SIAM J. NUMER. ANAL.
Vol. 17, No. 2, April 1980

0036-1429/80/1702-0005$01.00/0

MONOTONE PIECEWISE CUBIC INTERPOLATION*

F. N. FRITSCH† AND R. E. CARLSON‡

Abstract. Necessary and sufficient conditions are derived for a cubic to be monotone on an interval. These conditions are used to develop an algorithm which constructs a visually pleasing monotone piecewise cubic interpolant to monotone data. Several examples are given which compare this algorithm with other interpolation methods.

1. Introduction. Scientists and engineers usually demand that approximation methods accurately represent physical reality (at least as they perceive it). Typical of their demands is that of producing a monotone function to fit monotone data. Using standard techniques it is often necessary to sacrifice interpolation of the data in order to preserve monotonicity, or conversely, to sacrifice monotonicity in order to preserve interpolation. We assume here that the data are sufficiently accurate to warrant interpolation, rather than a least squares or other approximation method.

In this paper we derive necessary and sufficient conditions for a cubic to be monotone in an interval. These conditions are then used to develop an algorithm which constructs a $\mathscr{C}^1$ monotone piecewise cubic interpolant to monotone data. The curve produced contains no extraneous "bumps" or "wiggles", which makes it more readily acceptable to scientists and engineers. Examples are included which compare this algorithm with other piecewise cubic interpolation methods.

2. Preliminary results. Let $\pi: a = x_1 < x_2 < \cdots < x_n = b$ be a partition of the interval $I = [a, b]$. Let $\{f_i : i = 1, 2, \cdots, n\}$ be a given set of monotone data values at the partition points (knots); that is, we assume either $f_i \leqq f_{i+1} (i = 1, 2, \cdots, n-1)$ or $f_i \geqq f_{i+1}$ $(i = 1, 2, \cdots, n-1)$. Our goal is to construct on π a piecewise cubic function $p(x) \in \mathscr{C}^1[I]$ such that

$$p(x_i) = f_i, \qquad i = 1, 2, \cdots, n \tag{1}$$

and $p(x)$ is monotone.

In each subinterval $I_i = [x_i, x_{i+1}]$, $p(x)$ is a cubic polynomial which may be represented as follows:

$$p(x) = f_i H_1(x) + f_{i+1} H_2(x) + d_i H_3(x) + d_{i+1} H_4(x), \tag{2}$$

where $d_j = p'(x_j)$, $j = i, i+1$, and the $H_k(x)$ are the usual cubic Hermite basis functions for the interval I_i: $H_1(x) = \phi((x_{i+1} - x)/h_i)$, $H_2(x) = \phi((x - x_i)/h_i)$, $H_3(x) = -h_i \psi((x_{i+1} - x)/h_i)$, $H_4(x) = h_i \psi((x - x_i)/h_i)$, where $h_i = x_{i+1} - x_i$, $\phi(t) = 3t^2 - 2t^3$, $\psi(t) = t^3 - t^2$.

Therefore, an algorithm for constructing a piecewise cubic interpolant to $\{(x_i, f_i) : i = 1, 2, \cdots, n\}$ is essentially a procedure for calculating the derivative values $d_1, d_2, \cdots, d_n$. Standard algorithms such as the three point difference formula, the "geometric mean" used by Akima [1], the least squares procedure of Ellis and McLain [6], or requiring $p(x)$ to be a cubic spline do not guarantee monotonicity. Setting $d_i = 0$, $i = 1, 2, \cdots, n$ does produce a monotone interpolant (Passow [10]), but, as we shall see, this choice generally does not produce satisfactory results.

* Received by the editors March 27, 1979. This work was supported by the U.S. Department of Energy under Contract W-7405-ENG-48 and its Office of Basic Energy Sciences, Mathematical Sciences Branch.

† Lawrence Livermore Laboratory, Livermore, California 94550.

‡ Department of Mathematics, Grove City College, Grove City, Pennsylvania 16127.

This paper is an addition to the recent literature on shape preserving interpolation, which is reviewed in [8]. The basic idea here is to produce interpolants that preserve properties such as monotonicity or convexity that are present in the data. Compared to most other shape preserving methods, the method proposed in this paper is characterized by its efficiency, in terms of time required to determine the interpolant, storage required to represent it, and/or time required to evaluate it. We do not consider the various exponential splines that have been proposed [4], [5], [12], because they are too expensive to evaluate.

The taut spline of de Boor [2, pp. 303–314] provides a cubic spline interpolant that preserves the convexity of the data by inserting at most one additional breakpoint between each pair of data points. It does not guarantee monotonicity. It also involves a parameter γ that the user must choose in some way to control the "roughness" of the interpolant, and the proper choice of γ appears to be data dependent.

Pruess [11] describes another approach to shape preserving spline interpolation which (possibly) adds two knots per data interval. One of his algorithms preserves monotonicity, but requires a nonlinear iteration to determine the locations of the additional breakpoints.

Because of the additional breakpoints, both of these methods potentially require more storage and increased search time during evaluation than the method described here. Some computational results indicate that both of these methods tend to produce "flat spots" (that is, sections that are nearly piecewise linear) in the interpolant. Further, both of these methods are global, while the algorithm proposed here is local, in the sense that a single change in the data will affect the interpolant only in neighboring intervals. Thus, if the user does not require the second derivative continuity of these methods, the algorithm described here would seem to be a more efficient alternative.

Perhaps the closest competitor among recently proposed methods is the shape preserving quadratic spline of McAllister and Roulier [9]. By adding at most one breakpoint per data interval, they are able to produce a local, $\mathscr{C}^1$ interpolant which preserves both convexity and monotonicity of the data. The only drawback would appear to be the increased storage requirements due to the additional breakpoints.

3. Monotonicity in a single interval. In this section we examine $p(x)$ on the subinterval I_i in detail. Necessary and sufficient conditions are derived such that $p(x)$ is monotone on I_i. These conditions form the basis for developing a family of algorithms for monotone piecewise cubic interpolation in § 4.

Let $\Delta_i = (f_{i+1} - f_i)/h_i$ be the slope of the line segment joining the data to be interpolated. It is clear that a necessary condition for monotonicity is that[1]

$$\operatorname{sgn}(d_i) = \operatorname{sgn}(d_{i+1}) = \operatorname{sgn}(\Delta_i). \tag{3}$$

Further, if $\Delta_i = 0$, then $p(x)$ is monotone (i.e. constant) on I_i if and only if $d_i = d_{i+1} = 0$.

For the remainder of this section let us assume $\Delta_i \neq 0$ and that (3) is satisfied. Expanding $p(x)$ about $x = x_i$ we obtain

$$p(x) = \left[\frac{d_i + d_{i+1} - 2\Delta_i}{h_i^2}\right](x - x_i)^3 + \left[\frac{-2d_i - d_{i+1} + 3\Delta_i}{h_i}\right](x - x_i)^2 + d_i(x - x_i) + f_i. \tag{4}$$

Then

$$p'(x) = \left[\frac{3(d_i + d_{i+1} - 2\Delta_i)}{h_i^2}\right](x - x_i)^2 + \left[\frac{2(-2d_i - d_{i+1} + 3\Delta_i)}{h_i}\right](x - x_i) + d_i \tag{5}$$

[1] Here sgn (0) matches any sign, by convention.

and

$$p''(x) = \left[\frac{6(d_i + d_{i+1} - 2\Delta_i)}{h_i^2}\right](x - x_i) + \left[\frac{2(-2d_i - d_{i+1} + 3\Delta_i)}{h_i}\right]. \tag{6}$$

The following statements are directly obtainable from (4)–(6) for the two cases cited below.

Case I. $d_i + d_{i+1} - 2\Delta_i = 0$. In this case $p(x)$ is quadratic (or linear) and $p'(x)$ is linear (or constant). Since $\min(d_i, d_{i+1}) \leqq p'(x) \leqq \max(d_i, d_{i+1})$, it follows that (3) is also a sufficient condition for monotonicity.

Case II. $d_i + d_{i+1} - 2\Delta_i \neq 0$. In this case $p'(x)$ is quadratic. It is concave up if $d_i + d_{i+1} - 2\Delta_i > 0$ and concave down if $d_i + d_{i+1} - 2\Delta_i < 0$. Note that if $f_i < f_{i+1}$ and $p'(x)$ is concave down, then $p(x)$ is monotone increasing since $0 \leqq \min(d_i, d_{i+1}) \leqq p'(x)$. Similarly, if $f_i > f_{i+1}$ and $d_i + d_{i+1} - 2\Delta_i > 0$, then $p(x)$ is monotone decreasing since $p'(x) \leqq \max(d_i, d_{i+1}) \leqq 0$.

To accommodate both monotone increasing and decreasing in a single condition, let $\alpha_i = d_i/\Delta_i$ and $\beta_i = d_{i+1}/\Delta_i$ be the respective ratios of the endpoint derivatives to the slope of the secant line. It follows from the above discussion that $d_i + d_{i+1} - 2\Delta_i = (\alpha_i + \beta_i - 2)\Delta_i$ and $p(x)$ is monotone if $\alpha_i + \beta_i - 2 < 0$.

The results of Case I and Case II are summarized in

LEMMA 1. *If* $\alpha_i + \beta_i - 2 \leqq 0$, *then* $p(x)$ *is monotone on* I_i *if and only if* (3) *is satisfied.*

In the remainder of this section we restrict our attention to the case $\alpha_i + \beta_i - 2 > 0$. Note that whenever (3) is satisfied, α_i and β_i are nonnegative and that nonmonotonic behavior may result when α_i and/or β_i are "too large". Values of α_i and β_i which produce a monotone interpolant are given in Lemma 2. First, however, we observe that $p'(x)$ has a unique extremum at

$$x^* = x_i + \frac{h_i}{3}\left[\frac{2\alpha_i + \beta_i - 3}{\alpha_i + \beta_i - 2}\right] \tag{7}$$

and

$$p'(x^*) = \phi(\alpha_i, \beta_i)\Delta_i, \tag{8}$$

where

$$\phi(\alpha, \beta) = \alpha - \frac{1}{3}\frac{(2\alpha + \beta - 3)^2}{(\alpha + \beta - 2)}. \tag{9}$$

It is clear from (7)–(9) that $p(x)$ is monotone on I_i if and only if one of the following conditions is satisfied:

(i) $x^* \notin (x_i, x_{i+1})$;

(ii) $x^* \in (x_i, x_{i+1})$ and $\operatorname{sgn}(p'(x^*)) = \operatorname{sgn}(\Delta_i)$.

Condition (i) can be written as $2\alpha_i + \beta_i - 3 \leqq 0$ for $x^* \leqq x_i$ and $\alpha_i + 2\beta_i - 3 \leqq 0$ for $x^* \geqq x_{i+1}$. Condition (ii) is equivalent to $\phi(\alpha_i, \beta_i) \geqq 0$. These results are summarized with

LEMMA 2. *If* $\alpha_i + \beta_i - 2 > 0$, *and* (3) *is satisfied, then* $p(x)$ *is monotone on* I_i *if and only if one of the following conditions is satisfied*:

(i) $2\alpha_i + \beta_i - 3 \leqq 0$;

(ii) $\alpha_i + 2\beta_i - 3 \leqq 0$; *or*

(iii) $\phi(\alpha_i, \beta_i) \geqq 0$.

As a consequence of Lemmas 1 and 2 it is possible to construct a region $\mathcal{M}$ of acceptable values for α_i and β_i (hence d_i and d_{i+1}) to produce a monotone interpolant

on I_i. This region is shown in Fig. 1.[2] We note that the curve $\phi(\alpha, \beta) = 0$ is the ellipse $(\alpha-1)^2+(\alpha-1)(\beta-1)+(\beta-1)^2-3(\alpha+\beta-2)=0$, which is tangent to the coordinate axes at (3, 0) and (0, 3).

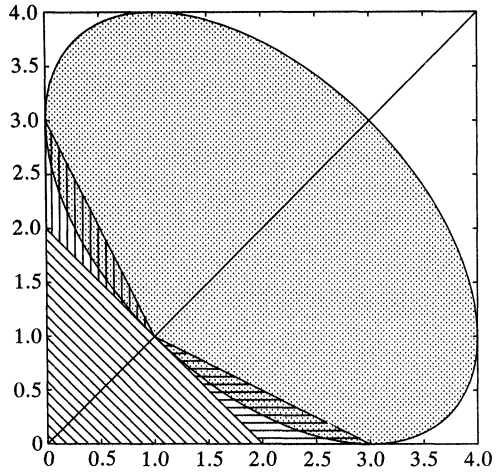

FIG. 1. *The monotonicity region $\mathcal{M}$. (α is the horizontal axis; β, vertical.) Diagonal hatching: $\alpha+\beta-2 \leqq 0$. Vertical hatching: $\alpha+\beta-2>0$ and $2\alpha+\beta-3 \leqq 0$. Horizontal hatching: $\alpha+\beta-2>0$ and $\alpha+2\beta-3 \leqq 0$. Dotted: $\phi(\alpha, \beta) \geqq 0$. Unshaded: Cubic is nonmonotone.*

4. Monotone piecewise cubic interpolation algorithm. The results of § 3 suggest the following two-step procedure for constructing monotone piecewise cubic interpolation algorithms.

Step 1. Initialize the derivatives d_i, $i = 1, 2, \cdots, n$ such that $\operatorname{sgn}(d_i) = \operatorname{sgn}(d_{i+1}) = \operatorname{sgn}(\Delta_i)$. If $\Delta_i = 0$, set $d_i = d_{i+1} = 0$.

Step 2. For each interval I_i in which $(\alpha_i, \beta_i) \notin \mathcal{M}$, modify d_i and d_{i+1} to d_i^* and d_{i+1}^* such that $(\alpha_i^*, \beta_i^*) \in \mathcal{M}$, where $\alpha_i^* = d_i^*/\Delta_i$ and $\beta_i^* = d_{i+1}^*/\Delta_i$.

In order to implement Step 2 it is necessary to note the interactions between adjacent intervals, i.e., $\beta_{i-1}\Delta_{i-1} = d_i = \alpha_i\Delta_i$. In modifying d_i to produce monotonicity on I_i we are also changing β_{i-1}. Care must be taken to preserve monotonicity on I_{i-1}. One way to accomplish this is to select a subset $\mathcal{S} \subset \mathcal{M}$ such that

(a) If $(\alpha, \beta) \in \mathcal{S}$, then $(\alpha^*, \beta^*) \in \mathcal{S}$ whenever $0 \leqq \alpha^* \leqq \alpha$ and $0 \leqq \beta^* \leqq \beta$.

(b) If $(\alpha, \beta) \in \mathcal{S}$, then $(\beta, \alpha) \in \mathcal{S}$.

While the symmetry property (b) is not essential, it is present in $\mathcal{M}$ and seems to be intuitively desirable. Therefore, Step 2 may be replaced with

Step 2A. For each I_i in which $(\alpha_i, \beta_i) \notin \mathcal{S}$, modify d_i and d_{i+1} to d_i^* and d_{i+1}^* such that $0 \leqq \alpha_i^* \leqq \alpha_i$, $0 \leqq \beta_i^* \leqq \beta_i$, and $(\alpha_i^*, \beta_i^*) \in \mathcal{S}$.

Thus we see that an algorithm for monotone piecewise cubic interpolation has three basic components:

(i) an initialization procedure for Step 1;

(ii) the choice of a subregion $\mathcal{S}$ of $\mathcal{M}$, satisfying properties (a) and (b);

(iii) the selection of an algorithm for mapping (α_i, β_i) to (α_i^*, β_i^*) for Step 2A.

[2] It is interesting to note that an essentially identical diagram (discovered by the authors only after the original version of this paper was written) appeared in Appendix 6 of Forrest [7], in the context of avoiding kinks in a rational cubic straight line.

To implement Step 1 we have found the standard three-point difference formula to be satisfactory for $d_2, d_3, \cdots, d_{n-1}$. For the end derivatives, the noncentered three point difference formula may be used, although it is sometimes necessary to modify d_1 and/or d_n if the signs are not appropriate. In these cases we have obtained better results setting d_1 or d_n equal to zero, rather than equal to the slope of the secant line.

Several choices for the set $\mathcal{S}$ which have been considered are the sets $\mathcal{S}_k$ described below and depicted in Fig. 2.

$\mathcal{S}_1$—The largest subset of $\mathcal{M}$ satisfying properties (a) and (b). It is bounded by the four lines[3] $\alpha = 0, 3$ and $\beta = 0, 3$.

$\mathcal{S}_2$—Circle centered at the origin of radius 3.

$\mathcal{S}_3$—The subset of $\mathcal{M}$ bounded by $\alpha + \beta - 3 = 0$.

$\mathcal{S}_4$—The subset of $\mathcal{M}$ bounded by $2\alpha + \beta - 3 = 0$ or $\alpha + 2\beta - 3 = 0$ (Lemma 2).

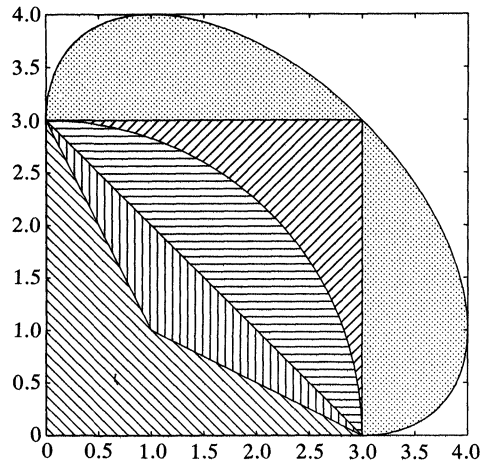

FIG. 2. *Subregions of $\mathcal{M}$. Diagonal hatching (−slope): $\mathcal{S}_4$; Vertical hatching: $\mathcal{S}_3 - \mathcal{S}_4$; Horizontal hatching: $\mathcal{S}_2 - \mathcal{S}_3$; Diagonal hatching (+slope): $\mathcal{S}_1 - \mathcal{S}_2$; Dotted: $\mathcal{M} - \mathcal{S}_1$.*

Sample data sets have been run using each $\mathcal{S}_k$ defined above. The choice of $\mathcal{S}_1$ produces the least change in the derivatives and the graph more closely resembles the graph obtained using the standard three point difference formula. (Compare Fig. 3a and Fig. 4b.) The choice of $\mathcal{S}_4$ produces the greatest change in the derivatives and the graph more closely resembles a piecewise linear function. The choices $\mathcal{S}_2$ and $\mathcal{S}_3$ lie somewhere in between. A poll of potential users has led to the choice of $\mathcal{S}_2$ as producing the most "pleasing" results. The two extreme cases are illustrated in Fig. 3; the corresponding result for $\mathcal{S}_2$ is given in Fig. 4d.

One procedure for modifying the derivative values in Step 2A is to construct the line joining the origin to the point (α_i, β_i). Let (α_i^*, β_i^*) be the point of intersection of this line with the boundary of $\mathcal{S}$. Then $d_i^* = \alpha_i^* \Delta_i$ and $d_{i+1}^* = \beta_i^* \Delta_i$. For $\mathcal{S} = \mathcal{S}_2$, $\alpha_i^* = \tau_i \alpha_i$, $\beta_i^* = \tau_i \beta_i$, where $\tau_i = 3(\alpha_i^2 + \beta_i^2)^{-1/2}$.

5. Numerical examples. In this section we compare the results of the method described in the previous section with several other piecewise cubic interpolation methods on two data sets.

[3] de Boor and Swartz [3] were apparently aware that $(\alpha_i, \beta_i) \in \mathcal{S}_1$ is sufficient for monotonicity, but they give no derivation of this fact.

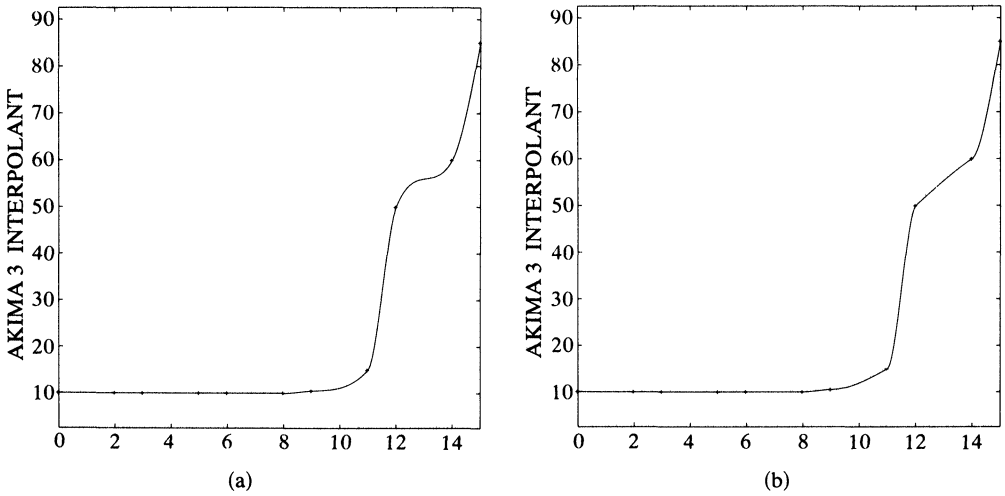

FIG. 3. *Effect of choice of $\mathcal{S}$ on shape of curve.* (a) $\mathcal{S} = \mathcal{S}_1$, (b) $\mathcal{S} = \mathcal{S}_4$.

The first data set, used in Figs. 3 and 4, is the third example from Akima [1], namely

x	0	2	3	5	6	8	9	11	12	14	15
y	10	10	10	10	10	10	10.5	15	50	60	85

The six methods compared in Fig. 4 are as follows.

SPLINE. This is cubic spline interpolation, in which $d_2, \cdots, d_{n-1}$ are chosen so that $f \in \mathcal{C}^2[a, b]$, with the two remaining degrees of freedom used to determine the end derivative values. For the curve in Fig. 4a we used noncentered three-point difference formulas, but the results are relatively insensitive to the choice of boundary conditions.

BESSEL. This is what de Boor [2, p. 53] calls cubic Bessel interpolation, in which the interior derivatives are set using the standard three point difference formula. The end derivatives are set as for SPLINE. Note that these are the initial derivatives used in Step 1 of the new algorithm described above. In Fig. 4b we see that the "wiggles" have disappeared from the flat portion of the curve, but there is still an unacceptable "bump" in interval (9, 11). We note that the Ellis–McLain algorithm [6] has produced qualitatively the same results as the simpler cubic Bessel interpolant on all examples we have tried.

AKIMA. This is the method proposed by Akima [1], in which the d_i are set to the following weighted average of Δ_{i-1} and Δ_i:

$$d_i = \frac{a_i}{a_i + b_i}\Delta_{i-1} + \frac{b_i}{a_i + b_i}\Delta_i, \qquad i = 3, \cdots, n-2,$$

where $a_i = |\Delta_{i+1} - \Delta_i|$, $b_i = |\Delta_{i-1} - \Delta_{i-2}|$. (See [1] for endpoint treament.) In Fig. 4c we see that Akima's method eliminates the "bump", but the interpolant is not monotone on interval (12, 14).

F-C. This is the method described in the previous section, with the three point formula used in Step 1 and $\mathcal{S} = \mathcal{S}_2$ in Step 2A. We see in Fig. 4d that the interpolant is now strictly monotonic where the data are.

ZERO D. To illustrate that monotonicity is not sufficient to produce an acceptable interpolant, we show in Fig. 4e the curve that results when we set $d_i = 0$, $i = 1, \cdots, n$.

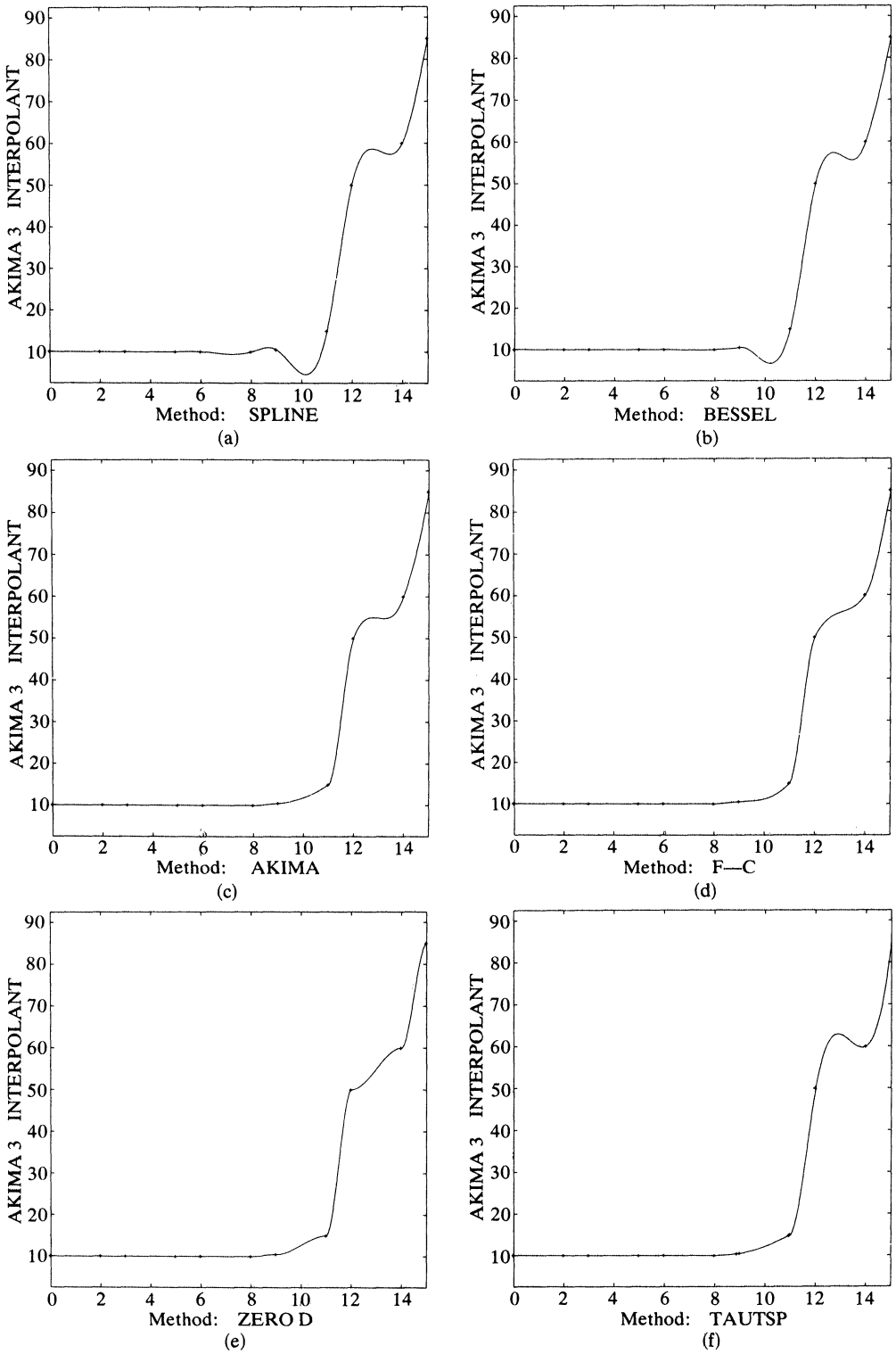

FIG. 4. *Results on Data Set AKIMA 3 for six methods*: (a) *Cubic spline interpolation*, (b) *Cubic Bessel interpolation*, (c) *Akima's method*, (d) *The method described here*, (e) *All zero derivatives*, (f) *de Boor's taut spline* ($\gamma = 0.5$).

TAUTSP. Although it does not fit directly into the scheme of the previous section, because breakpoints other than at data points are allowed, we present in Fig. 4f the result of using de Boor's [2, pp. 303–314] taut spline on these data, with parameter $\gamma = 0.5$. As with AKIMA and F-C, TAUTSP eliminates the "bump" and produces a fairly sharp bend near the data point at $x = 11$. Here, this is accomplished by adding breakpoints near 9 and 11. (Using the suggested value $\gamma = 2.5$ moves the added breakpoints, hence the sharp bend, to the left.) Note that TAUTSP does nothing about the "wiggle" in interval (12, 14), because it is trying to preserve convexity, rather than monotonicity.

The second data set, used in Fig. 5, is representative of the type of data that motivated this work. These are actual data from *LLL* radiochemical calculations.

x	7.99	8.09	8.19	8.7	9.2	10.	12.	15.	20.
y	0	2.76429$E-5$	4.37498$E-2$	0.169183	0.469428	0.943740	0.998636	0.999919	0.999994

We present results for four of the six methods described above. Comparing Figs. 5a and 5b we see that again Akima's method eliminates the wiggles from the flat part of the

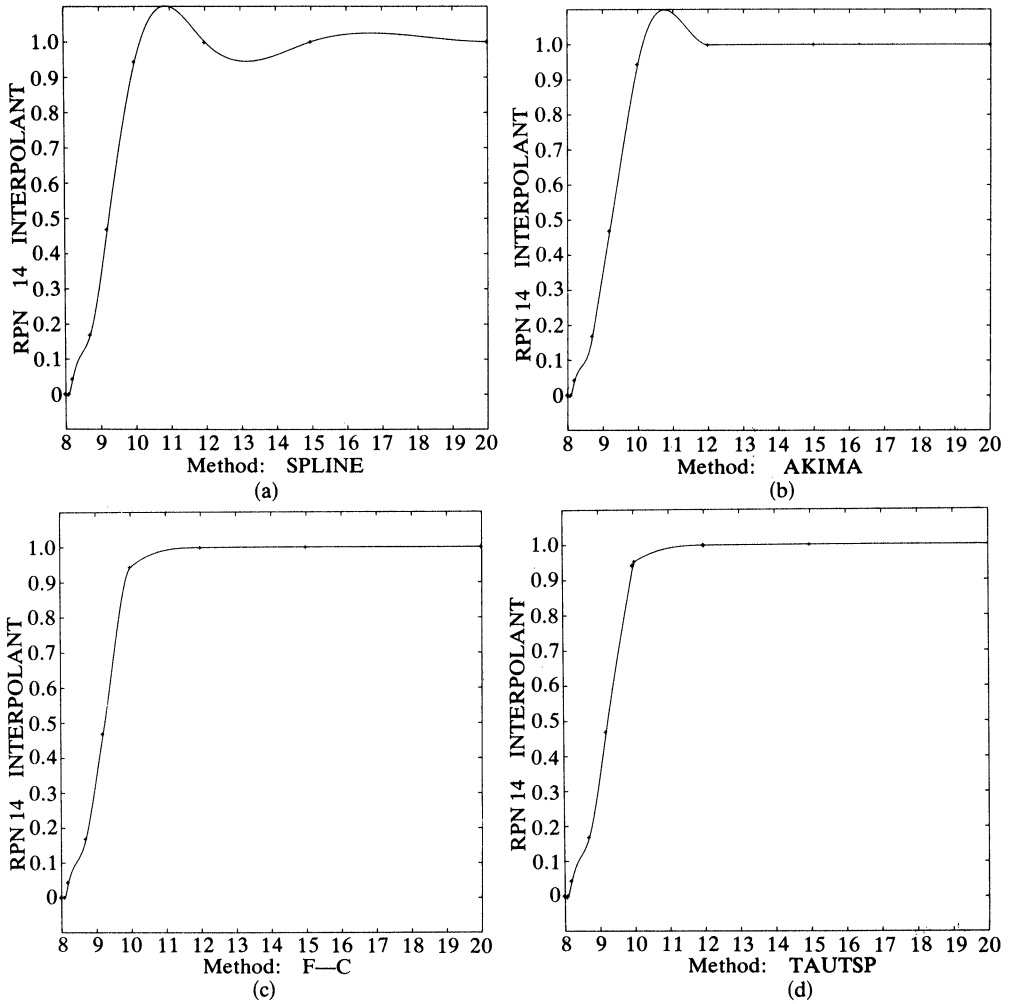

FIG. 5. *Results on Data Set RPN 14 for four methods*: (a) *Cubic spline interpolation*, (b) *Akima's method*, (c) *The method described here*, (d) *de Boor's taut spline* ($\gamma = 0.5$).

curve, but has an unacceptable "bump" in the interval (10, 12). The F-C algorithm (Fig. 5c) eliminates this "bump" and produces quite an acceptable interpolant. In Fig. 5d we see that TAUTSP with $\gamma = 0.5$ produces an almost identical curve by introducing three additional breakpoints, two near 10 and one near 12. (The suggested value $\gamma = 2.5$ produces an interpolant that is too nearly piecewise linear in (10, 12).)

6. Discussion. We have demonstrated the ability to produce "visually pleasing" monotone piecewise cubic interpolants. The algorithm is simple and the interpolant is affected only locally by changes in the data. The major open question in this area is whether it is possible to provide a sufficiently precise definition of "visually pleasing" so that a one-pass algorithm can be developed to compute the "best" piecewise cubic interpolant to a given set of data. Work is under way on extending these ideas to piecewise monotone interpolation and to the interpolation of two-dimensional data.

Acknowledgments. The authors wish to express their appreciation to Don Gardner, whose data motivated the development of this algorithm, and Paul Dubois, who provided many valuable discussions in the early stages of this research.

REFERENCES

[1] H. AKIMA, *A new method of interpolation and smooth curve fitting based on local procedures*, J. Assoc. Comput. Mach., 17 (1970), pp. 589–602.
[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. Approximation Theory, 21 (1977), pp. 411–416.
[4] A. K. CLINE, *Scalar- and planar-valued curve fitting using splines under tension*, Comm. ACM, 17 (1974), pp. 218–223.
[5] R. P. DUBE, *Univariate blending functions and alternatives*, Computer Graphics and Image Processing, 6 (1977), pp. 394–408.
[6] T. M. R. ELLIS AND D. H. MCLAIN, *Algorithm 514. A new method of cubic curve fitting using local data*, ACM Trans. Math. Software, 3 (1977), pp. 175–178.
[7] A. R. FORREST, *Curves and surfaces for computer-aided design*, Ph.D. Thesis, Univ. of Cambridge, Cambridge, England, July 1968.
[8] D. F. MCALLISTER, E. PASSOW AND J. A. ROULIER, *Algorithms for computing shape preserving spline interpolations to data*, Math. Comp., 31 (1977), pp. 717–725.
[9] D. F. MCALLISTER AND J. A. ROULIER, *An algorithm for computing a shape preserving osculatory quadratic spline*, ACM Trans. Math. Software, submitted.
[10] E. PASSOW, *Piecewise monotone spline interpolation*, J. Approximation Theory, 12 (1974), pp. 240–241.
[11] S. PRUESS, *Alternatives to the exponential spline in tension*, Math. Comp., to appear.
[12] H. SPÄTH, *Spline Algorithms for Curves and Surfaces*, Utilitas Mathematica, Winnipeg, Canada, 1974.