A remark on a contouring algorithm

D. C. Sutcliffe

*Department of Applied Mathematics and Computing Science, The University of Sheffield, Sheffield S10 2TN**

This paper describes an error in the contouring algorithm of McLain (1974) and suggests how it can be avoided. The result of the error is that under some circumstances the algorithm draws a curve which is not part of a contour.
(Received December 1974)

In the contouring algorithm of McLain (1974) an error arises when the contour passes the mouth of certain valleys. This has the effect that the algorithm may trace the bottom of these valleys rather than the true contours and so produce unwanted false contours.

Description of the method and the error

The contouring algorithm works as follows. The area under consideration is divided up into small rectangles. Within each rectangle, to draw a contour of height h of a function $f(x, y)$, firstly a zero of $f(x, y) - h$ is found along one of the sides. The contour is then traced through the rectangle by a series of steps in one of eight directions (N, NE, E, SE etc.). The direction of the next step to be taken is chosen from one of three which depend on the direction of the previous step. The three steps to be considered are one in the same direction as the previous step and two in the directions at 45° to either side of this, e.g. see **Fig. 1.**

For each point (1), (2), (3) the absolute value of $f(x, y) - h$ is calculated and the point with minimum value found. This is the point chosen for the next step. This process is repeated until the contour passes out of the rectangle. A record is kept of the x and y co-ordinates of the exit point. This is repeated for all other roots along the sides of the rectangle, first checking them against the current list of exit points (if they are members of this list then they are not used as starting points for tracing the contour).

However, in certain circumstances this algorithm does not follow the contour correctly. Consider the case when the surface has cross-section as in **Fig. 2** in the region of the height of the contour being drawn. The algorithm has the choice of three function values—one either side of the point where the true contour crosses the surface (1), (3) and one in the bottom of a 'valley' to the right (2). Since the algorithm only considers the absolute magnitude of the difference between the function value and contour being drawn, point (2) is wrongly chosen. Once this step has been made the algorithm will often have the choice of three values as for example in **Fig. 3.** Clearly point (1) is the point with the smallest absolute difference and so will be chosen. Consequently, rather than the contour being traced, this valley is mistakenly traced. The reason for this action is that the algorithm is restricted to tracing over a set of grid-points. If this were abandoned and actual roots were found at every step this problem would not arise. However, this is a far more complex method and will not be considered here.

It can easily be seen that (as only absolute values are considered) ridges as well as valleys can be followed in this way but, for simplicity, only valleys will be referred to in this paper.

When the algorithm was used to plot the contours of a set of quite complicated functions with a step size of about one/two hundredth of the length of the closed contours (i.e. a fairly large step size compared to the size of the geographical features) valleys were followed in approximately one third of the functions used. Thus it is a serious problem if this algorithm is to be used to draw contours other than for simple functions.

Example

Consider the function

$$f(x, y) = 8(4y + 7)^3 - 9(4y + 7)^2(2x + 3) + 3(4y + 7)(2x + 3)^2$$

in the area close to the origin. (In particular when the procedure is applied to the rectangle $|x|, |y| \leqslant 32$. Note that in this example the step size used is 1. This is not a measurement which would be used in practice but is just to make the example simpler. The function values for part of this rectangle are tabulated in **Table 1**). Consider the situation when the zero contour has been traced in this rectangle as far as $(-8, -2)$—see Table 1. It is now traced through $(-7, -2)$, $(-6, -2)$ as far as $(-1, -2)$. At this point, the traced line crosses the true contour to the point $(0, -1)$ and then follows the contour through the points $(1, -1)$, $(2, -1)$ to $(4, -1)$. Now the mouth of the valley is reached and there is a choice as indicated in Fig. 2 between the points $(5, -2)$, $(5, -1)$ and $(5, 0)$. Point $(5, 0)$ is chosen (having the smallest absolute value) and instead of the true contour being followed the valley is traced by way of the points (6, 0), (7, 0), (8, 1) etc.—see Table 1.

Various contours of this function (including the zero contour and the false contour) are shown in **Fig. 4**(*a*).

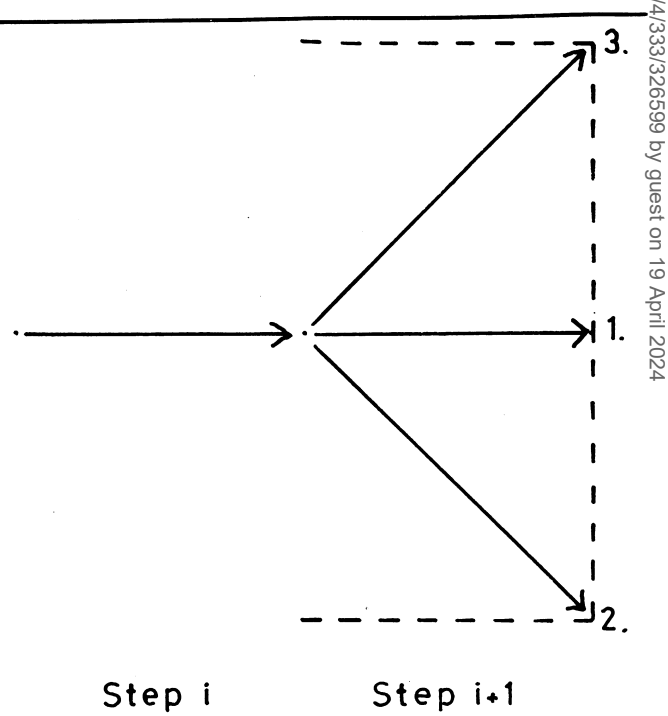

Fig. 1 Possible directions for step $i + 1$ based on step i

*Present address: Atlas Computing Division, Rutherford Laboratory, Chilton, Didcot, Oxon OX11 0QX

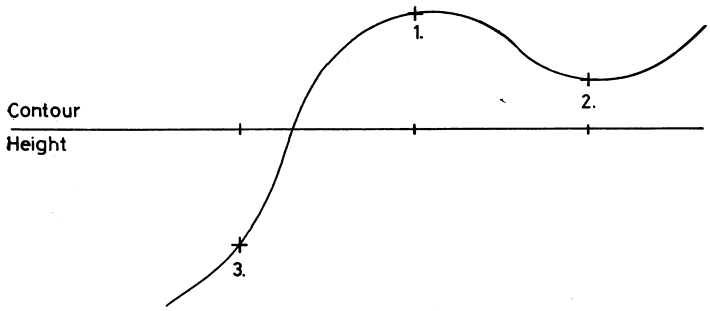

Fig. 2 Situation (showing choice of points) where error will occur

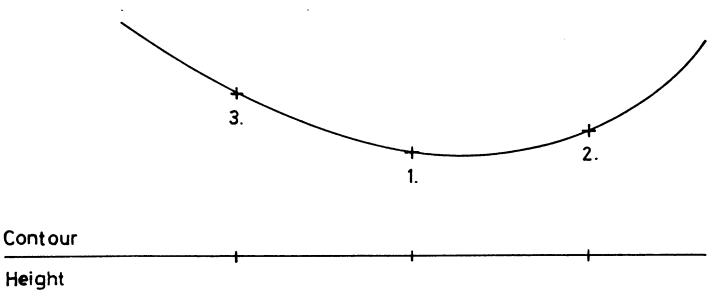

Fig. 3 Typical choice of points after error has occurred

Table 1

$f(x, y) = 8(4y+7)^3 - 9(4y+7)^2(2x+3) + 3(4y+7)(2x+3)^2$

The underlined values indicate the route taken by the original algorithm when tracing the zero contour starting from the left hand side of the rectangle.

	Y					
X	−3·0	−2·0	−1·0	0·0	1·0	2·0
−8·0	−610	−398	2790	12026	30382	60930
−7·0	−340	−272	2196	10136	26620	54720
−6·0	−190	−170	1674	8414	23122	48870
−5·0	−160	−92	1224	6860	19888	43380
−4·0	−250	−38	846	5474	16918	38250
−3·0	−460	−8	540	4256	14212	33480
−2·0	−790	−2	306	3206	11770	29070
−1·0	−1240	−20	144	2324	9592	25020
0·0	−1810	−62	54	1610	7678	21330
1·0	−2500	−128	36	1064	6028	18000
2·0	−3310	−218	90	686	4642	15030
3·0	−4240	−332	216	476	3520	12420
4·0	−5290	−470	414	434	2662	10170
5·0	−6460	−632	684	560	2068	8280
6·0	−7750	−818	1026	854	1738	6750
7·0	−9160	−1028	1440	1316	1672	5580
8·0	−10690	−1262	1926	1946	1870	4770
9·0	−12340	−1520	2484	2744	2332	4320
10·0	−14110	−1802	3114	3710	3058	4230
11·0	−16000	−2108	3816	4844	4048	4500
12·0	−18010	−2438	4590	6146	5302	5130

Recommended solution

The best solution to the problem appears to be a modification to the algorithm which determines the point when the algorithm starts following the valley and stops the tracing at that point. The other end of the contour will then be found, when the other roots along the sides of the rectangle are examined, and the rest of the contour will be traced from that end.

Consider the stage in the method where the algorithm has to choose between three values of $f(x, y) - h$. Look at the quadratic through the three points under consideration, assuming the centre one to have co-ordinate zero and the others minus one and plus one. A valley is being followed if the roots of the quadratic are imaginary or if the absolute value of the smallest is not sufficiently close to zero. Since this is a modification to the original algorithm which does not alter the path it follows but merely checks that it is correct, it will be equally as good. As it is successful in determining that a valley is being traced it will be better. However, it is rather time consuming to carry out this method at every step and so a simple process is employed to check the majority of the correct cases. Clearly, if not all three values of $f(x, y) - h$ are of the same sign, then the contour lies within a step size of the middle point and is being correctly traced. Hence, in a large number of cases, the quadratic method does not need to be employed.

Two points should be noted about the above method. If the contour crosses the side of the rectangle at a very narrow angle then for the first two steps the root would appear to be a long distance away. It is only after these first two steps that the algorithm is following the true contour—see **Fig. 5.** It is therefore recommended that the check is not applied until the third step and thereafter.

The second point is that when the previous step was diagonal the quadratic is being fitted through three points which are not

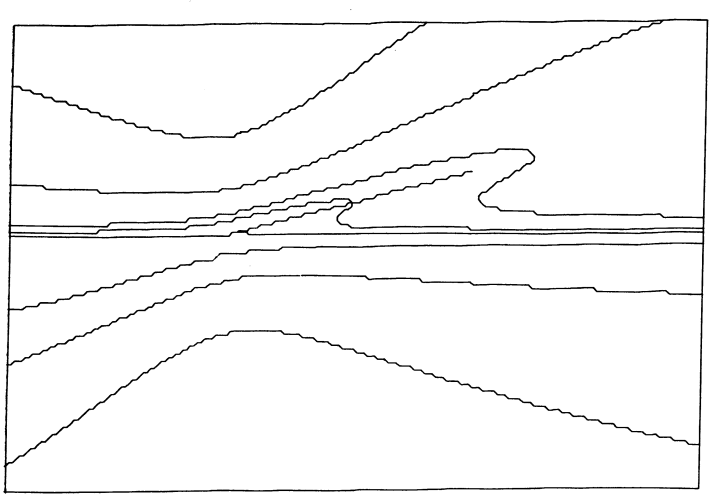

Fig. 4(*a*) Various contours of the example function drawn using the original algorithm (with a large step size for clarity)

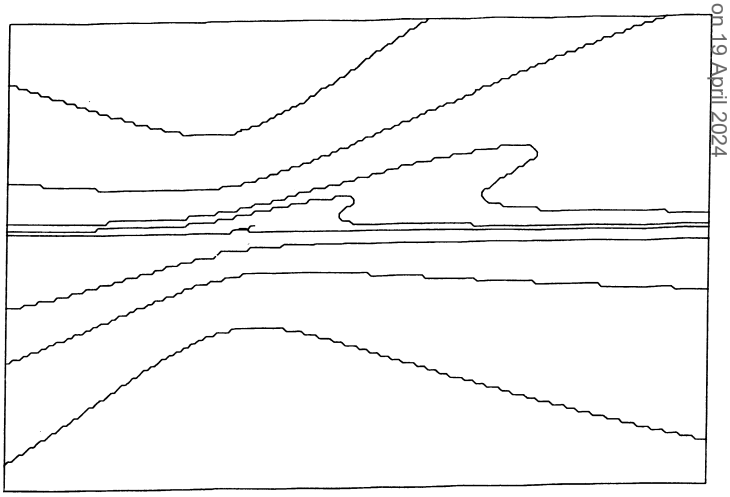

Fig. 4(*b*) The same contours of the example function drawn using the modified algorithm

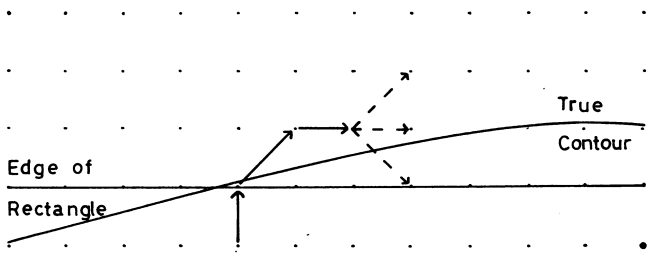

Fig. 5 Starting a contour

in a straight line. However, this does not in practice cause errors.

The algorithm modification necessary to implement this is outlined in the next section. The increase in computer run time in implementing this modification naturally depends on a number of factors, notably the complexity of the function $f(x, y)$. On an ICL 1907 the increase was found to be between seven per cent and nine per cent for simple functions such as $f(x, y) = x^2 + y^2 - 0{\cdot}25$, whereas for more complex functions the increase was below four per cent. Clearly, if the modification has prevented a valley being followed, the run time may actually be reduced.

Returning to the example function of Table 1 it is only after the step from $(-1, -2)$ to $(0, -1)$ that a quadratic need be fitted. For the next four steps it is not needed but it is employed after the step from $(4, -1)$ to $(5, 0)$. It is also employed after the next step when choosing between points $(7, -1)$, $(7, 0)$ and $(7, 1)$. In this case the roots are imaginary and so this step is not made. The valley has been detected and the tracing stops. The other end of the contour is picked up on the right hand side of the rectangle and is successfully traced all the way back through the rectangle. The contours of this function, drawn using the modified algorithm, are shown in Fig. 4(*b*).

This quadratic fit was considered preferable to two linear fits through the first two points and the second two points. In this case the points of intersection between the two straight lines and the contour height would be considered. Taking the co-ordinates of the points to be -1, 0 and $+1$ as before, the algorithm is considered to be working properly (i.e. following the true contour) if the point of intersection with the smallest absolute value is sufficiently small. This method correctly accepts points which are astride the contour and correctly rejects points which lie across the bottom of a valley if it is reasonably shallow. However, it tends to be rather insensitive to picking up valleys if one (or both) sides are steep and can trace a valley a long way before detecting it.

Recommended modification

In Appendix 2 of McLain (1974) the procedure drawcontour should be modified as follows.

The declaration

real array *z*[1:3];

should be added to the other declarations at the beginning of procedure drawcurvesintersectingside. The lines of that procedure

min := *abs* (*f*(*x* + *xstep*, *y* + *ystep*) − *contour*);
to **end** *of finding which of* 3 *directions to take*;
should be replaced by the following.

```
z[1] := f(x + xstep, y + ystep) − contour;
min := abs(z[1]);
for i: = 2, 3 do
begin z[i] := f(x + xinc[i], y + yinc[i]) − contour;
 temp := abs(z[i]);
 if temp < min then
 begin min := temp;
  xstep := xinc[i]; ystep := yinc[i]
 end
end of finding which of 3 directions to take;
begin
 Boolean procedure rootoutofrange(a, b, c);
 value a, b, c; real a, b, c;
 begin real discrim;
  comment this procedure calculates the absolute value of the
  root with the smallest absolute value of the quadratic
  a × x ↑ 2 + b × x + c. A value of false is returned unless
  no root exists or the root has a large value in which case the
  value true is returned;
  discrim := b × b − 4·0 × a × c;
  rootoutofrange := if a = 0·0 then
                    (if b = 0·0 then
                     (if c = 0·0 then false else true)
                    else abs(c/b) > 3)
                   else if discrim < 0·0 then true
                    else (if b < 0·0 then abs ((−b − sqrt
                         (discrim))/(2·0 × a))
                         else abs((−b + sqrt(discrim))/
                         (2·0 × a))) > 3
 end of rootoutofrange;
 if nstep > 2 then
 begin if abs(sign(z[1]) + sign(z[2]) + sign(z[3])) = 3 then
  begin if rootoutofrange (0·5 × (z[2] − 2·0 × z[1] + z[3]),
                           0·5 × (z[3] − z[2]), z[1])
   then goto ENDLINE
  end;
  comment a message may be output before the goto ENDLINE
  if desired;
 end
end of checking true contour being followed;
```

Acknowledgement

The author would like to thank the Science Research Council for a maintenance grant during this research.

Reference

McLain, D. H. (1974). Drawing contours from arbitrary data points, *The Computer Journal*, Vol. 17, p. 318.