MATHEMATICS OF COMPUTATION
VOLUME 52, NUMBER 186
APRIL 1989, PAGES 471–494

Nonnegativity-, Monotonicity-, or Convexity-Preserving Cubic and Quintic Hermite Interpolation*

By Randall L. Dougherty**, Alan Edelman***, and James M. Hyman

Dedicated to Professor Eugene Isaacson on the occasion of his 70th birthday

Abstract. The Hermite polynomials are simple, effective interpolants of discrete data. These interpolants can preserve local positivity, monotonicity, and convexity of the data if we restrict their derivatives to satisfy constraints at the data points. This paper describes the conditions that must be satisfied for cubic and quintic Hermite interpolants to preserve these properties when they exist in the discrete data. We construct algorithms to ensure that these constraints are satisfied and give numerical examples to illustrate the effectiveness of the algorithms on locally smooth and rough data.

1. Introduction. Piecewise polynomial interpolants, especially those based on Hermite polynomials (polynomials determined by their values and values of one or more derivatives at both ends of an interval), have a number of desirable properties. They are easy to compute once the derivative values are chosen. If the derivative values are chosen locally (e.g., by finite difference methods), then the interpolant at a given point will depend only on the given data at nearby mesh points. If the derivatives are computed by spline methods, then the interpolant will have an extra degree of continuity at the mesh points. In either case, the interpolant is linear in the given function values and has excellent convergence properties as the mesh spacing decreases.

These methods, however, do not necessarily preserve the shape of the given data. When the data arise from a physical experiment, it may be vital that the interpolant preserve nonnegativity ($f(x) \geq 0$), nonpositivity ($f(x) \leq 0$), monotonicity ($\dot{f}(x) \geq 0$ or $\dot{f}(x) \leq 0$), convexity ($\ddot{f}(x) \geq 0$), or concavity ($\ddot{f}(x) \leq 0$). In this and other cases, geometric considerations, such as preventing spurious behavior near rapid changes in the data, may be more important than the asymptotic accuracy of the interpolation method. One can construct a shape-preserving interpolant by constraining the derivatives for the Hermite polynomials to meet conditions which imply the desired properties ([4], [5], [8], [11]–[15], [20]), by adding new mesh points

Received September 15, 1988.
1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 41A05, 41A10, 41A15, 65D05, 65D10.
Key words and phrases. Approximation theory, convexity, interpolation, monotonicity, shape preservation, spline.
*Work performed under the auspices of the U.S. Department of Energy under contract W-7405-ENG-36 and the Office of Scientific Computing.
** *Current address*: Department of Mathematics, University of California, Los Angeles, CA 90024.
*** *Current address*: Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 01239.

and increasing the number of polynomial pieces ([6], [17]–[19], [22]), or by increasing the degree of the interpolating polynomials [16].

We have developed and tested practical shape-preserving interpolation algorithms for both cubic and quintic Hermite interpolation using the first of these methods (constraining the derivatives). These will be described in the remainder of this paper, as follows. First, we review the formulas for cubic and quintic Hermite interpolants. Next, we discuss sufficient (and necessary, in some cases) conditions for these interpolants to be nonnegative, monotone, or convex, and we give algorithms for modifying given derivative values so as to ensure that these conditions are satisfied. Finally, we give numerical examples to compare the proposed methods with other standard methods.

2. Hermite Interpolation. Let a mesh $\{x_i\}_{i=1}^n$ with $x_1 < x_2 < \cdots < x_n$ be given for the interval $[x_1, x_n]$, and let $\{f_i\}$, $f_i = f(x_i)$, be the corresponding data points. The local mesh spacing is $\Delta x_{i+1/2} = x_{i+1} - x_i$, and the slope of the piecewise linear interpolant between the data points is $S_{i+1/2} = \Delta f_{i+1/2}/\Delta x_{i+1/2}$. The data are *locally nonnegative* (*nonpositive*) in the interval $[x_i, x_{i+1}]$ if $f_i, f_{i+1} \geq 0$ (≤ 0). The interpolant Pf is *nonnegative* (*nonpositive*) in the interval $[x_i, x_{i+1}]$ if $(Pf)(x) \geq 0$ (≤ 0) for all $x \in [x_i, x_{i+1}]$. The data are *locally monotone* at x_i if $S_{i+1/2}S_{i-1/2} \geq 0$. The interpolant is *piecewise monotone* if $(Pf)'(x)$ does not change sign in any interval (x_i, x_{i+1}). The data are *locally convex* in the interval $[x_i, x_{i+1}]$ if $S_{i-1/2} \leq S_{i+1/2} \leq S_{i+3/2}$ and *locally concave* if $S_{i-1/2} \geq S_{i+1/2} \geq S_{i+3/2}$. The interpolant is of *class* C^k if $(Pf)(x)$ is continuous and has continuous derivatives for all orders less than or equal to k.

Given the data points $\{f_i\}$ and the slopes $\{\dot{f}_i\}$, the cubic Hermite interpolant is defined for $x_1 \leq x \leq x_n$ by

$$p(x) = c_0 + c_1\delta + c_2\delta^2 + c_3\delta^3, \tag{2.1}$$

where

$$c_0 = f_i, \qquad c_1 = \dot{f}_i,$$
$$c_2 = (3S_{i+1/2} - \dot{f}_{i+1} - 2\dot{f}_i)/(\Delta x_{i+1/2}),$$

and

$$c_3 = -(2S_{i+1/2} - \dot{f}_{i+1} - \dot{f}_i)/(\Delta x_{i+1/2})^2$$

for $x \in [x_i, x_{i+1}]$ and $\delta = x - x_i$.

The interpolant (2.1) has a continuous first derivative ($p(x) \in C^1$) and possibly, but not necessarily, a continuous second derivative. The continuity of the second derivative and the order of accuracy depend on how the slopes $\{\dot{f}_i\}$ are obtained. It is well known that the cubic Hermite interpolant is fourth-order accurate if the derivatives are exact; from the formulas above it follows that the interpolant is fourth-order if the derivatives are third-order, third-order if the derivatives are second-order, etc.

If, in addition, the second derivatives $\{\ddot{f}_i\}$ are available, the quintic Hermite interpolant can be defined by

$$q(x) = c_0 + c_1\delta + c_2\delta^2 + c_3\delta^3 + c_4\delta^4 + c_5\delta^5, \tag{2.2}$$

where

$$\begin{aligned}
&c_0 = f_i, \quad c_1 = \dot{f}_i, \quad c_2 = \ddot{f}_i/2,\\
&c_3 = (\ddot{f}_{i+1} - 3\ddot{f}_i)/(2\Delta x_{i+1/2}) + 2(5S_{i+1/2} - 3\dot{f}_i - 2\dot{f}_{i+1})/(\Delta x_{i+1/2}^2),\\
&c_4 = (3\ddot{f}_i - 2\ddot{f}_{i+1})/(2\Delta x_{i+1/2}^2) + (8\dot{f}_i + 7\dot{f}_{i+1} - 15S_{i+1/2})/(\Delta x_{i+1/2}^3),
\end{aligned}$$

and

$$c_5 = (\ddot{f}_{i+1} - \ddot{f}_i)/(2\Delta x_{i+1/2}^3) + 3(2S_{i+1/2} - \dot{f}_{i+1} - \dot{f}_i)/(\Delta x_{i+1/2}^4).$$

This interpolant is sixth-order accurate if $\dot{f}_i$ is fifth-order accurate and $\ddot{f}_i$ is fourth-order accurate.

Often only the data points $\{f_i\}$ are given, and the derivative must be numerically approximated by, for example, local Lagrange or least squares interpolants ([2], [11], [13], [16]). Note that, once the derivatives of f are given, (2.1) and (2.2) are local; changing the value of $f_i, \dot{f}_i$, or $\ddot{f}_i$ affects the interpolant only in the region $[x_{i-1}, x_{i+1}]$. If the calculation of $\dot{f}$ and $\ddot{f}$ is also local, only nearby data points need be available when interpolating between x_i and x_{i+1}. This localness is important when storage requirements are critical, such as for very large data sets, multidimensional interpolation or on parallel computers with local memory.

Any algorithm defining $\{\dot{f}_i\}$ that makes (2.1) a C^2 interpolant, or defining $\{\dot{f}_i, \ddot{f}_i\}$ that makes (2.2) a C^3 interpolant (for example, the complete spline interpolants with given endpoint derivatives [1]), is nonlocal. To gain total localness, we therefore sacrifice a degree of smoothness.

We begin the algorithms by generating an accurate, but not necessarily shape-preserving, interpolant using derivative approximations obtained from either finite differences or the spline method. When the interpolant satisfies conditions sufficient for shape preservation, it is left unchanged. When the conditions are not satisfied, we replace the derivatives at the data points with values that do satisfy the conditions and give the desired shape-preserving interpolant. Note that the constrained interpolant may not be linear in the data.

The constraints for nonnegativity (for cubics and quintics) and for monotonicity (for cubics) given in the next two sections are extremely local; the constraint for the derivatives at x_i does not involve derivatives elsewhere. In such cases, a more complicated algorithm than we use, but one that causes fewer jumps in the derivatives, involves first computing $\{\dot{f}_i\}$ ($\{\dot{f}_i, \ddot{f}_i\}$) for the complete cubic (quintic) spline interpolant in the interval $[x_1, x_n]$. If the interpolant is not shape-preserving, we locate the point x_j where the conditions fail most badly. We redefine the derivative(s) at x_j to meet the conditions and solve for the complete spline interpolant in $[x_1, x_j]$ and $[x_j, x_n]$ using f_j and $\dot{f}_j$ (and $\ddot{f}_j$) as boundary conditions. The resulting interpolant will have a break in the second (third) derivative only at x_j. If none of the resulting constraints is violated, the algorithm terminates. Otherwise, we repeat the process, breaking $[x_1, x_j]$ or $[x_j, x_n]$ into smaller subregions, and continue. This algorithm always will terminate if the derivatives at the boundaries are given and they satisfy the constraints.

The constraints can also be used in conjunction with a constrained optimization algorithm to, for example, minimize the L_p norm of the curvature (or some other quantity) subject to one or more of the shape-preservation constraints. Ferguson [9] has used similar methods to construct shape-preserving parametric cubic interpolants. These methods are more costly and complicated than the algorithms we propose, but often produce a visually more pleasing interpolant.

3. Nonnegativity. Retaining nonnegativity is important in many real-world situations ([9], [13]). For example, when the data represent the density or pressure of a material, negative values are not physically meaningful. The cubic Hermite interpolant will have the same sign as the piecewise linear interpolant if

$$-3\tau_i f_i/\Delta x_{i+1/2} \le \tau_i \dot{f}_i \le 3\tau_i f_i/\Delta x_{i-1/2}, \tag{3.1}$$

where $\tau_i = \operatorname{sgn}(f_i)$ ([4], [11]). The quintic Hermite interpolant preserves nonnegativity or nonpositivity if

$$-5\tau_i f_i/\Delta x_{i+1/2} \le \tau_i \dot{f}_i \le 5\tau_i f_i/\Delta x_{i-1/2} \tag{3.2a}$$

and

$$\tau_i \ddot{f}_i \ge \tau_i \max\left(\frac{8\dot{f}_i}{\Delta x_{i-1/2}} - \frac{20 f_i}{(\Delta x_{i-1/2})^2}, \frac{-8\dot{f}_i}{\Delta x_{i+1/2}} - \frac{20 f_i}{(\Delta x_{i+1/2})^2}\right). \tag{3.2b}$$

A simple constraining algorithm to ensure that the sign of the Hermite interpolant mimics that of the data is

$$\tau_i \dot{f}_i \leftarrow \min(K\tau_i f_i/\Delta x_{i+1/2}, \max(-K\tau_i f_i/\Delta x_{i-1/2}, \tau_i \dot{f}_i)), \tag{3.3}$$

where $K = 3$ for cubics and $K = 5$ for quintics. In addition, for quintics, $\ddot{f}_i$ must be constrained to satisfy (3.2b).

4. Monotonicity. We review some previously published results on monotonicity-preserving cubic Hermite interpolation ([9], [13], [14]), derive the monotonicity constraints for quintics, and describe an efficient constraining algorithm. First, we state some general properties of monotonicity-preserving polynomials.

Consider the vector space N of real polynomials of degree $2n+1$. Let $H = \{p \in N \mid p(0) = 0, p(1) = 1\}$ be a hyperplane in N. The monotonicity region M is defined as $\{p \in H \mid p' \ge 0 \text{ on } (0,1)\}$. M is closed and convex; since $\sup_{0\le x\le 1} |p(x)| = 1$, M is bounded and hence compact. Since the interior of M relative to H is $\{p \in H \mid p' > 0\}$, if p is on the boundary of M relative to H, then either $p'(0) = 0$ or $p'(1) = 0$, or the discriminant of p' is 0.

The derivative of the cubic Hermite polynomial $p(t)$ is

$$p'(t) = [3(\dot{p}_0 + \dot{p}_1 - 2)]t^2 + [2(-2\dot{p}_0 - \dot{p}_1 + 3)]t + \dot{p}_0,$$

where $p(0) = 0$, $p(1) = 1$, $p'(0) = \dot{p}_0$, and $p'(1) = \dot{p}_1$. Setting the discriminant to zero gives the ellipse

$$(\dot{p}_0 - 1)^2 + (\dot{p}_0 - 1)(\dot{p}_1 - 1) + (\dot{p}_1 - 1)^2 - 3(\dot{p}_0 + \dot{p}_1 - 2) = 0,$$

which, as Fritsch and Carlson [13] point out, is tangent to the coordinate axes at (3,0) and (0,3). The boundary of M must be some subset of this ellipse and the

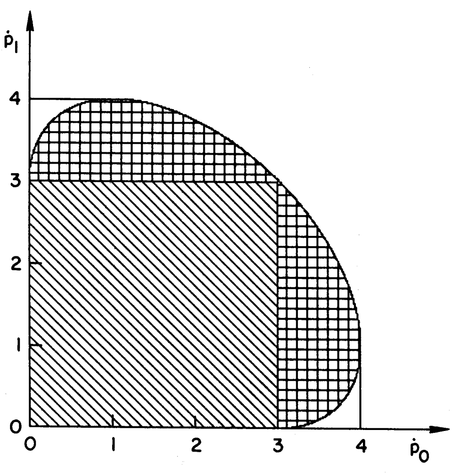


FIGURE 4.1
The Fritsch-Carlson monotonicity region ([10], [13]) *for the cubic Hermite polynomial* (*union of hatched areas*). *The diagonally hatched area is the de Boor-Swartz* [3] *box.*

coordinate axes. The region shown in Figure 4.1 is the only convex, compact one with nonempty interior whose boundary is in this set.

For higher-degree polynomials, given complete freedom to vary the higher-order derivatives, the first derivative must be contained within one of a nested sequence of regions bounded by the coordinate axes and ellipses (odd-degree polynomials) or line segments (even-degree polynomials) [7]. Figure 4.2 indicates the structure of these sets. The region for polynomials of degree $2n$ is triangular with vertices $(0,0), (n^2+n,0)$, and $(0,n^2+n)$. The region for polynomials of degree $2n-1$ is bounded by the coordinate axes and the outer part of an ellipse with center $(\frac{1}{2}n^2, \frac{1}{2}n^2)$ which is tangent to the coordinate axes at $(n^2-1,0)$ and $(0,n^2-1)$.

A. Cubic Polynomials.

1. *Monotonicity Constraints–Cubics.* A simple generalization of what was recognized by de Boor and Swartz [3] is that if

$$0 \le \dot{f}_i, \dot{f}_{i+1} \le 3S_{i+1/2} \quad \text{or} \quad 3S_{i+1/2} \le \dot{f}_i, \dot{f}_{i+1} \le 0, \tag{4.1}$$

the resulting interpolant is monotone in $[x_i, x_{i+1}]$. Ferguson and Miller [10] and Fritsch and Carlson [13] independently found an extension of this criterion that gives a necessary and sufficient condition for (2.1) to be monotone. The de Boor-Swartz criterion is a square inscribed within the Fritsch-Carlson monotonicity region.

A simple algorithm to guarantee a monotonicity-preserving cubic Hermite interpolant is to project the derivatives of the interpolant to the de Boor-Swartz box [14].

2. *Monotonicity Algorithm–Cubics.* When the data are locally monotone, we restrict $\{\dot{f}_i\}$ to the de Boor-Swartz piecewise monotonicity range (4.1). After calculating an accurate approximation to $\dot{f}_i$, we project it to the allowed monotonicity

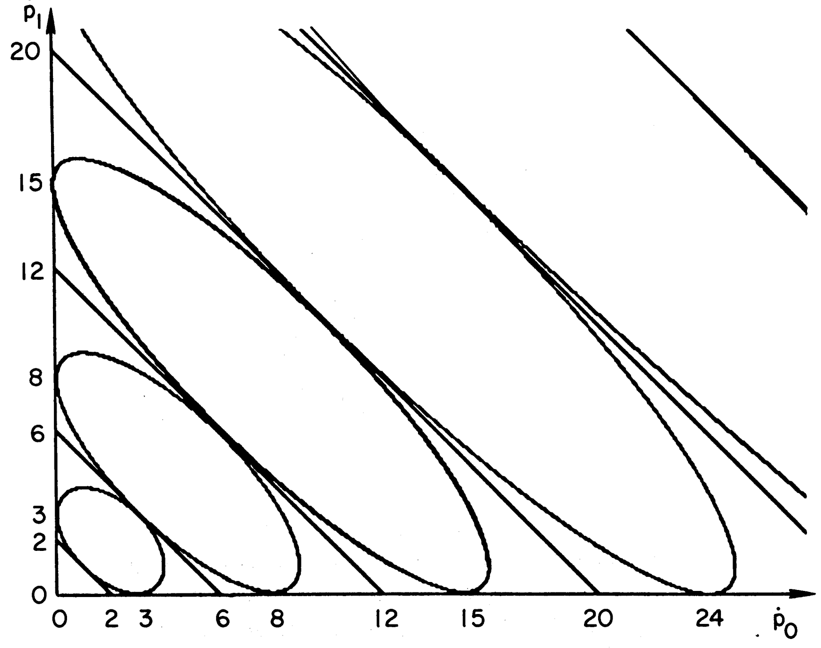


FIGURE 4.2
The monotonicity regions for the first derivatives of piecewise polynomial Hermite interpolants [7].

region according to

$$(4.2)\qquad \dot{f}_i \leftarrow \begin{cases} \min(\max(0,\dot{f}_i), 3\min(|S_{i-1/2}|, |S_{i+1/2}|)), & \sigma_i > 0, \\ \max(\min(0,\dot{f}_i), -3\min(|S_{i-1/2}|, |S_{i+1/2}|)), & \sigma_i < 0, \\ 0, & \sigma_i = 0, \end{cases}$$

where $\sigma_i = \text{sgn}(S_{i+1/2})$ if $S_{i+1/2}S_{i-1/2} > 0$ and $\sigma_i = 0$ otherwise.

Near the boundary, this constraint can be used if we define $S_{1/2}$ and $S_{n+1/2}$ to be $S_{3/2}$ and $S_{n-1/2}$, respectively.

If the given data are samples from a sufficiently smooth function f with positive derivative at a given point x, and the derivative estimates $\dot{f}_i$ are accurate approximations to the true derivatives $f'(x_i)$, then (4.1) will be satisfied near x once the mesh is sufficiently refined. Therefore, the interpolant is restricted by geometric considerations only near where the mesh is coarse, the underlying function is nonsmooth, or the underlying function has a critical point. In the latter case, accuracy can be degraded slightly even for smooth monotone functions, but Eisenstat, Jackson, and Lewis [8] show that the method is still third-order accurate (given second-order accurate values for $\dot{f}_i$).

When the data are not locally monotone, the interpolant also must have an extremum. Retaining piecewise monotonicity would require that $\dot{f}_i = 0$ when $S_{i+1/2}S_{i-1/2} < 0$ and would "clip" the interpolant by forcing it to have an extremum at x_i rather than at a possibly more appropriate nearby point. We believe that relaxing the piecewise monotonicity constraint in the interval pair next to the extremum produces a visually more pleasing curve. However, if new constraints should be imposed at extrema, the change in decision algorithms must still produce a stable interpolant. That is, a small change in the data should not create a large change in the interpolant. If we remove all constraints on the interpolant near locally nonmonotone data while retaining (4.1) elsewhere, the resulting interpolant will be unstable. If σ_i is defined in (4.2) by $\sigma_i = \operatorname{sgn}(\dot{f}_i)$ when $S_{i+1/2}S_{i-1/2} < 0$, then the third condition in (4.2) is unnecessary and the resulting constraint,

$$\dot{f}_i \leftarrow \begin{cases} \min(\max(0, \dot{f}_i), 3\min(|S_{i-1/2}|, |S_{i+1/2}|)), & \sigma_i \geq 0, \\ \max(\min(0, \dot{f}_i), -3\min(|S_{i-1/2}|, |S_{i+1/2}|)), & \sigma_i \leq 0, \end{cases} \tag{4.3}$$

is not as restrictive near extrema.

Even (4.3) can be overly restrictive, however, because it requires the constrained derivatives to be very small at any pair of mesh points where the piecewise linear interpolant has very small slope. As shown in Figure 4.3, this can introduce second-order errors into the interpolant even for very nice (albeit nonmonotone) data. To avoid this problem, we make two changes in the algorithm.

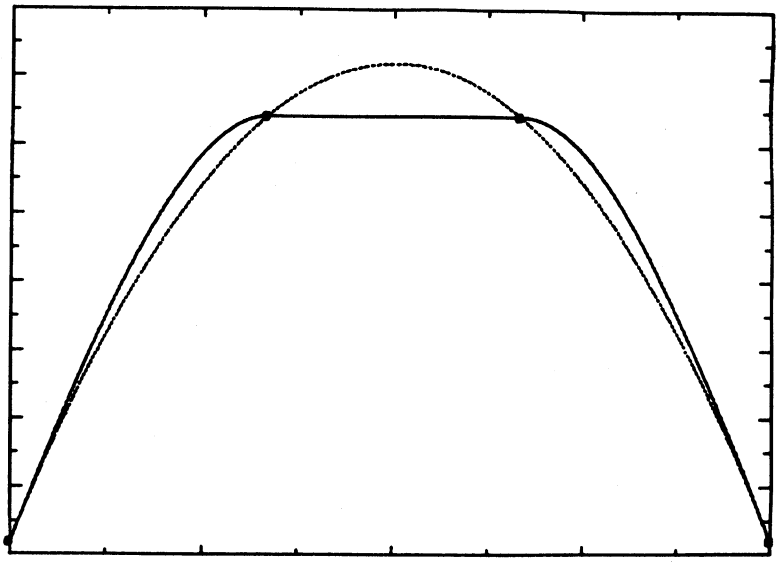


FIGURE 4.3
Interpolation of four points on a parabola, before (dotted) and after (solid) constraining by (4.3).

For any i and j such that $1 \leq i \leq n$ and $1 < i+j < n$, let p_i^j be the slope at x_i of the parabola through the points (x_k, f_k), $k = i+j-1, i+j, i+j+1$. We will use this only for $-1 \leq j \leq 1$, for which we have the following formulas:

$$\begin{aligned} p_i^{-1} &= \frac{S_{i-1/2}(2\Delta x_{i-1/2} + \Delta x_{i-3/2}) - S_{i-3/2}\Delta x_{i-1/2}}{\Delta x_{i-3/2} + \Delta x_{i-1/2}}, \\ p_i^0 &= \frac{S_{i-1/2}\Delta x_{i+1/2} + S_{i+1/2}\Delta x_{i-1/2}}{\Delta x_{i-1/2} + \Delta x_{i+1/2}}, \\ p_i^1 &= \frac{S_{i+1/2}(2\Delta x_{i+1/2} + \Delta x_{i+3/2}) - S_{i+3/2}\Delta x_{i+1/2}}{\Delta x_{i+1/2} + \Delta x_{i+3/2}}. \end{aligned} \tag{4.4}$$

The first change we make is to constrain $\dot{f}_i$ to lie between 0 and $3p_i^0$ (inclusive) for $1 < i < n$. This already follows from (4.3) if the data are locally monotone at

x_i; if the data are not locally monotone at x_i, it is an extra constraint, but not an unreasonable one.

The second change is to relax (4.3) in certain cases, as follows: if $S_{i-1/2} > \max(0, S_{i+1/2})$ and $S_{i+3/2} < \min(0, S_{i+1/2})$, we allow $\dot{f}_i$ to be as large as

$$\max(3\min(p_i^0, |S_{i-1/2}|, |S_{i+1/2}|), C\min(p_i^0, p_i^1))$$

if $p_i^0 > 0$, and we allow $-\dot{f}_{i+1}$ to be as large as

$$\max(3\min(-p_{i+1}^0, |S_{i+1/2}|, |S_{i+3/2}|), C\min(-p_{i+1}^0, -p_{i+1}^{-1}))$$

if $p_{i+1}^0 < 0$. (We will determine the constant C below.) If $S_{i-1/2} < \min(0, S_{i+1/2})$ and $S_{i+3/2} > \max(0, S_{i+1/2})$, we similarly relax the constraint on $-\dot{f}_i$ (if $p_i^0 < 0$) and on $\dot{f}_{i+1}$ (if $p_{i+1}^0 > 0$). One has to worry that this might introduce an extra pair of local extrema between x_i and x_{i+1} if, for example, $S_{i-1/2} < S_{i+1/2} < 0 < S_{i+3/2}$; however, this could only happen if the constrained value for $\dot{f}_{i+1}$ were negative, which would force p_{i+1}^0 to be negative by the first change we made, which in turn would imply $p_i^1 > -2S_{i+1/2}$. Therefore, as long as $C \leq 1.5$, the new value for $\dot{f}_i$ will be at least $1.5p_i^1 > -3S_{i+1/2}$, so there will be no extra extrema.

We now know that C should be at least 1 to handle the case in Figure 4.3 correctly, but C should be at most 1.5 in order to avoid extra local extrema. In order to handle data that are almost but not quite on a parabola, and in keeping with our philosophy of changing the original derivatives as little as possible, we have decided to use the largest permissible value of C, namely 1.5. Note that requiring $C \leq 1.5$ implies that the condition $S_{i-1/2} > 0 > S_{i+3/2}$ or $S_{i-1/2} < 0 < S_{i+3/2}$ need not be checked, since if they fail but $S_{i-1/2} > S_{i+1/2} > S_{i+3/2}$ or $S_{i-1/2} < S_{i+1/2} < S_{i+3/2}$ holds, then we will have $|p_i^0| \leq 2\min(|S_{i-1/2}|, |S_{i+1/2}|)$ and $|p_{i+1}^0| \leq 2\min(|S_{i+1/2}|, |S_{i+3/2}|)$, so (4.3) will not be relaxed anyway. Therefore, the final algorithm for modifying the original value of $\dot{f}_i$ $(1 < i < n)$ is:

```
Compute $p_i^{-1}, p_i^0, p_i^1$ as in (4.4);
Let $M_i \leftarrow 3\min(|S_{i-1/2}|, |S_{i+1/2}|, |p_i^0|)$;
If $i > 2$ and the numbers $p_i^0$, $p_i^{-1}$, $S_{i-1/2} - S_{i-3/2}$, and $S_{i+1/2} - S_{i-1/2}$
    all have the same sign, let $M_i \leftarrow \max(M_i, 1.5\min(|p_i^0|, |p_i^{-1}|))$;
If $i < n - 1$ and the numbers $-p_i^0, -p_i^1$, $S_{i+1/2} - S_{i-1/2}$, and $S_{i+3/2} - S_{i+1/2}$
    all have the same sign, let $M_i \leftarrow \max(M_i, 1.5\min(|p_i^0|, |p_i^1|))$;
Let $\dot{f}_i \leftarrow \begin{cases} (\operatorname{sgn}\dot{f}_i)\min(|\dot{f}_i|, M_i) & \text{if } \operatorname{sgn}\dot{f}_i = \operatorname{sgn}p_i^0 \\ 0 & \text{otherwise.} \end{cases}$
```

For $i = 1$, if $\operatorname{sgn}\dot{f}_i = \operatorname{sgn}S_{i+1/2}$ set $\dot{f}_i$ to $(\operatorname{sgn}\dot{f}_i)\min(|\dot{f}_i|, 3|S_{i+1/2}|)$, otherwise set $\dot{f}_i$ to 0; handle $i = n$ similarly.

We will now prove:

THEOREM. *The above algorithm generates a third-order accurate (in L_∞) interpolant, if the original derivative values are second-order accurate.*

Fix a function $f \in C^3[a, b]$. Suppose $h > 0$, and we have a mesh $a = x_1 < x_2 < \cdots < x_n = b$ such that $\Delta x_{i+1/2} \leq h$ for all i. We must show that, if values $\dot{f}_i$ are chosen so that $|\dot{f}_i - f'(x_i)| = O(h^2)$, these values are modified by the constraining algorithm, and the modified values are used to construct a cubic Hermite interpolant Pf, then $\|f - Pf\|_\infty = O(h^3)$. First note that, since second-order accurate derivatives produce a third-order accurate cubic Hermite interpolant, it suffices to show that the constrained derivatives are still second-order accurate. Second, since the constraining algorithm merely changes $\dot{f}_i$ to the nearest value in some interval, the function which maps unconstrained derivatives to constrained derivatives is Lipschitz with constant 1, so it suffices to show that, if the constraining algorithm is applied to the exact derivatives $f'(x_i)$, it changes them to values $\dot{f}_i$ such that $|\dot{f}_i - f'(x_i)| = O(h^2)$. To do this, we use two lemmas (the constants are not the best possible), which can easily be proven in contrapositive form using the Mean Value Theorem.

LEMMA 1. *If* $|f'''(x)| \le c$ *for* $x \in (x_i, x_{i+1})$, *then* $|f'(x_i) + f'(x_{i+1}) - 2S_{i+1/2}| \le ch^2$.

LEMMA 2. *If* $P(x)$ *is the quadratic function such that* $P(x_j) = f(x_j)$ *for* $j = i-1, i, i+1$, *and if* $|f'''(x)| \le c$ *for all* $x \in (x_{i-1}, x_{i+1})$, *then* $|P'(x_j) - f'(x_j)| \le 2ch^2$ *for* $j = i-1, i, i+1$.

We will now show that, for sufficiently small h, if $|f'''(x)| \le K$ for all $x \in [a, b]$, then $|\dot{f}_i - f'(x_i)| \le 3Kh^2$ for all i; this will complete the proof of the theorem. Suppose $|\dot{f}_i - f'(x_i)| > 3Kh^2$ for some i; then $f'(x_i)$ has violated some constraint by at least $3Kh^2$. For sufficiently small h, $S_{1+1/2}$ will have the same sign as $f'(x_1)$ if $f'(x_1) \ne 0$, and similarly for $S_{n-1/2}$ and $f'(x_n)$. If $1 < i < n$, then $|f'(x_i) - p_i^0| \le 3Kh^2$ by Lemma 2. Hence, the constraint violation must be of the form $|f'(x_i)| > 3\min(|S_{i-1/2}|, |S_{i+1/2}|) + 3Kh^2$; by symmetry, we may assume $f'(x_i) > 3|S_{i-1/2}| + 3Kh^2$. Then, by Lemma 1, we must have $f'(x_{i-1}) < -|S_{i-1/2}| - 2Kh^2$.

Next note that i cannot be n if h is sufficiently small. The reason for this is that, as h tends to 0, $f'(x_{n-1})$ tends to $f'(x_n)$ (which remains fixed), so if $f'(x_n) > 0$ as the above inequalities require, then $f'(x_{n-1}) > 0$ for sufficiently small h, so we cannot have $f'(x_{n-1}) < -|S_{n-1/2}| - 2Kh^2$. Similarly, i cannot be 2 if h is sufficiently small. Also, we must have $S_{i-3/2} < -|S_{i-1/2}| \le S_{i-1/2}$, since otherwise we would have $p_{i-1}^0 \ge -|S_{i-1/2}|$, contradicting Lemma 2. Similarly, $S_{i+1/2} > |S_{i-1/2}| \ge S_{i-1/2}$. Therefore, we are in the case where the constraining algorithm may relax (4.3); since Lemma 2 requires both p_i^0 and p_i^{-1} to be within $2Kh^2$ of $f'(x_i)$, the algorithm will relax (4.3) enough to give $|\dot{f}_i - f'(x_i)| \le 2Kh^2$, which is a contradiction. This completes the proof of the theorem.

B. Quintic Polynomials.

1. *Monotonicity Constraints-Quintics.* The monotonicity region for cubics is a simple region in a plane. The monotonicity region M for quintics is a compact convex region in a four-dimensional hyperplane H, which we can describe as a subset of R^4 using the coordinates $(\dot{f}_i, \ddot{f}_i, \dot{f}_{i+1}, \ddot{f}_{i+1})$. We first discuss a method of drawing planar slices of this monotonicity region. Only four such slices are needed for the algorithm.

Given fixed values of $\dot{f}_i$ and $\dot{f}_{i+1}$, it is desirable to have a picture of the points $(\ddot{f}_i, \ddot{f}_{i+1}) \in R^2$ for which $(\dot{f}_1, \ddot{f}_i, \dot{f}_{i+1}, \ddot{f}_{i+1}) \in M$. This is the intersection of M

with the plane where $\dot{f}_i$ and $\dot{f}_{i+1}$ are fixed. Note that, for a given $r \in (0,1)$, there is a unique quartic polynomial of the form

$$y(x) = (x-r)^2(ax^2+bx+c)$$

such that $y(0) = \dot{f}_i$, $y(1) = \dot{f}_{i+1}$, and $\int_0^1 y(x)\,dx = 1$.

We can compute $a, b, c, y'(0)$, and $y'(1)$ in terms of r:

$$\begin{aligned}
a &= \frac{(30r^4-40r^3+15r^2)\dot{f}_{i+1}+(30r^4-80r^3+75r^2-30r+5)\dot{f}_i-(60r^4-120r^3+60r^2)}{10r^6-30r^5+33r^4-16r^3+3r^2},\\
b &= \frac{\dot{f}_{i+1}}{(1-r)^2} - a - \frac{\dot{f}_i}{r^2},\\
c &= \dot{f}_i/r^2,\\
y'(0) &= r^2b - 2rc,
\end{aligned}$$

and

$$y'(1) = (1-r)^2(2a+b) + 2(1-r)(a+b+c).$$

The point $(\dot{f}_i, y'(0), \dot{f}_{i+1}, y'(1))$ is a good candidate for the boundary of the monotonicity region. If $ax^2+bx+c \geq 0$ on $[0,1]$, the point is on the boundary. As r ranges over $[0,1]$, we generate a loop which, if $\dot{f}_i\dot{f}_{i+1} \neq 0$, is the entire boundary of the cross section; if $\dot{f}_i\dot{f}_{i+1} = 0$, then one or both of the coordinate axes also form part of the boundary.

Four slices of the quintic monotonicity region are shown in Figure 4.4. The proposed monotonicity-preserving algorithm will restrict the derivatives to the inscribed rectangles in the same way that the cubic monotonicity-preserving algorithm restricted the derivatives to the de Boor-Swartz box.

The value 5 in Figure 4.4 is somewhat arbitrary. A value as high as 6 can be used; the cross section with $\dot{f}_i = \dot{f}_{i+1} = 6$ consists of the single point $(-60, 60)$, and rectangles can be chosen within the cross sections with $\dot{f}_i = 0$, $\dot{f}_{i+1} = 6$ and vice versa. The resulting algorithm would impose slightly weaker constraints on the first derivatives, but the intervals of permissible values for the second derivatives would be slightly smaller. The resulting interpolant would differ very little from that obtained using the original algorithm.

The convex hull of these four rectangles is a four-dimensional solid. This solid, when intersected with a plane of constant $\dot{f}_i$ and $\dot{f}_{i+1}$, forms a polygon that must be inside the convex, connected monotonicity region. Within this polygon we choose a rectangle defined by

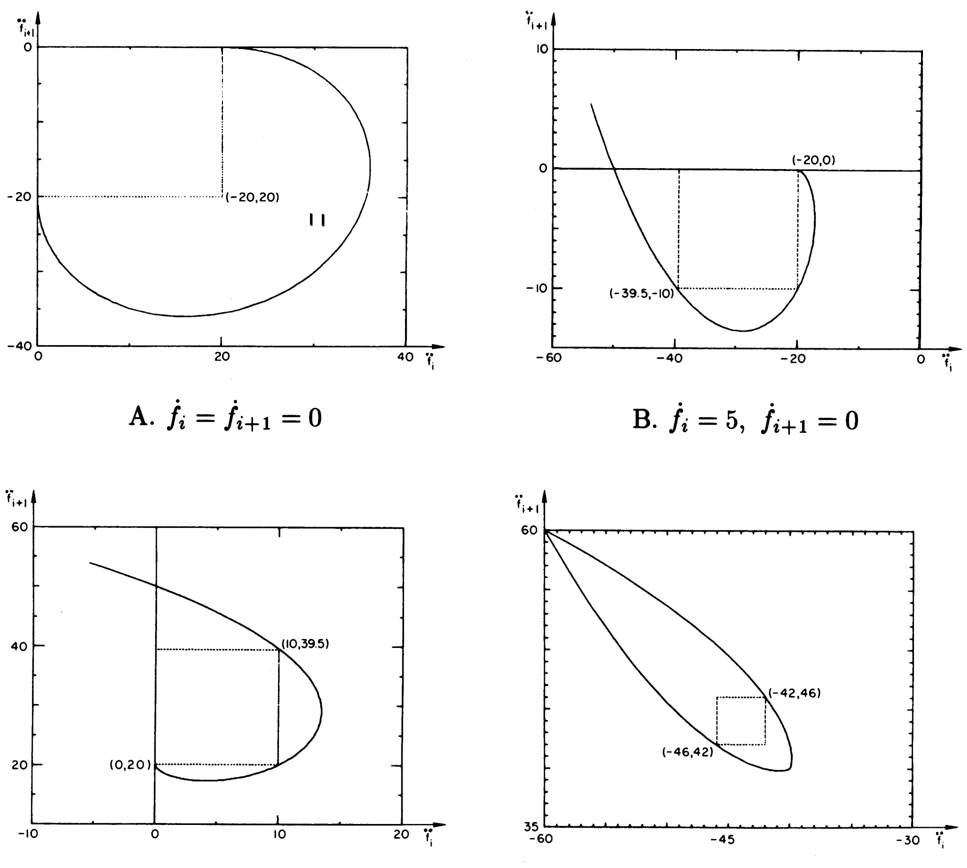
$$\ddot{f}_i \in [L_0^m(\dot{f}_i, \dot{f}_{i+1}), U_0^m(\dot{f}_i, \dot{f}_{i+1})]$$

and

$$\ddot{f}_{i+1} \in [L_1^m(\dot{f}_i, \dot{f}_{i+1}), U_1^m(\dot{f}_i, \dot{f}_{i+1})],$$

where

$$\begin{aligned}
L_0^m(a,b) &= -7.9a - 0.26ab,\\
U_0^m(a,b) &= 20 - 8a - 2b - 0.48ab,\\
L_1^m(a,b) &= -U_0^m(b,a),
\end{aligned}$$



A. $\dot f_i = \dot f_{i+1} = 0$ B. $\dot f_i = 5,\ \dot f_{i+1} = 0$

C. $\dot f_i = 0,\ \dot f_{i+1} = 5$ D. $\dot f_i = \dot f_{i+1} = 5$

FIGURE 4.4
Four cross sections of the quintic monotonicity region.

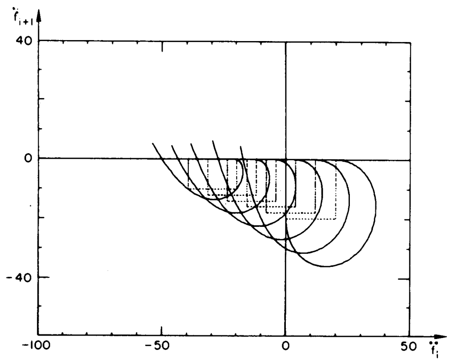
and

$$U_1^m(a,b) = -L_0^m(b,a).$$

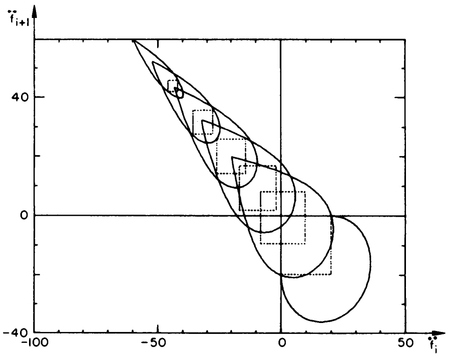
To confirm that this rectangle is contained within the monotonicity region for all $a, b \in [0,5]$, it suffices by convexity to verify that, for $a, b \in \{0,5\}$, the four corners of the rectangle are in the region; these sixteen verifications can be done directly.

Figure 4.5 shows two sequences of cross sections of the monotonicity region: one in which $\dot f_{i+1}$ is held at 0 while $\dot f_i$ varies from 0 to 5, and one in which $\dot f_i$ and $\dot f_{i+1}$ are equal and vary from 0 to 5. The inscribed rectangles indicate how much of the monotonicity region is used by the algorithm. Figures 4.5A and B show the cross sections and rectangles as they actually appear in the $(\ddot f_i, \ddot f_{i+1})$ plane; in Figures 4.5C and D, each cross section is rescaled to map the rectangles to the unit square.

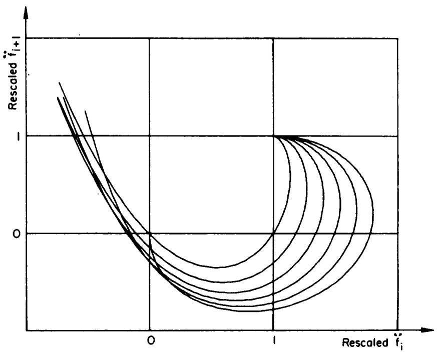
We now consider arbitrary data points $x_1 < x_2 < x_3 < \cdots < x_n$ and arbitrary function values $f_1, f_2, \ldots, f_n$ and describe how to construct a function $Pf(x)$ defined on $[x_1, x_n]$ such that $Pf(x)$ is a monotonicity-preserving quintic on $[x_i, x_{i+1}]$ for $i = 1, 2, \ldots, n-1$, Pf is twice differentiable at the points x_i, and $Pf(x_i) = f_i$.



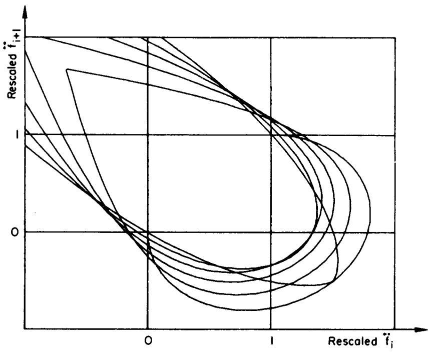
A. $\dot{f}_{i+1} = 0,\ \dot{f}_i = 0, 1, 2, 3, 4, 5$



B. $\dot{f}_i = \dot{f}_{i+1} = 0, 1, 2, 3, 4, 5$



C. $\dot{f}_{i+1} = 0$, rescaled



D. $\dot{f}_i = \dot{f}_{i+1}$, rescaled

FIGURE 4.5
Cross sections of the quintic monotonicity region.

Assume we have a quintic Hermite interpolant q for this data. If $f_i \neq f_{i+1}$, let

$$g_i(t) = [q([x_{i+1} - x_i]t + x_i) - f_i]/(f_{i+1} - f_i), \qquad 0 \le t \le 1;$$

g_i is defined on $[0,1]$ and $g_i(0) = 0$, $g_i(1) = 1$. Since g_i is a normalization of q, g_i is monotonic if and only if q restricted to $[x_i, x_{i+1}]$ is. If $f_i = f_{i+1}$, take $g_i(x) = 0$, because the only monotonic function defined on $[x_i, x_{i+1}]$ in this case is constant.

The function g_i is monotonic if the following conditions hold for $i = 1, 2, \ldots, n-1$:

$$L_0^m(\dot{g}_i(0), \dot{g}_i(1)) \le \ddot{g}_i(0) \le U_0^m(\dot{g}_i(0), \dot{g}_i(1))$$

and

$$L_1^m(\dot{g}_i(0), \dot{g}_i(1)) \le \ddot{g}_i(1) \le U_1^m(\dot{g}_i(0), \dot{g}_i(1)).$$

Note that these do indeed hold for $g_i \equiv 0$.

From the definition of g_i, we see that, if $f_i \neq f_{i+1}$, then

$$\begin{aligned}\dot{g}_i(0) &= \dot{f}_i/S_{i+1/2},\\ \dot{g}_i(1) &= \dot{f}_{i+1}/S_{i+1/2},\\ \ddot{g}_i(0) &= \ddot{f}_i/(S_{i+1/2}/\Delta x_{i+1/2}),\end{aligned}$$

and

$$\ddot{g}_i(1) = \ddot{f}_{i+1}/(S_{i+1/2}/\Delta x_{i+1/2}).$$

If $f_{i+1} = f_i$, all derivatives vanish. In the above equations and in the following analysis, we define $0/0 = 0$. The local conditions at x_i for $i = 2, \dots, n-1$ are that

$$\ddot{g}_i(0) = \ddot{f}_i/(S_{i+1/2}/\Delta x_{i+1/2}) \in [L_0^m(\dot{g}_i(0), \dot{g}_i(1)), U_0^m(\dot{g}_i(0), \dot{g}_i(1))]$$

and

$$\ddot{g}_{i-1}(1) = \ddot{f}_i/(S_{i-1/2}/\Delta x_{i-1/2}) \in [L_1^m(\dot{g}_{i-1}(0), \dot{g}_{i-1}(1)), U_1^m(\dot{g}_{i-1}(0), \dot{g}_{i-1}(1))].$$

2. *Monotonicity Algorithm–Quintics.* We constrain the values of $\dot{f}_i$ as follows:

$$(4.5) \qquad \dot{f}_i \leftarrow \begin{cases} \min(\max(0, \dot{f}_i), 5\min(|S_{i-1/2}|, |S_{i+1/2}|)), & \sigma_i \geq 0, \\ \max(\min(0, \dot{f}_i), -5\min(|S_{i-1/2}|, |S_{i+1/2}|)), & \sigma_i \leq 0, \end{cases}$$

where σ_i is defined as for (4.3). If monotonicity within each interval is desired, an analogue of (4.2) with 3 replaced by 5 can be used instead of (4.5).

If the data are monotonically increasing at x_i, the quintic Hermite interpolant will be monotonic if $\{\dot{f}_i\}$ is constrained by (4.5) and $\{\ddot{f}_i\}$ is constrained to satisfy

$$(4.6) \qquad \ddot{f}_i \in A \quad \text{and} \quad \ddot{f}_i \in B,$$

where

$$\begin{aligned} A &= [-7.9d^+ - 0.26d^+b,\ (20 - 2b)S_{i+1/2} - 8d^+ - 0.48d^+b]\Delta x_{i+1/2}^{-1},\\ B &= [(-20 + 2a)S_{i-1/2} + 8d^- + 0.48ad^-, 7.9d^- + 0.26ad^-]\Delta x_{i-1/2}^{-1},\\ a &= \max(0, \dot{f}_{i-1}/S_{i-1/2}), \qquad b = \max(0, \dot{f}_{i+1}/S_{i+1/2}),\end{aligned}$$

and $d^\pm$ is $\dot{f}_i$ if $\dot{f}_i S_{i\pm1/2} > 0$ and equal to 0 otherwise. The factor on the right of each interval is meant to multiply both endpoints of the interval.

If $\dot{f}_i \leq 0$, then, since $0 \leq a, b \leq 5$, the two intervals take the forms $A = [0, +]$ and $B = [-, 0]$, so they must intersect. If $\dot{f}_i > 0$, the two intervals may not intersect; in this case, $\dot{f}_i$ must be reduced beyond what is called for in (4.5). When this occurs, it is desirable to change $\dot{f}_i$ by the least possible amount, that is, to replace $\dot{f}_i$ with the highest value that allows the two intervals to intersect. To do this, we set the right endpoint of interval A equal to the left endpoint of interval B and solve for $\dot{f}_i$. The value is given by

$$(4.7) \qquad \dot{f}_i = \frac{(20 - 2b)S_{i+1/2}/\Delta x_{i+1/2} + (20 - 2a)S_{i-1/2}/\Delta x_{i-1/2}}{(8 + 0.48b)/\Delta x_{i+1/2} + (8 + 0.48a)/\Delta x_{i-1/2}}.$$

If $S_{i-1/2} < 0$ and $S_{i+1/2} < 0$, both intervals in (4.6) should be reversed. Equation (4.7) now gives the smallest $\dot{f}_i < 0$ for which the intervals intersect.

If $S_{i+1/2}S_{i-1/2} \leq 0$, one of the intervals in (4.6) should be reversed. Since either d^+ or d^- must be zero in this case, one of the intervals must be of the form $[0,+]$ or $[-,0]$. If $\dot{f}_i = 0$, the other interval will have the same form, so the two intervals will intersect; therefore, as in the two preceding cases, when the two intervals do not intersect, sufficiently reducing $|\dot{f}_i|$ remedies the problem.

In any of these cases, if values for $\dot{f}_{i-1}$, $\dot{f}_i$, and $\dot{f}_{i+1}$ are given that yield intersecting intervals and if $\dot{f}_{i-1}$ or $\dot{f}_{i+1}$ is reduced in absolute value (but its sign is not changed), then the new intervals will also intersect. Therefore, we may proceed sequentially from $i = 2$ to $i = n-1$, at each stage reducing $|\dot{f}_i|$ if necessary to obtain a nonempty set of permissible values for $\ddot{f}_i$. Afterwards, we can modify the second derivatives by setting $\ddot{f}_i$ to the value permitted by (4.6) that is closest to the original estimate for $\ddot{f}_i$.

Alternatively, we can do the reductions to $|\dot{f}_i|$ "in parallel"; that is, for each i, we can use the unreduced values of $\dot{f}_{i-1}$ and $\dot{f}_{i+1}$ when deciding how much to reduce $|\dot{f}_i|$. This may result in some derivatives being reduced more than they would have been by the preceding algorithm; however, it gives a completely local algorithm for constraining the derivatives.

We would like to be able to relax the constraints in the case shown in Figure 4.3 as we did for cubics, but we have not yet found a satisfactory way to do so. As it stands, the algorithm is therefore no better than second-order for nonmonotone data; we have not determined its order of convergence for monotone data. (We remind the reader that, for applications to sparse or nonsmooth data, the order of convergence is almost completely irrelevant.)

5. Convexity.

In the preceding sections, we presented algorithms to assign derivatives to a given data set that always yield a C^1 piecewise cubic or C^2 piecewise quintic interpolant that preserves monotonicity or positivity of the data. Unfortunately, there is no such algorithm for convexity. This can be easily seen by examining the function $f(x) = |x|$, $x \in [-1,1]$, on a mesh that includes the point 0. At $x_i = 0$, the backward derivative $\dot{f}_i^-$ must be equal to $S_{i-1/2} = -1$ to preserve convexity on $[x_{i-2}, x_i]$; the forward derivative $\dot{f}_i^+$ must equal $S_{i+1/2} = 1$ to preserve convexity on $[x_i, x_{i+2}]$. Thus, since $S_{i-1/2} \neq S_{i+1/2}$, the convex interpolant will not be differentiable at x_i.

We must therefore lower our expectations, accepting either nonconvex interpolants of convex data or nondifferentiable interpolants. This section gives algorithms for both of these options.

A. Cubic Polynomials.

1. *Convexity Constraints–Cubics.* The conditions on $\dot{f}_i^-$ and $\dot{f}_i^+$ ensuring that the cubic Hermite interpolant preserves convexity or concavity of the data are [15]

$$\rho_i S_{i-1/2} \leq \rho_i \dot{f}_i^- \leq \rho_i \dot{f}_i^+ \leq \rho_i S_{i+1/2} \tag{5.1}$$

and

$$-2\rho_i(\dot{f}_{i+1}^- - S_{i+1/2}) \leq \rho_i(\dot{f}_i^+ - S_{i+1/2}) \leq -\tfrac{1}{2}\rho_i(\dot{f}_{i+1}^- - S_{i+1/2}), \tag{5.2}$$

where

$$\rho_i = \begin{cases} 1 & \text{if } S_{i+1/2} \geq S_{i-1/2} \text{ (convex data)}, \\ -1 & \text{if } S_{i+1/2} < S_{i-1/2} \text{ (concave data)}. \end{cases}$$

The inequality (5.1) requires the slope at x_i to be between the slopes of the piecewise linear interpolant on either side of x_i and forces the jumps in $(Pf)'$ to be in the correct direction. The inequalities (5.2) are restrictions on $(Pf)''$ to ensure that it does not change sign in $[x_i, x_{i+1}]$.

Note that a solution to (5.1) and (5.2) exists since the piecewise linear interpolant ($\dot{f}_i^- = S_{i-1/2}, \dot{f}_i^+ = S_{i+1/2}$) satisfies these inequalities. The piecewise cubic interpolant, however, can be much smoother and more accurate while preserving convexity.

Combining (5.1) and (5.2) gives the more compact necessary conditions

$$L^-_{\min} \le \rho_i \dot{f}_i^- \le L^-_{\max} \tag{5.3a}$$

and

$$L^+_{\min} \le \rho_i \dot{f}_i^+ \le L^+_{\max}, \tag{5.3b}$$

where

$$L^-_{\min} = \max(\rho_i S_{i-1/2}, \tfrac{1}{2}\rho_i(3S_{i-1/2} - \dot{f}^+_{i-1})),$$
$$L^-_{\max} = \min(\rho_i(3S_{i-1/2} - 2\dot{f}^+_{i-1}), \rho_i \dot{f}_i^+),$$
$$L^+_{\min} = \max(\rho_i(3S_{i+1/2} - 2\dot{f}^-_{i+1}), \rho_i \dot{f}_i^-),$$

and

$$L^+_{\max} = \min(\rho_i S_{i+1/2}, \tfrac{1}{2}\rho_i(3S_{i+1/2} - \dot{f}^-_{i+1})).$$

If the underlying function has a continuous nonzero second derivative near a given point, (5.1) and (5.2) will be satisfied once the mesh is sufficiently refined. Let $x_- < x_+$ be two adjacent mesh points that approach the fixed point x as the mesh is refined, and let $S = (f(x_+) - f(x_-))/(x_+ - x_-)$. The differences $f'(x_-) - S$ and $f'(x_+) - S$ can be expressed as $-\frac{1}{2}(x_+ - x_-)f''(\eta_-)$ and $\frac{1}{2}(x_+ - x_-)f''(\eta_+)$, respectively, where $x_- < \eta_-, \eta_+ < x_+$; hence, once the mesh is sufficiently refined, these differences will have the correct signs to satisfy (5.1), and their ratio will be close enough to -1 for (5.2) to be satisfied.

2. *Convexity Algorithm–Cubics.* There is no local algorithm which produces a C^1 convex piecewise cubic Hermite interpolant whenever this is possible; it is easy to construct data where requirements (5.1) and (5.2) at i have consequences at distant j. An example of a nonlocal convexity-constraining algorithm is to first calculate accurate derivatives $\dot{f}$ at the data points, and then find the closest set of derivatives $\{\dot{f}^+\}$ and $\{\dot{f}^-\}$ to $\{\dot{f}\}$ that satisfy (5.3). That is, solve the optimization problem,

$$\min_{\dot{f}^+, \dot{f}^-} \|\dot{f}^- - \dot{f}\| + \|\dot{f}^+ - \dot{f}\|. \tag{5.4}$$

We want an interpolant to have $\dot{f}_i^- = \dot{f}_i^+$ whenever possible, to generate values $\dot{f}_i^-$ and $\dot{f}_i^+$ close to the original estimate $\dot{f}$, and to behave well for nonconvex data. Besides being computationally complex and expensive, the above algorithm has none of these properties. We now suggest a simpler and effective alternative algorithm.

As in the case of monotonicity, we cannot simply drop all restrictions when the data are not convex if we want a stable algorithm. Therefore, we replace requirement (5.2) with

$$\tfrac{1}{2}|\dot{f}^-_{i+1} - S_{i+1/2}| \le |\dot{f}_i^+ - S_{i+1/2}| \le 2|\dot{f}^-_{i+1} - S_{i+1/2}|. \tag{5.5}$$

We can now use (5.1) to obtain restrictions on $\dot{f}_i^+$ and $\dot{f}_i^-$ that do not depend on neighboring values:

$$\rho_i S_{i-1/2} \le \rho_i \dot{f}_i^+, \qquad \rho_i \dot{f}_i^- \le \rho_i S_{i+1/2}, \tag{5.6}$$

$$|\dot{f}_i^+ - S_{i+1/2}| \le 2|S_{i+3/2} - S_{i+1/2}|, \tag{5.7a}$$

and

$$|\dot{f}_i^- - S_{i-1/2}| \le 2|S_{i-3/2} - S_{i-1/2}|. \tag{5.7b}$$

At this point, we may have a problem: (5.7) may force $\dot{f}_i^+$ and $\dot{f}_i^-$ to be unequal. We must therefore decide whether to insist on convexity at the expense of differentiability (case 0) or to insist on differentiability at the expense of convexity (case 1). In case 0, we let (5.6) and (5.7) stand as the initial restrictions on $\dot{f}_i^+$ and $\dot{f}_i^-$; in case 1, we relax one or both of the inequalities (5.7) to make them satisfiable for some $\dot{f}_i^- = \dot{f}_i^+ = \dot{f}_i$.

We then have an initial interval of possible values for each derivative $\dot{f}_i^+, \dot{f}_i^-$. Next, we apply (5.5) in a sweep from $i = 1$ to $i = n$ to restrict these intervals further. That is, using the initial interval for $\dot{f}_1^+$ and (5.5), we compute a range of possible values for $\dot{f}_2^-$, which, when intersected with the old interval for $\dot{f}_2^-$, gives a new interval for $\dot{f}_2^-$. This then gives a new interval for $\dot{f}_2^+$, which will be the new interval for $\dot{f}_2^-$ intersected with the old interval for $\dot{f}_2^+$ unless, in case 0, we have been forced to allow $\dot{f}_2^-$ and $\dot{f}_2^+$ to differ. We now compute a new interval for $\dot{f}_3^-$ using (5.5), and so on.

Suppose that at some point we obtain an interval of negative length; let k be the i at which this happens. The region of data causing the problem can be localized by performing a backward sweep starting at k; at some i (define j to be this i), we will again find an empty interval. We now have two nonintersecting intervals for $\dot{f}_i^-$ or $\dot{f}_i^+$ for each i strictly between j and k; using these intervals, we modify one of the constraints from j to k, relaxing it just enough to make the new constraints from j to k satisfiable. In case 0, this is done by setting a certain amount by which $\dot{f}_i^-$ and $\dot{f}_i^+$ may differ; in case 1, we choose a positive number c and change (5.5) for some single value of i to

$$\tfrac{1}{2}|\dot{f}_{i+1}^- - S_{i+1/2}| - \tfrac{1}{2}c \le |\dot{f}_i^+ - S_{i+1/2}| \le 2|\dot{f}_{i+1}^- - S_{i+1/2}| + c. \tag{5.5$'$}$$

(Choosing a single constraint to relax is a source of instability in the algorithm, but this seems preferable to a stable algorithm in which one unsatisfiable set of constraints can cause a number of inflections in the interpolant.) Actually, sometimes relaxing two adjacent instances of (5.5) results in a more pleasing curve than does relaxing just one instance of (5.5); the programmer must choose.

Our method for choosing the constraint(s) to be relaxed in case 1 is as follows. For each i from j to $k-1$, let d_i^+ be the point in the forward-sweep interval at i which is closest to the backward-sweep interval at i, and let d_{i+1}^- be the point in the backward-sweep interval at $i+1$ which is closest to the forward-sweep interval at $i+1$. (For this purpose the "forward-sweep interval at k" is the interval computed by (5.5) from the forward-sweep interval at $k-1$, ignoring the original interval at k; the "backward-sweep interval at j" is treated analogously.) Using the value d_i^+ and d_{i+1}^-, we compute a value which represents the penalty for relaxing the constraint

(5.5) on the interval $[x_i, x_{i+1}]$. Currently we compute this penalty as follows. First, we compute an area. If d_i^+ and d_{i+1}^- are on the same side of $S_{i+1/2}$, this will be the area enclosed by the line segment L from (x_i, f_i) to (x_{i+1}, f_{i+1}) and the cubic Hermite curve through these points with slopes d_i^+ and d_{i+1}^-; otherwise, the area is that between this cubic Hermite curve and the tangent line at x_i or at x_{i+1}, depending on which of d_i^+ and d_{i+1}^- is closer to $S_{i+1/2}$. Next, we divide this area by the square root of the length of the segment L, because a "bulge" in the interpolant is visually less unpleasant if it is stretched out. Finally, we subtract a multiple of the old penalty if this constraint has already been relaxed; this tends to cause fewer constraints to be relaxed. (When computing the old penalty, we do not subtract off even older penalties.) The constraint chosen for relaxation is the one which gives the least penalty. However, if we can instead relax the constraints on two adjacent intervals $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$ so as to get a smaller total penalty (where the total penalty is computed as the sum of the penalty for d_{i-1}^+ and $\frac{1}{2}(d_i^- + d_i^+)$ on $[x_{i-1}, x_i]$ and the penalty for $\frac{1}{2}(d_i^- + d_i^+)$ and d_{i+1}^- on $[x_i, x_{i+1}]$), we do so. The algorithm for case 0 is similar except that we only consider relaxing one constraint, and the penalty, which is now a function of two slopes at the same point, is computed as the angle between the two slopes minus a multiple of the old penalty.

In any event, after a constraint is relaxed, we perform another forward sweep; this is repeated until a nonempty interval for $\dot{f}_n^-$ is obtained. (For reasonable data, very few iterations are needed; even the worst cases we examined required far fewer than the ostensible maximum of $\frac{1}{2}(n^2 - n) + 1$ iterations. Also, the repeat forward sweep can begin with the changed constraint rather than at $i = 1$.)

The resulting interval for $\dot{f}_n^-$ is the set of all possible values for $\dot{f}_n^-$ in a derivative assignment satisfying all of the current constraints. To find the corresponding intervals for the other derivatives, we perform a backward sweep using the intervals obtained from the final forward sweep. Finally, we choose the values to use from these intervals. A straightforward approach is to select some starting point i_0 and sweep backward and forward from it, at each point choosing derivatives as close as possible to the original estimates, but in the corresponding intervals, satisfying (5.5′) with respect to already chosen derivatives, and, in case 0, not allowing $\dot{f}_i^+$ and $\dot{f}_i^-$ to differ by more than the prescribed amount. We may choose i_0 to lie in a long range where the estimated derivatives already lie in the computed intervals; on the other hand, if we are strongly concerned with stability, we may wish to set $i_0 = n$ (with an added benefit: the backward sweep to compute the final intervals becomes superfluous).

An extra sweep may be performed in case 1 to detect situations where, because the data were highly nonconvex, two adjacent instances of (5.5) were independently relaxed, whereas relaxing one instance of (5.1) instead could bring better results. For example, consider $f(x) = |x - 1| - |x + 1| + 2x$, with data points at $x = -5, -3, -1, 0, 1, 3,$ and 5, and see Figure 5.1.

This algorithm puts a great deal of effort into deciding which constraints should be relaxed and by how much; therefore, it gives good results, but it is complicated and apparently may not run in linear time. An alternative which avoids these problems, but gives less pleasing curves in some cases, is to use a modification of the algorithm of Costantini and Morandi [5]. The basic idea here is that, when

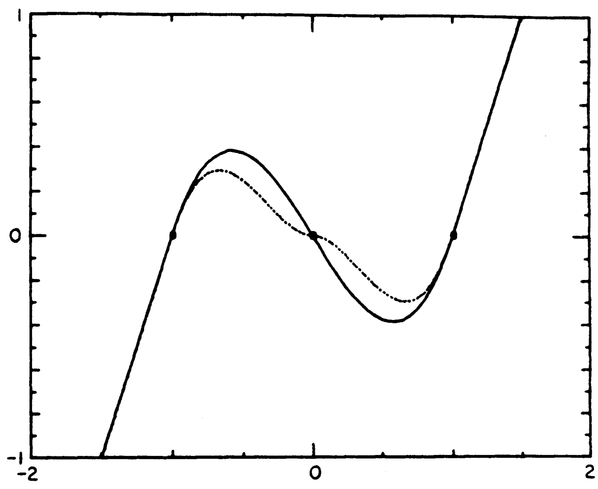


FIGURE 5.1
A case where (5.1) *should be relaxed. The dotted and solid curves show the constrained interpolant before and after the final sweep, respectively.*

it is discovered that the constraints are unsatisfiable, the most recently considered constraint should be relaxed. The new algorithm proceeds as follows in case 1 (case 0 is analogous). First, compute the initial intervals for $\dot{f}_i$ and perform a forward sweep as before. If this succeeds, continue as before; if it fails at $i = k_1$, set $\dot{f}_{k_1-1}$ to be that value from the forward-sweep interval which comes closest to making (5.5) on $[x_{k_1-1}, x_{k_1}]$ satisfiable, and sweep backward from here to get intervals for $\dot{f}_i$ $(1 \le i < k_1 - 1)$. Now start anew on the interval $[x_{k_1}, x_n]$; if the forward sweep here fails at k_2, set $\dot{f}_{k_2-1}$, get intervals for $\dot{f}_i$ for $k_1 \le i < k_2 - 1$, and proceed to $[x_{k_2}, x_n]$. Continue until every $\dot{f}_i$ has a value or an interval. Finally, for each k_m (including $k_0 = 1$), set $\dot{f}_{k_m}$ to be the value in its interval which comes closest to satisfying the preceding constraint (or to the original $\dot{f}_1$ if $m = 0$), and sweep forward to find the remaining values $\dot{f}_i$.

With Eric Van de Velde of New York University, we have developed a convexity-preserving algorithm for piecewise cubic parametric interpolants and are analyzing its effectiveness.

B. Quintic Polynomials.

1. *Convexity Constraints–Quintics.* As with monotonicity-preserving quintics, the region of values for $\dot{f}_i$, $\ddot{f}_i$, $\dot{f}_{i+1}$, and $\ddot{f}_{i+1}$ that give convexity-preserving quintics is far too complex to use in its entirety, but a judicious choice of rectangles within this region leads to a tractable algorithm for constraining the derivatives.

Consider a single interval $[x_i, x_{i+1}]$ with given values $f_i, \dot{f}_i, \ddot{f}_i, f_{i+1}, \dot{f}_{i+1}$, and $\ddot{f}_{i+1}$. By rescaling and possibly inverting one or both axes and subtracting a linear function, we may reduce the problem to the case where $x_i = 0$, $x_{i+1} = 1$, $f_i = f_{i+1} = 0$, and either $\dot{f}_i = \dot{f}_{i+1} = 0$ or $\dot{f}_i = -1$, $|\dot{f}_{i+1}| \le 1$. Clearly the only such convex quintic with $\dot{f}_i = \dot{f}_{i+1} = 0$ is the constant function 0 (that is, $\ddot{f}_i = \ddot{f}_{i+1} = 0$). Therefore, assume $\dot{f}_i = -1$. The convex region consisting of triples $(\dot{f}_{i+1}, \ddot{f}_i, \ddot{f}_{i+1})$ that give convex quintics is complicated, but we can verify that the quadrilaterals with corners $(1,0,0),(1,0,6),(1,6,0),(1,6,6)$ and $(\frac{1}{2},2,0),(\frac{1}{2},2,19/9),(\frac{1}{2},8,0),(\frac{1}{2},7,2)$ are contained within it. The point $(1-\sqrt{6}/3,4+\sqrt{6},0)$ is also in the region and gives the minimum possible value for $\dot{f}_{i+1}$ [7]. Hence, we find a square within the convexity region if we fix $\dot{f}_{i+1}=1$, a rectangle (with opposite corners $(\frac{1}{2},2,0)$ and $(\frac{1}{2},7,2)$) for $\dot{f}_{i+1}=\frac{1}{2}$, and a point for $\dot{f}_{i+1}=1-\sqrt{6}/3$; linear interpolation gives rectangles for intermediate values of $\dot{f}_{i+1}$.

2. *Convexity Algorithm–Quintics.* Although an algorithm giving a C^2 convex piecewise quintic interpolant, whenever such an interpolant exists, would be extremely complex and time-consuming, we now have the tools to construct a reasonable algorithm that gives good results in most cases. First, constrain the first derivatives $\dot{f}_i^+,\dot{f}_i^-$ using the cubic convexity algorithm. Next, consider the interval from x_i to x_{i+1}. By reversing one or both coordinate axes, we may assume $-\dot{f}_i^+\geq|\dot{f}_{i+1}^-|$. Next, find intervals in which $\ddot{f}_i^+$ and $\ddot{f}_{i+1}^-$ should lie. If $\dot{f}_i^+=0$, these intervals are both $[0,0]$; otherwise, they take the form $[cL_0^c(r),cU_0^c(r)]$ and $[cL_1^c(r),cU_1^c(r)]$, where $c=-\dot{f}_i^+/(x_{i+1}-x_i),r=-\dot{f}_i^-/\dot{f}_i^+$, and L_0^c,U_0^c,L_1^c,U_1^c are functions defined by linear interpolation of the following points:

$$\begin{array}{llllll}
L_0^c\colon & (-1,0) & (0,0) & (1-\sqrt{6}/3,4+\sqrt{6}) & (\frac{1}{2},2) & (1,0),\\
U_0^c\colon & (-1,15) & (0,15) & (1-\sqrt{6}/3,4+\sqrt{6}) & (\frac{1}{2},7) & (1,6),\\
L_1^c\colon & (-1,15) & (0,15) & (1-\sqrt{6}/3,0) & (\frac{1}{2},0) & (1,0),\\
U_1^c\colon & (-1,0) & (0,0) & (1-\sqrt{6}/3,0) & (\frac{1}{2},2) & (1,6).
\end{array}$$

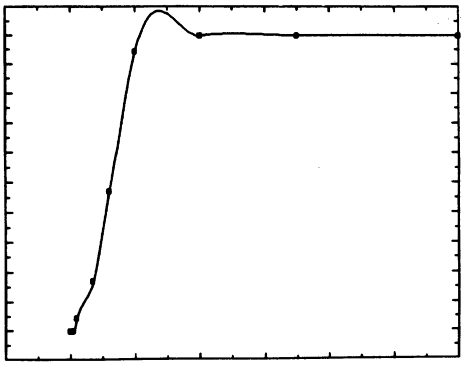
(The -1 and 0 values are chosen to increase the stability of the algorithm. The values 0 and 15 are not critical; the symmetry around $y=-x$ is.)

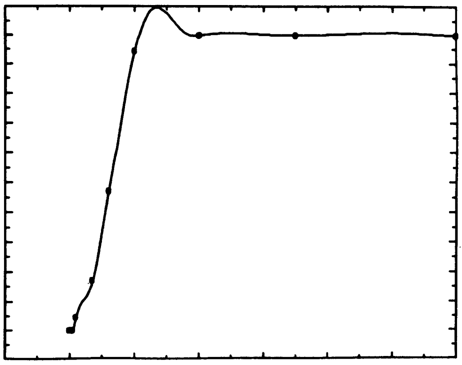
If the intervals for $\ddot{f}_i^+$ and $\ddot{f}_i^-$ intersect, set the constrained values for $\ddot{f}_i^+$ and $\ddot{f}_i^-$ to that point in the intersection closest to the original estimate for $\ddot{f}_i$. If the intervals do not intersect, but we insist on convexity rather than a C^2 interpolant, select $\ddot{f}_i^-$ and $\ddot{f}_i^+$ to be as close as possible to each other within their respective intervals. If we do insist on a C^2 interpolant, set $\ddot{f}_i=\ddot{f}_i^+=\ddot{f}_i^-$ to an average of the two endpoints of the intervals nearest each other. This average should not give equal weights to the intervals, as that would give unfortunate results for functions like $x+|x|$ near $x=0$; instead, we can compute a weight w for each interval as follows:

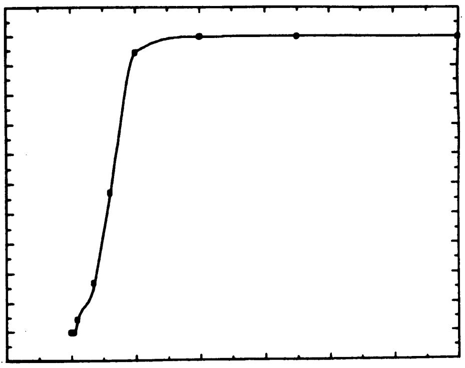
$$\begin{array}{ll}
[0,0]\ (\dot{f}_i^+=0)\colon & w^{-1}=0;\\
[cL_0^c(r),cU_0^c(r)]\colon & w^{-1}=c\max(0.1,U_0^c(r)-L_0^c(r));\\
[cL_1^c(r),cU_1^c(r)]\colon & w^{-1}=c(U_1^c(r)-L_1^c(r)).
\end{array}$$

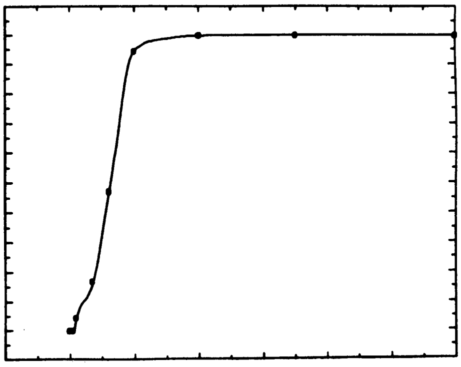
Of course, averaging numbers a_1 and a_2 with weights w_1 and w_2 is the same as averaging them with weights w_2^{-1} and w_1^{-1}, so we need not concern ourselves about one of the inverse weights w^{-1} being zero. They will not both be zero, because any interval with $w^{-1}=0$ contains the point 0, and we have assumed the two intervals do not meet.

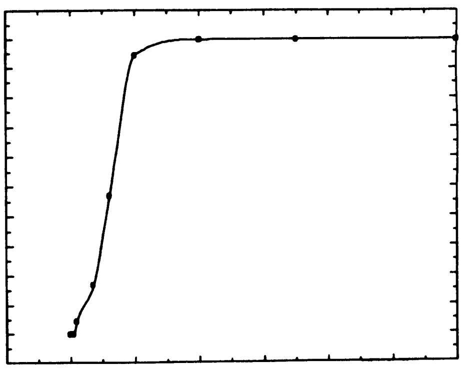
Nonintersecting intervals, however, will occur only for rough grids or for nonconvex or barely convex data. For convex data, if the first-derivative constrainer succeeds in satisfying (5.1) and (5.2) and the interval spacing does not vary too rapidly (specifically, no interval length $x_{i+1}-x_i$ is more than twice one of its neighbors), the convexity-preservation intervals for $\ddot{f}_i^+$ and $\ddot{f}_i^-$ will always intersect.


A. Unconstrained cubic


B. Unconstrained quintic


C. MC cubic


D. MC quintic


E. CC cubic

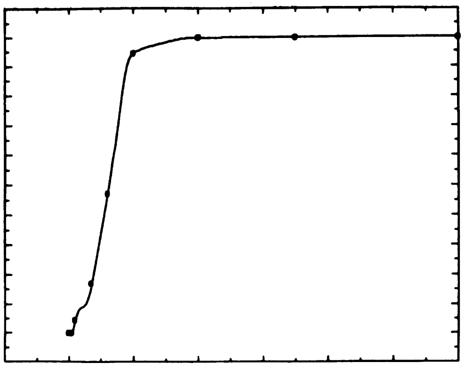
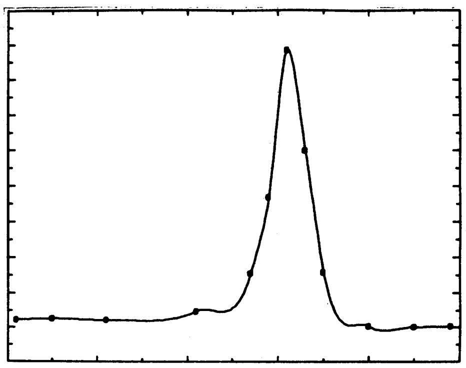
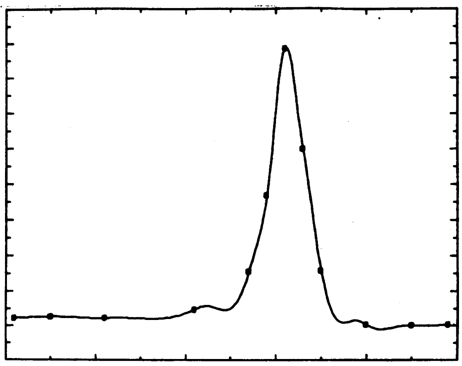
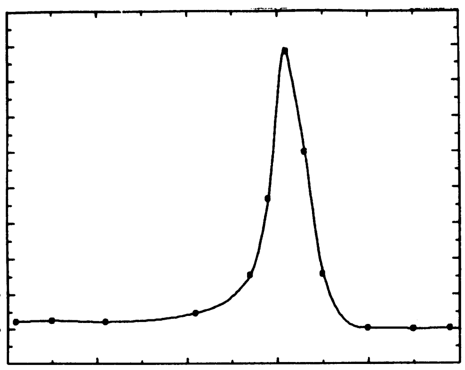
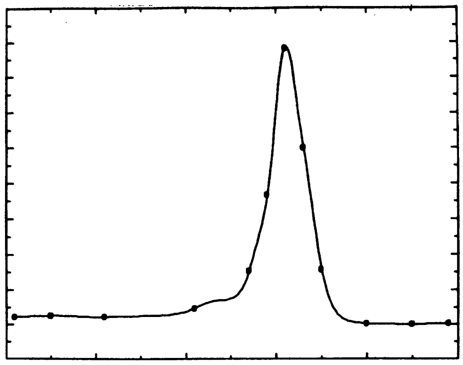

F. CC quintic

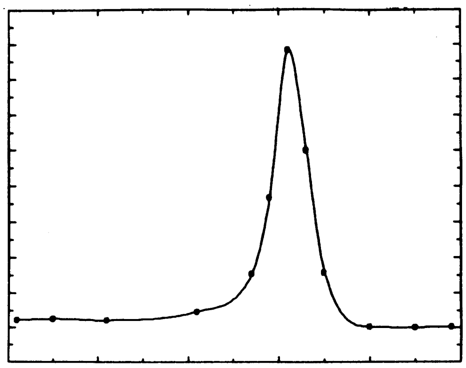
FIGURE 6.1
Interpolation curves for the RPN 14 *data in Table* 6.1.

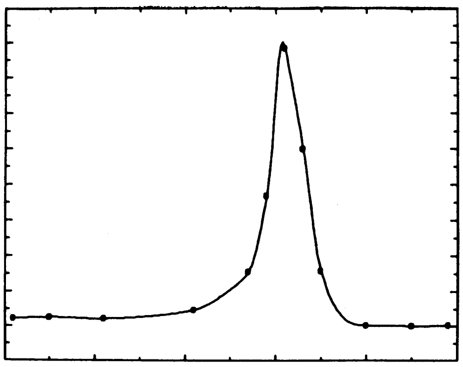

A. Unconstrained cubic


B. Unconstrained quintic


C. MC cubic


D. MC quintic


E. CC cubic


F. CC quintic

Interpolation curves for the titanium data in Table 6.1.

FIGURE 6.2

6. Numerical Examples.

In this section we compare the geometric properties and accuracy of the interpolants on both monotone and nonmonotone data sets. We use MC and CC to refer to interpolants obtained using derivatives that are constrained for monotonicity and for convexity, respectively. The original derivatives were obtained from second-order finite differences.

The Fritsch-Carlson RPN 14 radiochemical data [13] and the de Boor titanium equation-of-state data [1] have been used to compare many different algorithms. The data points are given in Table 6.1.

Figures 6.1 and 6.2 show the monotonicity- and convexity-constrained and unconstrained cubic and quintic Hermite interpolants. The constraints can convert a geometrically unacceptable interpolant, such as the cubic or quintic spline, into an excellent one.

We also compared the interpolation errors and convergence rates of the constrained and unconstrained interpolants of analytically defined functions. In the examples we ran on coarse meshes, the errors in the constrained interpolants were up to five times smaller than errors in unconstrained interpolants. When the mesh adequately resolved the underlying function, the constrained and unconstrained interpolants were identical except at a few isolated points.

TABLE 6.1

Data for Numerical Examples

RPN 14	Data	Titanium	Data
x	f	x	f
7.99	0	595	0.644
8.09	$2.76429E-5$	635	0.652
8.19	$4.37498E-2$	695	0.644
8.7	0.169183	795	0.694
9.2	0.469428	855	0.907
10	0.943740	875	1.336
12	0.998636	895	2.169
15	0.999919	915	1.598
20	0.999994	935	0.916
		985	0.607
		1035	0.603
		1075	0.608

7. Summary and Conclusions.

When geometric properties of a data set are important, the derivatives used for cubic and quintic piecewise polynomial interpolants should be constrained so that the resulting interpolant mimics any positivity, monotonicity, or convexity present in the data. Our two numerical examples illustrate the improved interpolated curves through rough data. When the data are smooth and the original derivative estimates accurate, the constraints are rarely needed. Thus, as the mesh is refined, the asymptotic convergence rate of the constrained interpolant is the same as that of the original unconstrained one, except near extrema or similar features of the function.

The algorithms we propose do not change the original derivative approximations by the least amount possible. Instead, they are designed to be effective and simple to implement. We also considered algorithms to project the original derivatives to the closest point within the shape-preservation region. The added complexity of this approach, in general, does not yield a significantly improved interpolant. One can find an "optimal" interpolant (in the sense of minimizing the changes in the original derivatives) subject to the shape-preservation constraints, using constrained optimization packages commonly available in computer software libraries. However, an interpolant which is as close as possible to a nonmonotone interpolant will be almost nonmonotone, in the sense of having nonextremal critical points; a similar statement holds for convexity. If one is willing to pay for expensive optimization methods, one should probably optimize a function which measures the geometric niceness of the interpolant.

Acknowledgments. We thank Blair Swartz for his comments and advice during our many discussions. Also, we are grateful to Sandria Kerr for testing and documenting the piecewise Hermite interpolation subroutine package used in the numerical examples.

Center for Nonlinear Studies
Theoretical Division, MS B284
Los Alamos National Laboratory
Los Alamos, New Mexico 87545
E-mail: jh@lanl.gov

1. C. DE BOOR, *A Practical Guide to Splines*, Springer-Verlag, New York, 1978.

2. J. BUTLAND, "A method of interpolating reasonable-shaped curves through any data," *Computer Graphics* 80 (R.J. Lansdown, ed.), Online Publications, Northwood Hills, Middlesex, 1980, pp. 409–422.

3. C. DE BOOR & B. SWARTZ, "Piecewise monotone interpolation," *J. Approx. Theory*, v. 21, 1977, pp. 411–416.

4. R. E. CARLSON & F. N. FRITSCH, "Monotone piecewise bicubic interpolation," *SIAM J. Numer. Anal.*, v. 22, 1985, pp. 386–400.

5. P. COSTANTINI & R. MORANDI, "Monotone and convex cubic spline interpolation," *Calcolo*, v. 21, 1984, pp. 281–294; and "An algorithm for computing shape-preserving cubic spline interpolation to data," *Calcolo*, v. 21, 1984, pp. 295–305.

6. B. DIMSDALE, "Convex cubic splines, " *IBM J. Res. Develop.*, v. 22, 1978, pp. 168–178.

7. A. EDELMAN & C. A. MICCHELLI, "Admissible slopes for monotone and convex interpolation," *Numer. Math.*, v. 51, 1987, pp. 441–458.

8. S. C. EISENSTAT, K. R. JACKSON & J. W. LEWIS, "The order of monotone piecewise cubic interpolation," *SIAM J. Numer. Anal.*, v. 22, 1985, pp. 1220–1237.

9. J. C. FERGUSON, *Shape Preserving Parametric Cubic Curve Interpolation*, Ph. D. thesis, University of New Mexico, 1984.

10. J. C. FERGUSON & K. MILLER, *Characterization of Shape in a Class of Third Degree Algebraic Curves*, TRW report 5322-3-5, 1969.

11. F. N. FRITSCH, *Use of the Bernstein Form to Derive Sufficient Conditions for Shape Preserving Piecewise Polynomial Interpolation*, Lawrence Livermore National Laboratory report UCRL-91392, March 1984.

12. F. N. FRITSCH & J. BUTLAND, "A method for constructing local monotone piecewise cubic interpolants", *SIAM J. Sci. Statist. Comput.*, v. 5, 1984, pp. 300–304.

13. F. N. FRITSCH & R. E. CARLSON, "Monotone piecewise cubic interpolation," *SIAM J. Numer. Anal.*, v. 17, 1980, pp. 238–246.

14. J. M. HYMAN, "Accurate monotonicity preserving cubic interpolation," *SIAM J. Sci. Statist. Comput.*, v. 4, 1983, pp. 645–654.

15. J. M. HYMAN, *Accurate Convexity-Preserving Cubic Interpolation*, informal report, Los Alamos Scientific Laboratory document, LA-UR-80-3700, Los Alamos, NM, November 1980.

16. J. M. HYMAN & B. LARROUTUROU, "The numerical differentiation of discrete functions using polynomial interpolation methods," *Numerical Grid Generation for Numerical Solution of Partial Differential Equations* (J.F. Thompson, ed.), Elsevier North-Holland, New York, 1982, pp. 487–506.

17. D. F. MCALLISTER, E. PASSOW & J. A. ROULIER, "Algorithms for computing shape preserving spline interpolations to data," *Math. Comp.*, v. 31, 1977, pp. 717–725.

18. D. F. MCALLISTER & J. A. ROULIER, "An algorithm for computing a shape preserving oscillatory quadratic spline," *ACM Trans. Math. Software*, v. 7, 1982, pp. 331–347.

19. D. F. MCALLISTER & J. A. ROULIER, "Interpolation by convex quadratic splines," *Math. Comp.*, v. 32, 1978, pp. 1154–1162.

20. H. METTKE, "Convex cubic Hermite-spline interpolation," *J. Comput. Appl. Math.*, v. 9, 1983, pp. 205–211, and v. 11, 1984, pp. 377–378.

21. E. NEUMAN, "Convex interpolating splines of arbitrary degree II," *BIT*, v. 22, 1982, pp. 331–338.

22. E. PASSOW & J. A. ROULIER, "Monotonic and convex spline interpolation," *SIAM J. Numer. Anal.*, v. 14, 1977, pp. 904–909.