# Two dimensional interpolation from random data

D. H. McLain

*Computing Centre, University of Sheffield, Sheffield, S10 2TN*

**A method is described for smooth interpolation between random data points in two or more dimensions. The method gives a smooth surface passing exactly through the given data points, and is suitable for graphical applications. It has practical advantages over other published algorithms, including that recently described by Maude (1973), to which it is similar, being both easier to implement and faster in computer operation.**

(Received March 1974)

## 1. Introduction

In this paper we describe a method for smooth interpolation in two dimensions between data provided at a set of points arbitrarily distributed on a plane. Because the method ensures the continuity of the resulting surface, and its first two derivatives, it is suitable for graphical applications.

Most approaches to this problem involve subdividing the plane into small regions, and determining different approximating functions within the different regions. Thus Bengtsson and Nordbeck (1964) partition the plane into triangles by connecting data points, and interpolate linearly within each triangle from the data values at the vertices. This method has the disadvantage that, although the surface is continuous across the edges of the triangles, its slope is not, so that, for example, a contour line usually has a kink when it crosses an edge. In the completely different and ingenious method of Hayes and Halliday (1972), Hayes (1973), the plane is divided into rectangles, and a bicubic spline determined using the edges of the rectangles as its nodes. The coefficients of the spline are not found from the values at the nodes, as in the usual applications of bicubic splines when the data are given on a regular mesh (De Boor, 1962), but are calculated using a statistical least-squares fit, so that the resulting surface fits as closely as possible with the data. The surface produced by this technique is, of course, smooth and visually acceptable, but it will not in general pass through all the data points. In a third, very different approach, Maude (1973) splits the plane into regions which are formed by the intersections of circles, one circle surrounding each data point; Maude's method will be discussed in more detail below.

Departing from the concept of subdivision of the plane into regions usually leads to unacceptably large computer time, unless the calculation is used only to calculate the interpolated function at the nodes of a regular mesh, so that a simpler formula, such as bicubic splines (De Boor, 1962) can subsequently be used (McLain, 1974). This approach is adopted in a number of proprietory contouring programs, but the resulting surface will not normally pass through any of the original data points which are not on the rectangular mesh.

The method of this paper is, in many ways, similar to that of Maude (1973), and in practice gives similar results over the regions where the later is valid. But instead of being based on circles and their intersections, it uses the triangles of Bengtsson and Nordbeck (1964). We shall show that there are substantial computational advantages in this.

## 2. Maude's method

For each data point $P_d$, Maude calculates a polynomial approximation

$$f(x, y) = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2$$

using the function values at that point and the five nearest points to determine the six coefficients. This function is used within the circle with centres $P_d$ and with radius just large enough to enclose the five nearest data points. The approximation at any point $P$ is a weighted average of the function values corresponding to all those circles which enclose $P$. The weights for the averaging process can be chosen to make the resulting function smooth across the boundaries of the separate regions, by ensuring that, at the circumference of each circle, the associated weight and its leading derivatives are zero. The concepts behind the averaging process are similar to those underlying the well known surfaces of Coons (1969), lucidly described in Forrest (1972), where the functions serving a similar purpose to Maude's weights are presented as 'blending functions'.

Unfortunately the method is both somewhat awkward to implement and inefficient in computer operation. The implementation difficulties arise from the number of special cases which should be included in the computer program to allow for degenerate situations. Consider, for example, a number $N$ of data points regularly spaced round a circle with centre $C$. For $N = 12$ the regions of validity of the approximations intersect as shown in **Fig. 1**, with 12 circular regions enclosing $C$. However as $N$ is increased the radii of these circular regions decreases, until when $N = 18$ they all pass precisely through $C$, the radii of the circles being equal to the radius of the original circle. Maude's approximation at the point $C$ is degenerate in this case, as all the weights are zero. When $N$ exceeds 18 none of these regions encloses $C$, their radii being smaller than that of the original circle, and a different type of degeneracy occurs.

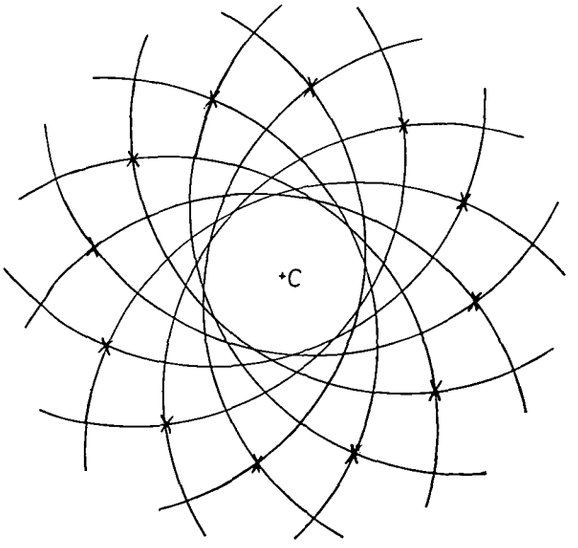


**Fig. 1 Regions of significance using Maude's method**

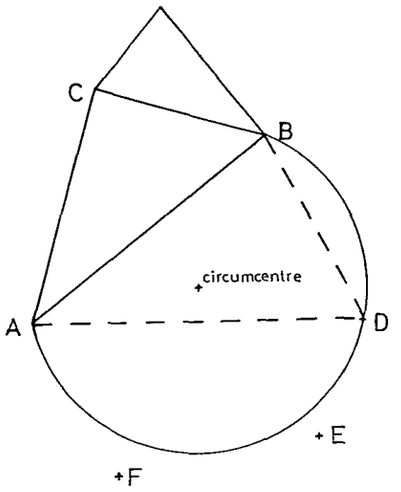


**Fig. 2 Triangulating the plane**

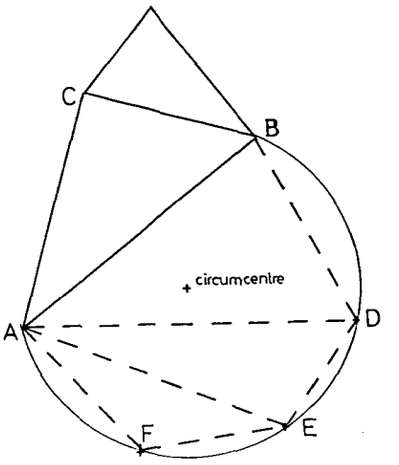


**Fig. 3 Triangulating the plane—the special case**

The cause of inefficiency in the computer run time is the complexity of the boundaries of the regions where the circles intersect. Probably the most efficient way to implement typical applications, such as contouring, where the approximation must be repeatedly calculated for a series of neighbouring points, is to retain not only a list of the circles in which the current point lies, but also a list of neighbouring circles and their distances, frequently updating both lists.

### 3. Triangulation of the plane

The first step in the present method is to partition the plane into triangles by connecting neighbouring data points. It is perhaps not as well known as it might be that a simple algorithm exists not only to do this, but to form an optimal partition, as described by Pitteway (1973). An optimal partition is one in which, for any point within any triangle, that point lies at least as close to one of the vertices of that triangle as to any other data point. The algorithm is as follows.

Suppose that a number of triangles have been selected, and that a line $AB$ is a side of one (not two) of these, $ABC$, as in **Fig. 2**. Then the algorithm should examine those points on the side of $AB$ opposite to $C$, e.g. the points $D$, $E$, $F$ in Fig. 2. For each of these, $D$ say, it should find the centre of the circle through the three points $A$, $B$, $D$ and the distance of this centre from the line $AB$, measured in such a direction that the distance from $C$ to $AB$ is negative. The triangle $ABD$ for which this distance is least is then to be included in the list. Note that this is not quite the same as choosing the triangle $ABD$ whose circumcentre lies closest to $AB$ since the sign of the distance is taken into account.

The algorithm may be started either with a line $AB$ on the boundary of the region or with a line joining any point to its nearest neighbour; it terminates when no point can be found on the appropriate side of any of the lines described above.

There is one complication which should be allowed for, when two or more triangles $ABD$, $ABE$, ... have the same circumcentre. This will be detected if, when the distances to two circumcentres are being compared, equality is found. The simplest action to take in this situation is to record which points are involved, to sort them into order, $D$, $E$, $F$, ..., say, as in **Fig. 3** around the circumference of their circumcircle, e.g. by finding (the cosines of) the angles $DAB$, $EAB$, ..., and to add all the triangles $ABD$, $ADE$, $AEF$, ... to the list.

Appendix 1 gives ALGOL procedures to analyse on which side of a line a point lies, and to find the distance of the circumcentre of a triangle $ABD$ from a side $AB$.

**Fig. 4** shows the result of applying this triangulation algorithm to the data points shown. The figure includes lines outward from the corners of the triangulated region to allow extrapolation outside the region of the data points, as described below.

### 4. The interpolation method

As in Maude's method, for each of the data points the program should calculate the coefficients of a polynomial approximation

$$f_{\text{data point}}(x, y) = a_{00} + {}_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2$$

One may do this by Maude's method, to give an exact fit at that point and the five nearest points. However, the author has found that, in practice, one gets slightly more accurate results, and at the same time avoids problems from degenerate cases, by using more data points than six, and the method of distance-weighted least squares fit of McLain (1974), in which the algorithm is given as Appendix 1. This method is a variant of the standard statistical technique which determines the coefficients $a_{ij}$ to minimise the sum of the squares of the deviations between the calculated values $f(x, y)$ and the given values at the data points, which weights the deviations so that remote points carry much less weight than neighbouring points.

In each triangle the final approximation is found as the

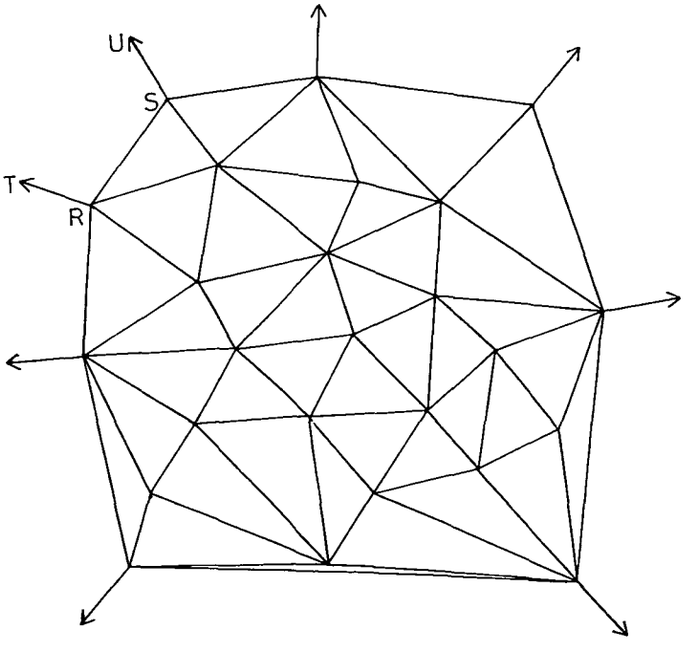


**Fig. 4 A triangulated plane using 25 points**

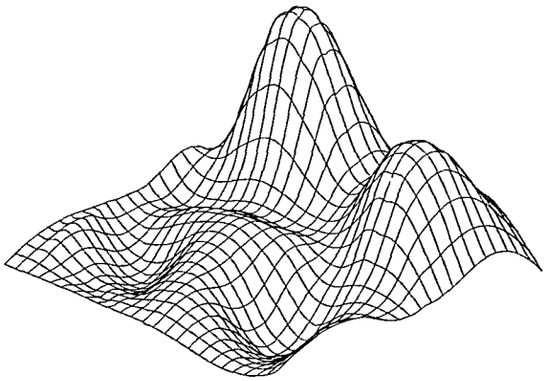

**Fig. 5 A surface through 25 data points**

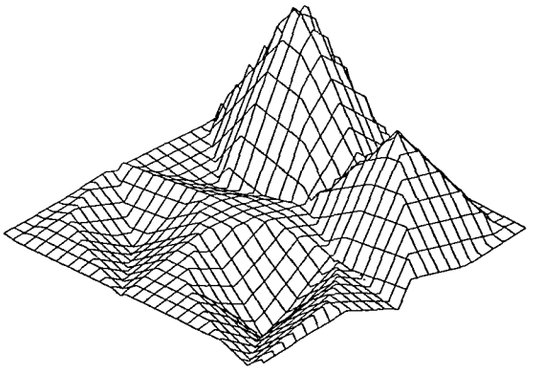

**Fig. 6 Linear interpolation through the same 25 data points**

weighted average of the three functions corresponding to the vertices:

$$F = w_1 f_1 + w_2 f_2 + w_3 f_3 \ .$$

To ensure smooth transition from one triangle to the next, we need only ensure that each weight $w_i$ and its leading derivatives are identically zero along the side of the triangle opposite to the $i$th vertex. This can be achieved by making $w_i$ proportional to the $n$th power $d_i^n$ of the distance from the side, for some $n$, typically 3. The distance $d_i$ of the point $(x, y)$ to the side is, of course, a linear function of $x$ and $y$

$$d_i = l_i x + m_i y + n_i$$

where the coefficients $l_i, m_i, n_i$ can be found easily and only once per side; they should be scaled to make $d_i = 1$ at the vertex $i$. Then

$$w_i = d_i^n/(d_1^n + d_2^n + d_3^n) \ .$$

In practice it is inadvisable to have the number $n$ larger than the value 3 which ensures continuous first and second derivatives from one triangle to another. The larger the value of $n$ is made, the more the transition between neighbouring functions $f_i$ will be concentrated close to the perpendicular bisectors of the sides of the triangles, giving visually unnatural surfaces.

*It is easy to extend the method to extrapolate outside the* convex hull of the data points, i.e. outside the region covered by the triangles. To do this one can partition the exterior of this region by extending lines outwards from the data points on the boundary, for example by bisecting the exterior angles at the corners. The approximation may then be found as a weighted average of the functions for the two boundary points, e.g. in Fig. 4 for the points $R$, $S$ in the region bounded by the lines $RS$, $RT$, $SU$.

### 5. Implementation in graphical applications

A major practical advantage arises within the present approach compared with that of Maude, because of the simple boundaries of the regions over which the same formulae are applied, i.e. the triangles. For example, if we are drawing contours, then we might handle the triangles one at a time, drawing all the contour lines within one triangle before considering the next. The test for whether the current point is still inside a triangle is merely a test of the signs of the three distances $d_i$ which have previously been calculated—these will all be positive only if the point is within the triangle.

**Fig. 5** shows a perspective view of the surface obtained from data provided at the 25 points shown in Fig. 4, the view being drawn using the method of Ellis (1975). For comparison, **Fig. 6** shows the surface obtained from the same data points using the linear interpolation method of Bengtsson and Nordbeck (1964), and the same triangulation of the plane.

### 6. Higher dimensions

It is straightforward to generalise the method to the problem of interpolation in three or more dimensions. Consider, for example, the case of three dimensions. The points should be grouped into fours to partition the region into tetrahedra, and a method similar to that of Section 3 is suitable to do this. Within each tetrahedron the interpolation is achieved by averaging the polynomials defined at the four vertices. The weights for the averaging process should be proportional to the cube of the distances to the faces, and again these distances are linear functions of the co-ordinates.

## Appendix 1

This appendix gives ALGOL procedures to assist in the triangulation of the plane, as described in Section 3, above.

The first procedure, *findlmn*, calculates the coefficients $l, m, n$ of the equation $lx + my + n = 0$ of the line joining two points $A, B$. The coefficients are adjusted so that the value of the expression $lx + my + n$ is positive or negative according to whether a point $(x, y)$ lies respectively on the opposite or the same side of the line $AB$ as a third point $C$, see Fig. 2. The co-ordinates of the three points are given as the six procedure parameters *xa, ya, xb, yb, xc, yc*. The procedure also calculates the value of a fourth variable *c1*, which is used in the procedure *distancetocentre*, below.

```
procedure findlmn(xa, ya, xb, yb, xc, yc, l, m, n, c1);
  value xa, ya, xb, yb, xc, yc;
  real xa, ya, xb, yb, xc, yc, l, m, n, c1;
begin
  l := yb − ya; m := xa − xb;
  n := − l × xa − m × ya;
  if l × xc + m × yc + n > 0 then
  begin l := −l; m := −m; n := −n end;
  c1 := m × (xa + xb) − l × (ya + yb)
end;
```

The second procedure *distancetocentre* may be used to compare the distances from the line $AB$ to the circumcentre of the triangle $ABD$, for various points $D$. The procedure actually finds a linear function of the distance from $AB$ to such a circumcentre (if $t$ is the true distance, the procedure calculates $2(\sqrt{l^2 + m^2}\, t - n)$), which is sufficient if, as described in Section 3, we need only find the minimum of these. It is assumed that the procedure *findlmn* has been called prior to *distancetocentre*.

The co-ordinates of the points $A$, $D$ are given as the parameters *xa, ya, xd, yd*, and the values of the other parameters $l, m$, *c1* are those calculated by the procedure *findlmn* for the line $AB$.

```
real procedure distancetocentre(xa, ya, xd, yd, l, m, c1);
  value xa, ya, xd, yd, l, m, c1;
  real xa, ya, xd, yd, l, m, c1;
```

```
begin real a2, b2, c2;
  a2 := xa − xd; b2 := ya − yd;
  c2 := a2 × (xa + xd) + b2 × (ya + yd);
  distancetocentre := (l × (b2 × c1 + l × c2) + m ×
                      (m × c2 − a2 × c1))/(m × b2 + l × a2)
end;
```


**References**

BENGTSSON, B. E., and NORDBECK, S. (1964). Construction of isarithms and isarithmic maps by computers, *BIT*, Vol. 4, p. 87.

BOOR, C. DE (1962). Bicubic spline interpolation, *J. Math. and Phys.*, Vol. 41, p. 212.

COONS, S. A. (1969). *Surfaces for computer aided design of space forms*, MAC-TR041, Massachusetts Institute of Technology.

ELLIS, T. M. R. (1975). *Some routines for plotting three dimensional surfaces*, Computing Services Report U3, University of Sheffield.

FORREST, A. R. (1972). On Coons and other methods for representation of curved surfaces, *Computer Graphics and Information Processing*, Vol. 1, pp. 341-359.

HAYES, J. G., and HALLIDAY, J. (1972). The least-squares fitting of cubic spline surfaces to general data sets, *NPL Report NAC 22*, National Physical Laboratory.

HAYES, J. G. (1973). Available algorithms for curve and surface fitting, *NPL Report NAC 39*, National Physical Laboratory.

MCLAIN, D. H. (1974). Drawing contours from arbitrary data points, *The Computer Journal*, Vol. 17, pp. 318-324.

MAUDE, A. D. (1973). Interpolation—mainly for graph plotters, *The Computer Journal*, Vol. 16, pp. 64-65.

PITTEWAY, N. L. K. (1973). Computer graphics research in an academic environment, *Datafair 73 Conference Proceedings*, Vol. 2, pp. 471-478.


## Advanced Study Institute on Man-computer Interaction

A NATO Advanced Study Institute on man-computer interaction (MCI) will be held at Mati near Athens, Greece from 5 to 18 September 1976.

The Institute will review current knowledge and recent research on human aspects of man-computer interaction. Topics will include hardware and software design, programming, interaction with different classes of user, training and modelling; the emphasis throughout will be upon the ergonomics/human factors problems and solutions.

Papers on the main topics will be presented by invited lecturers, and these will be followed up by seminars and discussions at which participants will be expected to contribute.

The Advanced Study Institute's programme sponsored by NATO aims to further international collaboration between scientists through in-depth study of important areas of research.

Priority will be given to applicants who were accepted for the 1975 ASI on MCI which had to be postponed, but there will be a number of places for new applicants.

Numbers are therefore restricted, so it is important to apply as early as possible. Financial assistance may be available to suitably qualified participants (usually doctoral and post-doctoral students). For application form write to:

Professor B. Shackel—Director ASI on MCI,
Department of Human Sciences,
University of Technology,
Loughborough, UK.

## Advanced Summer Institute on Computer Architecture

*Dates and Location:* September 12-24, 1976, St. Raphaël on the French Riviera.

*Sponsors:* NATO Scientific Affairs Division European Research Office.

*Programme:* Principles of Computing Systems; Fundamentals of Computer Architecture; Problem and/or Language Orientated Machines; Hardware components (including microprocessors); Associative Processing Techniques; Specification and Evaluation of Computer Structures; Computer Networks and Communications; PDP 11 Systems Design; MU5 Computer Architecture.

*List of Lecturers:* G. Amdahl, Amdahl Corporation, USA; D. Barber, NPL, UK; G. Bell, DEC, USA; K. Bowden, University of Essex, UK; B. Canet, Université de Rennes, France; G. Debruyne, INTEL, France; F. G. Heath, Heriot-Watt University, UK; F. H. Sumner, University of Manchester, UK; W. T. Wilner, Burroughs Corporation, USA.

*Directors of the School:* Professors G. Boulaye and D. Lewin.

*General Information:* All lecturers and delegates will be accommodated in a quiet comfortable hotel with ample facilities for informal discussion. Instruction will be by seminars and round-table discussions.

**Further information:** from
Professor D. Lewin
Department of Electrical Engineering and Electronics
Brunel University
Uxbridge
Middlesex UB8 3PH
(Tel.: Uxbridge 37188: ext. 118).