A PROGRAM PACKAGE FOR PACKING AND GENERALISING DIGITAL CARTOGRAPHIC DATA

PETR VANICEK
DAVID F. WOOLNOUGH

1972

GEODESY AND GEOMATICS ENGINEERING
UNB

TECHNICAL REPORT
NO. 23

PREFACE

In order to make our extensive series of technical reports more readily available, we have scanned the old master copies and produced electronic versions in Portable Document Format. The quality of the images varies depending on the quality of the originals. The images have not been converted to searchable text.

A PROGRAMME PACKAGE
FOR PACKING AND GENERALISING
DIGITAL CARTOGRAPHIC DATA

by

Petr Vaníček

and

David F. Woolnough

LIST OF CONTENTS

Acknowledgement

We would like to acknowledge the help and assistance of all those who contributed in any way to the compilation of this report. In particular, Mr. T. A. Porter, Department of Energy, Mines and Resources, Surveys and Mapping Branch, Ottawa; Dr. S. E. Masry, Department of Surveying Engineering, U.N.B.; Mr. L. Barton, Computing Centre, U.N.B. In addition, the senior author wishes to acknowledge the fact that he was introduced to the problem as well as encouraged to work on its mathematical formulation by Brig. L. Harris from Surveys and Mapping Branch, while employed by this institution.

1. INTRODUCTION

One of the main problems facing the field of automated cartography is the sheer volume of coordinates which have to be stored to allow an accurate representation of linear features on the final map. This problem becomes even more acute when the stored data are used to produce maps at reduced scale. Usually much fewer points are necessary to represent the curve to within the required accuracy, but reduction of the number of points inevitably leads to questions which raise doubts about mathematically rigid methods being able to reproduce, upon reduction, true cartographic shape and form.

In an attempt to reduce or pack the amount of input coordinate data of a curve without losing ultimate plotting accuracy, we are presenting a mathematical packing method which, given a specified tolerance (ε), i.e. the final plotting accuracy, would transform the input digitised coordinates into some other parameters of the curve. This would be done in such a way that the linear segments so produced would always be within a tube of width ε surrounding the original curve. The method results in a considerable reduction of the amount of input data without loss of accuracy in final plotting. In addition it is able to perform an automatic form of cartographic generalisation in which a given curve with many convolutions can, if the appropriate error parameter ε is introduced, be reduced to a simpler curve.

1.1 General Description of the Method

At present the mathematical calculations are performed by one Fortran IV programme, PACK, with two major subroutines, REDOUT and UPLOT. Also included are various plotting subroutines (see section 3) to enable the user to plot both the original and packed curves on either a 611 oscilloscope or a Calcomp drum plotter.

The input to the calculations is a set of x and y coordinates of the line to be packed, the input and output scales, the digitiser increment, δ, and the required final plotting accuracy ε.

The coefficients of pseudo-hyperbolae $y = \pm \frac{c_1 x + c_2}{x + c_3}$ are determined. By taking the average direction of the first three points in the stream of coordinates, further successive points are selected until they fail to lie within a tube $\pm 8\ \varepsilon$ wide. Using the beginning and end points for proper direction a rigorous check is made of points selected so they lie in a tube $\pm\ \varepsilon$ wide. The number of points selected is altered until this condition is met, at which time a segment length is computed.

The programme goes on to determine more segment lengths beyond the first by defining a pseudo-hyperbola with the vertex coinciding with the end of the last segment, and with axis oriented in the direction of the last line segment (figure 2.6). The next points in the coordinate stream are examined until one falls outside the defined pseudo-hyperbola, and then another line segment is identified whose end point is at the intersection of the stream of coordinates with the pseudo-hyperbola.

We choose the hyperbolic form since this is the approximate locus of the longest formable segments that would represent the curve with $\pm\varepsilon$ accuracy. The longer the segment lengths, the fewer the segments, and hence the greater the reduction in the amount of data stored. The details of this choice are given in section 2.4. A rigorous check is again made to ensure that all points lie within a tube $\pm\varepsilon$ wide around the segment. The segment's length is determined and signed positive if the line segment points above the axis, and negative below. This end point together with the preceding one determine the axis of a new hyperbola of the same family, and the process is repeated.

Thus the complete data packing consists in the production of coordinates (2 numbers per point) and interlying segment lengths (1 signed number only per segment). The points represented by two coordinates are referred to as corner points and the lengths as segments. This packed data is then stored. In order to obtain reduced coordinates from the packed data (which are not in "plottable" form) it is necessary to reverse the above procedure using the subroutine UPLOT.

This subroutine decodes into coordinate twotuples the packed data. The inputs to this routine are the two corner points, signed segment lengths and the tolerance ε. If only one segment is needed then two corner points alone are given and a line can be plotted. If more than one segment is needed it becomes necessary to take the signed segment lengths and compute coordinates. The coefficients of the pseudo hyperbola $y = \pm \frac{c_1 x + c_2}{x + c_3}$ are again calculated using the value ε. The initial segment is laid out in an east-west direction and directions of subsequent segments are related to it. The routine determines coordinates of intersections of line segments with the hyperbola. This is done in a local set of coordinates where the hyperbola vertex is related to the terminal point of the previous segment and its axis is rotated to the direction of the previous segment.

Using the final corner point these local coordinates are rotated and stretched so that the line segments are in the required direction and at the required scale. The output coordinates from UPLOT are then suitable for plotting.

The system documented here contains plotting routines for the University of New Brunswick's IBM 370 computer plotting system. After data packing and decoding, the original and packed curves are plotted out and hardcopy is obtained from the 611 oscilloscope. A "packing factor" is then calculated, being the ratio of the input number of points to the packed number of points. This is then printed along with details of input and output scales, and error values.

2. THE MATHEMATICAL BASIS OF THE METHOD

2.1 Digitized Curve

Let us consider an open, continuous, smooth curve C extending between initial and final points $\vec{r}_1 = (x_1, y_1)$, $\vec{r}_N = (x_N, y_N)$ with $\vec{r}_i$ denoting the radius vectors. We shall call the digitized image of C, C*, as a series of points $\vec{r}^*_i = (x^*_i, y^*_i)$, i = 1, 2, ..., N, representing C in the form of a set of isolated points. These points coincide with appropriate intersection points of a δ-square-grid, whose dimension δ is given by the last retained binary (decimal) place - see figure 2.1.

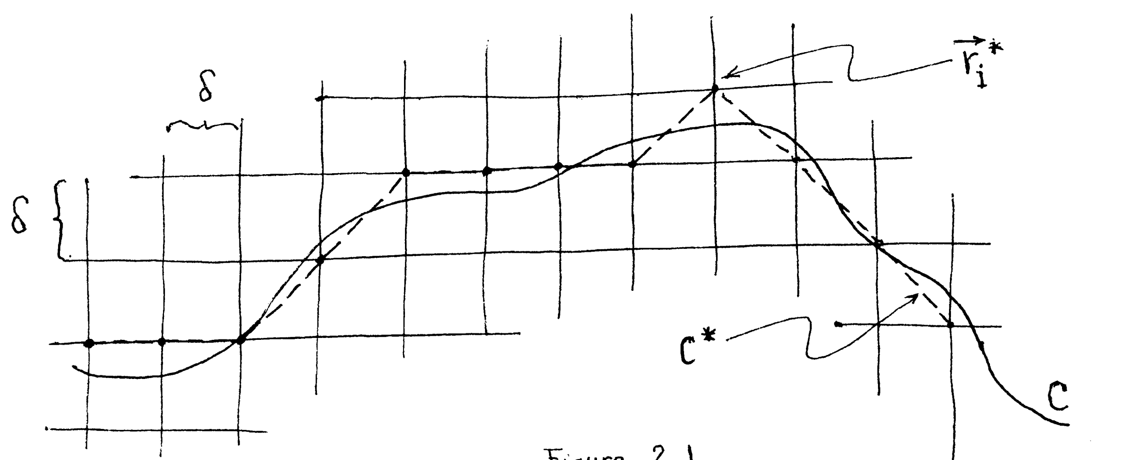


Figure 2.1

2.2 Representation of a Curve

We shall say that a curve C' represents C with a precision ε (strictly 1/ε) if and only if every point $\vec{r}' \varepsilon C'$ lies within the **ε** environment of at least one point $\vec{r} \varepsilon C$ and C' is continuous.

Note that if we regard the digitized curve C* as piece-wise linear curve with the points of discontinuous first derivative coinciding with $\vec{r}^*$'s, we can say that C^* represents C with precision δ. In the forthcoming development we shall always assume $\varepsilon>\delta$ and shall refer to the representation of C as actually the representation of C^*.

2.3 Purpose of Coding

The number of digitized points $\vec{r}^*$, as usually supplied by a digitizer, is generally unnecessarily large to represent C with the required precision ε. If we intend to store the digitized image C^* on any kind of medium, we are evidently interested in keeping the amount of retained information to a minimum. The problem becomes more pressing whenever we store an excessive number of such curves as is the case with cartography, pictorial images or many other practical digitized images.

Thus the ultimate aim of an optimum coding will be to replace C^* by such a curve C' which:- (i) is continuous; (ii) has minimum number of representative points $\vec{r}'$ expressed by minimum necessary number of parameters; (iii) represents C^* with required precision ε.

Another requirement, particular to cartographic applications, is that C' representing a smooth curve should "look smooth" to the eye. Since this requirement does not lend itself to an easy mathematical formulation, it will be assumed that ε can be selected in such a way that C' "looks smooth" enough when plotted. In other words, we assume that if ε corresponds to the graphical precision of plotting (usually about .1 mm) it will take care of this aspect automatically.

Our approach will be based on the piece-wise linear representation assuming thus availability of a linear plotter only. If a more flexible plotter, that will plot circular or parabolic arcs as well, is available further reduction in the necessary number of parameters may be achieved either by simply increasing the value of ε or through adding some qualifying criteria to the technique. This, however, will not be the aim of this report but a subject for future development.

2.4 Maximum Allowable Length of Linear Segments

It is obvious that the maximum length of a linear segment, that is to replace the original curved segment with precision ε, is inversely proportional to the curvature of the original segment. The larger the curvature, the shorter the linear segment and vice versa. The following formula can be deduced from figure 2.2 for the relationship of the linear segment $\overline{\Delta S}$ and the linear deviation dR of the linear and curved segment ΔS

$$\overline{\Delta S} = \sqrt{(8RdR-4dR^2)}\,. \qquad (1)$$

Here R is the local radius of curvature. According to 2.2, we can allow dR to become as large as ε for the maximum segment $\overline{\Delta S}_{max}$. We can hence write:

$$\overline{\Delta S}_{max} = \sqrt{(8R\varepsilon-4\varepsilon^2)}\,. \qquad (2)$$

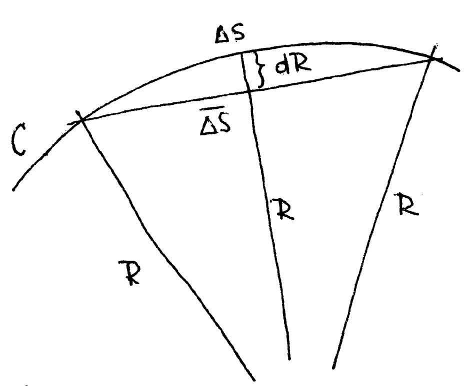


Figure 2.2

To illustrate the quantities we are dealing with we can draw a table showing the relationship of R and $\overline{\Delta S}_{max}$ for ε = .1mm:

R[mm]	0.2	0.3	0.4	0.5	1	5	10	100
$\overline{\Delta S}_{max}$[mm]	0.40	0.49	0.57	0.63	0.90	2.00	2.83	8.95

Table 2.1

Considering a digitized curve C*, 100 mm long, consisting of 4000 points $\vec{r^*}$ .025 mm apart, we get the minimum number of linear segments necessary to represent C* with precision ε =.1mm, 111, 50, 35, 11 corresponding to mean radii of curvature of 1, 5, 10, 100 mm respectively. Hence, defining the packing factor as $\frac{\text{no. of input points}}{\text{no. of output points}}$ we get for the packing factor between C* and C': 36, 80, 114, 364 respectively.

2.5 Reduction in the Necessary Number of Parameters

Having established the maximum spacing of the representative points $\vec{r'}\varepsilon C'$ (as related to the local radius of curvature) we can show that it is not necessary to identify each of those points by a pair of coordinates. For this purpose, let us transform the original coordinates x_i, y_i of a point $\vec{r'}_{i+1}$ into the local coordinates $\overline{\Delta S}_i$, α_i (see figure 2.3).

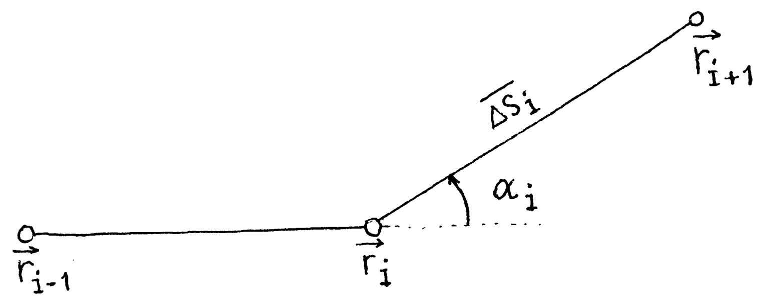


Figure 2.3

If $\overline{\Delta S}_i$ is made a function of α_i

$$\overline{\Delta S}_i = f(\alpha_i), \quad \alpha_i = f^{-1}\,(\overline{\Delta S}_i) \tag{3}$$

we do not have to retain both $\overline{\Delta S}_i$ and α_i since the knowledge of one of these is sufficient to furnish us with the other coordinate. Thus, providing the relationship $\overline{\Delta S} = f(\alpha)$ is established, a curve can be represented by a stream of single parameters, rather than a stream of pairs of coordinates, which may lead to a considerable saving of storage medium.

The question remains as how to select the function f to satisfy the other requirements. We are going to show that the selection can be done in such a way that f is the approximate locus L of all the $\overline{\Delta S}_{max}$.

Providing the curve C has approximately the same curvature in the vicinity of $\vec{r}_i'$, we can, according to figure 2.4 write:

$$R \cos \frac{\alpha}{2} + dR \doteq R. \tag{4}$$

Substituting again ε for dR and $(\overline{\Delta S}^2_{max} + 4\varepsilon^2)/(8\varepsilon)$ for R (from eq. 2) we get:

$$\cos \frac{\alpha}{2} \doteq 1 - \frac{8\varepsilon^2}{\overline{\Delta S}^2_{max} + 4\varepsilon^2} \tag{5}$$

and after some development, using trig. identity $\tan^2 \frac{\alpha}{4} = \frac{1-\cos(\alpha/2)}{1+\cos(\alpha/2)}$, we obtain

$$\alpha \doteq 4 \text{ arctg } \frac{2\varepsilon}{\overline{\Delta S}_{max}} \quad . \tag{6}$$

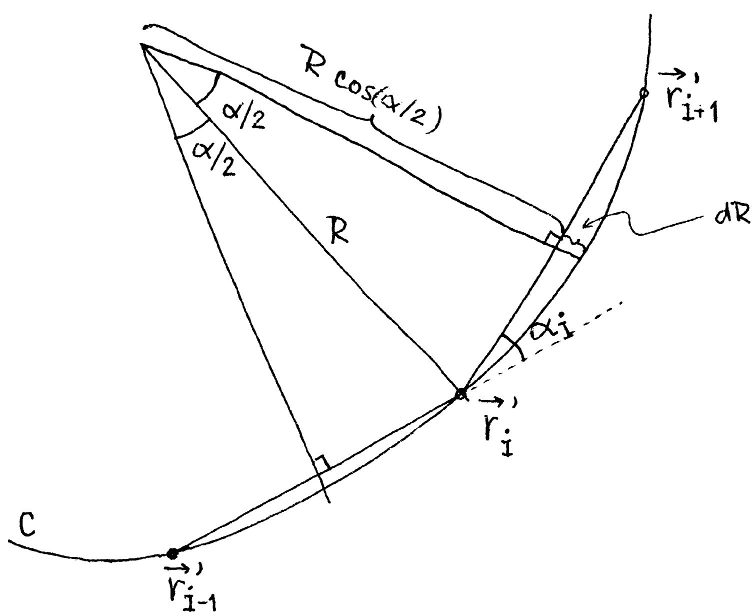


Figure 2.4

The derivation of eq. (6) is based on the assumption of almost uniform curvature in the vicinity of $\vec{r}'_i$ which may or may not be fully satisfied. In any case, it can be regarded as an approximate formula and no considerable damage will be done if we replace it by another yet approximate relationship more convenient for numerical treatment. The only result of such approximation will be that the ΔS's will not be the absolute allowable maximum and therefore the reduction will not be the absolute maximum. This should not be unduly worrying since we shall, in practice, be dealing with C^* instead of C and have therefore to expect some irregularities due to the limited precision in digitizing that will "spoil" the smoothness of C.

To make the development of an approximate equation of the locus easier, let us introduce a local right-handed coordinate system ξ, η - see figure 2.5.

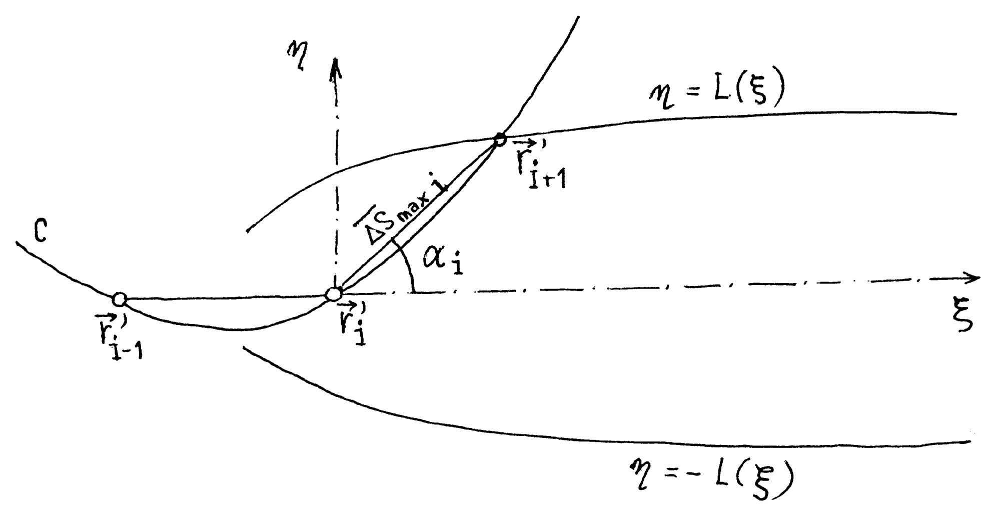


Figure 2.5

Transforming α, $\overline{\Delta S}_{max}$ to ξ,η we get the equation of the locus (6) as follows:

$$\alpha = \operatorname{arctg} \eta/\xi \doteq 4 \operatorname{arctg} \frac{2\varepsilon}{\sqrt{(\xi^2+\eta^2)}} \; . \tag{7}$$

Eq. (7) can be rewritten as

$$\operatorname{arccotg} \xi/\eta \doteq 4 \operatorname{arccotg} \frac{\sqrt{(\xi^2+\eta^2)}}{2\varepsilon} \; .$$

Considering the trig. identity

$$\operatorname{arccotg} x = \arccos \frac{x}{\sqrt{(1+x^2)}}$$

we obtain

$$\arccos \frac{\xi}{\sqrt{(\xi^2+\eta^2)}} \doteq 4 \arccos \sqrt{\frac{\xi^2+\eta^2}{\xi^2+\eta^2+4\varepsilon^2}} \tag{8}$$

and
$$\xi/\sqrt{(\xi^2+\eta^2)} = \cos\left(4\arccos \sqrt{\frac{\xi^2+\eta^2}{\xi^2+\eta^2+4\varepsilon^2}}\right) = \tau_4\left(\sqrt{\frac{\xi^2+\eta^2}{\xi^2+\eta^2+4\varepsilon^2}}\right) \tag{9}$$

Here τ_4 is the Tchebyshev's polynomial of 4-th order (see, for instance, Ralston, 1965). After rewriting τ_4 in the form of power series (polynomial of 4th order in $\sqrt{\frac{\xi^2+\eta^2}{\xi^2+\eta^2+4\varepsilon^2}}$) we obtain the final expression as a mixed algebraic polynomial of 20th order in ξ and η. Such a polynomial would not obviously be convenient for numerical computation either and has therefore to be approximated by simpler formula.

It can be shown, using numerical evaluation of eq. (6) that each branch of L may be approximated by a hyperbola:

$$L'(\xi) \equiv \eta = \frac{c_1\xi + c_2}{\xi + c_3} \quad . \tag{10}$$

The coefficients c_1, c_2, c_3 can be determined, for instance, by using the least-squares technique for the whole curve or more simply (and less precisely) on the basis of three common points.

In our case, the three points were selected in such a way as to provide an easy computation. Using formula (6) one gets

$$\overline{\Delta S}_{max} = 2\varepsilon/\mathrm{tg}\,\frac{\alpha}{4} \; .$$

On the other hand: $\overline{\Delta S}_{max} = \xi^2+\eta^2$. Hence for $\alpha = \frac{\pi}{2}$ we get $\xi = 0$ and $\eta = \overline{\Delta S}_{max} = 2\varepsilon/\mathrm{tg}\,\frac{\pi}{8} \doteq 4.828\varepsilon$. For $\alpha = \frac{\pi}{4}$ we have $\xi = \eta = \overline{\Delta S}_{max}/\sqrt{2} = 2\varepsilon/(\sqrt{2}\ \mathrm{tg}\,\frac{\pi}{16}) \doteq 7.110\varepsilon$. The third point was selected for $\xi = 1000\varepsilon$. Developing formula (7) into power series in $2\varepsilon/\sqrt{(\xi^2+\eta^2)} = q$ one gets $\eta/\xi = 4q$ + terms of 4th and higher order in q. Thus, for large ξ:

$$\eta \doteq \xi \; \frac{8\varepsilon}{\sqrt{(\xi^2+\eta^2)}} = \frac{8\varepsilon}{\sqrt{(1+\eta^2/\xi^2)}} = 8\varepsilon\left(1 - \frac{1}{2}\frac{\eta^2}{\xi^2} + \ldots\right)$$

which tends to 8ε for growing ξ. Hence, we shall not make any serious mistake taking $\eta = 8\varepsilon$ for $\xi = 1000\varepsilon$.

Using the selected three points one obtains:

$$c_1 = 8.00895\varepsilon,\; c_2 = 13.615\varepsilon^2,\; c_3 = 2.82\varepsilon \; . \tag{11}$$

These values provide us with L' good enough for all practical purposes. The shape of L' can be seen on figure 2.6.

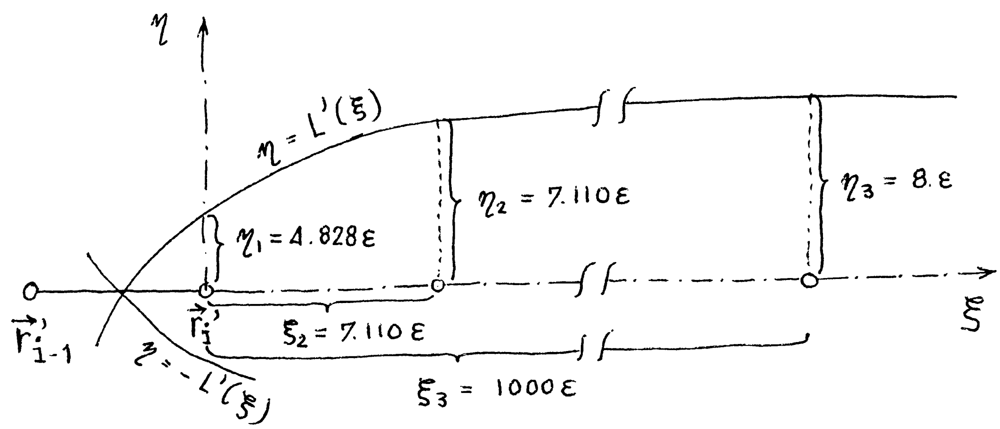


Figure 2.6.

2.6 Determination of the "next" Representative Point

Once we have decided to adopt a certain L', the determination of the "next" point becomes easy. Having two consecutive representative points $\vec{r}'_{i-1}$, $\vec{r}'_i$ described by their pairs of coordinates (x_{i-1}, y_{i-1}), (x_i, y_i), we can transform all the subsequent points belonging to C* into the local ξ,η system of the point $\vec{r}'_i$. If $\vec{r}^*=(x,y)$ is a running point from C* we have for its local coordinates:

$$\begin{aligned} \xi &= T_1(x-x_i) - T_2(y-y_i) \\ \eta &= T_2(x-x_i) + T_1(y-y_i) \end{aligned} \qquad (12)$$

where $T_1 = (x_i - x_{i-1})/\sqrt{[(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2]}$

$$T_2 = -(y_i - y_{i-1})/\sqrt{[(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2]}. \quad (13)$$

Thus, we can take the sequence of all $\vec{r}^*$'s following $\vec{r}_i'$, compute their ξ and η coordinates and decide whether each of them lies either within or outside the area bound by $\pm$ L'(ξ). If the inequality

$$|\eta| \leqslant |L'(\xi)| \quad (14)$$

for a point (ξ,η) is satisfied, the point lies in the area and vice versa.

Hence, we eventually find a pair of running points, $\vec{r}_k^*$ and $\vec{r}_{k+1}^*$ say, of which the first lies within and the second outside the area. If the whole "rest of C*" lies within the area, then the end point is taken instead of $\vec{r}_{i+1}'$, its coordinates are retained and the segment is not computed because it is not needed. The point $\vec{r}_i'$ is declared a "corner point" (see later) and retained by coordinates. Otherwise the "next" representative point $\vec{r}_{i+1}'$ is the point, where L'(ξ) intersects the straight line connecting $\vec{r}_k^*$ with $\vec{r}_{k+1}^*$ (see figure 2.7).

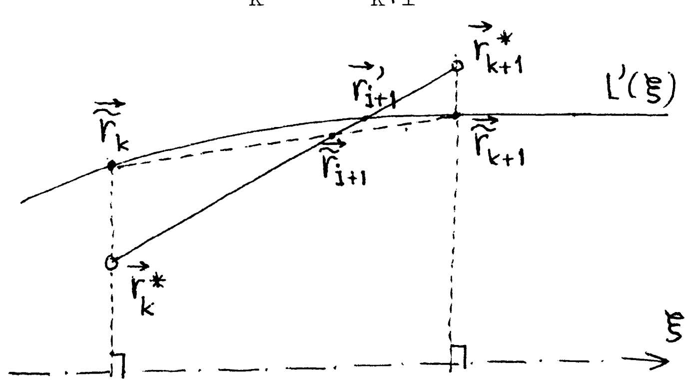


Figure 2.7

To establish $\vec{r}'_{i+1}$ with the required precision, any numerical method may be used. In our approach we have opted for the "chord method" which is likely to converge fast since L' is ever-increasing and relatively flat. Because $\pm$ L' is concave, the approximation of $\vec{r}'_{i+1}$, $\vec{\tilde{r}}_{i+1}$ say, will always lie between $\vec{r}^*_k$, $\vec{r}'_{i+1}$ and $\vec{\tilde{r}}_{i+1}$ can be thus taken instead of $\vec{r}^*_k$ for the next iteration. The chord method, therefore, will furnish a succession of points on $\vec{r}^*_k$, $\vec{r}'_{i+1}$ converging to $\vec{r}'_{i+1}$ the faster the further away we are from $\vec{r}'_i$.

Providing the point $\vec{r}'_{i+1}$, selected on the basis of the last iteration, is made to lie on $\pm$ L'(ξ) in a relatively narrow environment of $\vec{r}'_{i+1}$ - which can be achieved by taking $\eta_{i+1} = L'(\xi_{i+1})$ for ξ_{i+1} belonging to the last iteration $\vec{r}_{i+1}$ - it can be as much away from $\vec{r}^*_k$, $\vec{r}^*_{k+1}$ as $\pm\ \delta/2$. C* represents C with precision $\delta/2$ so that it would not make sense to insist on the representative points to perform any better fit to C* than $\pm\ \delta/2$.

Once ξ_{i+1}, η_{i+1} are obtained, we can compute the quasi-maximum segment $\overline{\Delta S'_i}$, determining the position of $\vec{r}'_{i+1}$ uniquely with respect to $\vec{r}'_{i-1}$, $\vec{r}'_i$, from following formula:

$$\overline{\Delta S'_i} = \text{sign}\ (\eta_{i+1})\surd(\xi^2_{i+1}+\eta^2_{i+1})\ . \tag{15}$$

The coordinates x_{i+1}, y_{i+1} of $\vec{r}'_{i+1}$, necessary for locating the next representative point $\vec{r}'_{i+2}$, are obtained by applying the transformation inverse to (12):

$$x_{i+1} = x_i + T_1 \xi_{i+1} + T_2 \eta_{i+1}$$
$$y_{i+1} = y_{i+1} - T_2 \xi_{i+1} + T_1 \eta_{i+1} \,. \qquad (16)$$

Note that from the point of view of error propagation, the segments $\overline{\Delta S}'$ can be considered as errorless, i.e. if we do not commit any error in the decoding process we would end up with the curve C' representing C* in the manner described above. More will be said about this subject later.

2.7 Initiation of the Process

The process described in the previous paragraph is able to determine only the position of a "next" point with the assumption that the two immediately preceding points from C' are already known. Thus, it cannot obviously be applied at the beginning and we have to establish the first segment by using an altogether different approach, i.e. we have to initiate the process somehow.

The initiation should provide us with as long a segment as can be achieved for the actual C* and ε. The first reason is that we try to represent C* by as few points as possible. The second, more important reason, is that the technique using the locus L' is based on the idea that both segments $\overline{\Delta S}'_{i-1}$ and $\overline{\Delta S}'_i$ are about the same, since C is assumed to have in the vicinity of r'_i an approximately uniform curvature. If we chose the first segment too short, the second shall be too long and vice versa. The following segments would be influenced accordingly.

An iterative approach was hence devised that selects for $\vec{r}'_2$ ($\vec{r}'_1$ remaining equal to $\vec{r}^*_1$) such a point $\vec{r}^*_k$ which

(i) is furthest away from $\vec{r}^*_1$;

(ii) yet still all the points $\vec{r}^*_j$, j<k, lie no further than $\pm\ \varepsilon$ away from $\vec{r}^*_i\vec{r}^*_k \equiv p_k$.

The equation of the line p_k can be written as

$$p_k \equiv A_{ok} + A_{1k}\,x + A_{2k}y = 0 \tag{17}$$

where

$$\begin{aligned} A_{ok} &= y_1/(y_k-y_1) - x_1/(x_k-x_1) \\ A_{1k} &= 1/(x_k-x_1) \\ A_{2k} &= -1/(y_k-y_1)\,. \end{aligned} \tag{18}$$

The distance d of a running point $\vec{r^*} = (x,y)$ from p_k is given by (see, for instance Bush and Obreanu, 1965):

$$d = \frac{|A_o + A_1x + A_2y|}{\sqrt{(A_1^2+A_2^2)}} = |B_o + B_1x + B_2y|\,. \tag{19}$$

This distance must, for all $\vec{r}^*_j$, j<k, be smaller than ε. Thus, after some development we may write following inequality to be satisfied for all the $\vec{r}^*_j$:

$$(A_{1k}(y_i-y_j) + A_{2k}(x_1-x_j))^2 \lessgtr (\overline{\Delta S}\varepsilon)^2 \tag{20}$$

where

$$\overline{\Delta S}^2 = (x_k-x_1)^2 + (y_k-y_1)^2\,. \tag{21}$$

We can notice that evaluation of (20) does require adding, subtracting and multiplication only and is, therefore, quite fast for computation. The index k can be iterated for so long until (20) becomes satisfied for k=n and fails for k=n+1. Then $\vec{r}^*_n = \vec{r}'_i$. If (20) is satisfied even for the end point of the curve then only the first and the end points are retained.

2.8 ε Test

There is one more point, within the process based on locus, that must not escape our attention. It is evident that finding $\vec{r}'_{i+1}$ as the intersection of L' and C* does not automatically ensure that all the points $\vec{r}^* \varepsilon C^*$ between $\vec{r}'_i$ and $\vec{r}'_{i+1}$ lie within the distance of ε from C' (represented here by the straight line joining $\vec{r}'_i$ and $\vec{r}'_{i+1}$). It is conceivable that if the curvature of C in the area changes rapidly the assumption for the method does not hold any more and the locus L' looses its fundamental meaning. In such a case, we have to declare $\vec{r}'_i$ the "corner" point and start again with initiating the process in exactly the same manner as described in section 2.7.

To check whether the $\pm\varepsilon$ belt around C' contains all the points $\vec{r}^* \varepsilon (\vec{r}'_i , \vec{r}'_{i+1})$ the inequality (20) can be used. When substituting (0,0) for (x_1, y_1), (ξ,η) for (x_k, y_k) and (ξ_j, η_j) for (x_j, y_j) the inequality simplifies considerably and we get:

$$|\eta\xi_j - \xi\eta_j| \leq |\overline{\Delta S'_i}|\varepsilon \quad . \qquad (22)$$

2.9 Decoding of the Curve C'

The described method supplies us with a coded version of C'. The C' is presented as a stream of pairs of coordinates (x, y), belonging to the corner points, with varying number of sections $\overline{\Delta S}'$ sandwiched in between any two adjacent coordinate pairs. The decoding will be necessary to apply always on the succession of segment between two adjacent corner points. Its goal will be to attach a pair of coordinates (in the x,y system) to the end of each segment, i.e. to each of the representative points.

Let us consider such a "smooth" piece C_1' of C', coded as

$$C_1' \equiv \{x_1, y_1, \overline{\Delta S}_1', \overline{\Delta S}'_2, \ldots, \overline{\Delta S}'_{n-1}, x_n, y_n\} \subset C'. \tag{23}$$

It is not difficult to see that each $\overline{\Delta S}_i'$ can be split into the two coordinate increments ξ_{i+1}, η_{i+1} related, as in section 2.5, to the local right-handed coordinate system originating in $\vec{r}_i'$ with $\xi_{i-1} = -|\overline{\Delta S}_i'|$. To split it, we can use the formula (10) in conjunction with the Pythagoras law:

$$\xi_{i+1}^2 + \eta_{i+1}^2 = \overline{\Delta S}_i'^2 . \tag{24}$$

In (24) express $\overline{\Delta S}_i'^2 - \xi_{i+1}^2$ as $(|\overline{\Delta S}_i'| - \xi_{i+1})(|\overline{\Delta S}_i'| + \xi_{i+1})$ and substitute for η_{i+1} from eq. (10). It then follows that

$$\xi_{i+1} = |\overline{\Delta S}_i'| - \frac{(c_1\xi_{i+1} + c_2)^2}{(\xi_{i+1} + |\overline{\Delta S}_i'|)(\xi_{i+1} + c_3)^2} = |\overline{\Delta S}_i'| - Q \tag{25}$$

with Q being a function of $|\overline{\Delta S}'_i|$ and ξ_{i+1} only.

This equation can be regarded as a recurrence formula for ξ_{i+1}, and ξ_{i+1} can be determined by an iterative process using (25). The convergence of such a process is ensured and is the faster the larger is $|\overline{\Delta S}'_i|$.

Once ξ_{i+1} is established with sufficient precision, η_{i+1} can be computed from eq. (24) taking into account the sign of $\overline{\Delta S}'_i$. We have

$$\eta_{i+1} = \mathrm{sign}(\overline{\Delta S}'_i)\ \sqrt{(\overline{\Delta S}^2_i - \xi^2_{i+1})}\ . \qquad (26)$$

The coordinate increments ξ_{i+1}, η_{i+1} can then be transformed into the reference coordinate system x', y' for which we can take the local system of $\vec{r}'_1$, i.e. we define

$$x'_1 = 0,\ y'_1 = 0,\ x'_2 = |\overline{\Delta S}'_1|,\ y'_2 = 0\ . \qquad (27)$$

Thus, we get

$$\begin{aligned} x'_{i+1} &= x'_i + T'_1\ \xi_{i+1} - T'_2\ \eta_{i+1} \\ y'_{i+1} &= y'_i + T'_2\ \xi_{i+1} + T'_1\ \eta_{i+1} \end{aligned} \qquad (28)$$

where

$$\begin{aligned} T'_1 &= (x'_i - x'_{i-1})/|\overline{\Delta S}'_{i-1}| \\ T'_2 &= (y'_i - y'_{i-1})/|\overline{\Delta S}'_{i-1}|\ . \end{aligned} \qquad (29)$$

Continuing in the described way, we eventually end up with the coordinates of the last point, x'_n, y'_n. The final coordinates (x_i, y_i), i=1, 2, ..., n, can be obtained by another transformation yet:

$$x_i = x_1 + T_1'' x_i' - T_2'' y_i'$$
$$y_i = y_1 + T_2'' x_i' + T_1'' y_i' \qquad (30)$$

where

$$T_1'' = (y_n'(y_n-y_1) + x_n'(x_n-x_1))/D$$
$$T_2'' = (x_n'(y_n-y_1) - y_n'(x_n-x_1))/D \qquad (31)$$
$$D = \sqrt{}\,(x_n' + y_n')\,.$$

Three things should be noted here:

(i) For the transformation (30, 31) to work, C' must be an open curve - as required in section 2.1.

(ii) Should the need arise to transform the coded C' into a new coordinate system by a conformal transformation, only the corner points have to be transformed. The linear segments may be regarded as "shape parameters" which are not liable to change under any conformal transformation.

(iii) The value of ε, used for determining the coefficients c_1, c_2, c_3 when coding the curve, must be used for computing the coefficients c_1, c_2, c_3 in eqns. (25, 26) necessary for decoding the C'.

2.10 Propagation of errors and required precision

We require the maximum error in the position of any point of C' to be smaller than ε. Due to the conformal transformation represented by eq. (30), the error in the end point as well as the error in the first point will vanish and we can expect the maximum error to occur for $\vec{r}'_{n/2}$. We shall therefore investigate what precision is required in iterating the ξ's from eq. (25) to obtain the position error of $\vec{r}'_{n/2}$ smaller than ε.

Denoting the errors in coordinates, x_i, y_i by δx_i, δy_i we define the position error δ_i as

$$|\delta_i| = \sqrt{(\delta x_i^2 + \delta y_i^2)} \tag{32}$$

and the requirement is that

$$|\delta_{n/2}| < \varepsilon . \tag{33}$$

Assuming the transformation coefficients T''_1, T''_2 in eq. (30) to be smaller or equal to 1, i.e. assuming that the segments are expressed in equal or larger scale than the coordinates x, y, we get the most pessimistic estimate of δx, δy from following formulae:

$$|\delta x| < \sqrt{2}|\delta'| \qquad |\delta y| < \sqrt{2}|\delta'| \tag{34}$$

where δ' is the error in either x' or y'. Hence, we may write:

$$|\delta_i| \leq \sqrt{(2\delta_i'^2 + 2\delta_i'^2)} = 2|\delta_i'| . \tag{35}$$

On the other hand, x'_k, y'_k can be expressed as

$$x'_k = \sum_{i=1}^{k} \Delta x'_i , \qquad y'_k = \sum_{i=1}^{k} \Delta y'_i \tag{36}$$

where $\Delta x'_i$, $\Delta y'_i$ are given by eq. (28).

Thus, denoting by $\delta\Delta_i$ the error in either $\Delta x_i'$ or $\Delta y_i'$ we get

$$|\delta_k'| \doteq \sqrt{\sum_{i=1}^{k} \delta\Delta_i^2} \tag{37}$$

providing the individual $\delta\Delta_i^2$ are distributed more or less at random.

Taking all the $\delta\Delta_i$ equal to $\delta\Delta$ we end up with the expression

$$|\delta_k'| \doteq \sqrt{k}\ |\delta\Delta| \tag{38}$$

and

$$|\delta_k| \leqslant 2\sqrt{k}|\delta\Delta| . \tag{39}$$

The transformation coefficients in eq. (28) are again smaller or equal to 1. Hence the combined influence of the errors $\delta\xi$, $\delta\eta$, in ξ and η, is at most 2-times larger than that of the individual error $\delta\xi$:

$$|\delta\Delta| \leqslant \sqrt{2}|\delta\xi| \tag{40}$$

and substitution of (40) into (39) yeilds:

$$|\delta_k| \leqslant 2\sqrt{(2k)}|\delta\xi| . \tag{41}$$

The criterion for $\delta\xi$ can thus be set up using eq. (33):

$$|\delta\xi| \leqslant \frac{\varepsilon}{2\sqrt{(2n/2)}} = \frac{\varepsilon}{2\sqrt{n}} . \tag{42}$$

In order to achieve the required precision in position, each ξ, iterated from eq. (25), must be determined with a precision better than $\varepsilon/(2\sqrt{n})$.

How do we recognize that the required precision has been reached during the process of iteration? For this purpose, let us introduce a magnitude δs given by:

$$\delta s = |\overline{\Delta S'}| - \sqrt{(\xi^2+\eta^2(\xi))} \tag{43}$$

where $\eta(\xi)$ is prescribed by eq. (10). Obviously $|\delta s|>|\delta\xi|$ and $\delta\xi$ in eq. (42) can be replaced by δs. From the computing point of view it is convenient though to evaluate δs from an approximate equation:

$$\delta s \doteq (\xi^2+(\frac{c_1\xi+c_2}{\xi+c_3})^2 - \overline{\Delta S'}^2)/(2\overline{\Delta S'}) \tag{44}$$

and the final criterion which must be satisfied for the last iteration of ξ reads:

$$|\xi^2 +(\frac{c_1\xi+c_2}{\xi+c_3})^2 - \overline{\Delta S'}^2| < \frac{|\overline{\Delta S'}|\varepsilon}{\sqrt{n}} . \tag{45}$$

3. PROGRAMMES AND PARAMETERS

The programmes and subroutines presently form an integrated packing and plotting package, and are dimensioned to accept 5000 points per curve. They are written in Fortran IV language and have been tested on the University of New Brunswick's IBM 370/155 computer, and the Univac 1108 and PDP-10 computer of the Department of Energy, Mines and Resources, Ottawa. All times and storage refer to the 370 system using the level G compiler. To pack one line with 375 points in it (the example shown in Appendix A) required 0.2 seconds. The plotting routines on the 611 oscilloscope required a further 5 seconds. The storage necessary depends on the number of points per curve, and is presently 129,264 bytes. These requirements include the university's system generated plotting routines. The results and restrictions of the package in its present form are listed in section 4. This section deals only with the programming requirements of each routine.

3.1 Main Programme PACK

Storage: 126,032 bytes in single precision

Subroutines called: REDOUT, UPLOT, AREA, GRID, SETPLT, NOWPLT, ENDPLT, PRNTCH

Input Parameters: Input parameters are given as though the data were on punched cards.

Card 1: Format 2F4.0. The allowable plotting error at reduced scale in micrometres, ERR. This is followed by the least count of the digitiser in micrometres, DELTA.

Card 2: Format I4. The number of points in the line whose coordinates follow, N. This can, with a minor change, be left open for on-line work.

Card 3: Format 2I10. The denominator of the scale of the input data, ISD. This is followed by the denominator of the scale of the output data, IOSD.

It should be pointed out at this point that it is perfectly possible to stop the technique after the packing process. The programme subroutine UPLOT, which reduces the packed data to the scale required for output can be called at a later date. In this way only packed parameters are stored, thus cutting down on storage space. At the same time the option exists, just before final plotting, to change the scale of the output. The present version is set up for simultaneous packing and plotting.

Rest of cards: Format 5 (F7.0, 1X, F7.0, 1X). The x and y coordinates of the points on the line, 5 points per card, as shown in figure 3.1. It follows that these should be N/5 data cards with coordinates.

CARD	X	Y	X	Y	X	Y	X	Y	X	Y
0514	0012665	0026915	0012801	0026855	0012933	0026839	0013061	0026803	0013173	0026771
0515	0013299	0026727	0013401	0026681	0013519	0026623	0013637	0026583	0013741	0026531
0516	0013865	0026473	0013997	0026427	0014087	0026359	0014207	0026325	0014333	0026265
0517	0014455	0026191	0014575	0026127	0014683	0026049	0014767	0025955	0014871	0025889
0518	0014975	0025835	0015093	0025739	0015209	0025707	0015327	0025659	0015471	0025559
0519	0015561	0025425	0015479	0025269	0015375	0025163	0015197	0025061	0015053	0025011
0520	0014995	0025111	0014955	0025041	0014749	0024959	0014643	0024901	0014495	0024861
0521	0014359	0024779	0014213	0024657	0014151	0024597	0014049	0024515	0013997	0024609
0522	0013999	0024765	0013929	0024939	0013909	0024991	0013869	0025111	0013783	0025231
0523	0013683	0025345	0013605	0025521	0013455	0025545	0013289	0025583	0013191	0025567
0524	0013103	0025581	0013093	0025707	0012993	0025837	0012915	0025945	0012757	0026023
0525	0012557	0026007	0012411	0025905	0012261	0025801	0012109	0025709	0012059	0025581
0526	0011951	0025627	0011783	0025485	0011707	0025361	0011699	0025239	0011797	0025283
0527	0011397	0025311	0011975	0025299	0012085	0025305	0012221	0025349	0012351	0025413
0528	0012445	0025513	0012527	0025613	0012651	0025669	0012743	0025683	0012827	0025607
0529	0012871	0025435	0012961	0025383	0012813	0025269	0012755	0025217	0012671	0025189
0530	0012599	0025141	0012675	0025127	0012755	0025129	0012831	0025151	0012923	0025195
0531	0013021	0025263	0013133	0025291	0013247	0025261	0013341	0025217	0013389	0025113
0532	0013439	0025021	0013495	0024933	0013579	0024825	0013505	0024799	0013387	0024805
0533	0013255	0024805	0013139	0024757	0013069	0024651	0012953	0024587	0012939	0024481
0534	0012837	0024375	0012939	0024293	0012947	0024203	0012819	0024125	0012725	0024019
0535	0012649	0023943	0012539	0023859	0012689	0023853	0012759	0023931	0012819	0024035
0536	0012967	0024087	0012975	0024117	0013067	0024145	0013121	0024235	0013145	0024373
0537	0013159	0024473	0013277	0024533	0013359	0024507	0013419	0024433	0013471	0024355
0538	0013511	0024275	0013549	0024195	0013587	0024107	0013627	0024045	0013699	0023981
0539	0013799	0023939	0013973	0023927	0013975	0023927	0014081	0023941	0014153	0024009
0540	0014241	0024091	0014323	0024155	0014375	0024219	0014435	0024303	0014533	0024343
0541	0014641	0024379	0014755	0024391	0014857	0024411	0014937	0024423	0015031	0024407
0542	0015105	0024401	0015187	0024415	0015297	0024439	0015373	0024467	0015451	0024497
0543	0015509	0024529	0015565	0024575	0015613	0024647	0015693	0024697	0015781	0024729
0544	0015841	0024655	0015877	0024559	0015899	0024491	0015957	0024393	0016023	0024299
0545	0016137	0024237	0016217	0024161	0016303	0024067	0016387	0023981	0016439	0023883
0546	0016483	0023799	0016539	0023721	0016591	0023657	0016637	0023579	0016637	0023507
0547	0016701	0023475	0016781	0023387	0016905	0023341	0016987	0023297	0017103	0023247
0548	0017223	0023193	0017321	0023145	0017393	0023091	0017495	0023049	0017569	0023009
0549	0017639	0022965	0017721	0022927	0017789	0022875	0017831	0022833	0017887	0022775
0550	0017961	0022721	0018029	0022691	0018095	0022611	0018163	0022547	0018231	0022497
0551	0018319	0022449	0018397	0022425	0018463	0022339	0018459	0022203	0018447	0022105
0552	0018407	0022035	0018337	0021975	0018281	0021895	0018245	0021811	0018205	0021737
0553	0018139	0021661	0018079	0021593	0018021	0021559	0017947	0021513	0017893	0021417
0554	0017833	0021323	0017895	0021203	0017903	0021105	0017929	0021011	0017929	0020927
0555	0017929	0020835	0017941	0020745	0017937	0020665	0017909	0020575	0017877	0020515
0556	0017841	0020437	0017825	0020349	0017807	0020251	0017807	0020151	0017807	0020055
0557	0017811	0019999	0017833	0019945	0017833	0019857	0017801	0019763	0017767	0019693
0558	0017787	0019627	0017797	0019545	0017771	0019455	0017767	0019379	0017797	0019303
0559	0017829	0019221	0017857	0019151	0017893	0019089	0017911	0019021	0017949	0018967
0560	0017997	0018907	0018053	0018851	0018121	0018815	0018169	0018751	0018187	0018687
0561	0018217	0018611	0018263	0018557	0018331	0018503	0018397	0018401	0018395	0018301
0562	0018363	0018233	0018371	0018153	0018405	0018081	0018441	0018011	0018477	0017945
0563	0018523	0017867	0018553	0017797	0018559	0017727	0018559	0017655	0018559	0017575
0564	0018543	0017515	0018485	0017479	0018421	0017473	0018333	0017475	0018255	0017475
0565	0018187	0017469	0018117	0017469	0018041	0017507	0017953	0017533	0017873	0017539
0566	0017321	0017539	0017769	0017575	0017691	0017575	0017601	0017575	0017537	0017531
0567	0017499	0017471	0017489	0017375	0017521	0017301	0017567	0017235	0017627	0017177
0568	0017699	0017175	0017757	0017131	0017837	0017129	0017925	0017141	0017999	0017125
0569	0018077	0017129	0018177	0017125	0018273	0017177	0018395	0017159	0018447	0017093
0570	0018401	0017013	0018573	0016913	0018565	0016799	0018551	0016709	0018557	0016625

FIGURE 3.1

Output: The coordinates of the corner points, the segment lengths and the packed coordinates are printed out. Then follow the packing factor, the allowable plotting error, and the input and output scale denominators. The present version, with simultaneous packing and plotting, plots a copy of the original and packed curves on the 611 oscilloscope. The packed coordinates are stored in arrays XXD and YYD.

After packing, the programme at present loops back and repeats the packing procedure with double the tolerance request.

3.2 Subroutine REDOUT

Storage: 1220 bytes in single precision.

Subroutines called: None

Calling parameters: J - the number of segment lengths. An arbitrary maximum of 20 has been set.

X1, Y1 - the coordinates of the initial point in this set.

SEG - the array of segment lengths in this set, J in number

XP, YP - the coordinates of the final point in this set.

EPS - the specified tolerance.

Output: REDOUT is the routine which prints out the coordinates of the corner points and the segment lengths.

3.3 Subroutine UPLOT

Storage: 2012 bytes in single precision

Subroutines called: None

Calling parameters: M - the number of segment lengths

XI, YI - the coordinate of the initial corner points.

S - the array of segment lengths

XN, YN - the coordinates of the next coordinate point.

E - the specified tolerance.

Output: This routine outputs the packed coordinates, and also stores these in arrays XXD and YYD.

3.4 Subroutines AREA, GRID, SETPLT, NOWPLT, ENDPLT, PRNTCH

These subroutines are university generated routines for plotting. They are detailed in Gujar (1972).

4. TESTS AND COMMENTS

4.1 Tests and results

The present version of the programme has been tested in the Department of Surveying Engineering using digitised contour data obtained from the Analytical Plotter AP-2/C. This data is an exact replica of that which would be obtained from an automatically digitising line follower. That is, the action of an automatic digitiser has been simulated in all respects. A sample of the packing factor obtained with the corresponding error tolerances is shown in table 4.1.

Error Tolerance (Micrometres)	Packing Factor
1	1
50	4.21
100	7.35
200	11.36
400	19.74
800	37.50
1600	75.00
3200	93.75
6400	125.00

Table 4.1

For example, for this particular line, if we wanted to plot it at the original scale, but allowed a plotting error of 50 micrometres, we would reduce the necessary storage to about one quarter. What this means in terms of reduction of scale is that if we wanted to reproduce the original line at 1/50 scale, we would only require one quarter of the storage to plot it accurate to one micrometre.

There is, of course, no direct relationship between packing factor and error tolerance. Obviously the greater the allowable error, the greater will be the packing factor obtained from the process. The original and packed curves corresponding to table 4.1 are shown in Appendix A. The ultimate reduction and generalisation is shown by figure A-9 in which the curve becomes a straight line. It should be noted that nowhere is a packed curve further away from the original than the error tolerance, while the stages of cartographic generalisation from exact to approximate are represented by figures A-1 to A-9. With respect to generalisation it should also be noted that the generalisations still retain the basic characteristics of the original curve. Graph A-9, for example, shows what the curve would look like if reduced by 1/6400 and plotted to micrometre accuracy. (It has, of course, been enlarged in the Appendix)

4.2 Restrictions and Comments

The following points about the present procedure should be noted:

a) the procedure will not work for closed loops due to the characteristics of the scaling process mentioned in section 2.9. The loop must be split into two arc segments.

b) The programme will pack coordinates for curves which are to be reproduced at the same scale. It may be that there are more than enough digitised points on a curve to reproduce it with a given accuracy without reduction. The programme will reduce the number of points to the minimum number required for any given accuracy.

c) The packing factors in table 4.1 are seen to refer to different error tolerances. These can also be thought of as reductions to different scales with the same plotting error. The packing factor is also a function of the smoothness of the curve. The smoother a curve, the greater will be the packing factor, since fewer hyperbolae and segments are needed, that is, the necessary number of parameters are fewer.

d) At present the programme input unit is the 2501 card reader. This is not, of course, mandatory. Generally, digitised data will be on magnetic tape, disk, or paper tape, and appropriate corrections can be made. Ideally the input will be directly on-line, through some device such as the 1827 Data Control Unit.

e) The doubling of the error tolerance in the programmes' present form is purely for testing purposes. Normally the input and output scales, and the error tolerance will be known, and only one packing will be required.

f) The joining together of the packed points should be done by straight lines. The use of curvilinear plotting methods may generate points outside the error tube.

g) Other tests with this programme package were carried out by the Surveys and Mapping Branch, Department of Energy, Mines and Resources in Ottawa using the Branch's automated cartography PDP-10 system, and also the Departments' Univac 1108, and the packing obtained was satisfactory. Modification of the technique to fit such systems is left up to the prospective user.

APPENDIX A

611 OSCILLOSCOPE PLOTS OF A SAMPLE CURVE

FIGURE A-1

ORIGINAL CURVE

FIGURE A-3

PACKED CURVE, PACKING FACTOR = 4.21

ERROR = 50 MICROMETRES

GRID = 1000 MICROMETRES

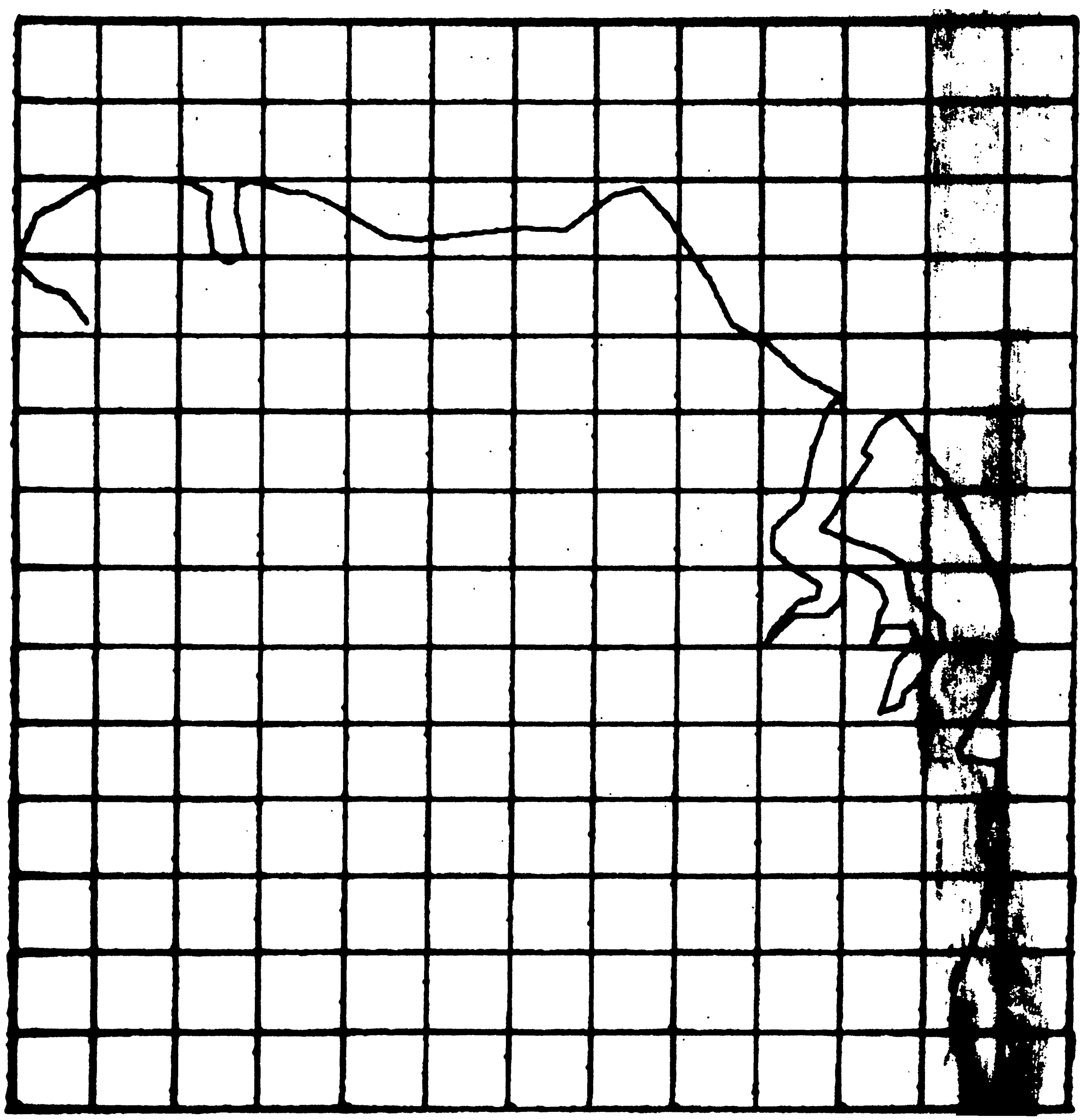


FIGURE A-3

PACKED CURVE, PACKING FACTOR = 7.35
ERROR = 50 MICROMETRES
GRID = 1000 MICROMETRES

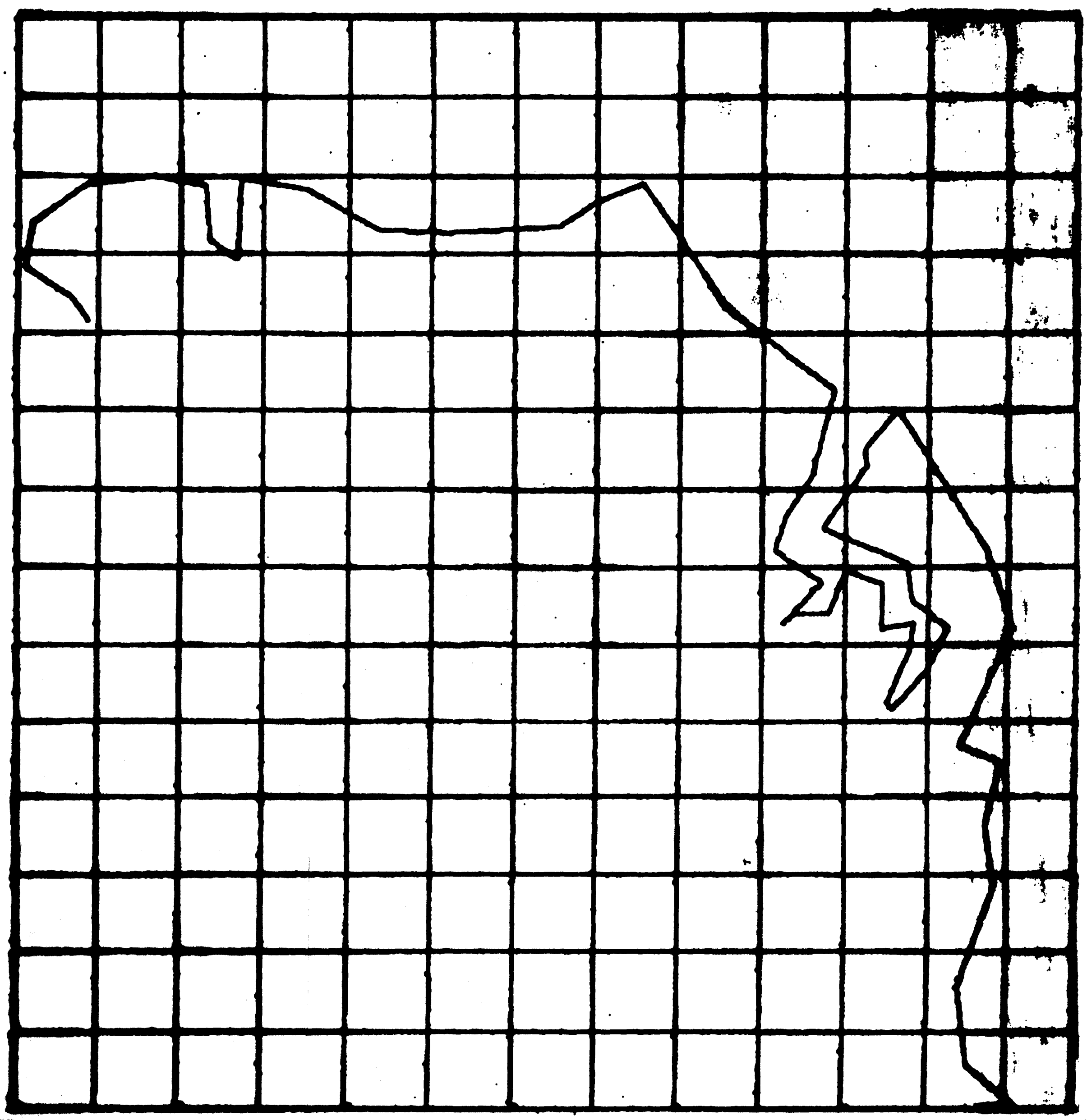


FIGURE A-4

PACKED CURVE, PACKING FACTOR = 11.36

ERROR = 200 MICROMETRES

GRID = 1000 MICROMETRES

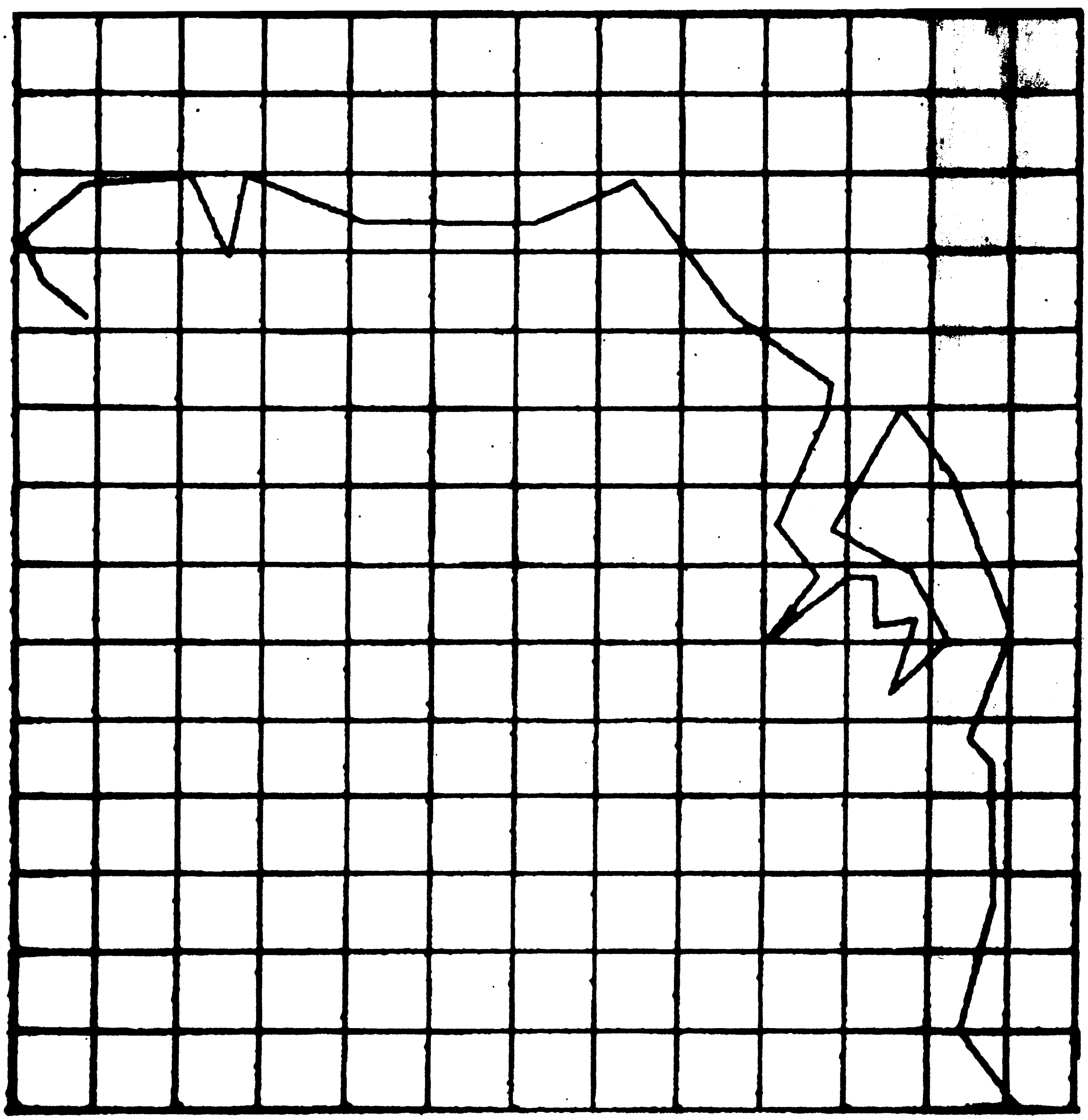


FIGURE A-5

PACKED CURVE, PACKING FACTOR = 19.74

ERROR = 400 MICROMETRES

GRID = 1000 MICROMETRES

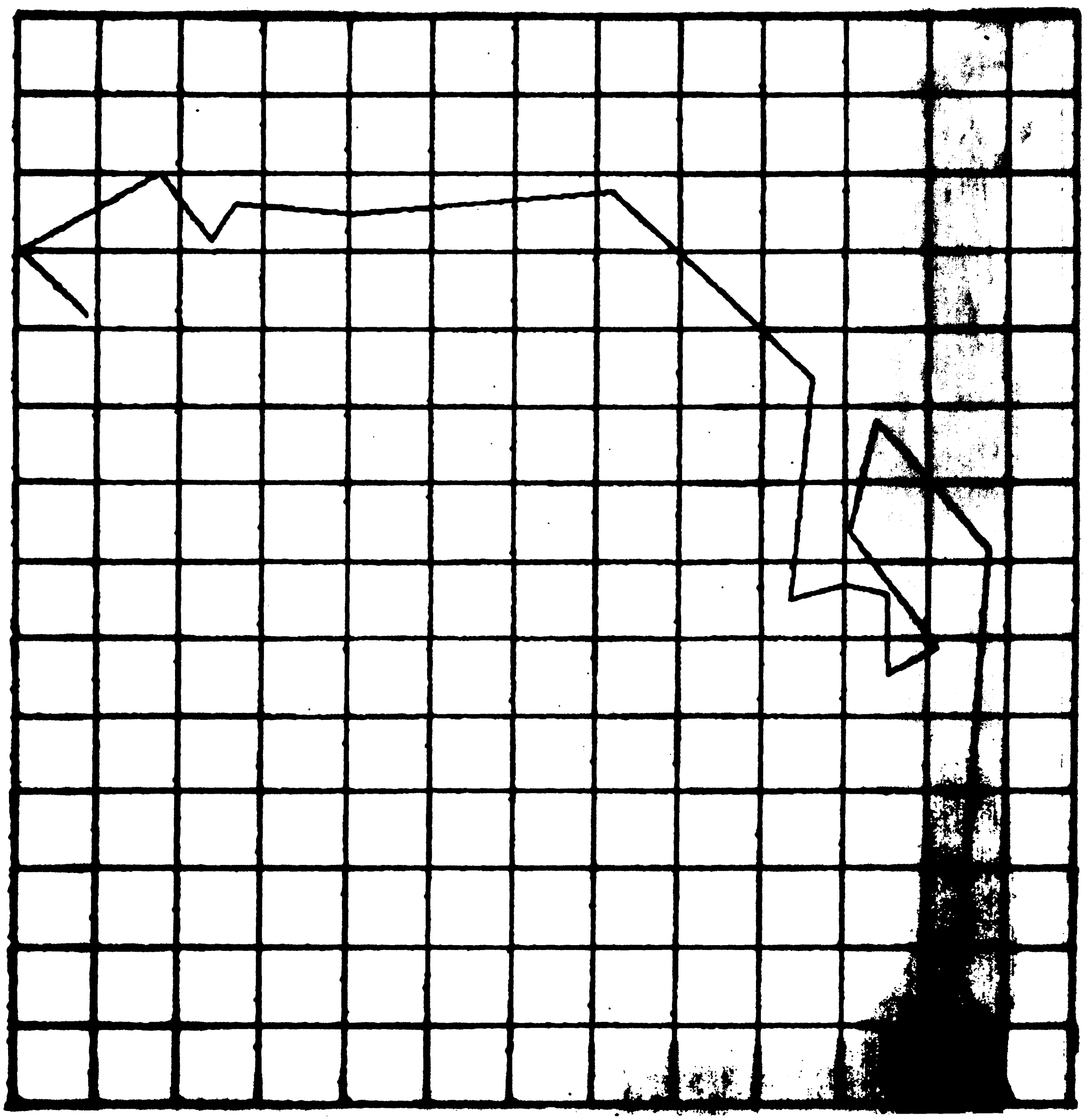


FIGURE A-6

PACKED CURVE, PACKING FACTOR = 37.5

ERROR = 800 MICROMETRES

GRID = 1000 MICROMETRES

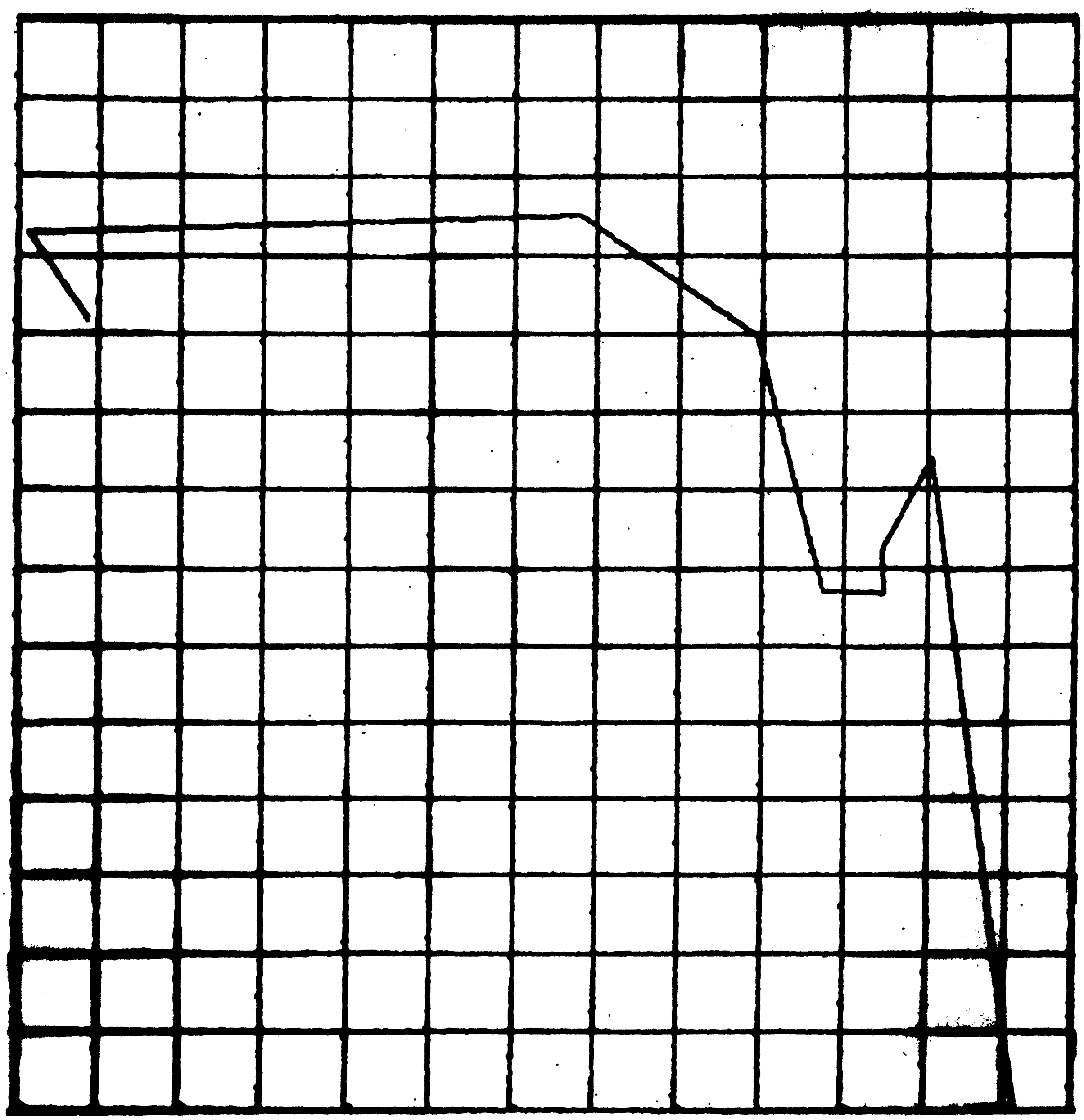


FIGURE A-7

PACKED CURVE, PACKING FACTOR = 75.00

ERROR = 1600 MICROMETRES

GRID = 1000 micrometres

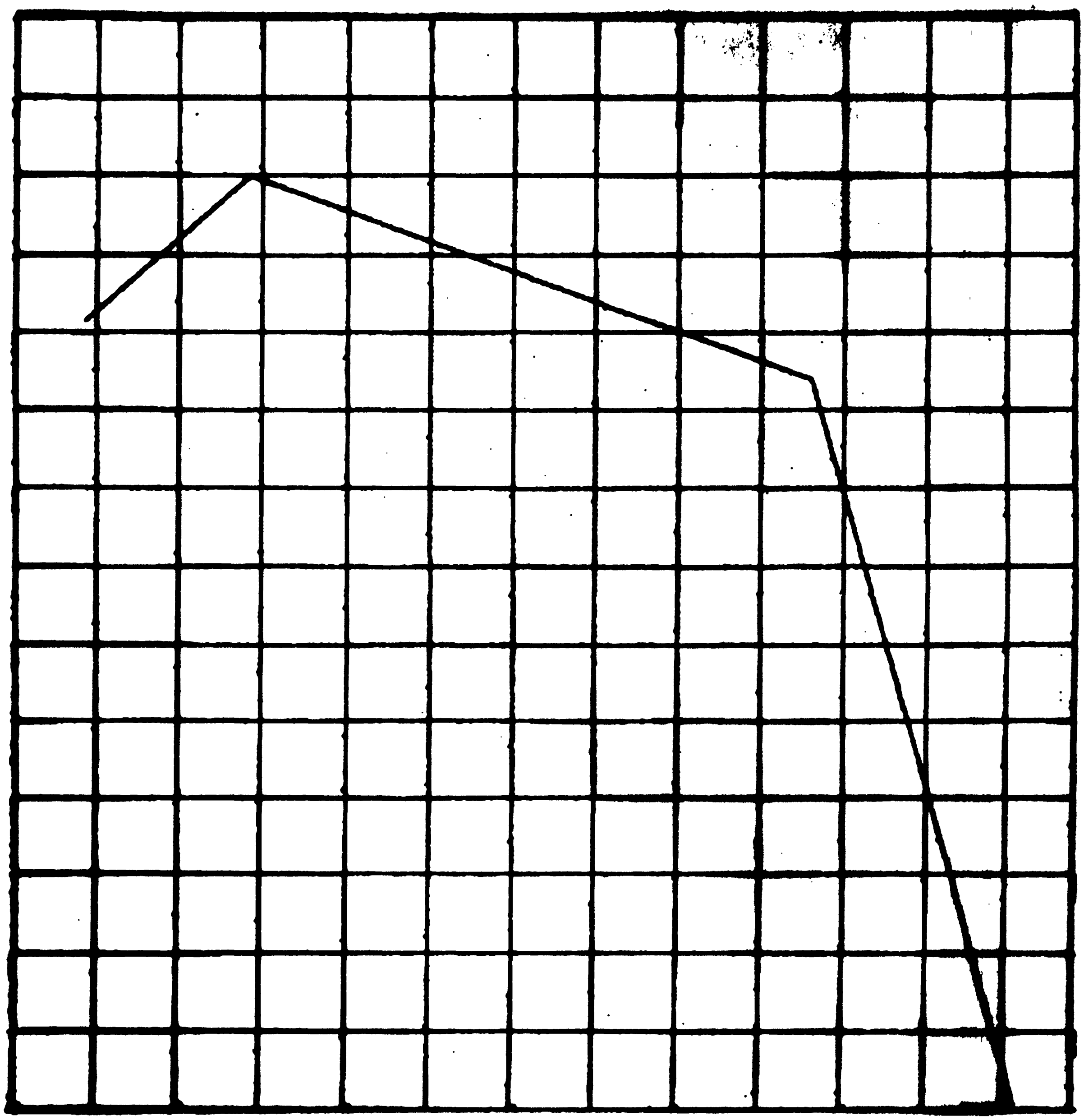


FIGURE A-8

PACKED CURVE, PACKING FACTOR = 93.75

ERROR = 3200 MICROMETRES

GRID = 1000 MICROMETRES

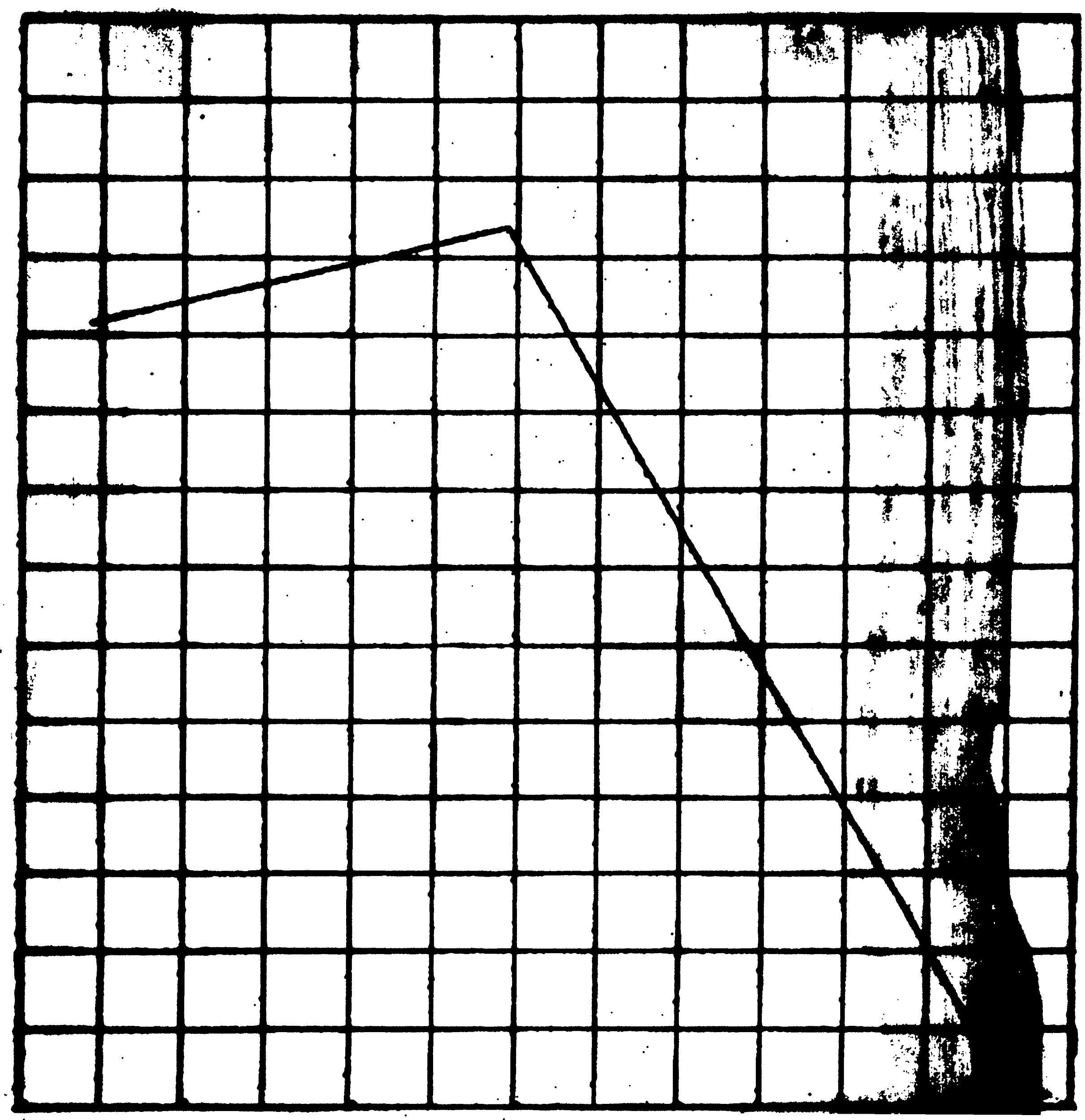
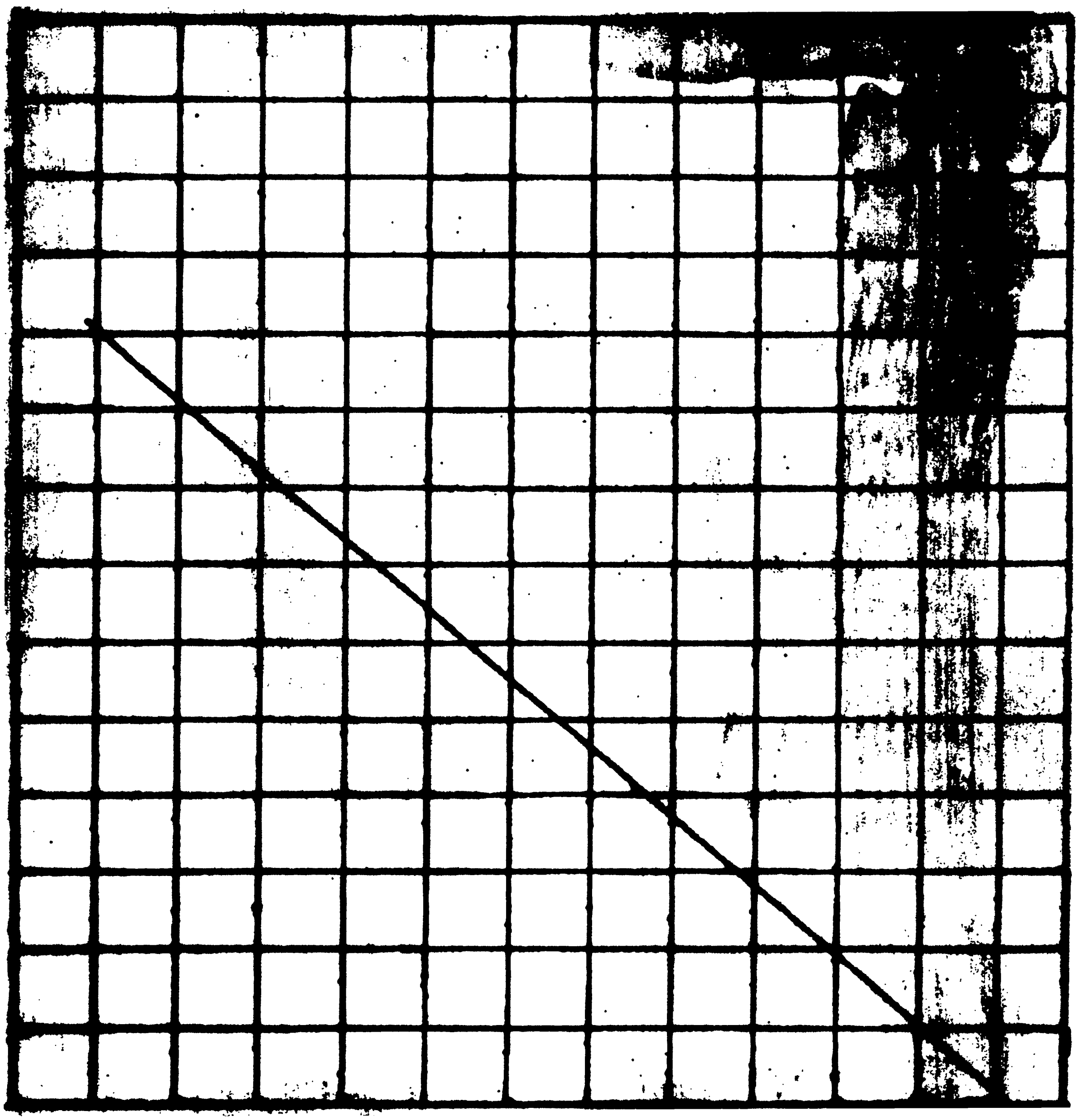


FIGURE A-9

PACKED CURVE, PACKING FACTOR = 125.00

ERROR = 6400 MICROMETRES

GRID = 1000 MICROMETRES



APPENDIX B

COMPUTER PROGRAMME LISTINGS

```
C
C               $$$$$$$$$$$$$$$$$$$$$$
C               *                    *
C               *      PACK          *
C               *                    *
C               $$$$$$$$$$$$$$$$$$$$$$
C
C     FOR A FULL DESCRIPTION,SEE ACCOMPANYING PAPER
C
C     PACK REDUCES AND OUTPUTS DATA
C     PROGRAMME IS DIMENSIONED TO TAKE 5000 DATA POINTS
C     N IS NUMBER OF POINTS
C      X,Y COORDS OF N POINTS
C
      DIMENSION X(5000),Y(5000),DX(5000),DY(5000),SEG(20),TAA(2)
      DIMENSION XX(5000),YY(5000)
        COMMON/INOUT/ITYP,ITAP,ITAPO
        COMMON/FEAT/ICDE,ISUBCD,J,ISCC,ISEG
      COMMON/DAVE/XXD(5000),YYD(5000),IJI
      IJI=0
      IFLAG=1
      XMAXD=0.
      YMAXD=0.
      XMIND=1000000.
      YMIND=1000000.
C
      DO 657 JJ=1,20
657   SEG(JJ)=0
C
C     DELTA IS DIGITISER INCREMENT IN MICRONS  I.E. LEAST COUNT
C
C     ERR IS THE  ALLOWABLE PLOTTING ERROR  AT REDUCED SCALE  IN MICRONS
C
C     ISD IS THE DENOMINATOR OF THE SCALE OF THE INPUT DATA
C
C     IOSD IS THE DENOMINATOR OF THE SCALE OF THE OUTPUT DATA
C
      READ(5,104)ERR,DELTA
      READ(5,100)N
      READ(5,374)ISD,IOSD
374   FORMAT(2I10)
      DRAT=IOSD/ISD
C
C     DRAT IS THE RATIO OF INPUT /OUTPUT SCALES
```

```
C
      ERR=ERR*DRAT
C
C     ERROR IS NOW THE EQUIVALENT ERROR AT INPUT SCALE
C
      ND=N
100   FORMAT(I4)
      NNNN=N/5
      JD=1
      DO 101 JJ=1,NNNN
C
C     READING X AND Y COORDINATES,5 SETS AT A TIME
C
      READ(5,102)(X(JD),Y(JD),X(JD+1),Y(JD+1),X(JD+2),Y(JD+2),
     *X(JD+3),Y(JD+3),X(JD+4),Y(JD+4))
      YY(JD)=X(JD)/1000.
C
C     SCALING X AND Y COORDINATES,PURELY FOR UNE PLOTTING PURPOSES
C
      XX(JD)=Y(JD)/1000.
      YY(JD+1)=X(JD+1)/1000.
      YY(JD+2)=X(JD+2)/1000.
      YY(JD+3)=X(JD+3)/1000.
      YY(JD+4)=X(JD+4)/1000.
      XX(JD+1)=Y(JD+1)/1000.
      XX(JD+2)=Y(JD+2)/1000.
      XX(JD+3)=Y(JD+3)/1000.
      XX(JD+4)=Y(JD+4)/1000.
      JD=JD+5
101   CONTINUE
C
C     SEARCHING FOR MAXIMUM AND MINIMUM COORDINATES,PURELY FOR UNB PLOTTING
C                                                              PURPOSES
      DO 133 JJ=1,N
      IF(X(JJ).GT.XMAXD)XMAXD=X(JJ)
      IF(X(JJ).LT.XMIND)XMIND=X(JJ)
      IF(Y(JJ).GT.YMAXD)YMAXD=Y(JJ)
      IF(Y(JJ).LT.YMIND)YMIND=Y(JJ)
133   CONTINUE
      XMAXD=XMAXD/1000.
      YMAXD=YMAXD/1000.
      XMIND=XMIND/1000.
      YMIND=YMIND/1000.
      XMING=XMIND-2.
```

```
      YMING=YMIND-2.
      XMAXG=XMAXD+2.
      YMAXG=YMAXD+2.
999   FORMAT(' ',2F20.5)
183   CONTINUE
104   FORMAT(2F4.0)
102   FORMAT(5(F7.0,1X,F7.0,1X))
      WRITE(6,167)
167   FORMAT('1')
C
C     THE PLOTTING SEQUENCE USED ON UNB COMPUTER PLOTTING SYSTEM
C
C     IFLAG=1 GIVES PLOT OF ORIGINAL COORDINATES
C     IFLAG = 2 GIVES PLOT OF PACKED COORDINATES
C
      CALL DEVICE(611)
      CALL AREA(2.,2.)
      CALL GRID(YMING,XMING,YMAXG,XMAXG,1.,1.)
      IF(IFLAG.EQ.1)CALL PRNTCH('$')
      IF(IFLAG.NE.1)CALL PRNTCH('*')
      IF(IFLAG.NE.1)N=IJI
      CALL SETPLT(YMIND,XMIND,YMAXD,XMAXD)
      IF(IFLAG.NE.1)CALL NOWPLT(0.,XX(1),YY(1))
      IF(IFLAG.NE.1)CALL NEWPLT(1.,XXD(1),YYD(1))
      IF(IFLAG.EQ.1)CALL NOWPLT(0.,XX(1),YY(1))
      DO 135 IJ=2,N
      IF(IFLAG.NE.1)CALL NOWPLT(1.,XXD(IJ),YYD(IJ))
      IF(IFLAG.EQ.1)CALL NOWPLT(1.,XX(IJ),YY(IJ))
135   CONTINUE
      IF(IFLAG.NE.1)CALL NEWPLT(1.,XX(ND),YY(ND))
      CALL ENDPLT
      IF(IFLAG.NE.1)GO TO 7968
C
C
C     DEVICE ASSIGNMENTS AT UNB --  6  IS THE  LINE PRINTER
C     5 IS THE CARD READER
C
7969  ITYPE=6
      ITAP=5
      ITAPO=6
C
C     CODES FOR INDIVIDUAL MAP ELEMENTS. THE PRESENT SYSTEM ASSUMES ALL COORDS
C     WILL BE OF THE SAME ELEMENT
C
```

```
      ICOE=1
      ISURCD=1
      ISCO=1
      ISCI=1
      ISEG=1
      ISCOF=ISCO-1
C
C        ISCOF FORMER NUMBER OF OUTPUT POINTS
C
      EPS=ERR
C
C      EPS IS NUMBER OF DIGITIZER STEPS
C
      EPSI=1.5*EPS
      EPSH=8.0*EPS
      C2=13.615*EPS*EPS
      C1=8.000894*EPS
      C3=2.82*EPS
C
C        COEFFICIENTS OF HYPERBOLA
C
      NSEG=0
      IB=1
      XP=X(1)
      YP=Y(1)
    1 X1=XP
      Y1=YP
      J=1
      IF(IB.EQ.N)IFLAG=2
      IF(IB.EQ.N)GO TO 183
      DO 2 I=1,2
      DXX=X(IB+I)-X(IB+I-1)
      DYY=Y(IB+I)-Y(IB+I-1)
      DS=1.0/SQRT(DXX*DXX+DYY*DYY)
    2 TAA(I)=DS*DXX
C
C        TAA IS COS
C
      TA=0.5*(TAA(1)+TAA(2))
C
C        TA AVERAGE OF 2 COSINES
C
      TB=SQRT(1-TA*TA)
C
```

```
C          TB CORRESPONDING AVERGE SINE
C
      DO 3 NN=IB,N
      IF(ABS(TA*(Y(NN)-YP)-TB*(X(NN)-XP)).GT.EPSE) GO TO 4
C
C        ARE POINTS IN TUBE EPSE WIDE
C
3        CONTINUE
      NN=N
4     L=1
      ST=0.2
5     DXX=X(NN)-XP
      DYY=Y(NN)-YP
      D=1.0/SQRT(DXX*DXX+DYY*DYY)
C
C          D IS 1/SEGMENT LENGTH
C
      TX=D*DYY
C
C          TX IS SIN THETA
C
      TY=D*DXX
C
C          TY IS COS THETA
C
      DO 6 I=IB,NN
         DNN=I-IB+1
      IF(ABS(TX*(X(I)-XP)-TY*(Y(I)-YP)).GT.EPS) GO TO 16
C
C          CHECK RIGOROUSLY IF POINTS IN TUBE EPS WIDE
C
6     CONTINUE
      IF(NN.GE.N) GO TO 30
      IF(L.EQ.1) GO TO 18
        IF(ABS(ST*DNN).LE.1.0) GO TO 30
19    ST=-0.5*ST
      L=-L
18       IF(ABS(ST*DNN).LE.1) GO TO 17
      NN=NN+IFIX(ST*DNN)
      IF(NN.LE.N)GO TO 14
         NN=N
      GO TO 5
17      NN=NN+L
14      IF(NN.GE.IB) GO TO 5
```

```
      NN=IB
      GO TO 5
16    IF(L.EQ.1) GO TO 19
      GO TO 18
30    SEG(J)=1.0/D
      IF(NN.LT.N) GO TO 7
C
C     OPTIONAL PRINTOUT OF ORIGINAL COORDINATES
C
      DO 962 I=1,N
963   FORMAT(' ',50X,2F12.5)
962   CONTINUE
C
          CALL REDOUT(J,X1,Y1,SEG,X(N),Y(N),EPS)
C
C     UPLOT CALLED TO REDUCE PACKED DATA
C
      CALL UPLOT(J,X1,Y1,SEG,X(N),Y(N),EPS)
      IF(IB.EQ.N)IFLAG=2
      IF(IB.EQ.N)GO TO 183
          ISCOF=ISCO-ISCOF
C
C         ISCOF NOW NUMBER OF OUTPUT POINTS IN THIS BLOCK
C
      ITYP=6
C
C     PRESENT SYSTEM IS SET UP TO DOUBLE ERROR EPSILON AND RE-ITERATE
C     IF ONLY ONE PACKED SET OF DATA IS WANTED(I.E. TO THE ORIGINAL ERROR SPECS
C     READ IN),REPLACE THE TWO FOLLOWING CARDS WITH ONE CARD ---STOP
C
      IFLAG=2
      GO TO 183
7     IB=NN+1
      XPP=XP
      YPP=YP
      XP=X(NN)
      YP=Y(NN)
8     T1=D*(XP-XPP)
C
C          COS OF LINE SEGMENT
C
      T2=D*(YP-YPP)
C
C         SIN OF LINE SEGMENT
```

```
C
      DO 10 I=IB,N
C
C        CHECK POINTS BEYOND SEGMENT
C
      DX(I)=T1*(X(I)-XP)+T2*(Y(I)-YP)
C
C        DX  DISPLACEMENT IN LINE DIRECTION
C
      DY(I)=T1*(Y(I)-YP)-T2*(X(I)-XP)
C
C        DY DISPLACEMENT  IN DIRECTION NORMAL TO LINE
C
      IF(DX(I).LE.0.0) GO TO 9
C
C        IF LINE TURNS BACK PAST FIRST POINT THEN CORNER POINT
C
      IF(ABS(DY(I))*(DX(I)+C3)-(C1*DX(I)+C2).GT.0.0) GO TO 11
C
C        CHECK IF DY STILL WITHIN HYPERBOLA
C
10    CONTINUE
      GO TO 9
11    C1B=SIGN(C1,DY(I))
      C2B=SIGN(C2,DY(I))
C
C        C1B,C2B SIGNS SAME AS DY
C
      DDX=DX(I)-DX(I-1)
C
C        DDX DIFFERENCE BETWEEN LAST 2DX'S
C
      IF(ABS(DDX).GT.2.0E-8) GO TO 12
C
C        IS HYPERBOLA CUT BY NEARLY NORMAL LINE TO AXIS
C
      DXI=DX(I)
      GO TO 13
12    AL=(DY(I)-DY(I-1))/DDX
C
C        AL IS SLOPE OF LAST TWO POINTS
C
      Q=DY(I)-AL*DX(I)
C
```

```
C          Q IS Y INTERCEPT OF STRAIGHT LINE BETWEEN LAST 2 POINTS
C
      CCC=C1B*C3-C2B
      DXBB=DX(I)
      DXI=DX(I-1)
      QB1=(C1B*DXBB+C2B)/(DXBB+C3)
      DEN1=DXBB+C3
      DEN2=C3*DEN1
31    DXIB=DXI
      ALB=CCC/(DXIB*DEN1+DEN2)
      QB=QB1-ALB*DXBB
      DXI=(Q-QB)/(ALB-AL)
      IF(ABS(DXI-DXIB).GT.0.5*DELTA) GO TO 31
13    DYI=(C1B*DXI+C2B)/(DXI+C3)
      SEM=SQRT(DXI*DXI+DYI*DYI)
      TX=DYI/SEM
      TY=DXI/SEM
      II=I-1
      DO 15 L=IB,II
      IF(ABS(TX*DX(L)-TY*DY(L)).GT.EPS) GO TO 9
15    CONTINUE
      IB=I
      J=J+1
      SEG(J)=SIGN(SEM,DYI)
      XPP=XP
      YPP=YP
      S=1.0/SEM
      DDXI=DXI-DX(I)
      DDYI=DYI-DY(I)
      XP=X(I)+T1*DDXI-T2*DDYI
      YP=Y(I)+T1*DDYI+T2*DDXI
      GO TO 8
9        CALL REDOUT(J,X1,Y1,SEG,XP,YP,EPS)
C
C
C     UPLOT CALLED TO REDUCE PACKED DATA
C
      CALL UPLOT(J,X1,Y1,SEG,XP,YP,EPS)
      IF(IB.EQ.N)IFLAG=2
      IF(IB.EQ.N)GO TO 183
C
C          OUTPUT BLOCK OF REDUCED DATA
C
      GO TO 1
```

```
7968  IFLAG=1
C
C     CALCULATION OF PACKING FACTOR, AND PRINTOUT OF ERRORS AND SCALES
C
      FACTP=FLOAT(ND)/(FLOAT(IJI)+2.0)
      WRITE(6,5432)FACTP
5432  FORMAT(' ',20X,'PACKING FACTOR  = ',F20.2)
      XERR=ERR/DRAT
      WRITE(6,5434)XERR
5434  FORMAT(' ',20X,'ERROR AT REDUCED SCALE IN MICRONS  = ',F20.10)
5433  FORMAT(' ',20X,' ERROR AT ORIGINAL SCALE =  ',F20.10)
      WRITE(6,5433)ERR
      WRITE(6,5435)ISD,IOSD
5435  FORMAT(' ','INPUT DENOMINATOR = ',I10,'    OUTPUT DENOMINATOR = ',
     *I10)
C
C     THIS SECTION DOUBLES THE ERRCR TUBE AND REPEATS THE PACKING PROCEDURE
C
      ERR=ERR*2.
      IJI=0
      IB=N-1
      N=ND
      IF(ERR.GT.10000.)STOP
C     MAXIMUM ERROR ALLOWED = 10000 MICROMETRES
C
      GO TO 7969
40       FORMAT(' ',2X,6(5X,I4),5X,F8.2)
      END
```

```
      SUBROUTINE REDOUT(J,X1,Y1,SEG,XP,YP,EPS)
C
C      REDOUT OUTPUTS RESULTS OF DATRED ONTO ITAPO AFTER REDUCTION
C        J NUMBER OF SEGMENTS IN THIS RECORD
C       X1,Y1 INITIAL POINT IN THIS RECORD
C       SEG ARRAY OF SEGMENT LENGTHS... J OF THEM
C       XP,YP FINAL POINT IN THIS RECORD
C        EPS TOLERANCE SPECIFIED FOR THIS REDUCTION
C
      DIMENSION SEG(J),ISSEG(20)
C      EQUIVALENCE OUTPUTS TO INTEGERS
      COMMON/INOUT/ITP,ITAP,ITAPO
      COMMON/FEAT/ICDE,ISUBCD,ISCI,ISCO,ISEG
      ITAPO=6
C
C     IF INTERMEDIATE STORAGE OF DATA REQUIRED,CHANGE UNIT NUMBER
C
C
C       ICDE,ISUBCD ARE CODE AND SUBCODE OF FEATURE
C       J NUMBER OF SEGMENTS IN THIS RECORD
C       ISCO OUTPUT RECORD NUMBER
C
      IF(ISCO .GT. -1)GO TO 1
      ISEG=0
C
C      ISEG NUMBER OF SEGMENT LENGTHS OUTPUT PER FEATURE
C
      IX1=X1+.5
      IY1=Y1+.5
C
C     INITIAL POINT MADE AN INTEGER
C
      WRITE(ITAPO,40)IX1,IY1,ICDE,ISUBCD
      IEPS=EPS+.5
C
C      SET ERROR AS INTEGER
C
      WRITE(ITAPO,41)IEPS
C
C       IF FIRST RECORD OF FEATURE WRITE FIRST POINT X1,Y1 AND EPS
C
1     IXP=XP+.5
      IYP=YP+.5
C
```

```
C        MAKE INTEGERS OF FINAL POINTS
C
      WRITE(ITAPO,40)IXP,IYP,ISCC,J
C
C        WRITE FINAL POINT XP,YP RECORD ISCO AND NO SEG LENGTHS J
C
      ISCO=ISCO+1
      IF(J .EQ. 1) RETURN
      DO 100 I=1,J
      ISSEG(I)=SEG(I)+.5
100   CONTINUE
C
C        CHANGE SEGMENT LENGTHS TO INTEGERS
C
      ISEG=ISEG+J
      WRITE(ITAPC,42)(ISSEG(I),I=1,J)
C
C        WRITE J SEGMENT LENGTHS
C
      RETURN
40    FORMAT(' ',2I6,2I4)
41    FORMAT(' ',I6)
42    FORMAT(' ',20X,'SEGMENT LENGTHS ARE',20X//20I6)
      END
```

```
      SUBROUTINE UPLOT(M,XI,YI,S,XN,YN,E)
C
C     SUBROUTINE TO UNPACK PACKED DATA
C
      DIMENSION S(20),GX(22),GY(22)
       COMMON/INOUT/ITYP,ITAP,ITAPO
      COMMON/DAVE/XXD(5000),YYD(5000),IJI
      DATA STI/1.0000/
C
C     IF A CONVERSION OF UNITS IS REQUIRED,CHANGE STI
C
      IF(M .LE. 0) RETURN
C
C      M LE 0 FOR DUMMY CALL
C
      IF(M.GE.2) GO TO 200
C
C     FOR SINGLE SEGMENT NEED ONLY PLOT END POINTS
C
      EX=XN*STI
      EY=YN*STI
      WRITE(6,199)EX,EY
      IJI=IJI+1
      XXD(IJI)=EY/1000.
      YYD(IJI)=EX/1000.
C
C     SCALING FOR UNB PLOTTING PURPOSES ONLY
C
      IFLAG=2
199   FORMAT(' ',4X,'UPLOT COORDINATES ARE  ',2F20.10)
      RETURN
C
C     CALCULATION OF UNPACKING
C
200    GY(1)=0.0
      GX(1)=0.0
      GX(2)=S(1)
      GY(2)=0.0
      C01=8.000894*E
      C02=13.615*E*E
      C03=2.82*E
C
C       COEFFS OF HYPERBOLA
C
```

```
      A0=C02*C02
      A1=2*C01*C02
      A2=C01*C01
      V=E/SQRT(FLOAT(M))
      DO 20 I=2,M
      SS=S(I)*S(I)
      SI=ABS(S(I))
      B0=C03*C03*SI
      B1=2*C03*SI+C03*C03
      B2=2*C03+SI
      X=ABS(S(I))
21    P2=((A2*X)+A1)*X+A0
      P3=((X+B2)*X+B1)*X+B0
      X=SI-P2/P3
      Y=(C01*X+C02)/(X+C03)
      IF(ABS(X*X+Y*Y-SS) .GT. ABS(S(I)*V)) GO TO 21
      Y=SQRT(SS-X*X)
      Y=SIGN(Y,S(I))
      T1=(GX(I)-GX(I-1))/ABS(S(I-1))
      T2=(GY(I)-GY(I-1))/ABS(S(I-1))
C        T1,T2 COS, SIN OF LINE SEGMENT ANGLE REL TO AXIS
      GX(I+1)=GX(I)+T1*X-T2*Y
20    GY(I+1)=GY(I)+T2*X+T1*Y
C
C        GX,GY REL COORDS OF END PT OF LINE SEG
C
24    MM=M+1
      D=1.0/(GX(MM)*GX(MM)+GY(MM)*GY(MM))
      C1=(GY(MM)*(YN-YI)+GX(MM)*(XN-XI))*D
      C2=(-GY(MM)*(XN-XI)+GX(MM)*(YN-YI))*D
C
C        EVERYTHING EXPRESSED IN DIGITIZER STEPS
C
      DO 41 I=2,MM
      EX=(XI+C1*GX(I)-C2*GY(I))*STI
      EY=(YI+C2*GX(I)+C1*GY(I))*STI
C
C       COMPUTE END POINTS FOR EACH LINE SEGMENT
C
      WRITE(6,199)EX,EY
      IJI=IJI+1
      XXD(IJI)=EY/1000.
      YYD(IJI)=EX/1000.
C
```

```
C       SCALING FOR UNB PLOTTING PURPOSES ONLY
C
        IFLAG=2
C         DRAW LINE SEGMENTS
41        CONTINUE
        RETURN
        END
```

REFERENCES

Bush, G.G. and Obrean, P.E. (1965), "Basic Concepts of Mathematics". Holt, Rinehart and Winston, New York.

Gujar, U. (1972), "Computer Plotting". Publication #CL278, Computing Centre, University of New Brunswick, Fredericton.

Ralston, A., (1965), "A First Course in Numerical Analysis". McGraw-Hill, New York.