MATHEMATICS OF COMPUTATION
VOLUME 40, NUMBER 161
JANUARY 1983, PAGES 253-271

A Method for Interpolating Scattered Data Based Upon a Minimum Norm Network*

By Gregory M. Nielson

Abstract. A method for interpolating scattered data is described. Given (x_i, y_i, z_i), $i = 1,\dots,N$, a bivariate function S with continuous first order partial derivatives is defined which has the property that $S(x_i, y_i) = z_i$, $i = 1,\dots,N$. The method is based upon a triangulation of the domain and a curve network which has certain minimum pseudonorm properties. Algorithms and examples are included.

1. Introduction. In this paper, we present a new method for interpolating scattered data. Given the data (x_i, y_i, z_i), $i = 1,\dots,N$, we describe the construction of a bivariate function S which has continuous first order partial derivatives and $S(x_i, y_i) = z_i$, $i = 1,\dots,N$. The method consists of three separate steps:

(i) *Triangulation.* The points $V_i = (x_i, y_i)$, $i = 1,\dots,N$, are used as the vertices of a triangulation of a domain D.

(ii) *Curve Network.* The approximation S and its first order partial derivatives, S_x and S_y, are defined on the subset consisting of the union of all edges.

(iii) *Blending.* S is extended to D by means of a blending method which will assume arbitrary position and slope on the boundary of a triangular domain.

The basic idea of an interpolant which is defined in a piecewise fashion over triangles is not new. Both Lawson [7] and Akima [1] have described such methods. Lawson's paper contains a good discussion of many of the aspects of triangulating the convex hull of V_i, $i = 1,\dots,N$. Both of these methods make use of a discrete C^1 interpolant (i.e., a C^1 finite element) for each triangle followed by a local method for estimating certain partial derivatives. Even though our method is based upon the approach of a curve network followed by the use of a triangular blending (transfinite) interpolant, the particular method we eventually propose can be viewed as an assembly of discrete C^1 interpolants along with a technique for estimating partial derivatives.

2. The Curve Network. We assume $N \geqslant 3$ and that the points V_i, $i = 1,\dots,N$, are not collinear. Let T_{ijk} denote the triangle with vertices V_i, V_j and V_k, $i \neq j \neq k \neq i$. The list of triple indices which determines the triangulation is denoted by I_t so that $D = \bigcup_{ijk \in I_t} T_{ijk}$. Let e_{ij} represent the line segment with endpoints V_i and V_j, and let $N_e = \{ij\colon i, j \in \{\alpha, \beta, \gamma\}, \alpha\beta\gamma \in I_t\}$ be a list of the indices representing the edges of the triangulation. In terms of this notation, the domain of the curve network is

Received January 22, 1979; revised April 13, 1981 and June 21, 1982.

1980 *Mathematics Subject Classification.* Primary 65D05; Secondary 41A15.

Key words and phrases. Bivariate interpolation, scattered data interpolation, random data interpolation, splines, multivariate approximation and interpolation.

*This research was supported by the Office of Naval Research under Contract NR 044-443.

$E = \bigcup_{ij \in N_e} e_{ij}$. The derivative normal to an edge e_{ij} is given by

$$\frac{\partial F}{\partial n_{ij}} = \frac{(y_j - y_i)}{\|e_{ij}\|}\frac{\partial F}{\partial x} - \frac{(x_j - x_i)}{\|e_{ij}\|}\frac{\partial F}{\partial y},$$

where $\|e_{ij}\|$ is the length of e_{ij}. The derivative along an edge is given by

$$\frac{\partial F}{\partial e_{ij}} = \frac{(x_j - x_i)}{\|e_{ij}\|}\frac{\partial F}{\partial x} + \frac{(y_j - y_i)}{\|e_{ij}\|}\frac{\partial F}{\partial y}.$$

Therefore,

$$\frac{\partial F}{\partial x} = \frac{(x_j - x_i)}{\|e_{ij}\|}\frac{\partial F}{\partial e_{ij}} + \frac{(y_j - y_i)}{\|e_{ij}\|}\frac{\partial F}{\partial n_{ij}},$$

$$\frac{\partial F}{\partial y} = \frac{(y_j - y_i)}{\|e_{ij}\|}\frac{\partial F}{\partial e_{ij}} - \frac{(x_j - x_i)}{\|e_{ij}\|}\frac{\partial F}{\partial n_{ij}},$$

and so it is clear that if S and $\partial S/\partial n_{ij}$ are known on each edge, the information required for step (ii) is available. It is more convenient to specify S and its normal derivatives on E since these values can be defined independent of each other at all points except the vertices. We will first define S on E, but prior to this, we review some material concerning univariate cubic splines which motivates our particular choice of the curve network.

Given the data (t_i, s_i), $i = 1, \ldots, n$, where $t_1 < t_2 < \cdots < t_n$, the univariate natural spline of interpolation can be characterized (cf. de Boor and Lynch [4]) as the unique solution to the problem

$$\text{(2.1)} \qquad \begin{gathered} \operatorname*{Min}_{f \in H[t_1, t_n]} \int_{t_1}^{t_n} [f''(t)]^2 \, dt, \\ \text{subject to: } f(t_i) = s_i, \end{gathered}$$

where $H[t_1, t_n] = \{f\colon f \in C[t_1, t_n],\ f' \text{ is absolutely continuous, } f'' \in L^2[t_1, t_n]\}$. From this point of view, the mathematical spline is an analogue of the physical spline. As a result of this minimization, it can be shown that s is a piecewise cubic polynomial which has a continuous second derivative and $s''(t_1) = s''(t_n) = 0$. Towards the definition of a curve network with an analogous characterization, we introduce $C[E] = \{F\colon F$ is the restriction to E of some C^1 function defined on D and the univariate function obtained as the restriction of F to e_{ij} is an element of $H[e_{ij}]\}$. Analogous to the minimum pseudonorm property of univariate splines, we consider the problem of finding an interpolating curve network which minimizes

$$\text{(2.2)} \qquad \sigma(F) = \sum_{ij \in N_e} \int_{e_{ij}} \left[\frac{\partial^2 F}{\partial e_{ij}^2}\right]^2 ds_{ij},$$

where ds_{ij} represents the element of arc length on the curve consisting of the line segment e_{ij}.

We find it convenient to view each $F \in C[E]$ as a collection of univariate functions

$$\text{(2.3)} \qquad f_{ij}(t) = F\big((1-t)V_i + tV_j\big), \qquad ij \in N_e,\ 0 \leqslant t \leqslant 1.$$

With the following parametrization of the curve e_{ij}:

$$x(t) = (1-t)x_i + tx_j, \qquad y(t) = (1-t)y_i + ty_j,$$

and the fact that

$$\frac{1}{\|e_{ij}\|^2} f_{ij}'' = \frac{\partial^2 F}{\partial e_{ij}^2}\Big|_{e_{ij}},$$

we have that

$$\begin{aligned}\sigma(F) &= \sum_{ij \in N_e} \int_0^1 \left[\frac{f_{ij}''(t)}{\|e_{ij}\|^2}\right]^2 \sqrt{[x'(t)]^2 + [y'(t)]^2}\, dt \\ &= \sum_{ij \in N_e} \frac{1}{\|e_{ij}\|^3} \int_0^1 [f_{ij}''(t)]^2\, dt.\end{aligned}$$

THEOREM 2.1. *Let $S \in C[F]$ be the unique piecewise cubic network with the properties that $S(V_i) = z_i$, $i = 1,\dots,N$, and*

$$\text{(2.4)} \quad \begin{aligned}&\sum_{ij \in N_i} \frac{(x_j - x_i)}{\|e_{ij}\|^3}\Big[(x_j - x_i)S_x(V_i) + (y_j - y_i)S_y(V_i) \\ &\qquad + \frac{1}{2}(x_j - x_i)S_x(V_j) + \frac{1}{2}(y_j - y_i)S_y(V_j) + \frac{3}{2}(z_i - z_j)\Big] = 0,\\ &\sum_{ij \in N_i} \frac{(y_j - y_i)}{\|e_{ij}\|^3}\Big[(x_j - x_i)S_x(V_i) + (y_j - y_i)S_y(V_i) \\ &\qquad + \frac{1}{2}(x_j - x_i)S_x(V_j) + \frac{1}{2}(y_j - y_i)S_y(V_j) + \frac{3}{2}(z_i - z_j)\Big] = 0,\end{aligned}$$

where

$$N_i = \{ij\colon e_{ij} \textit{ is the edge of the triangulation with the endpoint } V_i\}.$$

Then, among all functions $F \in C[E]$, $F(V_i) = z_i$, $i = 1,\dots,N$, the function S uniquely minimizes $\sigma(F)$.

Proof. We first define the inner product

$$\langle F, G\rangle = \sum_{ij \in N_e} \frac{1}{\|e_{ij}\|^3} \int_0^1 f_{ij}''(t) g_{ij}''(t)\, dt$$

and note that

$$\sigma(F) - \sigma(S) = \langle F - S, F - S\rangle + 2\langle S, F - S\rangle.$$

For the moment, we assume that (2.4) has a solution and write

$$\begin{aligned}s_{ij}(t) &= t^2(3-2t)z_j + (1-t)^2(2t+1)z_i \\ &\quad + t(1-t)^2\big[(x_j - x_i)S_x(V_i) + (y_j - y_i)S_y(V_i)\big] \\ &\quad + t^2(t-1)\big[(x_j - x_i)S_x(V_j) + (y_j - y_i)S_y(V_j)\big].\end{aligned}$$

If F is any element of $C[E]$ which interpolates, then

$$\int_0^1 s_{ij}''(t)\left[f_{ij}''(t)-s_{ij}''(t)\right]dt$$

$$= s_{ij}''\left[f_{ij}'-s_{ij}'\right]\Big|_0^1 - s_{ij}'''\left[f_{ij}-s_{ij}\right]\Big|_0^1 + \int_0^1 s_{ij}^{\text{iv}}(t)\left[f_{ij}(t)-s_{ij}(t)\right]dt$$

$$= s_{ij}''\left[f_{ij}'-s_{ij}'\right]\Big|_0^1.$$

Since

$$\begin{aligned} s_{ij}''(1) &= 6[z_i-z_j] + 2\left[(x_j-x_i)S_x(V_i)+(y_j-y_i)S_y(V_i)\right] \\ &\quad +4\left[(x_j-x_i)S_x(V_j)+(y_j-y_i)S_y(V_j)\right], \\ s_{ij}''(0) &= 6[z_j-z_i] - 4\left[(x_j-x_i)S_x(V_i)+(y_j-y_i)S_y(V_i)\right] \\ &\quad -2\left[(x_j-x_i)S_x(V_j)+(y_j-y_i)S_y(V_j)\right], \end{aligned}$$

we conclude that

$$(2.5)\ \sum_{ij\in N_e}\frac{1}{\|e_{ij}\|^3}\int_0^1 s''(t)\left[f''(t)-s''(t)\right]dt$$

$$= \sum_{i=1}^{N}\left\{\sum_{ij\in N_i}\frac{1}{\|e_{ij}\|^3}\left[6(z_i-z_j)+2(x_j-x_i)\big(S_x(V_j)+2S_x(V_i)\big)\right.\right.$$

$$\left.+2(y_j-y_i)\big(S_y(V_j)+2S_y(V_i)\big)\right]$$

$$\left.\times\left[(x_j-x_i)\left[F_x(V_i)-S_x(V_i)\right]+(y_j-y_i)\left[F_y(V_i)-S_y(V_i)\right]\right]\right\}.$$

The change in summation is allowable because $s_{ij}''(1)=s_{ji}''(0)$. This last equation points out the fact that $\langle S, F-S\rangle$ is independent of whether ij or ji is listed in N_e. This is the reason we were not definite about this before. Applying (2.4) to (2.5), we have that

$$(2.6)\qquad \langle S, F-S\rangle = 0,$$

and so

$$\sigma(F)-\sigma(S)=\sigma(F-S)\geqslant 0$$

for any curve network F such that $F(V_i)=z_i$. This establishes the minimum property assuming that S exists. The existence of S requires a solution of the $2N$ linear equations of (2.4). We will show that this system has a solution by showing that the homogeneous system ($z_i=0$, $i=1,\ldots,N$) has only the trivial solution. This argument will also establish the uniqueness of S. Let $\bar{S}$ be the piecewise cubic curve network associated with the solution $\bar{S}_x(V_i)$, $\bar{S}_y(V_i)$, $i=1,\ldots,N$, of the homogeneous system. The same line of reasoning which led to (2.6) can be used to conclude that

$$\langle \bar{S}, \bar{S}\rangle = 0.$$

That is, we replace both S and $F - S$ with $\bar{S}$. Therefore, $\int_0^1[\bar{s}''_{ij}(t)]^2\,dt = 0$, $ij \in N_e$, which implies that $\bar{s}_{ij}$ is linear. But $\bar{s}_{ij}(0) = \bar{s}_{ij}(1) = 0$, and so

$$(x_j - x_i)\bar{S}_x(V_i) + (y_j - y_i)\bar{S}_y(V_i) = 0, \qquad ij \in N_i,\ i = 1,\dots,N.$$

Since each V_i is the vertex of some nondegenerate triangle, this is sufficient to imply that $\bar{S}_x(V_i) = \bar{S}_y(V_i) = 0$, $i = 1,\dots,N$, and so the proof is complete.

COROLLARY 2.2. *If the data* (x_i, y_i, z_i), $i = 1,\dots,N$, *lie on a plane, then the curve network* S *of Theorem* 2.1 *will also lie on this plane.*

Proof. Let P represent the plane, and let

$$\hat{S} = P\,|_E$$

be the restriction of this plane to the edges. Then $\hat{S} \in C[E]$, $\hat{S}$ interpolates and $\sigma(\hat{S}) = 0$. Since the minimum norm network is unique, it must be that case that $S = \hat{S}$.

In order to complete the information required by step (ii), we need to define the normal derivative on each edge. We make a particularly simple choice here and take the normal derivatives to be linear. That is,

$$\frac{\partial S}{\partial n_{ij}}\big((1-t)V_i + tV_j\big) = (1-t)\left[\frac{(y_j - y_i)S_x(V_i) - (x_j - x_i)S_y(V_i)}{\|e_{ij}\|}\right]$$
$$+t\left[\frac{(y_j - y_i)S_x(V_j) - (x_j - x_i)S_y(V_j)}{\|e_{ij}\|}\right], \qquad ij \in N_e.$$

3. The Blending Method. We now discuss the choice of the triangular blending method to be used to extend the curve network to the domain D. For these purposes, we let T represent an arbitrary triangle with vertices V_i, $i = 1, 2, 3$, and $b_i = b_i(x, y)$, $i = 1, 2, 3$, denote the barycentric coordinates given by

$$\begin{aligned} x &= b_1x_1 + b_2x_2 + b_3x_3,\\ y &= b_1y_1 + b_2y_2 + b_3y_3,\\ 1 &= b_1 + b_2 + b_3. \end{aligned}$$

Let $I = \{(1,2,3), (2,3,1), (3,1,2)\}$.

The first method of approximation to assume predescribed values of a function and its first order derivatives on the boundary of a triangle is due to Barnhill, Birkhoff and Gordon [3]. This method requires the specification and compatibility of the cross partials:

$$\frac{\partial^2 F}{\partial e_{ik}\partial e_{ij}}(V_i) = \frac{\partial^2 F}{\partial e_{ij}\partial e_{ik}}(V_i), \qquad (i, j, k) \in I.$$

While those values are obtainable from the curve network information, in general they will not be compatible.

In fact,

$$\frac{\partial^2 S}{\partial e_{ij}\partial e_{ik}}(V_i) = s''_{ij}(0)c_{kji} + \alpha_{ji}d_{kji}, \qquad (i, j, k) \in I,$$

where

$$\alpha_{mn} = (y_m - y_n)(S_x(V_m) - S_x(V_n)) + (x_m - x_n)(S_y(V_n) - S_y(V_m)),$$

$$c_{lmn} = \frac{(x_l - x_n)(x_m - x_n) + (y_l - y_n)(y_m - y_n)}{\|e_{nl}\| \, \|e_{mn}\|^3}$$

and

$$d_{lmn} = \frac{(x_l - x_n)(y_m - y_n) - (y_l - y_n)(x_m - x_n)}{\|e_{nl}\| \, \|e_{mn}\|^3}.$$

Therefore $\partial^2 S(V_i)/\partial e_{ij}\partial e_{ik}$ will involve z_j, $S_x(V_j)$ and $S_y(V_j)$, while $\partial^2 S(V_i)/\partial e_{ik}\partial e_{ij}$ will involve z_k, $S_x(V_k)$ and $S_y(V_k)$, and so in general these two partials will not be equal. A recently developed method which does not explicitly involve or require the compatibility of the cross partials is described in [9]. This method is based upon the combination of three interpolants each having certain miminum norm properties. When the boundary values given by the curve network are substituted into this triangular blending method we obtain the following nine-parameter C^1 interpolant defined over T.

$$\text{(3.1)} \qquad \begin{aligned} S_T(x, y) = \sum_{(i,j,k)\in I} & S(V_i)\left[b_i^2(3 - 2b_i) + 6wb_i(b_k\alpha_{ij} + b_j\alpha_{ik})\right] \\ & + S_k'(V_i)\left[b_i^2 b_k + wb_i(3b_k\alpha_{ij} + b_j - b_k)\right] \\ & + S_j'(V_i)\left[b_i^2 b_j + wb_i(3b_j\alpha_{ik} + b_k - b_j)\right], \end{aligned}$$

where

$$S_j'(V_i) = (x_j - x_i)S_x(V_i) + (y_j - y_i)S_y(V_i),$$

$$w = \frac{b_1 b_2 b_3}{b_1 b_2 + b_1 b_3 + b_2 b_3},$$

$$\alpha_{ij} = \frac{\|e_{jk}\|^2 + \|e_{ik}\|^2 - \|e_{ij}\|^2}{2\|e_{ik}\|^2}.$$

If S_{ijk} is used to represent the same discrete interpolant for the triangle T_{ijk}, then the final interpolant can be represented as

$$S(x, y) = S_{ijk}(x, y) \quad \text{for } (x, y) \in T_{ijk}.$$

Concerning the degree of algebraic precision of S, it can be shown that S_T will reproduce quadratics but the curve network is limited to linear precision and so the final interpolation operator has precision of degree one.

4. Algorithms and Examples. The first step in applying the approximation S requires a triangulation of D. In those cases where D is the convex hull of V_i, $i = 1, \ldots, N$, we have incorporated an algorithm described by Lawson [7] which selects a particular triangulation on the basis of the max-min angle criterion. The program that implements this algorithm produces information describing the boundary along with three arrays n_1, n_2 and n_3, each of length N_t, which form a list of the vertices of each triangle of the triangulation. An example of a triangulation produced by this program is given in Table 1 and Figure 1.

TABLE 1

Data

i	x_i	y_i
1	.21	.88
2	.46	.93
3	.83	.89
4	.97	.54
5	.67	.71
6	.53	.74
7	.28	.77
8	.07	.70
9	.06	.43
10	.25	.56
11	.48	.61
12	.67	.54
13	.77	.45
14	.90	.31
15	.66	.35
16	.50	.47
17	.32	.44
18	.25	.31
19	.46	.33
20	.57	.20
21	.75	.25
22	.94	.05
23	.46	.07
24	.18	.19
25	.14	.06

Boundary

1, yes 0, no
1
1
1
1
0
0
0
1
1
0
0
0
0
0
0
0
0
0
0
0
0
1
0
0
1

Triangulation

i	$n_1(i)$	$n_2(i)$	$n_3(i)$
1	7	1	8
2	1	7	2
3	7	8	10
4	7	10	11
5	2	7	6
6	6	7	11
7	10	8	9
8	11	10	17
9	17	10	9
10	11	17	16
11	11	16	12
12	6	11	5
13	2	6	5
14	5	11	12
15	17	9	18
16	17	19	16
17	19	17	18
18	16	19	15
19	12	4	5
20	2	5	3
21	12	16	15
22	12	15	13
23	18	9	24
24	18	23	19
25	23	18	24
26	4	12	13
27	15	19	20
28	15	20	21
29	13	15	21
30	24	9	25
31	19	23	20
32	23	24	25
33	13	21	14
34	3	5	4
35	20	22	21
36	22	20	23
37	4	13	14
38	23	25	22
39	14	21	22
40	4	14	22

The next step of our method requires the solution of (2.4). The coefficient matrix of this linear system is in general quite sparse, but the structure is sufficiently complicated to eliminate the use of a direct method which takes advantage of this sparseness. We have found that an iterative method based upon the following equivalent form of (2.4) works quite well:

$$\begin{pmatrix} Sx_i \\ Sy_i \end{pmatrix} = -\begin{pmatrix} \alpha_i & \beta_i \\ \beta_i & \gamma_i \end{pmatrix}^{-1} \begin{pmatrix} \sum\limits_{ij\in N_i} \alpha_{ij} Sx_j + \sum\limits_{ij\in N_i} \beta_{ij} Sy_j - Zx_i \\ \sum\limits_{ij\in N_i} \beta_{ij} Sx_j + \sum\limits_{ij\in N_i} \gamma_{ij} Sy_j - Zy_i \end{pmatrix}, \qquad i = 1, \ldots, N,$$

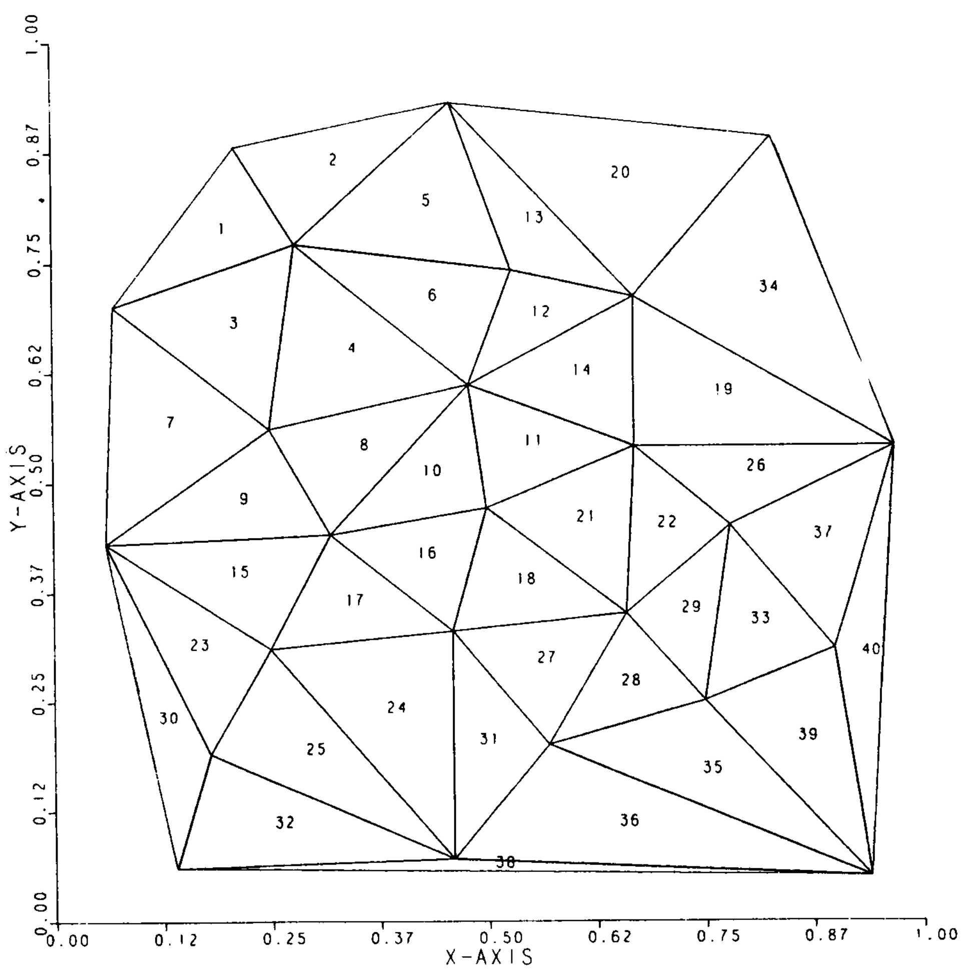

FIGURE 1

where $Sx_j = S_x(V_j)$, $Sy_j = S_y(V_j)$

$$\alpha_{ij} = \frac{(x_j - x_i)^2}{2\|e_{ij}\|^3}, \qquad \alpha_i = 2 \sum_{ij \in N_i} \alpha_{ij},$$

$$\beta_{ij} = \frac{(x_j - x_i)(y_j - y_i)}{2\|e_{ij}\|^3}, \qquad \beta_i = 2 \sum_{ij \in N_i} \beta_{ij},$$

$$\gamma_{ij} = \frac{(y_j - y_i)^2}{2\|e_{ij}\|^3}, \qquad \gamma_i = 2 \sum_{ij \in N_i} \gamma_{ij},$$

$$Zx_i = \frac{3}{2} \sum_{ij \in N_i} \frac{(z_j - z_i)(x_j - x_i)}{\|e_{ij}\|^3},$$

and

$$Zy_i = \frac{3}{2} \sum_{ij \in N_i} \frac{(z_j - z_i)(y_j - y_i)}{\|e_{ij}\|^3}.$$

For an initial approximation, we first obtain the first order partial derivatives of each plane which interpolates over each triangle. Then for each vertex, we compute the average of these derivatives for each triangle that involves this vertex. This amounts to the following computation:

$$Sx_i^0 = \frac{1}{|M_i|} \sum_{ijk \in M_i} \frac{f_{ijk}}{A_{ijk}}, \qquad Sy_i^0 = \frac{1}{|M_i|} \sum_{ijk \in M_i} \frac{g_{ijk}}{A_{ijk}},$$

where $M_i = \{abc\colon T_{abc}$ is a triangle with vertex $V_i\}$,

$|M_i|$ is the number of elements of M_i,

$f_{ijk} = (y_j - y_k)z_i + (y_k - y_i)z_j + (y_i - y_j)z_k,$

$g_{ijk} = (x_k - x_j)z_i + (x_i - x_k)z_j + (x_j - x_i)z_i,$ and

$A_{ijk} = (x_i - x_j)(y_i - y_k) - (y_i - y_j)(x_i - x_k).$

The first pass of the following algorithm computes the initial values Sx_i^0, Sy_i^0, $i = 1,\dots,N$, as well as α_i, β_i, γ_i, Zx_i, Zy_i, $i = 1,\dots,N$, which remain constant throughout the iteration process.

```
For $l = 1,\dots,N_t$, do:
    For $(i, j, k) \in I$, do:
        $a = n_i(l), b = n_j(l), c = n_k(l)$
        $|M_a| = |M_a| + 1$
        $Sx_a^0 = Sx_a^0 + f_{abc}/A_{abc}$
        $Sy_a^0 = Sy_a^0 + g_{abc}/A_{abc}$
        $\alpha_a = \alpha_1 + \tilde{\alpha}_{ab} + \tilde{\alpha}_{ac}$
        $\beta_a = \beta_a + \tilde{\beta}_{ab} + \tilde{\beta}_{ac}$
        $\gamma_a = \gamma_a + \tilde{\gamma}_{ab} + \tilde{\gamma}_{ac}$
        $Zx_a = Zx_a + \frac{3}{2}\left[\frac{\tau_{ab}(x_b - x_a)(z_b - z_a)}{\|e_{ab}\|^3} + \frac{\tau_{ac}(x_c - x_a)(z_c - z_a)}{\|e_{ac}\|^3}\right]$
        $Zy_a = Zy_a + \frac{3}{2}\left[\frac{\tau_{ab}(y_b - y_a)(z_b - z_a)}{\|e_{ab}\|^3} + \frac{\tau_{ac}(y_c - y_a)(z_c - z_a)}{\|e_{ac}\|^3}\right]$
For $i = 1,\dots,N$, do:
    $Sx_i^0 = \frac{Sx_i^0}{|M_i|}$
    $Sy_i^0 = \frac{Sy_i^0}{|M_i|}$
    $\delta = \alpha_i\gamma_i - \beta_i^2$
    $\alpha_i^{-1} = \frac{\gamma_i}{\delta}$
    $\beta_i^{-1} = \frac{-\beta_i}{\delta}$
    $\gamma_i^{-1} = \frac{\alpha_i}{\delta}$
```

We have used the notation

$$\tilde{\alpha}_{ab} = \tau_{ab}\alpha_{ab}, \quad \tilde{\beta}_{ab} = \tau_{ab}\beta_{ab}, \quad \tilde{\gamma}_{ab} = \tau_{ab}\gamma_{ab},$$

where

$$\tau_{ab} = \begin{cases} 1 & \text{if } V_a \text{ and } V_b \text{ are on the boundary,} \\ 1/2 & \text{otherwise.} \end{cases}$$

The factor of τ_{ab} is necessary because we go sequentially through the list of triangles causing each interior edge to be processed twice.

The iterative part of the algorithm can be described as follows:

```
For $i = 1,\ldots,N$, do:
    $\delta_i = Zx_i$, $\varepsilon_i = Zy_i$
For $n = 0, 1, \ldots$, until satisfied, do:
    For $l = 1,\ldots,N_t$, do:
        For $(i, j, k) \in I$, do:
            $a = n_i(l), b = n_j(l), c = n_k(l)$
            $\delta_a = \delta_a - \frac{1}{4}[\tilde{\alpha}_{ab}Sx_b^n + \tilde{\alpha}_{ac}Sx_c^n + \tilde{\beta}_{ab}Sy_b^n + \tilde{\beta}_{ac}Sy_c^n]$
            $\varepsilon_a = \varepsilon_a - \frac{1}{4}[\tilde{\beta}_{ab}Sx_b^n + \tilde{\beta}_{ac}Sx_c^n + \tilde{\gamma}_{ab}Sy_b^n + \tilde{\gamma}_{ac}Sy_c^n]$
    For $i = 1,\ldots,N$, do:
        $Sx_i^{n+1} = \alpha_i^{-1}\delta_i + \beta_i^{-1}\varepsilon_i$
        $Sy_i^{n+1} = \beta_i^{-1}\delta_i + \gamma_i^{-1}\varepsilon_i$
        $\delta_i = Zx_i$, $\varepsilon_i = Zy_i$
```

In Figure 2, we show an example of the results of the above algorithm. The values z_i, $i = 1,\ldots,N$, are obtained from the function $.25\,\mathrm{EXP}(-16((x - .5)^2 + (y - .5)^2))$ and the triangulation is that of Figure 1. In practice, we have found that, on the average, about a dozen iterations will yield five or six digits of accuracy. Although they are rare, we have encountered cases that take as many as eighteen and as few as one to obtain this same accuracy.

The final step requires the evaluation of S given by (3.1) on the proper triangle. In order to obtain a perspective plot of the surface, we evaluate the approximation on a uniform rectangular grid. These values are stored in rectangular array S with

$$S(i, j) = S(\bar{x}_i, \bar{y}_j), \qquad i = 1,\ldots,NR, j = 1,\ldots,NC,$$

where

$$\bar{x}_i = XL + (i - 1)\,DX, \qquad \bar{y}_j = YL + (j - 1)\,DY,$$

$$DX = \frac{XH - XL}{NR - 1}, \quad \text{and} \quad DY = \frac{YH - YL}{NC - 1}.$$

We take the approximation to be zero for those points which lie in the display rectangle $[XL, XH] \times [YL, YH]$ but outside D. Rather than stepping through the values of $(\bar{x}_i, \bar{y}_j)$ and asking which triangle these points lie in, our algorithm goes through the list of triangles and computes the values of $S(i, j)$ which are defined by a given triangle. More precisely,

```
For $l = 1,\dots,N_t$, do:
    $a = n_1(l), b = n_2(l), c = n_3(l)$
    $IL = (\min(x_a, x_b, x_c) - XL)/DX$
    $JL = (\min(y_a, y_b, y_c) - YL)/DY$
    $IH = (\max(x_a, x_b, x_c) - XL)/DX$
    $JH = (\max(y_a, y_b, y_c) - YL)/DY$
        For $i = IL, IL + 1,\dots,IH$, do:
            For $j = JL, JL + 1,\dots,JH$, do:
                $x = XL + (i - 1)\,DX$
                $y = YL + (j - 1)\,DY$
                Solve $x = b_1x_a + b_2x_b + b_3x_c$
                      $y = b_2y_a + b_3y_b + b_3y_c$
                      $1 = b_1 + b_2 + b_3$
                to obtain $b_1, b_2, b_3$
                If $b_i \geqslant 0, i = 1,2,3$ then $S(i, j) = S_{abc}(x, y)$
```

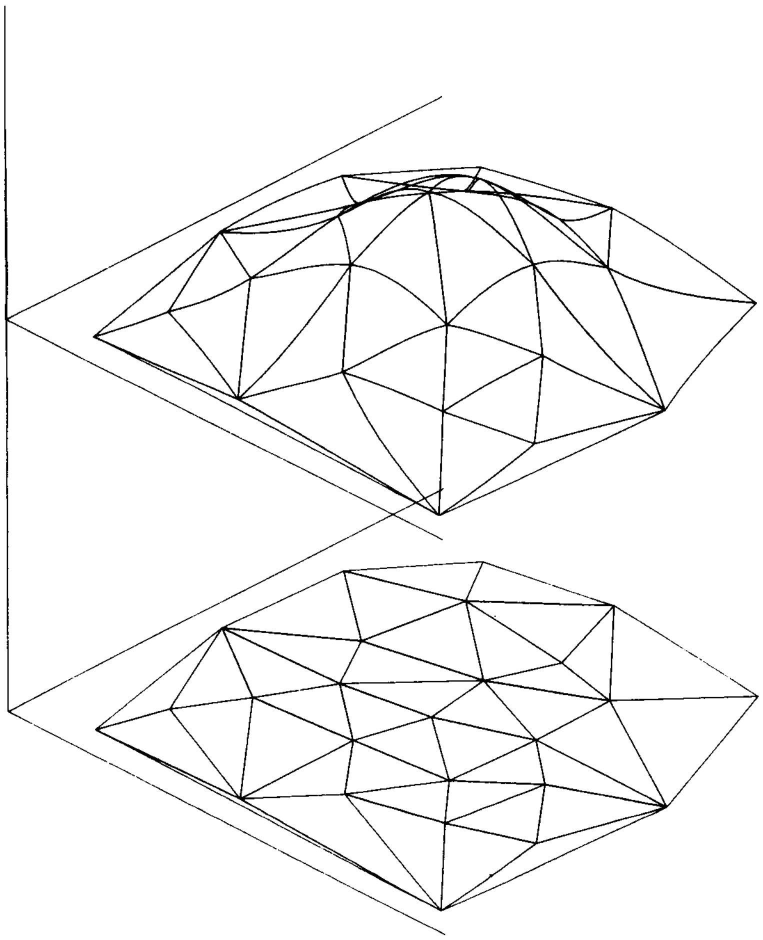

FIGURE 2

In Figure 3a, we show the final interpolant based upon the curve network of Figure 2. For comparison, in Figure 3b we include the interpolant which uses the initial derivatives (Sx_i^0, Sy_i^0), $i = 1,\dots,N$.

Our next example is comparable to one discussed by Lawson (cf. [7, Figure 7, p. 175]). The values z_i, $i = 1,\dots,n$, are obtained from the function

$$\mathrm{EXP}\left(-8\left[(x-.5)^2+(y-.5)^2\right]\right).$$

The 26 data points and a contour plot of the interpolant are shown in Figure 4a. The contours are at the values $S(x, y) = .2, .4, .6$ and $.8$. A plot of the interpolant using the initial values for the derivatives is shown in Figure 4b.

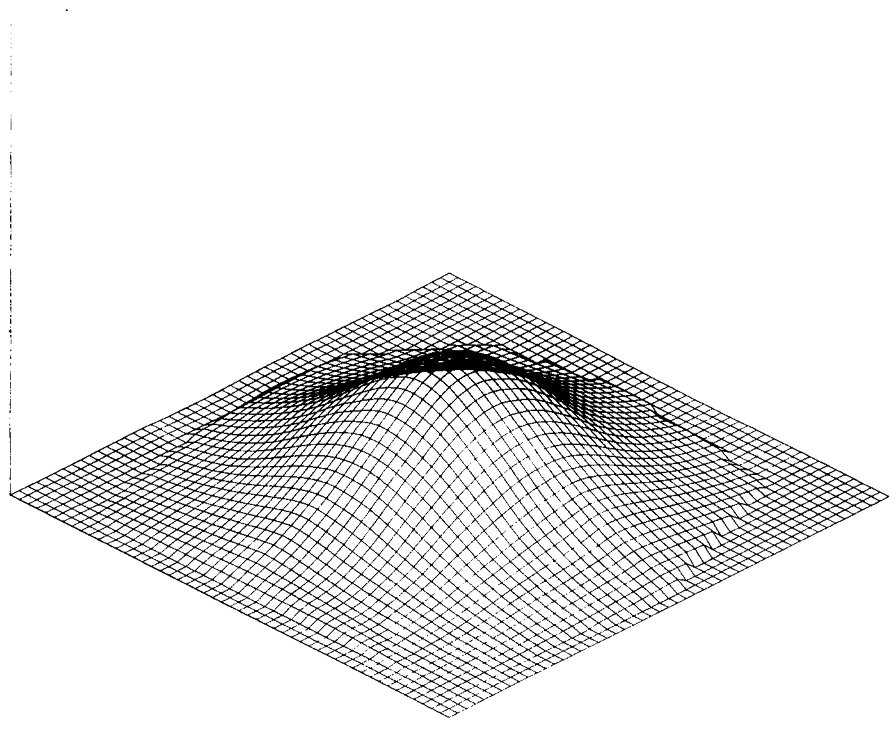

FIGURE 3a

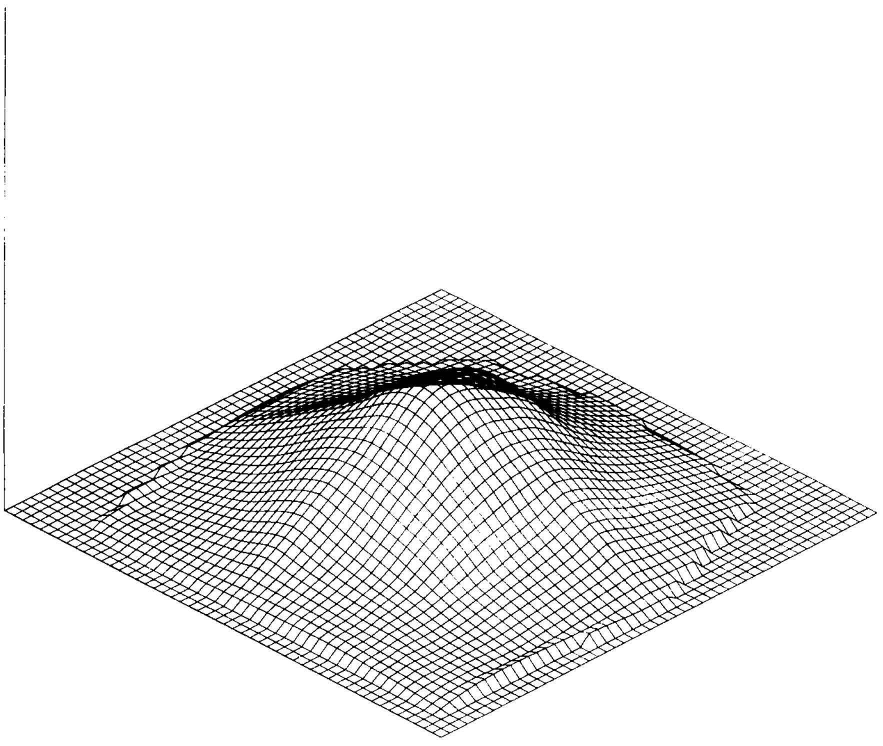

FIGURE 3b

Our third example involves a nonconvex, multiply-connected domain and is included mainly to point out the possibility of using such domains with the present method. Further discussion on the problem of triangulating this type of domain and an algorithm can be found in [8]. In Figure 5a we show the triangulation and the final interpolant. The values z_i, $i = 1, \ldots, N$, are obtained from the same function as used in the previous example. Figure 5b contains a plot of the interpolant using the same data but over a triangulation of the convex hull.

The last example is based upon data provided by the United States Geological Survey [6]. The ordinates represent elevations of a mostly subterranean formation of granite called Hawk Rock which is located in the southeastern desert of Arizona. We found this example particularly interesting because of the unique configuration of the data. Due to the techniques of collection, this data consists of subsets which lie on certain line segments. Figures 6a and 6b show these lines of data along with the triangulation of it. Two views of the interpolant are shown in Figures 6c and 6d.

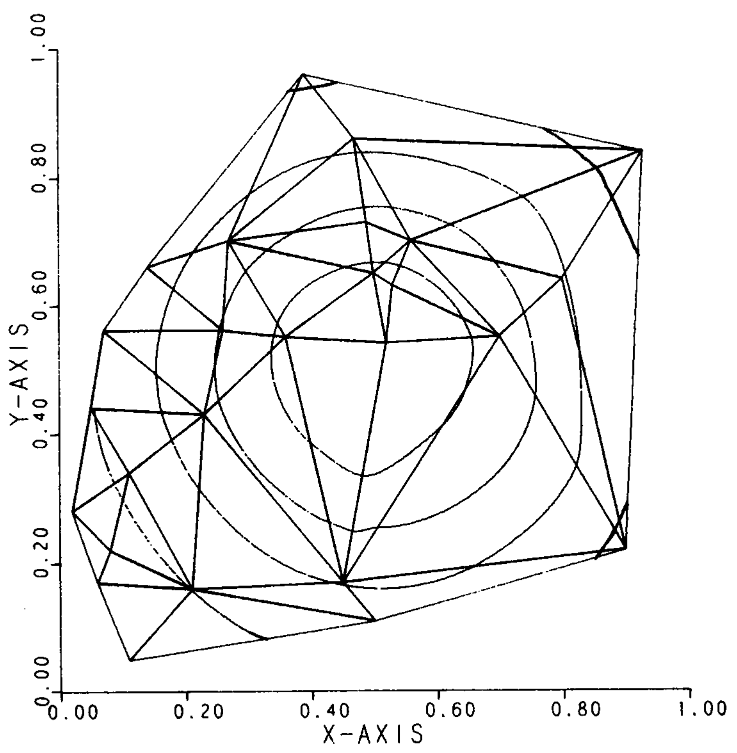

FIGURE 4a

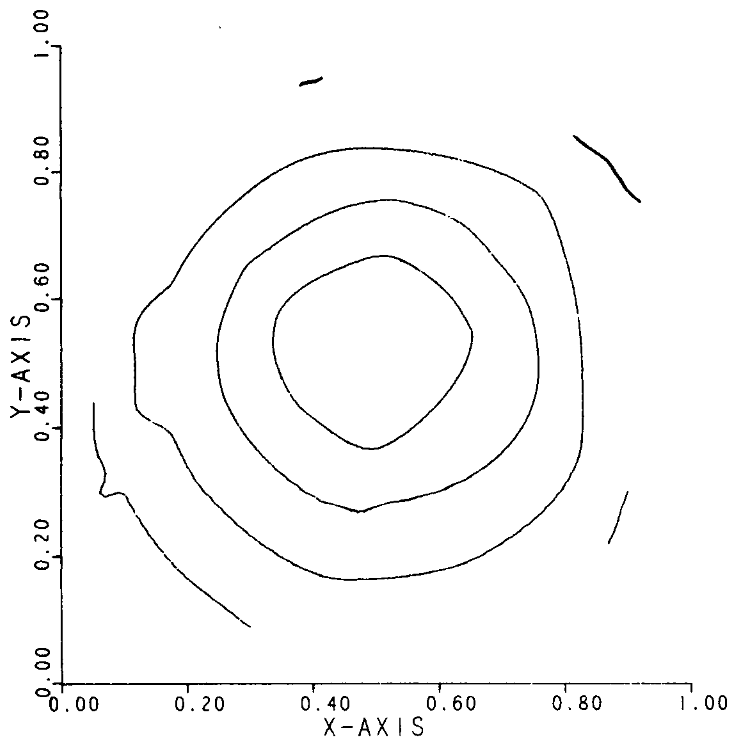

FIGURE 4b

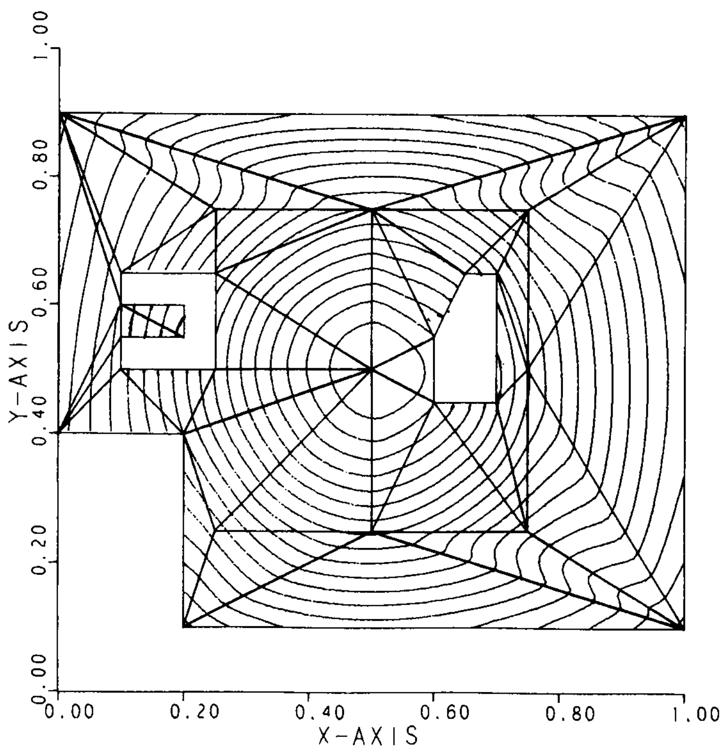

FIGURE 5a

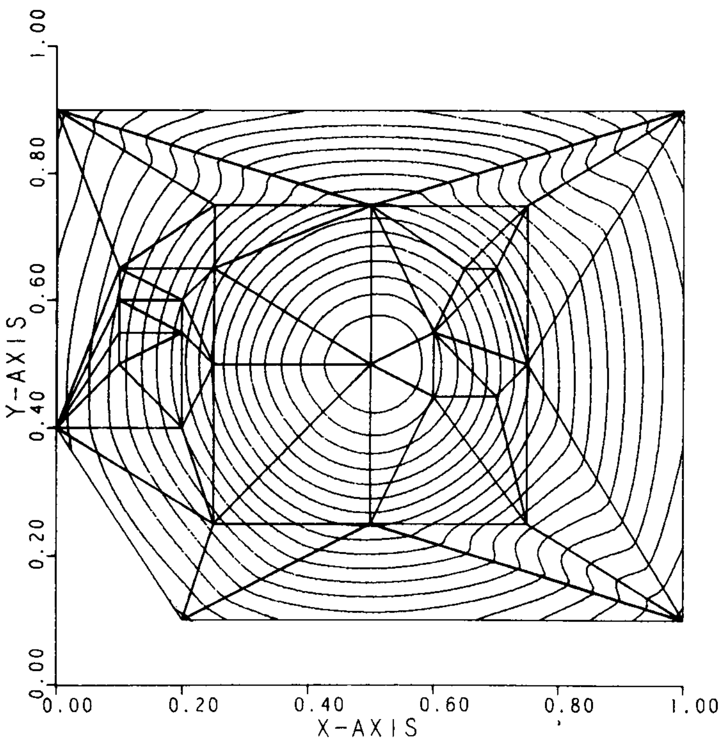

FIGURE 5b

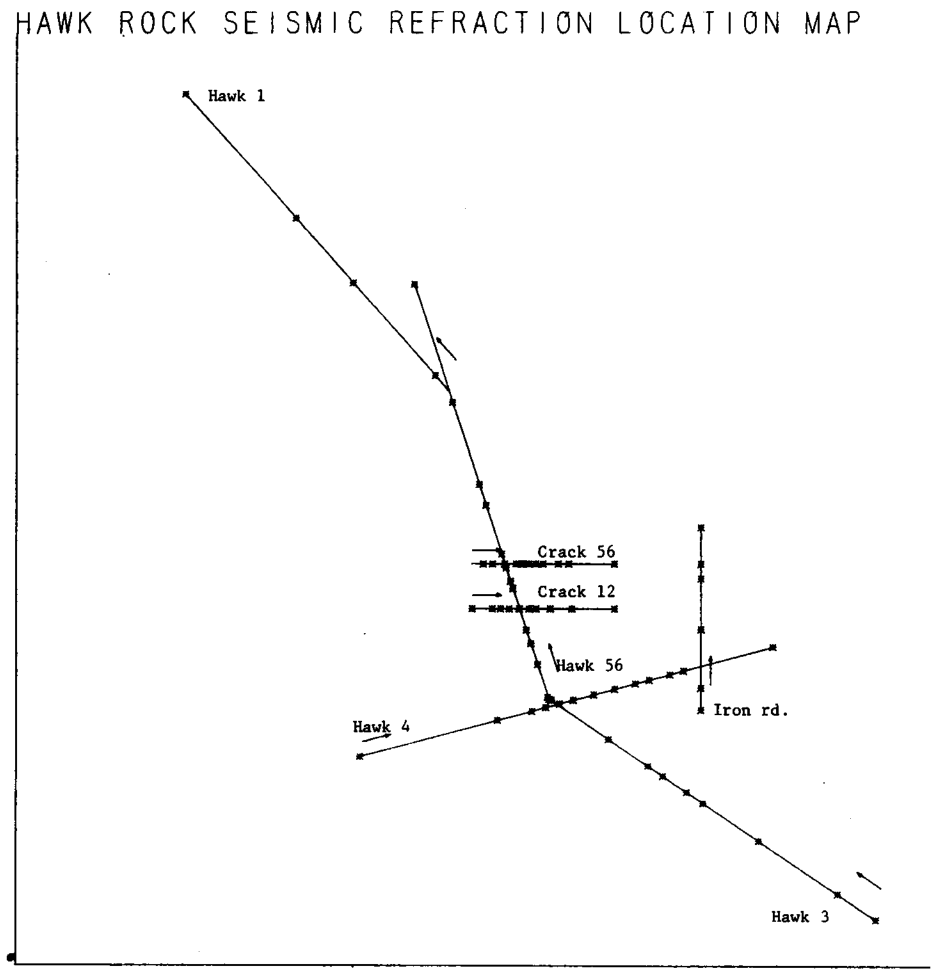
HAWK ROCK SEISMIC REFRACTION LOCATION MAP
Hawk 1
Crack 56
Crack 12
Hawk 56
Iron rd.
Hawk 4
Hawk 3

FIGURE 6a

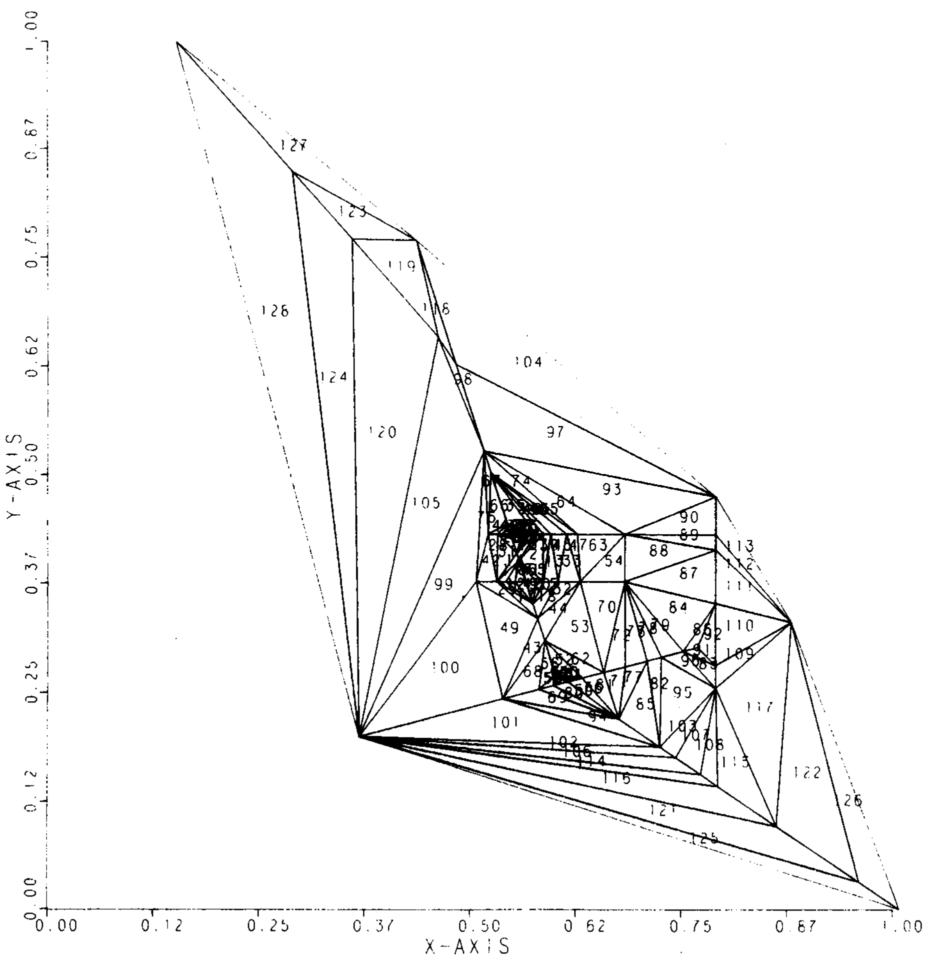

FIGURE 6b

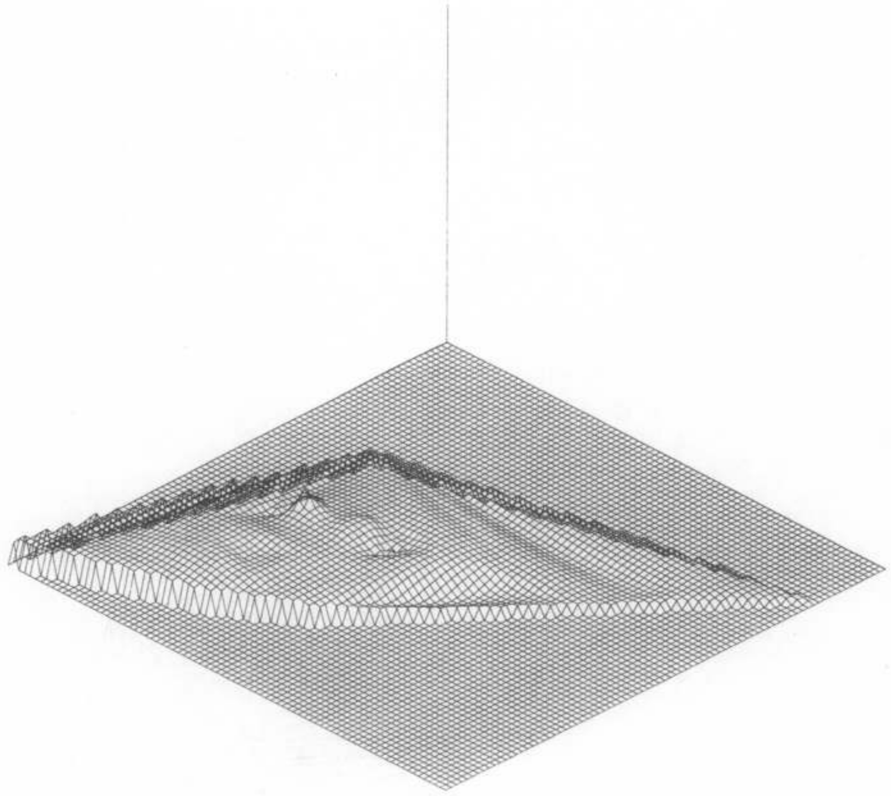
FIGURE 6c

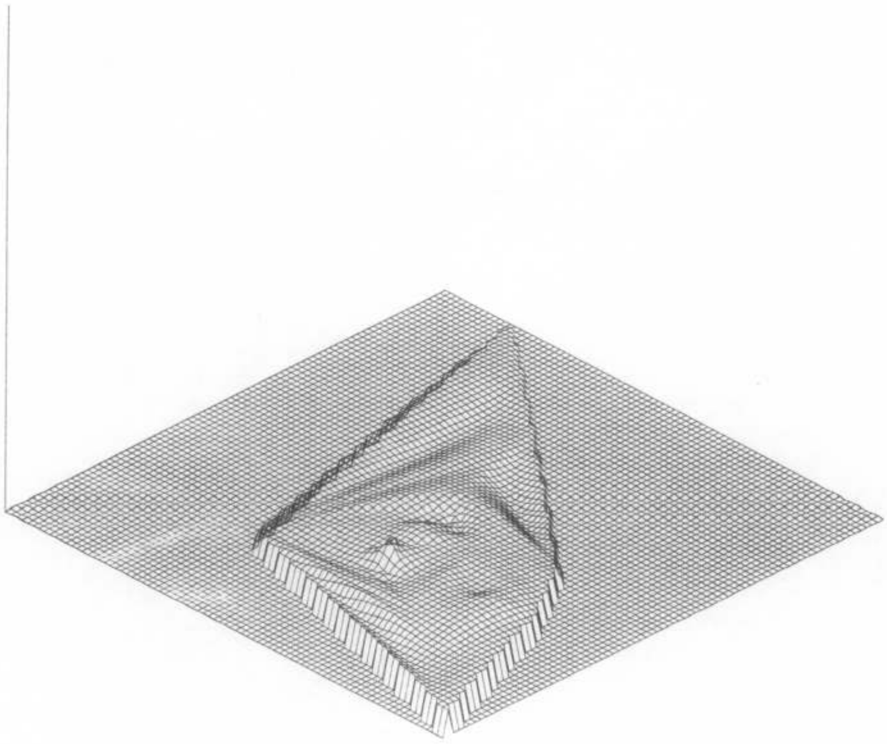
FIGURE 6d

Recently, Franke [5] has reported on the results of a project devoted to the comparison of some 29 methods for interpolating scattered data. Included in this report are the results of our implementation of the present method which is based upon the max-min angle optimization of the triangulation of the convex hull, the algorithm of this section for computing the minimum norm network and the algorithm of this section for evaluating the interpolant on a rectangular grid. In addition to certain assessments of the fitting characteristics of each of the methods, Franke's report includes storage requirements and timing results. The storage requirements tabulated by Franke are given in terms of additional storage required beyond that needed for the data (x_i, y_i, z_i), $i = 1, \ldots, N$, and the output array of evaluations. For the implementation of the present method, this amounts to approximately 32N. Franke's timing results are based on the use of an IBM 360/67. We have run all of our examples on a UNIVAC 1100/42. In Table 2 we give some approximations of the running time (in seconds) required for the three steps of our method. All programs mentioned have been written in FORTRAN.

TABLE 2

		Minimum	Evaluation	
		Norm	40×40	80×80
N	Triangulation	Network	Grid	Grid
25	.10	.88	.35	1.16
50	.35	2.14	.85	2.88
100	1.24	4.18	1.64	5.56
200	4.65	7.41	3.21	11.32

Department of Mathematics
Arizona State University
Tempe, Arizona 85281

1. Hiroshi Akima, "A method of bivariate interpolation and smooth surface fitting for values given at irregularly distributed points," *Trans. Math. Software*, v. 4, 1978, pp. 148–159.

2. R. E. Barnhill, "Representation and approximation of surfaces," *Mathematical Software* III (J. R. Rice, ed.), Academic Press, New York, 1977, pp. 68–119.

3. R. E. Barnhill, G. Birkhoff & W. J. Gordon, "Smooth interpolation in triangles," *J. Approx. Theory*, v. 8, 1973, pp. 114–128.

4. C. de Boor & R. E. Lynch, "On splines and their minimum properties," *J. Math. Mech.*, v. 15, 1966, pp. 953–969.

5. Richard Franke, *A Critical Comparison of Some Methods for Interpolation of Scattered Data*, Naval Postgraduate School Technical Report NPS-53-79-003, December 1979.

6. R. L. Laney, L. W. Pankratz & C. Zenone, Private Communication, Phoenix, Arizona, 1978.

7. C. L. Lawson, "Software for C^1 surface interpolation," *Mathematical Software* III (J. R. Rice, ed.), Academic Press, New York, 1977, pp. 161–194.

8. B. A. Lewis & J. S. Robinson, "Triangulation of planar regions with applications," *Comput. J.*, v. 21, 1978, pp. 324–332.

9. G. M. Nielson, "Minimum norm interpolation in triangles," *SIAM J. Numer. Anal.*, v. 17, 1980, pp. 44–62.