A Method of Bivariate Interpolation and Smooth Surface Fitting for Irregularly Distributed Data Points

HIROSHI AKIMA
Institute for Telecommunication Sciences

A method of bivariate interpolation and smooth surface fitting is developed for z values given at points irregularly distributed in the x-y plane. The interpolating function is a fifth-degree polynomial in x and y defined in each triangular cell which has projections of three data points in the x-y plane as its vertexes. Each polynomial is determined by the given values of z and estimated values of partial derivatives at the vertexes of the triangle. Procedures for dividing the x-y plane into a number of triangles, for estimating partial derivatives at each data point, and for determining the polynomial in each triangle are described A simple example of the application of the proposed method is shown.

Key Words and Phrases bivariate interpolation, interpolation, partial derivative, polynomial, smooth surface fitting
CR Categories: 5.13
The Algorithm Bivariate Interpolation and Smooth Surface Fitting for Irregularly Distributed Data Points *ACM Trans. Math. Software* 2, 1(June 1978), 160–164.

1. INTRODUCTION

In a previous study [2, 3] the author developed a method of bivariate interpolation and smooth surface fitting. The method was designed in such a way that the resulting surface would pass through all the given data points. Adopting local procedures, it successfully suppressed undulations in the resulting surface which are very likely to appear in surfaces fitted by other methods. Like many other methods, however, this method also has a serious drawback. Applicability is restricted to cases where the values of the function are given at rectangular grid points in a plane; i.e. if a two-dimensional Cartesian coordinate system with x and y axes is assumed in the plane, the values of $z = z(x, y)$ must be given as $z_{ij} = z(x_i, y_j)$ in the x-y plane, where $i = 1, 2, \ldots, n_x$ and $j = 1, 2, \ldots, n_y$. This restriction prevents application to cases where collection of data at rectangular grid points is impossible or otherwise impractical. The subject of the present study is bivariate interpolation and smooth surface fitting in the general case where the values of the function are given at irregularly distributed points in a plane; i.e. the case where the z values are given as $z_i = z(x_i, y_i)$, where $i = 1, 2, \ldots, n$.

Two types of approaches are possible: one using a single global function, and the other based on a collection of local functions. In the former approach, the

Author's address: U.S. Department of Commerce, NTIA, Institute for Telecommunication Sciences, Boulder, CO 80303.

procedure often becomes too complicated to manage as the number of given data points increases. Moreover, the surface resulting from the former sometimes exhibits excessive undulations. For these reasons, only the latter approach is considered in the present study.

Shepard [9] suggested a method based on weighted averages of the given z values. The basic weighting function is the square of the reciprocal of the distance between the projection of each data point and that of the point at which interpolation is to be performed. The actual weighting function is an improvement of this basic weighting function in that the actual function corresponding to a distant data point vanishes. Through this improvement the originally global procedures in this method became local. This method has several desirable properties. It takes into account the "shadowing" of the influence of a data point by a nearer one in the same direction. It yields reasonable slopes at the given data points. However, it fails to produce a plane when all the given data points lie in a slanted plane; this property is considered to be a serious drawback.

Bengtsson and Nordbeck [6], Heap [7], and Lawson [8] suggested methods based on partitioning the x-y plane into a number of triangles (each triangle having projections of three data points in the x-y plane as its vertexes) and on fitting a plane to the surface in each triangle. These methods are useful for some applications. Obviously, however, the resulting surface is not smooth on the sides of the triangles although it is continuous.

Whitten and Koelling [10] attempted to fit a curved surface to each triangle in such a way that the surface in the triangle would smoothly connect to other surfaces fitted to neighboring triangles. Their function for each triangle has 15 coefficients to be determined by 15 conditions that assure smooth connection with the neighboring triangles. As shown in their example, however, these 15 conditions sometimes cannot be satisfied. Moreover, even when the 15 conditions are satisfied and the 15 coefficients are satisfactorily determined, the function is not smooth along the bisector of the triangle due to the use of a "discontinuous function."

In conjunction with variational problems containing second-order derivatives Zlamal [12] discussed an approximation procedure using fifth-degree polynomials in x and y over triangular regions in the x-y plane. To determine the coefficients of the polynomial for each triangle, he uses, in addition to the z values and the first and second partial derivatives (i.e. z_x, z_y, z_{xx}, z_{xy}, and z_{yy}) at the three vertexes of the triangle, three normal derivatives at the midpoints of the sides of the triangle. (A normal derivative is a derivative differentiated in the direction normal to the side.) The theory was generalized to $(4m + 1)$th-degree polynomials for functions m-times continuously differentiable on a closed triangular domain by Zenisek [11]. Although a comprehensive interpolation method is not suggested in their papers, their papers were instrumental in stimulating portions of the ideas developed in the present study.

Recently the author [4] proposed a method of bivariate interpolation and smooth surface fitting that is applicable to z values given at irregularly distributed points in the x-y plane. The interpolating function is a smooth function; i.e. the interpolating function and its first-order partial derivatives are continuous. The proposed method is based on local procedures. The surface resulting from the proposed method will pass through all the given data points.

The method proposed in the aforementioned report [4] has been improved further: a better criterion for triangulation (partitioning into a number of triangles) of the plane suggested by Lawson [8] has been adopted; and the algorithm that implements the method has been improved substantially, both in the required storage area and in the computation time. In this paper, the proposed method is outlined in Section 2, some programming problems are discussed in Section 3, a simple example that illustrates the application of the proposed method is shown in Section 4, and some pertinent remarks are set forth in Section 5. An improved Fortran subprogram package that implements the proposed method is presented in an algorithm developed by the author, ACM Algorithm 526 [5].

2. OUTLINE OF THE PROPOSED METHOD

In the proposed method the x-y plane is triangulated or divided into a number of triangular cells, each having projections of three data points in the plane as its vertexes, and a bivariate fifth-degree polynomial in x and y is applied to each triangular cell. Estimated values of partial derivatives at each data point are used in determining the polynomial.

For triangulation of the x-y plane, we adopt the max-min angle criterion suggested by Lawson [8]. This criterion dictates that, when a set of four points are vertexes of a quadrilateral with each internal angle smaller than π, one chooses, out of two possible ways of partitioning the quadrilateral into a pair of triangles, the partitioning that maximizes the minimum interior angle of the two triangles produced. In triangulating the x-y plane, we first connect the closest pair of points. We next add a point at a time in ascending order of the distance from the midpoint of the closest pair of points. This ordering in adding new points assures that a new point to be added always lies outside the polygon constructed with the old points; the new point lies outside the circle that is centered at the midpoint of the closest pair of points and passes through the most lately added old point while the polygon lies inside the circle. Each time a new point is added we construct triangles by connecting the new point with the old points that are visible from the new point and, whenever necessary, "exchange" triangles.

Interpolation of z values in a triangle is based on the following three assumptions:

(i) The value of the function at point (x, y) in a triangle is interpolated by a bivariate fifth-degree polynomial in x and y; i.e.

$$z(x, y) = \sum_{j=0}^{5} \sum_{k=0}^{5-j} q_{jk} x^j y^k.$$

Note that there are 21 coefficients to be determined.

(ii) The values of the function and its first-order and second-order partial derivatives (i.e. z, z_x, z_y, z_{xx}, z_{xy}, and z_{yy}) are given at each vertex of the triangle. This assumption yields 18 independent conditions.

(iii) The partial derivative of the function differentiated in the direction perpendicular (or normal) to each side of the triangle is a polynomial of degree three, at most, in the variable measured in the direction of the side of the triangle. Since a triangle has three sides, this assumption yields three additional conditions.

The purpose of the third assumption is twofold. This assumption adds three independent conditions to the 18 conditions dictated by the second assumption and thus enables one to determine the 21 coefficients of the polynomial. (A detailed description of step-by-step procedures for determining the coefficients is found elsewhere [4].) It also assures smoothness of interpolated values as described in the following paragraph.

Smoothness of the interpolated values and therefore smoothness of the resulting surface along the side of the triangle can be proved as follows. It is convenient to introduce another Cartesian coordinate system, which we call the s-t system, in such a way that the s axis is parallel to a side of the triangle. Since the coordinate transformation between the x-y system and the s-t system is linear, the values of z_x, z_y, z_{xx}, z_{xy}, and z_{yy} at each vertex uniquely determine the values of z_s, z_t, z_{ss}, z_{st}, and z_{tt} at the same vertex, each of the latter as a linear combination of the former. Then the z, z_s, and z_{ss} values at two vertexes uniquely determine a fifth-degree polynomial in s for z on the side between these vetexes. Since two fifth-degree polynomials in x and y representing z values in two triangles that share the common side are reduced to fifth-degree polynomials in s on the side, these two polynomials in x and y coincide with each other on the common side. This proves continuity of the interpolated z values along a side of a triangle. Similarly, the values of z_t and $z_{st} = (z_t)_s$ at two vertexes uniquely determine a third-degree polynomial in s for z_t on the side. Since the polynomial representing z_t is assumed to be third degree at most with respect to s, two polynomials representing z_t in two triangles that share the common side also coincide with each other on the side. This proves continuity of z_t and thus smoothness of z along the side of the triangle.

Although the proposed method is intended for interpolation in the polygon in the x-y plane formed by the projections of given data points, capability of extrapolation outside the polygon is desirable. A high degree of accuracy in extrapolation is not expected, but it is desirable if the extrapolated values are smooth, connect smoothly to the interpolated values inside the polygon, and do not diverge at least in the immediate neighborhood of the polygon. In a semi-infinite rectangular area on the border line segment, extrapolation of the desirable nature can be done with a bivariate polynomial that is of the fifth degree in the variable measured in the direction of the line segment and of the second degree in the distance from the line segment. In a semi-infinite triangular area between the two semi-infinite rectangles, extrapolation can be done with a bivariate second-degree polynomial in x and y that smoothly connects to the two polynomials in the two neighboring rectangles.

Procedures for estimating the five partial derivatives locally at each data point are not unique. The derivatives could be determined as partial derivatives of a second-degree polynomial in x and y that coincides with the given z values at six data points consisting of five data points, the projections of which are closest to the projection of the data point in question and the data point itself. This procedure is a bivariate extension of the one used in the univariate osculatory interpolation [1]. Adoption of this procedure has an advantage that, when z is a second-degree polynomial in x and y, the method yields exact results. As will be shown in Section 4, however, this procedure sometimes yields very unreasonable results.

We take a different approach and estimate the partial derivatives in two steps, i.e. the first-order derivatives in the first step and the second-order derivatives in the second step. To estimate the first-order partial derivatives at data point P_0 we use several additional data points $P_i(i = 1, 2, \ldots, n_c)$ the projections of which are closest to the projections of P_0 selected from all data points other than P_0. We take two data points P_i and P_j out of the n_c points and construct the vector product of $\overline{P_0P_i}$ and $\overline{P_0P_j}$, i.e. a vector that is perpendicular to both $\overline{P_0P_i}$ and $\overline{P_0P_j}$ with the right-hand rule and has a magnitude equal to the area of the parallelogram formed by $\overline{P_0P_i}$ and $\overline{P_0P_j}$. We take P_i and P_j in such a way that the resulting vector product always points upward (i.e. the z component of the vector product is always positive). We construct vector products for all possible combinations of $\overline{P_0P_i}$ and $\overline{P_0P_j}(i \neq j)$ and take a vector sum of all the vector products thus constructed. Then we assume that the first-order partial derivatives z_x and z_y at P_0 are estimated as those of a plane that is normal to the resultant vector sum thus composed. Note that, when $n_c = 2$, the estimated z_x and z_y are equal to the partial derivatives of a plane that passes through P_0, P_1, and P_2. Also note that, when $n_c = 3$ and the projection of P_0 in the x-y plane lies inside the triangle formed by the projections of P_1, P_2, and P_3, the estimated values of z_x and z_y are equal to the partial derivatives of a plane that passes through P_1, P_2, and P_3.

In the second step we apply the procedure of partial differentiation described in the preceding paragraph to the estimated z_x values at $P_i(i = 0, 1, 2, \ldots, n_c)$ and obtain estimates of $z_{xx} = (z_x)_x$ and $z_{xy} = (z_x)_y$ at P_0. We repeat the same procedure with the estimated z_y values and obtain estimates of $z_{xy} = (z_y)_x$ and $z_{yy} = (z_y)_y$. We adopt a simple arithmetic mean of two z_{xy} values thus estimated as our estimate for z_{xy} at P_0.

The selection of n_c is again not unique. Obviously, n_c cannot be less than 2. Also, it must be less than the total number of data points. Other than those requirements, there seems to exist no theory that dictates a definite value for n_c. The best we can say is that, based on the example shown in Section 4 and on some others, we recommend a number between 3 and 5 (inclusive) for n_c.

When the projections of data point P_0 and all selected n_c data points $P_i(i = 1, 2, \ldots, n_c)$ are collinear in the x-y plane, the procedure of estimating the partial derivatives described above does not work. Should this happen, we will replace a selected data point that is the farthest from P_0 among the n_c selected data points by a data point that is the next closest to P_0 and is not colinear with P_0 and other selected data points in the x-y plane.

3. PROGRAMMING CONSIDERATIONS

In this section we describe some programming problems that we have considered in implementing the proposed method presented in Algorithm 526 [5] that accompanies this paper.

Regarding configurations of the points at which bivariate interpolation is to be performed, there are two typical cases. A user may wish to perform bivariate interpolation at a set of points of his choice. Or, he may wish to fit a smooth surface by interpolating the values at rectangular grid points. It is considered desirable to program for each case a separate subroutine that interfaces with the user as a master subroutine.

The proposed method consists of the following five procedures: (1) triangulation (i.e. partitioning into a number of triangles) of the x-y plane; (2) determination or selection of several data points that are closest to each data point and are used for estimating the partial derivatives; (3) location of the output point at which bivariate interpolation is to be performed (i.e. determination of the triangle in which the point lies), or organization of the output grid points for smooth surface fitting by sorting them with respect to triangle numbers; (4) estimation of partial derivatives at each data point; and (5) punctual interpolation (i.e. interpolation at a point) at each output point. Since these procedures are computationally independent of each other, it is considered desirable to program a separate subroutine for each procedure.

In some applications, repeated computations with the same x and y coordinate values of the data points are required. In such applications the first two procedures described in the preceding paragraph do not have to be repeated if the intermediate results are properly saved. In other applications, repeated computations with the same x and y values of both the data points and the output points are required; thus the first three procedures do not have to be repeated. It is considered desirable to devise the algorithm in such a way that the user can save computation time in these applications by avoiding unnecessary repetitions.

This algorithm requires a work area that is used during computation to store intermediate results. This work area can be reserved either in the subprograms included in the algorithm or in the calling program by the user. For flexibility of the algorithm, we prefer the latter.

Triangulation is a time-consuming procedure. The triangulation algorithm in the previous report [4] required a long computation time that was proportional to the cube of the number of data points. It also required a work area that was proportional to the square of the number of data points. Because of these requirements, its applicability was practically limited to 100 data points at most on most computers. By implementing the procedures described in Section 2, we have improved the triangulation algorithm substantially. The triangulation algorithm presented in Algorithm 526 [5] requires a computation time that is approximately proportional to the square of the number of data points (e.g. 0.005, 0.48, and 32.0 seconds for 10, 100, and 1000 data points, respectively, on the CDC-6600 computer). It requires a work area that is proportional to (and about 20 times) the number of data points.

Selection of several data points that are closest to a specified data point is another time-consuming procedure. The selection algorithm presented in the previous report [4] was inefficient, and an improvement has been made. The improved selection algorithm in Algorithm 526 [5] requires a computation time that is approximately proportional to the square of the number of data points and is less than half of that required for triangulation for the same number of data points.

Locating a point in the triangular grid (i.e. determining the triangle in which the point lies) is also a time-consuming procedure. We have also improved the algorithm for this procedure by dividing the x-y plane into nine rectangular areas and by listing triangles associated with each rectangle. The location algorithm presented in Algorithm 526 [5] requires a computation time that is approximately proportional to the square root of the number of data points; it is several times faster than the one in the previous report [4].

For smooth surface fitting, all rectangular grid points are located and interpolation is performed at all grid points. In this case, both location of rectangular grid points and interpolation of values are faster on triangle by triangle basis than on grid-point by grid-point basis. We have programmed a new subroutine that organizes the grid points for surface fitting by sorting them with respect to triangle numbers; for each triangle, this subroutine sweeps the rectangular grid points, searching grid points that lie inside the triangle, and lists them in a table. In the surface fitting subroutine, interpolation is performed at grid points in a triangle consecutively so that unnecessary recalculations of the polynomial coefficients are avoided.

The choice of computing at the beginning and saving the estimates of partial derivatives at all data points or computing the estimates at a point each time is another programming problem. We have chosen the former because the first step of the proposed method for estimating partial derivatives at a data point involves estimation at several data points in addition to the data point in question and it is simpler to do the first step for all data points first. Obviously, this choice is inefficient in the case of many data points and few output points. By reducing data points beforehand in such a case, however, the user can save computation times not only for the derivative estimation but also for other procedures including triangulation.

4. APPLICATIONS

Using a simple example taken from the previous study [2, 3], we illustrate the application of the proposed method. We take a quarter of the surface shown in the example in the previous study and sample 50 data points from the surface randomly. The coordinate values of the sampled data points are shown in Table I. Knowing from the physical nature of the phenomenon that $z(x, y)$ is a single-valued smooth function of x and y, we try to interpolate the z values and to fit a smooth surface to the given data points.

Figure 1 depicts contour maps of the surface resulting from the 30 data points with asterisks in Table I, while Figure 2, from all the 50 data points in the table. In these contour maps, projections of the data points are marked with encircled points. In each figure, the original surface from which the data points were sampled is shown in (a). The surface fitted with piecewise planes (i.e. the surface consisting of a number of pieces of planes, each applicable to one triangle) is shown in (b). Of course, such a surface is continuous but not smooth. The surface fitted by the method that estimates the partial derivatives with a second-degree polynomial is shown in (c). The surfaces fitted by the proposed method using three, four, and five additional data points for estimation of partial derivatives at each data point

Table I. An Example Set of Data Points

(Thirty points with asterisks are used in Figure 1, while all 50 points are used in Figure 2.)

i		x_i	y_i	z_i	i		x_i	y_i	z_i
1	*	11.16	1.24	22.15	26		3.22	16.78	39.93
2	*	24.20	16.23	2.83	27	*	0.00	0.00	58.20
3		12.85	3.06	22.11	28	*	9.66	20.00	4.73
4	*	19.85	10.72	7.97	29		2.56	3.02	50.55
5	*	10.35	4.11	22.33	30	*	5.22	14.66	40.36
6		24.67	2.40	10.25	31	*	11.77	10.47	13.62
7	*	19.72	1.39	16.83	32		17.25	19.57	6.43
8		15.91	7.74	15.30	33	*	15.10	17.19	12.57
9	*	0.00	20.00	34.60	34	*	25.00	3.87	8.74
10	*	20.87	20.00	5.74	35		12.13	10.79	13.71
11		6.71	6.26	30.97	36	*	25.00	0.00	12.00
12		3.45	12.78	41.24	37		22.33	6.21	10.25
13	*	19.99	4.62	14.72	38		11.52	8.53	15.74
14		14.26	17.87	10.74	39	*	14.59	8.71	14.81
15	*	10.28	15.16	21.59	40	*	15.20	0.00	21.60
16	*	4.51	20.00	15.61	41		7.54	10.69	19.31
17		17.43	3.46	18.60	42	*	5.23	10.72	26.50
18		22.80	12.39	5.47	43		17.32	13.78	12.11
19	*	0.00	4.48	61.77	44	*	2.14	15.03	53.10
20		7.58	1.98	29.87	45	*	0.51	8.37	49.43
21	*	16.70	19.65	6.31	46		22.69	19.63	3.25
22	*	6.08	4.58	35.74	47	*	25.00	20.00	0.60
23		1.99	5.60	51.81	48		5.47	17.13	28.63
24	*	25.00	11.87	4.40	49	*	21.67	14.36	5.52
25	*	14.90	3.12	21.70	50	*	3.31	0.13	44.08

are shown in (d), (e), and (f), respectively. In drawing these contour maps, the z values were interpolated by their respective methods at the nodes of a grid consisting of 100 by 80 squares; in each square, the z values were interpolated linearly.

Figures 1 and 2 indicate that the proposed method yields reasonable results although these results might not necessarily be satisfactory for some applications. In these figures very little difference is exhibited in the resulting surfaces due to the difference in the number of data points used for the estimation of partial derivatives in the proposed method. Figures 1(c) and 2(c) demonstrate a peculiar idiosyncracy of the method based on second-degree polynomials: more data points yield a much worse result in this example.

The decision whether or not the proposed method is applicable to a particular problem rests on each prospective user of the method. The examples given here are

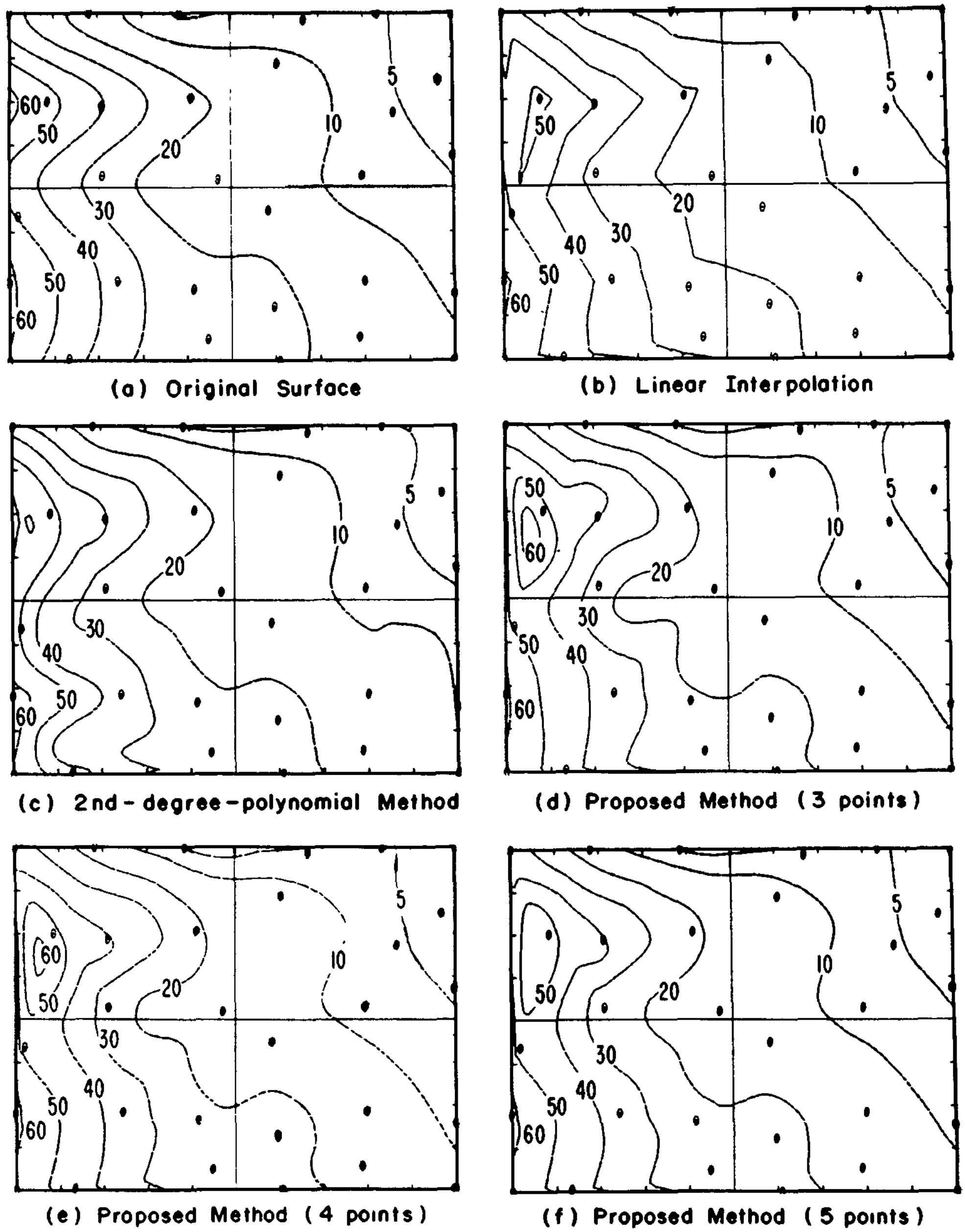

Fig. 1. Contour maps for the surfaces fitted to 30 data points given by asterisks in Table I. (The number of points in the captions for (d), (e), and (f) are the number of additional data points used for estimating partial derivatives at each data point.)

expected to aid one in making such a decision. Comparison of (d), (e), or (f) fitted by the proposed method with (a) the original surface or (b) the piecewise-plane surface in each figure should be helpful for such a decision. Also, comparison of Figures 1 and 2 gives one some idea of the dependence of the resulting surfaces upon the total number of data points and the complexity of original surfaces.

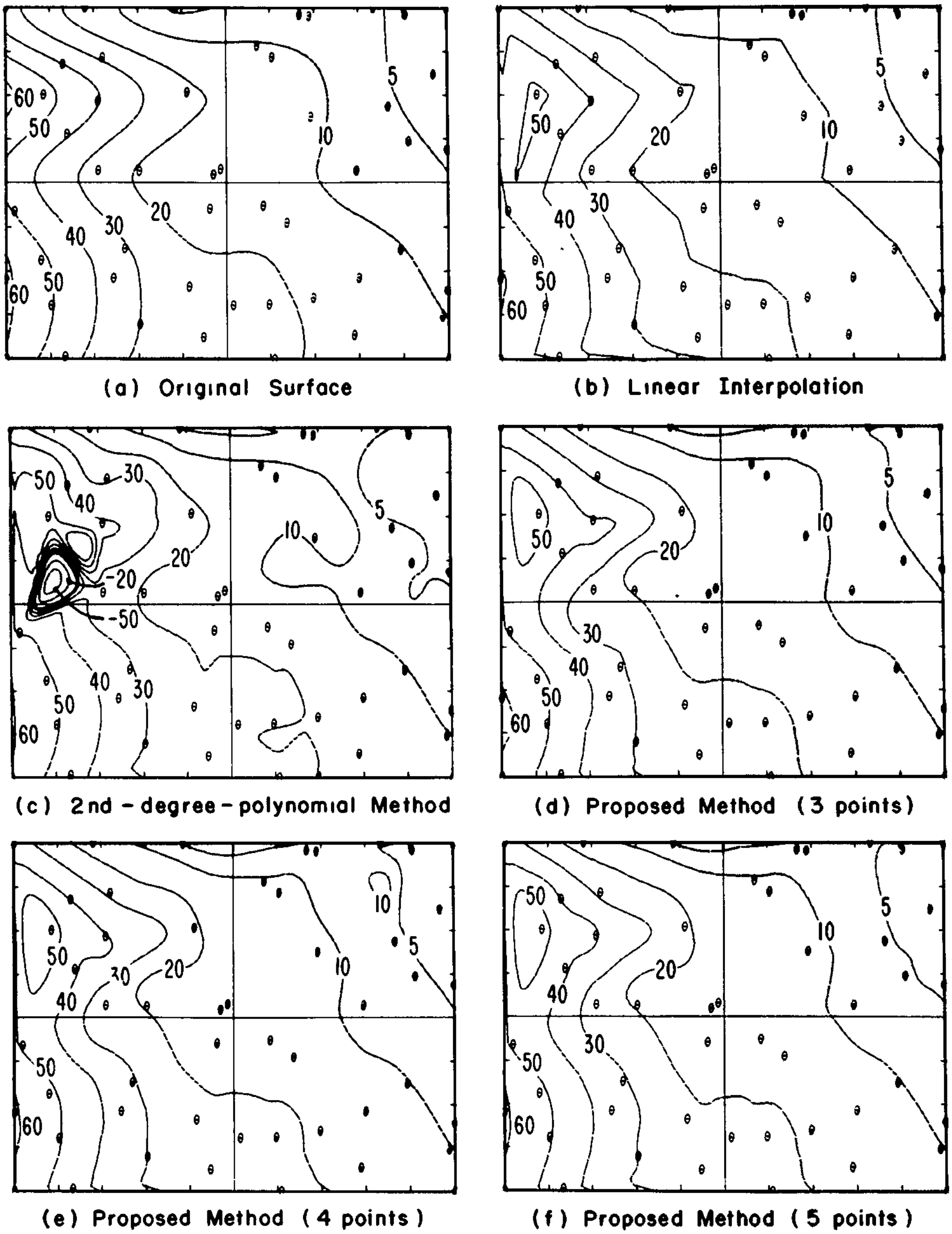

Fig. 2. Contour maps for the surfaces fitted to 50 data points given in Table I. (See the note in the caption for Figure 1.)

5. CONCLUDING REMARKS

We have described a method of bivariate interpolation and smooth surface fitting that is applicable when z values are given at points irregularly distributed in an x-y plane. For proper application of the method, the following remarks seem pertinent:

(i) The method does not smooth the data. In other words, the resulting surface passes through all the given points if the method is applied to smooth surface fitting. Therefore, the method is applicable only when the precise z values are given or when the errors are negligible.

(ii) As is true for any method of interpolation, the accuracy of interpolation cannot be guaranteed, unless the method in question has been checked in advance against precise values or a functional form.

(iii) The result of the method is invariant under a rotation of the x-y coordinate system.

(iv) The method is linear. In other words, if $z(x_i, y_i) = az'(x_i, y_i) + bz''(x_i, y_i)$ for all i, the interpolated values satisfy $z(x, y) = az'(x, y) + bz''(x, y)$, where a and b are arbitrary real constants.

(v) The method gives exact results when $z(x, y)$ represents a plane; i.e. $z(x, y) = a_{00} + a_{10}x + a_{01}y$, where a_{00}, a_{10}, and a_{01} are arbitrary real constants.

(vi) The method requires only straightforward procedures. No problem concerning computational stability or convergence exists in the application of the method.

A computer subprogram package that implements the proposed method is presented in Algorithm 526 [5].

ACKNOWLEDGMENT

A sincere acknowledgment is due to an anonymous reviewer who suggested numerous changes and improvements on the original version of this paper and the accompanying algorithm. The author is grateful to James C. Ferguson of Teledyne-Ryan Aeronautical, San Diego, Calif., for his information concerning the past works in the subject field, and to Harold T. Dougherty, Theodore W. Hildebrandt, Rayner K. Rosich, and Vincent D. Wayland of the NTIA, Boulder, Colo., for their helpful discussions.

REFERENCES

1. Ackland, T.G. On osculatory interpolation, where the given values are at unequal intervals. *J. Inst Actuar.* *49* (1915), 369–375.
2. Akima, H. A method of bivariate interpolation and smooth surface fitting based on local procedures. *Comm. ACM 17*, 1 (Jan. 1974), 18–20.
3. Akima, H. Algorithm 474. Bivariate interpolation and smooth surface fitting based on local procedures. *Comm. ACM 17*, 1 (Jan. 1974), 26–31.
4. Akima, H. A method of bivariate interpolation and smooth surface fitting for values given at irregularly distributed points. OT Rep. 75-70, U.S. Govt. Printing Office, Washington, D.C., Aug. 1975.
5. Akima, H. Algorithm 525. Bivariate interpolation and smooth surface fitting for irregularly distributed data points. *ACM TOMS 4*, 2 (June 1978), 160–164.
6. Bengtsson, B.-E., and Nordbeck, S. Construction of isarithms and isarithmic maps by computers. *BIT 4*, 2 (April 1964), 87–105.
7. Heap, B.R. Algorithms for the production of contour maps. NPL Rep. NAC-10, Nat. Phys. Lab., Teddington, England, Feb. 1972.

8. LAWSON, C.L. Generation of a triangular grid with application to contour plotting. Tech. Memo. 299, Sect. 914, Jet Propulsion Lab., Calif. Inst. Tech., Pasadena, Calif., Feb. 1972.
9. SHEPARD, D. A two-dimensional interpolation function for irregularly-spaced data. Proc. 1968 ACM Nat. Conf., pp. 517–524.
10. WHITTEN, E.H.T., AND KOELLING, M.E.V. Computation of bicubic-spline surfaces for irregularly-spaced data. Tech. Rep. 3, Dept. of Geol. Sci., Northwestern U. Evanston, Ill., Jan. 1975.
11. ZENISEK, A. Interpolation polynomials on the triangle. *Numer. Math. 15* (1970), 283–296.
12. ZLAMAL, M. On the finite element method. *Numer. Math. 12* (1968), 394–409.

Received December 1976; revised June 1977