SPLINES AND THEIR APPLICATIONS

STEVEN PRUESS
Department of Mathematical and Computer Sciences
Colorado School of Mines
Golden, Colorado 80401-1887

Abstract

This paper is the summary of a cursillo given from May 18 through May 29, 1987, at the Departamento de Matemáticas, Facultad de Ciencias, La Universidad Autónoma de México.

VÍNCULOS MATEMÁTICOS No. 169, 1989.

COMUNICACIONES INTERNAS: Publicaciones del Departamento de Matemáticas del la Facultad de Ciencias de la UNAM. Tel. 5-50-52-15 extensión 3912.

Contents

1 Introduction

I. Schoenberg, with his work starting in the 1940's, is considered the pioneer of splines, though some of the fundamental ideas go back to Euler. Initially piecewise polynomials were studied for their curve-fitting properties; in fact, the word spline was only used for a particular C^2 interpolatory piecewise cubic (given in Section 2.3 below). Later the definition was broadened to include any piecewise polynomial; more recently, some researchers use spline for any function that is piecewise-defined. Since all the standard functions can be considered piecewise defined with one piece, this last definition of spline includes every function!

Besides the applications of splines in curve-fitting, they have been very widely used in the finite element method to estimate solutions of ordinary and partial differential equations. Today splines are widely used in industry (both for design and in finite element codes), and are still the object of mathematical research (especially splines in several variables).

1.1 Polynomials and interpolation

Given a set of data $(x_i, f(x_i))$ for $i = 1, 2, \ldots, N$ with the $\{x_i\}$ distinct, there is exactly one polynomial $p_N(x)$ in $\mathcal{P}_N$ which interpolates this data, i.e.,

$$p_N(x_i) = f(x_i) \text{ for } i = 1, 2, \ldots, N. \tag{1.1}$$

The existence and uniqueness of this polynomial is well known and proofs can be found in almost any introductory numerical analysis text. Here the symbol $\mathcal{P}_N$ stands for the space of polynomials of *order* N:

$$\mathcal{P}_N = \{\text{polynomials of degree } \textit{strictly less than } N\}.$$

Even though $p_N(x)$ is determined uniquely for a given set of data, there are many ways of representing the interpolating polynomial.

Lagrange representation:

$$p_N(x) = \sum_{i=1}^{N} f(x_i) L_{i,N}(x)$$

where

$$L_{i,N}(x) = \sum_{j \neq i} \frac{x - x_j}{x_i - x_j} \ . \tag{1.2}$$

This is an example of a *cardinal* or *nodal* representation.

Monomial representation:

$$p_N(x) = c_1 + c_2 x + \cdots + c_N x^{N-1} \ .$$

This is the simplest form to use, unfortunately, the usual way of computing the coefficients $\{c_i\}$ is to set up and solve a linear system arising from the interpolation conditions (1.1). This system is often ill-conditioned and the algorithm (in general) requires more work than other methods.

Divided difference representation:

$$p_N(x) = p_{N-1}(x) + [x_1, x_2, \ldots, x_N]f * (x - x_1) \cdots (x - x_N) \tag{1.3}$$

where the $(N-1)$st divided difference operator $[x_1, \ldots, x_N]$ is given recursively by

$$[x_1, \ldots, x_N]f = \begin{cases} \dfrac{[x_2, \ldots, x_N]f - [x_1, \ldots, x_{N-1}]f}{x_N - x_1} & x_1 \neq x_N \\ \\ f^{(N-1)}(x_1)/(N-1)! & \text{all } x_i \text{ equal} \end{cases} .$$

The zeroth order divided difference is just $[x_1]f = f(x_1)$. It is advantageous to think of $[x_1, \ldots, x_N]f$ as the leading coefficient of the interpolating polynomial for f over $\{x_1, \ldots, x_N\}$.

1.2 Error theorems for polynomial approximation

The standard form of the error theorem for polynomial interpolation assumes f has N continuous derivatives (though the form involving the Nth divided difference of f over $x_1, \ldots, x_N, x$ is often useful).

Theorem 1 *If $f \in C^N[a,b]$ with $\{x_i\} \subset [a,b]$, then for any x in $[a,b]$ there exists a ξ_x in the smallest interval containing $x_1, \ldots, x_N$, and x such that*

$$f(x) - p_N(x) = f^{(N)}(\xi_x)(x-x_1)\cdots(x-x_N)/N! \tag{1.4}$$

Before discussing this result, it will be useful to consider the best approximation to $f(x)$ from $\mathcal{P}_N$. Let

$$||f||_{[a,b]} = \max_{x\in[a,b]} |f(x)| \ ,$$

then since $\mathcal{P}_N$ is finite dimensional, for any f in $C[a,b]$ there is a best approximation $P_N(x)$ in $\mathcal{P}_N$ to f, i.e.,

$$||f - P_N||_{[a,b]} = \min_{P\in\mathcal{P}_N} ||f-P||_{[a,b]} \ .$$

The standard error theorem for polynomial best approximations is due to Jackson (see deBoor[2], p. 33).

Theorem 2 *If $f \in C^k[a,b]$ for some $k < N-1$, then*

$$||f - P_N||_{[a,b]} \le c_k \left(\frac{b-a}{N-1}\right)^k ||f^{(k)}||_{[a,b]} \ .$$

Both error theorems indicate possible problems as $N \to \infty$ if derivatives of f grow rapidly with N (or k), but for fixed N (or k) convergence is guaranteed as $(b-a) \to 0$. This suggests the possible advantages of piecewise polynomial approximation for functions with rapidly growing derivatives.

2 Simple interpolatory splines

We will begin with the lowest order continuous interpolatory piecewise polynomial, the linear spline. For the remainder of the material assume

$$x_1 < x_2 < \cdots < x_N \ .$$

It will also be convenient to set

$$h_i = x_{i+1} - x_i, \text{ and}$$
$$h = \max_i h_i \ .$$

2.1 Linear spline

On each (x_i, x_{i+1}) let

$$\begin{aligned} s(x) &= f(x_i)\frac{x-x_{i+1}}{x_i - x_{i+1}} + f(x_{i+1})\frac{x-x_i}{x_{i+1}-x_i} \\ &= f(x_i) + [x_i, x_{i+1}]f * (x - x_i) \end{aligned}$$

The first formula is a Lagrange representation, while the second is a divided difference (or a shifted monomial) representation. An error bound for $s(x)$ is a simple consequence of (1.4).

$$\begin{aligned} ||f-s||_{[x_i,x_{i+1}]} &\le ||f''||_{[x_i,x_{i+1}]}||(x-x_i)(x-x_{i+1})||_{[x_i,x_{i+1}]}/2! \\ &\le ||f''||_{[x_i,x_{i+1}]}h_i^2/8 \end{aligned}$$

or by maximizing over i

$$||f-s||_{[a,b]} \le ||f''||_{[a,b]}h^2/8 \ .$$

Clearly convergence is guaranteed as $h \to 0$ (though not at a particularly rapid rate). Jackson's Theorem yields the rate for f less smooth. The main disadvantage of linear splines is that s' is discontinuous.

It is also simple to study the error in the best approximation to f from $\mathcal{S}$, the space of continuous linear splines with breakpoints at $\{x_i\}$. Define the operator M from $C[x_1, x_N] \to \mathcal{S}$ by

$$Mf = \text{ linear spline interpolant to } f \text{ over } \{x_i\}$$

then

$$\begin{aligned} ||Mf||_{[x_i,x_{i+1}]} &= \max_{[x_i,x_{i+1}]} |(f(x_i)(x_{i+1}-x) + f(x_{i+1})(x-x_i))/h_i| \\ &\le ||f||_{[x_i,x_{i+1}]} |(x_{i+1}-x)/h_i + (x-x_i)/h_i| \\ &= ||f||_{[x_i,x_{i+1}]} \end{aligned}$$

so $||M||_{[x_i,x_{i+1}]} \le 1$ (in fact equality holds since M reproduces lines). Hence, on all of $[a,b]$ (omitting the $[a,b]$ subscript on the norm) $||M|| = 1$. Let ℓ be the *best* approximation to f in $\mathcal{S}$, i.e.,

$$||f - \ell|| = \min_{s \in \mathcal{S}} ||f - s|| \; .$$

Then

$$\begin{aligned} ||f-\ell|| &\le ||f - Mf|| = ||f - Mf + M\ell - \ell|| \\ &= ||(1-M)(f-\ell)|| \\ &\le (1 + ||M||)||f-\ell|| = 2||f-\ell|| \; . \end{aligned}$$

The conclusion is that the considerable effort required to compute the best approximation ℓ is probably not worthwhile, as the trivial interpolant Mf has at most twice the error.

2.2 C^1 Hermite cubic spline

Given $f(x_i)$ and $f'(x_i)$ for $i = 1, 2, \ldots, N$ there is a unique piecewise cubic function $H(x)$ such that

$$H(x_i) = f(x_i) \text{ and } H'(x_i) = f'(x_i) \text{ for } i = 1, 2, \ldots, N.$$

Theory and formulas follow immediately from observing that on each $[x_i, x_{i+1}]$, $H(x)$ is in $\mathcal{P}_4$ and matches the four values $f(x_i)$, $f(x_{i+1})$, $f'(x_i)$, and $f'(x_{i+1})$. Hence, the divided difference representation (1.3) holds on each piece (the Lagrange representation can be generalized too), and (1.4) says

$$||f - H||_{[a,b]} \le (h^4/384)||f^{(4)}||_[a,b] \; .$$

The main disadvantages are the need for $f'(x_i)$ values (though approximations can be used), and in some applications the discontinuity of H''.

The fact that the formulas and theory for this cubic spline and the linear spline were so simple is no accident. It follows from the fact that the splines were completely characterized by *local* conditions; the polynomial formula in $[x_i, x_{i+1}]$ was completely determined by data at x_i and x_{i+1}. Equivalently, changing one data value means formulas on at most two polynomial pieces must be changed.

Is it possible to construct a smoother interpolatory spline using cubic pieces? The answer, of course, is yes and is the classical cubic interpolatory spline. Given $N-1$ pieces, a cubic spline must have $4N-4$ parameters. On each interval there are two interpolation conditions (one at each end) or $2N-2$ conditions. The remaining $2N-2$ degrees of freedom can be used for continuity conditions. Continuity of s' results in $N-2$ conditions (continuity across each interior x_i), and similarly forcing continuity of s'' requires another $2N-2$ conditions. There are still two degrees of freedom left, not enough to require s''' to be smooth. Hence C^2 is the smoothest we can make the piecewise cubic (and still allow different cubics on different pieces).

2.3 C^2 cubic interpolatory spline

The above continuity and interpolation conditions are such that the solution $s(x)$ cannot be written in closed form, but a tridiagonal system needs to be solved to produce formulas for $s(x)$ (for details see deBoor [2, pp.54-57]). This means $s(x)$ is <u>not</u> local, if one data value is changed then the entire $s(x)$

must be recomputed (though the biggest change in $s(x)$ occurs near where the data changed). The theory is also more difficult, but it can be shown that

$$||f-s||_{[x_1,x_N]} \le (5/384)h^4||f^{4)}||_{[x_1,x_N]}$$

for f in $C^4[x_1,x_N]$ (with a certain type of end condition). One of the attractive features of the smooth cubic spline is its "minimum curvature property". The average curvature on $[x_1,x_N]$ of a smooth function $w(x)$ is defined to be

$$I(w) = \int_{x_1}^{x_N} [w''(x)]^2 \, dx \, .$$

(Actually this is only a linearized model of the actual curvature and is accurate only if w'' is small.) The following theorem was known to Euler.

Theorem 3 *Of all $C^2[x_1,x_N]$ functions $w(x)$ which interpolate $(x_i,f(x_i))$ for $1 \le i \le N$, $s(x)$ satisfying the end conditions $s''(x_1) = s''(x_N) = 0$ (the so-called natural spline) minimizes the average curvature $I(w)$.*

Proof. deBoor [2, p.66].

This is an example of a variational characterization of a spline (as the minimum of some functional).

2.4 Generalizations

$H(x)$ is a *Hermite spline* interpolant of order $2k$ if
 (i) H is in $\mathcal{P}_{2k}$ on each (x_i,x_{i+1}),
 (ii) H is in $C^{k-1}[x_1,x_N]$,
 (iii) $H^{(j-1)}(x_i) = f^{(j-1)}(x_i) \qquad 1 \le j \le k, 1 \le i \le N$.
Algorithms and theory are relatively easy for $H(x)$ because it is a "local" approximation.

If $S(x)$ is a general spline interpolant of order $2k$, how smooth can it be? There are $2k(N-1)$ free coefficients in $S(x)$, $2(N-1)$ of these are determined by interpolation conditions. The remaining $2(k-1)(N-1)$ degrees of freedom can be used for smoothness:

$$S^{(j-1)}(x_i^+) = S^{(j-1)}(x_i^-) \qquad 2 \le i \le N-1$$

for $j = 2,3,\ldots,2k-1$. Hence, the *smooth spline* interpolant of order $2k$ is in $C^{2k-2}[x_1,x_N]$; this smooth spline is not unique as $2(k-1)$ end conditions can be applied. For $k > 1$ these splines are not "local" so algorithms and theory are a little more complicated than in the Hermite case.

$S(x)$ is a *natural spline* interpolant of order $2k$ if
 (i) S is in $\mathcal{P}_{2k}$ on each (x_i,x_{i+1}),
 (ii) $S \in C^{2k-2}[x_1,x_N]$,
 (iii) $S(x_i) = f(x_i)$ for $i = 1,2,\ldots,N$,
 (iv) $S^{(j)}(x_1) = S^{(j)}(x_N) = 0$ for $j = k, k+1, \ldots, 2k-2$.
The variational characterization of this spline is

Theorem 4 *Of all $C^{2k-2}[x_1,x_N]$ functions w which interpolate $(x_i,f(x_i))$ for $1 \le i \le N$, $S(x)$ minimizes*

$$I(w) = \int_{x_1}^{x_N} [w^{(2k-2)}]^2 \, dx \, .$$

This is a good place to generalize to non-polynomial splines. Let L be some kth order differential operator (subject to some restrictions!), then the function $s(x)$ which minimizes

$$I(w) = \int_{x_1}^{x_N} (Lw)^2 \, dx$$

subject to the interpolation conditions $w(x_i) = f(x_i)$ for $1 \le i \le N$, satisfies
 (i) $L * Ls = 0$ in each (x_i,x_{i+1}),

(ii) $s \in C^{2k-2}[x_1, x_N]$,
(iii) $s(x_i) = f(x_i)$ for $i = 1, 2, \ldots, N$,
(iv) $2(k-1)$ "natural" end conditions.
For a reference see Schultz and Varga [23]. The standard examples of these *L-splines* are (with D signifying d/dx)
(1) $L = D^2$ (the cubic spline)
(2) $L = D$ (the linear spline)
(3) $L = D^2 - \tau D$ (exponential spline "in tension").

3 The B-spline basis

Notation: let $\mathcal{P}_{k,\underline{\xi}}$ be the space of k-th order piecewise polynomials with breakpoints $(\xi_1, \ldots, \xi_{\ell+1})$, i.e., $s \in \mathcal{P}_{k,\underline{\xi}}$ if and only if $s \in \mathcal{P}_k$ on each (ξ_i, ξ_{i+1}). Clearly, the dimension of $\mathcal{P}_{k,\underline{\xi}}$ is ℓk. To require smoothness of s we need another vector, called an *incidence vector*, $\underline{\nu} = (\nu_2, \ldots, \nu_\ell)$. Then $s \in \mathcal{P}_{k,\underline{\xi},\underline{\nu}}$ if and only if $s \in \mathcal{P}_{k,\underline{\xi}}$ and

$$s^{(j-1)}(\xi_i^+) = s^{(j-1)}(\xi_i^-) \qquad j = 1, 2, \ldots, \nu_i \ .$$

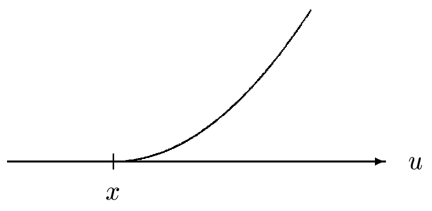
Examples:
(i) $\mathcal{P}_{4,\underline{\xi},\underline{\nu}}$ with $\underline{\nu} = (2, 2, \ldots, 2)$ is the space of cubic Hermite splines;
(ii) $\mathcal{P}_{4,\underline{\xi},\underline{\nu}}$ with $\underline{\nu} = (3, 3, \ldots, 3)$ is the space of smooth cubic splines;
(iii) $\mathcal{P}_{4,\underline{\xi},\underline{\nu}} = \mathcal{P}_{k,\underline{\xi}}$ if $\underline{\nu} = (0, 0, \ldots, 0)$.
Note, in general the dimension of $\mathcal{P}_{k,\underline{\xi},\underline{\nu}}$ is $n := \ell k - \sum_{i=2}^{\ell} \nu_i$. We seek a basis for $\mathcal{P}_{k,\underline{\xi},\underline{\nu}}$ consisting of n linearly independent functions. These should possess the following desirable properties:
(1) the existence of an efficient computational algorithm,
(2) well-conditioning,
(3) applications should have efficient, stable algorithms.

An excellent choice (though not the only one) is given by the B-splines $\{B_{i,k,\underline{t}}\}$. Given a set of knots $\underline{t} = (t_1, \ldots, t_{n+k})$ where $t_i \le t_{i+1}\ \forall i$, and $t_i < t_{i+k}\ \forall i$, define

$$B_{i,k,\underline{t}}(x) = (t_{i+k} - t_i)[t_i, \ldots, t_{i+k}](\cdot - x)_+^{k-1} \ .$$

Remarks:
(i) Some or all of the subscripts of $B(x)$ are often omitted if it is clear from the context what the correct values should be.
(ii) The $+$ subscript means "the positive part", i.e.,

$$(u - x)_+^{k-1} := \begin{cases} (u-x)^{k-1} & u \ge x \\ 0 & u \le x \end{cases} \ .$$



Example: $k = 1$, $t_1 < t_2 < \cdots < t_{n+1}$

$$\begin{aligned} B_i(x) &= (t_{i+1} - t_i)[t_i, t_{i+1}](\cdot - x)_+^0 \\ &= (t_{i+1} - x)_+^0 - (t_i - x)_+^0 \\ &= \begin{cases} 0 & x < t_i \\ 1 & t_i < x < t_{i+1} \\ 0 & t_{i+1} < x \end{cases} . \end{aligned}$$

3.1 Properties of the B-splines

(1) $B_i(x) = 0$ if $x \notin [t_i, t_{i+k}]$ (small support),
(2) $B_i(x) \geq 0$ for every x,
(3) $\sum_i B_i(x) = 1$ for every x (with a suitable convention at points of discontinuity),
(4) for a given interval $[t_j, t_{j+1}]$, at most k B-splines are nonzero there, namely, B_{j-k+1}, B_{j-k+2}, ..., B_j.
(5) If ξ occurs in $t_i, t_{i+1}, \ldots, t_{i+k}$ with multiplicity μ, then

$$B_i^{(j-1)}(\xi^-) = B_i^{(j-1)}(\xi^+) \quad j = 1, 2, \ldots, k - \mu$$

but

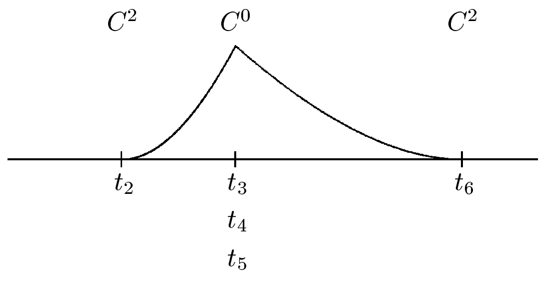
$$B_i^{(k-\mu)}(\xi^-) \neq B_i^{(k-\mu)}(\xi^+) .$$

(6) (Variation diminishing property) Let $S^-\underline{\alpha}$ be the number of sign changes in $\alpha_1, \ldots, \alpha_n$ and $S^- f$ be the number of sign changes in the spline $f(x) = \sum \alpha_i B_i(x)$, then

$$S^- f \leq S^- \underline{\alpha} .$$

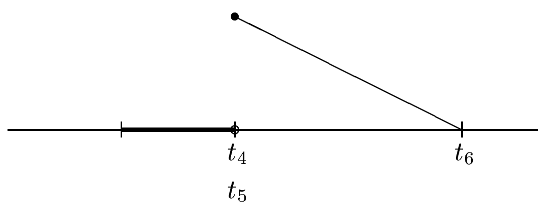
Examples: assume the knots are $t_1 = t_2 < t_3 = t_4 = t_5 < t_6$. Then

$$B_{2,4,\underline{t}}(x) = (t_6 - t_2)[t_2, t_3, t_4, t_5, t_6](\cdot - x)_+^3$$



and

$$B_{4,2,\underline{t}}(x) = (t_6 - t_4)[t_4, t_5, t_6](\cdot - x)_+^1$$



Theorem 5 *Given $\underline{\xi} = (\xi_1, \ldots, \xi_{\ell+1})$ and $\underline{\nu} = (\nu_2, \ldots, \nu_\ell)$ with $0 \le \nu_i < k\ \forall i$, define $\underline{t}$ by*

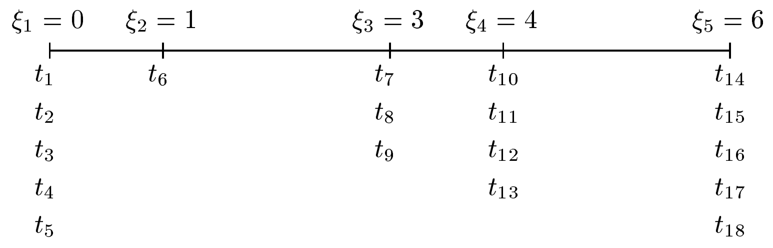
(1) $t_1 \le t_2 \le \cdots \le t_k \le \xi_1$, $\xi_{\ell+1} \le t_{n+1} \le \cdots \le t_{n+k}$,

(2) for $i = 2, 3, \ldots, \ell$ the number ξ_i occurs exactly $k - \nu_i$ times in $\underline{t}$.

Then, each $B_{i,k,\underline{t}}$ is in $\mathcal{P}_{k,\underline{\xi},\underline{t}}$ and $\{B_{i,k,\underline{t}}\}$ $(i = 1, \ldots, n)$ is linearly independent on the interval $[t_k, t_{n+1}]$. Hence, $\{B_{i,k,\underline{t}}\}$ is a basis for $\mathcal{P}_{k,\underline{\xi},\underline{t}}$.

Remark: for a basis over some given interval $[a, b]$, the usual choice is $a = \xi_1 = t_1 = \cdots = t_k$ and $b = \xi_{\ell+1} = t_{n+1} = \cdots = t_{n+k}$.

Example: if you want a piecewise fourth degree polynomial with breakpoints only at 0,1,3,4,6 and to be C^3 at 1, C^1 at 3, C at 4, then


$\xi_1 = 0$
$\xi_2 = 1$
$\xi_3 = 3$
$\xi_4 = 4$
$\xi_5 = 6$
t_1 t_2 t_3 t_4 t_5
t_6
t_7 t_8 t_9
t_{10} t_{11} t_{12} t_{13}
t_{14} t_{15} t_{16} t_{17} t_{18}

so $k = 5$, $\ell = 4$, and $n = 13$.

3.2 Recursive algorithms

$\boxed{1}$ Let $M_{ik}(t) = B_{ik}(x)/(t_{i+k} - t_i)$, then we have

$$M_{ik}(x) = \frac{x - t_i}{t_{i+k} - t_i} M_{i,k-1}(x) + \frac{t_{i+k} - x}{t_{i+k} - t_i} M_{i+1,k-1}(x) \ ,$$

a convex combination of non-negative quantities. Initially,

$$M_{i1}(x) = \begin{cases} 1/(t_{i+k} - t_i) & t_i \le x < t_{i+1} \\ 0 & \text{otherwise} \end{cases} .$$

In terms of B_i

$$B_{ik}(x) = \frac{x - t_i}{t_{i+k-1} - t_i} B_{i,k-1}(x) + \frac{t_{i+k} - x}{t_{i+k} - t_{i+1}} B_{i+1,k-1}(x) \ .$$

$\boxed{2}$ $B'_{ik}(x) = -(k-1)\{M_{i+1,k-1}(x) - M_{i,k-1}(x)\}$ and this is easily extended to higher derivative formulas.

$\boxed{3}$ If $s(x) = \sum \alpha_i B_{ik}(x)$ then for $j = 1, 2, \ldots, k$

$$s^{(j-1)}(x) = \sum \alpha_i^{(j)} B_{i,k-j+1}(x)$$

where

$$\alpha_i^{(j)} = \begin{cases} \alpha_i & j = 1 \\ \dfrac{\alpha_i^{(j-1)} - \alpha_{i-1}^{(j-1)}}{t_{i+k-j+1} - t_i}(k - j) & j > 1 \end{cases} .$$

4 Applications to Curve Fitting

The problem of *shape preserving* interpolation is to find a smooth interpolant which is "visually pleasing"; this usually means that the interpolant has the same monotonicity and/or convexity properties as the underlying data. Mathematically this can be interpreted many ways. Let $f(x)$be the function the data came from and $F(x)$ the approximation to be constructed. Certainly we want $F'(x_i) \geq 0$ if $f(x_{i-1}) \leq f(x_i) \leq f(x_{i+1})$, and maybe we want $F'(x) \geq 0$ in $[x_i, x_{i+1}]$ when $f(x_i) \leq f(x_{i+1})$. Similarly we certainly want $F''(x_i) \geq 0$if $[x_{i-1}, x_i, x_{i+1}]f \geq 0$, and maybe we want $F''(x) \geq 0$ when both $[x_{i-1}, x_i, x_{i+1}]f$ and $[x_i, x_{i+1}, x_{i+2}]f$ are non-negative.

In general, interpolating polynomials of even moderate degree oscillate too much to be shape preserving. Even the standard C^2 cubic interpolatory spline with its minimum curvature property can fail to be shape preserving on problems with large curvature in a few isolated regions (the spline has smaller *average* curvature but it can oscillate in order to achieve this).

Because of the many important applications to computer-aided-design this problem has received considerable attention in the literature.

$\boxed{\text{Method 1}}$ spline in tension (Schweikert [24])

Locally $F^{(4)} - \tau^2 F'' = 0$ for some parameter τ, hence, as $\tau \to 0$, F approaches the C^2 cubic interpolatory spline, and as $\tau \to \infty$, F approaches the linear spline. The latter is not smooth, but is shape preserving, so it is hoped that for some finite nonzero τ we can have both. An improvement (studied by Späth [25] and Pruess [19]) is to let τ be different in different intervals. The main criticisms of this method are the expense in evaluating exponential functions, the difficulty in choosing parameters automatically (but see Renka [22]), and the fact that the graphs often appear to have flat spots. The latter implies that even though monotonicity and convexity may be mathematically preserved, the resulting curve still may not be visually pleasing.

$\boxed{\text{Method 2}}$ cubic spline in tension or taut spline

These were studied by Nielson [16], deBoor [2, Chapter 16], and Pruess [20] in varying degrees of generality. The idea is to introduce extra knots if the standard C^2 cubic interpolatory spline is inadequate. It is not difficult to add extra knots to match convexity, but forcing monotonicity is not as simple. The reason for this is that none of the examples considered are local, so all pieces of $F(x)$ are coupled. This is a severe disadvantage in computer-aided-design.
The remaining methods are all local which leads to simpler algorithms (and mathematics!).

$\boxed{\text{Method 3}}$ C^1 cubic Hermite

This was introduced by Ferguson [12], and rediscovered by Fritsch and Carlson [13]. On each $[x_i, x_{i+1}]$ $F(x)$ is a piecewise cubic satisfying

$$F(x_i) = f(x_i) \qquad F'(x_i) = s'_i \qquad F(x_{i+1}) = f(x_{i+1}) \qquad F'(x_{i+1}) = s'_{i+1}$$

with the slopes to be chosen to control shape. Let D_i be the secant slope $[x_i, x_{i+1}]f$, $\alpha_i = s'_i/D_i$, and $\beta_i = s_{i+1}/D_i$. To force monotonicity in $[x_i, x_{i+1}]$ (so $F'(x)$ has the same sign as D_i) it is sufficient to have

$$0 \leq \alpha_i \leq 3 \quad \text{and} \quad 0 \leq \beta_i \leq 3.$$

Fritsch and Carlson (and others) have algorithms for automatically choosing α_i and β_i, and hence, $\{s'_i\}$. Usually, $||f - F|| = O(h^2)$ so some sacrifice is made in accuracy.

$\boxed{\text{Method 4}}$ C^2 quintic Hermite (Hyman [14])

Here $k = 6$ and for all i $F(x_i) = f(x_i)$, $F'(x_i) = s'_i$, and $F''(x_i) = s''_i$ where $\{s'_i\}$ and $\{s''_i\}$ are to be chosen to preserve the shape. The resulting inequalities are quite complicated because of the four dimensional geometry on each piece.

$\boxed{\text{Method 5}}$ C^2 piecewise cubic (Pruess [21])

Introduce two extra break points ξ_i and η_i in each $[x_i, x_{i+1}]$, then there are enough degrees of freedom to have a local scheme which is in $C^2[x_1, x_N]$. There are 12 coefficients for each $[x_i, x_{i+1}]$; the corresponding 12 conditions are:

$$F(x_i) = f(x_i) \qquad F'(x_i) = s'_i \qquad F''(x_i) = s''_i,$$

$$F(x_{i+1}) = f(x_{i+1}) \qquad F'(x_{i+1}) = s'_{i+1} \qquad F''(x_{i+1}) = s''_{i+1},$$

and for $j = 1, 2, 3$

$$F^{(j-1)}(\xi_i^-) = F^{(j-1)}(\xi_i^+) \qquad F^{(j-1)}(\eta_i^-) = F^{(j-1)}(\eta_i^+).$$

The linear parameters $\{s'_i\}$ and $\{s''_i\}$, and the nonlinear ones $\{\xi_i, \eta_i\}$ can be used to control shape. To match convexity requires two inequalities (for each i) giving the correct sign of $s''(\xi_i)$ and $s''(\eta_i)$. Monotonicity is more difficult. As yet, there is no automatic algorithm for choosing these parameters, but research is continuing.

$\boxed{\text{Method 6}}$ Variation diminishing splines (<u>not</u> interpolatory)

Given knots $\{t_i\}$ $1 \le i \le n+k$ for some $k \ge 2$, $n \ge 1$, and a function $f(x)$ defined on $[t_k, t_{n+1}]$, the k-th order *variation diminishing spline* approximation to f is

$$(V_k f)(x) = \sum_{j=1}^{n} f(t_j^*) B_{jk}(x)$$

where $t_j^* = (t_{j+1} + \cdots + t_{j+k-1})/(k-1)$.
Properties:
(i) the value of $V_k f$ in $[t_i, t_{i+1}]$ depends only on $f(t_{j-k+1}^*), \ldots, f(t_i^*)$. (local scheme)
(ii) If f is a straight line, then so is $V_k f$.
(iii) $||f - V_k f||_{[t_k, t_{n+1}]} = O(h^2)$.
(iv) $V_k f$ is variation diminishing (see section 3.1):

$$S^-(V_k f) \le S^-(f(t_i^*)) \le S^-(f)\ .$$

(v) $V_k f$ is convex if f is.
(vi) $V_k f$ is monotone if f is.
(vii) If $t_i = t_{i+1} = \cdots = t_{i+k-1}$ then $V_k f \equiv f$ there (so interpolation is possible if smoothness is sacrificed).
To construct an algorithm based on the variation diminishing splines is not simple unless the data $\{x_i\}$ is equally spaced. For example (from [11]), let $x_i = a + (i-1)h$, $h = (b-a)/(N-1)$, choose $k = 3$ (quadratic spline), $n = N$,

$$\begin{aligned} &t_1 = t_2 = a - h/2, \\ &t_i = a + (i - 5/2)h \qquad 3 \le i \le N+1 \\ &t_{N+2} = t_{N+3} = b + h/2\ . \end{aligned}$$

Then, $t_j^* = (t_{j+1} + t_{j+2})/2 = a + (j-1)h = x_j$ and

$$(V_3 f)(x) = \sum_{j=1}^{n} f(x_j) B_{j3}(x).$$

$\boxed{\text{Method 7}}$ Bernstein-Bézier and parametric B-spline curves

These will be covered in the next section in arbitrary dimensions.

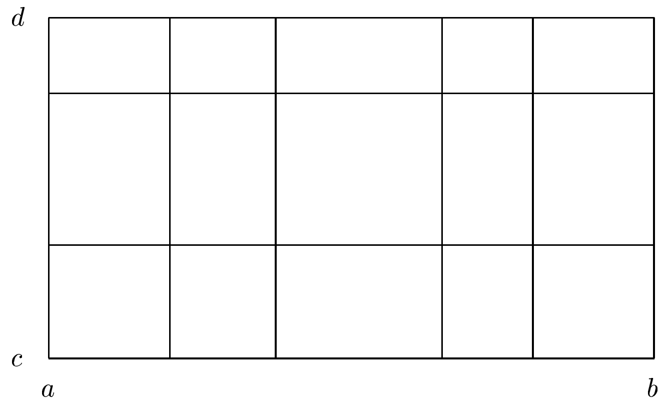
5 Approximation in higher dimensions.

Tensor product approximations

These are the simplest of the standard schemes, both mathematically and computationally, and are straightforward to apply in any dimension. In two dimensions, on the rectangle $a \le x \le b$, $c \le y \le d$, choose knots $\{t_i\}$ $1 \le i \le n_x$ for $[a,b]$, and $\{\tau_j\}$ $1 \le j \le n_y$ for $[c,d]$, and use

$$s(x,y) = \sum_{i=1}^{n_x}\sum_{j=1}^{n_y} \alpha_{ij} B_{i,k_x,\underline{t}}(x) B_{j,k_y,\underline{\tau}}(y) \ .$$

The coefficients $\{\alpha_{ij}\}$ can be chosen by interpolation or least squares or whatever. In three dimensions there is an additional sum, α has an additional subscript, and there is an additional $B_{???}(z)$ factor. Clearly, this extends to arbitrary dimensions.
Examples: in two dimensions grids have the form



(1) bilinear splines ($k_x = k_y = 2$)

On each subrectangle $s(x,y)$ is a linear combination of 1, x, y, and xy. This is more than first degree, but is not a full quadratic (because x^2 and y^2 are missing). The name <u>bi</u>linear is given because $s(x,y)$ is linear in each variable when the other is held constant. Note that $s(x,y)$ is continuous on $[a,b]\times[c,d]$ but s_x and s_y are not.

(2) bicubic splines ($k_x = k_y = 4$)

On each subrectangle $s(x,y)$ is a linear combination of 1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3, x^3y, x^2y^2, xy^3, x^3y^2, x^2y^3, and x^3y^3. This is more than a cubic but far less than a general sixth degree polynomial. It is cubic in each variable when the other is constant. Globally s, s_x, s_y, s_{xx}, s_{xy}, s_{yy}, s_{xxy}, s_{xyy}, and s_{xxyy} are all continuous but, in general, any partial derivative

$$\frac{\partial^{i+j}s}{\partial x^i \partial y^j}$$

with $i \le 3$ or $j \le 3$ is discontinuous.

Blending methods

Since one dimensional formulas are usually easier to generate, a simple way of constructing higher dimensional formulas is to blend together one-dimensional ones. Again, this is easiest to see in two dimensions. Given curves $f(x_i, y)$ $1 \le i \le M$, and $f(x, y_j)$ $1 \le j \le N$, define

$$P_1 f = \sum_{i=1}^{M} f(x_i, y) L_{iM}(x) \qquad L_{iM}(x) = \prod_{k\ne i} \frac{x - x_k}{x_i - x_k}$$

$$P_2 f = \sum_{j=1}^{N} f(x, y_j) L_{jN}(y) \qquad L_{jN}(y) = \prod_{k\ne j} \frac{y - y_k}{y_j - y_k}$$

and

$$F(x,y) = (P_1 \oplus P_2)f = P_1 f + P_2 f - P_1 P_2 f \qquad \text{(Boolean sum)}.$$

When the Lagrange polynomials are used to blend the resulting F is only continuous, but Hermite blending functions can also be used if greater smoothness is desired. (For the latter case you also need to know f_x and f_y, etc.) For irregular domains it is possible to blend over triangular pieces rather than rectangular ones.

Parametric representations

Given vectors $\underline{F}_i$ $1 \le i \le n$ in $\mathcal{R}^N$, define
(1) a Bernstein-Bézier-Casteljau curve

$$\underline{F}(t) = \sum_{j=1}^{n} \underline{F}_j b_j(t)$$

(2) a B-spline curve

$$\underline{F}(t) = \sum_{j=1}^{n} \underline{F}_j B_{jk}(t)$$

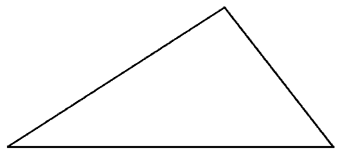
where t is a variable parameterizing the curve, and in (2) $\{B_{jk}\}$ is a set of B-splines for some k and choice of knots, and in (1)

$$b_j(x) = \frac{n!}{j!(n-j)!} x^j (1-x)^{n-j} \qquad 0 \le x \le 1 \;.$$

These approximations are both variation diminishing, the curve $\underline{F}(t)$ lies in the convex hull of $\{\underline{F}_i\}$. In general, neither interpolates, though the B-spline curve will if multiple knots are used (smoothness then suffers). The B-spline curve is more general (because it is so easy to control smoothness) and perhaps a little easier to implement. The Bézier curve can be done piecewise too, in order to keep the degree of the polynomials from getting too big.

Splines on triangulated domains

For more general domains than rectangles (in two dimensions), generalizations of splines are far more complex. First approximate the boundary (if necessary) by a polygon, then triangulate the resulting polygonal region (see Joe[15] for a triangulation algorithm). On each subtriangle approximate by polynomials. For a generic triangle



there are many local schemes.

C^0 continuity globally

Let $F(x,y) = \alpha + \beta x + \gamma y$ where α, β, and γ are chosen so that F interpolates data at the vertices. Then, along each edge F is a straight line and is uniquely determined by the adjoining vertex values. This guarantees the global continuity.

Let $F(x,y) = \alpha + \beta x + \gamma y + \delta x^2 + \epsilon xy + \zeta y^2$ (a full quadratic) where the six parameters are chosen so that F interpolates data at vertices and edge midpoints. Along each edge F is a quadratic which is uniquely determined by three edge values, guaranteeing global continuity.

C^1 continuity globally

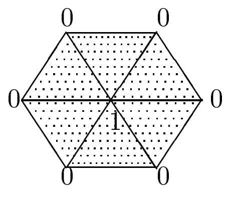
By subdividing the generic triangle it is possible to generate a C^1 surface. This requires values of f, f_x, and f_y at vertices, and f_n at edge midpoints. Clough and Tucker (see [26] for details) did this with cubics and 3 subtriangles, Powell and Sabin [18] used quadratics and 12 subtriangles.

More recently people have used global schemes which do not require subtriangles, but generate large, sparse linear systems to be solved. A good general reference is Barnhill and Böhm [1].

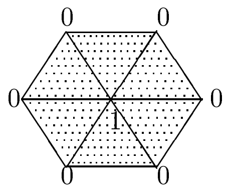
Can B-splines be generalized?

In the past 10 to 12 years many people have attempted to generalize B-splines to higher dimensions, e.g., deBoor, Höllig, Dahmen, Böhm, Hakopian, and Micchelli. The simplest cases (box splines) involve choosing a lattice of points (corresponding to the knots in one dimension) - the geometry of the lattice determines the smoothness of the splines. Notation, theory, and formulas are very complicated. For example, in two dimensions on a uniform triangular grid (equilateral triangles) there are the following cases with the lattice and values at vertices indicated.

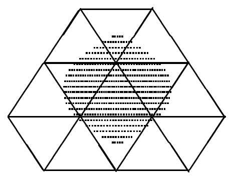
$k = 2$ (linear) globally C^0



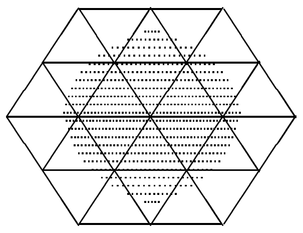
$k = 3$ (quadratic) globally C^0



$k = 4$ (cubic) globally C^1



$k = 5$ (quartic) globally C^2



The shaded area is the support. In general, global smoothness is C^ℓ where $\ell = int[2(k-2)/3]$. Unfortunately, these box splines may not be a basis for the k-th order piecewise polynomials corresponding to these grids. For example, the $k=4$ splines do not span, while the $k=3$ splines are linearly dependent. Some references are [1], [8], and [10].

6 Applications to linear operator equations

Only linear equations will be considered as typically some sort of Newton or quasi-Newton iteration is used to reduce a nonlinear problem to a sequence of linear ones. Consider the equation $Lu = f$ with L linear and f given.
Examples:

(1) $Lu = \dfrac{d^2u}{dx^2} + p(x)\dfrac{du}{dx} + q(x)u \qquad p,\ q$ given

(2) $Lu = \lambda u(x) + \int_a^b K(x,t)u(t)\,dt \qquad \lambda,\ K$ given

(3) $Lu = \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + c(x,y)u \qquad c$ given.

Finite element methods first partition the domain of the independent variable(s) into a finite number of pieces, called elements. Then u is approximated on each piece by simple functions, usually polynomials. In the end we again have splines.

Choose a basis for the desired spline space, this could be a B-spline basis but it doesn't have to be. In one dimension, approximate $u(x)$ by $\hat{u}(x) = \sum_j \alpha_j \phi_j(x)$ where $\{\phi_j\}$ is the basis. Then try to solve $L\hat{u} \approx f$ for the $\{\alpha_j\}$. To specify $\approx$, the operator L must be approximated in some manner. The standard methods are called weighted residual methods since they involve the residual $r = L\hat{u} - f$. Choose a set of linearly independent, continuous linear functionals $\{\mu_i\}$ and require

$$\mu_i(L\hat{u} - f) = 0 \qquad \forall i\ ; \tag{6.1}$$

there results the linear system

$$\sum_j \alpha_j(\mu_i L\phi_j) = \mu_i f \qquad \forall i\ .$$

There are two standard choices for the functionals:
(1) collocation
Choose a set of points $\{z_i\}$ in $[a,b]$ and set

$$\mu_i F := F(z_i)\ ; \square$$

(2) Ritz
Choose a set of functions $\{\psi_i\}$ and set

$$\mu_i F := \int_a^b F(t)\psi_i(t)\,dt\ . \tag{6.2}$$

Some possible ψ_i are ϕ_i, ϕ_i', ϕ_i'', or $L\phi_i$. The last choice gives a least squares method. $\square$

In general, the collocation method never produces symmetric coefficient matrices, the Ritz method does if L is self-adjoint, and the least squares method always does. Small support bases are an advantage in all cases (less so if L is an integral operator). For differential equations, the Ritz method is usually applied only to the "weak form" of the problem in which case it is known as Galerkin's method.

Example: $u'' + p(x)u' + q(x)u = f(x) \qquad a \le x \le b$

$$u(a) = u(b) = 0.$$

Write $\hat{u}(x) = \sum_j \alpha_j \phi_j(x)$ where it is assumed that $\phi_j(a) = \phi_j(b) = 0$ so $\hat{u}$ satisfies the boundary conditions (this can be done differently).

Collocation

$$\hat{u}''(z_i) + p(z_i)\hat{u}'(z_i) + q(z_i)\hat{u}(z_i) = f(z_i)$$

or

$$\sum_j \alpha_j \left(\phi_j'' + p\phi_j' + q\phi_j\right)\big|_{z_i} = f(z_i)\ .$$

Galerkin

To get the weak form of the equation multiply the differential equation by the "test function" ψ and integrate from a to b. An integration by parts then produces

$$-u'\psi\big|_a^b + \int_a^b (-u'\psi' + pu\psi + qu\psi)\, dx = \int_a^b f\psi\, dx\ .$$

Replace u by $\hat{u}$ and let $\psi = \phi_i$, then $-\hat{u}'\phi_i|_a^b = 0$ and

$$\int_a^b (-\hat{u}'\phi_i' + p\hat{u}'\phi_i + q\hat{u}\phi_i)\, dx = \int_a^b f\phi_i\, dx$$

or

$$\sum_j \alpha_j \int_a^b \left(-\phi_i'\phi_j' + p\phi_i\phi_j' + q\phi_i\phi_j\right) dx = \int_a^b f\phi_i\, dx\ .$$

Least squares

This is just (6.1) and (6.2) with $\psi_i = L\phi_i$, i.e.,

$$\sum_j \alpha_j \int_a^b (-\phi_i'' + p\phi_i' + q\phi_i)\left(-\phi_j'' + p\phi_j' + q\phi_j\right) dx = \int_a^b f\,(\phi_i'' + p\phi_i' + q\phi_i)\, dx\ .$$

Collocation algorithms for scalar ODE's

Let

$$L = \frac{d^m}{dx^m} + \sum_{j=1}^m c_j(x)\frac{d^{m-j}}{dx^{m-j}}$$

with boundary conditions

$$\sum_{j=1}^m \beta_{ij}u^{(j-1)}(a) = \gamma_i \qquad 1 \le i \le m'$$

$$\sum_{j=1}^m \beta_{ij}u^{(j-1)}(b) = \gamma_i \qquad m' < i \le m\ .$$

Assume there is a unique solution $u(x)$ for every $f(x)$ and $\underline{\gamma}$; equivalently, there is a Green's function $G(x,t)$ such that

$$u(x) = w(x) + \int_a^b G(x,t)f(t)\, dt \tag{6.3}$$

where $Lw = 0$ and w satisfies the boundary conditions.

If $\hat{u} \in \mathcal{P}_{m+k,\underline{\xi}} \cap C^p[a,b]$, then each ξ_i $(2 \le i \le \ell)$ has multiplicity $m+k-(p+1)$; thus, the dimension of the spline space is

$$n = m + k + \sum_{i=2}^{\ell}(m+k-p-1) = \ell(m+k-p-1) + p + 1\ .$$

After the m boundary conditions are imposed, there remain

$$n - m = \ell(m+k-p-1) + p + 1 - m$$

degrees of freedom to be determined by collocation. There are several standard choices:
(1) smooth splines
Take $p = m + k - 2$, then

$$n - m = \ell + p + 1 - m = \ell + k - 1 \, .$$

How should the collocation points be chosen? They probably should vary with $\underline{\xi}$, but the numbers don't come out even. In general, $||u - \hat{u}|| = O(h^k)$, not $O(h^{m+k})$ as we would expect given the order of the splines used. Can this be improved? This area has not received much attention for a while, but clearly has several unanswered questions. □
(2) Hermite splines.
Take $p = m - 1$ (the minimum for $L\hat{u}$ to make sense), then $n - m = \ell k$ so it is reasonable to choose k collocation points per interval. In general, choosing them equally spaced yields $||u - \hat{u}|| = O(h^k)$, where as in the previous case we expect $O(h^{m+k})$. What is happening? To analyze the error let $\hat{Q}$ be the map which takes a piecewise continuous function into the spline in $\mathcal{P}_{k,\underline{\xi}}$ which interpolates it at the k collocation points per interval. Then

$$\hat{Q}L\hat{u} = \hat{Q}f$$

and since $\hat{u}$ satisfies the same boundary conditions as u

$$\hat{u}(x) = w(x) + \int_a^b G(x,t) L\hat{u} \, dt \, .$$

Combining this with the earlier equations, we have

$$\begin{aligned} u(x) - \hat{u}(x) &= \int_a^b G(x,t)(f - L\hat{u}) \, dt \\ &= \int_a^b G(x,t)(I - \hat{Q})(f - L\hat{u}) \, dt \, . \end{aligned}$$

From this, we expect only $O(h^k)$ errors because of the range of $\hat{Q}$. In general, this cannot be improved; however, the expected $O(h^{m+k})$ can be attained with the right choice of collocation points. If the $G(x,t)$ factor were not present, we would have a standard quadrature problem where the optimal choice of interpolation points is the Gauss-Legendre points. It turns out that this set of points works even with the $G(x,t)$ factor there. It can be shown (deBoor and Swartz [9]) that
(i) $||u^{(j-1)} - \hat{u}^{(j-1)}|| \leq \text{constant } \cdot h^{m+k}$
(ii) $|u^{(j-1)}(\xi_i) - \hat{u}^{(j-1)}(\xi_i)| \leq \text{constant } \cdot h^{\max(2k, m+k+1-j)}$
for $1 \leq j \leq m$. The higher order convergence at the breakpoints is called "superconvergence". □

Implementation considerations

All implementations require the solution of a linear system; if a B-spline basis is used, the coefficient matrix can be ill-conditioned for irregular meshes (which arise naturally in singular perturbation problems). A better choice for the Hermite spline case is a local monomial representation as studied by Ascher, Pruess, and Russell [6], Paine and Russell [17], and Swartz [27], and implemented by Bader and Ascher [7]. There are several methods for automatically choosing the breakpoints $\{\xi_i\}$; for one, see Chapter 12 of deBoor[2].

Partial differential equations

These can get very complicated so we examine only two simple examples to give an idea of what is involved.

(1) A parabolic problem in one space dimension.
For $c(x) > 0$ consider

$$u_t = c(x)u_{xx} + f(x,t) \qquad a \leq x \leq b$$
$$u(a,t) = u(b,t) = 0$$
$$u(x,0) = G(x) \, .$$

Try $\hat{u}(x,t) = \sum_j \alpha_j(t)\phi_j(x)$ with $\{\phi_j\}$ a basis for some spline space and $\{\alpha_j(t)\}$ to be determined. A combination called method-of-lines/collocation produces

$$\sum_j \alpha_j(t)\phi_j(z_i) = c(z_i)\sum_j \alpha_j(t)\phi_j''(z_i) + f(z_i,t)$$

a linear system of ordinary differential equations for $\underline{\alpha}(t)$. Initial conditions come from

$$\sum_j \alpha_j(0)\phi_j(z_i) = G(z_i)$$

(this assumes $\phi_i(a) = \phi_(b) = 0\ \forall i$). In theory this can be solved with standard initial value solvers, but since the system is usually stiff, modern stiff solvers should be used. Variations are to use Galerkin's method in space, or finite differences in time. □

(2) An elliptic problem in two space dimensions.
For $p(x,y) > 0$ and $q(x,y) > 0$ consider

$$(p(x,y)u_x)_x + (q(x,y)u_y)_y + r(x,y)u = f(x,y) \text{ in some } \Omega$$

$$u \text{ given on the boundary of } \Omega.$$

If Ω is a rectangle, then try

$$\hat{u}(x,y) = \sum_{ij} \alpha_{ij}\phi_i(x)\phi_j(y)\ .$$

For a general region Ω, first approximate by a polygonal region, then triangulate, and finally try one of the spline approximations discussed at the end of section 5. Again, collocation or Galerkin may be used (in theory). In practice, the latter is more attractive because it has $\hat{u}$ in C^0 rather than C^1 which collocation requires. The extra smoothness is no problem for tensor product splines, but is more difficult to achieve on triangulated domains. □

One final comment about problems involving partial differential equations; frequently the difficulty is due to the geometry of the domain, not the operator itself. In such cases piecewise polynomial ϕ_i may not be the best choice. Ideally one should use the eigenfunctions of L, but this takes us away from splines (except in the broadest possible interpretation of the word spline!).

GENERAL REFERENCES

[1] R. Barnhill and W. Böhm, editors, *Surfaces in Computer Aided Design*, North Holland, Amsterdam, 1983.

[2] C. deBoor, *A Practical Guide to Splines*, Springer-Verlag, New York, 1978.

[3] M. Powell, *Approximation Theory and Methods*, Cambridge Univ. Press, Cambridge, 1981.

[4] I. Schoenberg, *Cardinal Spline Interpolation*, SIAM, Philadelphia, 1973.

[5] L.Schumaker, *Spline Functions: Basic Theory*, Wiley, New York, 1981.

OTHER REFERENCES

[6] U. Ascher, S. Pruess, and R. Russell, On spline basis selection for solving differential equations, SIAM J. Numer. Anal., 20 (1983), 121–147.

[7] G. Bader and U. Ascher, A new basis implementation for a mixed order boundary value ODE solver, SIAM J. Sci. and Stat. Comp., 8 (1987), 483–500.

[8] C. deBoor and K. Höllig, Bivariate box splines and smooth pp functions on a three-direction mesh, J. Comput. Appl. Math., 9 (1983), 13–28.

[9] C. deBoor and B. Swartz, Collocation at Gaussian Points, SIAM J. Numer. Anal., 10 (1973), 585–606.

[10] C. deBoor and R. deVore, Approximation by smooth multivariate splines, Trans. Amer. Math. Soc., 276 (1983), 775–788.

[11] W. Coughran Jr., E. Grosse, and D. Rose, Variation diminishing splines in simulation, SIAM J. Sci. and Stat. Comp., 7 (1986), 696–705.

[12] J. Ferguson and K. Miller, Characterization of shape in a class of third degree algebraic curves, TRW report 5322-3-5, 1969.

[13] F. Fritsch and R. Carlson, Monotone piecewise cubic interpolation, SIAM J. Numer. Anal., 17 (1980), 238–246.

[14] J. Hyman, R, Dougherty, A. Edelman, Positivity-, monotonicity-, or convexity-preserving cubic and quintic hermite interpolation, Math. Comp., 52 (1989), 471–494.

[15] B. Joe, Delauney triangular meshes in convex polygons, SIAM J. Sci. and Stat. Comp., 7 (1986), 514–539.

[16] G. Nielson, Some piecewise polynomial alternatives to splines under tension, in *Computer Aided Design*, Barnhill and Riesenfeld, eds., Academic Press, New York, 1974, 209–236.

[17] J. Paine and R. Russell, Conditioning of collocation matrices and discrete Green's functions, SIAM J. Numer. Anal., 23 (1986), 376–392.

[18] M. Powell and M. Sabin, Piecewise quadratic approximation on triangles, ACM Trans. on Math. Software, 3 (1977), 316–325.

[19] S. Pruess, Properties of splines in tension, J. of Approx. Theory, 17 (1976), 86–96.

[20] S. Pruess, Alternatives to the exponential spline in tension, Math. Comp., 33 (1979), 1273–1281.

[21] S. Pruess, Shape preserving C^2 cubic spline interpolation, IMA J. Numer. Anal., 13 (1993), 493–507.

[22] R. Renka, Interpolatory tension splines with automatic selection of tension factors, SIAM J. Sci. and Stat. Comp., 8 (1987), 393–415.

[23] M. Schultz and R. Varga, L-splines, Numer. Math., 10 (1967), 319–345.

[24] D. Schweikert, An interpolation curve using splines in tension, J. Math. Phys., 45 (1966), 312–317.

[25] H. Späth, Exponential spline interpolation, Computing, 4(1969), 225–233.

[26] G. Strang and G. Fix, *An Analysis of the Finite Element Method*, Prentice Hall, Englewood Cliffs, 1973.

[27] B. Swartz, Conditioning collocation, SIAM J., Numer. Anal., 25 (1988), 124–147.