c1

INTERACTIVE SPLINE APPROXIMATION

by

MARIAN MERCHANT

B.Sc., Simon Fraser University, 1971

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in the Department

of

Computer Science

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

January, 1974

Department of Computer Science

The University of British Columbia
Vancouver 8, Canada

Date Feb. 8, 1974

ABSTRACT

The use of spline basis functions in solving least squares approximation problems is investigated. The question as to which are appropriate basis functions to use is discussed along with the reasons why the final choice was made. The Householder transformation method for solving the fixed knot spline approximation problem is examined. Descriptions of both an automatic procedure using function minimization and an interactive procedure using a graphics terminal for solving the variable knot spline approximation problem are given. In conclusion, numerical results using the interactive system are presented and analyzed.

TABLE OF CONTENTS

TABLES

FIGURES

ACKNOWLEDGEMENT

There are many people who aided in the development of this thesis. Since I cannot hope to include all individuals who provided help; I shall mention only those who provided major assistance and apologize to anyone who may feel slighted by being omitted.

I would like to thank Dr. J. M. Varah who provided academic assistance through our discussions of the topic. Also I am grateful for the financial aid which he provided in the form of a reasearch assistantship.

I would like to thank the Department of Computer Science and the Computing Centre for the assistantships they provided. In particular I appreciated the IBM fellowship which I received for research in Interactive Numerical Analysis. This provided the initial ideas for this thesis.

Finally, I would like to thank my husband Peter for all the dishes he did during the writing of this thesis.

NOTATION

The following common notation is used throughout the thesis:

[n], [n,pp.---]	refers to reference n in the bibliography;
$x \in [a,b]$	$a \leq x \leq b$;
$x \in (a,b)$	$a < x < b$;
$s \in C^m[a,b]$	s has m continuous derivatives on [a,b];
$\left.\frac{d^j}{dx^j} s(x)\right\|_\delta$	is the j-th derivative of s with respect to x evaluated at δ;
$\omega'(x)$	is the first derivative of the function ω with respect to x;
$\Delta \equiv \{\delta_i : i=1,2,\ldots,k\}$	is the set Δ with elements $\delta_1, \delta_2, \ldots, \delta_k$;
$\{(x_\ell, y_\ell) : \ell=1,2,\ldots,n\}$	is the set of orderd pairs (x_ℓ, y_ℓ);
$\underline{a}$	is the vector $\underline{a}$; that is, $\underline{a} = \begin{pmatrix} a_1 \\ a_2 \\ \cdot \\ \cdot \\ \cdot \\ a_m \end{pmatrix}$
$\|\|\underline{a}\|\|$	is the least squares norm of $\underline{a}$;
$\min_{\underline{a}} f(\underline{a})$	is the minimum of the function f with respect to $\underline{a}$;
$\max\{c,d\}$	is the maximum of c and d;
Q^T	is the transpose of the matrix Q .

Section 1

SPLINES IN INTERACTIVE APPROXIMATION

Polynomials are frequently desired as a set of basis functions for approximation. Problems in obtaining accurate results with the standard set of basis functions $\{1,x,x^2,\ldots,x^m\}$ lead to the development of orthogonal polynomials as the set of basis functions. The use of orthogonal polynomials is a well-established technique in approximation. Consequently, they have been studied in depth and are not discussed further here.

Polynomials do have one disadvantage in approximation. That is, their nature over the entire region of approximation is determined by their behavior in only a small area of this region. Higher-order polynomials do not alleviate this problem but merely impose more oscillatory behavior on the approximation. However, polynomial splines can counteract this restrictive nature of polynomials.

Polynomial splines consist of piecewise polynomials connected at points known as knots over the region of approximation. Each piecewise polynomial determines the shape of the approximation in a small area relatively independently of surrounding areas. The amount of dependence is determined by imposing continuity requirements at the knots.

It is precisely the involvement of these knots that makes splines ideally suited for interactive approximation. Previously, automatic procedures involving the minimization of the least squares error were used to determine the best locations for the knots. As the knots occur non-linearly, appropriate initial guesses had to be made and there exists the possibility that the procedure would converge to a local minimum rather than the global minimum. Such a local minimum might not have been a suitable approximation.

Graphical interaction allows immediate contact with the solution procedure. The locations of the knots can be chosen visually and the approximation attempted with this set can be observed. The interactive procedure enables poor initial estimates for the knot locations to be eliminated. Also it allows for the manipulation of the knots until a satisfactory approximation is obtained. Using the knots obtained from the interactive procedure as initial guesses, an automatic procedure should converge rapidly to a suitable minimum.

THE SPLINE REPRESENTATION PROBLEM

2.1 Introduction

There is no universally accepted representation for spline basis functions. There are several known representations each with its own particular advantages and disadvantages.

Carasso and Laurent [4] discuss three methods of numerical construction of splines - a projection method, a method of direct resolution and a method using a basis. Of these, they recommend the use of a method involving a basis. With the choice of a reasonable basis, Carasso and Laurent conclude that this method provides more accurate results than the projection method and three times less computation than the method of direct resolution.

Greville [8] provides a comprehensive overview of basis functions for splines. From the definition of a spline function, he develops a representation using truncated power functions (discussed in Section 2.4) and one using B-splines (discussed in Section 2.5). de Boor and Rice [5] summarize these representations and also include a representation involving piecewise continuous polynomials (discussed in Section 2.3).

Schultz [17] gives a general basis for B-splines. In [18], Schultz describes the representation for cubic B-splines in more detail. The basis function resulting from applying the set of cubic B-splines to the special case of uniformly spaced knots is stated. The derivation of this result is given in detail in Section 2.6.

2.2 Definition of a Spline Function

Although splines exist in engineering and drafting as a device for curve smoothing, the basic mathematical formulation of a spline function comes from piecewise continuous polynomials. The mathematical definition formalizes the engineering concept.

Definition 1; A (polynomial) spline function of degree m on $[a,b]$ is a polynomial of degree m which is in $C^{m-1}[a,b]$.

Although this definition incorporates the basic notion of a spline function, it does not provide the essential components needed for the use of splines in numerical problem-solving. For this purpose, the following, more constructive definition of splines is better.

Definition 2; Given a partition $a = \delta_0 < \delta_1 < \ldots < \delta_k < \delta_{k+1} = b$ then a (polynomial) spline function of degree m with k internal knots $\delta_1, \delta_2, \ldots, \delta_k$ on $[a,b]$ is a function $S(x)$ with the following properties:

1. $S(x)$ is a polynomial of degree m or less in $[\delta_i, \delta_{i+1}]$, $i = 1, 2, \ldots, k$;
2. $S(x)$ and its derivatives of orders $1, 2, \ldots, m-1$ are continuous everywhere.

Let $\Delta \equiv \{\delta_i \; ; \; i = 0, 1, \ldots, k+1\}$ be the set of knots and $S(x) \equiv \{s_i(x) \; ; \; i = 0, 1, \ldots, k\}$ be the set of polynomials such that $s_i(x)$ is in $[\delta_i, \delta_{i+1}]$ for $i = 0, 1, \ldots, k$. The set $S(x)$ must satisfy the following continuity conditions at the knots:

$$\frac{d^j}{dx^j} s_{i-1}(x)\Big|_{\delta_i} = \frac{d^j}{dx^j} s_i(x)\Big|_{\delta_i}$$

for $i = 1, 2, \ldots, k$; $j = 0, 1, \ldots, m - 1$.

2.3 Piecewise Continuous Polynomial Spline Representation

The piecewise continuous polynomial definition (2) can be formulated into an approximation problem. Suppose that on each interval $[\delta_i, \delta_{i+1}]$ for $i = 0, 1, \ldots, k$; the data is approximated by a polynomial of degree m or less. Given the set of coefficients $\{c_{ij} : i = 0, 1, \ldots, k;\ j = 1, 1, \ldots, m\}$ the problem is to find the c_{ij}'s where

$$S(x) \equiv s_i(x) \equiv \sum_{j=0}^{m} c_{ij}(x-\delta_i)^j$$

for $\delta_i \leq x \leq \delta_{i+1}$ where $i = 0, 1, \ldots, k$.

This system results in $(m + 1)(k + 1) = mk + m + k + 1$ unknowns c_{ij} which is mk more than are required for a non-redundant spline representation. Therefore it is only necessary to compute the set $\{c_{ij} : i = 0$ and $j = 0, 1, \ldots, m$ and $i = 1, 2, \ldots, k$ and $j = m\}$. The remaining coefficients can be computed from the constraints derived from the continuity conditions; that is

$$c_{ij} = \frac{1}{j!} \frac{d^j}{dx^j} s_{i-1}(x)\Big|_{\delta_i}$$

for $i = 1, 2, \ldots, k$; $j = 0, 1, \ldots, m - 1$.

This system is useful for approximation as it gives an explicit representation for each piecewise polynomial between each knot pair. But despite the fact that the basis functions are defined over a particular knot interval, they must be computed over the entire interval. Also, the system of equations formed tend to be ill-conditioned; that is, the resulting solution is not accurate. Since this representation is identical to the mathematical representation between each knot pair, ill-conditioning occurs for the same reason (as described in Section 2.4).

2.4 Mathematical Spline Representation

The standard representation of splines is that of elementary splines (alias truncated power functions). This representation is used mainly in mathematical analysis. Most theorems involving spline functions are derived and proved using elementary splines as they are easy to manipulate analytically.

Definition 3: An elementary spline function of degree m, y_+^m, is defined by

$$y_+^m = \begin{cases} y^m & \text{for } y > 0 \\ 0 & \text{for } y \leq 0 \end{cases} .$$

Elementary splines give rise to a set of basis functions for splines. In particular the set

$$\{1, x, x^2, \ldots, x^m, (x-\delta_1)_+^m, \ldots, (x-\delta_k)_+^m\}$$

forms a set of basis functions for a spline of degree m. An example of these basis functions for a cubic spline with four uniformly spaced internal knots is given in Figure 1 .

These basis functions can also be formulated into an approximation problem. Given the set of coefficients $\{a_i : i = 1, 2, \ldots, m + k + 1\}$ the problem is the determination of the a_i's where

$$S(x) = \sum_{i=1}^{m+1} a_i x^{i-1} + \sum_{i=1}^{k} a_{i+m+1}(x-\delta_i)_+^m \quad .$$

This system results in $m + k + 1$ unknowns - exactly the number needed for a unique representation. Therefore all the coefficients must be computed.

Despite its simplicity the mathematical representation of splines should never be used for computational purposes. Splines computed by this method will produce ill-conditioned systems of equations as $m + k + 1$ increases. Intuitively, a reason for this can be seen from the example plotted in Figure 1. Notice that the last basis function $(x-\delta_4)_+^3$ is zero nearly everywhere as well as being extremely small relative to the other basis functions. Consequently it is possible to produce a linear combination of these basis functions which is almost zero; that is, the set of basis functions is almost linearly dependent. Using this set will produce a system of linear equations for the approximation problem whose corresponding matrix is nearly singular. This matrix will be ill-conditioned in most cases. Hence it is better to choose basis functions which are more difficult to conceive analytically but are more stable computationally.

ELEMENTARY CUBIC SPLINE BASIS FUNCTIONS

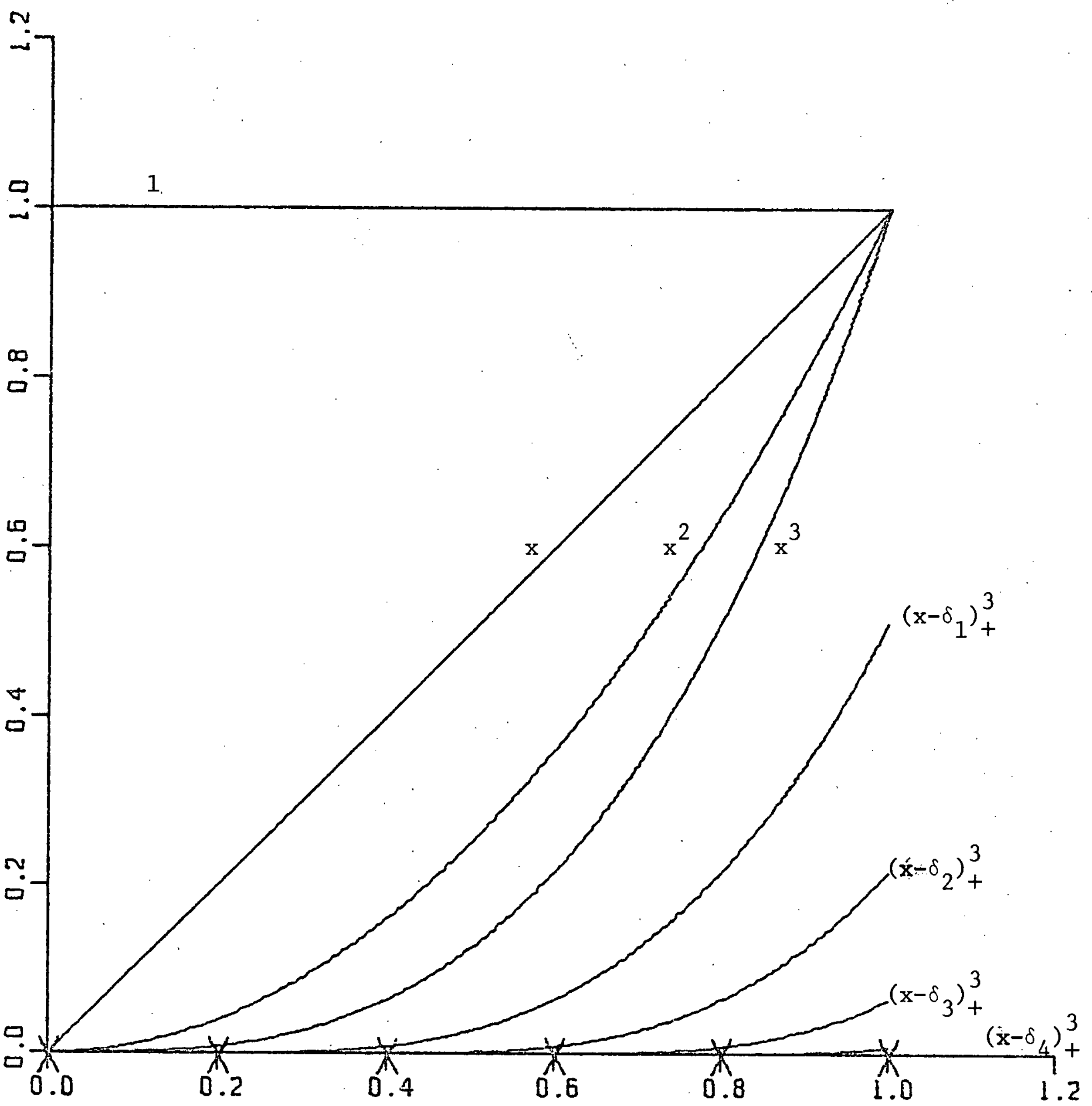


FIGURE 1

2.5 B-spline Representation

In deriving a set of basis functions which produce well-conditioned systems; one would like to satisfy these two criteria;

1. That the support (region in which the functional value is non-zero) of the splines is finite;
2. That the number of knot intervals involved in the support is minimal (as small as possible).

To this end a set of basis functions is derived on the concept of divided differences.

Definition 4: The functional D defined by

$$Df = f(\delta_i, \delta_{i+1}, \ldots, \delta_{i+m+1}) = \frac{f(\delta_{i+1}, \ldots, \delta_{i+m+1}) - f(\delta_i, \ldots, \delta_{i+m})}{\delta_{i+m+1} - \delta_i}$$

is the divided difference of f of order $m + 1$.

The divided difference depends linearly on $f(x)$. Also, and more important, the divided difference of order $m + 1$ is zero for any polynomial of degree m.

Now, for any function g, by the Lagrange interpolation formula,

$$g(\delta_i, \delta_{i+1}, \ldots, \delta_{i+m+1}) = \sum_{n=0}^{m+1} \frac{g(\delta_{i+n})}{\omega'(\delta_{i+n})}$$

where $\omega(x) = (x-\delta_i)\cdot(x-\delta_{i+1})\cdots(x-\delta_{i+m+1})$. In order to define B-splines let the function $g(\delta_{i+n})$ be the (m+1)st divided difference of $(\delta_{i+n}-x)_+^m$. Substituting for g, the Lagrange interpolation formula gives

$$g(\delta_i,\delta_{i+1},\ldots,\delta_{i+m+1}) = \sum_{n=0}^{m+1} \frac{(\delta_{i+n}-x)_+^m}{\omega'(\delta_{i+n})}$$

where $\omega(x)$ is as previously defined.

Letting i range over $-m$ to k gives exactly the number of B-splines required to form a set of basis functions; that is $m + k + 1$ functions. Thus, the set $\{s_i(x) : i = -m, -m+1, \ldots, k\}$ where

$$s_i(x) = \sum_{n=0}^{m+1} \frac{(\delta_{i+n}-x)_+^m}{\omega'(\delta_{i+n})}$$

with $\omega(x) = (x-\delta_i)\cdot(x-\delta_{i+1})\cdots(x-\delta_{i+m+1})$ is precisely the set of B-spline basis functions.

To complete the definition, this set requires the addition of 2m supplementary knots to the original knot set $\Delta \equiv \{\delta_0,\delta_1,\ldots,\delta_{k+1}\}$. These knots must be external to the original knot set with m knots less than δ_0 and m knots greater than δ_{k+1} . One possible method of choosing these extra knots is the following:

$$\delta_{-i} = \delta_0 - \frac{(\delta_{k+1}-\delta_0)\cdot i}{k+1}$$

$$\delta_{i+k+1} = \delta_{k+1} + \frac{(\delta_{k+1}-\delta_0)\cdot i}{k+1}$$

for $i = 1, 2, \ldots, m$.

For B-splines it can be shown that:

1. $s_i(x)$ is strictly positive in $[\delta_i, \delta_{i+m+1}]$;
2. The support of $s_i(x)$ is finite and restricted to the interval $[\delta_i, \delta_{i+m+1}]$;
3. Any spline $S(x)$ can be uniquely represented as a linear combination of

$$\{s_i(x) : i = -m, -m+1, \ldots, k\} .$$

It is simple to formulate an approximation problem from B-spline functions. Given the set of coefficients $\{a_i : i = -m, \ldots, k\}$ the problem is to find the a_i's in

$$S(x) = \sum_{i=-m}^{k} a_i s_i(x)$$

where $\{s_i(x) : i = -m, -m+1, \ldots, k\}$ is the set of basis functions for B-splines. This system results in $m + k + 1$ unknowns as in the mathematical system. There are no redundant parameters and hence all the coefficients must be computed.

The system of equations derived using B-splines remains well-conditioned as $m + k + 1$ increases. In fact, one can produce numerical upper bounds on the condition number of the matrix of normal equations for a uniformly spaced knot set following the method described in Schultz [18, pp. 70-72]. Also, because the basis functions give minimal support the systems produced are banded with the band-width dependent on the degree of the spline.

2.6 Example of B-spline Representation

To give a concrete example of what B-spline basis functions look like, consider the particular case of cubic splines on a uniform partition. An explicit representation for the basis functions $\{s_i(x) : i = -3, -2, \ldots, k\}$ can be developed in the following manner:

Given a uniform partition, the mesh length is

$$h = \frac{1}{k+1} \quad \text{and therefore the} \quad i\text{-th knot is} \quad \delta_i = \frac{i}{k+1}$$

for $i = -3, -2, \ldots, k+3, k+4$. Thus

$$s_i(x) = \sum_{n=0}^{4} \frac{(\delta_{i+n} - x)_+^3}{\omega'(\delta_{i+n})}$$

for $i = -3, -2, \ldots, k$ with

$$\omega'(x) = \prod_{\substack{j=0 \\ j \neq n}}^{4} (x - \delta_{i+j}) \; .$$

Substituting for the knots gives

$$s_i(x) = \sum_{n=0}^{4} \frac{\left(\frac{i+n}{k+1} - x\right)_+^3}{\omega'\left(\frac{i+n}{k+1}\right)}$$

$$= \sum_{n=0}^{4} \frac{(i+n - (k+1)x)_+^3}{(k+1)^3} \cdot \frac{1}{\omega'\left(\frac{i+n}{k+1}\right)} \; .$$

Now

$$\omega'\left(\frac{i+n}{k+1}\right) = \prod_{\substack{j=0 \\ j\neq n}}^{4} \frac{(i+n-i-j)}{k+1}$$

$$= \prod_{\substack{j=0 \\ j\neq n}}^{4} \frac{(n-j)}{k+1}$$

$$= \frac{1}{(k+1)^4} \prod_{\substack{j=0 \\ j\neq n}}^{4} (n-j) \; .$$

Substituting back into $s_i(x)$:

$$s_i(x) = \sum_{n=0}^{4} \frac{(i+n-(k+1)x)_+^3}{(k+1)^3} \cdot \frac{(k+1)^4}{\prod_{\substack{j=0 \\ j\neq n}}^{4} (n-j)}$$

$$= (k+1) \sum_{n=0}^{4} \frac{(i+n-(k+1)x)_+^3}{\prod_{\substack{j=0 \\ j\neq n}}^{4} (n-j)}$$

Expanding for n and j :

$$s_i(x) = (k+1) \left\{ \frac{(i-(k+1)x)_+^3}{24} - \frac{(i+1-(k+1)x)_+^3}{6} + \frac{(i+2-(k+1)x)_+^3}{4} - \frac{(i+3-(k+1)x)_+^3}{6} + \frac{(i+4-(k+1)x)_+^3}{24} \right\}$$

for $i = -3, -2, \ldots, k$.

Letting $x' = (k+1)x - i - 2$ gives

$$s_i(x) = (k+1)\{\frac{(-2-x')_+^3}{24} - \frac{(-1-x')_+^3}{6} - \frac{(x')_+^3}{4} - \frac{(1-x')_+^3}{6} + \frac{(2-x')_+^3}{24}\}$$

and defining $S(x') = s_i((k+1)x - i - 2)$:

$$S(x') \equiv (k+1)\begin{cases} 0 & x' \leq -2 \\ \frac{(1+x')^3}{6} - \frac{(x')^3}{4} - \frac{(1-x')^3}{6} + \frac{(2-x')^3}{24} & -2 \leq x' \leq -1 \\ -\frac{(x')^3}{4} - \frac{(1-x')^3}{6} + \frac{(2-x')^3}{24} & -1 \leq x' \leq 0 \\ -\frac{(1-x')^3}{6} + \frac{(2-x')^3}{24} & 0 \leq x' \leq 1 \\ \frac{(2-x')^3}{24} & 1 \leq x' \leq 2 \\ 0 & 2 \leq x' . \end{cases}$$

When the explicit representation is used on a uniform partition of [0,1] with four internal knots; the set $\{s_i(x) : i = -3, -2, \ldots, 4\}$ is as shown in Figure 2.

Equivalence of Spline Representations

Although B-splines provide a suitable method for solving spline approximation problems the coefficients obtained are not extremely useful. In particular, it is preferable to know the coefficients of the piecewise continuous polynomials between adjacent knots than to know the

CUBIC B-SPLINE BASIS FUNCTIONS

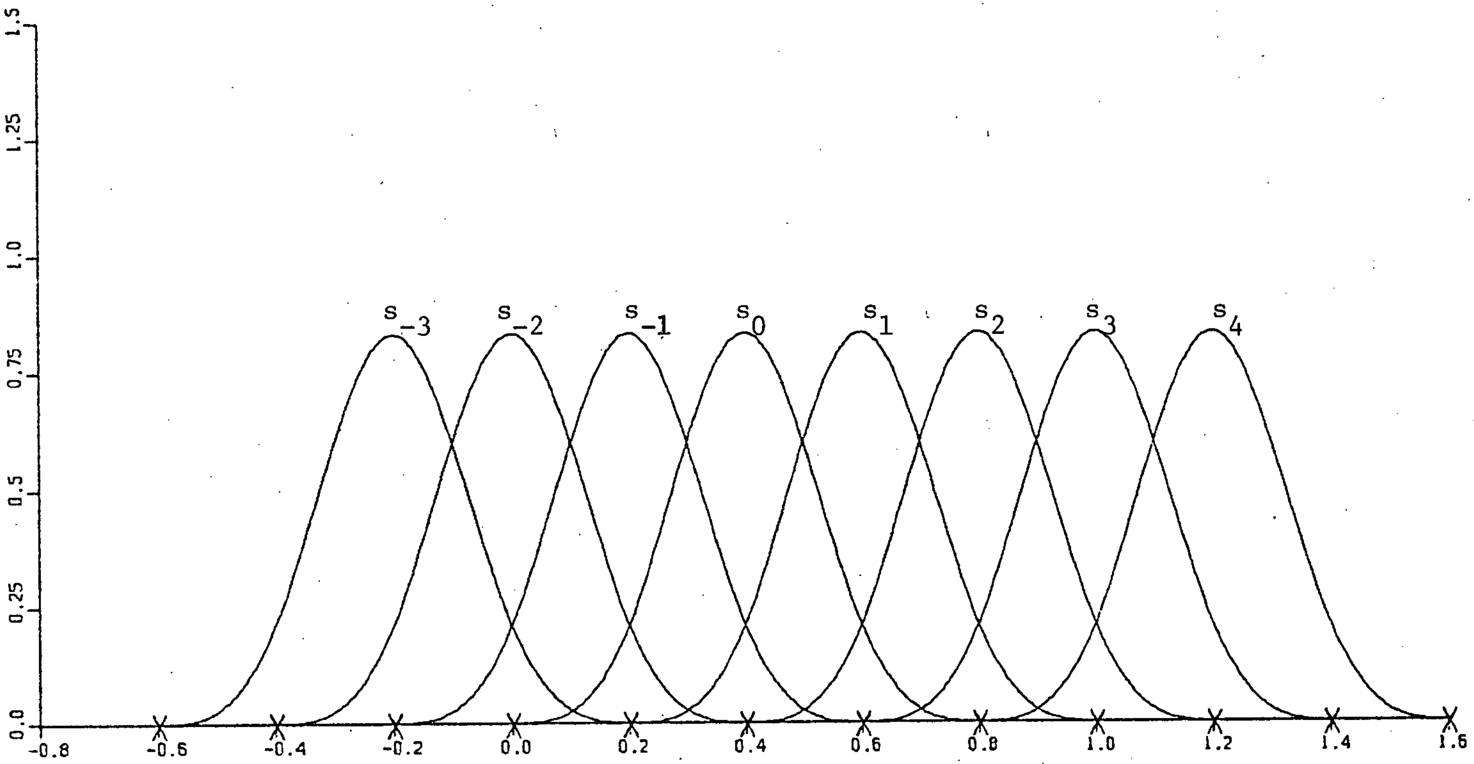


FIGURE 2

coefficients of B-spline basis functions. To this end, it is appropriate to derive the unknown set of the piecewise continuous polynomial coefficients $\{c_{ij} : i = 0, 1, \ldots, k, j = 0, 1, \ldots, m\}$ from the known values of the B-spline coefficients $\{a_i : i = 1, 2, \ldots, m + k + 1\}$.

Consider the functional value of the spline for each knot; that is, at $x = \delta_i$.

Case I : For piecewise continuous polynomials

$$c_{ij} = \frac{1}{j!} \frac{d^j}{dx^j} s_{i-1}(x)\Big|_{\delta_i} = \frac{1}{j!} \frac{d^j}{dx^j} S(x)\Big|_{\delta_i}$$

for $i = 1, 2, \ldots, k + 1; \quad j = 0, 1, \ldots, m$.

Case II: For B-splines

$$\frac{d^j}{dx^j} S(x)\Big|_{\delta_i} = \frac{d^j}{dx^j} \sum_{\ell=-m}^{k} a_\ell s_\ell(x)\Big|_{\delta_i}$$

$$= \sum_{\ell=-m}^{k} a_\ell \frac{d^j}{dx^j} s_\ell(x)\Big|_{\delta_i}$$

for $i = 1, 2, \ldots, k + 1; \quad j = 0, 1, \ldots, m$.

Now

$$\frac{d^j}{dx^j} s_\ell(x)\Big|_{\delta_i} = \frac{d^j}{dx^j} \sum_{n=0}^{m+1} \frac{(\delta_{\ell+n} - x)_+^m}{\omega'(\delta_{\ell+n})}\Bigg|_{\delta_i}$$

$$= \sum_{n=0}^{m+1} \frac{1}{\omega'(\delta_{\ell+n})} \frac{d^j}{dx^j} (\delta_{\ell+n} - x)_+^m \Big|_{\delta_i}$$

Differentiating with respect to x :

$$\frac{d^j}{dx^j} (\delta_{\ell+n} - x)_+^m \Big|_{\delta_i} = \begin{cases} (m-j)\,!\ (\delta_{\ell+n} - x)_+^{m-j} & \delta_{\ell+n} \geq \delta_i \\ 0 & \delta_{\ell+n} < \delta_i \end{cases} .$$

By equating the terms of the derivatives:

$$c_{ij} = \frac{1}{j!} \sum_{\ell=-m}^{k} a_\ell \frac{d^j}{dx^j} s_\ell(x) \Big|_{\delta_i}$$

for $i = 1, 2, \ldots, k + 1$; $j = 0, 1, \ldots, m$.

Section 3

THE LINEAR APPROXIMATION PROBLEM

3.1 Introduction

Several methods are known for solving linear approximation problems. These methods can be applied to problems involving general sets of basis functions. Spline approximation with a fixed knot set is a particular application of the general problem.

de Boor and Rice [5] describe an approximation method involving orthogonal projection. The basic idea is to minimize the error $||y-u||$ of approximating y by u by the orthogonal projection Py of y. Py is best calculated using an orthonormal basis. Therefore, given a general set of basis functions for the approximation, an orthonormal set of basis functions must be derived. de Boor and Rice use a modified Gram-Schmidt process to generate such a set of basis functions.

The most common technique used for solving linear approximation problems is the method of normal equations (described in Section 3.3). Patent [13] discusses the general linear least squares problem and linear least squares problem using splines in detail giving results concerning the uniqueness of the solution and the symmetric and positive definite properties of the associated least squares matrix. Patent also includes a program solving the spline approximtion problem with fixed knots. The basis functions used in generating the system of normal equations were B-splines.

Smith [19, pp. 110-119] develops the method of normal equations for the particular case of spline approximation. The particular set of basis functions used are the mathematical representation. The least squares matrix derived using these basis functions is given in full.

Golub [7] develops a method using Householder transformations (described in Section 3.3). An Algol program based on this procedure for a general set of basis functions is given in Businger and Golub [3].

Although a suitable set of basis functions is available which prevents ill-conditioning as the number of knots increases, there is still the problem of preventing ill-conditioning as the knots become non-uniformly spaced. The orthogonal projection method counteracts this problem because the basis functions are orthonormalized before solution. The method of normal equations frequently produces ill-conditioned systems as shown in the example cited in Golub [7]. Solving the spline approximation problem using normal equations on a non-uniformly knot set is a prime example of this ill-conditioning. However, the method of Householder transformations counteracts this problem because of the orthogonality of the transformations. Consequently, this method is necessary for a stable solution to the spline approximation problem.

3.2 Definition of the Linear Problem

All aspects of the linear problem are incorporated in the following definition:

Definition 5: Given a discrete set of data

$$\{(x_\ell, y_\ell) : \ell = 1, 2, \ldots, n\}$$

and a function

$$S(x) = \sum_{i=1}^{m} a_i s_i(x)$$

where $\{s_i(x) : i = 1, 2, \ldots, m\}$ is a set of basis functions and $\{a_i : i = 1, 2, \ldots, m\}$ is a set of unknown coefficients occurring linearly; the linear approximation problem is to determine values for the a_i's to produce the "best fit" of $S(x)$ to the set of data.

3.3 Solution of the Linear Problem

Most approximation problems consider the minimization of the least squares error as satisfying the "best fit" criteria. The particular form of the least squares error used in this case is the square of the least squares norm where:

Definition 6: Given a vector $\underline{v} = \begin{pmatrix} v_1 \\ v_2 \\ \cdot \\ \cdot \\ \cdot \\ v_n \end{pmatrix}$;

the least squares norm of $\underline{v}$, $||\underline{v}||$, is

$$\left(\sum_{\ell=1}^{n} v_\ell^2\right)^{\frac{1}{2}} .$$

Thus, given the function $S(x)$ and the data set $\{(x_\ell, y_\ell) : \ell = 1, 2, \ldots, n\}$ the solution of the linear problem becomes the minimization of the least squares error by the appropriate choice of the unknowns $\{a_i : i = 1, 2, \ldots, m\}$. That is, by finding

$$\min_{\{a_i\}} \sum_{\ell=1}^{n} [y_\ell - \sum_{i=1}^{m} a_i s_i(x_\ell)]^2 .$$

The usual method of solution, that of normal equations, is developed in the following way:

In order to find the minimum

$$\min_{\{a_i\}} \sum_{\ell=1}^{n} [y_\ell - \sum_{i=1}^{m} a_i s_i(x_\ell)]^2 .$$

differentiate the summation with respect to each of the parameters $\{a_i : i = 1, 2, \ldots, m\}$ and set to zero. Thus

$$\frac{\partial}{\partial a_j} \{\sum_{\ell=1}^{n} [y_\ell - \sum_{i=1}^{m} a_i s_i(x_\ell)]^2\}$$

$$= -2 \sum_{\ell=1}^{n} [y_\ell - \sum_{i=1}^{m} a_i s_i(x_\ell)] \cdot s_j(x_\ell)$$

$$= 0$$

for $j = 1, 2, \ldots, m$. Rearranging the terms gives

$$\sum_{i=1}^{m} a_i [\sum_{\ell=1}^{n} s_i(x_\ell) \cdot s_j(x_\ell)] = \sum_{\ell=1}^{n} y_\ell s_j(x_\ell)$$

for $j = 1, 2, \ldots, m$.

Using the following changes;

1. Denote by S the m x m matrix

$$S \equiv s_{ij} := \sum_{\ell=1}^{n} s_j(x_\ell) \cdot s_i(x_\ell)$$

for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, m$;

2. Denote by $\underline{y}$ the m-dimension vector

$$\underline{y} \equiv y_j = \sum_{\ell=1}^{n} y_\ell s_j(x_\ell)$$

for $j = 1, 2, \ldots, m$; and

3. Denote by $\underline{a}$ the m-dimension vector of unknown coefficients $\underline{a} \equiv \{a_i : i = 1, 2, \ldots, m\}$;

the problem becomes one of finding the solution to the system of equations $S\underline{a} = \underline{y}$.

Another method of solving

$$\min_{\{a_i\}} \sum_{\ell=1}^{n} [y_\ell - \sum_{i=1}^{m} a_i s_i(x_\ell)]^2$$

is by using orthogonal transformations. This requires the following changes:

1. Denote by S the n x m matrix of function values; that is,

$$S \equiv s_{\ell i} = s_i(x_\ell)$$

for $\ell = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, m$;

2. Denote by $\underline{y}$ the n-dimension vector of ordinates

$$\underline{y} \equiv \{y_\ell : \ell = 1, 2, \ldots, n\} \text{ ; and}$$

3. Denote by $\underline{a}$ the m-dimension vector of unknown coefficients

$$\underline{a} \equiv \{a_i : i = 1, 2, \ldots, m\} \text{ ;}$$

then the problem becomes to find

$$\min_{\underline{a}} ||\underline{y} - S\underline{a}||^2 \, .$$

Consider multiplying the previous equation by an orthogonal matrix Q^T. Because multiplying by an orthogonal matrix does not change the norm; the linear least squares problem remains the same. Thus the problem becomes to find

$$\min_{\underline{a}} ||Q^T\underline{y} - Q^TS\underline{a}||^2 \, .$$

Now consider Q^T to be a series of orthogonal transformations which transforms Q^TS into an upper triangular matrix R. If such a series can be found then the linear least squares problem reduces to finding

$$\min_{\underline{a}} ||Q^T\underline{y} - R\underline{a}||^2 \, .$$

Since the zero part of R is independent of $\underline{a}$; it is only necessary to solve the system $R\underline{a} = \underline{b}$ where $b_i = (Q^T\underline{y})_i$ for $i = 1, 2, \ldots, m$.

The remainder of $Q^T \underline{y}$ contains the least squares error; that is,

$$||\underline{y} - S\underline{a}||^2 = \sum_{i=m+1}^{n} (Q^T y)_i^2 .$$

There exists a series of orthogonal transformations Q^T which will reduce S to an upper triangular matrix known as Householder transformations. They can be constructed as follows:

> Given a vector $\underline{v}$ construct a symmetric orthogonal matrix P such that $P\underline{v} = \underline{w}$ where $\underline{w}$ is a unit vector whose first element is $\pm||\underline{v}||$ and whose remaining elements are zero.

Householder showed that for any two vectors $\underline{v}$, $\underline{w}$ with $\underline{v}^T\underline{v} = \underline{w}^T\underline{w}$ there exists a symmetric orthogonal matrix $P = I - 2\underline{u}\underline{u}^T$ such that $\underline{w} = P\underline{v}$. The symmetry and orthogonality of P is proven in Acton [1, p. 327] .

The problem now is to determine the required vector $\underline{u}$ in P . The method is described in Acton [1, pp. 324-329] with slight modifications and the derived $\underline{u}$ is

$$\underline{u} = \frac{1}{K} \begin{pmatrix} v_1 \pm ||\underline{v}|| \\ v_2 \\ \cdot \\ \cdot \\ \cdot \\ v_n \end{pmatrix} \quad \text{where} \quad K^2 = 2||\underline{v}||^2 \pm 2v_1||\underline{v}|| .$$

Two computational considerations come into effect when using Householder transformations. The first is which sign to choose in computing $\underline{u}$. The choice is to pick K^2 such that

$$K^2 = \max\{\underbrace{2||\underline{v}||^2 + 2v_1||\underline{v}||}_{K_1}, \underbrace{2||\underline{v}||^2 - 2v_1||\underline{v}||}_{K_2}\}$$

$$= \max\{K_1, K_2\}$$

in order to avoid cancellation. Thus if $v_1 \geq 0$ choose K_1; if $v_1 < 0$ choose K_2 .

The second consideration is the computation of $P\underline{v}$. Rewriting

$$P\underline{v} = (I - 2\underline{u}\underline{u}^T)\underline{v}$$

as

$$P\underline{v} = I\underline{v} - 2\underline{u}\underline{u}^T\underline{v}$$

$$= \underline{v} - \underline{u}(2\underline{u}^T\underline{v})$$

the scalar $2\underline{u}^T\underline{v}$ is computed first followed by the vector subtraction. Hence the matrix P need never be formed explicitly. This method is far more efficient than forming P and performing a matrix multiplication.

To manipulate Householder transformations to form the upper triangular matrix consider applying P to the matrix S . This is equivalent to applying P to each column in S . That is, there is a P_1 such that P_1S reduces column 1 of S, $\underline{w}_1$, to

$$\underline{w}_1' = \begin{pmatrix} \pm||\underline{w}_1|| \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$$

where $\underline{w}_1'$ is the transformed column 1 of S . Note that all the other columns of S are altered by this transformation.

Similarly, there is a P_2 such that P_2S reduces column 2 of S, $\underline{w}_2$, to

$$\underline{w}_2' = \begin{pmatrix} \pm||\underline{w}_2|| \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$$

where $\underline{w}_2'$ is the transformed column 2 of S . However all other columns of S are altered also. In particular, column 1 reverts back to non-zero status which is not desirable.

Therefore, in order to preserve the zeroes in column 1 let P_2 be the Householder transformation which reduces column 2 of S, $\underline{w}_2$, to

$$\underline{w}_2' = \begin{pmatrix} w_2^1 \\ \pm||\underline{w}_2|| \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$$

where $\underline{w}_2'$ is the transformed column 2 of S and w_2^1 is the first element of $\underline{w}_2'$. This transformation will leave column 1 unchanged but will alter all the remaining columns of S . Continuing the process m times; S can be reduced to an upper triangular matrix of the form

$$P_m \cdots P_2 P_1 S = \begin{pmatrix} \pm\|\underline{w}_1\| & w_2^1 & w_3^1 & \cdot & \cdot & \cdot & w_m^1 \\ & \pm\|\underline{w}_2\| & w_3^2 & & & & \cdot \\ 0 & 0 & \cdot & & & & \cdot \\ \cdot & & & \cdot & & & \cdot \\ \cdot & & & & \cdot & & \cdot \\ \cdot & & & & & \cdot & w_m^{m-1} \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \pm\|\underline{w}_m\| \\ 0 & & & & & & 0 \\ \cdot & & & & & & \cdot \\ \cdot & & & & & & \cdot \\ \cdot & & & & & & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & 0 & 0 \end{pmatrix}.$$

Thus in the linear least squares problem using $Q^T = P_m \cdots P_2 P_1$ and applying Q^T to S, Q^T reduces S to the upper triangular matrix R.

3.4 Solution of the Fixed Knot Problem

The fixed knot problem for splines is a particular application of the general linear least squares problem. In the definition 5 for the linear problem allow the functions $s_i(x)$ to be the B-spline basis functions with a fixed knot set. This creates the fixed knot least squares approximation problem for splines. This problem can be solved exactly like the general problem using Householder transformations. However because a particular basis set is being used a few computational considerations come into effect.

The first of these is that of basis function evaluation. Because the basis functions are evaluated often it is important that an efficient method of computation be available. In the fixed knot case the values of the basis function denominator $\omega'(x)$ remain constant throughout all computations. Therefore it is advantageous to calculate $\omega'(x)$ initially and retain the values for future use.

Also, because the basis functions have minimal support it is only necessary to compute a basis functional value if it is within the range of support. Otherwise the functional value is 0 . It is possible to calculate the function at all points without testing for the region of support. This has two disadvantages. First it is more time-consuming to calculate a value than to test for the range. Second because of round-off error, the summation will be the order of machine accuracy rather than zero.

The second computational consideration is the reduction of the resulting least squares matrix S . Because the basis functions have limited support the matrix S has a banded structure. This banded structure is not the familar bandedness usually associated with matrices but rather a special structure dependent on the location of the abscissa point x_ℓ . If the data point x_ℓ is outside the support region, that is, $x_\ell \notin [\delta_i, \delta_{i+m+1}]$ then the $s_{\ell i}$ element of S will be 0 . Hence the matrix will have blocks of non-zero elements along the diagonal where each non-zero block of the matrix is formed from the values of x_ℓ where $\delta_i \le x_\ell \le \delta_{i+m+1}$. This special banded structure can be used in the fixed knot approximation. Since the zeroes are unchanged by manipulation by Householder transformations it is only necessary to reduce the banded part of S . This causes a considerable reduction of computational effort.

The third computational consideration is the condition number of the matrix. Since S is initially a non-square matrix it is difficult to state anything about its condition number. However the orthogonal transformations do not affect the norm (and hence the condition number) of S in any way. Thus, by determining the condition number of the square upper triangular part of the matrix R the condition of S has effectively been determined. The stipulation for a well-conditioned matrix is that the condition number be less than $m + k + 1$ and that the condition number for a given problem should remain independent of the location of the knots.

Section 4

THE VARIABLE KNOT APPROXIMATION PROBLEM

4.1 Introduction

The fixed knot problem has been investigated thoroughly and adequate methods exist for its solution. The variable knot problem involves choosing locations for the knots so as to provide the "best approximation" possible. Because the knots occur non-linearly, this problem is more complex.

One alternative to solving the variable knot problem is to minimize the least squares error with respect to the knots. Another alternative is to position the knots visually to arrive at an approximation which is reasonable to the eye although not necessarily the "best" in the least squares sense. This requires graphical interaction with the spline approximation problem. By interaction a reasonable approximation can be derived manually and good initial guesses for an automatic minimization technique can be obtained.

de Boor and Rice [6] solve the variable knot problem by an automatic technique. They minimize the least squares error in integral form

$$\{\int_a^b [y - S(x,\Delta)]^2\}^{\frac{1}{2}}$$

over all splines of degree m with k knots. The trapezoidal rule is used to obtain an approximation to this integral. A discrete Newton's method is applied to minimize each knot individually while the rest of the knots remain stationary. The knots are optimized by sweeping through the knot set from right to left and re-evaluating the fixed knot problem each time.

Smith [19, p. 110-119] presents the use of splines in interactive data fitting. Although he does not allow for the possibility of respecifying certain knot locations; he does allow the possibility of respecifying the entire knot set and attempting the fixed knot problem again.

4.2 Definition of the Variable Knot Problem

Whereas the linear least squares problem can be generalized to any set of basis functions; the definition of the non-linear approximation problem is restricted to spline functions and is referred to as the variable knot problem.

Definition 7: Given a discrete set of data

$$\{(x_\ell, y_\ell) : \ell = 1, 2, \ldots, n\} ;$$

a set of knots in strictly increasing order

$$\Delta = \{\delta_i : i = 1, 2, \ldots, k\} ;$$

a set of spline basis functions

$$\{s_i(x) : i = -m, -m + 1, \ldots, k\} ;$$

and a set of linear coefficients

$$\{a_i : i = -m, -m + 1, \ldots, k\}$$

with $S(x,\Delta) = \sum_{i=-m}^{k} a_i s_i(x,\Delta)$

the variable knot problem is to determine values for the set Δ to produce the "best fit" of S to the set of data.

Note that in this case the unknown parameters are the knots - not the linear coefficients $\{a_i : i = -m, -m + 1, \ldots, k\}$. To be completely correct both the coefficients and the knots should be evaluated simultaneously to produce the "best fit". This problem is much more difficult. However, a feasible alternative is for each knot set Δ, to pick the coefficients $\{a_i : i = -m, -m + 1, \ldots, k\}$ by solving the fixed knot problem.

4.3 Solution of the Variable Knot Problem

In the introduction to this section two alternatives were proposed for solving the variable knot problem: an automatic procedure and an interactive procedure. It is best to discuss these in the reverse order since the first follows inherently from the second.

The use of interaction for solving spline approximation problems can be summarized in the following steps:

1. Read in the data set and graph it on a graphics terminal.
2. Specify an initial set of knots and overlay their location on the x-axis.
3. Solve the fixed knot problem using this knot set and overlay the resulting spline curve.
4. Allow respecification of the location of any of the knots.
5. Recalculate the fixed knot problem using the new knot set and graph the resulting spline approximation curve if it is wanted.
6. Allow the options of respecifying knots, changing the number of knots, or optimizing the knot set.

This interactive procedure produces a reasonable fit much faster than an automatic procedure. For example, if the original knot set gives a poor approximation, the situation can immediately be remedied by manipulating the knot set drastically as opposed to the more cautious procedures of automatic techniques. This approach allows an extremely fast initial approach to a good fit. Then automatic refinement could use the resulting knot set to reach an optimal knot set quickly.

There are several possible approaches for positioning the knots. de Boor and Rice [6, p. 12-18] present some of these for an automatic procedure but variations seem suitable for interactive placement.

One possible approach suggested is that additional knots be placed near the location of the maximum error. This seems reasonable initially as it is usually desired to produce a better fit in that area. Eventually, however, the data in that region will become interpolated which is not the desired phenomenon. Also this procedure does not compensate for the appropriate placement of fewer knots over the entire range rather than the concentration of knots in one place.

Another possibility is to place knots at positions of rapid change in the data. This allows the polynomial to determine its own shape in the interval nearly independently of the surrounding interval. This is because at positions of rapid change, the highest order term of the piecewise polynomial dominates. It is precisely this term that is not included in the continuity constraints. This helps overcome one of the basic problems with polynomials - that their oscillatory nature makes it difficult for them to adequately approximate data.

With a graph of the data within reach it is possible to make reasonable predictions about knot placement. This is the greatest value of graphical interaction - the data and the ability to manipulate the approximation are directly at hand.

4.4 Knot Optimization

Despite the fact that reasonable approximations can be made to a set of data interactively, often a more formal fit criteria is desired. This can be achieved by automatically refining the existing knot set locally by minimizing the least squares error. Given the function $S(x,\Delta)$ and the data set $\{(x_\ell,y_\ell) : \ell = 1, 2, \ldots, n\}$ the solution of the variable knot problem becomes the minimization of the least squares error over the knot set Δ . That is, find

$$\min_{\Delta} \min_{\{a_i\}} \sum_{\ell=1}^{n} [y_\ell - \sum_{i=-m}^{k} a_i s_i(x_\ell,\Delta)]^2 .$$

It is not too crucial to obtain the global minimum in the interactive case as a reasonable estimate of the knot set already exists. The purpose of optimization is merely to refine this estimate.

The minimization method chosen is known as COMPLEX which essentially involves reflecting the function around its centroid. The details are not discussed here but are adequately described in Box [2]. Reasons for choosing COMPLEX involve the fact that it does not require derivatives and that it will converge fairly rapidly towards the minimum.

COMPLEX contains one additional feature. This is that constraints can be imposed on the function. These constraints can be either <u>explicit</u> - meaning that the independent variable can be bounded by some function or constant; or <u>implicit</u> - meaning that the functional value can be bounded by some function or constant.

In approximation using spline functions it is only necessary to have explicit constraints to prevent the knots from coalescing. These constraints involve keeping the knots separated by a certain distance. How this distance is determined is partially dependent on the machine precision and hence the matrix used for solving the fixed knot problem. Because of machine precision the knots must be separated by a distance as least as great as the machine accuracy. Otherwise the least squares matrix will be singular.

More important, however, is the condition number of the matrix used in the fixed knot problem. As two knots converge towards each other, the two corresponding rows of the least squares matrix become more linearly dependent causing the condition number to rise. Therefore adequate constraints must be put on the knots to prevent them from causing numerical instabilities. For these reasons, the following constraint was placed on each knot:

set $h = \delta_{k+1} - \delta_0$ and constrain each knot δ_i by

$$\delta_{i-1} + .0001 \cdot h \leq \delta_i \leq \delta_{i+1} - .0001 \cdot h \,.$$

for $i = 1, 2, \ldots, k$.

NUMERICAL RESULTS

5.1 Introduction

There are three main areas where least squares approximation can be used. These are: the approximation of mathematical functions; the approximation of experimental data; and computer-aided design. In the first of these splines do not provide sufficient accuracy to warrant their use as functional approximations. In the second area splines give good results. In the third area splines are able to fit the contours of a design extremely well because of their piecewise nature.

In order to demonstrate the possibilities of interactive spline approximation three examples are given. The first example fits cubic splines to an interesting set of data mainly to demonstrate the possibilities of the method. The second example fits a cubic spline curve to a data set given in de Boor and Rice [5], [6] involving data from a Titanium heat experiment. The final example is the approximation of the outline of a Volkswagen. This result is merely intended to demonstrate the possibilities of splines in computer-aided design rather than having any practical importance.

All examples were run on an Adage Graphics Terminal connected to an IBM 360/67 duplex operating under MTS (Michigan Terminal System) located at the University of British Columbia. Hardcopy plots were obtained on a Calcomp plotter. Program listings and a user's guide are presented in Appendices A and B respectively.

The interactive system produces the following output:

1. Questions regarding what option the user would like are printed on a conversational terminal. An example of one such session with the interactive system is given in Appendix B.
2. A graph of the data points, knots and fitted curve are produced on the graphics terminal. This is identical to the hardcopy that can be produced from it as, for example, that of Figure 3.
3. A hardcopy plot of the graph can be produced upon request. This plot is identified by a title specified as input and a run number 'n' which indicates that it is the n-th hardcopy of the current terminal session.
4. A hardcopy printout as given in Table I which corresponds to the plot. It can be matched to the plot by the title and run number. The hardcopy printout contains the following information:

   the abscissae and ordinates of the data points (the original input to the system);

   the fitted ordinates (the approximation to the ordinates by the system);

   the residuals (the difference between the ordinates and the fitted ordinates);

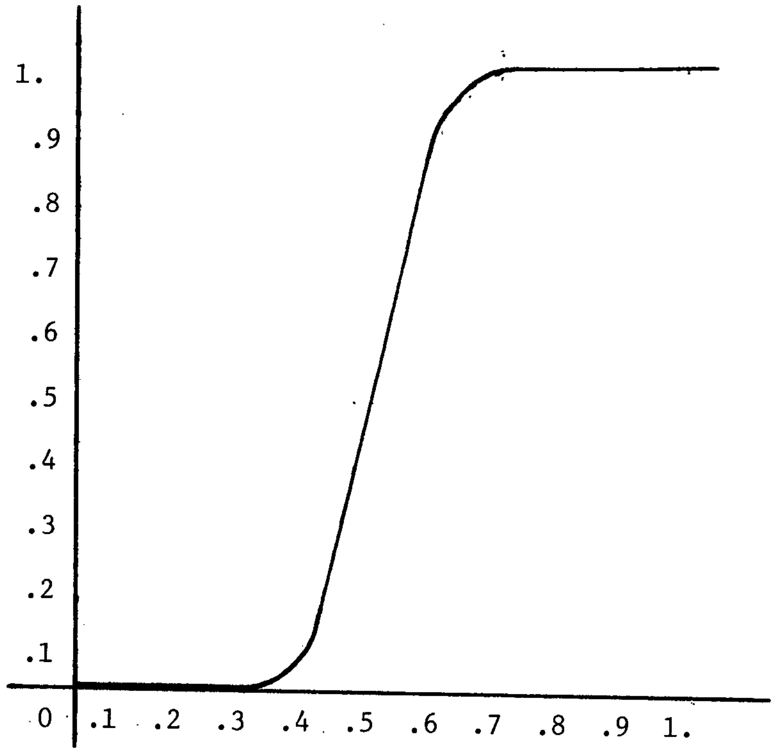
   the least squares error (the square root of the sum of the squares of the residuals);

   the location of the knots $\delta_0, \delta_1, \ldots, \delta_{k+1}$ and the coefficients of the piecewise continuous polynomial $\{s_i(x) : i = 0, 1, \ldots, k\}$ between each knot. That is, between knot pair $[\delta_i, \delta_{i+1}]$ the polynomial coefficient is the value for c_{ij} in

$$s_i(x) = \sum_{j=0}^{m} c_{ij}(x - \delta_i)^j .$$

5.2 Use of the System

To demonstrate the use of the interactive system a simple set of 11 data points was chosen. As can be seen from the plot in Figure 3 (ignoring the fitted curve for the moment) the eye tends to approximate the data with the following curve:



One would like to manipulate splines so that they also approximate this data with the above curve.

The first attempt was made to approximate the data with two uniformly spaced internal knots over [0,1] . As can be seen from the plot in Figure 3 and least squares error of .1104, the result was not suitable.

A second attempt was made by moving the two internal knots further apart to .25 and .75 respectively. This resulted in the approximation given in Figure 5 and Table I. The results are somewhat worse than the previous approximation (the least squares error was .1574 as compared to .1104). Consequently this approximation was eliminated and the previous uniformly spaced knot set retained.

Further attempts were made to produce a better approximation by moving the two knots closer together. When it became apparent that the least squares error had been reduced adequately with the knots located at .4 and .6 respectively (plot and results not given), these values were given to the minimization procedure to find the optimal knot locations. The results are given in Figure 4 and Table II. The optimization was terminated by the constraints on the knots. However, the least squares error was reduced significantly (the final error was .0544). Presumably this is the best approximation possible with two knots.

The next step in the procedure would then be to increase the number of knots to three. This was done and the initial results of three uniformly spaced internal knots are shown in Figure 5 and Table II. The interesting thing to note is that the least squares error and approximation are <u>identical</u> to that in Table I. This is because the third knot located at the point of symmetry is inert and does not contribute to the approximation at all. Consequently the piecewise polynomial between .5 and .75 is merely the polynomial between .25 and .75 shifted.

Since it was useless to continue with three knots the number was increased to four. The initial approximation with four uniform internal knots gave the result in Figure 6. This result is better than the optimized result with 2 knots (least squares error of .0247 as compared with .0544). But the curve in the end regions, although smaller, contains more oscillations.

The next step is to vary the knots interactively. First, adjustment of the first and last knots towards the boundaries produced significantly better results which terminated around .25 and .75 . Next, adjustment of the second and third knots produced better results continuously. The movement of the two knots was finally terminated at .49999 and .50001 as it was felt that they were coalescing too much (although the condition number of the least squares matrix was still only 5.96 and remaining fairly constant). The results of this approximation are spectacular. As can be seen from Table VI and Figure 7 the least squares error was .0000043 and the plot resembles the one expected.

It is interesting to note the similarity of the knot locations in Figures 5 and 7. Although Figure 7 represents two nearly equal knots at the center the difference in the approximations obtained indicates that it is not necessarily a good strategy to replace two coalescing knots by one knot and reducing the order of the system by one.

This example demonstrates the power of the interactive system to approximate data. In particular, the results of the interactive procedure on the four knot approximation produced such impressive results that automatic refinement was unnecessary.

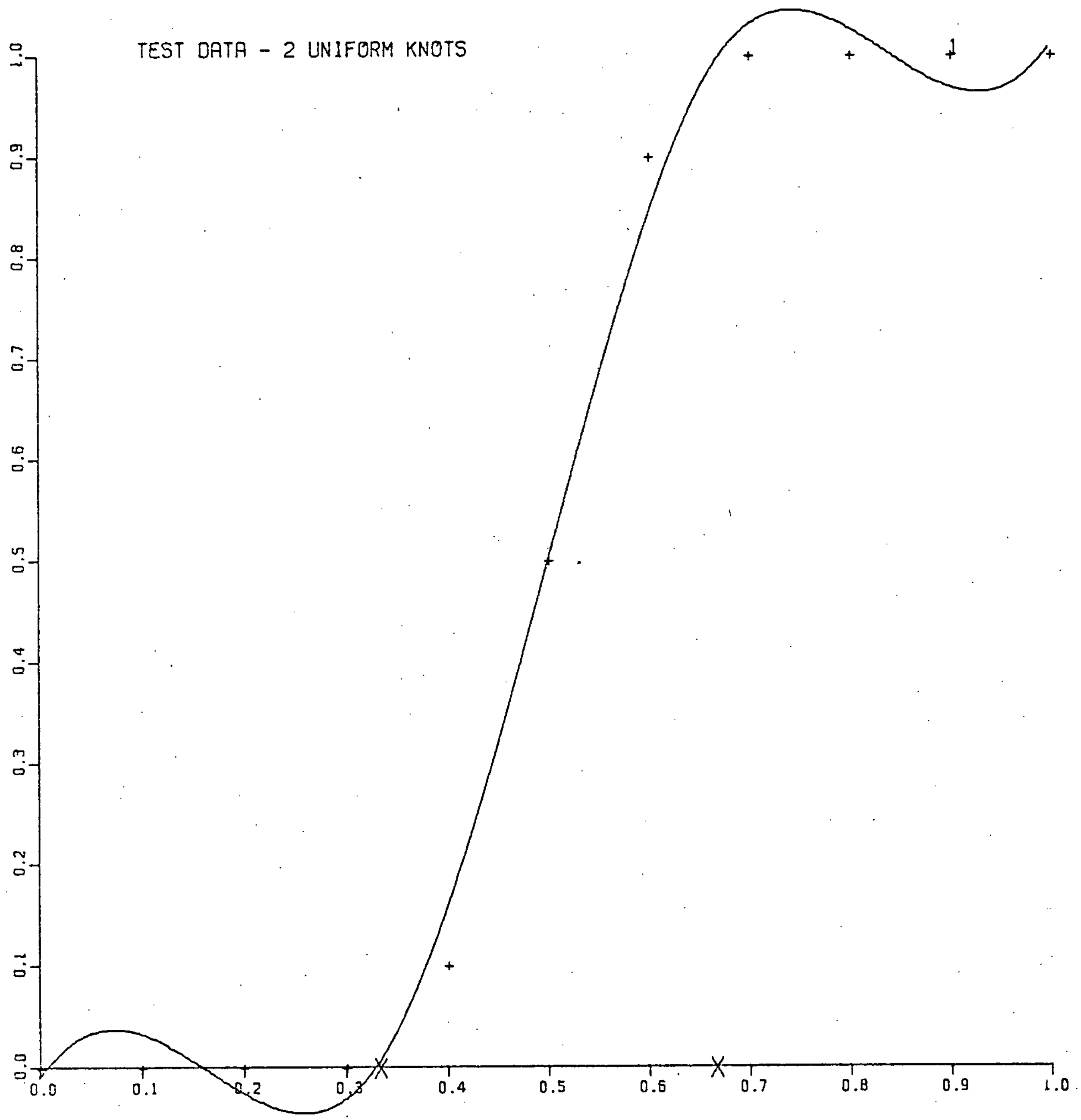


FIGURE 3

TEST DATA - 2 NON-UNIFORM KNOTS 1

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
0.0	0.0	-7.871840E-03	7.871840E-03
1.000000E-01	0.0	3.829566E-02	-3.829566E-02
2.000000E-01	0.0	-4.988915E-02	4.988915E-02
3.000000E-01	0.0	-2.595019E-02	2.595019E-02
4.000000E-01	1.000000E-01	1.877502E-01	-8.775020E-02
5.000000E-01	5.000000E-01	5.000004E-01	-4.619360E-07
6.000000E-01	9.000000E-01	8.122501E-01	8.774978E-02
7.000000E-01	1.000000E 00	1.025949E 00	-2.594873E-02
8.000000E-01	1.000000E 00	1.049888E 00	-4.988807E-02
9.000000E-01	1.000000E 00	9.617022E-01	3.829774E-02
1.000000E 00	1.000000E 00	1.007872E 00	-7.872522E-03

THE LEAST SQUARES ERROR IS 1.574225E-01

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	0.0		
		0	-7.871840E-03
		1	1.979496E 00
		2	-1.940852E 01
		3	4.230307E 01
1	2.500000E-01		
		0	-6.504732E-02
		1	2.070669E-01
		2	1.231876E 01
		3	-1.642503E 01
2	7.500000E-01		
		0	1.065044E 00
		1	2.070355E-01
		2	-1.231879E 01
		3	4.230357E 01
3	1.000000E 00		

TEST DATA - 2 NON-UNIFORM KNOTS

Table I

TEST DATA - 2 KNOTS OPTIMIZED 2

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
0.0	0.0	-7.379014E-03	7.379014E-03
1.000000E-01	0.0	2.207708E-02	-2.207708E-02
2.000000E-01	0.0	-1.452189E-02	1.452189E-02
3.000000E-01	0.0	-1.511128E-02	1.511128E-02
4.000000E-01	1.000000E-01	1.223733E-01	-2.237328E-02
5.000000E-01	5.000000E-01	4.999992E-01	8.030709E-07
6.000000E-01	9.000000E-01	8.776275E-01	2.237248E-02
7.000000E-01	1.000000E 00	1.015110E 00	-1.511062E-02
8.000000E-01	1.000000E 00	1.014520E 00	-1.452056E-02
9.000000E-01	1.000000E 00	9.779216E-01	2.207839E-02
1.000000E 00	1.000000E 00	1.007378E 00	-7.378042E-03

THE LEAST SQUARES ERROR IS 5.443568E-02

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	0.0		
		0	-7.379014E-03
		1	9.650481E-01
		2	-8.405960E 00
		3	1.701073E 01
1	4.999000E-01		
		0	4.994677E-01
		1	5.313705E 00
		2	1.710503E 01
		3	-5.700724E 04
2	5.001000E-01		
		0	5.005385E-01
		1	5.313683E 00
		2	-1.710493E 01
		3	1.701070E 01
3	1.000000E 00		

TEST DATA - 2 KNOTS OPTIMIZED

Table II

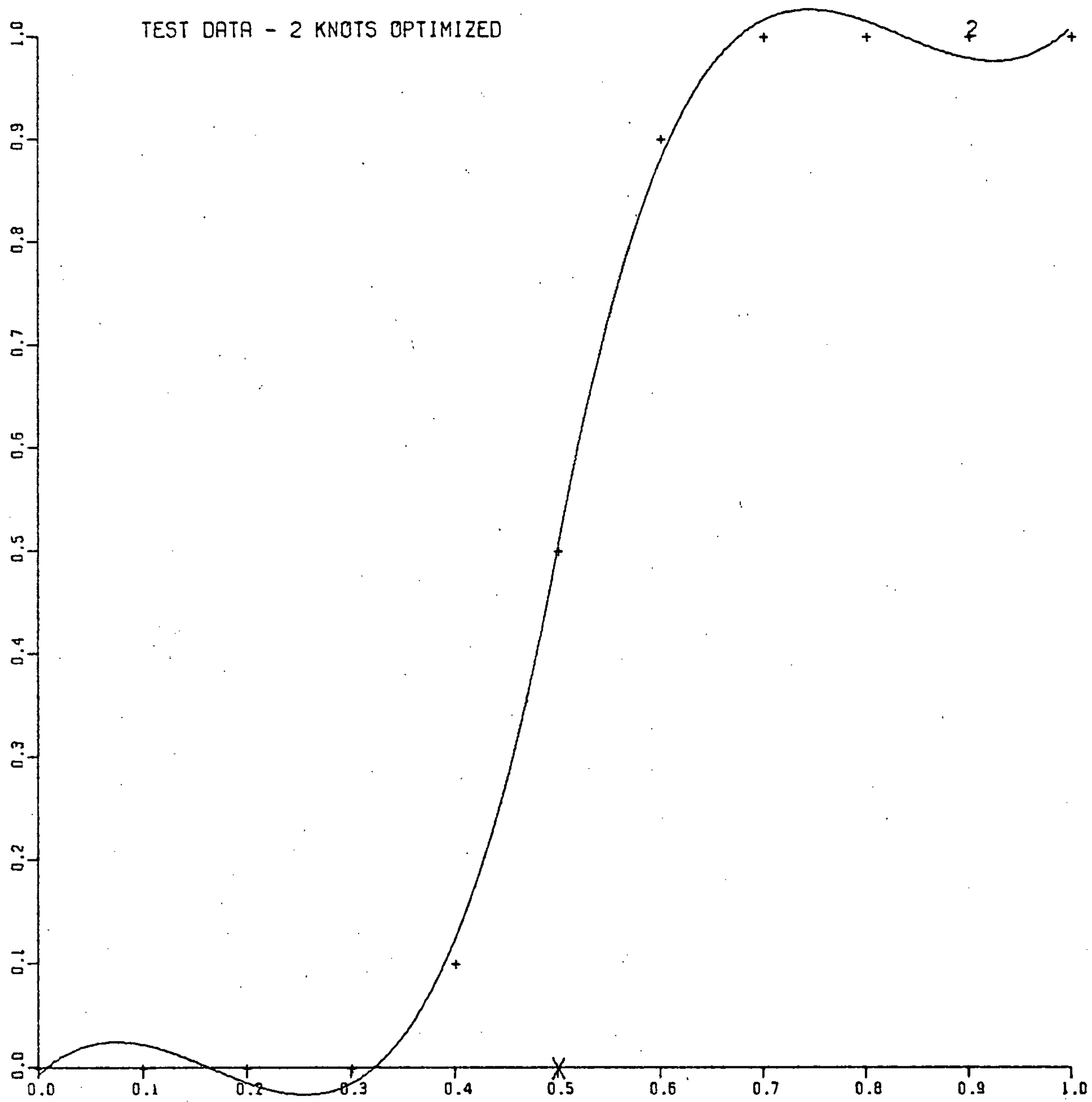

TEST DATA - 2 KNOTS OPTIMIZED
2
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

FIGURE 4

TEST DATA - 3 UNIFORM KNOTS 2

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
0.0	0.0	-7.872656E-03	7.872656E-03
1.000000E-01	0.0	3.829633E-02	-3.829633E-02
2.000000E-01	0.0	-4.988962E-02	4.988962E-02
3.000000E-01	0.0	-2.595067E-02	2.595067E-02
4.000000E-01	1.000000E-01	1.877483E-01	-8.774823E-02
5.000000E-01	5.000000E-01	4.999984E-01	1.564622E-06
6.000000E-01	9.000000E-01	8.122493E-01	8.775061E-02
7.000000E-01	1.000000E 00	1.025949E 00	-2.594928E-02
8.000000E-01	1.000000E 00	1.049888E 00	-4.988800E-02
9.000000E-01	1.000000E 00	9.617013E-01	3.829866E-02
1.000000E 00	1.000000E 00	1.007872E 00	-7.871866E-03

THE LEAST SQUARES ERROR IS 1.574225E-01

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	0.0		
		0	-7.872656E-03
		1	1.979532E 00
		2	-1.940877E 01
		3	4.230345E 01
1	2.500000E-01		
		0	-6.504667E-02
		1	2.070408E-01
		2	1.231880E 01
		3	-1.642502E 01
2	5.000000E-01		
		0	4.999984E-01
		1	3.286754E 00
		2	5.340576E-05
		3	-1.642529E 01
3	7.500000E-01		
		0	1.065044E 00
		1	2.070377E-01
		2	-1.231889E 01
		3	4.230394E 01
4	1.000000E 00		

TEST DATA - 3 UNIFORM KNOTS

Table III

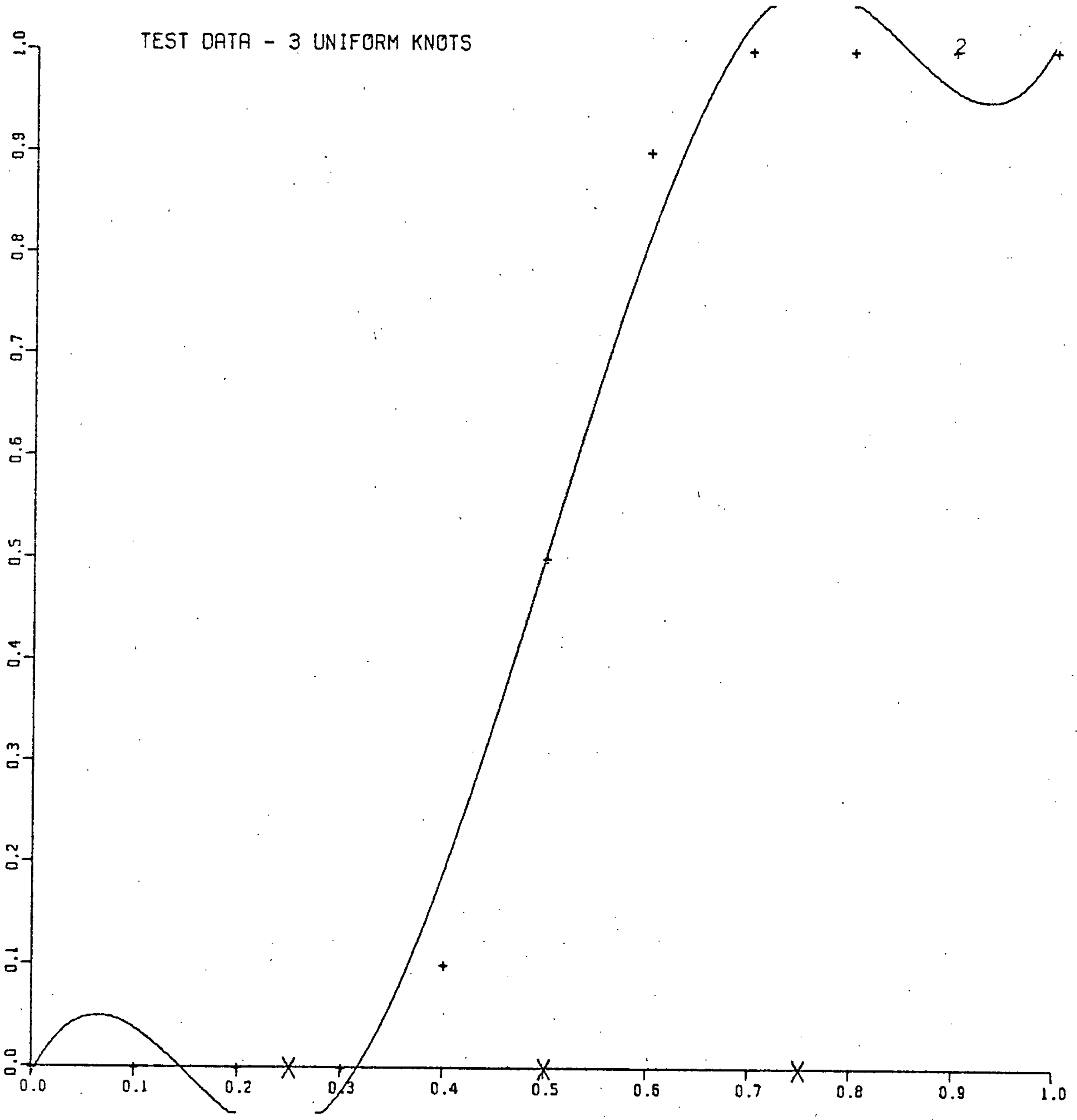


FIGURE 5

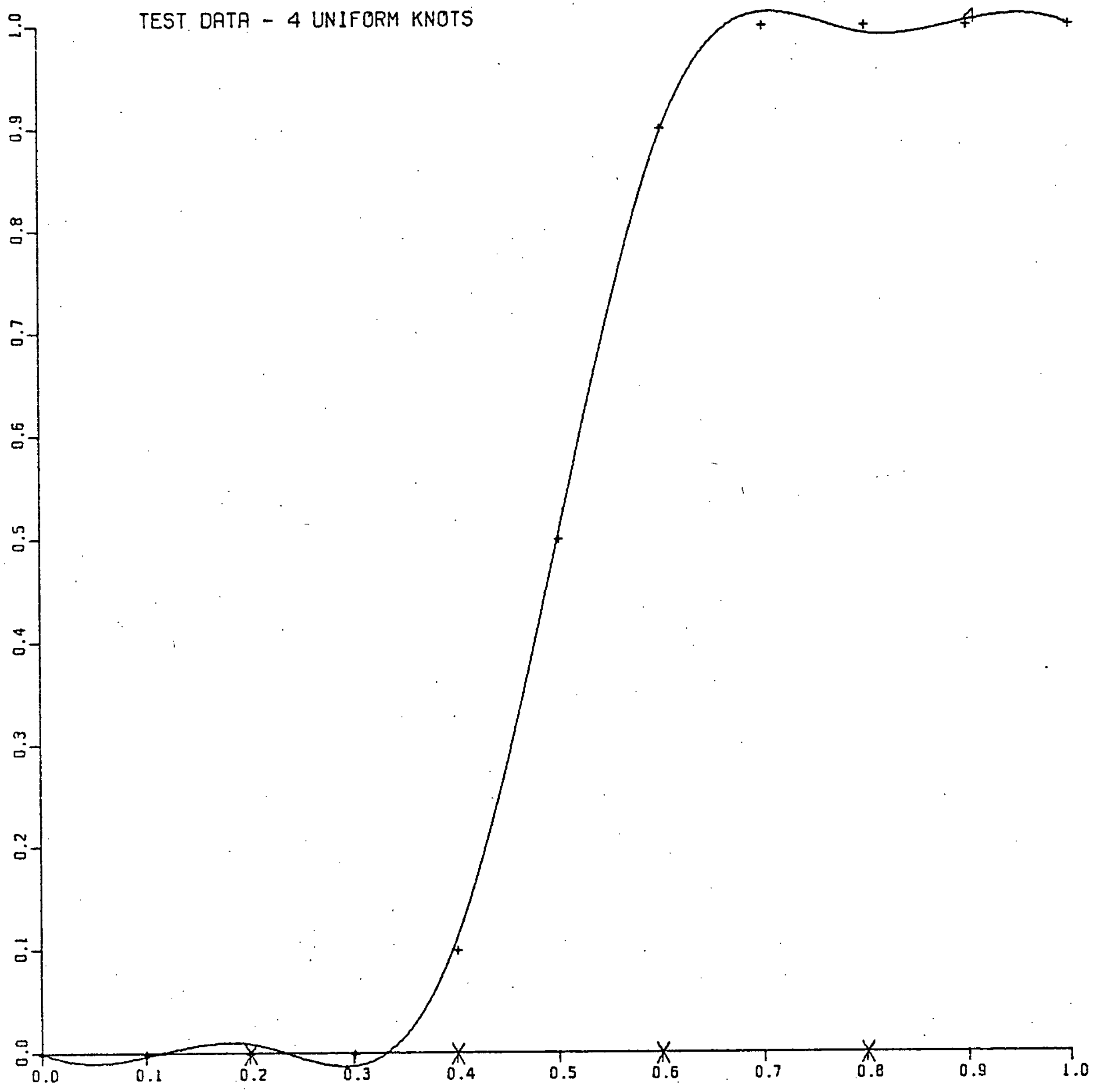


FIGURE 6

TEST DATA - 4 NON-UNIFORM KNOTS 1

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
0.0	0.0	3.961031E-08	-3.961031E-08
1.000000E-01	0.0	-1.974404E-07	1.974404E-07
2.000000E-01	0.0	2.697052E-07	-2.697052E-07
3.000000E-01	0.0	2.083834E-07	-2.083834E-07
4.000000E-01	1.000000E-01	9.999895E-02	1.032065E-06
5.000000E-01	5.000000E-01	5.000019E-01	-1.941880E-06
6.000000E-01	9.000000E-01	8.999965E-01	3.502471E-06
7.000000E-01	1.000000E 00	1.000000E 00	-4.593458E-07
8.000000E-01	1.000000E 00	9.999991E-01	8.458737E-07
9.000000E-01	1.000000E 00	9.999999E-01	7.450581E-08
1.000000E 00	1.000000E 00	9.999999E-01	1.192093E-07

THE LEAST SQUARES ERROR IS 4.266889E-06

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	0.0		
		0	3.961031E-08
		1	-1.450448E-02
		2	2.175139E-01
		3	-7.249289E-01
1	2.500000E-01		
		0	-1.357395E-03
		1	-4.168792E-02
		2	-3.261279E-01
		3	3.405853E 01
2	4.999900E-01		
		0	4.999400E-01
		1	6.180718E 00
		2	2.521675E 01
		3	-8.394170E 05
3	5.000100E-01		
		0	5.000489E-01
		1	6.180779E 00
		2	-2.521672E 01
		3	3.405827E 01
4	7.500000E-01		
		0	1.001356E 00
		1	-4.164546E-02
		2	3.259857E-01
		3	-7.245331E-01
5	1.000000E 00		

TEST DATA - 4 NON-UNIFORM KNOTS

Table IV

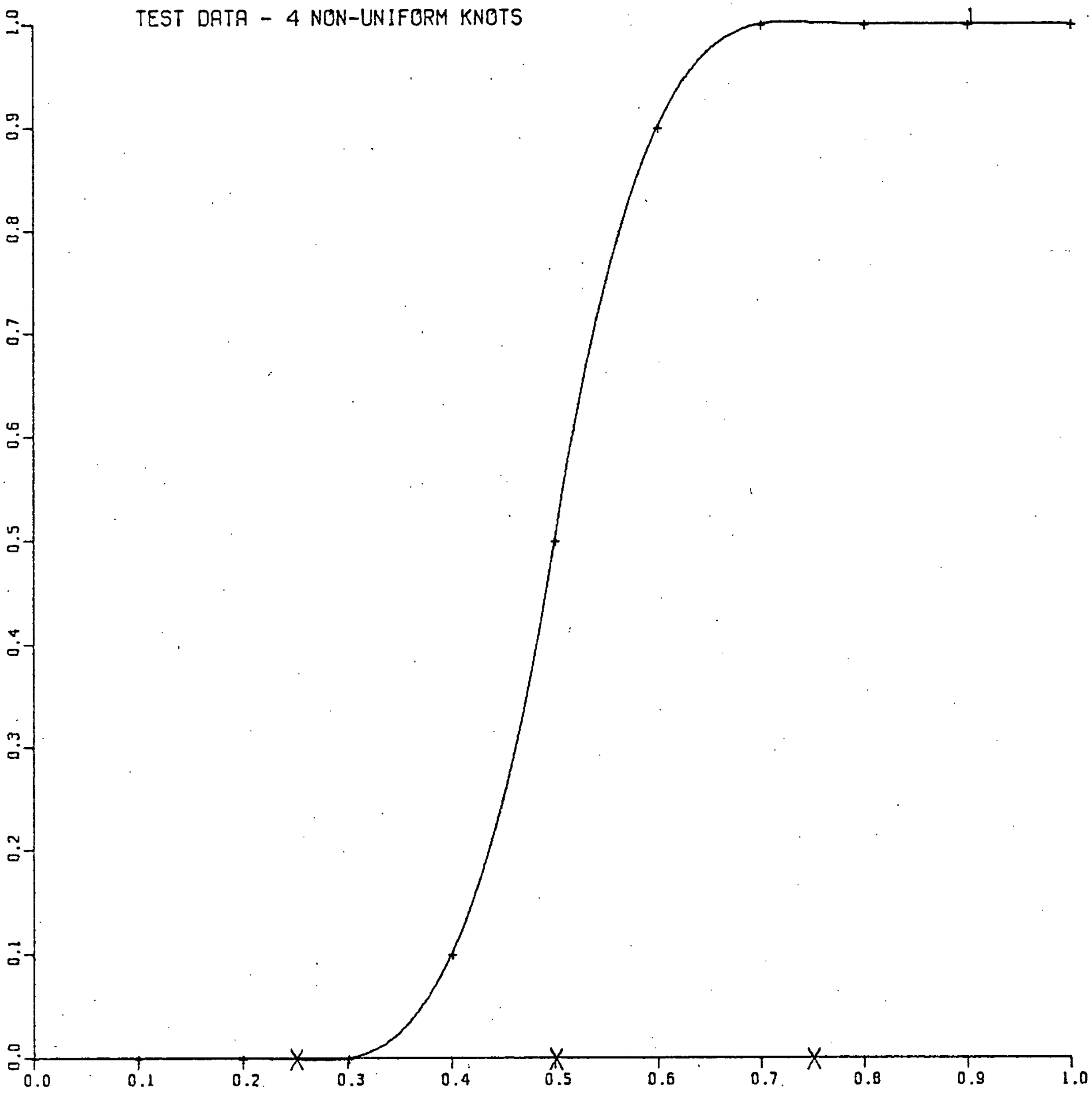


FIGURE 7

5.3 Titanium Heat Data

de Boor and Rice [5], [6] presented a set of data obtained from a heat experiment involving Titanium. This data was used to demonstrate the fixed knot program using a knot set with five equally spaced knots [5, p. 18] and the variable knot program which involved the optimization of this fixed knot set. Further studies involved the optimization of a non-uniform knot set containing five knots.

One of the best ways to test a system is to try it on previously done results and check the answers. Initially the interactive system was done with the uniform knot set specified in de Boor and Rice [5, p. 18]. The results and plot are given in Table V and Figure 8. These results compare favorable with those in de Boor and Rice (the residual was 1.16 as compared to their 1.24). However, as can be seen from the plot, the resulting approximation is none too satisfactory.

It is here that the power of the interactive system comes into effect. After moving the knots so that the plot produced is more accurate, better results are obtained. The results given in Table VI and Figure 9 indicate substantial improvement. The resulting knot set was used in an initial guess for knot optimization giving final result shown in Table VII and Figure 10. The reduction in the least squares error was substantial (.092 as compared to 1.16) largely because the interactive knot placement allowed for the deriving of accurate starting values.

TITANIUM HEAT DATA - 5 UNIFORM KNOTS 1

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
5.950000E 02	6.440000E-01	6.235025E-01	2.049747E-02
6.050000E 02	6.220000E-01	6.425120E-01	-2.051199E-02
6.150000E 02	6.380000E-01	6.516960E-01	-1.369604E-02
6.250000E 02	6.490000E-01	6.534591E-01	-4.459172E-03
6.350000E 02	6.520000E-01	6.502057E-01	1.794243E-03
6.450000E 02	6.390000E-01	6.443403E-01	-5.340301E-03
6.550000E 02	6.460000E-01	6.382672E-01	7.732764E-03
6.650000E 02	6.570000E-01	6.343911E-01	2.260887E-02
6.750000E 02	6.520000E-01	6.351163E-01	1.688367E-02
6.850000E 02	6.550000E-01	6.420718E-01	1.292809E-02
6.950000E 02	6.640000E-01	6.537848E-01	1.021518E-02
7.050000E 02	6.630000E-01	6.680065E-01	-5.006507E-03
7.150000E 02	6.630000E-01	6.824884E-01	-1.948840E-02
7.250000E 02	6.680000E-01	6.949821E-01	-2.698214E-02
7.350000E 02	6.760000E-01	7.032389E-01	-2.723893E-02
7.450000E 02	6.760000E-01	7.050105E-01	-2.901048E-02
7.550000E 02	6.860000E-01	6.980482E-01	-1.204824E-02
7.650000E 02	6.790000E-01	6.815894E-01	-2.589412E-03
7.750000E 02	6.780000E-01	6.608155E-01	1.718443E-02
7.850000E 02	6.830000E-01	6.423933E-01	4.060671E-02
7.950000E 02	6.940000E-01	6.329899E-01	6.101005E-02
8.050000E 02	6.990000E-01	6.392726E-01	5.972740E-02
8.150000E 02	7.100000E-01	6.679080E-01	4.209192E-02
8.250000E 02	7.300000E-01	7.255636E-01	4.436404E-03
8.350000E 02	7.630000E-01	8.189063E-01	-5.590630E-02
8.450000E 02	8.120000E-01	9.505556E-01	-1.385556E-01
8.550000E 02	9.070000E-01	1.106939E 00	-1.999393E-01
8.650000E 02	1.044000E 00	1.270436E 00	-2.264375E-01
8.750000E 02	1.336000E 00	1.423430E 00	-8.743048E-02
8.850000E 02	1.881000E 00	1.548299E 00	3.326998E-01
8.950000E 02	2.169000E 00	1.627424E 00	5.415753E-01
9.050000E 02	2.075000E 00	1.643184E 00	4.318160E-01
9.150000E 02	1.598000E 00	1.583930E 00	1.406908E-02
9.250000E 02	1.211000E 00	1.461898E 00	-2.508975E-01
9.350000E 02	9.160000E-01	1.295290E 00	-3.792901E-01
9.450000E 02	7.460000E-01	1.102311E 00	-3.563114E-01
9.550000E 02	6.720000E-01	9.011649E-01	-2.291650E-01
9.650000E 02	6.270000E-01	7.100576E-01	-8.305758E-02
9.750000E 02	6.150000E-01	5.471910E-01	6.780899E-02
9.850000E 02	6.070000E-01	4.307705E-01	1.762294E-01
9.950000E 02	6.060000E-01	3.790006E-01	2.269993E-01
1.005000E 03	6.090000E-01	4.022449E-01	2.067550E-01
1.015000E 03	6.030000E-01	4.795058E-01	1.234941E-01
1.025000E 03	6.010000E-01	5.819451E-01	1.905493E-02
1.035000E 03	6.030000E-01	6.807239E-01	-7.772392E-02
1.045000E 03	6.010000E-01	7.470052E-01	-1.460052E-01
1.055000E 03	6.110000E-01	7.519506E-01	-1.409506E-01
1.065000E 03	6.010000E-01	6.667216E-01	-6.572163E-02
1.075000E 03	6.080000E-01	4.624799E-01	1.455200E-01

THE LEAST SQUARES ERROR IS 1.157334E 00

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	5.950000E 02		

		0	6.235024E-01
		1	2.472371E-03
		2	-6.114945E-05
		3	4.007443E-07
1	6.750000E 02		
		0	6.351163E-01
		1	3.827438E-04
		2	3.502920E-05
		3	-3.747538E-07
2	7.550000E 02		
		0	6.980482E-01
		1	-1.207871E-03
		2	-5.491161E-05
		3	1.111178E-06
3	8.350000E 02		
		0	8.189063E-01
		1	1.134087E-02
		2	2.117710E-04
		3	-2.936617E-06
4	9.050000E 02		
		0	1.643183E 00
		1	-2.179519E-03
		2	-4.049190E-04
		3	3.034045E-06
5	9.950000E 02		
		0	3.790006E-01
		1	-1.337662E-03
		2	4.142732E-04
		3	-4.806359E-06
6	1.075000E 03		

TITANIUM HEAT DATA - 5 UNIFORM KNOTS

Table V

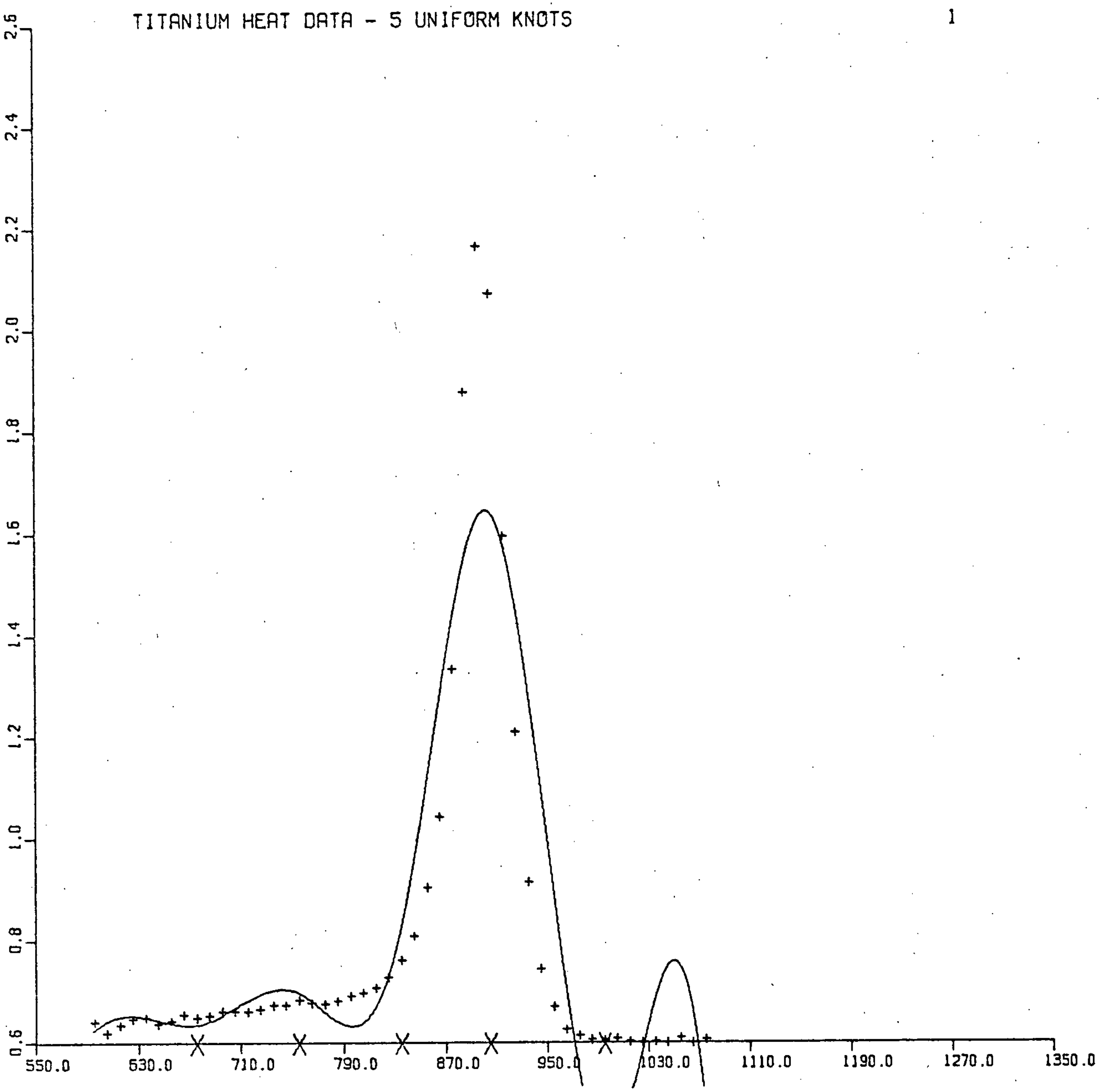

TITANIUM HEAT DATA - 5 UNIFORM KNOTS
1
2.6
2.4
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
550.0
630.0
710.0
790.0
870.0
950.0
1030.0
1110.0
1190.0
1270.0
1350.0

FIGURE 8

TITANIUM HEAT DATA - 5 NON-UNIFORM KNOTS 1

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
5.950000E 02	6.440000E-01	6.252043E-01	1.879563E-02
6.050000E 02	6.220000E-01	6.338547E-01	-1.185482E-02
6.150000E 02	6.380000E-01	6.407921E-01	-2.792177E-03
6.250000E 02	6.490000E-01	6.462172E-01	2.782739E-03
6.350000E 02	6.520000E-01	6.503309E-01	1.669100E-03
6.450000E 02	6.390000E-01	6.533335E-01	-1.433358E-02
6.550000E 02	6.460000E-01	6.554263E-01	-9.426299E-03
6.650000E 02	6.570000E-01	6.568095E-01	1.904927E-04
6.750000E 02	6.520000E-01	6.576843E-01	-5.684260E-03
6.850000E 02	6.550000E-01	6.582511E-01	-3.251180E-03
6.950000E 02	6.640000E-01	6.587108E-01	5.289115E-03
7.050000E 02	6.630000E-01	6.592643E-01	3.735680E-03
7.150000E 02	6.630000E-01	6.601120E-01	2.888002E-03
7.250000E 02	6.680000E-01	6.614549E-01	6.545052E-03
7.350000E 02	6.760000E-01	6.634936E-01	1.250640E-02
7.450000E 02	6.760000E-01	6.664289E-01	9.571020E-03
7.550000E 02	6.860000E-01	6.704616E-01	1.553836E-02
7.650000E 02	6.790000E-01	6.757923E-01	3.207672E-03
7.750000E 02	6.780000E-01	6.826219E-01	-4.621979E-03
7.850000E 02	6.830000E-01	6.911511E-01	-8.151092E-03
7.950000E 02	6.940000E-01	7.015807E-01	-7.580712E-03
8.050000E 02	6.990000E-01	7.141112E-01	-1.511123E-02
8.150000E 02	7.100000E-01	7.289435E-01	-1.894355E-02
8.250000E 02	7.300000E-01	7.462784E-01	-1.627839E-02
8.350000E 02	7.630000E-01	7.663165E-01	-3.316541E-03
8.450000E 02	8.120000E-01	7.908105E-01	2.118939E-02
8.550000E 02	9.070000E-01	8.572057E-01	4.979425E-02
8.650000E 02	1.044000E 00	1.038639E 00	5.359702E-03
8.750000E 02	1.336000E 00	1.402930E 00	-6.693059E-02
8.850000E 02	1.881000E 00	1.859838E 00	2.116160E-02
8.950000E 02	2.169000E 00	2.161066E 00	7.933423E-03
9.050000E 02	2.075000E 00	2.064366E 00	1.063279E-02
9.150000E 02	1.598000E 00	1.624727E 00	-2.672801E-02
9.250000E 02	1.211000E 00	1.185819E 00	2.518102E-02
9.350000E 02	9.160000E-01	9.078093E-01	8.190691E-03
9.450000E 02	7.460000E-01	7.544307E-01	-8.430697E-03
9.550000E 02	6.720000E-01	6.808758E-01	-8.875843E-03
9.650000E 02	6.270000E-01	6.432318E-01	-1.623188E-02
9.750000E 02	6.150000E-01	6.181573E-01	-3.157331E-03
9.850000E 02	6.070000E-01	6.028832E-01	4.116789E-03
9.950000E 02	6.060000E-01	5.955343E-01	1.046569E-02
1.005000E 03	6.090000E-01	5.942356E-01	1.476442E-02
1.015000E 03	6.030000E-01	5.971119E-01	5.888034E-03
1.025000E 03	6.010000E-01	6.022885E-01	-1.288563E-03
1.035000E 03	6.030000E-01	6.078902E-01	-4.890207E-03
1.045000E 03	6.010000E-01	6.120420E-01	-1.104198E-02
1.055000E 03	6.110000E-01	6.128688E-01	-1.868859E-03
1.065000E 03	6.010000E-01	6.084959E-01	-7.495925E-03
1.075000E 03	6.080000E-01	5.970479E-01	1.095206E-02

THE LEAST SQUARES ERROR IS 1.142650E-01

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	5.950000E 02		

		0	6.252043E-01
		1	9.573922E-04
		2	-9.568968E-06
		3	3.345709E-08
1	8.400000E 02		
		0	7.774121E-01
		1	2.293389E-03
		2	1.502197E-05
		3	1.244829E-05
2	8.700000E 02		
		0	1.195837E 00
		1	3.680510E-02
		2	1.135369E-03
		3	-4.252815E-05
3	9.000000E 02		
		0	2.173558E 00
		1	-9.898979E-03
		2	-2.692169E-03
		3	6.085630E-05
4	9.200000E 02		
		0	1.385562E 00
		1	-4.455817E-02
		2	9.592120E-04
		3	-7.467897E-06
5	9.600000E 02		
		0	6.600300E-01
		1	-3.667146E-03
		2	6.306361E-05
		3	-3.125027E-07
6	1.075000E 03		

TITANIUM HEAT DATA - 5 NON-UNIFORM KNOTS

Table VI

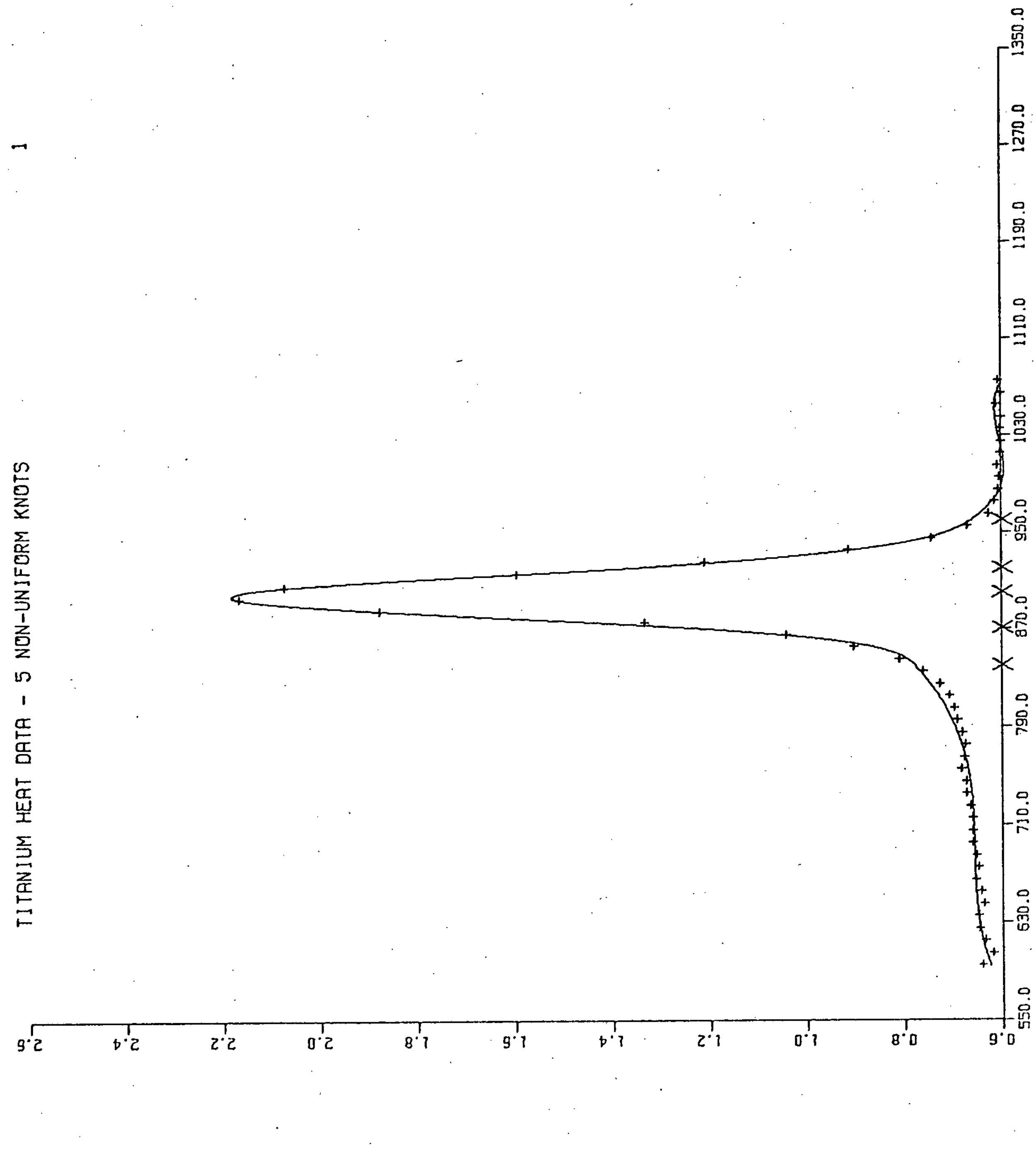

TITANIUM HEAT DATA - 5 NON-UNIFORM KNOTS
1
550.0
630.0
710.0
790.0
870.0
950.0
1030.0
1110.0
1190.0
1270.0
1350.0
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6

FIGURE 9

TITANIUM HEAT DATA - 5 KNOTS OPTIMIZED 2

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
5.950000E 02	6.440000E-01	6.225190E-01	2.148103E-02
6.050000E 02	6.220000E-01	6.327376E-01	-1.073763E-02
6.150000E 02	6.380000E-01	6.408629E-01	-2.862922E-03
6.250000E 02	6.490000E-01	6.471329E-01	1.867094E-03
6.350000E 02	6.520000E-01	6.517856E-01	2.144140E-04
6.450000E 02	6.390000E-01	6.550590E-01	-1.605903E-02
6.550000E 02	6.460000E-01	6.571912E-01	-1.119116E-02
6.650000E 02	6.570000E-01	6.584202E-01	-1.420241E-03
6.750000E 02	6.520000E-01	6.589841E-01	-6.984055E-03
6.850000E 02	6.550000E-01	6.591210E-01	-4.121054E-03
6.950000E 02	6.640000E-01	6.590686E-01	4.931286E-03
7.050000E 02	6.630000E-01	6.590654E-01	3.934622E-03
7.150000E 02	6.630000E-01	6.593490E-01	3.650941E-03
7.250000E 02	6.680000E-01	6.601578E-01	7.842168E-03
7.350000E 02	6.760000E-01	6.617296E-01	1.427035E-02
7.450000E 02	6.760000E-01	6.643026E-01	1.169730E-02
7.550000E 02	6.860000E-01	6.681147E-01	1.788527E-02
7.650000E 02	6.790000E-01	6.734042E-01	5.595822E-03
7.750000E 02	6.780000E-01	6.804088E-01	-2.408907E-03
7.850000E 02	6.830000E-01	6.893667E-01	-6.366715E-03
7.950000E 02	6.940000E-01	7.005159E-01	-6.515928E-03
8.050000E 02	6.990000E-01	7.140946E-01	-1.509461E-02
8.150000E 02	7.100000E-01	7.303407E-01	-2.034071E-02
8.250000E 02	7.300000E-01	7.494920E-01	-1.949206E-02
8.350000E 02	7.630000E-01	7.717870E-01	-8.786995E-03
8.450000E 02	8.120000E-01	7.992001E-01	1.279991E-02
8.550000E 02	9.070000E-01	8.665996E-01	4.040042E-02
8.650000E 02	1.044000E 00	1.038002E 00	5.996864E-03
8.750000E 02	1.336000E 00	1.378159E 00	-4.215827E-02
8.850000E 02	1.881000E 00	1.851600E 00	2.939950E-02
8.950000E 02	2.169000E 00	2.178109E 00	-9.109914E-03
9.050000E 02	2.075000E 00	2.067245E 00	7.754855E-03
9.150000E 02	1.598000E 00	1.616635E 00	-1.863577E-02
9.250000E 02	1.211000E 00	1.193341E 00	1.765861E-02
9.350000E 02	9.160000E-01	9.150418E-01	9.581815E-04
9.450000E 02	7.460000E-01	7.509134E-01	-4.913419E-03
9.550000E 02	6.720000E-01	6.683230E-01	3.676936E-03
9.650000E 02	6.270000E-01	6.346373E-01	-7.637367E-03
9.750000E 02	6.150000E-01	6.188952E-01	-3.895170E-03
9.850000E 02	6.070000E-01	6.086131E-01	-1.613097E-03
9.950000E 02	6.060000E-01	6.027005E-01	3.299463E-03
1.005000E 03	6.090000E-01	6.002449E-01	8.755121E-03
1.015000E 03	6.030000E-01	6.003335E-01	2.666444E-03
1.025000E 03	6.010000E-01	6.020537E-01	-1.053706E-03
1.035000E 03	6.030000E-01	6.044927E-01	-1.492694E-03
1.045000E 03	6.010000E-01	6.067377E-01	-5.737711E-03
1.055000E 03	6.110000E-01	6.078761E-01	3.123831E-03
1.065000E 03	6.010000E-01	6.069952E-01	-5.995244E-03
1.075000E 03	6.080000E-01	6.031824E-01	4.817545E-03

THE LEAST SQUARES ERROR IS 9.286332E-02

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	5.950000E 02		

		0	6.225189E-01
		1	1.134473E-03
		2	-1.165708E-05
		3	3.967317E-08
1	8.395486E 02		
		0	7.830166E-01
		1	2.551379E-03
		2	1.744409E-05
		3	1.084204E-05
2	8.733201E 02		
		0	1.306674E 00
		1	4.082611E-02
		2	1.115898E-03
		3	-5.281300E-05
3	8.989514E 02		
		0	2.196890E 00
		1	-6.059237E-03
		2	-2.945114E-03
		3	6.665658E-05
4	9.179270E 02		
		0	1.476898E 00
		1	-4.582613E-02
		2	8.494323E-04
		3	-5.438899E-06
5	9.681765E 02		
		0	6.288878E-01
		1	-1.658810E-03
		2	2.952565E-05
		3	-1.521178E-07
6	1.075000E 03		

TITANIUM HEAT DATA - 5 KNOTS OPTIMIZED

Table VIII

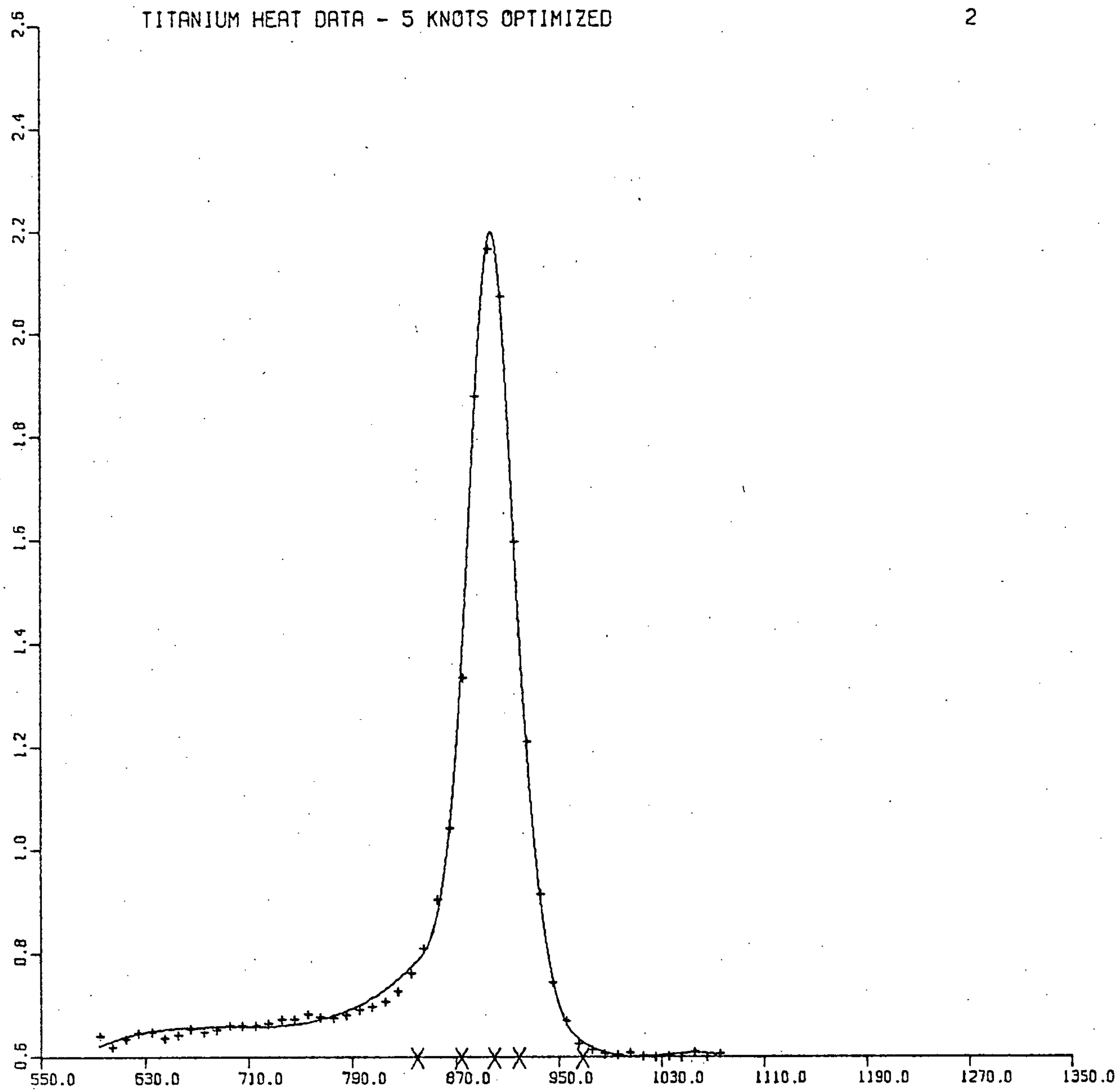

TITANIUM HEAT DATA - 5 KNOTS OPTIMIZED
2
2.6
2.4
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
550.0
630.0
710.0
790.0
870.0
950.0
1030.0
1110.0
1190.0
1270.0
1350.0

FIGURE 10

5.4 The Bug

The use of spline approximation in computer-aided design is a relatively unexplored area. The Volkswagen data presented in this section demonstrates possible applications of splines in the area. Also the approximation demonstrates the use of a large knot set.

The initial approximation of 18 equally spaced knots produced a Volkswagen with a dented hood as can be seen in Figure 11. Interactive manipulation of the knots produced the more reasonable resemblance to a Volkswagen shown in Figure 12 and Table VIII. Since the desired results were based on representing the data accurately by sight rather than minimizing the least squares error the knots were not optimized.

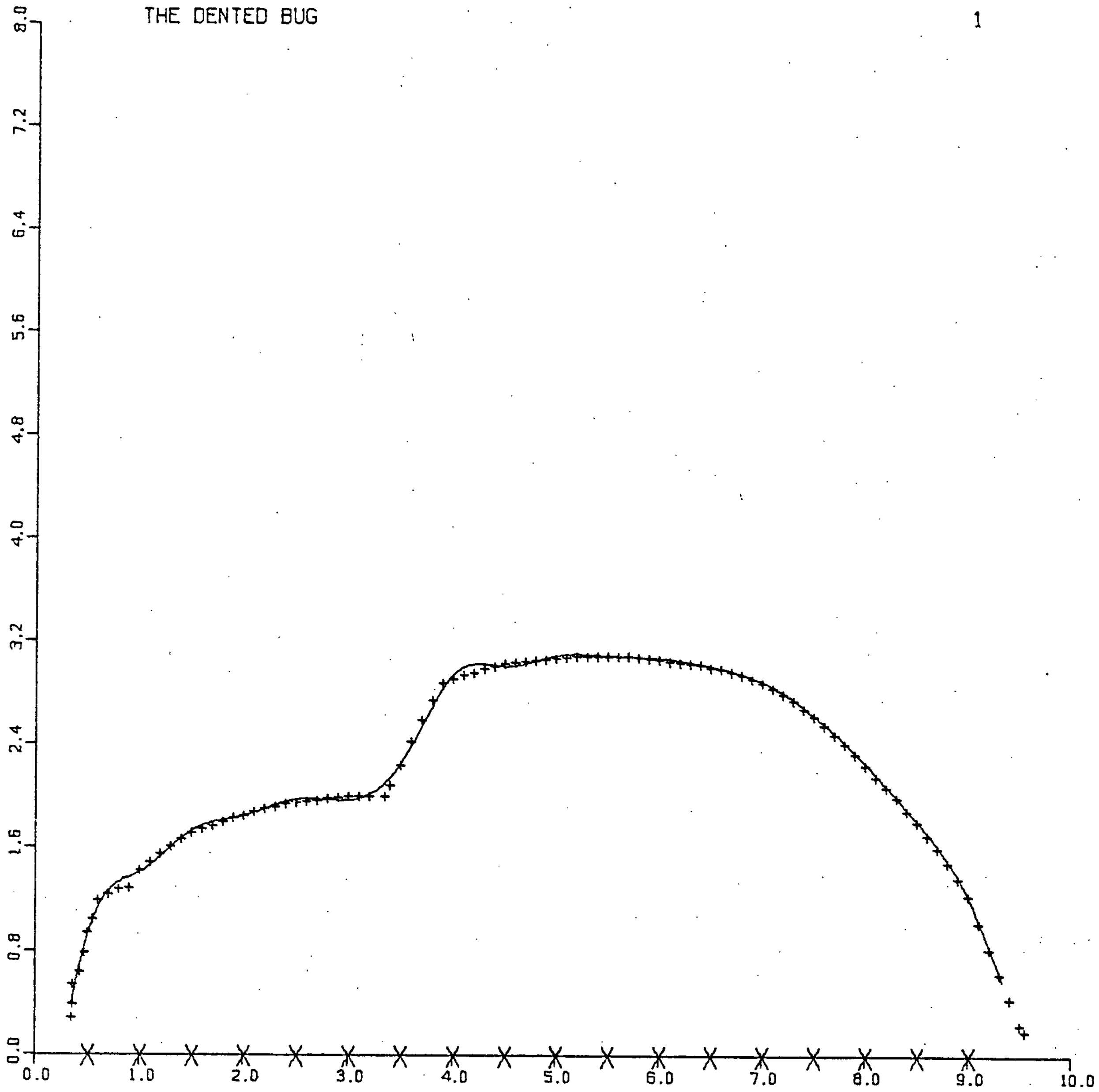

THE DENTED BUG
1
8.0
7.2
6.4
5.6
4.8
4.0
3.2
2.4
1.6
0.8
0.0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0

FIGURE 11

THE BUG 2

ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS
3.500000E-01	3.000000E-01	4.035096E-01	-1.035095E-01
3.525000E-01	4.000000E-01	4.113406E-01	-1.134065E-02
3.550000E-01	5.500000E-01	4.192352E-01	1.307648E-01
4.300000E-01	6.500000E-01	6.758505E-01	-2.585047E-02
4.700000E-01	8.000000E-01	8.169750E-01	-1.697499E-02
5.000000E-01	9.500000E-01	9.183491E-01	3.165080E-02
5.500000E-01	1.060000E 00	1.068680E 00	-8.680243E-03
6.000000E-01	1.200000E 00	1.182031E 00	1.796837E-02
7.000000E-01	1.250000E 00	1.275474E 00	-2.547405E-02
8.000000E-01	1.290000E 00	1.276578E 00	1.342188E-02
9.000000E-01	1.300000E 00	1.304118E 00	-4.118513E-03
1.000000E 00	1.430000E 00	1.421022E 00	8.977752E-03
1.100000E 00	1.500000E 00	1.509212E 00	-9.212483E-03
1.200000E 00	1.560000E 00	1.564281E 00	-4.281554E-03
1.300000E 00	1.620000E 00	1.616894E 00	3.105875E-03
1.400000E 00	1.675000E 00	1.668297E 00	6.702900E-03
1.500000E 00	1.720000E 00	1.713498E 00	6.501570E-03
1.600000E 00	1.750000E 00	1.752367E 00	-2.367944E-03
1.700000E 00	1.780000E 00	1.785924E 00	-5.924519E-03
1.800000E 00	1.810000E 00	1.815188E 00	-5.188584E-03
1.900000E 00	1.840000E 00	1.841183E 00	-1.182909E-03
2.000000E 00	1.860000E 00	1.864781E 00	-4.782557E-03
2.100000E 00	1.890000E 00	1.887129E 00	2.871351E-03
2.200000E 00	1.915000E 00	1.908055E 00	6.944049E-03
2.300000E 00	1.930000E 00	1.927378E 00	2.621699E-03
2.400000E 00	1.950000E 00	1.944908E 00	5.091667E-03
2.500000E 00	1.960000E 00	1.960458E 00	-4.580617E-04
2.600000E 00	1.970000E 00	1.973843E 00	-3.842760E-03
2.700000E 00	1.980000E 00	1.984876E 00	-4.876371E-03
2.800000E 00	1.990000E 00	1.993369E 00	-3.369596E-03
2.900000E 00	2.000000E 00	1.999138E 00	8.617891E-04
3.000000E 00	2.005000E 00	2.002012E 00	2.987819E-03
3.100000E 00	2.005000E 00	2.002145E 00	2.855293E-03
3.200000E 00	2.005000E 00	2.001301E 00	3.698572E-03
3.350000E 00	2.005000E 00	2.034211E 00	-2.921108E-02
3.400000E 00	2.100000E 00	2.079496E 00	2.050392E-02
3.500000E 00	2.250000E 00	2.238708E 00	1.129068E-02
3.600000E 00	2.430000E 00	2.428134E 00	1.865786E-03
3.700000E 00	2.600000E 00	2.612069E 00	-1.206875E-02
3.800000E 00	2.750000E 00	2.767555E 00	-1.755604E-02
3.900000E 00	2.890000E 00	2.871636E 00	1.836338E-02
4.000000E 00	2.920000E 00	2.916960E 00	3.039550E-03
4.100000E 00	2.950000E 00	2.946946E 00	3.053293E-03
4.200000E 00	2.970000E 00	2.973553E 00	-3.552493E-03
4.300000E 00	3.000000E 00	2.996966E 00	3.032722E-03
4.400000E 00	3.020000E 00	3.017381E 00	2.619695E-03
4.500000E 00	3.040000E 00	3.034980E 00	5.019724E-03
4.600000E 00	3.050000E 00	3.049955E 00	4.446507E-05
4.700000E 00	3.055000E 00	3.062496E 00	-7.496823E-03
4.800000E 00	3.065000E 00	3.072790E 00	-7.791314E-03
4.900000E 00	3.075000E 00	3.081028E 00	-6.028723E-03
5.000000E 00	3.080000E 00	3.087397E 00	-7.396754E-03

5.100000E 00	3.090000E 00	3.092069E 00	-2.069191E-03
5.200000E 00	3.100000E 00	3.095146E 00	4.853774E-03
5.300000E 00	3.100000E 00	3.096716E 00	3.283732E-03
5.400000E 00	3.100000E 00	3.096868E 00	3.132608E-03
5.500000E 00	3.100000E 00	3.095680E 00	4.319567E-03
5.600000E 00	3.100000E 00	3.093245E 00	6.755099E-03
5.700000E 00	3.100000E 00	3.089649E 00	1.035092E-02
5.800000E 00	3.090000E 00	3.084974E 00	5.025774E-03
5.900000E 00	3.080000E 00	3.079309E 00	6.905058E-04
6.000000E 00	3.070000E 00	3.072740E 00	-2.739966E-03
6.100000E 00	3.060000E 00	3.065260E 00	-5.259581E-03
6.200000E 00	3.050000E 00	3.056500E 00	-6.500497E-03
6.300000E 00	3.040000E 00	3.046000E 00	-5.999859E-03
6.400000E 00	3.030000E 00	3.033295E 00	-3.295366E-03
6.500000E 00	3.010000E 00	3.017928E 00	-7.928025E-03
6.600000E 00	3.000000E 00	2.999435E 00	5.641840E-04
6.700000E 00	2.980000E 00	2.977354E 00	2.645336E-03
6.800000E 00	2.950000E 00	2.951224E 00	-1.224987E-03
6.900000E 00	2.920000E 00	2.920587E 00	-5.867698E-04
7.000000E 00	2.890000E 00	2.884996E 00	5.003192E-03
7.100000E 00	2.850000E 00	2.844051E 00	5.948193E-03
7.200000E 00	2.800000E 00	2.797825E 00	2.174579E-03
7.300000E 00	2.750000E 00	2.746466E 00	3.533684E-03
7.400000E 00	2.690000E 00	2.690125E 00	-1.264103E-04
7.500000E 00	2.630000E 00	2.628954E 00	1.045644E-03
7.600000E 00	2.560000E 00	2.563101E 00	-3.100801E-03
7.700000E 00	2.490000E 00	2.492720E 00	-2.719902E-03
7.800000E 00	2.420000E 00	2.417955E 00	2.044205E-03
7.900000E 00	2.340000E 00	2.338963E 00	1.036749E-03
8.000000E 00	2.250000E 00	2.255881E 00	-5.881384E-03
8.100000E 00	2.160000E 00	2.169028E 00	-9.028815E-03
8.200000E 00	2.080000E 00	2.079279E 00	7.202886E-04
8.300000E 00	2.000000E 00	1.987667E 00	1.233252E-02
8.400000E 00	1.900000E 00	1.895224E 00	4.775889E-03
8.500000E 00	1.805000E 00	1.802980E 00	2.018948E-03
8.600000E 00	1.705000E 00	1.710693E 00	-5.694207E-03
8.700000E 00	1.605000E 00	1.613008E 00	-8.008420E-03
8.800000E 00	1.500000E 00	1.503284E 00	-3.283981E-03
8.900000E 00	1.375000E 00	1.374889E 00	1.102020E-04
9.000000E 00	1.240000E 00	1.221188E 00	1.881186E-02
9.100000E 00	1.030000E 00	1.038939E 00	-8.940164E-03
9.200000E 00	8.300000E-01	8.384878E-01	-8.487869E-03
9.300000E 00	6.400000E-01	6.335645E-01	6.435510E-03
9.400000E 00	4.450000E-01	4.379095E-01	7.090468E-03
9.500000E 00	2.500000E-01	2.652552E-01	-1.525517E-02
9.550000E 00	2.000000E-01	1.918465E-01	8.153468E-03

THE LEAST SQUARES ERROR IS 1.897547E-01

KNOT	LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT
0	3.500000E-01		
		0	4.035096E-01
		1	3.119434E 00
		2	5.245779E 00
		3	-2.106841E 01
1	6.000000E-01		

		0	1.182031E 00
		1	1.792031E 00
		2	-1.055562E 01
		3	1.979568E 01
2	9.000000E-01		
		0	1.304118E 00
		1	8.034888E-01
		2	7.260439E 00
		3	-3.605061E 01
3	1.000000E 00		
		0	1.421021E 00
		1	1.174047E 00
		2	-3.554706E 00
		3	6.331032E 00
4	1.200000E 00		
		0	1.564281E 00
		1	5.119148E-01
		2	2.439298E-01
		3	-1.019521E 00
5	1.400000E 00		
		0	1.668296E 00
		1	4.870993E-01
		2	-3.677540E-01
		3	1.702099E-01
6	2.000000E 00		
		0	1.864781E 00
		1	2.299367E-01
		2	-6.161657E-02
		3	-3.110615E-02
7	3.000000E 00		
		0	2.002012E 00
		1	1.324511E-02
		2	-1.546146E-01
		3	3.537191E-01
8	3.200000E 00		
		0	2.001300E 00
		1	-6.130829E-03
		2	5.760336E-02
		3	9.639282E 00
9	3.400000E 00		
		0	2.079494E 00
		1	1.173615E 00
		2	5.841224E 00
		3	-1.655806E 01
10	3.500000E 00		
		0	2.238708E 00
		1	1.845140E 00
		2	8.737149E-01
		3	-3.827115E 00
11	3.900000E 00		
		0	2.871635E 00
		1	7.071088E-01
		2	-3.718792E 00
		3	1.180118E 01

12	4.000000E 00		
		0	2.916960E 00
		1	3.173848E-01
		2	-1.784239E-01
		3	3.148174E-02
13	5.000000E 00		
		0	3.087396E 00
		1	5.497074E-02
		2	-8.398247E-02
		3	1.435216E-02
14	6.000000E 00		
		0	3.072739E 00
		1	-6.993294E-02
		2	-4.092169E-02
		3	-7.690954E-02
15	7.000000E 00		
		0	2.884996E 00
		1	-3.825288E-01
		2	-2.716408E-01
		3	2.506399E-02
16	8.000000E 00		
		0	2.255880E 00
		1	-8.505980E-01
		2	-1.964577E-01
		3	1.721067E-01
17	8.500000E 00		
		0	1.802979E 00
		1	-9.179758E-01
		2	6.170338E-02
		3	-1.105849E 00
18	9.000000E 00		
		0	1.221187E 00
		1	-1.685660E 00
		2	-1.597072E 00
		3	2.289328E 00
19	9.550000E 00		

THE BUG

Table VIII

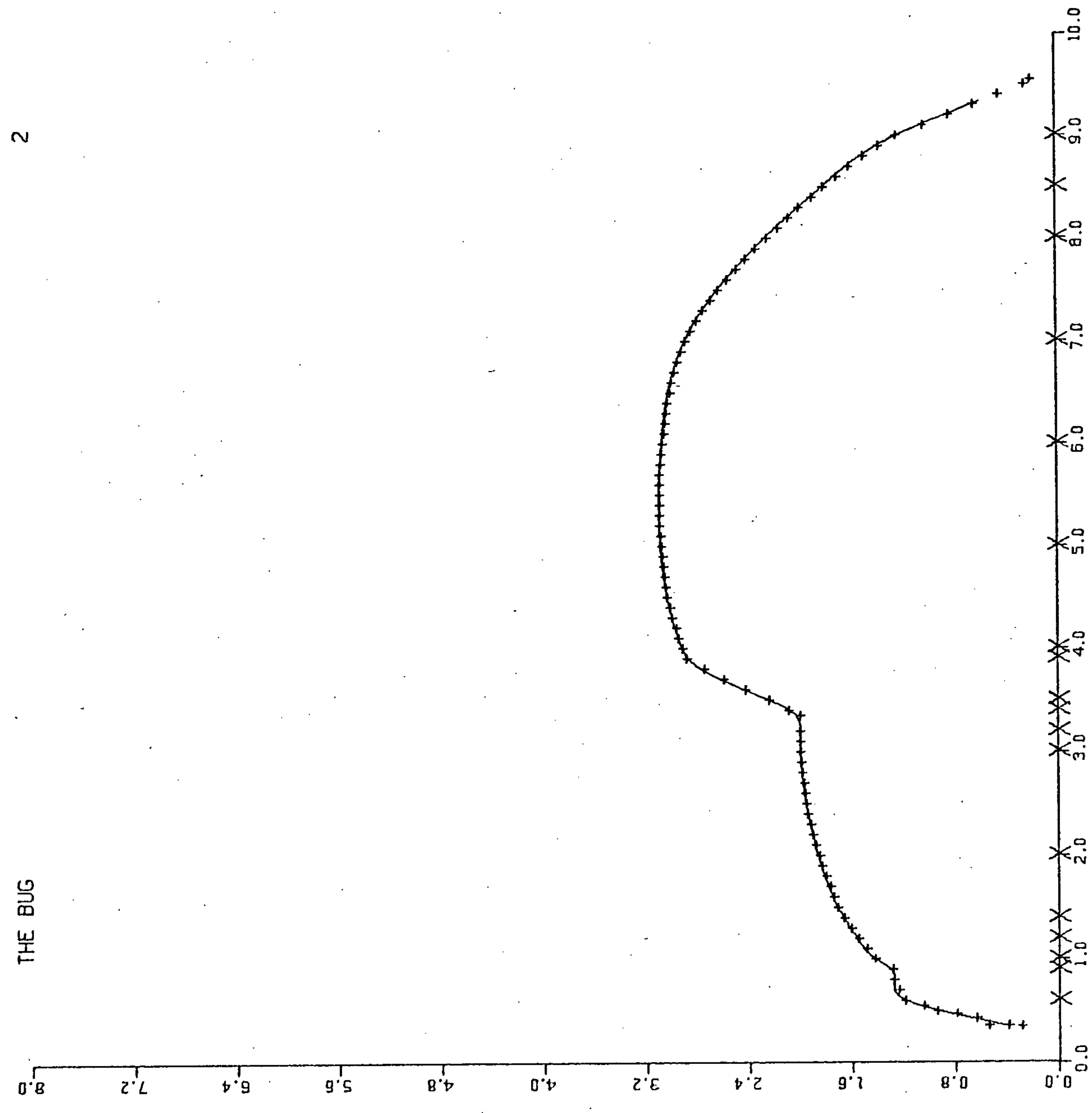

THE BUG
2
0.0
0.8
1.6
2.4
3.2
4.0
4.8
5.6
6.4
7.2
8.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0

Figure 12

Section 6

SUMMARY AND FUTURE POSSIBILITIES

The results shown in the previous section provide a brief outlook into the possibilities of interactive least squares approximation using spline functions. There remain, however, many problems and possibilities for further research in this area.

With regards to the basis functions; a possibility is to find more appropriate methods of choosing the external knots. Schultz [18, p. 73] suggests choosing the external knots so that

$$\delta_{i+1} = \delta_i = \delta_1 - \delta_0 \qquad -m \leq i \leq -1 \; ;$$

$$\delta_{i+1} - \delta_i = \delta_{k+1} - \delta_k \qquad k+1 \leq i \leq k+m \; .$$

For the case of a uniform knot set this is equivalent to the method described in Section 2. However, for the non-uniform knot set the knot spacing is different. Schultz's choice is poor when the boundary knot and first interior knot are coalescing as it causes high peaks in the basis functions involving the external and boundary knots. This causes a sharp increase in the condition number of the least squares matrix - a situation which should be avoided.

The second possibility (interconnected with the first) is to find optimal weights for the basis functions involving the external knots so as to reduce the condition number. At present the first and last basis function are arbitrarily multiplied by $(m + 1)$ so as to obtain an adequate portion of these basis functions when deriving the least squares matrix. An interesting sutdy could be made of the theoretical and practical importance of such weights for producing good bounds for the norm of the least squares matrix.

A third possibility is to produce a more efficient method of computing the B-spline basis functions using recursive formulae. These methods are described in Lyche and Schumaker [10]. They provide substantial reductions in the time involved in evaluating the B-splines as identical terms are not recomputed. Since the B-spline basis function evaluation uses the most significant amount of time in the system; derivation of an improved, more efficient algorithm for spline basis function evaluation would be highly justified.

With regards to the fixed knot least squares problem; a better estimate of the least squares norm might aid in the approximation process. Both de Boor and Rice and Patent use an integral norm in their approximation as opposed to the discretized norm (sum of the squares of the residuals) used in this system. de Boor and Rice [5], [6] estimate the norm by the trapezoidal rule; hence, in effect, obtaining a linear interpolation between fitted data points. Patent [13] used Filon quadrature (an interpolary quadrature), the theory of which is discussed in his thesis. Since the main interest of this system was interactive approximation rather than least squares approximation, the discretized norm was adequate.

With regards to the variable knot problem; better algorithms are needed for the non-linear least squares problem. In particular, an algorithm for the specific case of knot optimization would be useful.

Also the possibility of keeping knots stationary while optimizing other knots could be useful in the interactive case.

Interactive techniques are a vital asset to numerical computation. For too long, numerical analysts have been developing 'black box' algorithms in which numbers are read in and (hopefully) correct answers are printed out. By interacting with the procedure, what is happening to the solution can be observed immediately. Interaction gives the power to interrupt the solution process and change the starting conditions of the problem.

Part of the future of numerical computation depends on the development of interactive methods. Hopefully, the use of an interactive approach rather than a 'black box' approach will be valuable in solving many numerical problems.

BIBLIOGRAPHY

[1] Acton, F. S., Numerical Methods that (Usually) Work, New York: Harper and Row, 1970.

[2] Box, M. J., "A New Method of Constrained Optimization and a Comparison with Other Methods". The Computer Journal 8. 1965, pp. 42-52.

[3] Businger, P. and Golub, G., "Linear Least Squares Solutions by Householder Transformations". Numerische Mathematik 7, 1965, pp. 269-276.

[4] Carasso, C. and Laurent, P. J., "On the Numerical Construction and the Practical Use of Interpolating Spline Functions". Proceedings of the IFIP Congress, 1968, pp. A6-A9.

[5] de Boor C. and Rice, J. R., "Least Squares Cubic Spline Approximation I - Fixed Knots". Purdue University: Technical Report CSD TR 20, 1968.

[6] de Boor, C. and Rice, J. R., "Least Squares Cubic Spline Approximation II - Variable Knots". Purdue University: Technical Report CSD TR 21, 1968.

[7] Golub, G., "Numerical Methods for Solving Linear Least Squares Problems". Numerische Mathematik 7, 1965, pp. 206-216.

[8] Greville, T. N. E., "Introduction to Spline Functions". Theory and Application of Spline Functions, 1969, pp. 1-35.

[9] Greville, T. N. E. (ed.), Theory and Application of Spline Functions, New York: Academic Press, 1969.

[10] Lyche, T. and Schumaker, L., "Computation of Smoothing and Interpolating Natural Splines via Local Bases". University of Texas Center for Numerical Analysis, 1971.

[11] O'Reilly, D., "Introduction to Non-linear Function Optimization". University of British Columbia Computing Centre, 1971.

[12] O'Reilly, D., "Non-linear Function Optimization - The Complex Method of M. J. Box". University of British Columbia Computing Centre, 1971.

[13] Patent, P. D., "Least Square Polynomial Spline Approximation". California Institute of Technology: Ph.D. Dissertion, 1972.

[14] Rice, J. R., The Approximation of Functions - Vol. II, Reading, Massachusetts: Addison-Wesley, 1969.

[15] Richardson, J. A. and Kuester, J. L., "The Complex Method for Constrained Optimization". Comm. A.C.M. 8, 1973, pp. 487-489.

[16] Schoenberg, I. J. (ed.), Approximations with Special Emphasis on Spline Functions, New York: Academic Press, 1969.

[17] Schultz, M., "Multivariate Spline Functions and Elliptic Problems". Approximations with Special Emphasis on Spline Functions, 1969, pp. 279-347.

[18] Schultz, M., Spline Analysis, New York: Prentice-Hall, 1973.

[19] Smith, L., The Use of Man-Machine Interaction in Data-Fitting Problems". Stanford University: Technical Report CS 131, 1969.

Appendix A

PROGRAM LISTINGS

The following pages contain listings of the majority of the program used in the interactive spline approximation. The subroutines consist of a main control routine which calls the various subroutines in the proper order.

As is the problem with most programs written for specific hardware configurations most of the input/output subroutines are machine dependent. This program is no exception. However the subroutines should not be difficult to change to a system consisting of corresponding hardware; that is, a conversational terminal, a graphics terminal and a plotter. The routines in question are:

1. INFREE - which is a free format input routine. All calls to INFREE can be replaced by FORTRAN READ and FORMAT statements.
2. The plot subroutines PLOT, SCALE, AXIS, SYMBOL and PLOTND need only minor modifications from one plot system to another. Note, however, that PSEND is a specific subroutine for the graphics terminal which simply plots all currently available plot information.
3. AGPLOT creates a hardcopy of a plot currently being displayed on a graphics terminal. If such a routine is unavailable this routine can be omitted. However this causes the loss of the ability to keep the plot information permanently.

There is also one matrix manipulation subroutine which is not given in the program. The subroutine INVERT is used to obtain the condition number of the matrix. However a similar subroutine is available almost everywhere.

The code for the function minimization is also not given as it was not written by the author. However a similar subroutine is available in Richardson and Kuester [15].

```
C**********************************************************************
C*
C*          C O N T R O L   R O U T I N E
C*          -----------------------------
C*
C**********************************************************************
      DIMENSION  X(200), Y(200), YF(200), RES(200)
      DIMENSION A(200), DELTA(50)
      INTEGER YES /'YES'/, NO /'NO'/, ANSWER /' '/
C
C ***  ENTER AND PLOT THE DATA POINTS
207   CALL DATAIN (X, Y, N)
205   CALL DATAPL (X, Y, XMIN, XMN, XMAX, DX, YMIN, DY, N)
C
C ***  ENTER AND PLOT THE ORIGINAL KNOT SET
206   ERRORP = 1000.
      CALL KNOTIN (X, DELTA, M, N, MK, &206)
      CALL KNOTPL (DELTA, XMIN, DX, YMIN, DY, M, MK)
C
C ***  COMPUTE THE INITIAL SPLINE APPROXIMATION
      CALL FIXED (X, Y, YF, RES, A, DELTA, ERROR, ERRORP, M, N, MK)
C
C ***  PLOT THE SPLINE APPROXIMATION
      CALL SPLNPL (A, DELTA, XMIN, XMN, XMAX, DX, YMIN, DY, M, MK)
C
C ***  ASK IF HARDCOPY OUTPUT IS WANTED
      CALL SPLOUT (X, Y, YF, RES, A, DELTA, ERROR, N, M, MK)
C
C ***  ASK WHETHER OR NOT TO CONTINUE ITERATING
202   WRITE (6,600)
600   FORMAT ('&DO YOU WISH TO VARY THE KNOTS')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 200
      IF (ANSWER.EQ.NO) GO TO 201
      WRITE (6,601)
601   FORMAT ('&PLEASE ANSWER YES OR NO')
      GO TO 202
C
C ***  VARY THE KNOTS
200   CALL CONTRL (X, Y, YF, RES, A, DELTA, ERROR, N, M, MK)
      GO TO 202
C
C ***  ASK WHETHER OR NOT TO OPTIMIZE THE KNOT SET
201   WRITE (6,602)
602   FORMAT ('&DO YOU WISH TO OPTIMIZE THE KNOT SET')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 203
      IF (ANSWER.EQ.NO) GO TO 204
      WRITE (6,601)
      GO TO 201
C
C ***  OPTIMIZE THE KNOT SET
203   CALL OPT (X, Y, YF, RES, A, DELTA, ERROR, N, M, MK)
C
```

```
C ***  ASK WHETHER OR NOT TO RESTART WITH A NEW KNOT SET
204    WRITE (6,603)
603    FORMAT ('&DO YOU WISH TO RESTART WITH A NEW KNOT SET')
       CALL INFREE (2058, ANSWER, 4)
       IF (ANSWER.EQ.YES) GO TO 205
       IF (ANSWER.EQ.NO) GO TO 209
       WRITE (6,601)
       GO TO 204
C
C ***  ASK WHETHER OF NOT TO RESTART
209    WRITE (6,604)
604    FORMAT ('&DO YOU WISH TO RESTART WITH A NEW DATA SET')
       CALL INFREE (2058, ANSWER, 4)
       IF (ANSWER.EQ.YES) GO TO 207
       IF (ANSWER.EQ.NO) GO TO 208
       WRITE (6,601)
       GO TO 209
208    CALL PLOTND
       STOP
       END
```

```
      SUBROUTINE DATAIN (X, Y, N)
C
C*****************************************************************************
C*
C*          I N P U T   O F   D A T A   P O I N T S
C*          ---------------------------------------
C*
C*****************************************************************************
      DIMENSION  X(1), Y(1)
C
C ***  READ IN THE NUMBER OF DATA POINTS
      CALL INFREE (4, N)
C
C ***  READ IN THE ABSCISSA AND ORDINATE VALUES
      DO 100 L=1,N
            CALL INFREE (20, X(L), Y(L))
100   CONTINUE
C
C ///  DEBUGGING INFORMATION
      WRITE (3,301) N
301   FORMAT ('0THE ', I3, ' DATA POINTS ARE:')
      DO 400 L=1,N
            WRITE (3,300) X(L), Y(L)
400   CONTINUE
300   FORMAT (' ', 1P2E15.6)
      RETURN
      END
```

```
      SUBROUTINE KNOTIN (X, DELTA, M, N, MK, *)
C
C*********************************************************************
C*
C*          I N P U T   O F   K N O T   S E T
C*          ---------------------------------
C*
C*********************************************************************
      DIMENSION X(1), DELTA(1)
C
C ***  REQUEST THE DEGREE OF THE SPLINE
      WRITE (6,600)
600   FORMAT ('&DEGREE OF THE SPLINE')
      CALL INFREE (10, M)
C
C ***  READ IN THE NUMBER OF KNOTS
      WRITE (6,601)
601   FORMAT ('&NUMBER OF KNOTS')
      CALL INFREE (10, K)
C
C ***  CALCULATE THE TOTAL NUMBER OF KNOTS
      MK = M + K + 1
C
C ***  CHECK FOR AN UNDERDETERMINED SYSTEM
      IF (MK.LE.N) GO TO 200
      WRITE (6,603)
603   FORMAT ('0SYSTEM IS UNDERDETERMINED:'/
     *        ' REDUCE DEGREE OF SPLINE OR NUMBER OF KNOTS')
      RETURN 1
C
C ***  READ IN THE POSITION OF THE KNOTS
200   DO 100 I=1,K
            IM = I + M + 1
            WRITE (6, 602) I
            CALL INFREE (26, DELTA(IM))
100   CONTINUE
602   FORMAT ('&POSITION OF KNOT (', I2, ')')
C
C ***  CALCULATE THE POSITION OF THE SUPPLEMENTARY KNOTS
      H = (X(N) - X(1)) / (K + 1)
      M1 = M + 1
      DO 101 I=1,M1
            DELTA(I) = X(1) + (I - M1) * H
            DELTA(MK+I) = X(N) + (I - 1) * H
101   CONTINUE

      RETURN
      END
```

```
      SUBROUTINE DATAPL (X, Y, XMIN, XMN, XMAX, DX, YMIN, DY, N)
C
C***************************************************************************
C*
C*          P L O T   O F   D A T A   P O I N T S
C*          ---------------------------------------
C*
C***************************************************************************
      DIMENSION X(1), Y(1)
      DIMENSION XS(200), YS(200)
C
C ***  RESET THE PLOTTER
      CALL PLOT (15., 0., -3)
C
C ***  PROTECT THE ORIGINAL DATA POINTS
      XMN = X(1)
      XMAX = X(N)
      DO 100 L=1,N
            XS(L) = X(L)
            YS(L) = Y(L)
100   CONTINUE
C
C ***  SCALE THE ABSCISSA AND ORDINATE VALUES
      CALL SCALE (XS, N, 10., XMIN, DX, 1)
      CALL SCALE (YS, N, 10., YMIN, DY, 1)
C
C ***  PLOT THE DATA POINTS
      DO 101 L=1,N
            CALL SYMBOL (XS(L)-.035, YS(L)-.07, .14, '+', 0., 1)
101   CONTINUE
C ***  PLOT THE AXES
      ORY = -YMIN / DY
      IF (ORY.LT.0.) ORY = 0.
      CALL AXIS (0., ORY, ' ', -1, 10., 0., XMIN, DX)
      ORX = -XMIN / DX
      IF (ORX.LT.0.) ORX = 0.
      CALL AXIS (ORX, 0., ' ', 1, 10., 90., YMIN, DY)
C
C ***  DISPLAY THE PLOT
      CALL PSEND
      RETURN
      END
```

```
      SUBROUTINE SPLNPL (A, DELTA, XMIN, XMN, XMAX, DX, YMIN,
     #                  DY, M, MK)
C
C***********************************************************************
C*
C*          P L O T   O F   S P L I N E   C U R V E
C*          ---------------------------------------
C*
C***********************************************************************
      DIMENSION A(1), DELTA(1)
      COMMON /DW/ DOMEGA(50,5)
C
C ***  CALCULATE THE SPLINE FUNCTION DENOMINATOR
      CALL OMEGA (DELTA, M, MK)
C
C ***  RAISE THE PEN TO REPOSITION IT
      IPEN = 3
      X = XMN
      XSTEP = .02 * DX
      NP = 50 * INT ((XMAX - XMN) / DX + .5)
      DO 100 J=1,NP
C
C ***  CALCULATE THE SPLINE FUNCTION VALUE AT THE CURRENT X VALUE
            Y = 0.
            DO 101 I=1,MK
                  Y = Y + A(I) * SPLINE (DELTA, X, M, MK, I, 0)
101         CONTINUE
C
C *** SCALE THE X AND Y VALUES
            XP = (X - XMIN) / DX
            YP = (Y - YMIN) / DY
C
C ***  IF THE POINT IS WITHIN THE RANGE
            IF ((YP.GT.10.5).OR.(YP.LT.-.5)) GO TO 200
C
C ***  THEN PLOT IT
            CALL PLOT (XP, YP, IPEN)
            IPEN = 2
            GO TO 201
C
C ***  OTHERWISE RAISE THE PEN
200         IPEN = 3
201         X = XMN + J * XSTEP
100   CONTINUE
C
C ***  DISPLAY THE SPLINE CURVE
      CALL PSEND
      RETURN
      END
```

```
      SUBROUTINE KNOTPL (DELTA, XMIN, DX, YMIN, DY, M, MK)
C
C***********************************************************************
C*
C*          P L O T   O F   K N O T   S E T
C*          -------------------------------
C*
C***********************************************************************
      DIMENSION DELTA(1)
C
C ***  SCALE THE KNOT ORDINATE ONTO THE X-AXIS
      YP = -YMIN / DY
      IF (YP.LT.0.0) YP = 0.0
      M2 = M + 2
      DO 100 I=M2,MK
C
C ***  SCALE THE KNOT
            XP = (DELTA(I) - XMIN) / DX
C
C ***  PLOT THE KNOT
            CALL SYMBOL (XP-.0525, YP-.115, .21, 'X', 0., 1)
100   CONTINUE
C
C ***  DISPLAY THE KNOTS
      CALL PSEND
      RETURN
      END
```

```
      SUBROUTINE FIXED(X, Y, YF, RES, A, DELTA, ERROR, ERRORP, M, N, MK
C
C**********************************************************************
C*
C*                 F I X E D   K N O T   L I N E A R
C*           L E A S T   S Q U A R E S   A P P R O X I M A T I O N
C*           -----------------------------------------------------
C*
C**********************************************************************
      DIMENSION X(1), Y(1), YF(1), RES(1), A(1), DELTA(1)
      DIMENSION S(200,50)
C
C ***  CALCULATE THE SPLINE FUNCTION DENOMINATOR
      CALL OMEGA (DELTA, M, MK)
C
C ***  FORM THE MATRIX OF SPLINE BASIS FUNCTION VALUES
      DO 100 I=1,MK
            DO 101 L=1,N
                  S(L,I) = SPLINE (DELTA, X(L), M, MK, I, 0)
101         CONTINUE
100   CONTINUE
C
C ///  DEBUGGING INFORMATION
      WRITE (3,300)
300   FORMAT ('0AT ', 13X, 'THE HOUSEHOLDER MATRIX IS:')
      DO 400 L=1,N
            WRITE (3,301) X(L)
            WRITE (3,302) (S(L,I), I=1,MK)
400   CONTINUE
301   FORMAT (' ', 1P8E15.6)
302   FORMAT (' ', 15X, 1P7E15.6)
C
C ***  TRANSFER Y INTO A
      DO 102 L=1,N
            A(L) = Y(L)
102   CONTINUE
C
C ///  DEBUGGING INFORMATION
      WRITE (3,303)
303   FORMAT ('0THE HOUSEHOLDER VECTOR IS:')
      WRITE (3,301) (A(L), L=1,N)
C
C ***  PERFORM THE HOUSEHOLDER TRANSFORMATIONS
      CALL TRANS (S, A, N, MK)
C
C ***  CALCULATE THE RESIDUALS AND LEAST SQUARES ERROR
      ERROR = RESERR (X, Y, YF, RES, A, DELTA, N, M, MK)
      CALL ITER (ERRORP, ERROR)
      RETURN
      END
```

```
      FUNCTION RESERR (X, Y, YF, RES, A, DELTA, N, M, MK)
C
C**************************************************************************
C*
C*                         R E S I D U A L S   A N D
C*                   L E A S T   S Q U A R E S   E R R O R
C*                   -------------------------------------
C*
C**************************************************************************
      DIMENSION X(1), Y(1), YF(1), RES(1), A(1), DELTA(1)
      DOUBLE PRECISION DY, DRES, DERR
      COMMON /DW/ DOMEGA(50,5)
C
C ***  CALCULATE THE SPLINE FUNCTION DENOMINATOR
      CALL OMEGA (DELTA, M, MK)
      DERR = 0.D0
      DO 100 L=1,N
C
C ***  CALCULATE THE SPLINE FUNCTION VALUES AT THE DATA POINTS
            DY = 0.D0
            DO 101 I=1,MK
                  DY = DY + A(I) * SPLINE (DELTA, X(L), M, MK, I, 0)
101         CONTINUE
            YF(L) = DY
C
C ***  CALCULATE THE RESIDUALS
            DRES = Y(L) - DY
            RES(L) = DRES
C
C ***  CALCULATE THE LEAST SQUARES ERROR
            DERR = DERR + DRES * DRES
100   CONTINUE
      RESERR = DSQRT (DERR)
C
C ///  DEBUGGING INFORMATION
      WRITE (3,300) RESERR
300   FORMAT ('0THE LEAST SQUARES ERROR IS ', 1PE15.6)
      RETURN
      END
```

```
      SUBROUTINE CONTRL (X, Y, YF, RES, A, DELTA, ERRORP, N, M, MK)
C
C*******************************************************************************
C*
C*
C*          V A R I A B L E   K N O T   C O N T R O L
C*          -----------------------------------------
C*
C*******************************************************************************
      DIMENSION X(1), Y(1), YF(1), RES(1), A(1), DELTA(1)
      DIMENSION AP(200), DELTAP(50), RESP(200), YFP(200)
      INTEGER YES /'YES'/, NO /'NO'/, ANSWER /' '/
      MMK = MK + M + 1
C
C ***  TRANSFER THE PREVIOUS KNOT SET
      DO 100 I=1,MMK
           DELTAP(I) = DELTA(I)
100   CONTINUE
601   FORMAT ('&PLEASE ANSWER YES OR NO')
C
C ***  OBTAIN WHICH KNOT TO VARY
200   CALL VARYIN (I)
C
C ***  REPOSITION THAT KNOT
      IM = I + M + 1
      CALL VARYKT (DELTAP(IM), I)
C
C *** ASK WHETHER OR NOT TO REPOSITION OTHER KNOTS
204   WRITE (6,602)
602   FORMAT ('&DO YOU WISH TO VARY ANOTHER KNOT')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 200
      IF (ANSWER.EQ.NO) GO TO 203
      WRITE (6,601)
      GO TO 204
203   CONTINUE
C
C ///  DEBUGGING INFORMATION
      WRITE (3,300)
300   FORMAT ('0THE NEW KNOT SET IS:')
      WRITE (3,301) (DELTAP(I), I=1,MMK)
301   FORMAT (' ', 1P8E15.6)
C
C ***  RECALCULATE THE SPLINE APPROXIMATION
      CALL FIXED (X, Y, YFP, RESP, AP, DELTAP, ERROR, ERRORP, M, N, *MK)
C
C ***  ASK WHETHER OR NOT TO PLOT THE NEW APPROXIMATION
207   WRITE (6,603)
603   FORMAT ('&DO YOU WISH TO SEE THE NEW APPROXIMATION')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 205
      IF (ANSWER.EQ.NO) GO TO 201
      WRITE (6,601)
      GO TO 207
```

```
 205    CALL DATAPL (X, Y, XMIN, XMN, XMAX, DX, YMIN, DY, N)
        CALL KNOTPL (DELTAP, XMIN, DX, YMIN, DY, M, MK)
        CALL SPLNPL (AP, DELTAP, XMIN, XMN, XMAX, DX, YMIN, DY, M, MK)
C
C ***   DECIDE WHETHER OR NOT TO KEEP NEW APPROXIMATION
 206    WRITE (6,604)
 604    FORMAT ('&DO YOU WISH TO CONTINUE WITH THE NEW APPROXIMATION')
        CALL INFREE (2058, ANSWER, 4)
        IF (ANSWER.EQ.YES) GO TO 208
        IF (ANSWER.EQ.NO) GO TO 209
        WRITE (6,601)
        GO TO 206
C
C ***   IF SO, RETAIN THE NEW VALUES
 208    DO 101 I=1,MK
               A(I) = AP(I)
 101    CONTINUE
        DO 102 I=1,MMK
               DELTA(I) = DELTAP(I)
 102    CONTINUE
        DO 103 L=1,N
               YF(L) = YFP(L)
               RES(L) = RESP(L)
 103    CONTINUE
C
C ***   ASK IF A HARDCOPY IS WANTED
        CALL SPLOUT (X, Y, YF, RES, A, DELTA, ERROR, N, M, MK)
        ERRORP = ERROR
        GO TO 201
C
C ***   REPLOT THE OLD SPLINE
 209    CALL DATAPL (X, Y, XMIN, XMN, XMAX, DX, YMIN, DY, N)
        CALL KNOTPL (DELTA, XMIN, DX, YMIN, DY, M, MK)
        CALL SPLNPL (A, DELTA, XMIN, XMN, XMAX, DX, YMIN, DY, M, MK)
 201    RETURN
        END
```

```
      SUBROUTINE VARYIN (I)
C
C*************************************************************************
C*
C*          I N P U T   O F   V A R I A B L E   K N O T
C*          -------------------------------------------
C*
C*************************************************************************
      WRITE (6,600)
600   FORMAT ('&KNOT TO BE VARIED')
      CALL INFREE (10, I)
      RETURN
      END



      SUBROUTINE VARYKT (DELTAI, I)
C
C*************************************************************************
C*
C*          V A R I A B L E   K N O T   V A L U E
C*          -------------------------------------
C*
C*************************************************************************
      WRITE (6,600) I
600   FORMAT ('&NEW POSITION FOR KNOT (', I2, ')')
      CALL INFREE (26, DELTAI)
      RETURN
      END
```

```
      SUBROUTINE ITER (ERRORP, ERROR)
C
C**************************************************************************
C*
C*          I T E R A T I O N   O U T P U T
C*          --------------------------------
C*
C**************************************************************************
      WRITE (6,600) ERRORP
600   FORMAT ('0THE LEAST SQUARES ERROR OF THE PREVIOUS ',
     *        'APPROXIMATION WAS ', 1PE15.6)
      WRITE (6,601) ERROR
601   FORMAT (' THE LEAST SQUARES ERROR OF THE NEW ',
     *           'APPROXIMATION IS ', 1PE15.6)
      RETURN
      END
```

```
      SUBROUTINE SPLOUT (X, Y, YF, RES, A, DELTA, ERROR, N, M, MK)
C
C***********************************************************************
C*
C*          O U T P U T
C*          -----------
C*
C***********************************************************************
      DIMENSION  X(1), Y(1), YF(1), RES(1), A(1), DELTA(1)
      INTEGER YES /'YES'/, NO /'NO'/, ANSWER /' '/
      INTEGER TITLE(10), BLANK /'    '/, NUM /1/
C
C ***  ASK WHETHER OR NOT TO RETAIN RESULTS
200   WRITE (6,600)
600   FORMAT ('&DO YOU WISH TO RETAIN THE APPROXIMATION RESULTS')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 201
      IF (ANSWER.EQ.NO) GO TO 202
      WRITE (6,601)
601   FORMAT ('&PLEASE ANSWER YES OR NO')
      GO TO 200
C
C ***  BLANK OUT THE TITLE
201   DO 101 I=1,10
            TITLE(I) = BLANK
101   CONTINUE
C
C ***  ASK FOR THE TITLE OF THE PLOT
      WRITE (6,602)
602   FORMAT ('&TITLE FOR PLOT')
      CALL INFREE (2058, TITLE, 40)
C
C ***  HARDCOPY PRINTOUT
      WRITE (8, 603) TITLE, NUM
603   FORMAT ('1', 10A4, 20X, I5)
      WRITE (8, 604)
604   FORMAT ('0', 5X, 'ABSCISSAE', 5X, 'ORDINATES', 3X,
     *        'FITTED ORDINATES', 3X, 'RESIDUALS')
C
C ***  PRINT OUT THE DATA, FITTED RESULTS AND RESIDUALS
      DO 100 L=1,N
            WRITE (8, 605) X(L), Y(L), YF(L), RES(L)
100   CONTINUE
605   FORMAT (' ', 1P5E15.6)
C
C ***  PRINT OUT THE LEAST SQUARES ERROR
      WRITE (8,606) ERROR
606   FORMAT ('0THE LEAST SQUARES ERROR IS', 1PE15.6)
C
C ***  CALCULATE THE POLYNOMIAL COEFFICIENTS
      CALL COEFF (X, Y, A, DELTA, N, M, MK)
C
C ***  PLOT A HARDCOPY OF THE APPROXIMATION
      CALL SYMBOL (1., 10., .14, TITLE, 0., 40)
```

```
      FNUM = NUM
      CALL NUMBER (9., 10., .14, FNUM, 0., -1)
      CALL PSEND
C     CALL AGPLOT (10.0, 6000)
      NUM = NUM + 1
202   RETURN
      END
```

```
      SUBROUTINE NORM (S, MK)
C
C*********************************************************************
C*
C*          M A T R I X   C O N D I T I O N   N U M B E R
C*          ---------------------------------------------
C*
C*********************************************************************
      DIMENSION S(200,50), ST(50,50)
C
C ***  SAVE THE ORIGINAL MATRIX
      DO 100 I=1,MK
            DO 100 L=1,MK
                  ST(L,I) = S(L,I)
100   CONTINUE
C
C ***  INVERT THE MATRIX AND CALCULATE ITS CONDITION NUMBER
      CALL INVERT (ST, MK, 50, DET, COND)
C
C ///  DEBUGGING INFORMATION
      WRITE (3,300)
300   FORMAT ('0THE INVERTED TRANSFORMED HOUSEHOLDER MATRIX IS:')
      DO 400 I=1,MK
            WRITE (3,301)  (ST(L,I), L=1,MK)
            WRITE (3,302)
400   CONTINUE
301   FORMAT (' ', 1P8E15.6)
302   FORMAT (' ')
C
C ***  PRINT OUT THE CONDITION NUMBER
      WRITE (6,600) COND
600   FORMAT ('0THE CONDITION NUMBER IS', 1PE15.6)
      RETURN
      END
```

```
      SUBROUTINE TRANS (S, B, N, MK)
C
C************************************************************************
C*
C*          H O U S E H O L D E R   T R A N S F O R M A T I O N
C*                C O N T R O L   R O U T I N E
C*                -------------------------------
C*
C************************************************************************
      DIMENSION S(200,50), V(200)
      DIMENSION B(1)
      REAL KSQ
C
C ***  AUGMENT THE MATRIX S BY B
      MK1 = MK + 1
      DO 100 L=1,N
            S(L,MK1) = B(L)
100   CONTINUE
C
C ***  PERFORM THE HOUSEHOLDER TRANSFORMS
      DO 101 I=1,MK
C
C ***  TRANSFER CURRENT COLUMN OF HOUSEHOLDER MATRIX
            DO 102 L=1,N
                  V(L) = S(L,I)
102         CONTINUE
C
C ***  DERIVE HOUSEHOLDER TRANSFORMATION VECTOR
            CALL UVEC (V, N, I, KSQ)
C
C ***  PERFORM HOUSEHOLDER TRANSFORMATION ON THE VECTOR
            CALL HOUSE (S, V, N, MK1, I, KSQ)
101   CONTINUE
C
C ///  DEBUGGING INFORMATION
      WRITE (3,300)
300   FORMAT ('OTHE TRANSFORMED HOUSEHOLDER MATRIX IS:')
      DO 400 L=1,N
            WRITE (3,301) (S(L,I), I=1,MK)
            WRITE (3,302)
400   CONTINUE
301   FORMAT (' ', 1P8E15.6)
302   FORMAT (' ')
C
C ***  CALCULATE THE CONDITION NUMBER OF THE MATRIX
      CALL NORM (S, MK)
C
C ***  PERFORM THE BACKWARDS SUBSTITUTION
      CALL SOLVE (S, B, N, MK)
      RETURN
      END
```

```
      SUBROUTINE UVEC (V, N, I, KSQ)
C
C***************************************************************************
C*
C*          H O U S E H O L D E R   V E C T O R
C*          -----------------------------------
C*
C***************************************************************************
      DIMENSION V(1)
      DOUBLE PRECISION DNORM
      REAL KSQ
C
C ***  COMPUTE THE NORM OF V
      DNORM = 0.D0
      DO 100 L=I,N
            DNORM = DNORM + V(L) * V(L)
100   CONTINUE
      DNORM = DSQRT (DNORM)
C
C ***  CALCULATE THE COEFFICIENT
      NSGN = 1
      IF (V(I).LT.0.) NSGN = -1
      KSQ = 2 * DNORM * (DNORM + NSGN * V(I))
C
C ***  CREATE THE HOUSEHOLDER VECTOR
      V(I) = V(I) + NSGN * DNORM
      RETURN
      END
```

```
      SUBROUTINE HOUSE (S, U, N, MK1, I, KSQ)
C
C***********************************************************************
C*
C*          H O U S E H O L D E R   T R A N S F O R M A T I O N
C*          ----------------------------------------------------
C*
C***********************************************************************
      DIMENSION S(200,50), U(1)
      REAL KSQ
C
C ***  PERFORM HOUSEHOLDER TRANSFORMATION ON REQUIRED COLUMNS OF MATRIX
      DO 100 J=I,MK1
C
C ***  MULTIPLY THE TRANSPOSE OF THE VECTOR
C      BY THE COLUMN OF THE MATRIX
            UT = 0.
            DO 101 L=I,N
                  UT = UT + U(L) * S(L,J)
101         CONTINUE
            UT = 2. * UT / KSQ
C
C ***  PERFORM THE HOUSEHOLDER TRANSFORMATION
            DO 102 L=I,N
                  S(L,J) = S(L,J) - UT * U(L)
102         CONTINUE
100   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SOLVE (S, B, N, MK)
C
C***********************************************************************
C*
C*          B A C K W A R D S   S U B S T I T U T I O N
C*          -------------------------------------------
C*
C***********************************************************************
      DIMENSION S(200,1), B(1)
      MK1 = MK + 1
C
C ***  COMPUTE THE FINAL ELEMENT
      B(MK) = S(MK,MK1) / S(MK,MK)
      IF (MK.EQ.1) GO TO 200
      MKM1 = MK - 1
      DO 100 I=1,MKM1
            MKI = MK - I
            B(MKI) = 0.
            MKI1 = MKI + 1
            DO 101 J=MKI1,MK
                  B(MKI) = B(MKI) + B(J) * S(MKI,J)
101         CONTINUE
            B(MKI) = (S(MKI,MK1) - B(MKI)) / S(MKI,MKI)
100   CONTINUE
C
C ***  TRANSFER THE REMAINING ELEMENTS INTO B
200   IF (N.EQ.MK) GO TO 201
      DO 102 L=MK1,N
            B(L) = S(L,MK1)
102   CONTINUE
201   RETURN
      END
```

```
      SUBROUTINE OPT (X, Y, YF, RES, A, DELTA, ERRORP, N, M, MK)
C
C************************************************************************
C*
C*          O P T I M I Z A T I O N   O F   K N O T S
C*          -----------------------------------------
C*
C************************************************************************
      DIMENSION X(1), Y(1), YF(1), RES(1), A(1), DELTA(1)
      DIMENSION  AP(200), DELTAP(50), YFP(200), RESP(200)
      DIMENSION P(50,75)
      INTEGER YES /'YES'/, NO /'NO'/, ANSWER /' '/
      EXTERNAL RESERR, GDELTA, HDELTA
C
C ***  TRANSFER KNOTS FOR OPTIMIZATION
      MMK = MK + M + 1
      DO 300 I=1,MMK
            DELTAP(I) = DELTA(I)
300   CONTINUE
C
C ***  TRANSFER PARAMETERS FOR OPTIMIZATION
      DO 301 I=1,MK
            AP(I) = A(I)
301   CONTINUE
C
C ***  TRANSFER THE KNOTS TO THE SIMPLEX
      DO 102 J=1,MMK
            P(J,1) = DELTAP(J)
102   CONTINUE
C
C ***  COMPUTE THE NUMBER OF POINTS IN THE SIMPLEX
      IF (MMK.GE.10) NPLUS = (3 * MMK + 1) / 2
      IF (MMK.LT.10) NPLUS = 2 * MMK
C
C ***  MINIMIZE THE LEAST SQUARES ERROR
      CALL COMPLX (X, Y, YFP, RESP, AP, DELTAP, ERRORP,
     #             P, 50, MMK, NPLUS, MMK, M, N, MK, 1.3, -1,
     #             100, 100, 1, -.0001, RESERR, GDELTA,
     #             HDELTA, GDELTA, &200, &200)
C
C ***  RECALCULATE THE FIXED KNOT APPROXIMATION
      CALL FIXED (X, Y, YFP, RESP, AP, DELTAP, ERROR, ERRORP, M, N, MK)
      GO TO 207
C
C ***  IF KNOT OPTIMIZATION FAILED, PRINT AN ERROR MESSAGE
200   WRITE (6,801)
801   FORMAT ('0OPTIMIZATION FAILED.')
C
C ***  ASK WHETHER OR NOT TO PLOT THE NEW APPROXIMATION
207   WRITE (6,603)
603   FORMAT ('&DO YOU WISH TO SEE THE NEW APPROXIMATION')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 205
      IF (ANSWER.EQ.NO) GO TO 201
```

```
      WRITE (6,601)
601   FORMAT ('&PLEASE ANSWER YES OR NO')
      GO TO 207
205   CALL DATAPL (X, Y, XMIN, XMN, XMAX, DX, YMIN, DY, N)
      CALL KNOTPL (DELTAP, XMIN, DX, YMIN, DY, M, MK)
      CALL SPLNPL (AP, DELTAP, XMIN, XMN, XMAX, DX, YMIN, DY, M, MK)
C
C ***  DECIDE WHETHER OR NOT TO KEEP NEW APPROXIMATION
206   WRITE (6,604)
604   FORMAT ('&DO YOU WISH TO CONTINUE WITH THE NEW APPROXIMATION')
      CALL INFREE (2058, ANSWER, 4)
      IF (ANSWER.EQ.YES) GO TO 208
      IF (ANSWER.EQ.NO) GO TO 209
      WRITE (6,601)
      GO TO 206
C
C ***  IF SO, RETAIN THE NEW VALUES
208   DO 101 I=1,MK
            A(I) = AP(I)
101   CONTINUE
      DO 302 I=1,MMK
            DELTA(I) = DELTAP(I)
302   CONTINUE
      DO 103 L=1,N
            YF(L) = YFP(L)
            RES(L) = RESP(L)
103   CONTINUE
C
C ***  ASK IF A HARDCOPY IS WANTED
      CALL SPLOUT (X, Y, YF, RES, A, DELTA, ERROR, N, M, MK)
      ERRORP = ERROR
      GO TO 201
C
C ***  REPLOT THE OLD SPLINE
209   CALL DATAPL (X, Y, XMIN, XMN, XMAX, DX, YMIN, DY, N)
      CALL KNOTPL (DELTA, XMIN, DX, YMIN, DY, M, MK)
      CALL SPLNPL (A, DELTA, XMIN, XMN, XMAX, DX, YMIN, DY, M, MK)
201   RETURN
      END
```

```
      FUNCTION GDELTA (T, M, MMK, J)
C
C***********************************************************************
C*
C*          L O W E R   B O U N D   C O N S T R A I N T
C*          -------------------------------------------
C*
C***********************************************************************
      DIMENSION T(1)
C
C ***  SET THE LOWER BOUNDS ON THE KNOTS
      IF ((J.LE.M+1).OR.(J.GT.(MMK-(M+1)))) GO TO 100
      H = T(MMK-(M+1)) - T(M+1)
      GDELTA = T(J-1) + .0001 * H
      GO TO 999
100   GDELTA = T(J)
999   RETURN
      END


      FUNCTION HDELTA (T, M, MMK, J)
C
C***********************************************************************
C*
C*          U P P E R   B O U N D   C O N S T R A I N T
C*          -------------------------------------------
C*
C***********************************************************************
      DIMENSION T(1)
C
C ***  SET THE UPPER BOUNDS ON THE KNOTS
      IF ((J.LE.M+1).OR.(J.GT.(MMK-(M+1)))) GO TO 100
      H = T(MMK-(M+1)) - T(M+1)
      HDELTA = T(J+1) - .0001 * H
       GO TO 999
100   HDELTA = T(J)
999   RETURN
      END
```

```
      SUBROUTINE COEFF (X, Y, A, DELTA, N, M, MK)
C
C***********************************************************************
C*
C*          P O L Y N O M I A L   C O E F F I C I E N T S
C*          ---------------------------------------------
C*
C***********************************************************************
      DIMENSION X(1), Y(1), A(1), DELTA(1)
      DIMENSION C(50,4)
      COMMON /DW/ DOMEGA(50,5)
C
C ***  CALCULATE THE SPLINE DENOMINATOR
      CALL OMEGA (DELTA, M, MK)
C
C ***  DETERMINE THE NUMBER OF PIECEWISE POLYNOMIALS
      K = MK - M
      M1 = M + 1
C
C ***  DETERMINE THE COEFFICIENTS OF THE PIECEWISE POLYNOMIALS
      JFACT = 1
      DO 100 J=1,M1
            J1 = J - 1
            DO 101 I=1,K
                  IM = I + M
                  C(I,J) = 0.0
                  DO 102 L=1,MK
                        C(I,J) = C(I,J) + A(L) *
     *                     SPLINE (DELTA, DELTA(IM), M, MK, L, J1)
102               CONTINUE
                  C(I,J) = C(I,J) / JFACT
101         CONTINUE
            JFACT = JFACT * J
100   CONTINUE
C
C ***  PRINT OUT THE COEFFICIENTS
      WRITE (8,800)
800   FORMAT ('0 KNOT', 5X, 'LOCATION', 5X, 'POLYNOMIAL POWER',
     *        5X, 'POLYNOMIAL COEFFICIENT')
      DO 103 I=1,K
            I1 = I - 1
            IM = I + M
            WRITE (8,801) I1, DELTA(IM)
            DO 104 J=1,M1
                  J1 = J - 1
                  WRITE (8,802) J1, C(I,J)
104         CONTINUE
103   CONTINUE
      MK1 = MK + 1
      WRITE (8,801) K, DELTA(MK1)
801   FORMAT (' ', I4, 2X, 1PE15.6)
802   FORMAT (' ', 30X, I1, 15X, 1PE15.6)
      RETURN
      END
```

```
      FUNCTION SPLINE (DELTA, X, M, MK, I, J)
C
C*****************************************************************************
C*
C*                    S P L I N E   F U N C T I O N
C*                    A N D   D E R I V A T I V E S
C*                    ----------------------------
C*
C*****************************************************************************
      DIMENSION DELTA(1)
      COMMON /DW/ DOMEGA(50,5)
      DOUBLE PRECISION DSPLN, DX, DDELTA, DY
      MCON = 1
      MJ = M - J
      DSPLN = 0.D0
C
C ***  CHECK IF THE POINT IS IN THE REGION OF SUPPORT
      M1 = M + 1
      IF ((X.LT.DELTA(I)).OR.(X.GT.DELTA(I+M1))) GO TO 200
C
C ***  EVALUATE THE SPLINE AT THE GIVEN POINT
      DX = X
      M2 = M + 2
      DO 100 L=1,M2
            LL = I + L - 1
            DDELTA = DELTA(LL)
            DY = DDELTA - DX
            IF (DY.LE.0.D0) GO TO 100
            DSPLN = DSPLN + DY ** MJ / DOMEGA (I,L)
100   CONTINUE
      IF ((I.EQ.1).OR.(I.EQ.MK)) DSPLN = M1 * DSPLN
C
C ***  COMPUTE THE CONSTANT TERM
      IF (J.EQ.0) GO TO 200
      MJ1 = MJ + 1
      DO 101 L=MJ1,M
            MCON = -MCON * L
101   CONTINUE
200   SPLINE = MCON * DSPLN
      RETURN
      END
```

```
      SUBROUTINE OMEGA (DELTA, M, MK)
C
C*****************************************************************************
C*
C*          S P L I N E   F U N C T I O N   D E N O M I N A T O R
C*          -------------------------------------------------------
C*
C*****************************************************************************
      DIMENSION DELTA(1)
      DOUBLE PRECISION DDELTA
      COMMON /DW/ DOMEGA(50,5)
C
C ***  CALCULATE THE MATRIX OF VALUES FOR THE DENOMINATOR
C ***  OF THE SPLINE FUNCTION
      M2 = M + 2
      DO 100 I=1,MK
           DO 100 J=1,M2
                  DOMEGA(I,J) = 1.
                  JJ = I + J - 1
                  DO 100 L=1,M2
                         IF (L.EQ.J) GO TO 100
                         LL = I + L - 1
                         DDELTA = DELTA(JJ) - DELTA(LL)
                         DOMEGA(I,J) = DOMEGA(I,J) * DDELTA
100   CONTINUE
C
C ///  DEBUGGING INFORMATION
      WRITE (3,300)
300   FORMAT ('OTHE BASIS FUNCTION DENOMINATOR VALUES ARE:')
      DO 400 I=1,MK
            WRITE (3,301) (DOMEGA(I,J), J=1,M2)
400   CONTINUE
301   FORMAT (' ', 1P5E15.6)
      RETURN
      END
```

Appendix B

INTERACTIVE SPLINE APPROXIMATION

Purpose

This system enables the use to perform least squares approximations using spline functions. These approximations are performed interactively with the aid of a graphics terminal. The approximations are displayed immediately after computation and the user can respecify knot locations and recalculate the fit. Features are included to optimize the knot set and change the number of knots in the knot set.

Type of Routine

This is a self-contained program written in FORTRAN IV.

How to Use

To run this program under MTS at the Adage Graphics Terminal; enter the command;

```
$RUN IRAM:SPLINE+AGT:BASIC 3=debugfile 4=datafile 8=printfile 9-plotfile
```

where

debugfile — contains the debugging information. Unless the system runs into problems with the approximation this should be set to *DUMMY* .

datafile — is the file containing the input data. The format is described in Section A: Data Input Format.

printfile is the file containing the printed output from an approximation corresponding to a plot.

plotfile is the file containing the plot information to be retained for a hardcopy.

Upon completion of the program a hardcopy of the printouts and plots which were saved can be obtained by issuing the following commands:

```
$COPY printfile *PRINT*

$R PLOT:Q PAR=plotfile
```

where 'printfile' contains the print output described above and 'plotfile' contains the plot information.

Description

A. Data Input Format

The data file has the following structure:
The first line states the number of points in the first data set followed by data lines with the sequence

ABSCISSA   ORDINATE.

The remaining data sets follow with an identical format. The data itself is in free-format providing that at least one blank delimits each entry.

B. Displays

Because of the interactive capabilities of the system, displays play an integral part in the structure. Inorder to best describe the flow of control through the system a sample run follows with comments inserted to augment the display information.

Example: Demonstration run

$SIG IRAM 'DEMONSTRATION RUN'

PASSWORD

Signon information

$COMMENT LOAD GRAPH IF NECESSARY

$COPY AGT:GRAPH > AGTI

$COMMENT SPLINE APPROXIMATION PROGRAM

$RUN IRAM:SPLINE+AGT:BASIC 3=*DUMMY* 4=DATA 8=-PRINT 9=-PLOT

EXECUTION BEGINS

(One data set from unit 4 is immediately read in. A plot of this set appears on the graphics display.)

DEGREE OF THE SPLINE? 3

(The degree of the piecewise polynomials wanted is requested, most frequently used degree is 3; that is, a cubic spline approximation.)

NUMBER OF KNOTS? 4

(The number of internal knots wanted is requested. Boundary knots and supplementary knots are computed by the program.)

POSITION OF KNOT (1)? .2

POSITION OF KNOT (2)? .4

POSITION OF KNOT (3)? .6

POSITION OF KNOT (4)? .8

(The location of the abscissa of each knot is requested. The knots must be entered in a strictly increasing sequence. Immediately following the input of all the knots the knot set is plotted along the x-axis.)

(A fixed knot least squares approximation is now computed. The resulting spline function is overlaid on the graphics display).

THE LEAST SQUARES ERROR OF THE PREVIOUS APPROXIMATION WAS 1.000000E 03

(The initial least squares error is arbitrarily set to 1000 since no previous approximation was done.)

THE LEAST SQUARES ERROR OF THE NEW APPROXIMATION IS 1.537876E-02

DO YOU WISH TO RETAIN THE APPROXIMATION RESULTS? NO

(A message asking whether or not to keep a hardcopy of the present approximation set is printed.)

DO YOU WISH TO VARY THE KNOT SET? YES

(A message asking whether or not to reposition any of the knots is printed.)

KNOT TO BE VARIED? 1

NEW POSITION FOR KNOT (1)? .3

DO YOU WISH TO VARY ANOTHER KNOT? YES

KNOT TO BE VARIED? 4

NEW POSITION FOR KNOT (4)? .7

DO YOU WISH TO VARY ANOTHER KNOT? NO

(A fixed knot least squares approximation is computed using the new knot set. The resulting least squares error information is printed.)

THE LEAST SQUARES ERROR OF THE PREVIOUS APPROXIMATION WAS 1.537876E-03

THE LEAST SQUARES ERROR OF THE NEW APPROXIMATION IS 9.576172E-03

DO YOU WISH TO SEE THE NEW APPROXIMATION? YES

(If the answer to this question is affirmative the new knot set and spline curve replaces the one currently being displayed on the graphics terminal.)

DO YOU WISH TO CONTINUE WITH THE NEW APPROXIMATION? YES

(The values of the new approximation are transferred to the proper arrays.)

DO YOU WISH TO RETAIN THE APPROXIMATION RESULTS? YES

TITLE FOR PLOT? 'TEST DATA'

(A title is printed on each graph and printout. These are also labelled with a number which is the number of hardcopies obtained thus far during the run. This method enables corresponding plots and printouts from a particular run to be uniquely identified.)

DO YOU WISH TO VARY THE KNOT SET? NO

DO YOU WISH TO OPTIMIZE THE KNOT SET? NO

(A request is printed as to whether or not to optimize the current knot set. WARNING: optimization is time-consuming and expensive. It is best done non-interactively.)

DO YOU WISH TO RESTART WITH A NEW KNOT SET? NO

(A request is issued whether or not to restart with an expanded or contracted knot set. If the answer is affirmative the data set is replotted and the program restarts from the beginning.)

DO YOU WISH TO RESTART WITH A NEW DATA SET? NO

(If the answer is affirmative, the program reads another data set from unit 4 and returns to the beginning of the program).

PLOTTING WILL TAKE APPROX. 2 MINS 32 SECONDS

STOP 0

EXECUTION TERMINATED

$COPY -PRINT *PRINT*

$RUN PLOT:Q PAR=-PLOT

$SIG

C. Output

Approximation information corresponding to a plot is put in a file. The format of each approximation printout is as follows:

TITLE			NUMBER
ABSCISSAE	ORDINATES	FITTED ORDINATES	RESIDUALS

for one data set

THE LEAST SQUARES ERROR IS

KNOT LOCATION	POLYNOMIAL POWER	POLYNOMIAL COEFFICIENT

This prints the coefficients of every power for each piece-wise polynomial between each knot pair. Plots obtained for the plotter correspond to this output. They can be matched by the title and label number.

Restrictions

The number of data points must be less that 200 .

The degree of the spline msut be less than 3 .

The number of knots must be less that 42 .