When contour lines are digitised in a stream mode, excessive data is generated. Some algorithms are needed to remove this excess. In this paper, these algorithms are reviewed and compared, and finally a new algorithm is proposed based on selecting local maxima and minima.

An Algorithm for Compressing Digital Contour Data

Zhilin Li

Dept. of Geography, Glasgow University, Glasgow G12 8QQ, UK

INTRODUCTION

Existing contour maps are one of the main sources for digital mapping and DTM production. Before use can be made of such data, it is necessary to convert it into a digital form, i.e. digital contour data (DCD). The digitising may be carried out either in raster mode or in line following mode, the latter being the dominant one at present. Line-following digitising may be executed either in stream mode or in point mode. If the former is employed, the digitised DCD is very dense. That is to say, there is a great redundancy of data with stream mode, which creates problems within the computer system – more storage, more editing and more central processing unit (CPU) time are required. Therefore, some algorithms need to be applied to remove the excess data before it is used to generate DTM data.

The removal of this excess DCD data is referred to as compression in this paper, the main purpose of which is to describe a new algorithm. In addition, existing algorithms used for the purpose are reviewed and compared.

ALGORITHMS USED FOR COMPRESSING DCD:

A number of algorithms have been in use to compress the excess DCD (Douglas and Peucker, 1973), (ECU, 1971), (Lang, 1969), (McMaster, 1983). These are as follows:

1. *Retaining every Nth point:* This is the simplest and most often used method, in which all but the Nth point is deleted along a contour line. In other words, it can be considered as selecting every Nth coordinate pair as a coordinate pair on the filtered line. N is a fixed integer selected beforehand *(Figure 1)*.

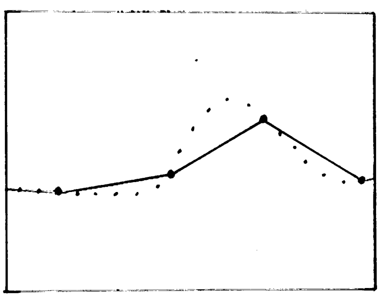

Figure 1. Data compression with Nth point algorithm.

There are two main disadvantages with this algorithm. The first is the frequent elimination or misrepresentation of important features along the line. The second is that the straight lines are still over-represented. (Douglas and Peucker, 1973).

2. *Retaining the point with a pre-defined distance (chord) from the last retained point:* In this method, the distance between present point and the last retained point is calculated and compared with the pre-defined distance (i.e. L in *Figure 9*). If the calculated distance is shorter than the pre-defined one, e.g. $AB<L$, then the present point, e.g. B, will not be selected. If the calculated distance is greater than L, then the present point, e.g. D will be selected. This algorithm has similar deficiencies to the first algorithm. This criterion can be modified to get a better result by replacing the distance between the present point and the last retained point with the accumulated distance from the last retained point to the present point. For example, $AB+BC+CD$ will be used instead of AD.

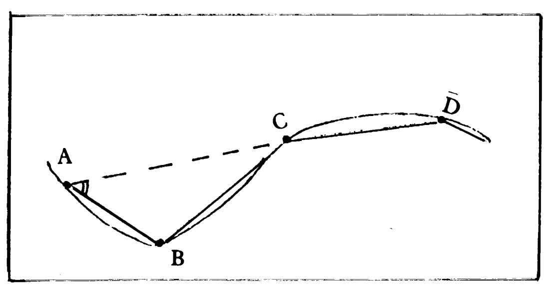

Figure 2. Data compression with angular algorithm.

3. *Filtering data with an angular tolerance:* In this method, the angle between the vector connecting the first point (A in *Figure 2*) and the third (C in the same figure) is calculated. If this angle is greater than an angular tolerance defined beforehand, then the point B is selected. If it is smaller, then it is deleted. This algorithm is very likely to delete important points along the line with small curvature. *(Figure 3)*.

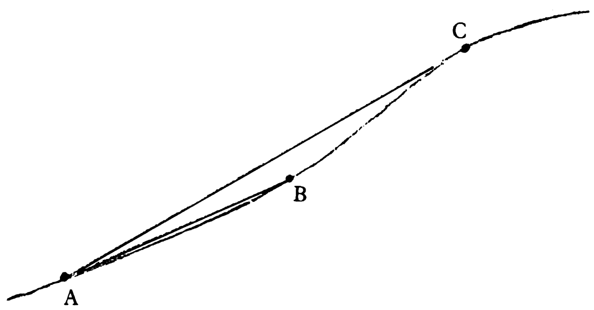

Figure 3. B is possibly omitted with angular algorithm.

4. *Filtering data using a perpendicular distance tolerance:* This method was first described by Lang (1969). The idea is illustrated in *Figure 4* P_0 is the start point, referred as the anchor point, and P_2 is the float point. If the distance from P_1 to line P_0 P_2 is greater than the distance tolerance (DT), then P_1 is selected as the next recorded point and becomes the new anchor point. Otherwise P_0 remains as the anchor and P_3 becomes new float point. Then the appropriate distances are checked again.

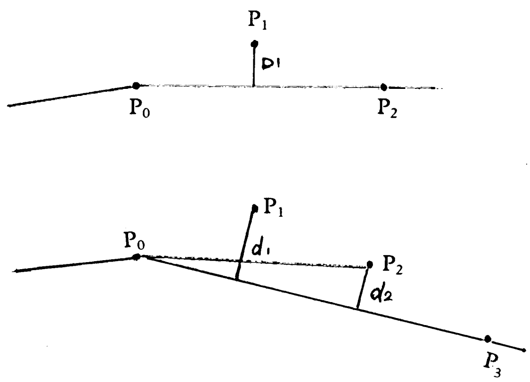

Figure 4. Data compression with distance tolerance.

As Douglas and Peucker (1973) state that "this algorithm was reported as producing an acceptable result, but needing too much computing time". They then borrowed the idea of distance tolerance to produce a modified algorithm, which is illustrated in *Figure 5.* The first point A is selected as the anchor point as usual, and the last point B is selected as first float point. Then the maximum distance, say FC, from the points digitised during the stream digitising process to the line AB is compared with DT. If $FC>DT$, then C become the new float point. Then maximum distance from points to AC, say GD, is compared with DT. If $GD>DT$, then D become new floating point, or else D is selected and become the new anchor point.

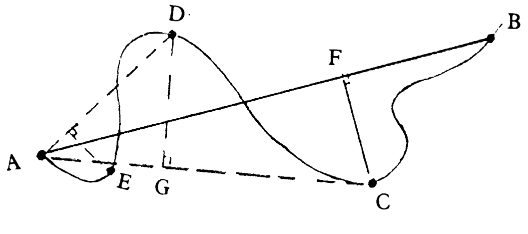

Figure 5. Data compression with D–P algorithm.

McMaster (1983) evaluates some of these algorithms and concudes that Douglas-Peucker algorithm produces the best result in terms of both vector displacement and area displacement. But this algorithm still needs a lot of computing time by calculating the distances of every point to the line connecting the anchor and last float point repeatedly. In addition, this algorithm seems to omit some important points in the very small curvature (very long radius) area, whose distances to the line between the anchor and the last float point are smaller than the predefined one, but they are very important in representing the feature of the contour line. In order to have DCD compressed more efficiently, a new algorithm based on selecting local minima and maxima has been developed, and the principle is described below.

AN ALGORITHM FOR COMPRESSING DCD ON SELECTING LOCAL MAXIMA AND MINIMA

The objective is to produce an algorithm which is able to give a result not only with high fidelity to the original DCD but also with less computing time (more efficiently) than that which is required by Douglas-Peucker algorithm. To achieve this, an ideal of selecting local maxima and minima is employed.

Selecting local maxima and minima involves selecting those points at which the slope of the line changes its sign, i.e. from negative to positive or from positive to negative. In order to emphasise the importance of selecting these points, an exaggerated example is shown in *Figure 6,* when $ACDB$ is the contour. In the Douglas-Peucker algorithm, if DD and DC are less than DT, then the straight line AB will be used to represent the curve $ACDB$. This is, of course, a very poor representation. In order to have a better representation, the points C and D should be selected.

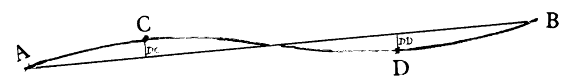

Figure 6. Points C and D are possibly omitted with D–P algorithm.

By close examination, it can be found that $C$and D are those points at which the positive slope value changes to a negative value or negative to a positive. This means that points C and D are local maxima and minima in the Y direction. Selecting local maximas and minimas in both X and Y directions is the main idea of detecting the most important points along a contour line in this algorithm.

The principle is illustrated in *Figure* 7. It can be seen from this diagram that the slopes change their signs at point C, D and E. In fact, Point C is local minima in X direction, and D and E are local maxima and minima in Y direction respectively. Therefore, these points together with A and B are very important in representation and should be selected. The detection of local maxima and minima is carried out by comparing the X and Y co-ordinates of three successive points. In *Figure 7,* M, D and N are three successive points. Firstly the sign of the differences between each two pairs of successive points are determined. For example, if $Y_M \leq Y_D$, then the sign of this difference of the Y coordinates is positive. If both signs are not the same, then point D is the local maxima or minima.

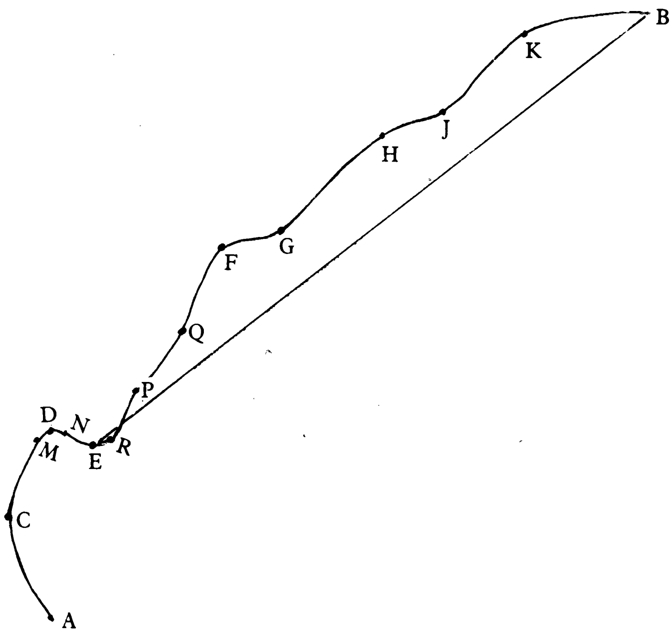

Figure 7. Data compression with the new algorithm.

But it can be seen that these points are not sufficient to represent this line appropriately. In particular, the large radius areas are omitted. This means that additional points must be selected. In this particular example, additional points must be selected in between point E and B.

Let us examine how a further selection is carried out by examining the points in between E and B. It is obvous that points F, G, H, J and K are particularly important in representing this segment. Therefore they should be selected. This suggests that we should make these points become local maxima or minima in order that they can be detected and selected.

One method to achieve this goal is to employ a conformal coordinate transformation. The transformation can be done by making line EB as new X coordinate axis. Thus points F, G, H, J, K and perhaps R become local maxima or minima in the new coordinate system, and can easily be detected. According to some predefined criterion, a further selection may also be carried out in between AC, CD, RF, etc. If a further selection is undertaken in between RF, then P and Q would be selected.

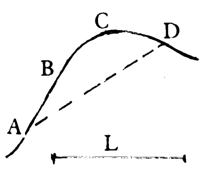

Figure 8. Data compression with chord tolerance.

A general conformal co-ordinate transformation consists of a translation and a rotation. However, the procedure could be simplified in this particular case by letting the new X axis pass through the same origin. *Figure 9* illustrates such a transformation. It can be seen that only a rotation remains necessary. The mathematical expression becomes:

$$
\begin{aligned}
X' &= X \cos(A) + Y \sin(A) \\
Y' &= -X \sin(A) + Y \cos(A) \\
A &= \arctan(DY/DX) \\
DX &= X_B - X_E \\
DY &= Y_B - Y_E
\end{aligned}
$$

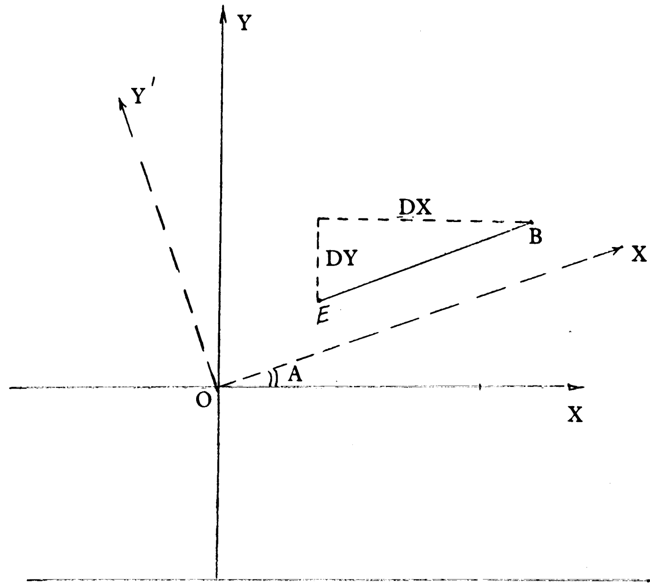

Figure 9. Conformal coordinate transformation with a rotation.

Whether any further points need to be selected can be specified by the user as the number of points within which there is no interest for selecting more points, or the number of selection runs, or a combination of both. Even the perpendicular distance criterion could be utilised. But, in this case the Y coordinate is such a distance if a general conformal transformation is employed.

TESTING RESULTS AND DISCUSSION

Four of these algorithms, the Nth point, the chord tolerance, the Douglas-Peucker (D-P) algorithm and the new algorithm have been programmed and tested using real contour data digitised in stream mode. *Figure 10 A—E* are the results. The number of points retained for each plot is about one fifth of the original one. It is obvious that some misrepresentations result from using the first two algorithms and the diagrams produced by both the new algorithm and the D–P algorithm are both with high fidelity to the original. This means that the dispersion from the original contour line is very small if both are overlaid together.

On the other hand, as far as the efficiency is concerned, the computing time required by the new algorithm is much less than that required by D–P algorithm. This is obvious because the new algorithm involves only a comparison and perhaps a conformal transformation in some portions of the line. But the D–P algorithm involves a lot of computation time in computing the distances repeatedly and also in comparison. In this particular example, the new algorithm takes 3972 micro seconds (ms) CPU time to compress the DCD, but D–P algorithm 4132 ms. (The others are: Nth points 3512 ms and Chord tolerance 3879 ms).

From the discussions above, the conclusion can be made that the new algorithm is more efficient than the D–P algorithm but the result is, like that from D–P algorithm, with high fidelity to the original line.

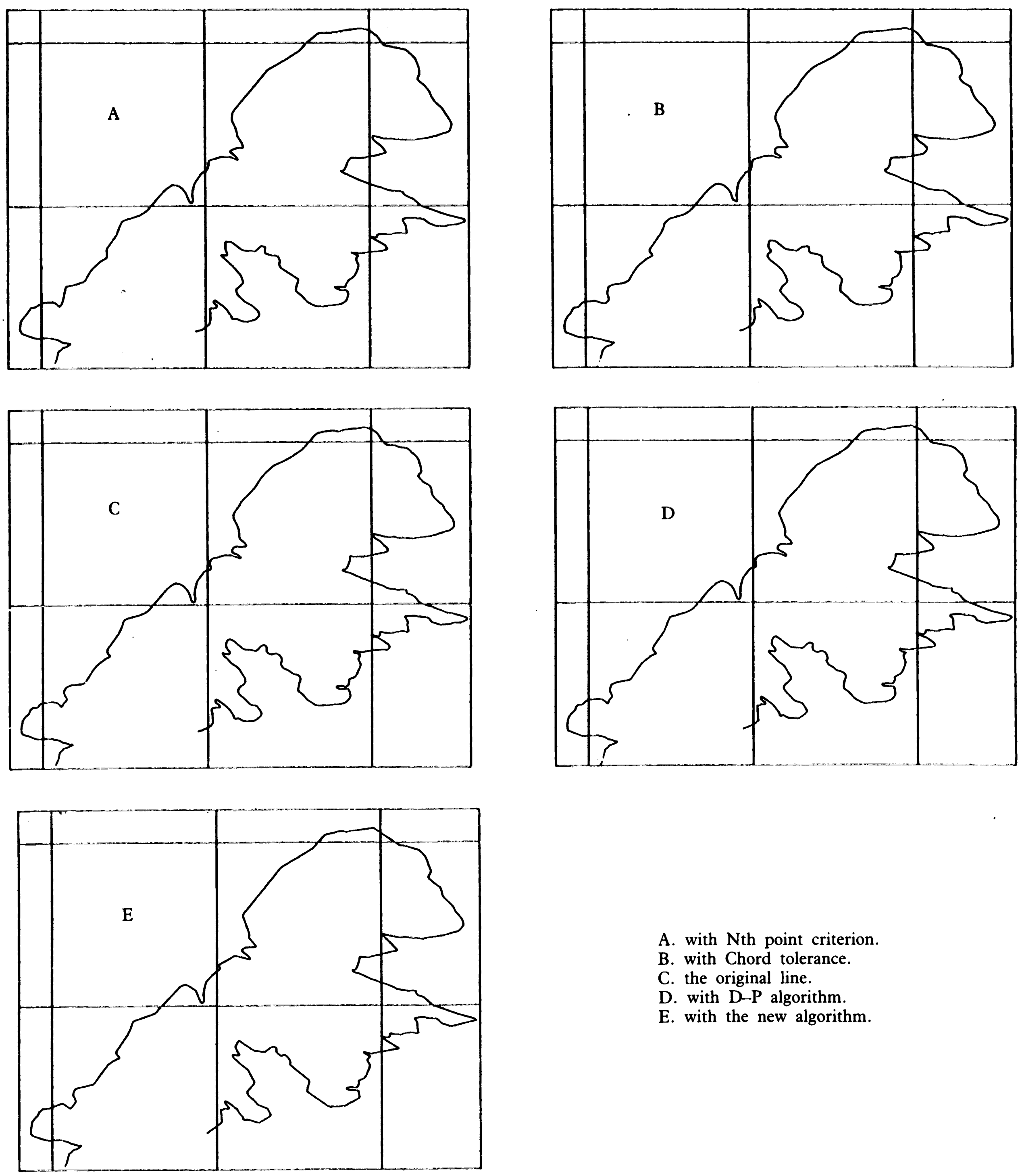

Figure 10. Diagrams produced with data from different algorithm, each with 1/5 points of the original.

REFERENCES

Douglas, D. & T. Peucker, 1973. Algorithms for the reduction of the number of points required to represent a digitised line or its caricature. *The Canadian Cartographer* Vol.10, pp. 112–122.

Experimental Cartography Unit, 1971. *Automated Cartography and Planning*. Architectural Press.

Lang, T. 1969. Rules for robot draughtsmen. *Geographical Magazine*, Vol. 42, pp. 50–51.

McMaster, R. 1983. A mathematical evaluation of simplification algorithms. *Auto Carto Six*, pp. 267–276.

ACKNOWLEDGEMENT

This paper was written during study in the University of Glasgow. The author would like to express his thanks to his supervisor, Prof. G. Petrie for his assistance in writing this paper, and to the University and the CVCP for their financial support of his study.