See page 178-forward

Software for C^1 Surface Interpolation

C. L. Lawson

1. INTRODUCTION.

This paper is a result of our fourth effort in software for surface representation. We developed subroutines for rectangular grid contour plotting in 1965 with N. Block and R. Garrett, least squares bicubic spline surface fitting in 1970 with R. Hanson, and contour plotting via triangular grid construction and linear interpolation in 1972.

The latter two subroutines deal with irregularly located data. However, applications continue to arise in which one would like the interpolatory capability of the triangular grid program but with at least C^1 continuity. Such an algorithm with underlying theory and implementing software are the topics of this paper.

In Secs. 2, 3, and 4 we introduce the problem and give a brief survey of the pertinent literature. Sections 5 through 10 describe our algorithm and conclude with examples of surfaces produced by our new subroutines. We express appreciation to Bob Barnhill and Frank Little for valuable discussions that particularly influenced our triangulation algorithm of Sec. 6.

There has been practically no theory to guide the development of algorithms for triangulation and no practical static global criterion to characterize a preferred triangulation. We are indebted to Michael Powell and Robin Sibson for conversations and correspondence in 1976 that introduced us to

Thiessen proximity regions and the fact that this concept can be used to define a triangulation as is related in Sec. 12.2.

In our initial effort to determine the relationship of the Thiessen criterion to the max-min angle criterion we had used in 1972, we discovered the circle criterion, which served as a convenient mathematical link between the other two. The outcome is the material of Sec. 11, showing the equivalence of these three criteria when used for local optimization of a triangular grid.

The local equivalence results opened the way to certain global equivalences reported in Sec. 12 and new algorithmic insights reported in Secs. 13 and 14.

Our conclusions regarding the state of the art for this problem appear in Sec. 15.

2. PROBLEM STATEMENT.

The following surface interpolation problem will be treated: Given a set of triples of data (x_i, y_i, z_i), $i=1, \ldots, n$, construct a conveniently computable C^1 function $f(x,y)$ satisfying the interpolation conditions

$$z_i = f(x_i, y_i), \quad i=1,\ldots,n$$

The data (x_i, y_i) are not assumed to lie in any special pattern such as at the nodes of a rectangular grid. It is assumed, however, that all (x_i, y_i) pairs are distinct; i.e., $(x_i, y_i) = (x_j, y_j)$ only if $i=j$.

3. EXPECTED APPLICATIONS.

The usual situation in which the author has seen a need for this type of computation is that in which a scientist or engineer has in hand a set of (x_i, y_i, z_i) data representing measured or computed values of some phenomenon and desires to obtain a visual impression of a smooth surface of the form $z = f(x, y)$ interpolating the data. In such a case, an interpolation algorithm, such as is treated in this paper, must be interfaced with algorithms for contour plotting or surface perspective plotting. If, as is the case at JPL, subroutines are available for doing contour or surface perspective plotting for data given on a rectangular grid, then the surface interpolation algorithm can be used to produce the values needed at the lattice points of a rectangular grid.

Other applications have arisen which can be regarded as the inverse of contour plotting. Certain handbook data is available in the form of contour plots. To use the data in a computer program it is necessary to produce a computable representation of the function depicted by the contour plots. A convenient way to do this is to develop a list of (x_i, y_i, z_i) values from appropriately spaced points along the contour lines and then use a surface interpolation algorithm such as is discussed in this paper.

We have also seen applications which can be regarded as implicit function problems. One may have a rectangular table or a contour plot giving z as a function of x and y, but then need to be able to determine x as a function of y and z in some computational procedure. If the data has appropriate monotonicity for this to make sense, then the interpolation algorithm of this paper can be used for such problems.

4. PUBLISHED WORK ON SURFACE INTERPOLATION TO IRREGULARLY LOCATED DATA

A variety of algorithmic ideas have been developed for this problem or closely related problems.

Two of the most recent papers giving methods for C^1 surface interpolation to irregularly located data are Akima (1975) and McLain (1976). Akima's report contains listings of a set of Fortran subroutines to handle this problem. This code and a second version of it using a more economical triangulation subroutine due to Lawson (1972) have been made available to requestors by Akima.

Both Akima (1975) and McLain (1976) contain introductory sections giving useful brief characterizations of other approaches, particularly those of Bengtsson and Nordbeck (1964), Shepard (1968), Maude (1973), and McLain (1974). Franke (1975) reports on computer tests of eleven methods constructed using a combination of ideas from Sard (1963), Mansfield (1972), Maude (1973), McLain (1974), Nielson (1974), and Barnhill and Nielson (1974).

Powell (1976) and Schumaker (1976) give surveys of methods for surface fitting and related problems of bivariate function representation.

The computerized representation of surfaces is a central issue in the field of computer-aided geometric design (CAGD) and plays an important role in the field of finite element methods (FEM). For discussions of surface representation from the point of view of CAGD see Forrest (1972) and Barnhill (1977). For descriptions of surface elements used in FEM see Birkhoff and Mansfield (1974) as well as any of the current books on FEM.

Some methods of building a triangular grid with a given set of nodes start by locating the boundary points of the convex hull of the point set. Algorithms for locating these boundary points are treated in Graham (1972), Jarvis (1973), Preparata and Hong (1977), and Eddy (1976).

Some interesting new triangular grid elements providing C^1 continuity through the use of piecewise quadratics are described in Powell and Sabin (1976).

5. OUTLINE OF THE ALGORITHMIC APPROACH SELECTED.

Our approach to the C^1 surface interpolation problem consists of the following four steps.

1. Construct a triangular grid covering the convex hull of the given set of (x_i, y_i) data using the (x_i, y_i) data points as vertices of the triangular cells.
2. Estimate first partial derivatives of z with respect to x and y at each of the (x_i, y_i) data points.
3. For an arbitrary (x, y) point, perform a lookup in the triangular grid to identify which triangle, if any, contains the point.
4. For an arbitrary (x, y) point in a triangle, compute an interpolated value of z and optionally of $\partial z/\partial x$ and $\partial z/\partial y$ also. Make use of the nine items of data associated with the triangle, i.e., the values of z_i and its two first partial derivatives at each of the three vertices.

This same top level description of the approach also characterizes the methods of Akima (1975) and McLain (1976), with the exception that their methods estimate different quantities at Step 2 for use in the interpolation at Step 4.

At the more detailed level of devising algorithms for each of the four steps, there are substantial differences between our approach and that of Akima and that of McLain.

The four steps will be discussed in the following four sections.

6. CONSTRUCTING A TRIANGULAR GRID.

Given the set S of distinct points, (x_i, y_i), $i=1,\cdots, n$, the triangular grid T to be constructed is to cover the convex hull of this set of points. Each triangular cell in the grid is to have three of the given points as its vertices and is to contain no other points of S as interior or boundary points.

Conceptually there is no difficulty in manually constructing such a triangular grid. For example, one can just start drawing edges connecting pairs of points and continue as long as edges can be drawn that do not intersect any previously drawn edges. An edge must not contain any points of S other than its own two endpoints.

In general, there exist many different triangulations of a set S. It is noteworthy, however, that all possible triangulations of S have the same number of triangles and the same number of edges. Let n_b denote the number of points of S on the boundary of the convex hull of S and let n_i denote the number in the interior so that $n = n_b + n_i$. Then the number of triangles is

$$n_t = n_b + 2(n_i - 1) \le 2n$$

and the number of edges is

$$n_e = 2n_b + 3(n_i - 1) \le 3n$$

Taking these relationships into account we selected the data structure illustrated by Figs. 1 and 2 to represent a triangular grid. Column 1 of Fig. 1 is not stored so each triangle is represented in storage by six integers. Since $n_t \le 2n$, a total of 12n integer storage locations suffices to represent the triangular grid for n points. This of course is in addition to storage for the (x_i, y_i) data.

Three subroutines for storing and fetching in this structure are used throughout the package so that the actual mode of storing these pointers can be "hidden" from the main

TRIANGLE INDEX	INDICES OF ADJACENT TRIANGLES IN COUNTERCLOCKWISE ORDER. A ZERO INDICATES THE REGION EXTERIOR TO THE TRIANGULAR GRID			INDICES OF VERTEX POINTS IN COUNTERCLOCKWISE ORDER. THE FIRST VERTEX IS AT THE POINT OF CONTACT OF THE THIRD AND FIRST NEIGHBORING TRIANGLE		
1	2	0	4	5	8	7
2	5	3	1	5	3	8
3	6	0	2	3	1	8
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.

Fig. 1. Data Structure Representing a Triangular Grid.

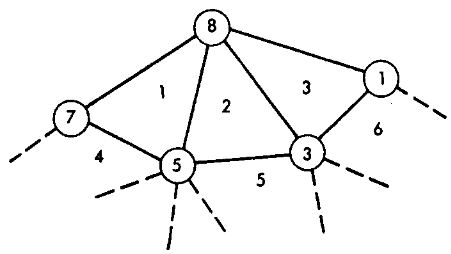

Fig. 2. The Portion of a Triangular Grid Described by the Data Structure of Fig. 1.

subroutines. On many computers one could easily pack two or three of these pointers per computer word. Our triangulation algorithm has the property that it makes additions and changes to the list of triangles but no deletions. Thus there is no garbage collection problem.

In some reasonable triangulation algorithms the number of boundary points of the convex hulls of a sequence of subsets of S has an effect on the operation count. Thus it is desirable to have some notion of the expected value of n_b as a function of n. Clearly, any value of n_b from 3 to n is possible.

Consider the case in which S is a random sample of n points from a uniform distribution on a disc. Let v_n denote the expected value of n_b in this case. Renyi and Sulanki (1963, 1964) and also Raynaud (1970) show that v_n is asymptotically proportional to $n^{1/3}$ as $n \to \infty$. Efron (1965) derives the following formula for v_n:

$$v_n = \frac{n(n-1)\pi^2}{3} \int_{-1}^{1} F^{n-2} f^3 \, dp$$

where

$$f(p) = \frac{2}{\pi}(1-p^2)^{1/2}$$

and

$$F(p) = \int_{-1}^{p} f(t)\,dt = \frac{1}{2} + \frac{1}{\pi}\left[p(1-p^2)^{1/2} + \arcsin(p)\right]$$

We have evaluated this formula by numerical integration and corroborated the results by a computer program that counted the number of boundary points of the convex hulls of a large number of point sets generated as pseudorandom samples from a uniform distribution on the unit disc. Selected values are shown in Table 1.

Since a given point set S generally admits many different triangulations, how is one to characterize and produce a preferred triangulation? So far there does not appear to be any fundamentally compelling best answer to this question.

A very satisfactory candidate, however, emerges from the theoretical results presented in Secs. 11 - 14. There it is shown that three differently stated criteria for choosing the preferred triangulation of a quadrilateral are in fact equivalent in that they produce the same decisions in all cases. It is also shown that if all quadrilaterals consisting of pairs of adjacent triangles in a triangulation satisfy one of these optimality criteria, then the triangulation as a whole has some pleasant properties, and in fact is unique to within some arbitrariness associated with subsets of four or more neighboring points lying on a common circle.

n	v_n	$c_n = v_n/n^{1/3}$
4	3.7	2.3
10	6.1	2.8
100	15.2	3.3
1000	33.6	3.4
10000	72.8	3.4

Table 1. v_n Is the Expected Number of Boundary Points in the Convex Hull of an n-point Sample from the Uniform Distribution on a Disc.

It is further shown that these local criteria have favorable properties for use in triangulation algorithms. In particular it is shown that the final triangulation is reached in a finite number of steps and that the operation of changing an optimized triangulation to include a new data point has some properties that can be exploited to simplify an algorithm.

Our triangulation subroutine TRIGRD presently uses the max-min angle criterion as its local optimization procedure. This is one of the three equivalent local criteria defined in Sec. 11.

The TRIGRD algorithm starts by finding the point of S having the smallest x coordinate. Any ties are broken by minimizing on y. The point p* found in this way is an extreme point of the convex hull of S. Finding p* requires O(n) operations.

The points of S are next sorted on increasing (squared) Euclidean distance from p*. Denote the points with this ordering by $q_1, q_2, \cdots, q_n$ with $q_1 = p^*$. We estimate the operation count for the sort to be O(n log n).

The first edge is drawn connecting q_1 and q_2. The next point in the q_i sequence not colinear with q_1 and q_2 is connected to q_1 and q_2 to form the first triangle. If this third vertex is not q_3 but rather q_k with $k > 3$, relabel the points q_3 through q_k so that q_k is called q_3 and the indices of the intervening points are each increased by one. These steps assure that q_j is strictly outside the convex hull of $\{q_1, \cdots, q_{j-1}\}$ for all $j = 4, 5, \cdots, n$.

Let c denote the centroid of $\Delta q_1 q_2 q_3$. Let r be the half ray from c through q_1. When an angular coordinate is needed for a point, the angle will be measured counterclockwise around c from r. Note that the angular coordinate of q_1 is zero, and all other points q_i for $i > 1$ have angular coordinates strictly between 0 and 2π. The program does not actually compute angles but rather computes a less expensive function monotonically related to the angle.

Build an initial boundary list consisting of q_1, q_2, q_3, and q_1 (again) along with their angles, assigning the angle 2π to the second occurrence of q_1.

This finishes the preliminaries. The algorithm proceeds to loop through the points q_k, $k=4,\cdots,n$, doing the following for each one:

Determine the angular coordinate of q_k and use that coordinate as a key to search for a pair of boundary points whose angles bracket that angle. This is a linear search requiring an average of $n_b^{(k)}/2$ scalar comparisons, where $n_b^{(k)}$ is the number of points on the boundary of the convex hull of $\{q_1, \cdots, q_{k-1}\}$. If we estimate $n_b^{(k)}$ to be about $3k^{1/3}$ (recall Table 1), then the total cost of this lookup as k runs from 4 to n is $O(k^{4/3})$. This appears to be the highest-order operation count in the triangulation algorithm.

Having found two boundary points to which q_k can be connected, attach q_k to these points and record the edge opposite q_k in the new triangle in a stack, identifying edges to be tested for possible swapping.

If the stack is nonempty, unstack one edge and apply the the local optimization procedure to it. If the decision is to swap the edge, do so, and stack the two edges that are opposite q_k in the two new triangles. Continue processing the stack until it is empty.

When the stack is empty try to connect q_k to another neighboring boundary point. If this is possible, then run through the stacking and testing again, starting with the edge opposite q_k in the new triangle. When q_k cannot be connected to any more boundary points, the processing of q_k is completed.

We estimate the average operation count to process q_k is a constant, independent of k. Thus the total cost to process all the points q_k, k=4, ···, n is O(n).

The total operation count for TRIGRD is thus estimated to be $O(n^{4/3}) + O(n \log n) + O(n)$. Actual timing on the Univac 1108 for cases having four different values of n are shown in Table 2.

The data of Table 2 suggests that in the range $25 \le n \le 500$, either $O(n^{4/3})$ or $O(n \log n)$ may be used as a model of the execution time of this triangulation algorithm.

7. ESTIMATING PARTIAL DERIVATIVES AT THE GRID NODES.

To estimate $\partial z/\partial x$ and $\partial z/\partial y$ at a nodal point $p = (x_k, y_k)$ of the triangular grid, the subroutine ESTPD sets up and solves a local least squares problem. All of the immediate grid neighbors of point p are used up to a maximum of 16 immediate neighbors. If the number of immediate neighbors is less than six, then additional nodes beyond the immediate neighbors are used to bring the total number of points in addition to p up to six.

A six-parameter quadratic polynominal in x and y is fit to the z data values at this set of points. The quadratic is forced to interpolate the z value at p, and it fits the remaining points in a weighted least squares sense. The weighting is used to diminish the effect of the more distant points.

n	t	$t/(n^{4/3})$	$t/(n \log n)$
25	0.061	0.00084	0.00175
100	0.346	0.00075	0.00173
200	0.951	0.00081	0.00207
500	2.211	0.00056	0.00164

Table 2. t Denotes the Time in Seconds for Execution of TRIGRD for a Case Having n points

The values at p of the first partial derivatives of this quadratic are stored as estimates of $\partial z/\partial x$ and $\partial z/\partial y$ at p. Execution time on the Univac 1108 averages 8 milliseconds per point at which partials are estimated.

This method of estimating derivatives is the most ad hoc part of our entire surface interpolation method. We intend to investigate the effect of various parametric and algorithmic changes in this procedure.

8. LOOKUP IN THE TRIANGULAR GRID.

Given an arbitrary point q = (x, y) and an index k in the range $1 \le k \le n_t$, where n_t is the total number of triangles, the subroutine TRFIND tests to see if q is in the triangle whose index is k.

If so, the index k is returned. If not, q must be outside one of the edges of the triangle. In this case, TRFIND resets k to be the index of the neighboring triangle on the other side of that edge and loops back to test q in this new triangle k. If there is no neighboring triangle, the fact that q is outside the triangular grid is reported.

This approach is particularly efficient for the case of interpolating to points of a rectangular grid, since the search can always be started at the triangle in which the previous point was found. When a new row of the rectangular grid is started, the search can be started in the triangle in which the first point of the previous row was found.

9. INTERPOLATION IN A TRIANGLE.

The interpolation subroutine TVAL makes use of the piecewise cubic macroelement of Clough and Tocher (1965). A tutorial derivation of this element and a discussion of some alternative ways to organize its computation are given in Lawson (1976a). Quadrature properties of the element are derived in Lawson (1976b).

Definition of this element involves partitioning the triangle into three subtriangles by drawing internal boundaries from the centroid to each vertex. In each of these three subtriangles the element is a cubic polynomial in x and y

$$z = \sum_{i=0}^{3} \sum_{j=0}^{3-i} a_{ij} x^i y^j$$

The element matches nine items of data, the function value and first partials with respect to x and y at the three vertices. It has C^1 continuity across the internal and external boundaries of the triangle. It is exact for quadratic data.

Since it is a piecewise cubic, it is straightforward to obtain expressions for computing its first partial derivatives. The subroutine TVAL includes an option to compute $\partial z/\partial x$ and $\partial z/\partial y$ as well as z.

Starting with the information x, y, z, $\partial z/\partial x$, and $\partial z/\partial y$ at the vertices of a triangle, this method requires 55 multiplies, 65 adds, and 4 divides to compute one interpolated value. There are various possibilities for saving computed quantities that depend only on the triangle's data in order to cut down the time to interpolate for a number of points in the same triangle.

Execution time on the Univac 1108 averages 750 microseconds per interpolated point. This is about 10 times the cost of EXP or SIN.

10. EXAMPLES.

Figure 3 shows a set S consisting of 26 points in the plane. Figure 4 is the triangular grid constructed for the set S by TRIGRD. In view of the results of Sec. 12, this is a Thiessen triangulation for S.

10.1. QUADRATIC TEST CASE.

Values are assigned at the points of Fig. 1 by computing $z = (-1 + 2x - 3y + 4x^2 - xy + 9y^2)/8$. Using ESTPD to estimate first partial derivatives and TRFIND and TVAL to interpolate to points of a 51 x 51 point rectangular grid, we then obtained the contour plot of Fig. 5 and the perspective plot of Fig. 6. This illustrates the exactness of the method for quadratic data.

This case required 1.9 sec of Univac 1108 CPU time to build the triangular grid and interpolate to the rectangular grid. It then used 15.3 sec in the plotting subroutines.

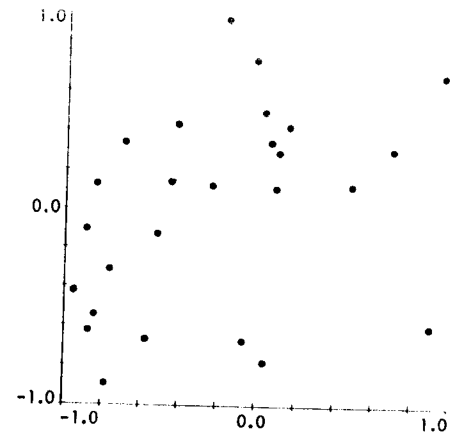

Fig. 3. Set of 26 (x_i, y_i) Points for Examples 1 and 2.

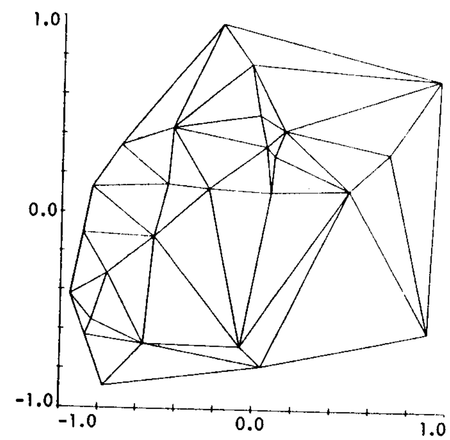

Fig. 4. Thiessen Triangular Grid for Examples 1 and 2.

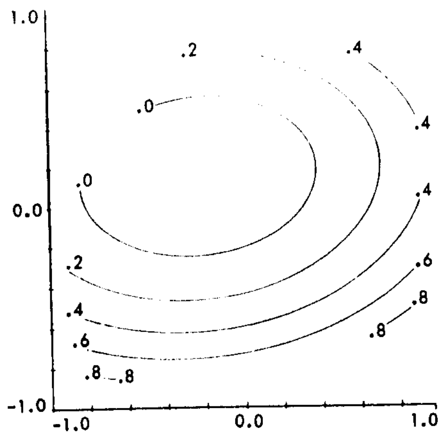

Fig. 5. Contour Plot for Example 1.

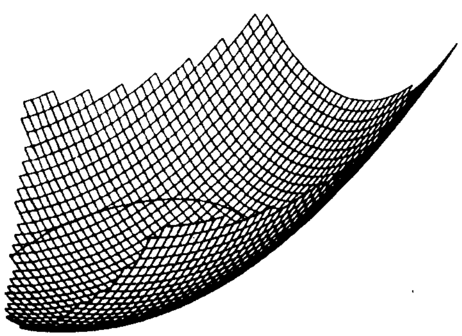

Fig. 6. Perspective Plot for Example 1.

10.2. EXPONENTIAL TEST CASE.

For this case the z data is computed as

$$z = e^{-2(x^2+y^2)}$$

Again estimating partial derivatives at the 26 data points and then interpolating to a 51 x 51 rectangular grid, Figs. 7 and 8 are obtained. The most noticeable defect in the surface produced is the kink in the contour plot near $(x,y) = (0.2,-0.4)$. This also appears as a groove in the perspective plot. From Fig. 3, however, it is seen that this corresponds to a region in which there is a lack of data. Computer time was similar to the first test case.

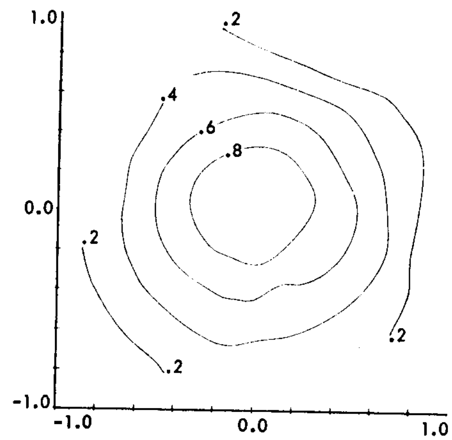

Fig. 7. Contour Plot for Example 2.

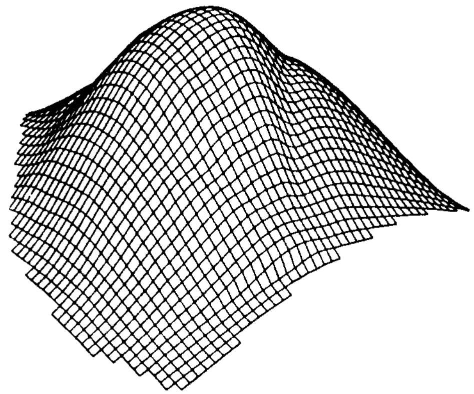

Fig. 8. Perspective Plot for Example 2.

10.3. A CASE WITH MORE POINTS.

The third test case is the same exponential function on a set of 500 points. The grid produced for this case has 985 triangles.

This data was interpolated to a 21 x 21 point rectangular grid for plotting (see Figs. 9, 10, and 11). This case used 6.25 sec of CPU time to triangulate and interpolate. It used 4.01 sec in the plotting subroutines. The plotting was faster because of the much coarser rectangular grid.

11. THREE CRITERIA FOR TRIANGULATION OF A STRICTLY CONVEX QUADRILATERAL.

We will call a quadrilateral Q strictly convex if each of its four interior angles measures less than 180°. Such a quadrilateral can be partitioned into two triangles in two possible ways. Three criteria will be described for choosing a preferred triangulation of a strictly convex quadrilateral.

11.1. THE MAX-MIN ANGLE CRITERION.

Choose the triangulation of Q that maximizes the minimum interior angle of the two resulting triangles. Either choice can be made in the case of a tie. For example, in Fig. 12 $\angle cab$ is the smallest angle in triangles f_1 and g_1, $\angle cdb$ is the

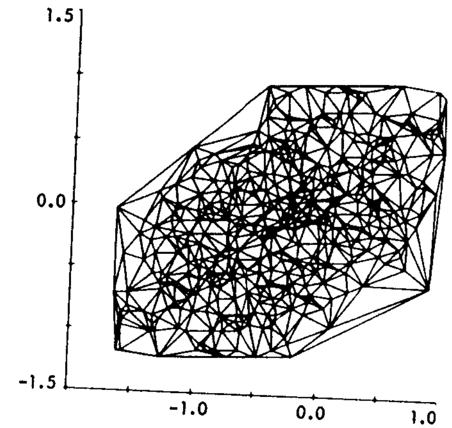

Fig. 9. Thiessen Triangular Grid for Example 3: 500 Points and 985 Triangles.

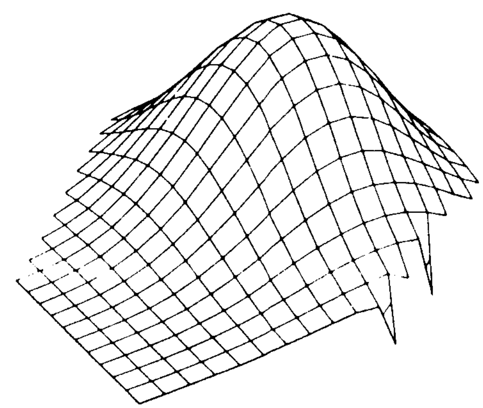

Fig. 10. Perspective Plot for Example 3.

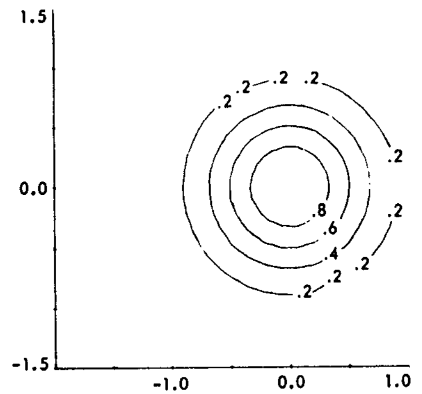

Fig. 11. Contour Plot for Example 3.

smallest angle in triangles f_2 and g_2, and $\angle cdb$ is larger than $\angle cab$. Thus, the triangulation (b) is preferred over (a).

11.2. THE CIRCLE CRITERION.

Let K denote a circle passing through three of the vertices of a strictly convex quadrilateral Q. If the fourth vertex is interior to K, insert the diagonal from this fourth vertex to the opposite vertex. If the fourth vertex is exterior to K, insert the other diagonal. If the fourth vertex is on K, insert either diagonal (see Fig. 13 for an example). Note that when all four vertices are not on a common circle, the same triangulation will be selected regardless of which set of three vertices is used to construct the circle.

11.3. THE THIESSEN REGION CRITERION.

Let R_a denote the closure of the region of the plane consisting of all points that are closer to point a than to points b, c, or d. Similarly define regions R_b, R_c, and R_d surrounding points b, c, and d, respectively. These regions are called Thiessen regions following Powell (1976) and Rhynsburger (1973). These proximity regions are also identified by other names in the mathematical literature.

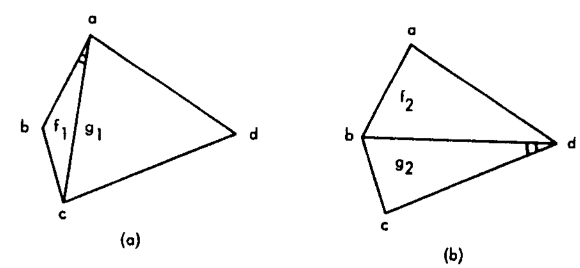

Fig. 12. The Max-Min Angle Criterion.

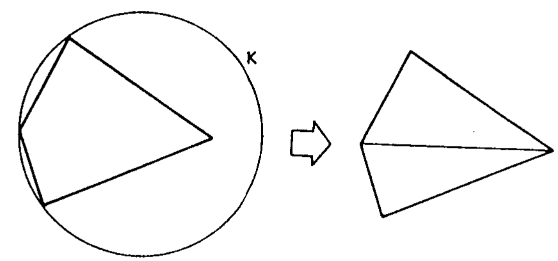

Fig. 13. The Circle Criterion.

Two of the points a,b,c, or d will be called Thiessen neighbors if their Thiessen regions are in contact. They are strong Thiessen neighbors if the contact is along a line segment of nonzero length. They are weak Thiessen neighbors if the contact is at one point only.

To triangulate a strictly convex quadrilateral Q, insert the diagonal that connects a pair of strong Thiessen neighbors. A strictly convex quadrilateral can have at most one pair of opposite vertices that are strong Thiessen neighbors. If neither pair of opposite vertices are strong Thiessen neighbors, then both pairs will be weak neighbors and either diagonal can be inserted (see Fig. 14 for an example).

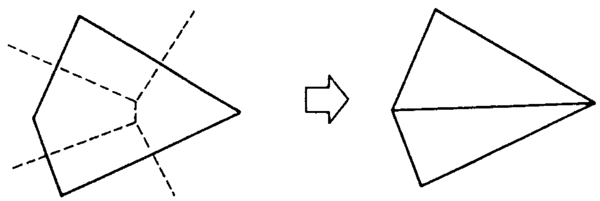

Fig. 14. The Thiessen Region Criterion.

11.4. EQUIVALENCE OF THESE THREE CRITERIA FOR STRICTLY CONVEX QUADRILATERALS.

The first observation to be made about these three criteria is that they give identical results for strictly convex quadrilaterals. This can be verified by noting that all three criteria have the same neutral case and then studying perturbations from the neutral case.

The neutral case for all three criteria is the case in which all four vertices of the quadrilateral lie on a common circle.

To verify this last statement consider a quadrilateral Q whose vertices a,b,c, and d all lie on a common circle K. See Fig. 15. Suppose arc $\widehat{bc}$ is shorter than arcs $\widehat{cd}$, $\widehat{da}$, or $\widehat{ab}$. If the angular measure of arc $\widehat{bc}$ is 2θ, then angles cab and cdb are each of measure θ, and these two angles are each the minimum angle for one of the possible triangulations. Thus, this is a tie case for the max-min angle criterion.

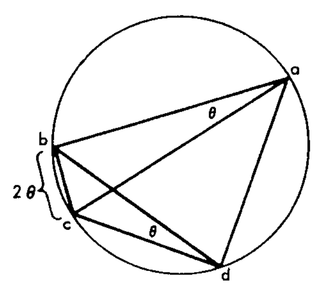

Fig. 15. The Neutral Case for the Max-Min Angle Criterion.

Constructing Thiessen regions for the case of four points on a common circle results in the four Thiessen regions meeting at the center of the circle, as in Fig. 16. Thus, each pair of opposite vertices are weak Thiessen neighbors, which is the neutral case for this criterion.

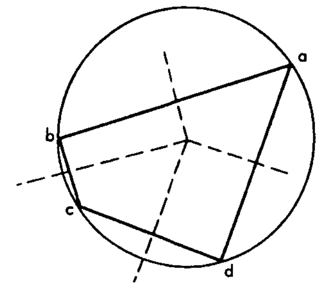

Fig. 16. The Neutral Case for the Thiessen Region Criterion.

Further analysis, the details of which will be omitted, shows that moving one point, say point d in Figs. 15 and 16, inside circle K causes $\angle cdb$ to become larger and causes points b and d to become strong Thiessen neighbors. Thus, all three criteria will choose to introduce the edge $\overline{bd}$.

11.5. A LOCAL OPTIMIZATION PROCEDURE.

Let e denote an internal edge in a triangular grid T. Consider the quadrilateral Q formed by the two triangles having e as a common edge. If Q is not strictly convex then e cannot be considered for swapping. Otherwise, if Q is strictly convex, apply any one of the three equivalent criteria discussed in the preceding sections. Replace e by the other diagonal of Q if this is preferred by the criterion. Otherwise if e is preferred or if the decision is neutral, leave e as it is.

If the criterion used is the circle test and if the circle used is the circumcircle of one of the triangles containing the edge e, then it is not necessary to do an initial test for Q being strictly convex since the correct decision will be made anyway. That is, if Q is not strictly convex, then the circumcircle of one triangle will not enclose the vertex of the other triangle so the decision will be not to swap the edge e.

An internal edge of a triangulation T will be called locally optimal if application of the local optimization procedure to it would not swap it.

12. GLOBAL CONSEQUENCES OF THE LOCAL OPTIMIZATION PROCEDURE.

The local optimization procedure has a number of consequences that are useful in suggesting, and proving properties of, a variety of possible triangulation algorithms.

12.1. A LINEAR ORDERING OF TRIANGULATIONS.

Let S be a set of n points in the plane and let $\mathcal{T}$ denote the set of all triangulations of S. As has previously been noted, all triangulations $T \in \mathcal{T}$ have the same number of triangles, say n_t.

With each $T \in \mathcal{T}$ associate an indicator vector of n_t components constructed as follows: Determine the measure of the smallest angle in each of the n_t triangles of T and sort these angular measures in nondecreasing order. The triangulations in $\mathcal{T}$ can then be linearly ordered by the lexicographic ordering of their associated vectors.

12.1.1. THEOREM.

Let T be a triangulation of a finite point set S. Let e be an internal edge of T. Suppose application of the local optimization procedure to e leads to a swap, replacing e by a new edge e', and thus producing a new triangulation T' of S. Then $T < T'$; i.e., T' strictly follows T in the linear ordering defined above.

Proof. Let v be the indicator vector for T. The measures of the smallest angles in the two triangles in T sharing the edge e occur as two of the components of v, say v_j and v_k, with $j < k$ and thus $v_j \le v_k$. Since a swap was made when e was tested, the smallest angles in both of the new triangles of T' sharing the edge e' must be strictly greater than v_j. It follows that the indicator vector v' for T' strictly follows v in lexicographic order and thus $T < T'$. ∎

This theorem can be used to prove finite termination for a variety of possible triangulation algorithms that repeatedly apply the local optimization procedure to all internal edges of a sequence of triangulations. Since there are only a finite number of possible triangulations of a point set S and each swap produced by the local optimization procedure causes a strict advance through a linear ordering of triangulations, it follows that after some finite number of swaps a triangulation T* will be reached such that each internal edge in T* will be left unswapped when tested by the local optimization procedure; i.e., all internal edges are locally optimal.

12.1.2. THEOREM.

All internal edges of a triangulation T of a finite point set S are locally optimal if and only if no point of S is interior to any circumcircle of a triangle of T.

Proof of "if". Assume no point of S is interior to any circumcircle of a triangle of T. Consider the application of the local optimization procedure to any internal edge e in T. Let f and g denote the two triangles sharing the edge e. Let d denote the vertex of g opposite to edge e. By hypothesis d is not interior to the circumcircle of triangle f. Thus the local optimization procedure will not swap e; that is, e is already locally optimal.

Proof of "only if". Assume all internal edges of T are locally optimal. Suppose the theorem is false; i.e., suppose there is a triangle f in T with vertices a, b, and c and circumcircle K such that there is a point p of S interior to K.

Without loss of generality assume edge $\overline{ac}$ is the nearest edge of Δabc to p as in Fig. 17. Denote the perpendicular distance from $\overline{ac}$ to p by δ. Among all triangles of T, whose circumcircle contains p as an interior point, assume without loss of generality that none is at a distance of less than δ from p.

Since p is on the opposite side of $\overline{ac}$ from b, the edge $\overline{ac}$ is not a boundary edge of T. Thus there is another triangle, say Δacq, sharing the edge $\overline{ac}$. The vertex q cannot be interior to the circle K as this would contradict the hypothesis that edge $\overline{ac}$ is locally optimal. Thus q is on K or exterior to K.

Without loss of generality, assume edge $\overline{cq}$ is the closest edge of Δacq to p as in Fig. 18. Note that the distance from

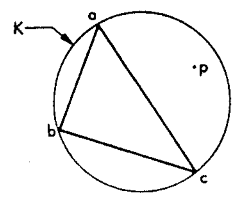

Fig. 17. Theorem 12.1.2.

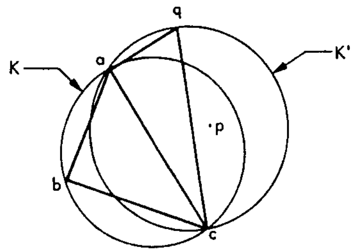

Fig. 18. Theorem 12.1.2.

$\overline{cq}$ to p is less than δ. Thus a contradiction will be reached if it is shown that the circumcircle K' of $\triangle$acq contains p as an interior point.

If q is on K, then K' = K, and so p lies interior to K'. If q is outside K, then K' intersects K only at a and c and encloses all of the interior of K to the right of $\overline{ac}$ in Fig. 18. In particular K' encloses p.∎

12.2. THE GLOBAL THIESSEN TRIANGULATION.

The notion of Thiessen regions and an associated triangulation was defined for strictly convex quadrilaterals in Sec. 11.3. The same definitions can be applied to any finite set of points in the plane. First the Thiessen region surrounding each point can be determined. Then line segments can be drawn connecting points that are strong Thiessen neighbors. This will provide a polygonal grid.

For points in general position, the grid will in fact consist of triangles. Any polygon of the grid that is not a triangle will have all of its vertices on a circle and all nonadjacent pairs of vertices will be weak Thiessen neighbors. Any such k-point polygon can be triangulated by connecting any k-3 pairs of its vertices as long as no crossing lines are drawn. A triangulation in which all strong Thiessen neighbors are connected will be called a Thiessen triangulation.

12.2.1. LEMMA.

Let S be a set of n points in the plane and let $\tilde{S}$ be a subset of S. Two points of $\tilde{S}$ that are Thiessen neighbors in S remain so in $\tilde{S}$ and if they are strong Thiessen neighbors in S they remain strong Thiessen neighbors in $\tilde{S}$.

Proof. Consider the effect of removing one point, say p, from S, leaving a subset S'. The only change that takes place in transforming the Thiessen regions for S to form the Thiessen regions for S' is a redistribution of the part of the plane that was the Thiessen region for p. This region will be partitioned, with various portions being absorbed into the neighboring Thiessen regions. In this process no boundaries between pairs of remaining Thiessen regions are shortened. Thus neighbors remain neighbors and strong neighbors remain strong neighbors. Clearly the same is true for the removal of any number of points from a finite set, as the removals can be done one at a time.∎

12.2.2. THEOREM.

All internal edges of a triangulation T of a finite point set S are locally optimal if and only if T is a Thiessen triangulation of S.

Proof of "if". Assume T is a Thiessen triangulation of S. Let e be an internal edge of T connecting vertices a and c and belonging to triangles $\triangle$abc and $\triangle$cda. If the quadrilateral Q = abcd is not strictly convex, then e cannot be swapped and is thus locally optimal.

Consider then the case of Q being strictly convex. By hypothesis, a and c are Thiessen neighbors in S. By Lemma 12.2.1 they are also Thiessen neighbors relative to the point set {a, b, c, d}. With the quadrilateral Q = abcd being strictly convex, this implies e is locally optimal.

Proof of "only if". Assume all internal edges of T are locally optimal. Suppose the theorem is false. Then there is some pair of strong Thiessen neighbors in S, say points p and q, that are not connected by an edge in the triangulation T.

Define B to be the polygonal curve whose constituent line segments are the segments that occur as edges opposite vertex q in those triangles that have q as a vertex. If q is not a boundary point of T, then B is a closed polygon with q in its interior and p lying exterior to it. Clearly a line segment from p to q would intersect B.

If q is a boundary point of T, then B is an open polygonal curve with end points on the boundary of T at the two points immediately adjacent to q on each side of q along the boundary. Although in this case B does not surround q, it still follows that $\overline{pq}$ must intersect B owing to the convexity of the region covered by T.

Since p and q are strong Thiessen neighbors in S, there can be no other points of S on the line segment $\overline{pq}$. Thus the intersection of $\overline{pq}$ with B is not at a point of S on B but must be strictly between a pair of points of S on B, say points r and s.

Thus the triangle Δqrs is a triangle of T having the property that r and s are on strictly opposite sides of $\overline{pq}$ and p and q are on strictly opposite sides of $\overline{rs}$, as is illustrated in Fig. 19.

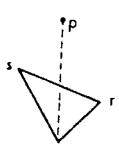

Fig. 19. Theorem 12.2.2.

By hypothesis, all internal edges of T are locally optimal. By Theorem 12.1.2 this implies that point p is not in the interior of the circumcircle K of Δqrs. From the equivalence

of the circle test and the Thiessen criterion for strictly convex quadrilaterals, this would imply that p is not a strong Thiessen neighbor of q relative to the point set {p, s, q, r}.

By Lemma 12.2.1, however, p and q are strong Thiessen neighbors relative to {p,s,q,r} since they are strong Thiessen neighbors in S. Thus a contradiction has been reached.∎

13. McLAIN'S TRIANGULATION METHOD.

The triangulation method described in McLain (1976) builds a grid one triangle at a time in such a way that each triangle constructed is a triangle of the final grid. This is in contrast to methods that involve triangle modification steps such as are used in Sec. 6, in Lawson (1972), and by Frank Little of the University of Utah CAGD group.

The paper, McLain (1976), with the subsequent errata, leaves open the question of the characterization of the grid the algorithm produces. We find that the results of the preceding sections can be used to show that the grid produced by McLain's method is in fact a Thiessen grid.

Let S be a set of n points. Define T_0 to be a single edge belonging to some Thiessen triangulation for S. For example, T_0 could be a boundary edge of the convex hull of S. For $k \geq 1$ define T_k to be a configuration of k triangles that is a subset of some Thiessen triangulation of S and contains T_{k-1} as a subset. In general, the configurations T_k are not necessarily convex.

Given some T_k, how can one more triangle of a Thiessen triangulation of S be found to advance to T_{k+1}?

Let $\overline{ab}$ be an edge belonging to just one triangle, say Δabc, in T_k. Let S_k be the subset of S consisting of the points lying on the opposite side of $\overline{ab}$ from c. (If there are none, then try another edge as $\overline{ab}$ or terminate.)

From our inductive assumption that T_k is a subset of a Thiessen triangulation of S, there must be a point p in S_k such that adjoining Δabp to T_k gives a configuration T_{k+1} that is also a subset of a Thiessen triangulation of S.

By Theorems 12.1.2 and 12.2.2 we know that the triangle Δabp must not contain any points of S_k in its interior. Such a triangle is just what is selected by McLain's method, since he selects p such that the signed distance from $\overline{ab}$ of the center of the circumcircle of Δabp is the algebraically smallest possible among all choices of p in S_k. The signed distance from $\overline{ab}$ is positive on the side of $\overline{ab}$ opposite to c.

14. LIMITS ON GRID CHANGES WHEN ADDING A NEW POINT.

This section presents results that limit the amount of edge testing needed in algorithms such as ours in Sec. 6 that transform from the Thiessen triangulation of one point set to that for the set augmented by one new point.

Let S_{n-1} be a set of n-1 points in the plane. Let p be a point not in S_{n-1} and define $S_n = S_{n-1} \cup \{p\}$. Let T^*_{n-1} be a Thiessen triangulation for S_{n-1}.

Given T^*_{n-1}, an initial triangulation $T_n^{(1)}$ for S_n can be constructed by inserting all edges that connect the new point p to points of S_{n-1} without crossing edges already present in T^*_{n-1}. A special case arises if p falls on an edge, say $\overline{ac}$, already present in T^*_{n-1}. Then the edge $\overline{ac}$ must be replaced by the two edges $\overline{ap}$ and $\overline{pc}$. For our theoretical discussion it will be easier to assume p does not fall on an edge of T^*_{n-1}. The case of p arbitrarily close to an edge of T^*_{n-1} is of course permitted, and analysis of this case can be used to justify the replacement of $\overline{ac}$ by $\overline{ap}$ and $\overline{pc}$.

An edge e in a triangulation will be called converged if it is locally optimal in the present triangulation and if in addition it can be proved that no sequence of applications of the local optimization procedure to the various edges could lead to a decision to swap e.

Assuming p does not lie on any edge of T^*_{n-1}, it will be shown that all of the initial edges inserted connecting p to points of S_{n-1} are converged. Any edge opposite to p in a triangle must be tested once using the local optimization procedure. If it remains unchanged after testing, then it is converged. If it is swapped by the procedure, then the new edge introduced is converged and the two edges opposite p in the two new triangles must each be tested.

14.1. THEOREM.

Assume p is strictly outside the convex hull of S_{n-1} and $T_n^{(1)}$ is formed by connecting p to all boundary points of T^*_{n-1} that can be reached without crossing any edges of T^*_{n-1}. Then all of the new edges are converged.

Proof. Let $\overline{pq}$ be a new edge connecting p to a point q on the boundary of T^*_{n-1}. If in the course of triangulating S_n it is ever to be decided to swap $\overline{pq}$ for some other edge $\overline{ab}$, then a and b must (at least) be points of S_{n-1} such that a and b are on strictly opposite sides of $\overline{pq}$ and p and q are on strictly opposite sides of $\overline{ab}$.

This is impossible since the line segment $\overline{ab}$ can not pass outside the convex hull of S_{n-1} and thus could not intersect $\overline{pq}$ strictly between the boundary point q and the point p which is exterior to the convex hull of S_{n-1}. ∎

14.2. THEOREM.

Assume p is interior to the convex hull of S_{n-1} and in fact interior to some triangle Δabc of T^*_{n-1}. Assume $T_n^{(1)}$ is formed by connecting p to a, b, and c. These three new edges are converged.

Proof. Since T^*_{n-1} is a Thiessen triangulation of S_{n-1}, the circumcircle K of Δabc contains no points of S_{n-1} in its interior. If in the course of triangulating S_n it is ever decided to swap $\overline{pa}$, for instance, for some other edge, $\overline{rs}$, then r and s must not be interior to K, r and s must be on strictly opposite sides of $\overline{pa}$, p and a must be on strictly opposite sides of $\overline{rs}$, and a must be strictly outside the circle K' through r, p, and s (see Fig. 20).

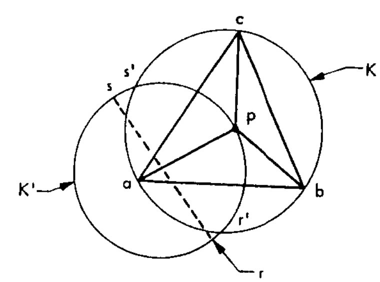

Fig. 20. Theorem 14.2.

Partition circle K' into the three arcs $\widehat{rp}$, $\widehat{ps}$, and $\widehat{sr}$. Note that arc $\widehat{rp}$ intersects circle K since p is inside K and r is on or outside K. Call this intersection point r'. Similarly, arc $\widehat{ps}$ intersects circle K, say at a point s'. Then arc $\widehat{r'ps'}$ of K' is inside K because p is inside K and r' and s' are intersection points of K' with K. It follows that arc $\widehat{s'srr'}$ of K' is outside K. The arc of K between s' and r' that lies inside K' contains the point a. Thus it is impossible for a to be outside K' as it must be for $\overline{pa}$ to be swapped.∎

14.3. THEOREM.

Let $T_n^{(i)}$ be a triangulation of $S_n = S_{n-1} \cup \{P\}$. Let Δcbp and Δabc be adjacent triangles of $T_n^{(i)}$ and assume Δabc was also a triangle of T^*_{n-1}. Suppose application of the local optimization procedure to edge $\overline{bc}$ leads to a swap, replacing $\overline{bc}$ by $\overline{pa}$. Then edge $\overline{pa}$ is converged.

Proof. Note that the symbols a, b, and c do not necessarily denote the first vertices to which p was connected as was the case in Theorem 14.2. The notation for this theorem was selected, however, to permit the proof to be identical to that of Theorem 14.2 (see Fig. 21).∎

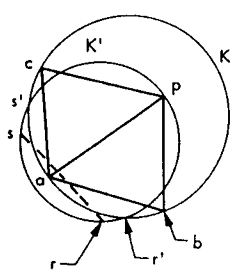

Fig. 21. Theorem 14.3.

15. CONCLUSIONS.

Triangular grid construction is certainly not as well understood as sorting of scalars, but at least a general notion of what to expect in running time, storage usage, and properties of the final triangulation is now available. Thus an algorithm with an $O(n^2)$ time estimate must be regarded as inefficient except possibly for small n. There remain a wide variety of possible triangulation algorithms with time estimates in the neighborhood of $O(n^{4/3})$. It will require more time, experience and direct comparative testing to sort these out.

C^1 interpolation on triangles is still very ad hoc. Three different methods are used by Akima (1975), McLain (1976), and the present paper. It appears that our method requires the least amount of auxiliary stored information per node (two first partial derivatives per node) and the fewest number of operations per interpolated value once the auxiliary nodal information has been computed and stored. Direct comparative tests would be needed, however, to assess accuracy and actual execution times.

There is much scope for additional work on this problem, including generalizations such as smoothing instead of interpolation, the introductions of constraints, and permitting the domain of the independent variables to be the surface of a sphere or three-dimensional space instead of the plane.

REFERENCES

1. Hiroshi Akima, (1975), A method of bivariate interpolation and smooth surface fitting for values given at irregularly distributed points, U.S. Depart. of Commerce OT Report 75-70. An improved version of this algorithm will appear in ACM TOMS.
2. Robert E. Barnhill, (1977), Representation and approximation of surfaces, this volume, pp. 68-119.
3. Robert E. Barnhill and Gregory M. Nelson, (1974), Reproducing kernel functions for Sard spaces of type B*, SIAM J. Numer. Anal., pp. 37-44.
4. Bengt-Erik Bengtsson and Stig Nordbeck, (1964), Construction of isarithms and isarithmic maps by computers, BIT, 4, pp. 87-105.
5. Garret Birkhoff and L. E. Mansfield, (1974), Compatible triangular finite elements, Jour. Math. Anal. Appl. 47, pp. 531-553.

6. R. W. Clough and J. L. Tocher, (1965), Finite element stiffness matrices for analysis of plates in bending, Proc. Conf. Matrix Methods in Struct. Mech., Air Force Inst. of Tech., Wright-Patterson A.F.B., Ohio.
7. W. F. Eddy, (1976), A new convex hull algorithm for planar sets, Carnegie-Mellon Univ., to appear in ACM TOMS.
8. Bradley Efron, (1965), The convex hull of a random set of points, Biometrika, 52, 3 and 4, p. 331.
9. Richard Franke, (1975), Locally determined smooth interpolation at irregularly spaced points in several variables, Naval Postgraduate School, AD-A010 814, Monterey, Calif
10. A. R. Forrest, (1972), On Coons and other methods for representation of curved surfaces, Comput. Graphics Inform. Process., 1, pp. 341-359.
11. R. L. Graham, (1972), An efficient algorithm for determining the convex hull of a finite planar set, Information Processing Letters 1, pp. 132-133.
12. R. A. Jarvis, (1973), On the identification of the convex hull of a finite set of points in the plane, Information Processing Letters 2, pp. 18-21.
13. C. L. Lawson, (1972), Generation of a triangular grid with application to contour plotting, Jet Propulsion Laboratory Section 914 Tech. Memo. No. 299 (an internal report).
14. C. L. Lawson, (1976a), C^1-Compatible interpolation over a triangle, Jet Propulsion Laboratory TM 33-770.
15. C. L. Lawson, (1976b), Integrals of a C^1-compatible triangular surface element, Jet Propulsion Laboratory TM 33-808.
16. L. E. Mansfield, (1972), Optimal approximation and error bounds in spaces of bivariate functions, J. Approx. Th., 5, pp. 77-96.
17. A. D. Maude, (1973), Interpolation - mainly for graph plotters, Comput. J., 16, pp. 64-65.
18. D. H. McLain, (1974), Drawing contours from arbitrary data points, Computer J., 17, pp. 318-324.

19. D. H. McLain, (1976), Two dimensional interpolation from random data, Comput. J., 19, No. 2, pp. 178-181. (See also Errata for this paper in Comput. J., 19, No. 4, Nov. 1976, p. 384.)
20. G. M. Nielson, (1974), Multivariate smoothing and interpolating splines, SIAM J. Numer. Anal., pp. 435-446.
21. M. J. D. Powell, (1976), Numerical methods for fitting functions of two variables, A.E.R.E. Harwell, Computer Science and Systems Division Report 30, presented at the IMA Conference on the State-of-the-Art in Numerical Analysis, University of York, April, 1976.
22. M. J. D. Powell and M. A. Sabin, (1976), Piecewise quadratic approximations on triangles, Univ. of Cambridge, to appear in ACM TOMS.
23. F. P. Preparata and S. J. Hong, (1977), Convex hulls of finite sets of points in two and three dimensions, Commun. ACM., 20, No. 2, pp. 87-93.
24. H. Raynaud, (1970), Sur l'enveloppe convex des nuages de points aleatoires dans R^n. I, J. Appl. Prob., 7, pp. 35-48.
25. A. Renyi and R. Sulanke, (1963-1964), Uber die convexe hulle von is zufallig gewahlten punkten, I and II, Z. Wahr. 2, pp. 75-84, 3, 138-148.
26. D. Rhynsburger, (1973), Analytic delineation of Thiessen polygons, Geograph. Anal., 5, pp. 133-144.
27. Arthur Sard, (1963), Linear approximation, Mathematical Surveys, No. 9, American Mathematical Society, Providence, R.I.
28. L. L. Schumaker, (1976), Fitting surfaces to scattered data, Approximation Theory II, edited by G. G. Lorentz, C. K. Chui, and L. L. Schumaker, Academic Press, pp. 203-268.

29. Donald Shepard, (1968), A two-dimensional interpolation function for irregularly-spaced data, Proc. 23rd Nat. Conf. ACM, pp. 517-524.

The author was supported by the National Aeronautics and Space Administration under Contract No. NAS 7-100.

Information Systems Division
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91103

Mathematical Softwa

W. R. Cr
L. D. Fos

ABSTRA

Locally constructed collec
tines are gradually being replac
that has been produced for broad
process of producing such softwa
analysis, and proceeds through s
documentation, to extensive test
ution and support of the softwar
demanding and costly activities
of skills that they are carried
jects. The costs and effort are
of high quality software, the ef
for general distribution, and the
conduit from research to applica

In this paper we first revi
developments in the field of mat
we examine the technical problems
production as an intellectual ac
criptions also serve to characte
software. Next we sketch three
jects with attention to their em
organization, and costs. Finall
future directions for mathematic
extrapolations of the present in
government laboratories, and pri