to the spline, they can do no work on it, or

$$\int_S \underset{\sim}{t}_i \underset{\sim}{u}_i dS = 0 .$$

Hence,

$$\mathcal{V} = \int_V W dv = U .$$

The principle of virtual work states that when a body is in equilibrium,

$$\delta \mathcal{V} = 0 .$$

Therefore,

$$\delta U = 0 , \tag{1.3}$$

or

$$\int_0^{\ell} \kappa^2 ds = \text{a local minimum} . \tag{1.4}$$

Of all the continuous curves which pass in turn through a given set of ordered points in the euclidean plane, having continuously turning tangents, and piecewise continuous curvature with discontinuities in curvature permitted on only a finite set of points, those satisfying (1.4) are referred to as <u>nonlinear interpolating spline functions</u>, or

simply, nonlinear splines. Birkhoff and de Boor (1965) gave an example which denies the existence of nonlinear splines in general. (See Section 3.2 for a discussion of this example.) It is an open question whether uniqueness holds for a given set of interpolation points. This question is complicated by the fact that the value of (1.4) can be made as small as desired by introducing large loops between supports, which modify the topology of the spline. The only known existence results for nonlinear splines are given in Section 3.2.

For deflections of the elastica which result in small slopes, $\kappa \cong y''$ and $ds \cong dx$. With this well known linearizing approximation, Equation (1.4) leads to the linear fourth-order ordinary differential equation known as the beam equation. The corresponding problems of interpolation have unique solutions called natural cubic spline functions. These solutions satisfy the natural boundary conditions of zero second derivatives at the ends. Existence and uniqueness for "linear" cubic splines follow from basic theorems of linear elasticity.

As one would intuitively expect, nonlinear splines are invariant under rigid rotations of the x-y coordinate system. However, the linear cubic splines are not invariant under rotations of the x-y coordinate system, and hence they are not well suited to fitting geometrical data in a euclidean plane, or other data where rigid rotations of the coordinate system make sense. On the other hand, nonlinear splines are not invariant with respect to changes in the y coordinate only, while linear splines are (see Podolsky and Denman (1964)).

As pointed out by Lee and Forsythe (1973), the term nonlinear spline has been used variously in the literature. However, in this case, the term indicates that the spline satisfies nonlinear Euler equations

which will be derived in the next section.

Little of the classical work on the elastica is directly applicable to nonlinear spline theory. The earliest known paper on nonlinear splines is that of Birkhoff and de Boor (1965). Early reports were written by Fowler and Wilson (1963) and Birkhoff, Burchard and Thomas (1965). Methods for computing nonlinear splines have been published by Glass (1966), Larkin (1966), and Woodford (1969). Mehlum (1969), in his Ph.D. dissertation, discusses nonlinear splines and presents an algorithm for approximating a nonlinear spline by a succession of circular arcs. Hosaka (1967) describes how to solve (1.4) and generate nonlinear spline functions on a differential analyzer. Some of these algorithms will be discussed in Section 3.

Methods for computing nonlinear splines have applications in the design of aircraft, ships and automobiles. A piece of sheet metal which is bent into an axially-symmetric configuration has a cross section which behaves like a nonlinear spline.

2. A Variational Formulation of the Large-Deflection, Small-Strain Problem of Interpolating Elastica

Given a fixed sequence of points $P_1, P_2, \ldots, P_n$, we define the interpolating elastica as the function $P(s)$, a function of arc-length s, which satisfies the equilibrium conditions for a thin beam, of constant cross section and constant linearly elastic properties, passing in turn through the n points P_i and acted on only by workless constraints at the points P_i. As we shall see, the natural boundary conditions resulting from the variational formulation will indicate exactly what kinds of boundary conditions are admissible and what combinations of them result in well-posed problems. We will also see that the forces $\underset{\sim}{f}_i$ acting on the elastica through the points P_i must satisfy certain restrictions which are entirely consistent with the assumptions of Section 1.

It is natural to specify each P_i by a pair of Cartesian coordinates (x_i, y_i); however a more efficient coordinate system for the development of the problem is (θ, s), where s is the arc length and $\theta = \theta(s)$ is the angle made at s by the curve $P(s)$ with some reference line. The curvature of the elastica at any point s is given by $\kappa(s) = d\theta/ds$.

In Section 1 it was shown that stable configurations of the elastica correspond to local minima of the functional

$$\int_0^{\ell} \kappa^2 ds , \tag{2.1}$$

where ℓ is the length of the elastica and the integral is taken along the deformed configuration. Note that ℓ is allowed to vary in minimizing

(2.1). It is equivalent to say

$$\delta \int_0^{\ell} \kappa^2 ds = 0 . \tag{2.2}$$

For the trivial case of collinear P_i and all $\underset{\sim}{f}_i = \underset{\sim}{0}$, the value of (2.1) for the equilibrium state is 0. To find the function $\theta(s)$ which satisfies (2.2) for a nontrivial set of P_i, the additional 2n-2 side conditions

$$\begin{aligned} &\int_{\ell_i}^{\ell_{i+1}} \cos\theta \, ds = x_{i+1} - x_i , \quad i=1, \ldots, n-1 , \\ &\int_{\ell_i}^{\ell_{i+1}} \sin\theta \, ds = y_{i+1} - y_i , \quad i=1, \ldots, n-1 , \end{aligned} \tag{2.3}$$

must be added to (2.2), where ℓ_i denotes the value of s at P_i. Using Lagrange multipliers, the constraints (2.3) can be added to (2.2) yielding

$$\delta \left\{ \sum_{i=1}^{n-1} \left[\int_{\ell_i}^{\ell_{i+1}} (\kappa^2 + \lambda_i \cos\theta + \mu_i \sin\theta) ds - \lambda_i (x_{i+1} - x_i) - \mu_i (y_{i+1} - y_i) \right] \right\} = 0 .$$

Now since certain kinds of constraints at the P_i allow the elastica to slide, the lengths ℓ_i are allowed to vary as well as the function θ.

Taking continuous variations, integrating by parts and rearranging terms gives

$$\sum_{i=1}^{n-1} \int_{\ell_i}^{\ell_{i+1}} \left[-2 \frac{d\kappa}{ds} - \lambda_i \sin\theta + \mu_i \cos\theta \right] \delta\theta \, ds$$

$$+ 2\kappa_n^- \delta\theta_n - 2\kappa_1^+ \delta\theta_1$$

$$+ 2 \sum_{i=2}^{n-1} (\kappa_i^- - \kappa_i^+) \delta\theta_i$$

$$+ [(\kappa_n^-)^2 + \lambda_{n-1} \cos\theta_n^- + \mu_{n-1} \sin\theta_n^-] \delta\ell_n \qquad (2.4)$$

$$- [(\kappa_1^+)^2 + \lambda_1 \cos\theta_1^+ + \mu_1 \sin\theta_1^+] \delta\ell_1$$

$$+ \sum_{i=2}^{n-1} [(\kappa_i^-)^2 - (\kappa_i^+)^2$$

$$+ \lambda_{i-1} \cos\theta_i^- + \mu_{i-1} \sin\theta_i^-$$

$$- \lambda_i \cos\theta_i^+ - \mu_i \sin\theta_i^+] \delta\ell_i = 0 \, ,$$

where the notation ξ_i^+ denotes $\lim_{\epsilon \to 0} \xi(\ell_i + \epsilon)$, and ξ_i^- denotes $\lim_{\epsilon \to 0} \xi(\ell_i - \epsilon)$.

By the fundamental lemma of the Calculus of Variations, the integral term in (2.4) yields

$$-2 \frac{d\kappa}{ds} - \lambda_i \sin\theta + \mu_i \cos\theta = 0 \, , \quad \ell_i < s < \ell_{i+1} \, , \qquad (2.5)$$

for $i = 1,\ldots,n-1$. Using (2.3), Equation (2.5) can be integrated for each open interval yielding

$$\kappa(s) = \kappa_i^+ - \frac{\lambda_i}{2} (y - y_i) + \frac{\mu_i}{2} (x - x_i) \, , \quad i=1,\ldots,n-1 \, . \qquad (2.6)$$

Equation (2.6) also results from the static equilibrium moment relation for a segment of spline to the right of the i-th knot (see Figure 2.1).

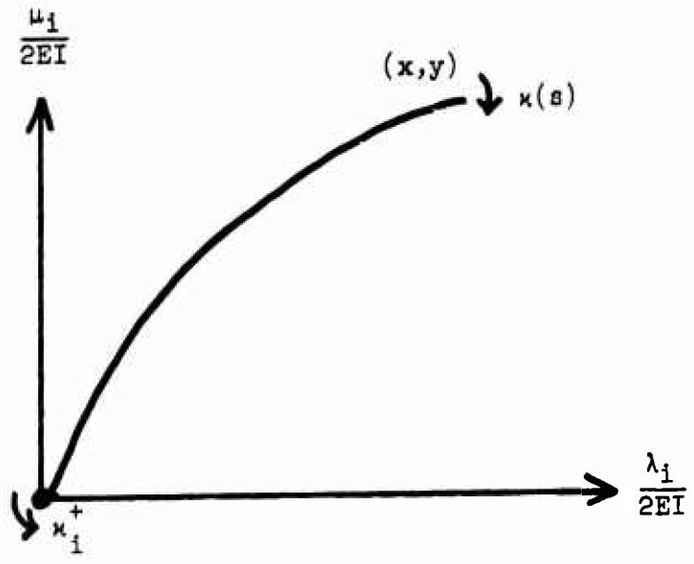


Figure 2.1. A segment of spline to the right of the i-th knot

Hence, the Lagrange multipliers are scaled force components acting on the spline at ℓ_i^+ .

The Euler equation can be deduced by differentiating (2.5) which gives

$$\frac{d^2\kappa}{ds^2} = \left(-\frac{\lambda_i}{2} \cos\theta - \frac{\mu_i}{2} \sin\theta \right) \kappa \ , \qquad (2.7)$$

So, if $\kappa \neq 0$,

$$\frac{d}{ds}\left(\frac{1}{\kappa} \frac{d^2\kappa}{ds^2} \right) = \left(\frac{\lambda_i}{2} \sin\theta - \frac{\mu_i}{2} \cos\theta \right) \kappa = -\kappa \frac{d\kappa}{ds} \ ,$$

or

$$\frac{d}{ds}\left(\tfrac{1}{2}\kappa^2 + \frac{1}{\kappa} \frac{d^2\kappa}{ds^2} \right) = 0 \ .$$

Hence, for $\kappa \neq 0$,

$$\frac{1}{\kappa} \frac{d^2\kappa}{ds^2} = -\tfrac{1}{2}\kappa^2 + c_i \ , \ \ell_i < s < \ell_{i+1} \ , \ i = 1,\ldots,n-1 \ , \qquad (2.8)$$

for constants c_i depending, in general, upon the interval i .

The remaining terms of (2.4) yield the natural boundary conditions. Since each of the $\delta\theta_i$ and $\delta\ell_i$ are arbitrary (for $i=1,\ldots,n$), the knots of the spline can be modeled by frictionlessly rotating sliders, as shown in Figure 2.2.

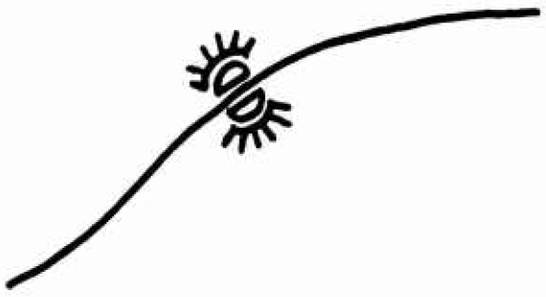


Figure 2.2. Frictionlessly rotating slider

The terms

$$2\kappa_n^-\delta\theta_n - 2\kappa_1^+\delta\theta_1$$

in (2.4) give the natural end conditions of zero curvature. The terms

$$2\sum_{i=2}^{n-1}(\kappa_i^- - \kappa_i^+)\delta\theta_i$$

demand that the curvature be continuous across each interior knot. If the angle θ is taken with reference to the x-axis, we have from (2.7)

$$T(s) = \frac{1}{\kappa}\frac{d^2\kappa}{ds^2}, \tag{2.9}$$

where $T(s)$ denotes the (scaled) force transmitted along the spline. (Tension is positive.) Thus, the terms

$$[(\kappa_n^-)^2 + \lambda_{n-1}\cos\theta_n^- + \mu_{n-1}\sin\theta_n^-]\delta\ell_n$$

$$-[(\kappa_1^+)^2 + \lambda_1\cos\theta_1^+ + \mu_1\sin\theta_1^+]\delta\ell_1$$

in (2.4) give the additional natural end conditions $T(\ell_1^+) = T(\ell_n^-) = 0$. That is, the ends of the spline have no longitudinal forces applied to them. Similarly, the terms

$$\sum_{i=2}^{n-1}[(\kappa_i^-)^2 - (\kappa_i^+)^2 + \lambda_{i-1}\cos\theta_i^- + \mu_{i-1}\sin\theta_i^- - \lambda_i\cos\theta_i^+ - \mu_i\sin\theta_i^+]\delta\theta_i$$

require that the forces $T(\ell_i^-) = T(\ell_i^+)$ $(i=2,\ldots,n-1)$, that is, the longitudinal force transmitted by the spline is continuous across each interior knot.

The constants of integration c_i $(i=1,\ldots,n-1)$ in (2.8) can now be determined for the natural open spline. Since $T(\ell_1^+) = \kappa_1^+ = 0$, (2.8) evaluated at ℓ_1^+ gives $c_1 = 0$. Both the curvature and the longitudinal force are continuous across each of the interior knots, hence we have $c_i = 0$ for $i=2,\ldots,n-1$. Thus we obtain the Euler equations

$$\frac{d^2\kappa}{ds^2} + \frac{1}{2}\kappa^3 = 0 \ , \quad \ell_i < s < \ell_{i+1} \ , \tag{2.10}$$

for $i=1,\ldots,n-1$, which, in view of (2.7), must be satisfied by the spline at every point within each open interval. Equations (2.10) were first published by Birkhoff and deBoor (1965).

The above analysis shows that the nonlinear spline having frictionlessly rotating slider constraints corresponds to the least-constraint interpolating elastica. Any further constraints on the elastica must be consistent with (2.4) and will produce a larger value of (2.1) unless they are also consistent with the nonlinear spline. For example, Hermite constraints are consistent with (2.4) since they imply $\delta\theta_i = 0$, $(i=1,\ldots,n)$. However, for this case, curvature is not necessarily continuous across the interior knots and, hence, (2.10) no longer holds. Numerous combinations of more restrictive boundary conditions are admissable according to (2.4). These include such things as fixed ends and pinned joints. We will only be concerned with the least-constraint, natural boundary conditions of the nonlinear interpolating splines.

Closed nonlinear splines have been studied by Lee and Forsythe (1973). In this case, $x_1 = x_n$ and $y_1 = y_n$. However, to avoid having to prescribe the total length of the spline, the variations $\delta\ell_1$, and $\delta\ell_n$ are taken

as unequal. Since elements of the curved arc are inserted into or deleted from the spline at $P_1 = P_n$, the variations at ℓ_1 and ℓ_n must satisfy

$$\delta\theta_1 + \kappa_1^+ \delta\ell_1 = \delta\theta_n + \kappa_n^- \delta\ell_n \ .$$

The corresponding natural boundary conditions are

$$\kappa_1^+ = \kappa_n^-$$

and

$$(\kappa_1^+)^2 + 2T_1^+ = (\kappa_n^-)^2 + 2T_n^- = 0 \ .$$

It follows that (2.10) also holds for the case of closed nonlinear splines.

3. Methods for Computing Open Nonlinear Spline Functions

A number of methods for computing nonlinear interpolating splines have been published during the past six years. In this section these results are briefly reviewed in chronological order. Also, two interesting results due to Larkin, which concern the existence of finite equilibrium solutions, are treated in Section 3.2.

3.1 Glass' method

The first published method for computing discrete approximations to the nonlinear interpolating spline is that of Glass (1966).

Glass uses Cartesian coordinates (x-y) and considers the Euler equation analogous to (2.10):

$$y^{(iv)} = \frac{5(y'')^3 + 20y'y''y'''}{2[1 + (y')^2]} - \frac{35(y')^2(y'')^3}{2[1 + (y')^2]^2} \qquad (3.1)$$

The knots are represented by their coordinates (x_i, y_i), i=1,2,...,n . Since the curvature

$$\kappa = \frac{y''}{[1 + (y')^2]^{3/2}} ,$$

and $ds = \sqrt{1 + (y')^2}\, dx$, the energy given by (2.1) becomes

$$E(y) = \sum_{i=1}^{n-1} \int_{x_i}^{x_{i+1}} \frac{(y'')^2}{[1 + (y')^2]^{5/2}}\, dx . \qquad (3.2)$$

Glass first considers the problem on one panel $[x_i, x_{i+1}]$, with the end points y_i and y_{i+1} prescribed. He assumes the end slopes y_i' and y_{i+1}' are also prescribed and proceeds to develop a method to solve

the ordinary differential equation (3.1), subject to these four boundary conditions. Using a Taylor's series expansion of (3.1) about an approximate solution $y^{(0)}$, he arrives at a linear differential equation in the correction to $y^{(0)}$ to obtain the better approximate solution $y^{(1)}$. Using central difference operators over a mesh of N points in the interval (with constant mesh size H), he obtains a system of linear difference equations in the correction terms at each mesh point. This 5-band system of linear algebraic equations is solved at each iteration until convergence.

The solution of the fourth-order boundary value problem is carried out in each panel for a given set of slopes at the knots: y_i', i=1,2,...,n, written $\underset{\sim}{y}'$. The functional (3.2) is approximated using this discrete solution and is now considered to be $E(\underset{\sim}{y}')$. Glass then proceeds to a solution with an outer iteration which minimizes $E(\underset{\sim}{y}')$ using a gradient method. He uses differencing to obtain partial derivatives and then rounds them to 1, 0, or -1. He uses an unusual heuristic method for choosing the step parameter λ for moving in the direction $-\text{grad}[E(\underset{\sim}{y}')]$.

Glass doesn't present the actual program; however, he claims that his method requires only about five gradient searches for satisfactory answers. He presents an example with nine panels (n=10), but does not mention the value of N or how the initial y' was obtained. For this example, Glass' method took approximately 10 minutes on the IBM 7094.

In addition to the "enormous computation time," Glass mentioned that instabilities occur in problems where the spline turns too rapidly. He suggests a technique for overcoming the instabilities at the expense of a substantial increase in computation time. The computational difficulties increase drastically for larger numbers of data points.

3.2 Larkin's method

Larkin (1966) presented perhaps the most interesting study of non-linear spline functions in a rather obscure unpublished report. His results and suggested computational method are summarized here.

Larkin first shows that (3.1) may be reduced to

$$\kappa = \frac{d\theta}{ds} = \lambda_i \sqrt{\cos(\theta - \epsilon_i)} \ , \quad \ell_i \leq s \leq \ell_{i+1} \ , \tag{3.3}$$

where the λ_i and ϵ_i are constants of integration. Equation (3.3) is precisely the energy integral of the equations of motion of Kirchhoff's kinetic analogue of a pendulum given in Love (1927), p. 401, which may also be written as

$$\kappa^2 = A_i \cos\theta + B_i \sin\theta \ , \quad \ell_i \leq s \leq \ell_{i+1} \ , \tag{3.4}$$

where the A_i and B_i are the constants of integration. Equation (3.4) is easily shown to satisfy (2.10) if there are no points of inflection.

Larkin integrates (3.3) to give $x(s)$ and $y(s)$ in terms of elliptic integrals. To obtain equations which can be used for computation, he needs a separate treatment for the case where the spline has a point of inflection. A point of inflection occurs when the curvature vanishes, i.e. where

$$\theta = \epsilon_i \pm \pi/2 \ . \tag{3.5}$$

Larkin's complete algorithm is quite complicated and goes roughly as follows:

Step 1: Estimate values of θ at the knots (i.e., θ_i , i=1,2,...,n).

Step 2: Determine the parameters λ_i and ϵ_i , for each panel. This requires the evaluation of three complicated functions, each involving elliptic integrals, as well as several trigonometric functions. Based upon the values of these functions, it can be determined whether or not a

point of inflection occurs in the interval, and if so, which one of two cases it corresponds to. The value of ϵ_i is then determined by finding the zero of the appropriate complicated transcendental equation. The value of λ_i is then obtained by the direct evaluation of a similar equation.

Step 3: Scan through the knots to find the one at which the largest discontinuity in curvature occurs. Replace the value of θ_i there by a value which makes the curvature continuous. This requires solving for the zero of another transcendental equation, and the values of ϵ_i and λ_i must be recomputed in each of the neighboring panels. Step 3 is repeated until the largest discontinuity in curvature becomes smaller than a specified tolerance.

The above algorithm was implemented on a KDF9 computer and graphs of a few solutions are included in Larkin's report. No mention is made of running times, or computational experience. However, it appears that the method is probably quite slow due to the large number of transcendental functions which must be evaluated.

The most interesting results in Larkin's report are contained in the following two theorems:

Theorem 3.1 (Larkin): A necessary condition for the existence of a finite equilibrium solution is

$$|\theta_{i+1} - \theta_i| \leq \pi , \quad i=1,2,\ldots,n-1 . \tag{3.6}$$

Proof: From Kirchhoff's analogy (3.3),

$$\kappa^2 = \lambda_i^2 \cos (\theta - \epsilon_i) \geq 0 .$$

Thus, $\cos (\theta - \epsilon_i) \geq 0$. Therefore, at ℓ_i and ℓ_{i+1} , we have

$$|\theta_i - \epsilon_i| \leq \pi/2 + 2p\pi ,$$

and

$$|\theta_{i+1} - \epsilon_i| \leq \pi/2 + 2q\pi ,$$

where p and q are integers. The only cases of interest are those where $p = 0$, and ϵ_i is evaluated so that $p = q = 0$. So we have

$$|(\theta_{i+1} - \epsilon_i) - (\theta_i - \epsilon_i)| \leq |\theta_{i+1} - \epsilon_i| + |\theta_i - \epsilon_i| ,$$

or $|\theta_{i+1} - \theta_i| \leq \pi$.

Unfortunately, this theorem doesn't provide a way to determine a priori whether a given set of knots has a finite equilibrium solution. However, this result is useful in an algorithm for determining when a solution is diverging. The cases where (3.6) is not satisfied correspond physically to a spline which continues to slide through the supports while reaching lower and lower energy values. An example of this is given in Birkhoff, Burchard and Thomas (1965). The knots in this example have the Cartesian coordinates (1,0), (2,0), (0,2), (0,1), with the spline threaded through them in that order. The spline never reaches an equilibrium state, and continues expanding to infinity. A lucid explanation of this example is given in Lee and Forsythe (1973).

Theorem 3.2 (Larkin): A necessary condition for the existence of a finite equilibrium which possesses no point of inflection is that there exists an integer m such that

$$\min (\theta_i,\theta_{i+1}) \leq \psi_i + m\pi \leq \max (\theta_i,\theta_{i+1}) , \qquad (3.7)$$

where ψ_i is the inclination of the straight line connecting the i-th and the (i+1)-th knots.

Proof: Roughly, this theorem follows from the fact that θ is a continuous function which must vary monotonically from θ_i to θ_{i+1} in the case that there is no point of inflection. And (3.7) is untrue only if θ is not monotonic. For a detailed proof, see Larkin (1966), Appendix A.

3.3 Woodford's method

The only published program for computing nonlinear splines is an Algol procedure given by Woodford (1969). His method appears to be considerably simpler and faster than those of Glass and Larkin.

Woodford uses cartesian coordinates and applies the calculus of variations to obtain (3.1). He notes that (3.1) can be reduced to

$$(y'')^2 = (A_i y' + B_i)(1 + (y')^2)^{5/2} \tag{3.8}$$

which is merely another form of Kirchhoff's kinetic analogue given by (3.3) and (3.4). He uses (3.8) to obtain an expression for the energy

$$E = \sum_{i=1}^{n-1} |A_i(y_{i+1} - y_i) + B_i(x_{i+1} - x_i)| , \tag{3.9}$$

which is easily derived using (3.4) and (2.1).

Woodford discretizes the intervals with a uniform mesh and starts with an initial solution $y^{(0)}$ provided by solving

$$y^{(iv)} = 0 ,$$

which gives the usual natural cubic interpolating spline. He uses the method of quasi-linearization and proceeds with an iteration based on the same Taylor series expansion about the current solution that Glass used. To solve this linear multi-boundary value problem, Woodford uses a derivative replacement scheme based on expressing y_{k+1}, y'_{k+1}, y''_{k+1}, and y'''_{k+1} with Taylor series expansions about the point y_k, immediately to the

left, and retaining all terms up through those in $y^{(iv)}$. This results in a 9-band linear algebraic system of equations to solve at each iteration. The iterations are terminated when the energy given by (3.9) on two successive iterations is nearly the same.

Woodford gives an example using seven data points: (0,0), (1,1.9), (2,2.7), (3,2.6), (4,1.6), (5,0.8), (6,1.2), and a mesh size of 0.025 (40 points per panel). Convergence occurred after four iterations. His example will be discussed again in Section 5. A listing of a Fortran translation of Woodford's Algol procedure follows.

```
      SUBROUTINE MEC(N, ORD, H, K, A, B, C, D, EPS, *)
C
C  THIS SUBROUTINE COMPUTES POINTS ON THE MINIMUM-ENERGY CURVE
C  (SOMETIMES CALLED A NONLINEAR SPL    WHICH PASSES THROUGH
C  THE N DATA POINTS WITH ORDINATES ORD(I), I=1,...,N, AND
C  EQUIDISTANT ABSCISSAE.  THE SOLUTION IS PRODUCED BY A STEP-BY-
C  STEP METHOD BASED ON TAYLOR SERIES.  THE NUMBER OF STEPS BETWEEN
C  CONSECUTIVE PAIRS OF DATA POINTS IS K, AND H IS THE LENGTH OF
C  EACH STEP.  USUALLY THE NUMBER OF STEPS K SHOULD BE AT LEAST 10
C  PER UNIT INTERVAL.  THE PARAMETER EPS CONTROLS THE CONVERGENCE,
C  USUALLY EPS SHOULD BE OF THE ORDER 1.E-5.  THE SOLUTION Y AND ITS
C  DERIVATIVES ARE STORED IN THE ARRAYS A, B, C, AND D.  A(I)
C  IS THE VALUE OF Y AT THE I-TH POINT, I=1,...,K*(N-1)+1.  SIMILARLY
C  B(I) IS THE VALUE OF THE FIRST DERIVATIVE, C(I) THE SECOND
C  DERIVATIVE AND D(I) THE THIRD DERIVATIVE.  SINCE WE ALLOW FOR
C  DISCONTINUITIES IN THE THIRD DERIVATIVE AT EACH INTERNAL DATA
C  POINT, D(I) FOR I = K+1, 2K+1, ..., (N-1)K+1 IS THE THIRD
C  DERIVATIVE TO THE IMMEDIATE RIGHT OF THE DATA POINT.
C  D((N-1)*K+1) IS LEFT UNDEFINED.  THE  SUBROUTINES BANDET AND
C  BANDSL MUST BE SUPPLIED FOR SOLVING A 5-BAND LINEAR SYSTEM
C  AT EACH ITERATION.  THE SUBROUTINE EXITS WITH A RETURN 1 IF
C  THE PROCEUDRE BANDET FINDS A ZERO PIVOT OR IF  THE  ALGORITHM
C  HAS NOT CONVERGED AFTER 10 ITERATIONS.
C
C  THIS SUBROUTINE IS A TRANSLATION OF AN ALGOL PROCEDURE GIVEN
C  BY C.H. WOODFORD IN "SMOOTH CURVE INTERPOLATION", BIT 9 (1969),
C  69-77.
C
C  MICHAEL A. MALCOLM
C  COMPUTER SCIENCE DEPARTMENT
C  STANFORD UNIVERSITY
C  AUGUST 16, 1971
C
      REAL*4  H, EPS, ORD(1), A(1), B(1), C(1), D(1), AL(1000,9),
     *  R(1000), EN(20), DEN(20), AA, BL, CL, DL, X, W, ZZ, XX, XX1,
     *  XX2, WW, WWW, D1, H1, H2, H3, H4, MBAND(1000, 4), NORM
      INTEGER Q, P, INT(1000)
      LOGICAL  L, Z
C
      D1 = 1.E10
      ITO = 0
      N1 = N-1
      H1 = H*K
      H2 = .5*H*H
      H3 = H2*H/3.
      H4 = H3*H*.25
      Q = N1 * (4*K-1)
      M1 = N1*K + 1
      M = M1-1
```

```
      DO 10 I=1,M1
         A(I) = 0.
         B(I) = 0.
         C(I) = 0.
         D(I) = 0.
10    CONTINUE
      EN(1) = 0.
      EN(N) = 0.
20    DO 30 I=1,Q
         DO 30 J=1,9
            AL(I,J) = 0.
30    CONTINUE
      I = 5
      J = 3
      DO 80 P=2,M
         L = (P/K)*K.EQ.P
         Z = ((P-1)/K)*K.EQ.(P-1)
         AA = A(P)
         BL = B(P)
         CL = C(P)
         DL = D(P)
         I1 = I+1
         I2 = I+2
         I3 = I+3
         I4 = 5-I
         I5 = 4-I
         I6 = 3-I
         I7 = 2-I
         IF (Z) GO TO 40
         AL(I,J+I4) = 1.
         J = J+1
40       AL(I,J+I4) = H + CL*H4
         AL(I1,J+I5) = 1. + CL*H3
         AL(I2,J+I6) = CL*H2
         IF (L) GO TO 42
         AL(I3, J+I7) = CL*H
42       J = J+1
         AL(I,J+I4) = H2 + BL*H4
         AL(I1,J+I5) = H + BL*H3
         AL(I2,J+I6) = 1. + BL*H2
         IF (L) GO TO 44
         AL(I3,J+I7) = H*BL
44       J = J+1
         AL(I,J+I4) = H3 + AA*H4
         AL(I1,J+I5) = H2 + AA*H3
         AL(I2,J+I6) = H + AA*H2
         IF (L) GO TO 46
         AL(I3,J+I7) = 1. + AA*H
46       J = J+1
         IF (L) GO TO 48
```

```
          AL(I,J+I4) = -1.
          J = J+1
 48       AL(I1,J+I5) = -1.
          J = J+1
          IF (P.NE.M) AL(I2,J+I6) = -1.
          J = J+1
          IF (L) GO TO 50
          AL(I3,J+I7) = -1.
 50       BL = DL*H4
          R(I1) = DL*H3
          R(I2) = DL*H2
          IF (.NOT.L) GO TO 60
          IPORD = P/K + 1
          R(I) = BL + ORD(IPORD)
          I = I3
          J = J-2
          GO TO 80
 60       IPORD = (P-1)/K + 1
          R(I) = BL
          IF (Z) R(I) = R(I) - ORD(IPORD)
          R(I3) = DL*H
          I = I+4
          J = J-3
 80    CONTINUE
       AA = A(1)
       CL = C(1)
       DL = D(1)
       AL(1,5) = H + CL*H4
       AL(1,6) = H3 + AA*H4
       AL(1,7) = -1.
       AL(2,4) = 1. + CL*H3
       AL(2,5) = H2 + AA*H3
       AL(2,7) = -1.
       AL(3,3) = CL*H2
       AL(3,4) = H + AA*H2
       AL(3,7) = -1.
       AL(4,3) = 1. + AA*H
       AL(4,2) = CL*H
       AL(4,7) = -1.
       R(1) = H4*DL - ORD(1)
       R(2) = H3*DL
       R(3) = H2*DL
       R(4) = H*DL
       CALL BANDET(MBAND, 1000, INT, AL, Q, 4, 9, &150)
       CALL BANDSL(MBAND, 1000, INT, R, AL, Q, 4, 9)
C  STORING CURRENT SOLUTION
       I = 4
       J = 2
       DEN(1) = R(1)
       DEN(N) = R(Q)
```

```
      DO 100 P=2,M
         X = R(I)
         W = R(I+1)
         ZZ = R(I+2)
         I = I+3
         IF ((P/K)*K.NE.P) I = I+1
         XX = X*X
         XX1 = 1./(1.+XX)
         XX2 = XX1 * XX1
         WW = W*W
         WWW = WW*W
         IF (((P-1)/K)*K.NE.(P-1)) GO TO 90
         EN(J) = WW*(XX1**2.5)
         DEN(J) = X
         J = J+1
 90      AA = 10.*X*W*XX1
         BL = (7.5*WW + 10.*X*ZZ)*XX1 - 52.5*XX*WW*XX2
         CL = 10.*W*ZZ*XX1 - X*WWW*XX2*(40. - 70.*XX*XX1)
     *        - 20.*XX*W*ZZ*XX2
         D(P) = -(2.5*WWW + 10.*X*W*ZZ) * XX1 + 17.5*XX*WWW*XX2
     *          +AA*ZZ + BL*W + CL*X
         A(P) = AA
         B(P) = BL
         C(P) = CL
 100  CONTINUE
C  TEST FOR CONVERGENCE
      X = 0.
      DO 110 I=1,N1
         AA = (EN(I+1) - EN(I))/(DEN(I+1) - DEN(I))
         BL = EN(I) - AA*DEN(I)
         X = X+ABS(AA*(ORD(I+1) - ORD(I)) + BL*H1)
 110  CONTINUE
      ITO = ITO+1
      NORM = ABS(D1-X)
      PRINT 115, ITO, NORM
 115  FORMAT(' ITER. NO. ', I4, '  NORM =', G10.3)
      IF (ABS(D1-X).LT.EPS) GO TO 120
      IF (ITO.EQ.10) RETURN 1
      D1 = X
      X = R(1)
      B(1) = 10.*X*R(2)/(1. + X*X)
      GO TO 20
 120  A(1) = ORD(1)
      B(1) = R(1)
      D(1) = R(2)
      I = 3
      J = K
      I1 = 2
      I2 = 2
```

```
125   DO 130 P = I2,J
         A(P) = R(I)
         B(P) = R(I+1)
         C(P) = R(I+2)
         D(P) = R(I+3)
         I = I+4
130   CONTINUE
      J = J+1
      A(J) = ORD(I1)
      I1 = I1+1
      B(J) = R(I)
      IF (I.EQ.Q) GO TO 140
      C(J) = R(I+1)
      D(J) = R(I+2)
      I = I+3
      J = J+K-1
      I2 = J+2-K
      GO TO 125
140   RETURN
150   RETURN 1
      END
```

```
      SUBROUTINE BANDET(M,IDIM,INT,A,N,M1,M3,*)
      DIMENSION M(IDIM,M1),INT(IDIM),A(IDIM,M3)
C
C     M-----AN NXM1 MATRIX FOR STORING LOWER TRIANGULAR
C           MATRIX OF LU DECOMPOSITION OF A
C     INT---AN NX1 VECTOR FOR RECORDING ROW INTERCHANGES
C           DURING DECOMPOSITION
C     A-----AN NX(M1+M2+1) MATRIX WHOSE COLUMNS ARE THE DIAGONALS
C           OF C, THE BAND MATRIX BEING DECOMPOSED
C           A(*,1) - A(*,M1)  ARE SUBDIAGONALS OF C
C           A(*,M1+1)         IS DIAGONAL OF C
C           A(*,M1+2) - A(*,M1+M2+1) ARE SUPERDIAGONALS OF C
C     N-----NUMBER OF ROWS IN A
C     M1----NUMBER OF SUBDIAGONALS IN C
C     M3----TOTAL NUMBER OF DIAGONALS IN C , I.E. WIDTH OF BAND
C           M3 = M1 (# SUBDIAGS) + M2 (# SUPERDIAGS) +1
C
C     BANDET AND BANDSL ARE TWO SUBROUTINES WHICH SOLVE C*X = B
C     WHEN C IS AN UNSYMMETRIC BAND MATRIX ( THEY WILL WORK WITH
C     SYMMETRIC BAND MATRICES BUT TAKE NO ADVANTAGE OF THEIR
C     STRUCTURE).
C
C     C HAS M1 SUBDIAGONALS AND M2 SUPERDIAGONALS.
C     THE MATRIX C IS TRANSFORMED TO A BY MAKING EACH DIAGONAL OF
C     C A COLUMN OF A.  THUS A IS NX(M1+M2+1) WHEN C IS NXN.
C     A TYPICAL A IS PICTURED BELOW.  C HERE IS 4X4, WITH
C     2 SUBDIAGONALS AND 1 SUPERDIAGONAL
C
C        0          0          C(1,1)          C(1,2)
C        0       C(2.1)        C(2,2)          C(2,3)
C   C(3,1)       C(3,2)        C(3,3)          C(3,4)
C   C(4,2)       C(4,3)        C(4,4)            0
C
C     THIS TRANSFORMATION IS THE FOLLOWING, ASSUMING THAT C(I,J)
C     IS A BAND ELEMENT IN C:
C     C(I,J) --> A(I,M1+1+(J-I))
C     ALL OTHER ELEMENTS OF A ARE 0
C
C     BANDET FINDS THE LU DECOMPOSITION OF A, STORING
C     THE LOWER TRIANGULAR MATRIX IN M AND NXM1 MATRIX,
C     AND OVERWRITING THE UPPER TRIANGULAR MATRIX INTO A.
C     BANDSL USES THIS DECOMPOSITION TO SOLVE A*X = B WHERE
C     THE RIGHTHAND SIDE IS INPUT IN THE VECTOR B, AND X IS
C     OUTPUT IN THE VECTOR B.
C
C     THESE ROUTINES WERE TRANSLATED FROM THOSE PRESENTED BY
C     J. WILKINSON IN NUMERISCHE MATHEMATIK VOL 9, P279
C     TRANSLATOR: BARBARA RYDER
C
```

```
      REAL M
25    L=M1
      DO 40 I=1,M1
            K2=M1+2-I
            DO 50 J=K2,M3
50          A(I,J-L)=A(I,J)
            L=L-1
            K2=M3-L
            DO 60 J=K2,M3
60          A(I,J)=0.0
40    CONTINUE
45    L=M1
      DO 70 K=1,N
      X=A(K,1)
      I=K
      K2=K+1
      IF (L.LT.N) L=L+1
72    IF (L.LT.K2) GO TO 81
79          DO 80 J=K2 ,L
            IF (ABS(A(J,1))-ABS(X)) 80,80,82
82          X=A(J,1)
            I=J
80          CONTINUE
81    INT(K)=I
      IF (X) 73,75,73
73    IF (I-K) 77,78,77
77          DO 90 J=1,M3
            X=A(K,J)
            A(K,J)=A(I,J)
90          A(I,J)=X
78          IF (L.LT.K2) GO TO 70
83          DO 100 J=K2 ,L
            M(K,J-K)=A(J,1)/A(K,1)
            X=M(K,J-K)
            DO 110 JJ=2,M3
110         A(J,JJ-1)=A(J,JJ)-A(K,JJ)*X
100          A(J,M3)=0.0
70    CONTINUE
      RETURN
75    RETURN 1
      END
```

```
      SUBROUTINE BANDSL (M,IDIM,INT,B,A,N,M1,M3)
      DIMENSION INT(IDIM),A(IDIM,M3),M(IDIM,M1),B(IDIM)
C
C     ALL PARAMETERS SAME AS IN BANDET EXCEPT FOR:
C     B-----RIGHTHAND SIDE OF LINEAR SYSTEM C*X = B
C           SOLUTION IS RETURNED IN B
C
      REAL M
      INTEGER W
      L=M1
      DO 10 K=1,N
      I=INT(K)
      IF (I-K) 11,12,11
 11   X=B(K)
      B(K)=B(I)
      B(I)=X
 12   K2=K+1
      IF (L.LT.N) L=L+1
 14   IF (L.LT.K2)  GO TO 10
 15        DO 20 I=K2,L
           X=M(K,I-K)
 20        B(I)=B(I)-X*B(K)
 10   CONTINUE
      L=1
      DO 30 II=1,N
      I=N+1-II
      X=B(I)
      W=I-1
      IF (L-1) 32,33,32
 32        DO 40 K=2,L
 40        X=X-A(I,K)*B(K+W)
 33   B(I)=X/A(I,1)
      IF (L-M3) 31,30,30
 31   L=L+1
 30   CONTINUE
      RETURN
      END
```

3.4 Mehlum's method

Even Mehlum (1969), in his Ph.D. dissertation, presented an algorithm for computing an approximation to nonlinear interpolating splines which he has implemented in a computer program called KURGLA (from Norwegian "KURveGLAtting" -- curve fitting). KURGLA is the basic subroutine in the ship fairing program included in the AUTOKON software which has been developed at the Central Institute for Industrial Research in Oslo, Norway. The AUTOKON computer software is used for numerical control of drawing machines and flame cutters in ship-building. The ship fairing program has been in use since 1965 and by 1969 had been used for fairing about 150 hulls.

Mehlum starts with a long derivation of the equation

$$\left(\frac{d\theta}{ds}\right)^2 = p \sin(\theta - \varphi) + \gamma \qquad (3.10)$$

which is yet another form of Kirchhoff's kinetic analogue (3.4). He also proves the following:

Theorem (3.3): Between neighboring knots of a nonlinear spline, there is always a direction along which the curvature varies linearly.

Proof: This follows from the fact that curvature is proportional to the bending moment which can be expressed as a linear function of x and y. (See Equation (2.6)). ∎

Mehlum makes the assumption that $\varphi = 0$, so that the direction along which the curvature varies linearly coincides with the x-axis. The curvature is then represented as a piecewise-linear continuous function which changes slope only at the knots. Mehlum further simplifies the problem by representing the linear curvature between each knot by a series of steps giving a piecewise constant function (noncontinuous, of course)

for the curvature. In other words, he approximates the solution to a somewhat different problem than (5.10) by a series of arcs of circles which form a continuous function with a continuous first derivative, but with discontinuous second derivatives. Combining this representation with (5.10) and integrating yields two constants of integration to be determined within each panel for a given set of curvatures specified at each knot. His algorithm thus breaks into an inner and an outer iteration. The outer iteration first guesses a set of knot curvatures and then tries to improve them based upon how close the resulting curves fit the data and specified end slopes. (He is not trying to produce natural splines.) The inner iteration then uses the current values for the knot curvatures and determines the constants of integration for each panel through some simple recurrence formulas so that the function interpolates the end knots. The output of Mehlum's subroutine is in the form of center coordinates and radii for the circles thus obtained.

Mehlum claims that many numerical control machines can use output of this form directly.

4. The Discrete Natural Cubic Interpolating Spline

The data points X_i , $Y_i = Y(X_i)$, $i = 1$, ... , n (where $X_1 < \ldots < X_n$) , may be interpolated as follows: Let

$$H_i = X_{i+1} - X_i \ , \ i = 1, \ \ldots \ , \ n-1 \ .$$

Assume that each of the H_i is an integral multiple of the mesh size parameter h . The interval $[X_1 - h \ , \ X_n + h]$ is divided into a partition $x_0 \ , \ x_1 \ , \ \ldots \ , \ x_m \ , \ x_{m+1}$, where

$$x_i = X_1 + (i-1)h \ , \ i = 0 \ , \ \ldots \ , \ m + 1 \ , \text{ and}$$

$$m = (X_n - X_1)/h + 1 \ .$$

Denote by X the set of abscissa data X_i , $i = 1$, ... , n , and let

$$L_1 = 0 \ ,$$

$$L_i = \sum_{j=1}^{i-1} H_j \ , \ i = 2 \ , \ \ldots \ , \ n \ .$$

The discrete natural cubic spline interpolating the points $(X_i \ , \ Y_i)$, $i = 1$, ... , n is defined to be the set of points $(x_i \ , \ y_i)$, $i = 0$, ... , $m + 1$, which minimize the value of S where

$$S = \sum_{i=1}^{m} \left(\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \right)^2 h$$

subject to the constraints of interpolation,

$$y_i = Y_j \ , \quad \text{if} \quad i = L_j/h + 1 \ .$$

Let δ denote the central difference operator. The mesh ordinates y_i , $i = 2 , \ldots , m-1$, are computed by solving the linear system of equations

$$\delta^4 y_i = 0 \quad \text{if} \quad x_i \notin X \quad \text{and} \quad 0 \leq i \leq m \ , \tag{4.1}$$

and

$$\delta^2 y_1 = \delta^2 y_m = 0 \ ,$$

which arise by setting $\partial S/\partial y_i = 0$, $i=0,2,\ldots,m+1$ (excluding every $i = L_j/h$, for $j=1,\ldots,n$), subject to the conditions

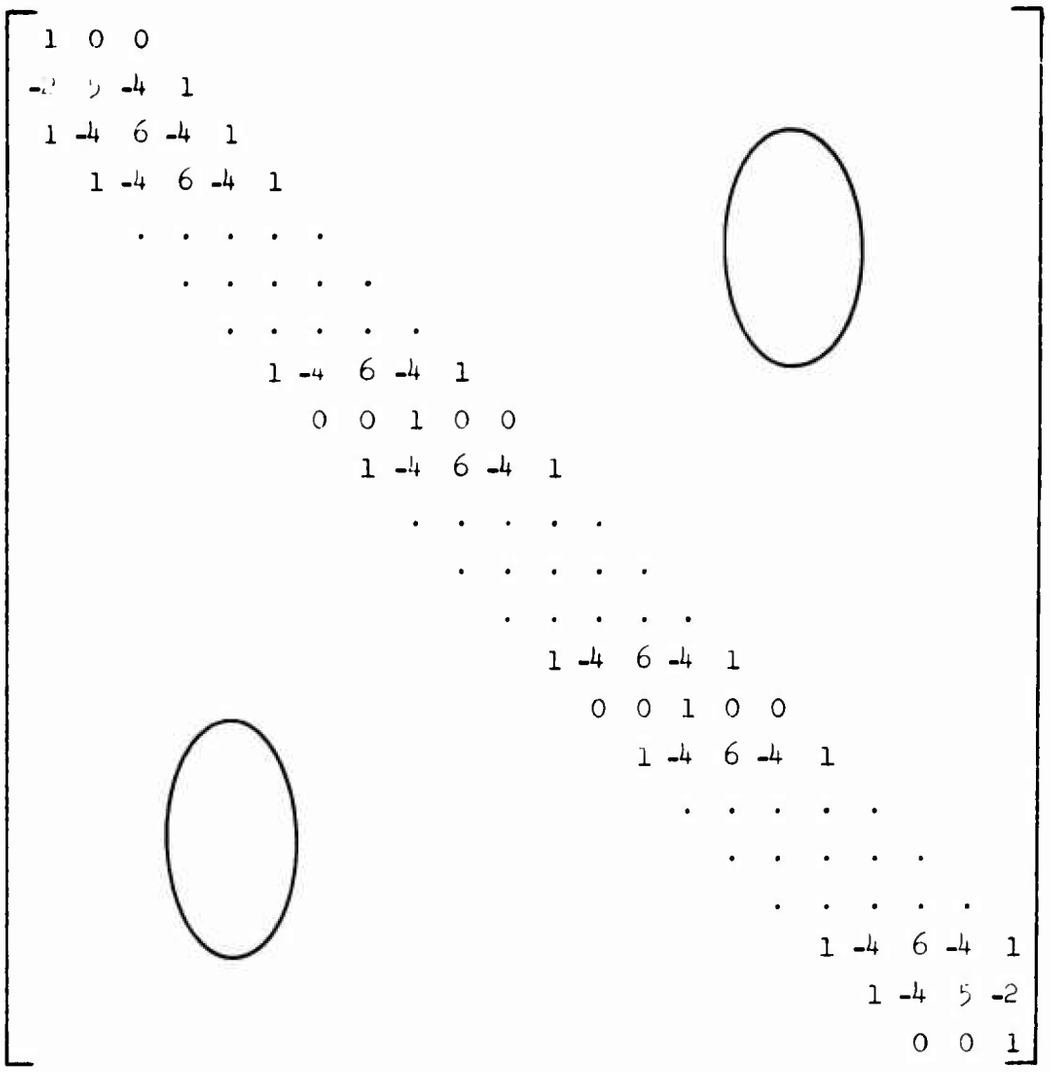
$$y_i = Y_j \quad \text{if} \quad i = L_j/h \ , \ i = 1,\ldots,m \ .$$

The values of y_0 and y_{m+1} are not explicitly computed, but are introduced into the formulation to accommodate the conditions on the second differences at the end nodes and the first and last of equations (4.1). Notice the similarity between this formulation and that of the continuous natural cubic spline.

A useful representation of the above linear system is given by

$$\underset{\sim}{A}\underset{\sim}{y} = \underset{\sim}{b} \ ,$$

where $\underset{\sim}{y} = (y_1 , \ldots , y_m)^T$, the matrix A has the structure



and $\underset{\sim}{b} = (Y_1 , 0 , \ldots , 0 , Y_2 , 0 , \ldots , 0 , Y_3 , \ldots 0 , Y_n)^T$. The rows of A with ones on the main diagonal correspond to the nonzero elements in the vector $\underset{\sim}{b}$. The second row of A combines the two equations

$$\delta^4 y_2 = 0 ,$$

and

$$\delta^2 y_1 = 0 .$$

The (m-1)th row of A is formed similarly.

The following analysis provides a way of solving $A\underset{\sim}{y} = \underset{\sim}{b}$ by solving a symmetric band system of smaller order than A. A bound on the condition number of this symmetric matrix is determined and found to be independent of the order of the matrix (total number of mesh points and knots) but highly dependent upon the number of mesh points between knots.

A symmetric matrix closely related to the matrix A can be determined as follows. Let the matrix B have the same structure as A except that the columns of B having ones on the main diagonal have all of the other components set to zero. Thus, the vector $\underset{\sim}{y}$ can be computed by solving the system

$$B\underset{\sim}{y} = \underset{\sim}{b}'$$

where the vector b' is an appropriate modification of the vector $\underset{\sim}{b}$. Now since B is a symmetric matrix, its condition number is the maximum ratio of the magnitudes of its eigenvalues. By expanding the determinant of $B - \lambda I$ along any row or column of B having a one on the main diagonal, it is obvious that $\lambda = 1$ is a multiple eigenvalue of B. These eigenvalues may be eliminated from B by dropping the rows and columns with 1 on the main diagonal, leaving the matrix

$$B^* = \begin{bmatrix}
5 & -4 & 1 & & & & & & & & & & \\
-4 & 6 & -4 & 1 & & & & & & & & & \\
1 & -4 & 6 & -4 & 1 & & & & & & & & \\
 & 1 & -4 & 6 & -4 & 1 & & & & & & & \\
 & & \ddots & \ddots & \ddots & \ddots & \ddots & & & & & & \\
 & & & \ddots & \ddots & \ddots & \ddots & \ddots & & & & & \\
 & & & & \ddots & \ddots & \ddots & \ddots & \ddots & & & & \\
 & & & & & 1 & -4 & 6 & -4 & 1 & & & \\
 & & & & & & 1 & -4 & 6 & -4 & & & \\
 & & & & & & & 1 & -4 & 6 & 1 & & \\
 & & & & & & & & 1 & 6 & -4 & 1 & \\
 & & & & & & & & & -4 & 6 & -4 & 1 \\
 & & & & & & & & & 1 & -4 & 6 & -4 \\
 & & & & & & & & & & \ddots & \ddots & \ddots \\
 & & & & & & & & & & & \ddots & \ddots \\
 & & & & & & & & & & & & 1 \\
 & & & & & & & & & & & 1 & -4 \\
 & & & & & & & & & & & & 1 & -4 & 5
\end{bmatrix}$$

where each eigenvalue of B^* is an eigenvalue of B. In practice, the unknown components of the vector $\underset{\sim}{y}$ are computed by solving a linear system having B^* as the coefficient matrix. Now

$$B^* = C + D \text{ , where}$$

$$C = \begin{bmatrix}
C_1 & & & & \\
 & C_2 & & & \\
 & & C_3 & & \\
 & & & \ddots & \\
 & & & & C_m
\end{bmatrix} ,$$

$$D = \begin{bmatrix} 0 & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & \ddots & & & & & & & \\ & & & 0 & & & & & & \\ & & & & 1 & 1 & & & & \\ & & & & 1 & 1 & & & & \\ & & & & & & 0 & & & \\ & & & & & & & \ddots & & \\ & & & & & & & & \ddots & \\ & & & & & & & & & 0 \end{bmatrix},$$

and each C_i is a square matrix of order $H_i/h-1$ of the form

$$C_i = \begin{bmatrix} 5 & -4 & 1 & & & & & & & \\ -4 & 6 & -4 & 1 & & & & & & \\ 1 & -4 & 6 & -4 & 1 & & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & & & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & & & 1 & -4 & 6 & -4 & 1 \\ & & & & & & & 1 & -4 & 6 & -4 \\ & & & & & & & & 1 & -4 & 5 \end{bmatrix}.$$

The matrix D has eigenvalues 0 and 2. If the eigenvalues $\lambda_i(B^*)$ and $\lambda_i(C)$, $i = 1, \ldots, m - n$, are arranged in a non-

increasing order, then by a corollary of the Courant-Fisher theorem (see Wilkinson (1965), p. 101),

$$\lambda_i(B^*) \geq \lambda_i(C) , \quad i = 1 , \ldots , m - n .$$

The eigenvalues of C are simply those of the C_i , $i = 1 , \ldots , m$. Now the C_i can be written as D_i^2 , where

$$D_i = \begin{bmatrix} -2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & 1 & -2 & 1 \\ & & & & & 1 & -2 \end{bmatrix} .$$

The characteristic polynomials of the upper-left hand minors of D_i satisfy the same three-term recurrence relation as the Chebyshev polynomials, namely

$$T_{-1}(\lambda) = 0 ,$$

$$T_0(\lambda) = 1 ,$$

$$T_{k+1}(\lambda) = (-2-\lambda)T_k(\lambda) - T_{k-1}(\lambda) , \quad k = 0,1,\ldots .$$

It follows that

$$\lambda_j(D_i) = -2(1-\cos \frac{j\pi}{\nu+1}) , \quad j = 1 , \ldots , \nu ,$$

where

$\nu = H_i/h-1$. Thus,

$$\min_j \lambda_j(C_i) = 4(1 - \cos \frac{\pi}{\nu+1})^2 ,$$

and therefore,

$$\min_j |\lambda_j(B)| \geq \min_i \; 4(1 - \cos \frac{h\pi}{H_i})^2 \; .$$

By Gerschgorin's theorem,

$$\max_j |\lambda_j(B)| \leq 16 \; .$$

This proves

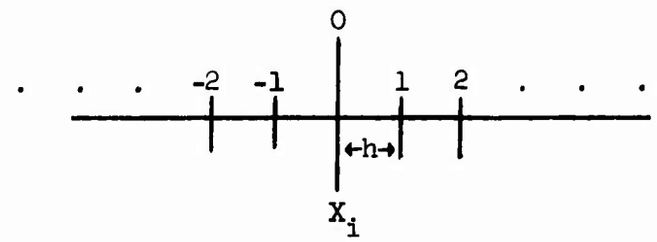
Theorem 4.1:
$$\text{cond } (B) \leq \frac{4}{\min\limits_i (1 - \cos \frac{h\pi}{H_i})^2} <$$

$$\frac{16}{\min\limits_i (\frac{h\pi}{H_i})^4} < 2 \max_i \left(\frac{H_i}{h}\right)^4 \; .$$

Thus, for many problems, the linear system resulting from the finite difference formulation is reasonably well-conditioned. More importantly, the bound on the condition number of B is independent of n, the number of data points.

The purpose of the following discussion is to analyze the difference between $\underset{\sim}{y}$ and the restriction of the continuous natural cubic spline function interpolating the points (X_i, Y_i), $i = 1, \ldots, n$, to the abscissae x_i, $i = 1, \ldots, m$. We call this difference the discretization error of $\underset{\sim}{y}$ and will now bound a certain norm of it.

Consider the neighborhood of X_i :



For convenience, the mesh nodes are now numbered ... , -2 , -1 , 0 , 1 , 2 , ... , such that $x_0 = X_i$.

Since $\delta^4 y_i = 0$ for $i = \ldots , -2 , -1$, it follows that the points y_i , for $i = \ldots , -2 , -1 , 0 , 1$ lie on a cubic polynomial. Similarly, the points y_i , for $i = -1 , 0 , 1 , 2 , \ldots$, lie on a (generally different) cubic polynomial. These two cubics coincide at the points -1 , 0 , and 1 . Let

$$\mathcal{Y}_i = a_i + b_i \xi + c_i \xi^2 + d_i \xi^3 \quad , \quad \text{where} \quad \xi = x - X_i \ ,$$

denote such a cubic for $x \in [X_i - h \ , \ X_{i+1} + h]$. The coefficients a_i , b_i , c_i and d_i , $i = 1 , \ldots , n - 1$, can actually be computed by observing the following: $\mathcal{Y}_i''$ is linear. Let $\mathcal{Y}(x) = \mathcal{Y}_i(x)$ if $x \in [X_i , X_{i+1}]$, $i = 1 , \ldots , n - 1$. Thus,

$$\mathcal{Y}_i''(x) = Y_i'' + \frac{x - X_i}{H_i} (Y_{i+1}'' - Y_i'') \ , \tag{4.2}$$

where H_i was previously defined as $H_i = X_{i+1} - X_i$, i=1 , ... , n-1 . Note that $\mathcal{Y}''(x)$ is continuous at each knot X_i since

$$\frac{\delta^2 y_0}{h^2} = \mathcal{Y}''(x_0)$$

for each cubic ($\mathcal{Y}_{i-1}$ and $\mathcal{Y}_i$) in the interval $[X_i - h \ , \ X_i + h]$. Integrating (4.2) gives

$$\mathcal{Y}_i'(x) = {}^{+}Y_i' + Y_i''(x - X_i) + (Y_{i+1}'' - Y_i'') \frac{(x - X_i)^2}{2H_i} \ , \tag{4.3}$$

where ${}^{+}Y_i'$ is a constant of integration, and

$$\mathcal{Y}_i(x) = Y_i + {}^{+}Y_i'(x - X_i) + Y_i'' \frac{(x - X_i)^2}{2} + \frac{Y_{i+1}'' - Y_i}{6H_i} (x - X_i)^3 \ . \tag{4.4}$$

Here ${}^{+}Y_i'$ denotes $\gamma_i'(X_i^{+})$. Since γ' is not necessarily continuous at the knots, ${}^{-}Y_i'$ will be used to denote $\gamma'(X_i^{-})$. Evaluating (4.4) at X_{i+1} gives

$$ {}^{+}Y'_i = \frac{Y_{i+1} - Y_i}{H_i} - Y''_i \frac{H_i}{3} - Y''_{i+1} \frac{H_i}{6} . \tag{4.5} $$

Replacing i by i - 1 in (4.3) and evaluating it at X_i gives

$$ {}^{-}Y_i' = {}^{+}Y'_{i-1} + (Y'_{i-1} + Y''_i) \frac{H_{i-1}}{2} . \tag{4.6} $$

Now, for γ_i ,

$$ \frac{\delta y_0}{2h} = \frac{y_1 - y_{-1}}{2h} = b_i + d_i h^2 $$

$$ = {}^{+}Y_i' + d_i h^2 , $$

and for γ_{i-1} ,

$$ \frac{\delta y_0}{2h} = \frac{1}{2h} \Big[a_{i-1} + b_{i-1} (H_{i-1} + h) + c_{i-1} (H_{i-1} + h)^2 + d_{i-1} (H_{i-1} + h)^3 $$

$$ - a_{i-1} - b_{i-1} (H_{i-1} - h) - c_i (H_{i-1} - h)^2 - d_i (H_{i-1} - h)^3 \Big] $$

$$ = b_{i-1} + 2c_{i-1} H_{i-1} + 3d_{i-1} H_{i-1}^2 + d_{i-1} h^2 $$

$$ = {}^{-}Y_i' + d_{i-1} h^2 . $$

Thus,

$$^{-}Y_i' = {}^{+}Y_i' + h^2(d_i - d_{i-1}) \ . \tag{4.7}$$

Combining (4.7) with (4.5) and (4.6) gives

$$Y''_{i-1} H_{i-1} + 2Y''_i (H_{i-1} + H_i) + Y''_{i+1} H_i + 6h^2(d_i - d_{i-1}) \tag{4.8}$$

$$= 6(\Delta_{i+1} - \Delta_i) \ , \quad i = 2 \ , \ \ldots \ , \ n - 1 \ ,$$

where

$$\Delta_i = (Y_i - Y_{i-1})/H_{i-1} \ , \quad i = 1 \ , \ \ldots \ , \ n \ .$$

The conditions $\delta^2 y_i = 0$ at the ends require that $Y''_1 = Y''_n = 0$. Equations (4.4) and (4.5) give

$$a_i = Y_i \ , \tag{4.9}$$

$$b_i = \frac{Y_{i+1} - Y_i}{H_i} - Y''_i \frac{H_i}{3} - Y''_{i+1} \frac{H_i}{6} \ , \tag{4.10}$$

$$c_i = Y''_i/2 \ , \text{ and} \tag{4.11}$$

$$d_i = \frac{Y''_{i+1} - Y''_i}{6H_i} \ , \quad i = 1 \ , \ \ldots \ , \ n - 1 \ . \tag{4.12}$$

Substituting (4.12) into (4.8) gives a linear system of equations in the unknowns Y''_i, $i = 2 \ , \ \ldots \ , \ n - 1$, the solution of which can be used to compute the coefficients a_i, b_i, c_i, and d_i, $i = 1 \ , \ \ldots \ , \ n - 1$, using (4.9) - (4.12).

A practical method for computing the coefficients of the continuous cubic spline interpolating (X_i , Y_i) , $i = 0 , \ldots , n$, can be formulated by following the above steps and replacing (4.6) with the continuity conditions

$$^{-}Y_i' = {}^{+}Y_i' , \quad i = 1 , \ldots , n - 1 .$$

This leads to a system of equations for Y_i'' which is identical to (4.8) without the $6h^2(d_i - d_{i-1})$ term. Thus we have the same system of equations for Y_i'' in the discrete case as in the continuous case, except for an $O(h^2)$ perturbation of the coefficient matrix.

Let $\underset{\sim}{s}''$ denote the vector of second derivatives (Y_i'') for the continuous case and let $\underset{\sim}{y}''$ denote the vector of second derivatives for the discrete case. Also, let

$$\underset{\sim}{\alpha} = \underset{\sim}{y}'' - \underset{\sim}{s}'' ,$$

$$A = \begin{bmatrix} 2(H_1+H_2) & H_2 & & & & & \\ H_2 & 2(H_2+H_3) & H_3 & & & & \\ & H_3 & 2(H_3+H_4) & H_4 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & H_{n-2} \\ & & & & & H_{n-2} & 2(H_{n-2}+H_{n-1}) \end{bmatrix} ,$$

and

$$\delta A = h^2 \begin{pmatrix} -\left(\frac{1}{H_1} + \frac{1}{H_2}\right) & \frac{1}{H_2} & & & & \\ \frac{1}{H_2} & -\left(\frac{1}{H_2} + \frac{1}{H_3}\right) & \frac{1}{H_3} & & & \\ & \frac{1}{H_3} & -\left(\frac{1}{H_3} + \frac{1}{H_4}\right) & \frac{1}{H_4} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \frac{1}{H_{n-2}} \\ & & & & \frac{1}{H_{n-2}} & -\left(\frac{1}{H_{n-2}} + \frac{1}{H_{n-1}}\right) \end{pmatrix} .$$

Let $\beta_i = 6(\Delta_{i+1} - \Delta_i)$. Then from (4.8),

$$(A + \delta A)\ (\underset{\sim}{s}'' + \underset{\sim}{\alpha}) = \underset{\sim}{\beta} .$$

So,

$$\underset{\sim}{\alpha} = [(A + \delta A)^{-1} - A^{-1}]\ \underset{\sim}{\beta} .$$

Setting $B = A + \delta A$ in the identity

$$B^{-1} - A^{-1} = A^{-1}(A - B)\ B^{-1} ,$$

and substituting gives

$$\underset{\sim}{\alpha} = -A^{-1}\delta A(A + \delta A)^{-1}\ \underset{\sim}{\beta} = -A^{-1}\delta A(\underset{\sim}{s}'' + \underset{\sim}{\alpha}) .$$

Taking spectral norms gives

$$\|\underset{\sim}{\alpha}\| \leq \|A^{-1}\| \, \|\delta A\| \, \|\underset{\sim}{s}'' + \underset{\sim}{\alpha}\| \ ,$$

or

$$\frac{\|\underset{\sim}{\alpha}\|}{\|\underset{\sim}{y}''\|} \leq \text{cond}(A) \, \frac{\|\delta A\|}{\|A\|} \ .$$

Now, by Gershgorin's theorem,

$$\min_i \, (H_i + H_{i+1}) \leq \lambda(A) \leq 3 \, \max_i \, (H_i + H_{i+1})$$

and

$$\lambda(\delta A) \leq \max_i \ 2h^2 \left(\frac{1}{H_i} + \frac{1}{H_{i+1}} \right) \leq \max_i \ \frac{4h^2}{H_i} \ .$$

Thus, $\text{cond}(A) \leq 3 \, \max_i \, (H_i + H_{i+1}) / \min_i \, (H_i + H_{i+1})$, and

$$\frac{\|\underset{\sim}{\alpha}\|}{\|\underset{\sim}{y}''\|} \leq \frac{4h^2}{(\min_i H_i) \, \min_i (H_i + H_{i+1})} \ .$$

Since the uniform norm of a vector is at most equal to the spectral norm,

$$|\alpha_j| \leq \frac{4h^2 \|\underset{\sim}{y}''\|}{H_{\min} \, \min_i (H_i + H_{i+1})} \ , \quad j = 1 \ , \ \ldots \ , \ n - 1 \ .$$

The coefficients a_i , b_i , c_i , and d_i are linear functions of the Y''_i , hence the difference between the continuous and discrete splines is $O(h^2)$. More precisely, denoting the continuous spline by $s(x)$, equations (4.9) - (4.12) yield

Theorem 4.2:

$$\max_x \, |s(x) - \mathcal{Y}(x)| \leq \frac{16}{3} \, \frac{(H_{\max})^2 \|\underset{\sim}{y}''\| \, h^2}{H_{\min} \, \min_i (H_i + H_{i+1})} \ , \quad x \in [X_0, X_n] \ .$$

Theorem 4.2 tells us that the discrete natural cubic interpolating spline coincides with the continuous natural cubic spline in the limit as the mesh size h is decreased to zero. As the mesh size is decreased, the error in approximating the continuous spline decreases at least as fast as the square of the mesh size. However, Theorem 4.1 tells us that the condition number of the coefficient matrix of the linear system may increase as fast as h^4. On a given machine, it should thus be possible to calculate a reasonable discrete spline approximation; however, problems with fine meshs could be difficult, if not impossible, to solve.

The work required to compute discrete cubic splines is considerably more than that required to compute continuous cubic splines. The primary reason for studying them here is to gain insight into the algorithm for computing discrete approximations to nonlinear splines, discussed in the following section. In view of the difficulty in analyzing linear discrete splines, it comes as no surprise that the nonlinear case has not been analyzed. The proofs of this section rely upon special properties of both the discrete and continuous cubic splines. Hence, few, if any, of the arguments would carry over to the nonlinear case.

Discrete cubic splines and generalized discrete splines have been characterized by Mangasarian and Schumaker (1971). They prove existence and uniqueness. Some weak convergence results for discrete splines have been published by Daniel (1971). Theorem 4.2 appears to be the only rate-of-convergence result for discrete splines.

5. A Finite-Difference Method for Computing Open Nonlinear Spline Functions

In this section, an iterative method is presented for computing a discrete approximation to the nonlinear spline interpolating the data points (X_i,Y_i) , i=1,...,n .

As in the previous section, we assume that $X_1 < X_2 < \ldots < X_n$, and the interval $[X_1 - h , X_n + h]$ is partitioned into a uniform mesh $x_0 , x_1 , \ldots , x_{m+1}$, where

$$x_i = X_1 + (i-1)h , \quad i=0,1,\ldots,m+1 ,$$

and

$$m = (X_n - X_1)/h + 1 ,$$

for some mesh size parameter h . For simplicity, assume that the X_i (i=1,...,n) are equally spaced with k mesh intervals between consecutive data points, i.e.,

$$k = (X_{i+1} - X_i)/h , \quad i=1,\ldots,n-1 .$$

We compute a discrete approximation $\underset{\sim}{y} = [y_1,\ldots,y_m]^T$ to the function y which minimizes the functional

$$E(y) = \int_{X_1}^{X_n} \frac{(y'')^2}{[1 + (y')^2]^{5/2}} \, dx . \tag{5.1}$$

The functional (5.1) is approximated by substituting a summation for the integral,

$$h^{-2}\delta^2 y_i \approx y''(x_i) ,$$

and

$$(y_{i+1} - y_{i-1})/2h \approx y'(x_i) .$$

This gives the function

$$E_h(\underset{\sim}{y}) = \sum_{i=1}^{m} \frac{(y_{i+1} - 2y_i + y_{i-1})^2/h^4}{[1 + (y_{i+1} - y_{i-1})^2/4h^2]^{5/2}} h \ , \qquad (5.2)$$

to be minimized subject to the interpolation constraints

$$y_i = Y_j \ , \text{ if } \ x_i = X_j \ , \ i=1,\ldots,m \ .$$

Again, the fictitious ordinates y_0 and y_{m+1} are introduced to accommodate the first and last terms of (5.2).

A necessary condition for $E_h(\underset{\sim}{y})$ to achieve a minimum is

$$\frac{\partial}{\partial y_i} E_h(\underset{\sim}{y}) = 0 \ , \quad \text{for all} \ \ i \not\equiv 1 \ (\text{mod} \ \ k) \ .$$

This yields the nonlinear system of equations

$$\delta^2 y_1 = 0 \ ,$$

$$\delta^2 y_m = 0 \ ,$$

$$\begin{aligned} &\varphi_{i-1}(y_i - 2y_{i-1} + y_{i-2}) - 2\varphi_i(y_{i+1} - 2y_i + y_{i-1}) \qquad (5.3) \\ &+ \varphi_{i+1}(y_{i+2} - 2y_{i+1} + y_i) \\ &- \psi_{i-1}(y_i - y_{i-2}) + \psi_{i+1}(y_{i+2} - y_i) = 0 \ , \end{aligned}$$

where

$$\varphi_i = [1 + (y_{i+1} - y_{i-1})^2/4h^2]^{-5/2} \ ,$$

$$\psi_i = \frac{5}{8h^2}(y_{i+1} - 2y_i + y_{i-1})^2[1 + (y_{i+1} - y_{i-1})^2/4h^2]^{-7/2} \ ,$$

$$(\text{for} \ \ i = 2,\ldots,m-1 \ , \ \text{except} \ \ i = 1 \ (\text{mod} \ \ k)) \ .$$

The System (5.3) can be solved in many ways. A simple, though probably not the fastest way is a Picard-type iteration described as

follows. We will compute a sequence of iterates $\underset{\sim}{y}^{(1)}$, $\underset{\sim}{y}^{(2)}$,... . Let

$$\xi_i^{(j)} = \xi_i(\underset{\sim}{y}^{(j)}) \text{ , and}$$

$$\mathfrak{L}_i^{(j)} = \mathfrak{L}_i(\underset{\sim}{y}^{(j)}) \text{ , } \quad j = 1,2,\ldots .$$

For the first iteration set

$$\xi_i^{(0)} = 1 \text{ , } i = 2,\ldots,m-1 \text{ ,}$$

$$\mathfrak{L}_i^{(0)} = 0 \text{ , } i = 1,\ldots,m \text{ .}$$

At the j-th iteration, the five-band linear system

$$\delta^2 y_1 = 0$$

$$\delta^2 y_m = 0$$

$$\xi_{i-1}^{(j-1)}(y_i^{(j)} - 2y_{i-1}^{(j)} + y_{i-2}^{(j)}) - 2\xi_i^{(j-1)}(y_{i+1}^{(j)} - 2y_i^{(j)} + y_{i-1}^{(j)})$$

$$+ \xi_{i+1}^{(j-1)}(y_{i+2}^{(j)} - 2y_{i+1}^{(j)} + y_i^{(j)})$$

$$- \mathfrak{L}_i^{(j-1)}(y_i^{(j)} - y_{i-2}^{(j)}) + \mathfrak{L}_{i+1}^{(j-1)}(y_{i+2}^{(j)} - y_i^{(j)}) = 0 \text{ ,}$$

$$(i = 2,\ldots,m-1 \text{ , except } i=1 \pmod{k}) \text{ ,}$$

is solved for $\underset{\sim}{y}^{(j)}$. Notice that $\underset{\sim}{y}^{(1)}$ is the discrete natural spline discussed in Section 4.

This algorithm is implemented in the Fortran subroutine SPLINE found at the end of this section.

A Cholesky method is used for the solution of (5.4). This choice is based on computational experience. The coefficient matrix of (5.4) is usually positive definite. Cases in which the coefficient matrix is indefinite appear to correspond to those in which either no single-valued solution exists or the outer iteration becomes unstable due to large slopes in the solution.

Various experiments have been carried out using both SPLINE and MEC given in Section 3.3. The first of these used the Woodford data given in Section 3.3. For this case, $n = 7$. Figure 5.1 shows times required to compute splines of similar accuracy on an IBM 360/67 for various values of k. These runs used EPS values of 10^{-3}. Figure 5.1 indicates that both SPLINE and MEC require time proportional to k, though spline appears to be about an order of magnitude faster for this problem. For values of k greater than 40, MEC failed - though this could be partly due to the fact that MEC is programmed in single precision. SPLINE worked for values of k as large as 140. Values of (5.2) are given in Table 5.1 for solutions given by MEC, SPLINE and the natural continuous cubic spline, for Woodford's data and various values of k. The values of $E_h(\underset{\sim}{y})$ for the discrete cubic spline agreed with those for the continuous cubic spline to three significant digits. The last column of Table 5.1 indicates that the results of SPLINE and MEC converge uniformly to one another as k increases. The nonlinear spline interpolating Woodford's data is shown in Figure 5.2 as plotted by a Versatec electrostatic plotter.

An experiment to determine the dependency of computation time upon n, the number of data points, was constructed as follows. The number of mesh intervals between adjacent data points was kept constant at $k = 10$. The mesh size was $h = .1$ giving $x_{i+1} - x_i = kh = 1$, $i = 1,\ldots,n-1$. The ordinates were chosen as

$$Y_i = (i \bmod 2)/5 \ , \ i = 1,\ldots,n \ ,$$

to insure a solution having relatively small slopes and curvatures. The parameters EPS were set to 10^{-3}. Measured CPU times are shown in Figure 5.3 for both SPLINE and MEC. The two values of n shown for

MEC in Figure 5.3 are $n = 5$ and $n = 10$. From these two times, MEC does not appear to require time proportional to n. For values of n greater than 10, MEC failed to converge. SPLINE, on the other hand, required relatively small amounts of time which are proportional to n. SPLINE performed well for values of n as large as 100. The only restrictions on n for SPLINE appear to be available core storage and computation time.

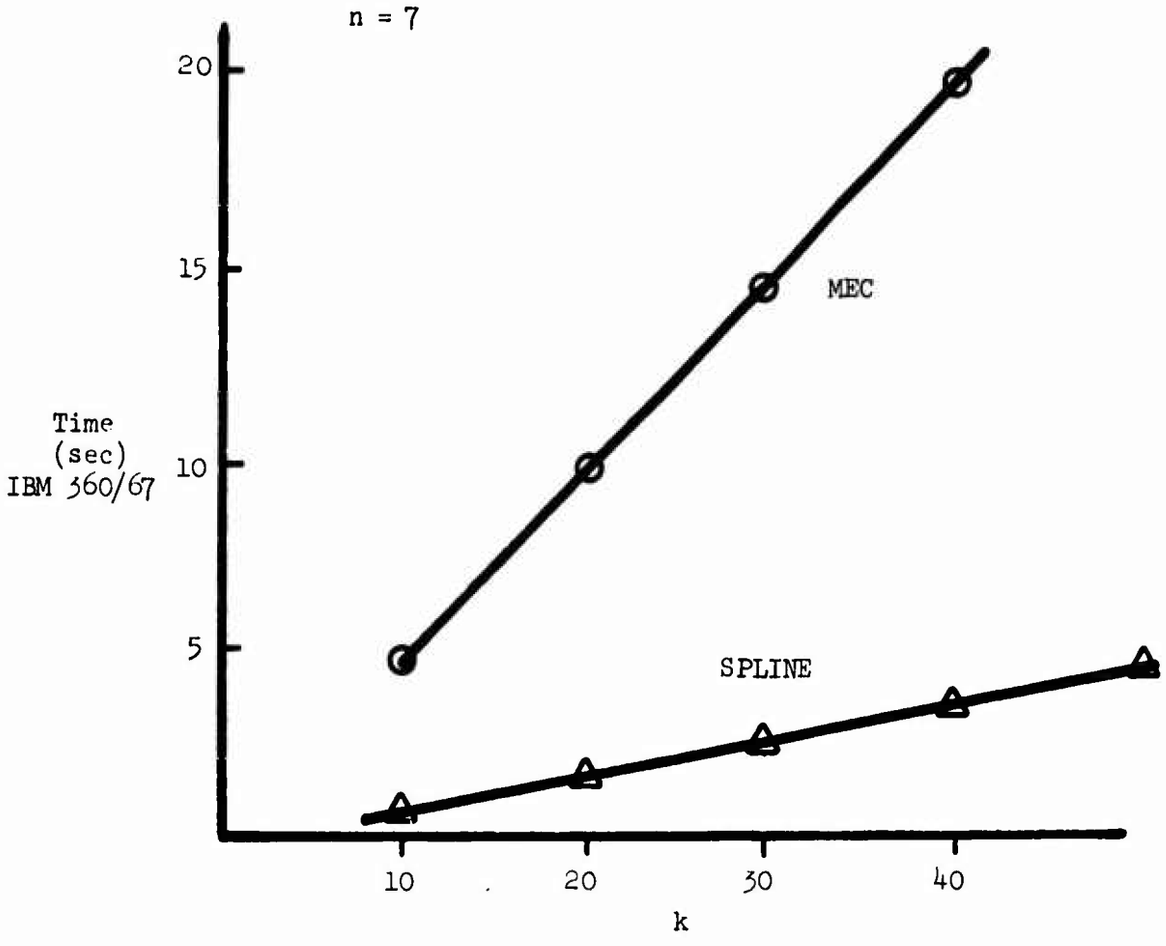


Figure 5.1. Timing comparisons of MEC and SPLINE for Woodford's data

k	Discrete Energy			$\|\text{MEC} - \text{SPLINE}\|_\infty$
	MEC	SPLINE	Cubic Spline	
10	2.53	2.52	2.69	.020
20	2.53	2.53	2.69	.011
30	2.53	2.53	2.70	.0070
40	2.53	2.53	2.70	.0051

Table 5.1. Values of $E_h(y)$ for Woodford's data for MEC, SPLINE, and the cubic spline, and a uniform-norm comparison of MEC with SPLINE.

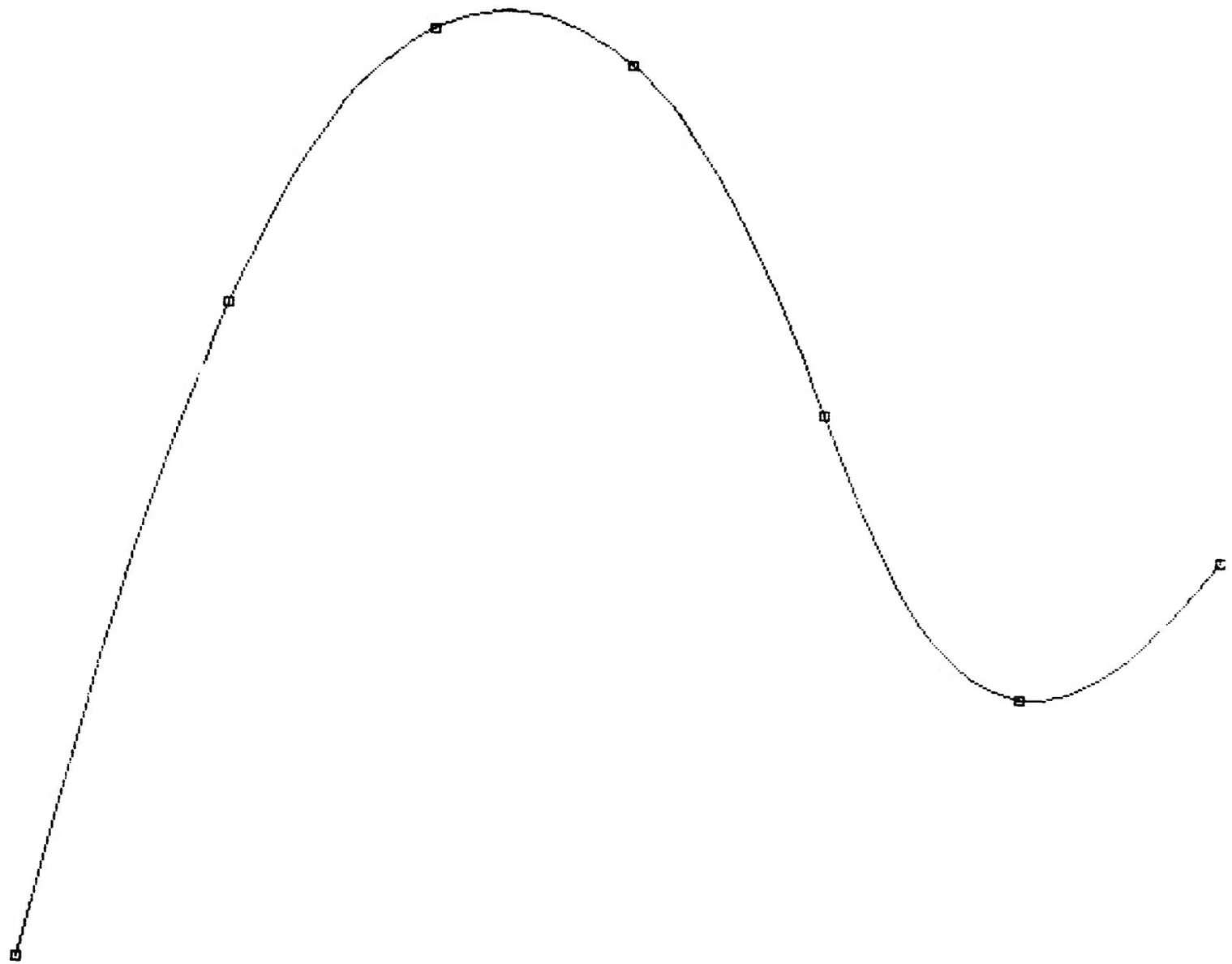


Figure 5.2. Nonlinear spline interpolating Woodford's data

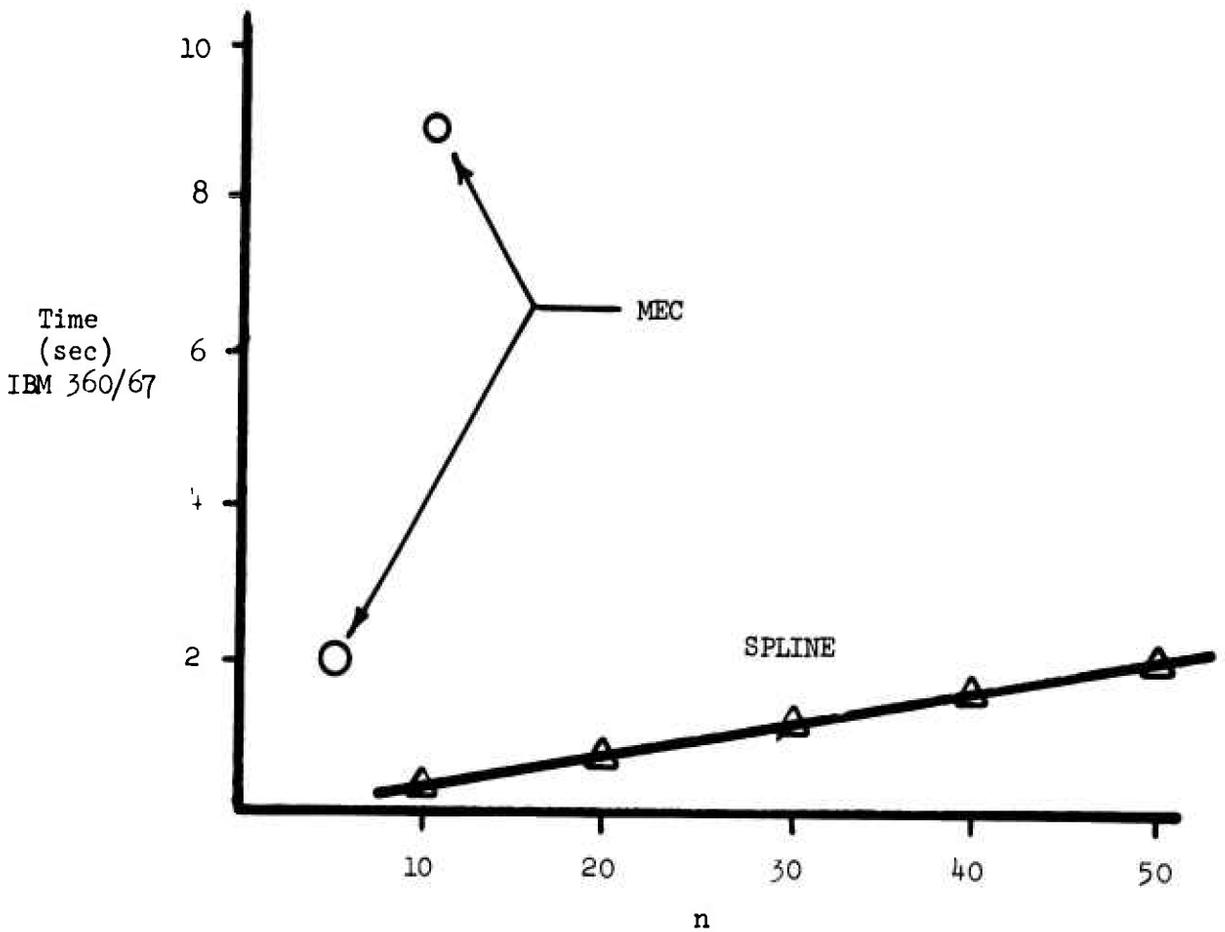


Figure 5.3. Dependence of SPLINE and MEC upon the number of data points n .

```
      SUBROUTINE  SPLINE (N, ORD, K, H, Y, EPS, *)
      INTEGER  N, K
      DOUBLE PRECISION  ORD(N), H, Y(1), EPS
C
C     THIS SUBROUTINE COMPUTES THE NONLINEAR INTERPOLATING SPLINE WHICH
C  PASSES THROUGH THE N DATA POINTS WITH ORDINATES ORD(I),
C  (I=1,...,N), AND EQUIDISTANT ABSCISSAE X(I), (I=1,...,N), SATISFYING
C  X(1).LT.X(2).LT. ... .LT.X(N), AND X(I+1)-X(I) = K*H, (I=1,...,N-1).
C  THE DISCRETE APPROXIMATION Y(I), (I=1,...,(N-1)*K+1) IS RETURNED
C  WITH ORDINATES CORRESPONDING TO THE MESH ABSCISSA X(1)+(I-1)*H, WHERE
C  H IS THE SPECIFIED MESH SIZE PARAMETER. K IS THE NUMBER OF MESH
C  INTERVALS BETWEEN SUCCESSIVE DATA POINTS. THE INPUT PARAMETER EPS
C  IS USED TO DETERMINE WHEN TO STOP THE ITERATIONS. ROUGHLY, IF P
C  SIGNIFICANT DIGITS ARE DESIRED IN A PROBLEM WHOSE SOLUTION IS O(1),
C  THEN EPS SHOULD BE ABOUT 1.D-P.
C     THE ERROR RETURN IS TAKEN IN THE EVENT THAT A RESULTING LINEAR
C  SYSTEM IS TOO ILL-CONDITIONED TO SOLVE.
C     K MUST BE AT LEAST 5.
C
C
      DOUBLE PRECISION  KZ(1000), C(1000,3), B(1000), NORM, H2, DIF
      DOUBLE PRECISION  F8H2, LZ(1000), Y1
      DOUBLE PRECISION  DABS
      INTEGER  M, M1, MM, K2, N1, I, J, L, J2
C
      H2 = H+H
      F8H2 = 5./(8.*H**2)
      M = K*(N-1)+1
      M1 = M-1
      K2 = k-2
      N1 = N-1
C
C  INITIALIZE  Y
C
      DO 5 I=1,M
         Y(I) = 0.
    5 CONTINUE
C
C  SET DATA ORDINATES IN Y
C
      J = 1
      DO 10 I=1,M,K
         Y(I) = ORD(J)
         J = J+1
   10 CONTINUE
C
C  INITIALIZE  KZ  TO ONES FOR THE FIRST ITERATION
C
```

```
      DO 20 I=2,M1
         KZ(I) = 1.
         LZ(I) = 0.
   20 CONTINUE
C
C  KZ(1),KZ(M),LZ(1), AND LZ(M)  ARE SET TO 0. TO SIMPLIFY BOOKKEEPING
C
      KZ(1) = 0.
      KZ(M) = 0.
      LZ(1) = 0.
      LZ(M) = 0.
C
C  GO DO FIRST ITERATION
C
      GO TO 50
C
C  BEGINNING OF MAIN LOOP,
C  RECOMPUTE NEW  KZ  AND  LZ  VECTORS
C
   30 DO 40 I=2,M1
         Y1 = 1. + ((Y(I+1)-Y(I-1))/H2)**2
         KZ(I) = Y1**(-2.5)
         LZ(I) = F8H2*(Y(I+1)-2.*Y(I)+Y(I-1))**2*KZ(I)/Y1
   40 CONTINUE
C
C  SET UP COEFFICIENT MATRIX AND RIGHT-HAND SIDE
C
   50 L = 0
      DO 80 I=1,N1
         L = L + 1
         MM = (I-1)*K
         C(L,1) = 0.
         C(L,2) = KZ(MM+1) + LZ(MM+1)
         C(L,3) = KZ(MM+1) + 4.*KZ(MM+2) + KZ(MM+3) - LZ(MM+1)-LZ(MM+3)
         B(L) = 2.*(KZ(MM+2)+KZ(MM+1))*Y(MM+1)
         L = L+1
         C(L,1) = 0.
         C(L,2) = -2.*(KZ(MM+2) + KZ(MM+3))
         C(L,3) = KZ(MM+2) + 4.*KZ(MM+3) + KZ(MM+4) - LZ(MM+2)-LZ(MM+4)
         B(L) = -(KZ(MM+2)+LZ(MM+2))*Y(MM+1)
         L = L+1
         IF (K2.LT.4) GO TO 70
         DO 60 J = 4,K2
            MM = (I-1)*K + J
            C(L,1) = KZ(MM-1) + LZ(MM-1)
            C(L,2) = -2.*(KZ(MM-1) + KZ(MM))
            C(L,3) = KZ(MM-1) + 4.*KZ(MM) + KZ(MM+1) - LZ(MM-1)-LZ(MM+1)
            B(L) = 0.
            L = L+1
   60    CONTINUE
```

```
   70    MM = I*K
         C(L,1) = KZ(MM-2) + LZ(MM-2)
         C(L,2) = -2.*(KZ(MM-1) + KZ(MM-2))
         C(L,3) = KZ(MM-2) + 4.*KZ(MM-1) + KZ(MM) - LZ(MM-2)-LZ(MM)
         B(L) = -(KZ(MM)+LZ(MM)) * Y(MM+1)
         L = L+1
         C(L,1) = KZ(MM-1) + LZ(MM-1)
         C(L,2) = -2.*(KZ(MM-1) + KZ(MM))
         C(L,3) = KZ(MM-1) + 4.*KZ(MM) + KZ(MM+1) - LZ(MM-1)-LZ(MM+1)
         B(L) = 2.*(KZ(MM)+KZ(MM+1))*Y(MM+1)
   80 CONTINUE
C
C  SOLVE 5-BAND LINEAR SYSTEM
C
      CALL CCOMP(L, 3, 1000, C, C, &100)
      CALL COLVE(L, 3, 1000, C, B, B)
C
C  UPDATE SOLUTION VECTOR  Y  AND CHECK FOR CONVERGENCE
C
      L = 1
      NORM = 0.
      DO 90 I=1,N1
         DO 90 J=2,K
            J2 = (I-1)*K+J
            DIF = DABS(Y(J2) - B(L))
            Y(J2) = B(L)
            IF (DIF.GT.NORM) NORM = DIF
            L = L+1
   90 CONTINUE
      IF (NORM.GT.EPS) GO TO 30
      RETURN
  100 RETURN 1
      END
```

```
      SUBROUTINE  CCOMP(N, M, IDIM, C, L, *)
      INTEGER  N, M, IDIM
      DOUBLE PRECISION  C(IDIM, M), L(IDIM, M)
C
C  THIS SUBROUTINE FINDS THE CHOLESKY DECOMPOSITION OF THE (2M-1)-BAND
C  MATRIX  A  OF ORDER  N  WHERE THE MAIN AND M-1 SUB DIAGONALS OF A
C  ARE STORED IN  C  AS SHOWN (M=3 IS USED FOR THIS EXAMPLE)
C
C     X         X         A(1,1)
C     X         A(2,1)    A(2,2)
C     A(3,1)    A(3,2)    A(3,3)
C     A(4,2)    A(4,3)    A(4,4)
C     .         .         .
C     .         .         .
C     .         .         .
C
C  THE LOWER TRIANGULAR PART OF THE CHOLESKY DECOMPOSITION  A = LU,
C  WHERE U IS L TRANSPOSE, IS RETURNED IN  L.  THE  RETURN 1  IS TAKEN
C  IN THE EVENT THAT THE MATRIX  A  IS FOUND TO BE SINGULAR, OR
C  NON POSITIVE DEFINITE.  C  AND  L  MAY BE THE SAME ARRAYS,
C  IF DESIRED.
C      THIS SUBROUTINE IS A PARTIAL TRANSLATION OF THE ALGOL 60
C  PROCEDURE  CHOBANDDET  BY R.S. MARTIN AND J.H. WILKINSON, FOUND
C  IN THE HANDBOOK FOR AUTOMATIC COMPUTATION, VOLUME II (LINEAR
C  ALGEBRA), 1971, J.H. WILKINSON AND C. REINSCH (EDITORS).
C
C
      INTEGER  I, P, R, S, Q
      DOUBLE PRECISION  Y
      DOUBLE PRECISION  DSQRT
C
      DO 50 I=1,N
         P = MAX0(1,M-I+1)
         R = I-M+P
         DO 40 J=P,M
            S = J-1
            Q = M-J+P
            Y = C(I,J)
            IF (P.GT.S) GO TO 20
            DO 10 K=P,S
               Y = Y - L(I,K)*L(R,Q)
               Q = Q+1
   10       CONTINUE
   20       IF (J.NE.M) GO TO 30
            IF (Y.LE.0.) RETURN 1
            L(I,J) = 1./DSQRT(Y)
            GO TO 40
   30       L(I,J) = Y*L(R,M)
            R = R+1
   40    CONTINUE
   50 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE  COLVE (N, M, IDIM, L, B, X)
      INTEGER  N, M, IDIM
      DOUBLE PRECISION  L(IDIM, M), B(1), X(1)
C
C  THIS SUBROUTINE SOLVES THE (2M-1)-BAND LINEAR SYSTEM OF
C  EQUATIONS
C
C        AX = B
C
C  USING THE LOWER PART OF THE CHOLESKY FACTORIZATION  L  OF  A,
C  WHICH HAS BEEN COMPUTED BY CCOMP.
C  B  AND  X  MAY BE THE SAME ARRAYS, IF DESIRED.
C      THIS SUBROUTINE IS A PARTIAL TRANSLATION OF THE ALGOL 60
C  PROCEDURE  CHOBANDSOL  BY R.S. MARTIN AND J.H. WILKINSON, FOUND
C  IN THE HANDBOOK FOR AUTOMATIC COMPUTATION, VOLUME II (LINEAR
C  ALGEBRA), 1971, J.H. WILKINSON AND C. REINSCH (EDITORS).
C
C
      INTEGER  S, J, P, Q, NP1, K1, I1
      DOUBLE PRECISION  Y
C
C  SOLUTION  OF  LY = B
C
      DO 20 I=1,N
         P = MIN0(I-1,M-1)
         Q = I
         Y = B(I)
         IF (P.LT.1) GO TO 15
         DO 10 K1=1,P
            K = M-K1
            Q = Q-1
            Y = Y - L(I,K)*X(Q)
   10    CONTINUE
   15    X(I) = Y*L(I,M)
   20 CONTINUE
C
C  SOLUTION OF  UX = Y
C
      NP1 = N+1
      DO 40 I1=1,N
         I = NP1-I1
         P = MIN0(I1-1,M-1)
         Y = X(I)
         Q = I
         IF (P.LT.1) GO TO 35
         DO 30 K1=1,P
            K = M-K1
            Q = Q+1
            Y = Y - L(Q,K)*X(Q)
   30    CONTINUE
   35    X(I) = Y*L(I,M)
   40 CONTINUE
      RETURN
      END
```

References

Ahlberg, J. H., E. N. Nilson and J. L. Walsh (1967), The Theory of Splines and their Applications. New York: Academic Press.

Bernoulli, Daniel (1742), the 26th letter to Euler (October) in Fuss, Correspondance Mathêmatique et Physique. T. 2, St. Petersburg, 1843. (German and Latin)

Birkhoff, G. and C. T. de Boor (1965), "Piecewise Polynomial Interpolation and Approximation," Approximation of Functions, Proceedings of General Motors Symposium of 1964, H. L. Garabedian (editor), New York and Amsterdam: Elsevier Publishing Co., 164-190.

Birkhoff, G., H. Burchard and D. Thomas (1965), "Nonlinear Interpolation by Splines, Pseudosplines, and Elastica," Research Publication 468, General Motors Research Laboratories, Warren, Michigan.

Daniel, J. W. (1971), "Convergence of a Discretization for Constrained Spline Function Problems," SIAM J. Control, Vol. 19, No. 1, 83-96.

Euler, L. (1744), Additamentum "De curvis elasticis," in Methodus Inveniendi Lineas Curvas Maximi Minimive Proprietate Gaudentes. Ser. 1, V. 24, Lausanne. (Latin)

Fowler, A. H. and C. W. Wilson (1963), "Cubic Spline, A Curve Fitting Routine," Report Y-1400, Oak Ridge National Laboratories, Oak Ridge, Tennessee.

Fung, Y. C. (1966), Foundations of Solid Mechanics, Englewood Cliffs, N.J.: Prentice-Hall.

Glass, J. M. (1966), "Smooth-curve Interpolation: A Generalized Spline-Fit Procedure," BIT 6, 277-293.

Hosaka, Mamoru (1967), "Fairing and Elastica," Conference of the Japanese Society of Information Processing, 49-50. (Japanese)

Larkin, F. M. (1966), "An Interpolation Procedure Based on Fitting Elasticas to Given Data Points," Culham Operating System - Note No. 5/66, Theory Division, Culham Laboratory, April.

Lee, E. H. and G. E. Forsythe (1973), "Variational Study of Nonlinear Spline Curves," SIAM Review, Vol. 15, No. 1.

Love, A. E. H. (1927), The Mathematical Theory of Elasticity, 4th edition. London: Cambridge University Press.

Mangasarian, O. L. and L. L. Schumaker (1971), "Discrete Splines via Mathematical Programming," SIAM J. Control, Vol. 9, No. 2, 174-183.

Mehlum, Even (1969), Curve and Surface Fitting Based on Variational Criteria for Smoothness. Ph.D. Dissertation, Department of Mathematics, University of Oslo, 1969.

Podolsky, Boris and Harry H. Denman (1964), "Conditions on Minimization Criteria for Smoothing," Math. Comp. 18, 441-448.

Schweikert, Daniel G. (1966), "An Interpolation Curve Using a Spline in Tension," J. Math. and Phys. 45, 312-317.

Wilkinson, J.H. (1965), The Algebraic Eigenvalue Problem, London: Oxford University Press.

Woodford, C.H. (1969), "Smooth Curve Interpolation," BIT 9, 69-77.