SIAM J. SCI. STAT. COMPUT.
Vol. 4, No. 4, December 1983

ACCURATE MONOTONICITY PRESERVING CUBIC INTERPOLATION*

JAMES M. HYMAN†

Abstract. A simple and effective algorithm to construct a monotonicity preserving cubic Hermite interpolant for data with rapid variations is presented. Constraining the derivatives of the interpolant according to geometric considerations makes the interpolant consistent with local monotonicity properties of the data. Numerical examples are given that compare the quality and accuracy of the proposed interpolation method with other standard interpolants.

Key words. approximation theory, interpolation, monotonicity, numerical analysis, shape preservation, spline

1. Introduction. Piecewise polynomial interpolation is used to deduce probable values for an implied function defined at a discrete set of points. For an accurate interpolation, we must carefully retain crucial properties of the data (such as monotonicity or convexity), and we must not introduce details or artifacts that cannot be ascertained from the data. This requires that the interpolant be designed according to both geometric and algebraic considerations.

The geometric qualities of an interpolant are based on how well the interpolated curve reflects the intrinsic shape inferred by the data points. A good geometric interpolant will produce a curve similar to the visually pleasing one of a draftsman. Although some mathematical properties, such as those that preserve monotonicity and convexity, can describe the goodness of fit in a geometric sense, choosing the better curve often is a heuristic decision based on human judgment rather than on firm mathematical theory.

A good geometric interpolant is most important when the data arise from a physical experiment and an underlying mathematical structure does not exist. For these data sets, geometric considerations such as preventing spurious behavior near rapid changes in the data may be more important than the method's asymptotic accuracy. In fact, maintaining monotonicity or convexity in the interpolation process may be necessary to represent physical reality. For example, if the data come from an equation-of-state table for density versus pressure, then a nonmonotone interpolant will have a negative derivative and will imply an imaginary sound speed for the material [10]. This error can destroy the accuracy of any calculation based on the interpolated data.

The functional or algebraic properties of a good interpolant in classic approximation theory are defined more precisely [2]. They include the order of accuracy as the mesh spacing becomes arbitrarily small, continuity or smoothness of some derivative of the interpolant, and invariance or linearity properties such as

$$P(\alpha f+g)=\alpha Pf+Pg, \tag{1.1}$$

where P is the interpolation operator, α is a scalar and f and g are functions. Note that none of these properties guarantees a good geometric interpolant.

Another consideration is a method's practicality, as evidenced by its simplicity, efficiency, and storage requirements. In this report, we restrict our analysis to local

* Received by the editors March 26, 1981, and in revised form September 20, 1982.

† Center for Nonlinear Studies, Theoretical Division, MS B284, Los Alamos National Laboratory, Los Alamos, New Mexico 87545.

piecewise polynomial interpolants that, when compared to other methods, rate high in these three categories.

We have developed and tested a practical algorithm that is excellent both geometrically and algebraically and is highly accurate when monotonicity properties of the interpolant do not intervene. Loss of accuracy in favor of shape preservation occurs only at isolated points where the grid is rough compared to the solution variation.

First, we briefly describe the piecewise polynomial cubic Hermite interpolant and the restrictions sufficient to guarantee monotonicity. We then describe some possible algorithms that compute the derivatives needed by the interpolant at the mesh points, and we give numerical examples that compare our method to similar methods.

2. Cubic Hermite interpolation. Let the mesh $\{x_i\}_{i=1}^n$ be a partition $x_1 < x_2 < \cdots < x_n$ of 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 monotone at* x_i if $S_{i+1/2}S_{i-1/2} > 0$. The interpolant is *piecewise monotone* if $(Pf)(x)$ is monotone between f_i and f_{i+1} for x between x_i and x_{i+1}. The interpolant Pf is *class* C^k if $(Pf)(x)$ is continuous and has continuous derivatives for all orders less than or equal to k.

A. The interpolation formula. Given the data points $\{f_i\}$, a numerical approximation of the slope $\dot{f}_i$ at x_i is calculated for $1 \leqq i \leqq n$. The cubic Hermite interpolant then is defined for $1 \leqq i < n$ as

$$P(x) = c_1 + (x - x_i)c_2 + (x - x_i)^2 c_3 + (x - x_i)^3 c_4, \tag{2.1}$$

where $x_i \leqq x \leqq x_{i+1}$,

$$c_1 = f_i, \qquad c_2 = \dot{f}_i,$$

$$c_3 = \frac{3S_{i+1/2} - \dot{f}_{i+1} - 2\dot{f}_i}{\Delta x_{i+1/2}}, \qquad c_4 = -\frac{2S_{i+1/2} - \dot{f}_{i+1} - \dot{f}_i}{\Delta x_{i+1/2}^2}.$$

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 $\{\dot{f}_i\}$ are calculated.

Note that once $\{\dot{f}_i\}$ are given, (2.1) becomes a local interpolation formula. By changing the value of f_i or $\dot{f}_i$ at a data point, the interpolant changes only in the region $[x_{i-1}, x_{i+1}]$. If the calculation of $\dot{f}$ also is 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 as is the case for very large data sets or multidimensional interpolation.

Localness of the interpolant also is desirable when data are being readjusted a few points at a time. This occurs in interactive graphics routines to avoid recalculating the interpolation function at all data points.

The numerical approximation of $\{\dot{f}_i\}$ which makes (2.1) a C^2 interpolant (for example, the complete spline interpolant[2]), is not local. Thus, to gain total localness for (2.1), we must sacrifice global continuity in the second derivative.

B. Monotonicity. Even when $\{\dot{f}_i\}$ are defined accurately, additional constraints may be necessary because (2.1) may fail to produce an acceptable interpolant in the geometric sense for certain data sets. A simple generalization of what was recognized by de Boor and Swartz [3] is that if the data are locally monotonically increasing at

x_i, and if

$$0 \leqq \dot{f}_j \leqq 3 \min (S_{j-1/2}, S_{j+1/2}) \tag{2.2}$$

for $j = i$ or $i+1$, then the resulting interpolant is monotone in $[x_i, x_{i+1}]$. Fritsch and Carlson [5] independently found an extension of this criteria giving 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.

Note that if $\{\dot{f}_i\}$ are calculated to make the resulting interpolant C^2, then (2.2) may not be satisfied. That is, there are monotone data sets for which there is no C^2 piecewise cubic Hermite interpolant.

When the data are locally monotone, we restrict $\{\dot{f}_i\}$ to the de Boor–Swartz piecewise monotonicity range of (2.2) as follows. After calculating an accurate approximation of $\dot{f}_i$ (for instance, finite differences or by the complete spline formula), we project it to the allowed monotonicity region according to

$$\dot{f}_i \leftarrow \begin{cases} \min [\max (0, \dot{f}_i), 3S^i_{\min}] & \text{if } 0 < S^i_{\min}, \\ \max [\min (0, \dot{f}_i), 3S^i_{\max}] & \text{if } 0 > S^i_{\max}, \\ 0 & \text{if } 0 \geqq S_{i-1/2}S_{i+1/2}, \end{cases} \tag{2.3}$$

where

$$S^i_{\min} = \min (S_{i-1/2}, S_{i+1/2}), \qquad S^i_{\max} = \max (S_{i-1/2}, S_{i+1/2}).$$

Near the boundary, the de Boor–Swartz constraint can be used by letting $S_{-1/2} = S_{1/2}$ and $S_{n+1/2} = S_{n-1/2}$.

When an $\dot{f}_i$ associated with a complete spline interpolant falls outside the range of (2.2), as it inevitably will when the variation between the data points is large, resetting $\dot{f}_i$ according to (2.3) will cause the second derivative of the interpolant to jump where $\dot{f}_i$ was reset and at the two nearest mesh points.

If the underlying function is strictly monotone and sufficiently smooth, and $\dot{f}_i$ is an accurate approximation to the derivative at x_i, then as the mesh is refined, (2.2) will be satisfied in the limit, because

$$\left.\frac{df}{dx}\right|_{x=x_i} = \dot{f}_i + O(\Delta x^?) = S_{i+1/2} + O(\Delta x) = S_{i-1/2} + O(\Delta x). \tag{2.4}$$

Thus, the interpolant is restricted by geometric considerations only when the mesh is coarse and the asymptotic accuracy in $\dot{f}$ is meaningless. When the mesh accurately resolves the function implied by the data, the accuracy in $\dot{f}$ is retained because (2.2) will be satisfied.

When the data are not locally monotone, the interpolant also must have an extrema. Retaining piecewise monotonicity would require that $\dot{f}_i = 0$ and would "clip" the interpolant by forcing inter-interval monotonicity on nonmonotone data. However, the piecewise monotonicity constraint can be relaxed in the interval pair next to the extrema to produce (in the author's opinion) a more visually pleasing curve. But if a new constraint is 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 (2.3) elsewhere, the resulting interpolant will be unstable.

We chose to extend (2.2) by requiring that $\dot{f}_i$ have the same sign as originally calculated, and that

$$|\dot{f}_i| \leqq 3 \min(|S_{i-1/2}|, |S_{i+1/2}|). \tag{2.5}$$

The constraining function extending (2.3) is

$$\dot{f}_i \leftarrow \begin{cases} \min[\max(0, \dot{f}_i), 3\min(|S_{i-1/2}|, |S_{i+1/2}|)], & \sigma > 0, \\ \max[\min(0, \dot{f}_i), -3\min(|S_{i-1/2}|, |S_{i+1/2}|)], & \sigma < 0, \end{cases} \tag{2.6}$$

where $\sigma = \operatorname{sign}(\dot{f}_i)$. The sign function sign $(S) = 1$ if $S \geqq 0$ and -1 otherwise.

Often the monotonicity of the interpolant's derivative is an important quality that can be incorporated into the interpolant; if the data are convex, a good geometric interpolant should preserve this convexity. However, a C^1 convexity preserving cubic Hermite interpolant does not exist for all data sets [11]. For example, a C^1 convex interpolant does not exist for $f = x + |x|$ when $x = 0$ is a data point. If the C^1 constraint is dropped, restrictions similar to (2.6) can be incorporated in (2.1) to preserve convexity [7].

A simple, necessary but not sufficient, and often effective convexity preserving constraint involves limiting the $\{\dot{f}_i\}$ so that

$$\min(S_{i-1/2}, S_{i+1/2}) \leqq \dot{f}_i \leqq \max(S_{i-1/2}, S_{i+1/2}) \tag{2.7}$$

by using

$$\dot{f}_i \leftarrow \max\{\min[\dot{f}_i, \max(S_{i-1/2}, S_{i+1/2})], \min(S_{i-1/2}, S_{i+1/2})\}. \tag{2.8}$$

3. Derivative approximation. The order of accuracy of (2.1) can be, at best, one order higher than the order of accuracy of $\dot{f}_i$. Therefore, it is prudent to calculate $\dot{f}_i$ accurately whenever possible. The difference approximations can be divided into two classes; local and nonlocal. The local schemes use only f values near x_i to calculate $\dot{f}_i$. The nonlocal schemes use all $\{f_i\}$ values and obtain $\{\dot{f}_i\}$ by solving a linear system of equations.

A. Local methods. De Boor and Swartz have shown that there are no linear algorithms yielding derivative approximations above first-order that also automatically satisfy (2.2). There are, however, many nonlinear formulas that do. The Butland [1] algorithm, for example, yields $\{\dot{f}_i\}$ which automatically satisfies (2.2) and is second-order on a uniform grid. The Fritsch–Butland [4] algorithm listed in Table 1 is a slight modification of this formula.

The parabolic interpolation method in Table 1 is linear, and the resulting $\{\dot{f}_i\}$ do not automatically satisfy (2.2). This formula can be multiplied by a nonlinear factor with magnitude $1 + O(\Delta x^2)$ to give the monotonicity preserving formula

$$\dot{f}_i = \frac{(2+\theta)S_{i+1/2}S_{i-1/2}}{S^2_{i+1/2} + S^2_{i-1/2} + \theta S_{i+1/2}S_{i-1/2}} \; \frac{\Delta x_{i-1/2}S_{i+1/2} + \Delta x_{i+1/2}S_{i-1/2}}{\Delta x_{i-1/2} + \Delta x_{i-1/2}}. \tag{3.1}$$

This formula is second-order for monotone data when

$$-2 < \theta \leqq 1 + 3\min\left(\frac{\Delta x_{i+1/2}}{\Delta x_{i-1/2}}, \frac{\Delta x_{i-1/2}}{\Delta x_{i+1/2}}\right).$$

The tests for the resulting interpolant with $\theta = 1$ are not included in this report, but they are very similar to and only slightly less accurate than the monotonicity constrained parabolic interpolant.

TABLE 1
Local formulas for $\dot{f}_i$ and their order of accuracy for smooth functions and mesh variations.

Method	Formula for $\dot{f}_i$	Order of accuracy
Akima[a]	$\dfrac{\lvert S_{i+3/2}-S_{i+1/2}\rvert S_{i-1/2}+\lvert S_{i-1/2}-S_{i-3/2}\rvert S_{i+1/2}}{\lvert S_{i+3/2}-S_{i+1/2}\rvert+\lvert S_{i-1/2}-S_{i-3/2}\rvert}$	$O(\Delta x^2)$
Fritsch–Butland[b]	$\dfrac{3S^i_{\min}S^i_{\max}}{S^i_{\max}+2S^i_{\min}}$	$O(\Delta x)$
Parabolic[c]	$\dfrac{\Delta x_{i-1/2}S_{i+1/2}+\Delta x_{i+1/2}S_{i-1/2}}{x_{i+1}-x_{i-1}}$	$O(\Delta x^2)$
Fourth-order finite difference[d]	$\dfrac{-f_{i+2}+8f_{i+1}-8f_{i-1}+f_{i-2}}{-x_{i+2}+8x_{i+1}-8x_{i-1}+x_{i-2}}$	$O(\Delta x^4)$

[a] See [2]. [b] See [4]. [c] See [9]. [d] See [9].

The fourth-order finite difference method [9] in Table 1 is based on first mapping $\{x_i\}$ to a standard equally spaced reference grid $\{r_i\}$, and then approximating each term in the identity

$$\frac{df}{dx}=\frac{df}{dr}\left(\frac{dx}{dr}\right)^{-1}, \tag{3.2}$$

with centered fourth-order finite differences for an equally spaced grid. If the mapping from x into r is not sufficiently smooth (that is, certain high derivatives of the map are not bounded independent of n), the order of accuracy of the method will be reduced accordingly. Thus, the finite difference formula is fourth-order on a smoothly varying mesh, but only first-order on a rougher mesh and, in fact, could become singular on a mesh having a local mesh ratio greater than 3.5. When this is the case, the parabolic method is to be preferred.

The Akima [2] and Fritsch–Butland formulas are nonlinear algorithms for each $\dot{f}_i$. Consequently, the sum of the interpolants for data sets (x, f) and (x, g) is different from the interpolant for the sum of the data sets $(x, f+z)$. The other formulas do not have this defect in their initial approximations of $\dot{f}_i$. However, after filtering according to (2.6), none of the interpolants are additive in the sense of (1.1) for $f=x+|x|$, $g=x-|x|$ when $x=0$ is a data point.

B. Nonlocal methods. The most common nonlocal method for computing $\dot{f}_i$ is the "not-a-knot" C^2 spline interpolation method [2], where $\dot{f}_i$ is $O(\Delta x^3)$ on an unequally spaced mesh or $O(\Delta x^4)$ away from the boundary on an equally spaced mesh. The $\dot{f}_i$'s are calculated so that the resulting interpolant has a continuous second derivative at the knots. As mentioned in § 2 the C^2 spline interpolant will not necessarily satisfy (2.2) for monotone data. By allowing isolated discontinuities in the second derivative, a slightly deficient C^1 monotone spline interpolant can be constructed in various ways. A possible solution is to filter the C^2 spline $\{\dot{f}_i\}$ according to (2.6). This is the approach taken in the numerical examples presented here.

An algorithm that keeps the number of jumps in the second derivative small involves first computing $\{\dot{f}_i\}$ for the complete spline interpolant in the interval $[x_1, x_n]$. If the interpolant is not locally monotone, we locate point x_j where $\dot{f}_j$ is farthest outside the monotonicity region. We redefine $\dot{f}_j$ according to (2.6) and solve for the complete spline interpolant in $[x_1, x_j]$ and $[x_j, x_n]$, using f_j and $\dot{f}_i$ as boundary conditions. The resulting interpolant will have a break in the second derivative only at x_j. If none of the resulting $\{\dot{f}_i\}$ violate (2.6), we are finished. If some do violate (2.6), we repeat the process, break $[x_1, x_j]$ or $[x_j, x_n]$ into smaller subregions and continue. This algorithm will always terminate if $\dot{f}_1$ and $\dot{f}_n$ are given at the boundaries and satisfy (2.6).

Another nonlocal approximation to $\dot{f}$ is the Fritsch–Carlson algorithm [5]. This method provides an approximation of $\dot{f}$ that preserves piecewise monotonicity in (2.1) with curves geometrically similar to those produced by the Fritsch–Butland method. Although we have not compared the Fritsch–Carlson method to the other methods in this report, we have included an example of specific data from their paper [5]. From this example and other similar ones from their paper, we have found that the monotonicity constrained algorithms using (2.3) or (2.6) perform very similarly to the Fritsch–Carlson algorithm. The major difference is that the constraints (2.3) and (2.6) are much easier to implement. Also the behavior of the (2.6) constrained interpolant at extrema in nonmonotone data sets is different. The Fritsch–Carlson algorithm clips the interpolant like the constraint (2.3) does.

C. Boundaries. At the boundaries we will use either the not-a-knot option for splines [2] or an uncentered difference approximation. The second-order uncentered parabolic method used with the Akima, Fritsch–Butland, and parabolic algorithms is

$$\dot{f}_i = \frac{(2\,\Delta x_{i+1/2} + \Delta x_{i+3/2})S_{i+1/2} - \Delta x_{i+1/2} S_{i+3/2}}{\Delta x_{i+1/2} + \Delta x_{i+3/2}}$$

or

$$\dot{f}_i = \frac{(2\,\Delta x_{i-1/2} + \Delta x_{i-3/2})S_{i-1/2} - \Delta x_{i-1/2} S_{i-3/2}}{\Delta x_{i-1/2} + \Delta x_{i-3/2}}.$$

The third-order uncentered finite difference approximations used with the fourth-order interior formula are

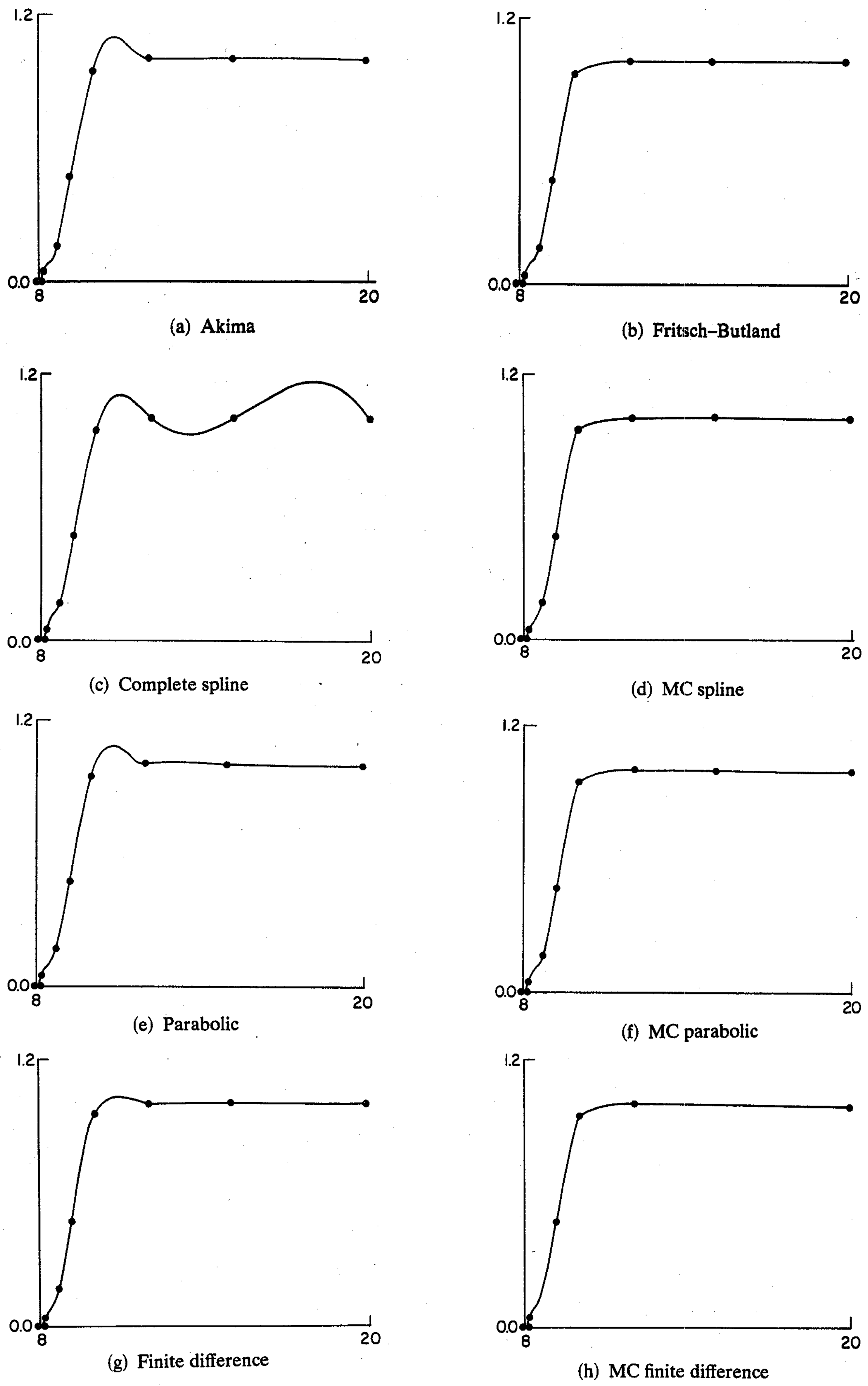
$$\dot{f}_i = \frac{-22f_i + 36f_{i+1} - 18f_{i+2} + 4f_{i+3}}{-22x_i + 36x_{i+1} - 18x_{i+2} + 4x_{i+3}},$$

$$\dot{f}_i = \frac{-2f_{i-1} - 3f_i + 6f_{i+1} - f_{i+2}}{-2x_{i-1} - 3x_i + 6x_{i+1} - x_{i+2}},$$

$$\dot{f}_i = \frac{22f_i - 36f_{i-1} + 18f_{i-2} - 4f_{i-3}}{22x_i - 36x_{i-1} + 18x_{i-2} - 4x_{i-3}},$$

$$\dot{f}_i = \frac{2f_{i+1} + 3f_i - 6f_{i-1} + f_{i-2}}{2x_{i+1} + 3x_i - 6x_{i-1} + x_{i-2}}.$$

4. Numerical examples. The geometric and accuracy properties of the interpolants are compared on both smoothly varying and rough data sets. When the derivatives are constrained by the extended de Boor–Swartz monotonicity limit (2.6), we call the resulting interpolant monotonically constrained (MC).



(a) Akima (b) Fritsch–Butland (c) Complete spline (d) MC spline (e) Parabolic (f) MC parabolic (g) Finite difference (h) MC finite difference

FIG. 1. *Interpolation curves for the* RPN 15A *data.*

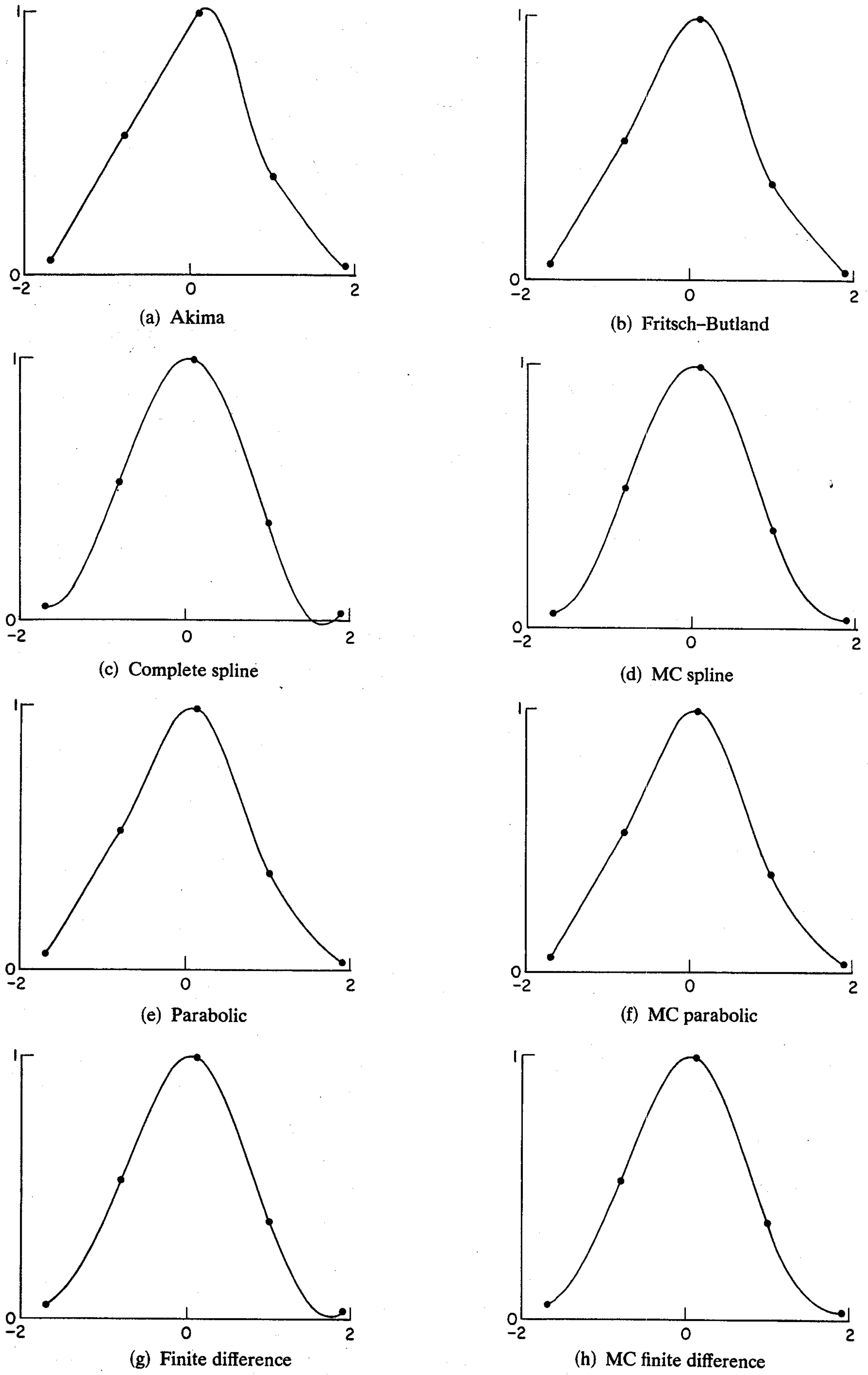


FIG. 2. *Interpolation curves for $f(x) = e^{-x^2}$.*

A. Monotone example. The Fritsch–Carlson RPN 15A data have been used to compare many different algorithms [4], [5], [8], some not included in this report. The data points are

x	f
7.99	0
8.09	2.76429E−5
8.19	4.37498E−2
8.7	0.169183
9.2	0.469428
10	0.943740
12	0.998636
15	0.999919
20	0.999994

Figure 1 shows interpolation curves of the Akima, Fritsch–Butland, parabolic, fourth-order finite difference, complete spline, MC parabolic, MC fourth-order finite difference, and MC spline methods. These data show clearly that the Fritsch–Butland and MC methods are geometrically superior to the unconstrained methods. The simple constraint of (2.6) can convert an unacceptable geometric interpolant, such as the complete spline, into an excellent one. Note that the Akima algorithm, which was designed as a good geometric interpolant, fails to preserve monotonicity in this relatively simple example.

B. Nonmonotone example. To interpolate the monotone function $f(x)=e^{-x^2}$, $x\varepsilon[-1.7, 1.9]$, the mesh was equally spaced with $\Delta x=3.6/(n-1)$. This domain was chosen so the mesh points would not be symmetrical about the point of symmetry for the function.

Figure 2 shows that the higher order MC interpolants can be geometrically superior to the Akima, Fritsch–Butland, parabolic and the unconstrained interpolants when $n=5$.

TABLE 2
Comparison of the methods for $f(x)=e^{-x^2}$.

Method	L_2 error			
	$n=5$	$n=9$	$n=17$	$n=33$
Akima	6.0E−2	6.4E−3	1.0E−3	1.3E−4
Fritsch–Butland	4.4E−2	7.3E−3	2.4E−3	1.6E−4
Parabolic method	3.9E−2	4.1E−3	4.1E−5	4.3E−5
Finite difference	2.2E−2	3.4E−3	7.4E−5	2.3E−6
Complete spline	3.5E−2	2.0E−3	4.0E−5	1.8E−6
MC parabolic	3.9E−2	4.1E−3	1.9E−3	4.3E−5
MC finite difference	1.5E−2	3.4E−3	1.9E−3	2.3E−6
MC spline	1.7E−2	2.0E−3	1.9E−3	1.8E−6

In Table 2, the errors,

$$L_2 \text{ error} = \left(\int_{-1.7}^{1.9} [(Pf)(x)-e^{-x^2}]^2\right)^{1/2},$$

of the interpolants are compared as the mesh is refined. Note that the higher order MC finite difference and spline methods are more accurate than the other methods both on the fine and coarse grids.

The MC methods agree with the corresponding unconstrained methods when $n = 9$ and 33, but not when $n = 17$, because the nonmonotonicity of the underlying function is being interpolated.

5. Summary and conclusions.

When only geometric considerations are important, any interpolant constrained to stay within the de Boor–Swartz monotonicity limits using algorithm (2.6) is acceptable. The Fritsch–Butland does this automatically (that is, there are no conditional statements) but the approximating derivatives in other procedures must be filtered.

When both geometric and accuracy considerations are important, the lower order methods (Akima, Fritsch–Butland, and parabolic) have larger truncation errors than the higher order constrained methods.

Therefore, we recommend first computing an approximation $\dot{f}_i$ to df/dx at the mesh points, using either the local fourth-order finite difference method (Table 1) or the nonlocal, but smoother, complete spline approximation. Before interpolating using (2.1), we filter $\{\dot{f}_i\}$ with (2.6), so the interpolant will retain the important local monotonicity properties of the data.

The simplicity of the filtering approach and the dramatic improvements in the interpolation curve far outweigh the cost of the extra few lines of code. Analysis of our numerical examples indicates that most cubic Hermite interpolation programs would be more versatile, robust, and often more accurate, if a monotonicity constraint such as (2.6) were an option.

Acknowledgment. I am grateful to Blair Swartz for providing me with much welcome advice in our many discussions during this work.

REFERENCES

[1] J. BUTLAND, *A method of interpolating reasonable-shaped curves through any data*, Proc. Computer Graphics 80, Online Publications Ltd., Northwood Hills, Middlesex, England, 1980, pp. 409–422.

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

[3] C. DE BOOR AND B. SWARTZ, *Piecewise monotone interpolation*, J. Approx. Theory, 21 (1977), pp. 411–416.

[4] F. N. FRITSCH AND J. BUTLAND, *An improved monotone piecewise cubic interpolation algorithm*, Lawrence Livermore National Laboratory preprint UCRL-85104, 1980.

[5] F. N. FRITSCH AND R. E. CARLSON, *Monotone piecewise cubic interpolation*, SIAM J. Numer. Anal., 17 (1980), pp. 238–246.

[6] F. N. FRITSCH AND R. E. CARLSON, *Piecewise cubic interpolation methods*, Lawrence Livermore National Laboratory preprint UCRL-81230, 1978.

[7] J. M. HYMAN, *Accurate convexity preserving cubic interpolation*, informal report, Los Alamos Scientific Laboratory, Los Alamos, NM, November 1980.

[8] ———, *Accurate cubic piecewise monotone interpolation*, Los Alamos Scientific Laboratory, Los Alamos, NM, LA-UR-80-3700, 1980.

[9] ———, *The numerical solution of time dependent* PDEs *on an adaptive mesh*, Los Alamos Scientific Laboratory, Los Alamos, NM, LA-UR-80-3702, 1980 to be published.

[10] G. I. KERLEY, *Rational function method of interpolation*, Los Alamos National Laboratory report LA-6903-MS, Los Alamos, NM, 1977.

[11] D. F. MCALLISTER, E. PASSOW AND J. A. ROULIER, *Algorithms for computing shape preserving spline interpolations to data*, Math. Comput., 31 (1977), pp. 717–725.