THE UNIVERSITY OF MANITOBA

ANATOMICAL SHAPE GENERATION USING MATHEMATICAL SPLINES

by

Gordon Stanley Pizey

A Thesis
Submitted to the Faculty of Graduate Studies
in Partial Fulfilment of the Requirements for the Degree
of Master of Science

Department of Mechanical Engineering

Winnipeg, Manitoba
May, 1974

ABSTRACT

Present techniques in the fabrication of artificial limbs require the extensive use of hand craftsmanship for accurate limb-shape reproduction. This study proposes a method by which a numerical control machine tool can rapidly manufacture a dimensionally accurate prosthesis. Spline functions were used to produce spatial coordinates which guide the cutting tool along the work piece.

A major constraint on the total design was one of keeping the system size to a minimum. This would enable a small shop or research centre to purchase such a system. To this end, a technique was devised which minimized the number of raw data points necessary for accurate limb shape definition (clustering). In addition, direct numerical control was implemented to bypass the requirement of a large machine tool language compiler. The machine tool control and mathematical shape definition was implemented on a Hewlett-Packard 2116C mini computer.

The results show that an entire limb shape can be duplicated using the method of clustering with the spline functions. The technique gives excellent results even when only a relatively small number of data points from the sampled limb are utilized.

ACKNOWLEDGEMENTS

The author would like to thank Dr. A. B. Thornton-Trump for his guidance and encouragement throughout the course of this work. Gratitude is also extended to Reinhardt Daher for his many hours of assistance. Finally, thanks are due to Lea Dipple for her careful typing of this thesis, and to Henry Scholz for his drafting assistance.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

a	first point in the function space
b	last point in the function space
h	chordal distance
m	nodal slope of the spline function
M	nodal moment of the spline function
s	cumulative chordal distance
$S_{\Delta}(x)$	spline function on Δ
x	coordinate in data set
y	coordinate in data set
x^c	x value from circumferential spline
y^c	y value from circumferential spline
x_{long}	longitudinal spline point in x direction
y_{long}	longitudinal spline point in y direction
z_{long}	longitudinal spline point in z direction
Δ	region of spline function
λ	end point curve estimator
μ	end point curve estimator

1 INTRODUCTION

1.1 PRESENT PRODUCTION TECHNIQUES

Production techniques in the manufacture of artificial limbs have until recently been rather limited in scope. The limbs are presently hand made (6), which has associated with it a number of problems, since the dimensional accuracy is only as good as the judgement of the person making the limb. It would be most desirable to have the size and shape of the replacement limb an image of the existing limb. Observations have shown that patients have a greater degree of confidence and well-being when beginning to walk with an artificial leg which looks natural (2). It has also been found that the faster the patient receives his new limb, the fewer will be the psychological orientation problems (8). It would thus be desirable to improve the dimensional accuracy of the fabricated appendage, and allow measurements to be taken prior to the arrival of the patient at the hospital. The overall process would result in early treatment for the patient, thus facilitating an early return to work. This means decreased costs to both the patient and the hospital.

1.2 DATA PROCESSING SYSTEM

It is possible to eliminate or at least reduce the negative aspects of the above-mentioned problems by employing the following system. A relatively portable, lightweight device which

is capable of measuring the dimensions of a patient's limb (5) is taken to the patient (a more detailed account of this is given in Appendix I). Voltage or frequency signals, indicative of limb dimensions, are sent through an acoustic coupler over the telephone lines. These signals are then processed through a digital voltmeter and entered into computer memory. The computer then processes this information and produces a tape to either directly enable a numerically controlled machine to produce the artificial limb, or to enable a compiler such as APT (Automatically Programmed Tools), to produce a tape which would allow a machine to make the artificial limb. A system must be designed to not only overcome the difficulties as stated in 1.1, it must also possess the potential advantage of being able to store a large number of these limb shapes in digitized form. From these shapes, it will be possible to generate statistically standard shapes. Using these standard shapes, only small adjustments need be made for particular requirements.

The main body of this thesis deals with the processing of the digitized limb data, and the production of a tape to allow the fabrication of an artificial limb of acceptable dimensional accuracy. A numerically controlled machine tool could then be used to produce a model of the limb.

1.3 MOULD PRODUCTION BY NUMERICAL CONTROL

The model from which the mould is fabricated will be produced by a numerically controlled machine tool. This tool

receives information allowing it to carve the shape from the processed input data. The initial data alone only approximates the limb dimensions, but in its raw form is far too sparse for the machine to interpolate and then cut smooth contours on the model.

The raw data yields smooth paths for the machine tool to follow only after spline functions have been used to generate a large number of intermediate points. These "interstitial points" (between the data points) are such that when the machine tool cuts along their locus, the mould which is produced will be almost identical to the sampled limb.

Once the model has been thus carved out, it is ready to be used to make the polyurethene cosmesis. The core of the cosmesis can be designed to accommodate the supporting structure of the limb. The support structure includes the attachment to the artificial foot, below knee socket, or knee joint, depending on level of amputation.

1.4 SOCKET PRODUCTION

In addition to manufacturing for purposes of cosmetic restoration by numerical control, it is also possible to use the above-mentioned technique to produce sockets. The socket is the receptor for the patient's stump, and for reasons of weight-bearing, ambulation, and comfort, must be properly designed.

It is possible to place the measuring device at the stump area and obtain stump formation data. Again using splines,

the contours found at the stump site could be modified to an accepted socket shape, ensuring a snug fit. The core of the cosmesis would be designed to accommodate the socket.

Due to stump dimensions continually changing, especially after recent amputation, new sockets must be fabricated. The method described in the thesis would result in a saving of fabrication time, and consequently in-hospital care. In addition, a continuing record of atrophy of the stump would be maintained.

1.5 IMAGE SENSING

Throughout this production process, the limb which is created is dimensionally accurate everywhere to one-sixteenth of an inch of the limb which is sensed. It must be remembered that points sensed on the actual limb are mirror-imaged to produce the cosmesis of the artificial limb. This assumes limb symmetry for the patient sampled.

The logical extension of this proposed system would be a generalized method of reproducing the variety of shapes required for custom cosmetic restoration. This system also provides a simple means of image sensing and also shape storage by using splines to generate virtually an infinite number of points.

1.6 OBJECTIVES

To summarize the objectives of this study, the primary goals of the equipment package design are the following:

i) to increase the confidence and willingness of the patient to use his prosthesis, by ensuring that the prosthesis quite closely resembles his own limb, and by allowing the patient to use the limb almost immediately (due to the rapid socket and limb production technique).

ii) to reduce the time spent by the patient in the hospital. This realizes savings for both patient and hospital.

iii) to keep the capital cost of the equipment to a minimum so that it is within the budget capabilities of a private venture or a hospital centre.

2 THEORY

2.1 INTRODUCTION

The problem is to define a three-dimensional shape within certain dimensional requirements, by using a minimum number of points. Having established a criteria for obtaining the minimum number of points, the next stage involves implementing a procedure which reproduces shapes economically. The two aspects of the problem are considered in this section.

2.2 SHAPE DEFINITION

2.2.1 Splines

Many draftsmen, instead of using French curves to fill in smooth lines between data points, have been using long narrow strips of relatively flexible material (for example plastic) to do the job. The pieces of material are called splines. The procedure is to attach weights on the spline at certain locations in order that a smooth line may be drawn through the specified data points. By changing the position of the spline, and changing the number and position of the weights, almost any curve of "reasonable" curvature and variability may be fitted.

The mathematical spline essentially produces the same result as the physical spline. The elasticity of the spline strip is replaced by fitting a cubic function between the data points.

This cubic is a piecewise continuous function only in the particular interval between any two adjacent data points. In general, a different function is created in each interval. At any given data point where one cubic function ends and another begins, there are certain discontinuities permitted. However, in its simplest form, the mathematical spline is continuous with continuous first and second derivatives. Thus only derivative values at the first and last point have to be specified. Placing the mathematical weights at the given points results in a system of linear equations which has a unique solution amenable to efficient computation.

Consider attempting to fit the shape in Figure 1 with an analytic function. The function to be fitted is not globally analytic (global in the sense that one function will not adequately describe or fit the entire curve). However, the global nature of any generated function would be determined from its behavior in some local neighborhood. The extrapolation from local to global would undoubtedly not work, since the curve that is to be fitted is not globally analytic. It is desirable to fit certain regions with analytic functions and have defined requirements at the end points of the regions. Spline functions are only piecewise analytic (analytic between any two consecutive data points). This property gives spline functions a great advantage over analytic functions in general.

The two types of spline functions (periodic - end points equivalent non-periodic - end points not equivalent) to be used

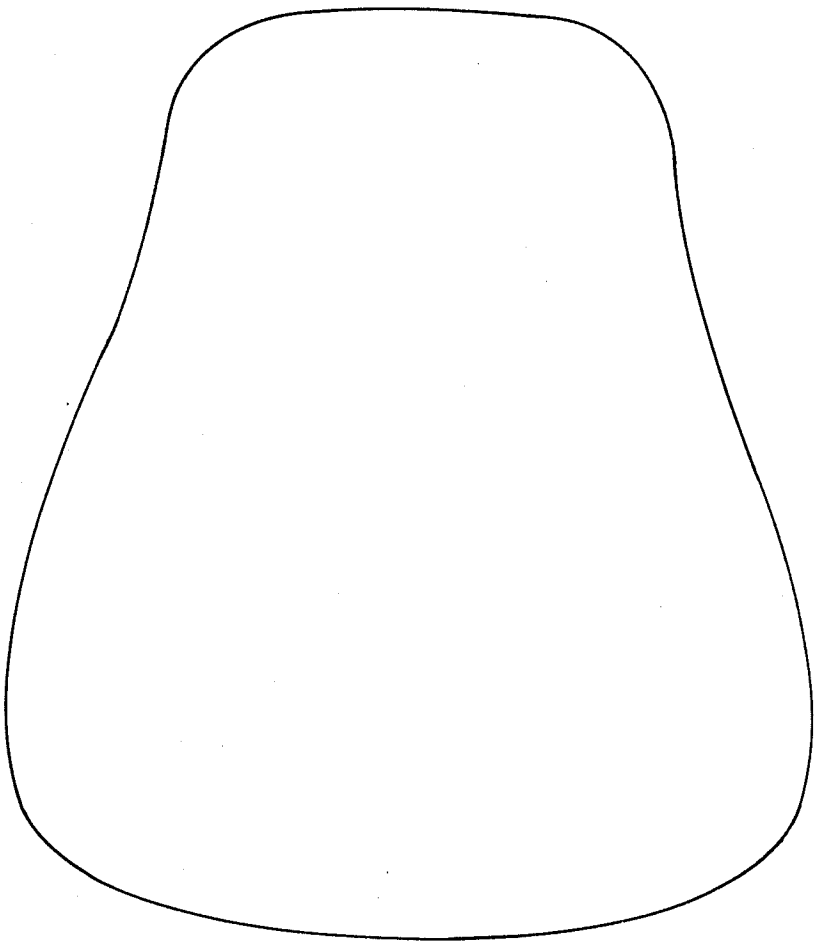


Figure 1 Hypothetical Limb Cross-Section

will be given, with a short theoretical introduction. A mathematical derivation of the defining equations is given in Appendix II (see also Reference 1).

2.2.2 Periodic Splines

Consider a closed interval [a,b] which is partitioned into subintervals by some partition Δ: $a = x_0 < x_1 < \ldots\ldots < x_n = b$. Suppose that the ordinates $y_0, y_1, \ldots\ldots y_n$ are also given. The problem is to pass a smooth curve through the points (x_0, y_0), (x_1, y_1), $\ldots\ldots$ (x_n, y_n). The mathematical spline, $S_\Delta(x)$, is represented in each interval $x_{j-1} \leq x \leq x_j$ as a cubic function of x. Each cubic function in the interval is pieced together so that the entire curve obtained is a continuous function of x with continuous first and second derivatives.

Suppose a set of points in two-dimensional space is given, i.e. the points (x_1, y_1), (x_2, y_2), $\ldots\ldots$ (x_n, y_n). Let the spacing between any two consecutive points, say between (x_j, y_j) and (x_{j+1}, y_{j+1}) be given by h_j, where $h_j = \sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2}$. The equation which defines y as a function of x is:

$$y = S_\Delta(x) = m_{j-1}(x_j-x)^2(x-x_{j-1})/h_j^2 - m_j(x-x_{j-1})^2(x_j-x)/h_j^2$$

$$+y_{j-1}(x_j-x)^2(2(x-x_{j-1})+h_j)/h_j^3 + y_j(x-x_{j-1})^2(2(x_j-x)+h_j)/h_j^3 \quad ..(1)$$

(Appendix II for derivation)

where for the periodic case (which means that $y_0 \equiv y_n$, $x_0 \equiv x_n$ as in Figure 1), the quantities m satisfy the relationship

$$\begin{bmatrix} 2 & \mu_1 & 0 & \cdots & 0 & 0 & 0 \\ \lambda_2 & 2 & \mu_2 & \cdots & 0 & 0 & 0 \\ 0 & \lambda_3 & 2 & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 2 & \mu_{n-2} & 0 \\ 0 & 0 & 0 & \cdots & \lambda_{n-1} & 2 & \mu_{n-1} \\ \mu_n & 0 & 0 & \cdots & 0 & \lambda_n & 2 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \cdot \\ \cdot \\ \cdot \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \cdot \\ \cdot \\ \cdot \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix} \quad ..(2)$$

where $\lambda_j = h_{j+1}/(h_j+h_{j+1})$ $\qquad$ $\lambda_n = h_1/(h_n+h_1)$

$\mu_j = 1-\lambda_j$ $\qquad$ $\mu_n = 1-\lambda_n$

and the quantity c_j represents

$$3\lambda_j(y_j-y_{j-1})/h_j + 3\mu_j(y_{j+1}-y_j)/h_{j+1}$$

The equation which defines x as a function of y is:

$$x = S_\Delta(y) = m_{j-1}(y_j-y)^2(y-y_{j-1})/h_j^2 - m_j(y-y_{j-1})^2(y_j-y)/h_j^2$$

$$+x_{j-1}(y_j-y)^2(2(y-y_{j-1})+h_j)/h_j^3 + x_j(y-y_{j-1})^2(2(y_j-y)+h_j)/h_j^3 \quad ..(3)$$

where for the periodic case ($y_0 \equiv y_n$, $x_0 \equiv x_n$), the quantities m satisfy:

$$\begin{bmatrix} 2 & \mu_1 & 0 & \cdots & 0 & 0 & 0 \\ \lambda_2 & 2 & \mu_2 & \cdots & 0 & 0 & 0 \\ 0 & \lambda_3 & 2 & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 2 & \mu_{n-2} & 0 \\ 0 & 0 & 0 & \cdots & \lambda_{n-1} & 2 & \mu_{n-1} \\ \mu_n & 0 & 0 & \cdots & 0 & \lambda_n & 2 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \cdot \\ \cdot \\ \cdot \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \cdot \\ \cdot \\ \cdot \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix} \quad \ldots(4)$$

and where $\lambda_j = h_{j+1}/(h_j+h_{j+1})$ $\qquad$ $\lambda_n = h_1/(h_n+h_1)$

$\mu_j = 1-\lambda_j$ $\qquad$ $\mu_n = 1-\lambda_n$

and the quantity c_j represents:

$$3\lambda_j(x_j-x_{j-1})/h_j + 3\mu_j(x_{j+1}-x_j)/h_j$$

Note that equations (1) and (3) are valid for the periodic case ($y_0 \equiv y_n$, $x_0 \equiv x_n$) as the first point in the data set is also the last point. This is valid, for the curves describing a cross-section of a human limb having starting and end

points identical.

However, for limb cross-sections, it is not possible to state the y coordinate is a function of x or the x coordinate is a function of y. A more useful concept is to interpolate between prescribed (x, y) values separately. This is achieved by using a parameter which is a function of both x and y values. The parameter is then used to generate x values between the prescribed x data points. The same parameter generates y values between the prescribed y data points. The only link between the x and y spline points so generated is the parameter.

The parameter used in this analysis is the cumulative chordal distance, s_j, where $h_j = \sqrt{(x_j-x_{j-1})^2+(y_j-y_{j-1})^2}$ and s_j is thus given by $s_j = h_1 + h_2 + \ldots + h_j$. s is any value of chordal distance lying between any two consecutive chordal distances s_j and s_{j-1}. By this method, the following equations are generated:

$$x = S_\Delta(s) = m_{j-1}(s_j-s)^2(s-s_{j-1})/h_j^2$$

$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2+x_{j-1}(s_j-s)(2(s-s_{j-1})+h_j)/h_j^3$$

$$+x_j(s-s_{j-1})^2(2(s_j-s)+h_j)/h_j^3 \quad \ldots\ldots\ (5)$$

where for the periodic case ($y_0 \equiv y_n$, $x_0 \equiv x_n$), the quantities m satisfy:

$$
\begin{bmatrix}
2 & \mu_1 & 0 & \cdots & 0 & 0 & 0 \\
\lambda_2 & 2 & \mu_2 & \cdots & 0 & 0 & 0 \\
0 & \lambda_3 & 2 & \cdots & 0 & 0 & 0 \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot \\
0 & 0 & 0 & \cdots & 2 & \mu_{n-2} & 0 \\
0 & 0 & 0 & \cdots & \lambda_{n-1} & 2 & \mu_{n-1} \\
\mu_n & 0 & 0 & \cdots & 0 & \lambda_n & 2
\end{bmatrix}
\begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \cdot \\ \cdot \\ \cdot \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix}
=
\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \cdot \\ \cdot \\ \cdot \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix}
\quad \ldots (6)
$$

where $\lambda_j = h_{j+1}/(h_j+h_{j+1})$ $\qquad$ $\lambda_n = h_1/(h_n+h_1)$

$\mu_j = 1-\lambda_j$ $\qquad$ $\mu_n = 1-\lambda_n$

and the quantity c_j represents:

$$3\lambda_j(x_j-x_{j-1})/h_j + 3\mu_j(x_{j+1}-x_j)/h_{j+1}$$

For obtaining y as a function of the parameter s:

$$y = S_\Delta(s) = m_{j-1}(s_j-s)^2(s-s_{k-1})/h_j^2$$

$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2$$

$$+y_{j-1}(s_j-s)^2(2(s-s_{j-1})+h_j)/h_j^3$$

$$+y_j(s-s_{j-1})^2(2(s_j-s)+h_j)/h_j^3 \quad \ldots\ldots\ldots \quad (7)$$

where for the periodic case, the quantities m satisfy:

$$\begin{array}{ccccccccc}
2 & \mu_1 & 0 & \ldots & 0 & 0 & 0 & m_1 & c_1 \\
\lambda_2 & 2 & \mu_2 & \ldots & 0 & 0 & 0 & m_2 & c_2 \\
0 & \lambda_3 & 2 & \ldots & 0 & 0 & 0 & m_3 & c_3 \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot & = \cdot & \cdot \\
\cdot & \cdot & \cdot & & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 & 0 & 0 & & 2 & \mu_{n-2} & 0 & m_{n-2} & c_{n-2} \\
0 & 0 & 0 & & \lambda_{n-1} & 2 & \mu_{n-1} & m_{n-1} & c_{n-1} \\
n & 0 & 0 & & 0 & \lambda_n & 2 & m_n & c_n
\end{array} \quad \ldots\ldots(8)$$

where $\lambda_j = h_{j+1}/(h_j+h_{j+1})$ $\qquad$ $\lambda_n = h_1/(h_n+h_1)$

$\mu_j = 1-\lambda_j$ $\qquad$ $\mu_n = 1-\lambda_n$

and the quantity c_j represents:

$$3\lambda_j(y_j-y_{j-1})/h_j + 3\mu_j(y_{j+1}-y_j)/h_{j+1}$$

Equations (5) and (7) give as many interpolating points between two adjacent points as may be required. Note that (5) and (7) are only useful for fitting curves that are periodic

($x_0 \equiv x_n$, $y_0 \equiv y_n$). Appendix III contains a computer program which generates spline points by the parametric method just described. Graphical and numerical results of this program are contained in the Appendix.

2.2.3 Non-Periodic Splines

This method successfully produces a properly fitting leg cross-section. The points generated can be used to dictate the movements of a numerically controlled machine tool which carves out the shape (the limb cross-section).

Spline function theory is again applied to fit the limb in the longitudinal direction. The main difference in this technique from that previously described is that the curve is not periodic. i.e. the end point is not identical to the starting point of the data set ($x_0 \not\equiv x_n$, $y_0 \neq y_n$). This alters the matrix equations (2, 4, 6 and 8). In addition, another coordinate enters into consideration, labelled as the z coordinate (Figure 2).

The points used to determine the spline functions in the longitudinal sense will be obtained from the circumferentially generated spline points. The procedure used in this total point production is the following: At each cross-section, a circumferential spline fit is obtained and assume fifty spline points between two consecutive basal (data) points are produced and saved. By using the first spline point of each cross-section, a longitudinal spline fit is obtained. Longitudinal spline points are then produced

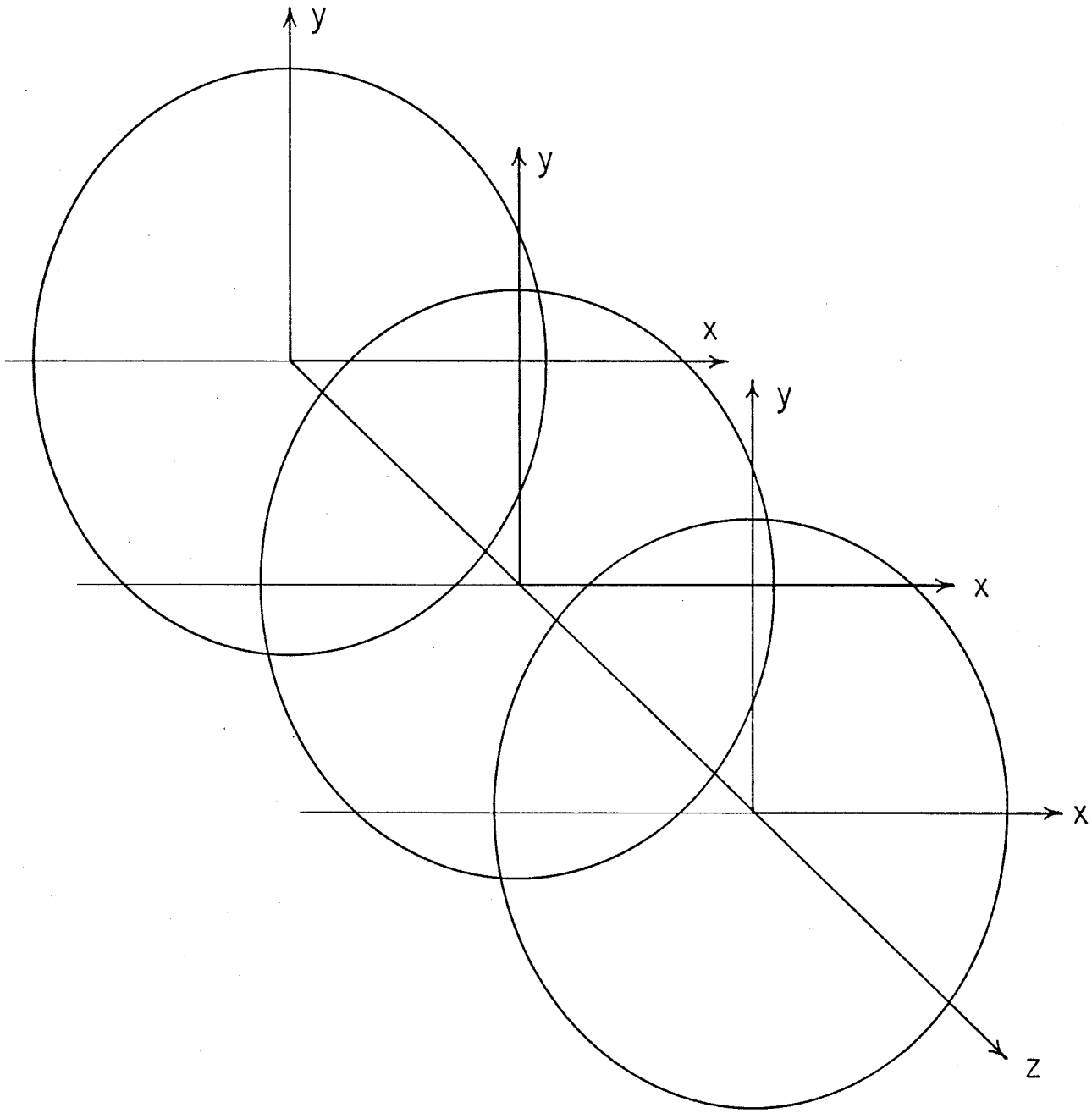


Figure 2 The 'z' Coordinate Consecutive Limb Profiles in the Longitudinal Direction.

along the entire limb length from all the first circumferential spline points. When the entire fifty circumferential spline points are used longitudinally, the next adjacent data point is taken which generates additional circumferential spline points. The process is repeated until the entire limb is reproduced, as shown in Figure 3.

The parametric equations (with parameter s, the cumulative chordal distance) which generate the longitudinal splines, are the following:

For x as a function of s

$$x_{long} \text{ (=x in the longitudinal direction)}$$

$$x_{long} = S_{\Delta}(s) = m_{j-1}(s_j-s)^2(s-s_{j-1})/h_j^2$$
$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2+x^C_{j-1}(s_j-s)^2(2(s-s_{j-1})+h_j)/h_j^3$$
$$+x^C_j(s-s_{j-1})^2(2(s_j-s)+h_j)/h_j^3 \quad \ldots\ldots\ldots \quad (9)$$

($x^C \equiv$ the x's generated from the circumferential spline) where for the nonperiodic case, the quantities m satisfy the relationship:

$$\begin{pmatrix} 2 & \mu_0 & 0 & \cdots & 0 & 0 & 0 \\ \lambda_1 & 2 & \mu_1 & \cdots & 0 & 0 & 0 \\ 0 & \lambda_2 & 2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & & 2 & \mu_{n-2} & 0 \\ 0 & 0 & 0 & & \lambda_{n-1} & 2 & \mu_{n-1} \\ 0 & 0 & 0 & & 0 & \lambda_n & 2 \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{n-2} \\ m_{n-1} \\ m_n \end{pmatrix} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{pmatrix} \quad \ldots\ldots(10)$$

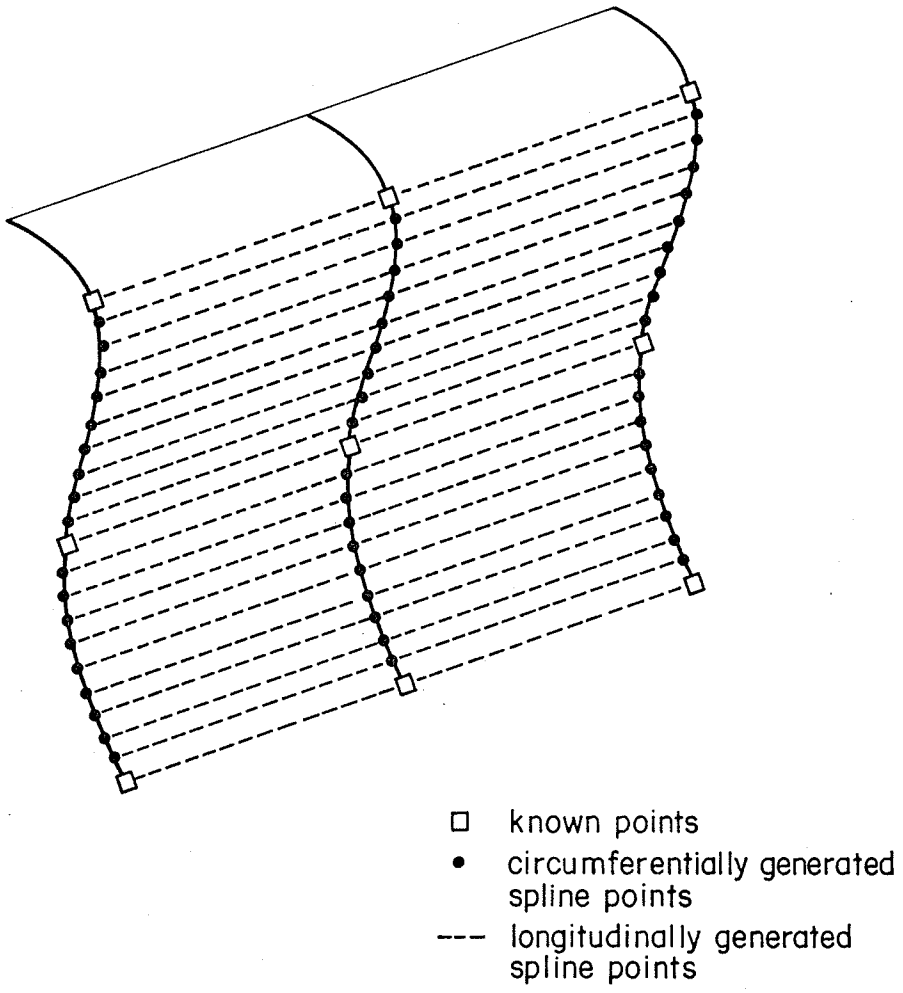


Figure 3 Total Curve Generation Procedure

where chordal distance $h_j = \sqrt{(x^c_{j+1}-x^c_j)^2 + (z_{j+1}-z_j)^2}$

and

$$\lambda_j = h_{j+1}/(h_j+h_{j+1}), \qquad \lambda_n = h_1/(h_n+h_1)$$

$$\mu_j = 1-\lambda_j, \qquad \mu_n = 1-\lambda_n$$

the quantities c_j represent:

$$3\lambda_j(x^c_j-x^c_{j-1})/h_j + 3\mu_j(x^c_{j+1}-x^c_j)/h_j$$

For z as a function of s

$$z_{long} \text{ (=z in the longitudinal direction)}$$

$$z_{long} = S(s) = m_{j-1}(s_j-s)^2(s-s_{j-1})/h_j^2$$

$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2+z_{j-1}(s_j-s)^2(2(s-s_{j-1})+h_j)/h_j^3$$

$$+z_j(s-s_{j-1})^2(2(s_j-s)+h_j)/h_j^3 \quad \ldots\ldots \quad (11)$$

where for the nonperiodic case, the quantities m satisfy:

$$\begin{bmatrix} 2 & \mu_0 & 0 & \cdots & 0 & 0 & 0 \\ \lambda_1 & 2 & \mu_1 & \cdots & 0 & 0 & 0 \\ 0 & \lambda_2 & 2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & & 2 & \mu_{n-2} & 0 \\ 0 & 0 & 0 & & \lambda_{n-1} & 2 & \mu_{n-1} \\ 0 & 0 & 0 & & 0 & \lambda_n & 2 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix} \quad \ldots\ldots (12)$$

where the h_j, λ_j, μ_j, λ_n, μ_n are the same as for equation (10) but the quantities c_j in (12) represent:

$$3\lambda_j(z_j-z_{j+1})/h_j + 3\mu_j(z_{j+1}-z_j)/h_j$$

For y as a function of s:

y_{long} (=y in the longitudinal direction)

$$y_{long} = S(s) = m_{j-1}(s_j-s)^2(s-s_{j-1})/h_j^2$$
$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2+y_{j-1}^C(s_j-s)^2(2(s-s_{j-1})+h_j)/h_j^3+y_j^C(s-s_{j-1})^2$$
$$(2(s_j-s)+h_j)/h_j^3 \qquad \text{..... (13)}$$

where for the nonperiodic case, the quantities m satisfy:

$$\begin{bmatrix} 2 & \mu_0 & 0 & \cdots & 0 & 0 & 0 \\ \lambda_1 & 2 & \mu_1 & \cdots & 0 & 0 & 0 \\ 0 & \lambda_2 & 2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & & 2 & \mu_{n-2} & 0 \\ 0 & 0 & 0 & & \lambda_{n-1} & 2 & \mu_{n-1} \\ 0 & 0 & 0 & & 0 & \lambda_n & 2 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix} \qquad \text{..(14)}$$

where chordal distance $h_j = \sqrt{(y^c_{j+1}-y^c_j)^2 + (z_{j+1}-z_j)^2}$

and

$$\lambda_j = h_{j+1}/(h_j+h_{j+1}), \qquad \lambda_n = h_1/(h_n+h_1)$$

$$\mu_j = 1-\lambda_j, \qquad \mu_n = 1-\lambda_n$$

The limb can thus be duplicated by a numerically controlled machine receiving limited input data. Appendix IV contains a computer program with graphical output written for this thesis based on equations (9) through (14). The program essentially operates as a data multiplier by creating points between the input data providing sufficient shape definition required for machining.

2.2.4 Clustering

The procedure developed in a previous section reproduces curvatures and contours with a minimum of actual data points. With reference to Appendix III, note that no attempt has been made to reduce the number of data points required to adequately fit the limb profile. For efficient use of computer core area, it is advantageous to minimize the number of points stored in order to reproduce a shape. Consequently, a method of clustering was devised.

Consider Figure 4, which contains a large number of points, typically obtained from a field sample. Assume that a tolerance limit of 1/16" for reproduction has been established.

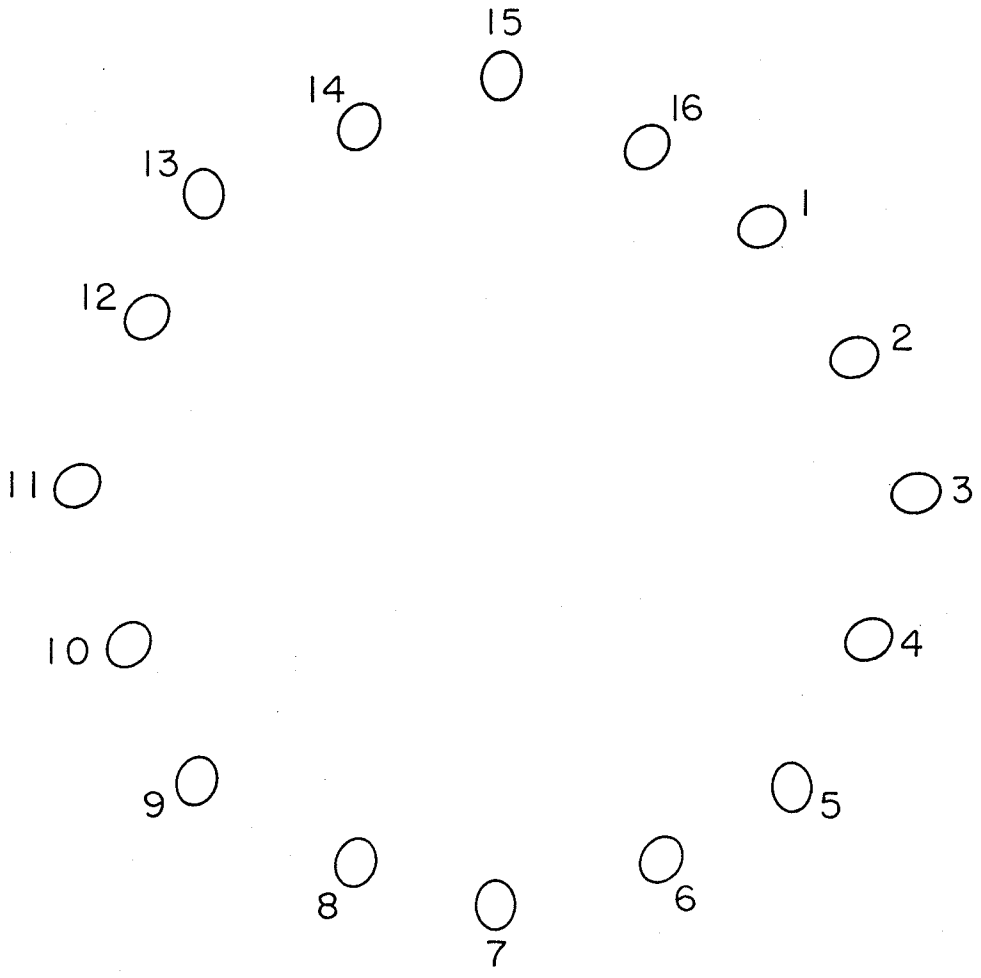


Figure 4. Field Data Set

Assuming that point 1 in the Figure was eliminated, the distance from point 1 to the spline curve generated from the remaining point set is less than 1/16", then point 1 may be dropped, and the specified tolerance is maintained. However, if the distance between the spline curve and the point is greater than 1/16", then the point may not be omitted from the data set.

This describes the concept of clustering. The effect of removing individual points from the data set is a function of the tolerances required for shape duplication accuracy. If the tolerance is made more critical (say 1/32"), the total number of required points will increase. Conversely, relaxing the tolerance (say to 1/8") will decrease the number of points.

In addition this method examines the effect of curvature on neighbouring and non-neighbouring points. If point 1 was dropped (Figure 5), and the spline curve generated fell within tolerances, the point would be discarded. If another spline curve was generated which omitted both points 1 and 2, and the tolerances are not met at either of the points, then point 2 must be included in the data set. In this way, the effect of curvature at neighbouring points is examined.

Continuing to suppose that point 1 has been dropped, point 2 examined and kept, one now examines point 3. This time, not only is 3 checked for tolerance, but also point 1 (the discarded point). Again, if the tolerance at either point 1 or 3

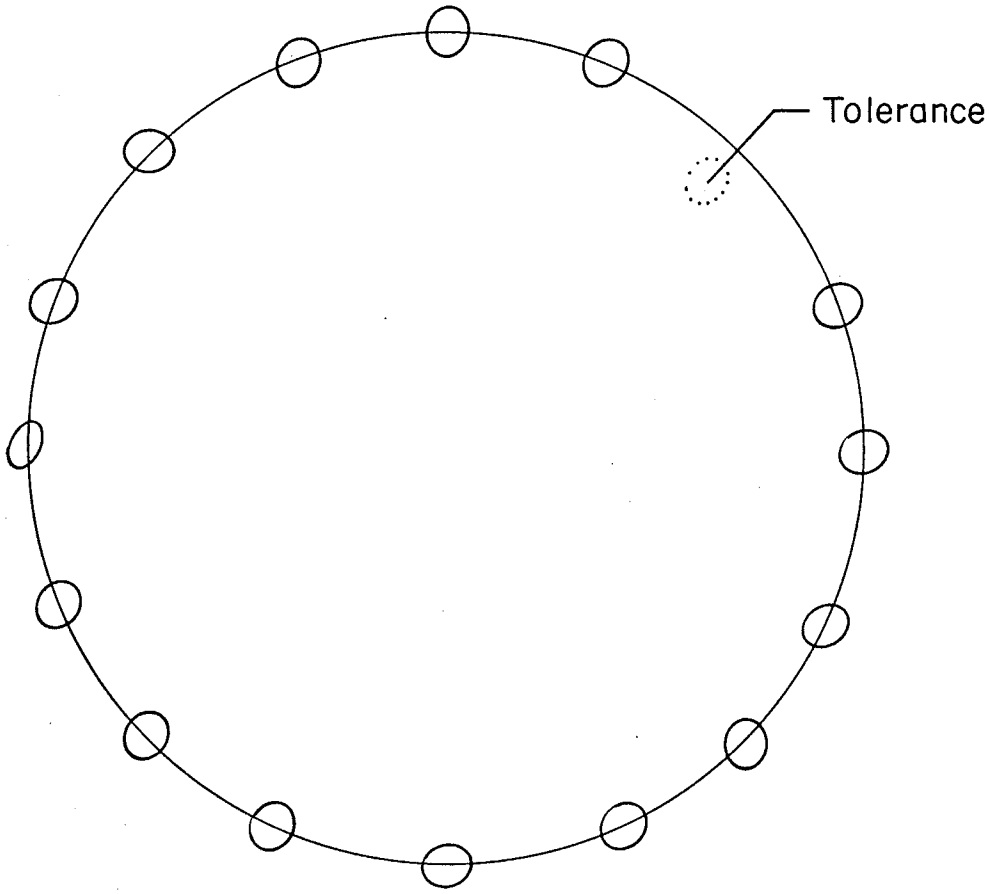


Figure 5. Distance-Tolerance Criteria

is not met, then 3 must be retained in the set. Consequently the effect of curvature at non-neighbouring points is checked.

As described it is possible to reduce a data set from a "large" number of points to a "small" set, depending on the tolerances. The computer program developed for purposes of clustering and the algorithm outlined above are presented in Appendix V.

Clustering is highly critical in the data sampling and storage process, as it dictates the minimum number of data points to be stored. A type of dichotomy is now evident in the sampling procedure; i.e., depending on end usage, a large number of points may be initially sampled and stored (to be later compared), or only the minimum crucial point locations may be sampled and stored (the crucial locations being found by experience with many limb shapes). For the present, a large number of image points must be sampled and then processed to determine the minimum set.

With reference to Figure 6 a minimum set (x, y) is determined for every slice along the limb (z direction). However, for each slice every (x, y) point may not have a corresponding member above or below (in the z direction). In order to generate longitudinal spline functions, it is necessary to have this "slice-to-slice" correspondence. By means of a circumferential (periodic) spline function, the "missing" point(s) on any particular z slice is defined. By generating longitudinal splines using these newly defined points the entire limb shape may be reproduced (Figure 7).

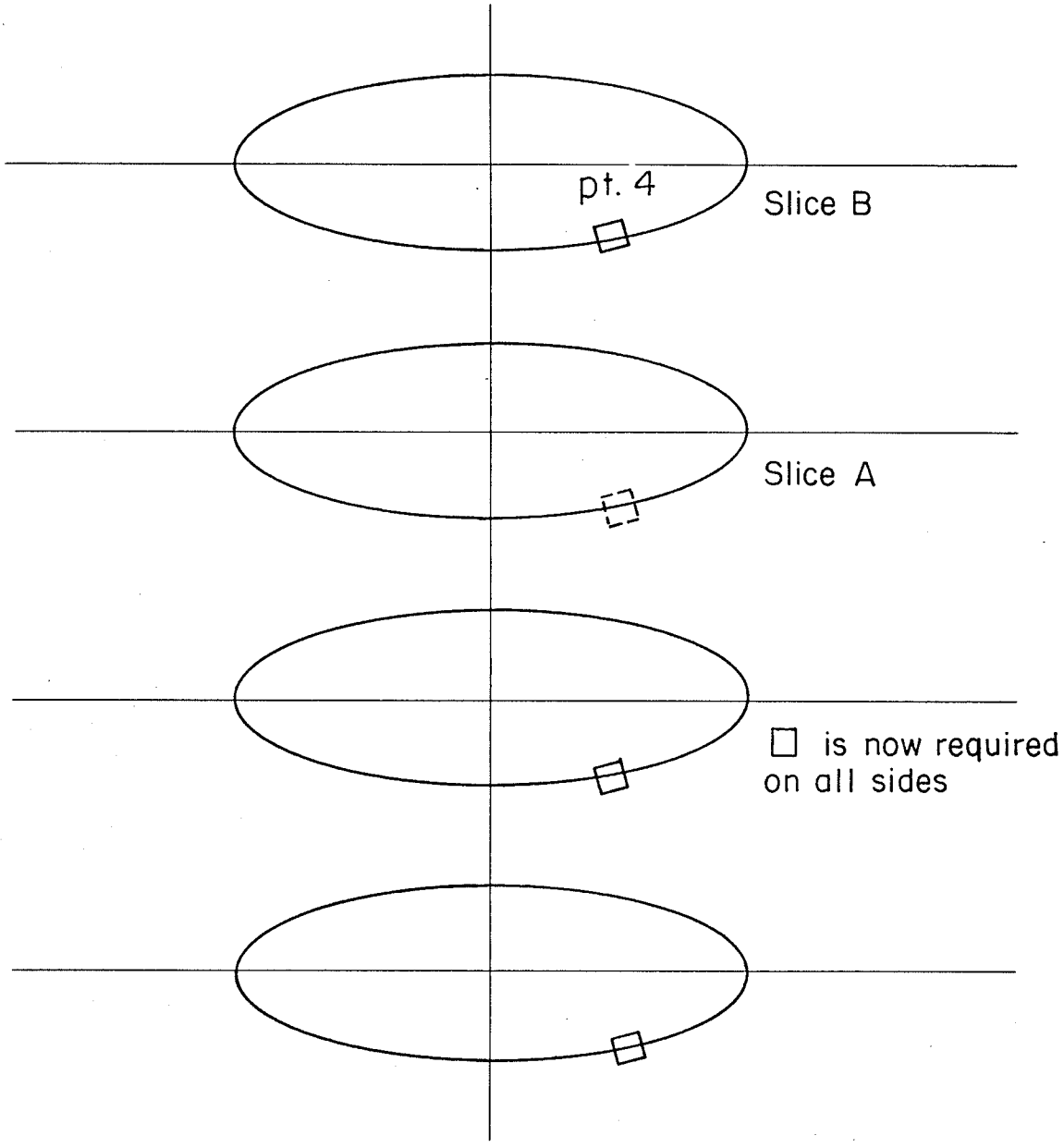


Figure 6. Point Inclusion

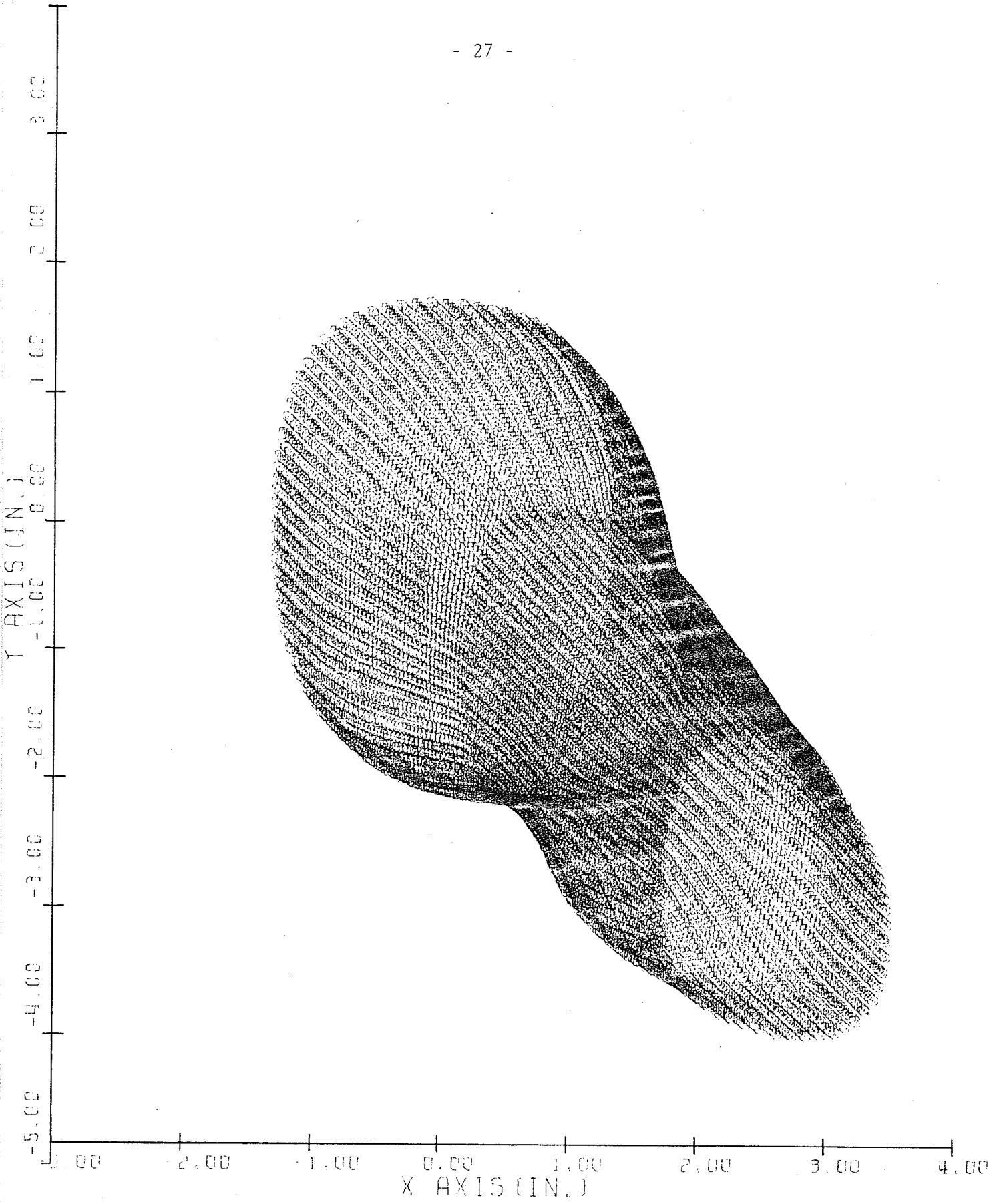


Figure 7. Reproduced Limb

An alternative method is to maintain corresponding (x, y) sets on each of the z slices. However different curvatures along the limb will lead to different clustering configurations. The various clusterings, when taken in an all-inclusive fashion, will generate a large number of required data points. This is not conducive to minimal data point storage and hence clustering.

The preceeding describes the adopted clustering procedure which results in the storage of the smallest possible data set required to correctly describe the desired shape.

2.3 COMPUTER CAPACITY

The preceeding method allows the use of a small, less expensive computer facility to generate required spline routines. In particular, the entire system is designed for use with a Hewlett-Packard Model 2116 C Computer, a small, general purpose digital machine, capable of performing a wide variety of computational assignments. Additional uses are in process control, media conversion and data reduction. The overall system proposal makes use of all the above capabilities.

The present computer system includes the following options and peripherals:

i) a fast paper tape reader (500 characters per second with associated interface card)

ii) a time base generator card

iii) a teletype telex (and associated interface card)

iv) a teletype telex unit with graphical display capability (with associated interface card)

v) a data source interface card

vi) an analogue to digital converter card

All cards refer to circuit boards which condition the signal from devices so as to be valid for use by the computer.

This particular system has 8196 words of core storage in the mainframe (expandable to 32K). Important standard features on all 2116 C computers include a 16 bit word size, a 1.6 microsecond memory cycle time, a set of 70 instructions with which to operate the computer, and two addressable accumulators. In addition, there are modular input-output (I/O) drivers, and a priority interrupt system for independent device servicing and programming.

Limitations on the core size of the machine restricts program complexity and data storage. By using spline functions and the data reduction procedure, it is possible to reproduce the limb shapes within the 8K system constraint.

The described method of data reduction allows the production of spline points using the conventional conversational BASIC language. Virtually an infinite number of spline points can be produced, depending on requirements of tolerances. The purpose is to demonstrate a rather elegant method of reproducing a curve shape, even when the compiler is rather large. Appendix VI outlines a

BASIC program which produces spline points plotted in Figure 8. This demonstrates the simplicity of the shape-definition method, inasmuch as the program can be fit and executed on an 8K machine, of which 5½ K is devoted to the compiler for the program language.

It is possible to create an entire limb shape by using the BASIC program and stacking the generated limb profiles one on top of the other. However, the method as described in Section 2.2.1 was chosen for use, as it was deemed to be least redundant in the calling of spline routines.

Having dealt with the problem of curve fitting, and related computer capacity, the process by which the generated numbers are transformed into tool head movements is described in the following section.

2.4 MANUFACTURING SYSTEM REQUIREMENTS - NUMERICAL CONTROL

The broadest definition of numerical control is "a process accomplished by symbolic programming and symbolic control" (4). Having reduced the computer requirements for mathematical shape definition, the numerical control process determines the overall expense of prosthesis production.

A block diagram approach to the N/C (numerical control) process is given in Figure 9. The Machine Control Unit (MCU) sends signals to the Machine in order that the tool will perform certain defined (programmed) tasks. The Machine in turn sends signals to the MCU indicating the tasks being performed.

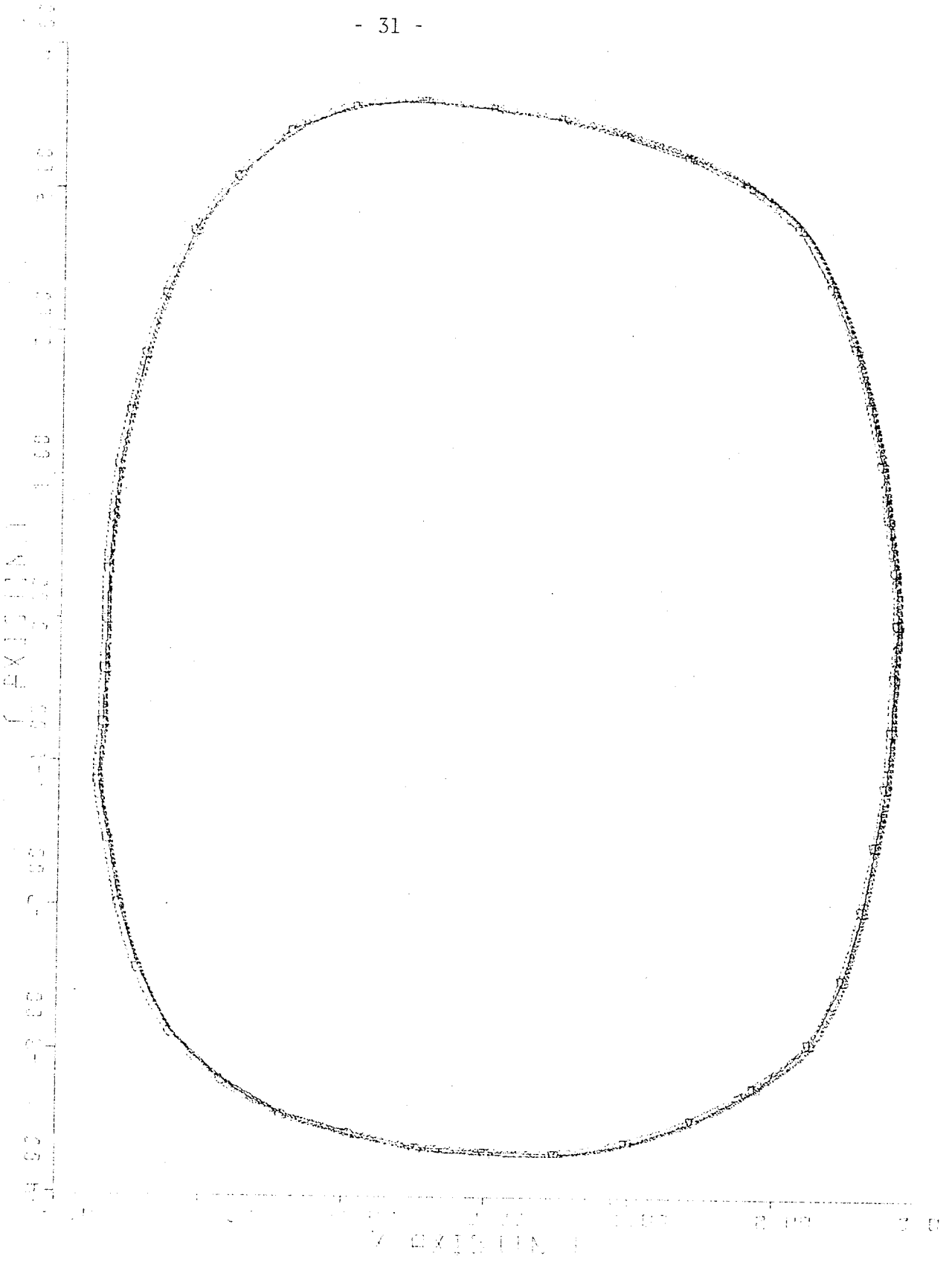


Figure 8 Spline Points Generated from Basic Program

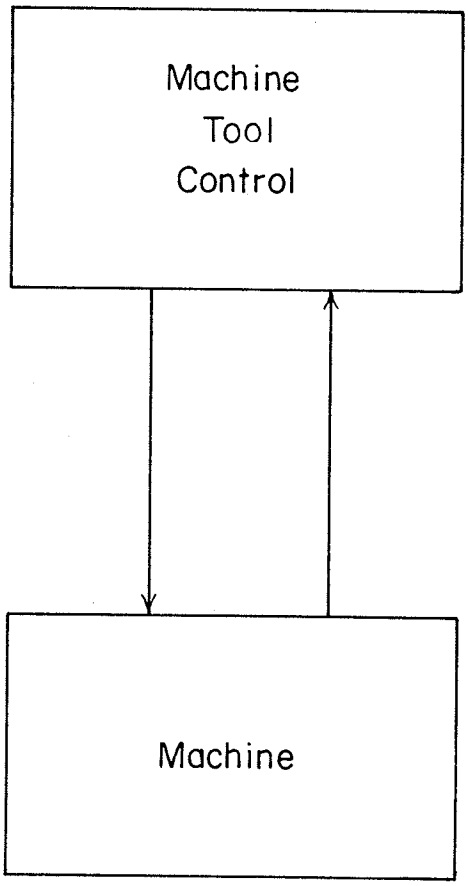


Figure 9 Numerical Control Block Diagram

A Machine Control Unit may be defined as "the combination of circuitry and drive motors that guide a machine table or spindle along a programmed path" (4). Within this limit is included the interpolation necessary to guide the Machine from point to point along the path defined by the splines. This programmed path defines the workpiece cuts.

For any part to be produced, various instructions are introduced into the MCU. These instructions to the MCU must completely define the component to be manufactured. This is accomplished by using one of many numerical control programming languages. The most advanced program language is APT (Automatically Programmed Tools) (3). Using APT, the programmer enters a program into a computer having an APT compiler. The compiler then operates on the submitted APT program producing a code which is then read into the Machine Control Unit. The MCU controls the machine tool and the required component is produced.

There are various types of numerical control machines available which are not necessarily controlled by identical codes. In addition to the large number of machines, there are various N/C compilers. Compatibility must be maintained by the various compilers and machines.

Compatibility is achieved with the implementation of a postprocessor. The postprocessor provides the interface between the particular numerical control programming language employed and the particular machine on which the part is to be manufactured. The

postprocessor is a computer program which modifies the code from the machine tool compiler language into a form that will fit the particular machine tool. Figure 10 illustrates the N/C process using a post-processor.

The numerical control process as outlined in Figure 10 requires a minimum of 52K computer storage to accommodate the APT compiler. This would render the process uneconomical for the average workshop.

Consequently, it is desirable to eliminate any potentially unnecessary interfaces between dimensional information and finished product. Numerical control operations which employ a postprocessor to furnish interpolated control points are in essence two steps out of sequence from the computer which originally provided the input information. Allowing the computer to directly control the machine tool movements eliminates all superfluous interfaces. This type of process is called direct numerical control.

The following section describes the implementation of direct numerical control using the Hewlett-Packard 2116 C computer.

2.5 USE OF THEORY - SYSTEM SYNTHESIZED

The final equipment configuration settled on is a computerized direct numerical control system which makes use of a 2116 C Hewlett-Packard computer. Refer to Figure 11 for the following discussion.

The sensing cylinder is placed around the patient's limb and activated (5). The limb is then removed and each probe is sampled

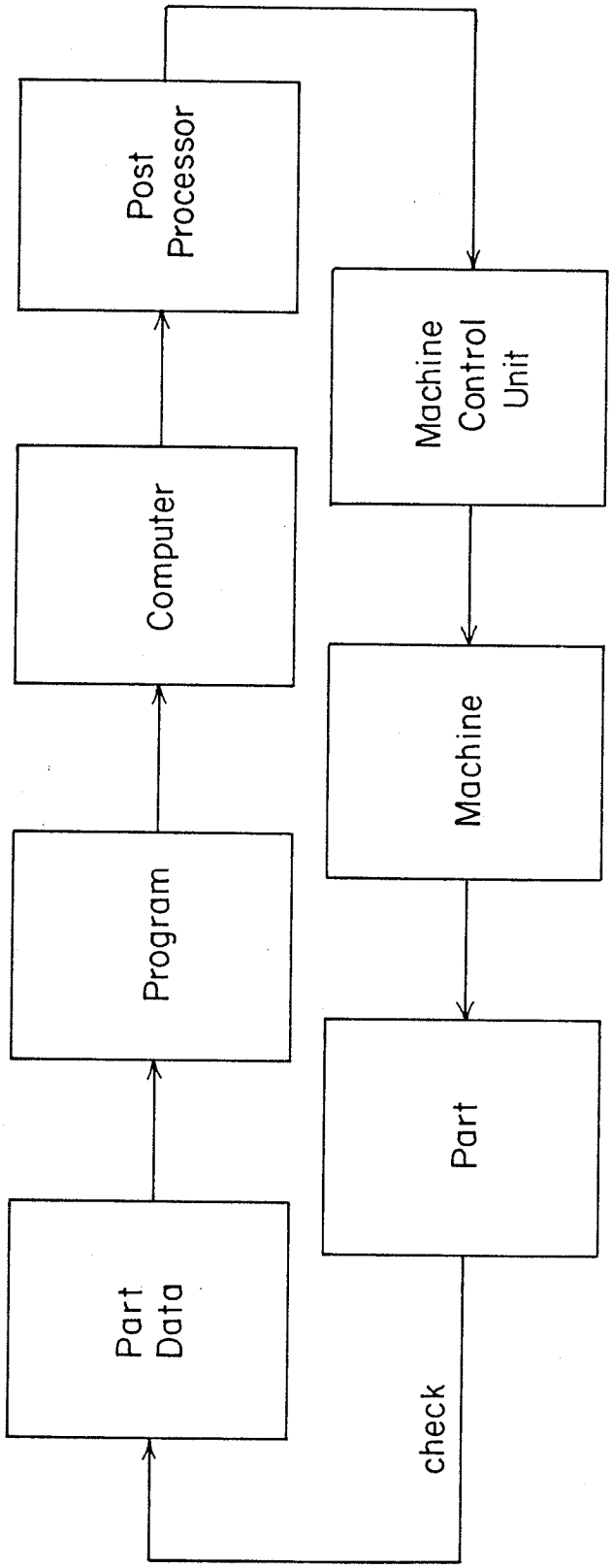


Figure 10. Numerical Control Process

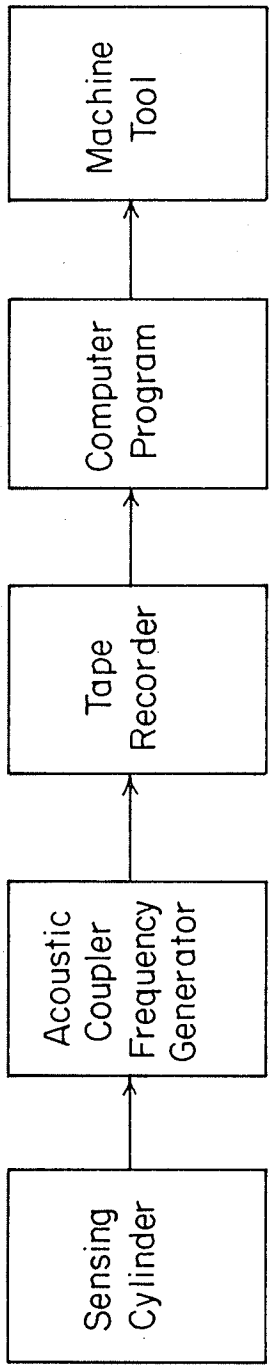


Figure 11. System Block Diagram

in the following fashion: An arbitrarily indicated probe is taken as a reference position. An acoustic coupler is used to generate a particular frequency which corresponds to the displacement of the probe. A tape recording is made of this specific frequency for five seconds and then both tape recording and frequency production are halted. The next probe is then sampled and a suitable frequency generated and recorded for an additional five seconds. This procedure continues until all probe positions on the highest "slice" are recorded. Fourteen probes are measured in this fashion. This is the minimum number in order to obtain dimensional accuracy without implementing clustering techniques.

The acoustic signals so produced represent the limb shape. The method the computer uses to digitize the signals to dimensional values is as follows: The computer starts the analogue-to-digital converter (ADC) 2½ seconds after the tape recorder begins to play back the frequencies. This ensures that the ADC will be sampling in the middle of each probe record. The ADC determines how many cycles per second were produced by each probe, and by use of a conversion factor, the proper dimensional values of radial distances are stored. The angular probe positions have been pre-stored. Program I in Appendix VII performs the frequency conversions. By means of the computer program described in Appendix IV, the computer produces spline points which describe the entire limb shape in longitudinal segments. Two time base generators are required, the first to initiate an analogue to digital reading every five seconds,

the second to measure the frequency corresponding to the specific probe displacement. Finally, a digital to analogue converter outputs voltage signals (proportional to the spline points) to three power sources, which in turn control machine tool movement. Program II of Appendix VII performs the signal output function.

This completes the description of the overall system.

3 DISCUSSION OF RESULTS

One of the prime aims of this thesis was to develop a technique which would accurately reproduce a limb cross-section from a given data set. To accomplish this, a number of curve fitting techniques were attempted. These techniques with their associated limitations are now described.

Sine Functions

Visual inspection of a cross section through the thigh area (Figure 12) suggests that if each half of the cross-section of the thigh was considered separately, sinusoidal functions could be applied. To assure a smooth fit, the end points of each half could be matched using a simple iterative technique. The preceeding assumption proved valid for all shapes which were naturally sinusoidal. However, small deviations from sinusoidal form were impossible to replace regardless of how the coefficients of the oscillating function were altered. Typical reproductions of the various limb cross-sections using the sine function technique are illustrated in Figure 13. As shown, this method was unsuitable for curve fitting within the specified limit of 1/16" deviation, for any non-sinusoidal shape encountered along the limb cross-sections.

Circular and Elliptic Functions

Although circular and elliptic functions were not limited

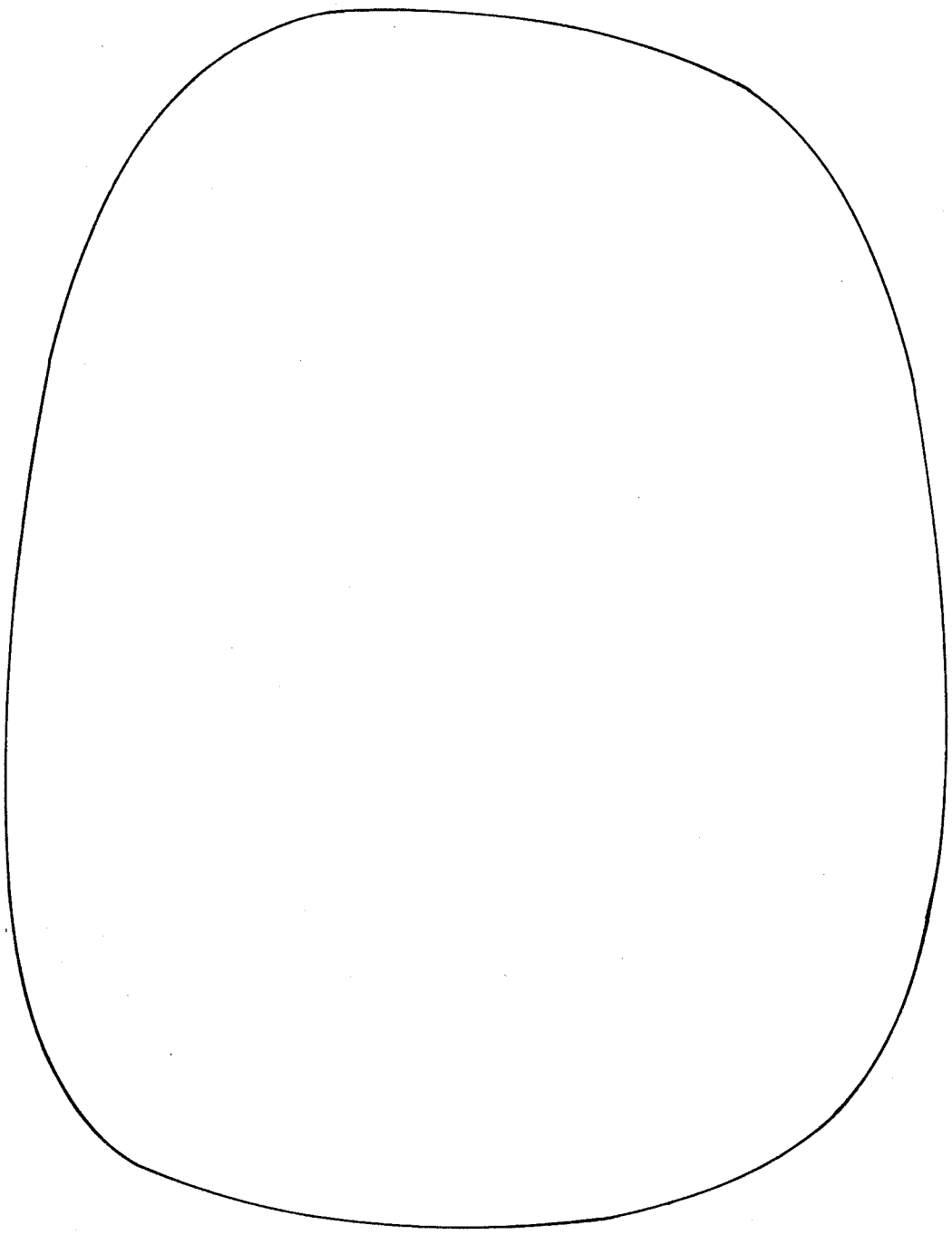


Figure 12. Thigh Cross-Section

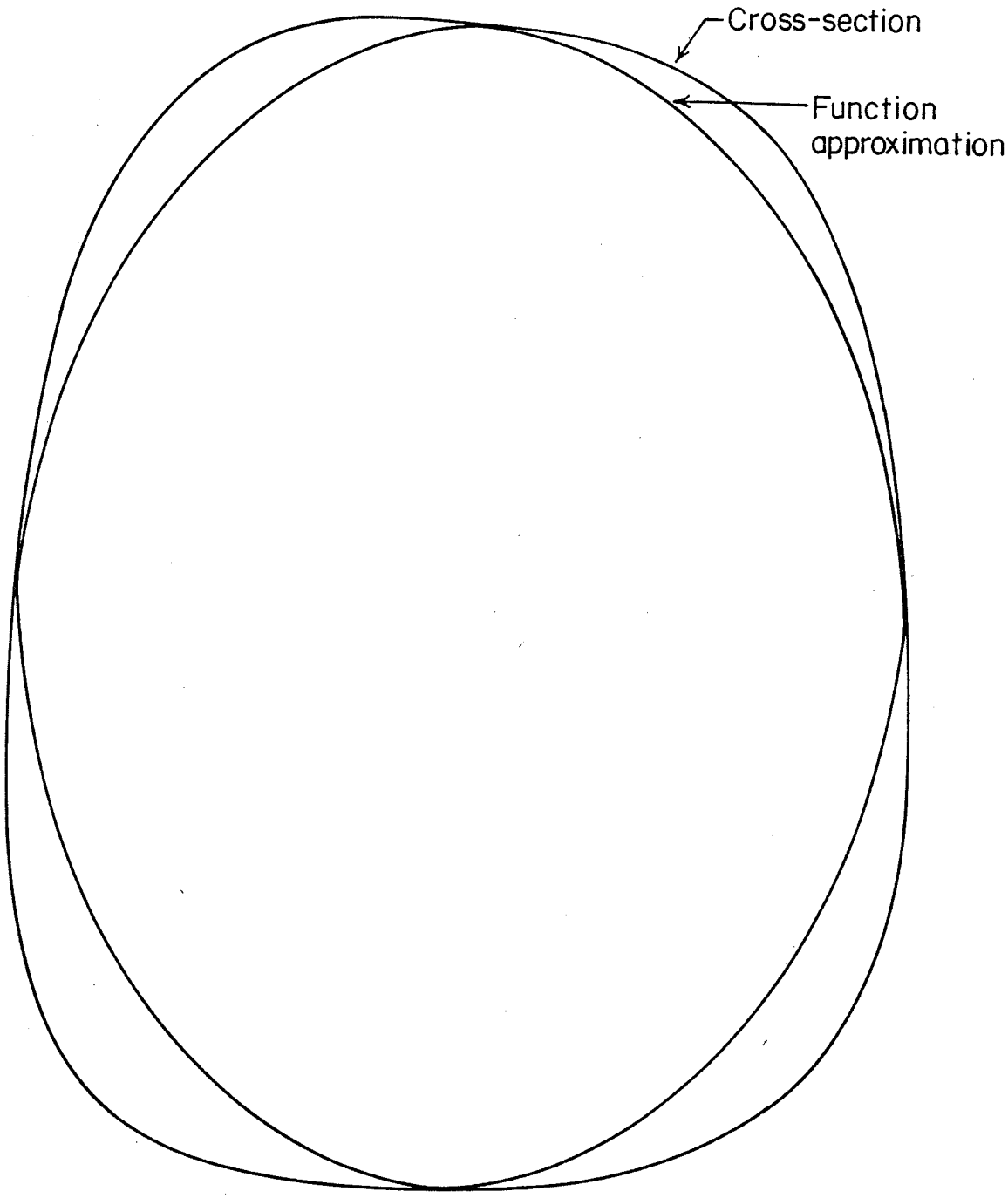


Figure 13a

Sinusoidal Function Fit-Thigh Cross-section

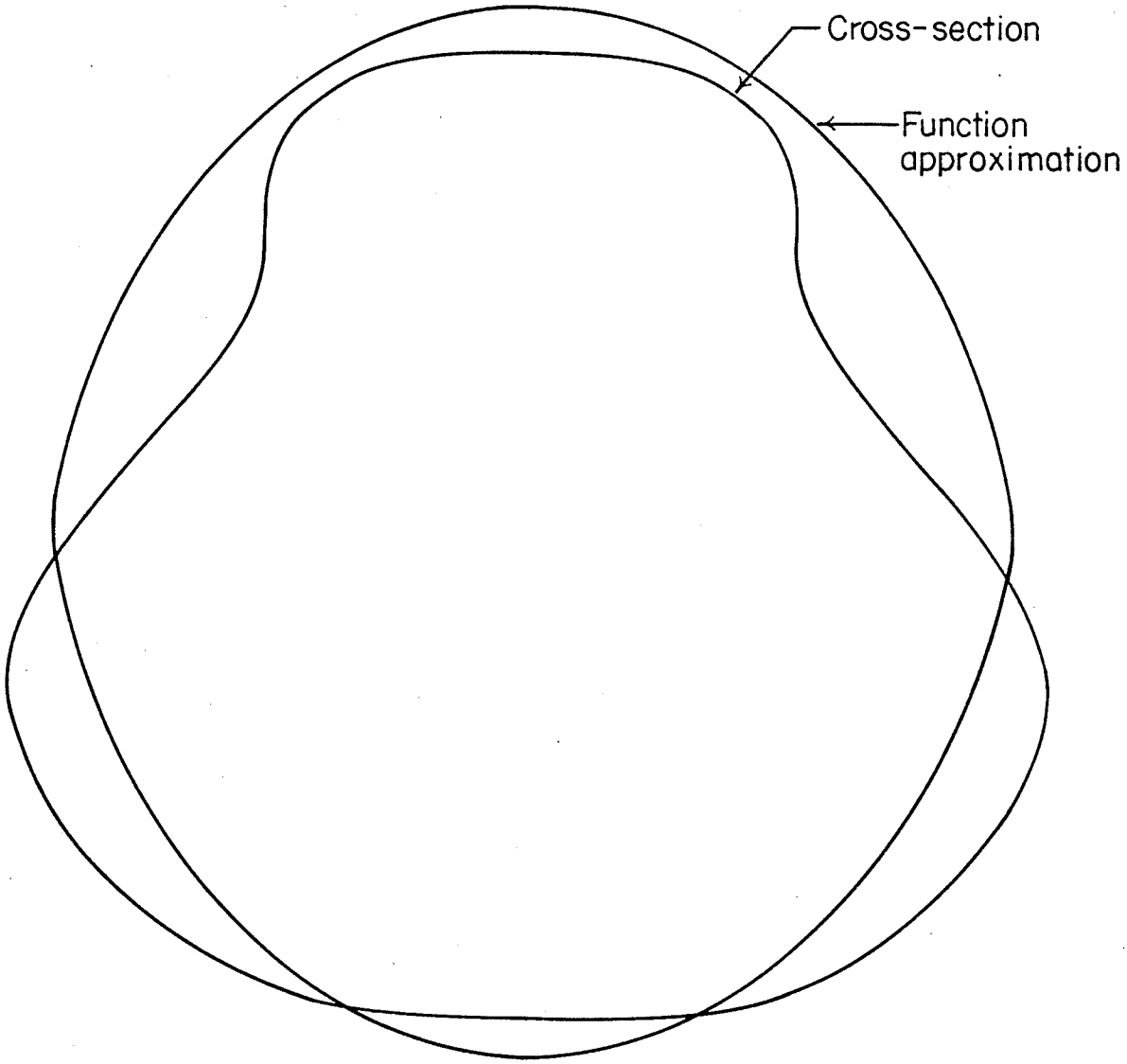


Figure 13b

Sinusoidal Function Fit - Knee Cross-section

to a specific shape as were the sinusoidal functions, only cross-sections of the limb having near constant curvature could be replicated. Using these functions, cross-sections of the thigh area could readily be reproduced within the specified tolerance (Figure 14a). However, the functions proved unsuitable for cross-sections in areas such as the knee region, where rapid changes of curvature occur (Figure 14b).

Polynomial Functions

As in the previous two methods, this fitting technique was applicable for cross-sections having near constant curvatures. Given the general nature of polynomial functions (i.e. constantly increasing value), it was apparent that the entire cross-section could not be described by one function. By dividing the cross-section into four segments and establishing continuity at the ends of each segment through an iteration procedure, it was found that all cross-sections of relatively constant curvature could be reproduced within the specified tolerances (Figure 15a). Again, the cross-sections in the knee area where gross curvature variations occurred, could not be descirbed by this technique (Figure 15b). Recognizing that small segments were required in areas of severe curvature changes lead directly into the use of splines.

Splines

With reference to the human limb, these functions were

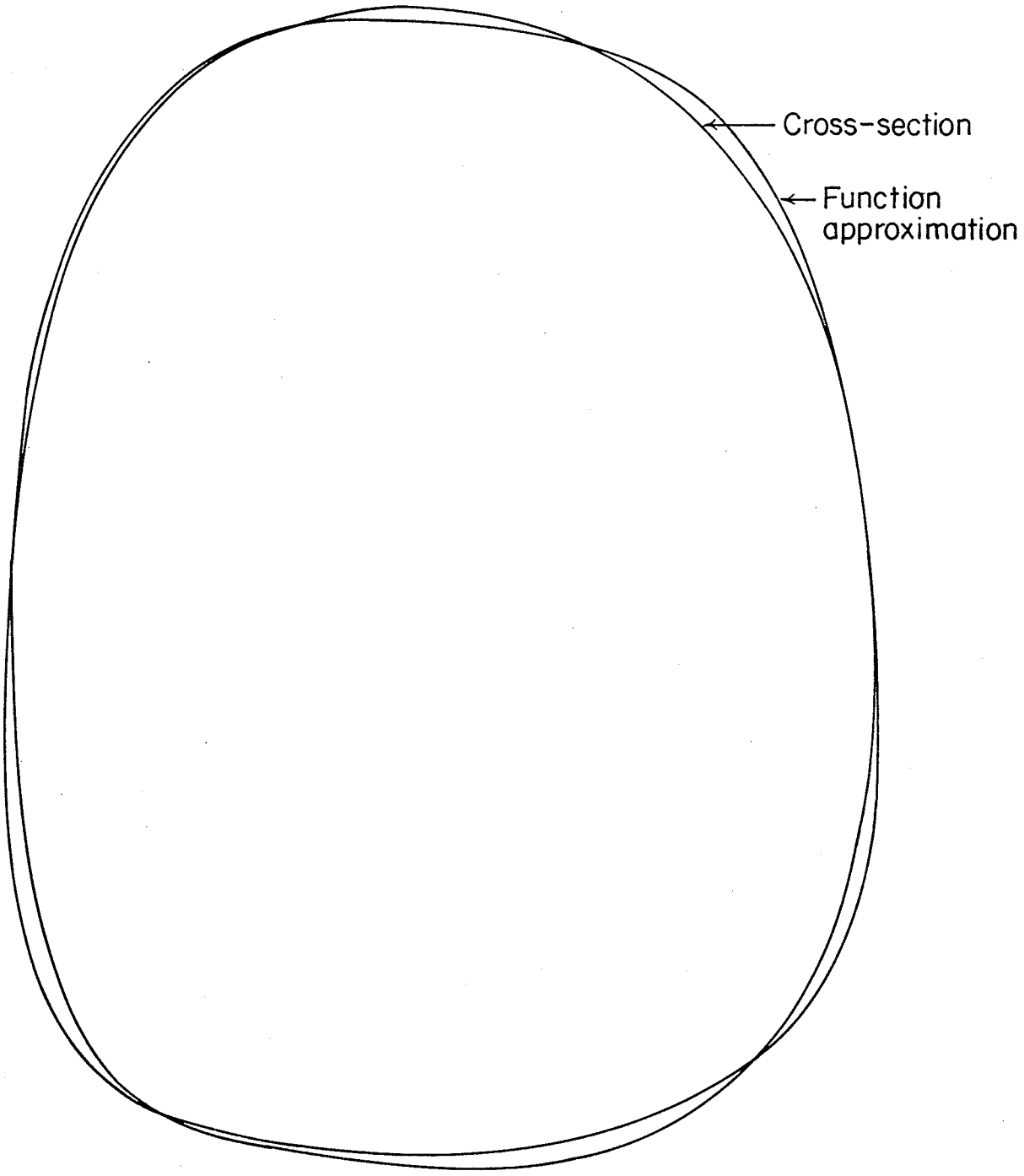


Figure 14a

Elliptic Function Fit-Thigh Cross-section

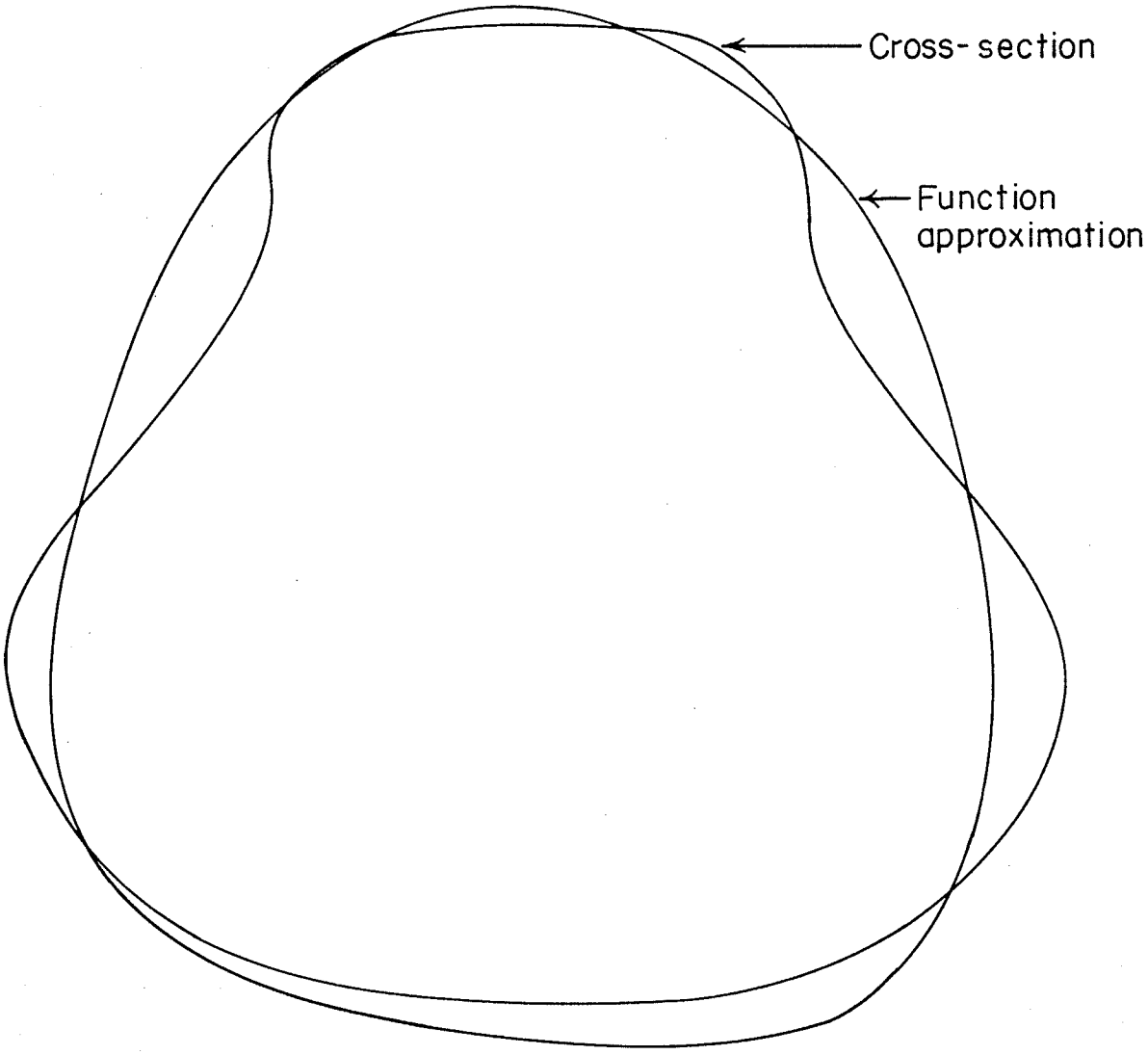


Figure 14b

Elliptic Function Fit- Knee Cross-section

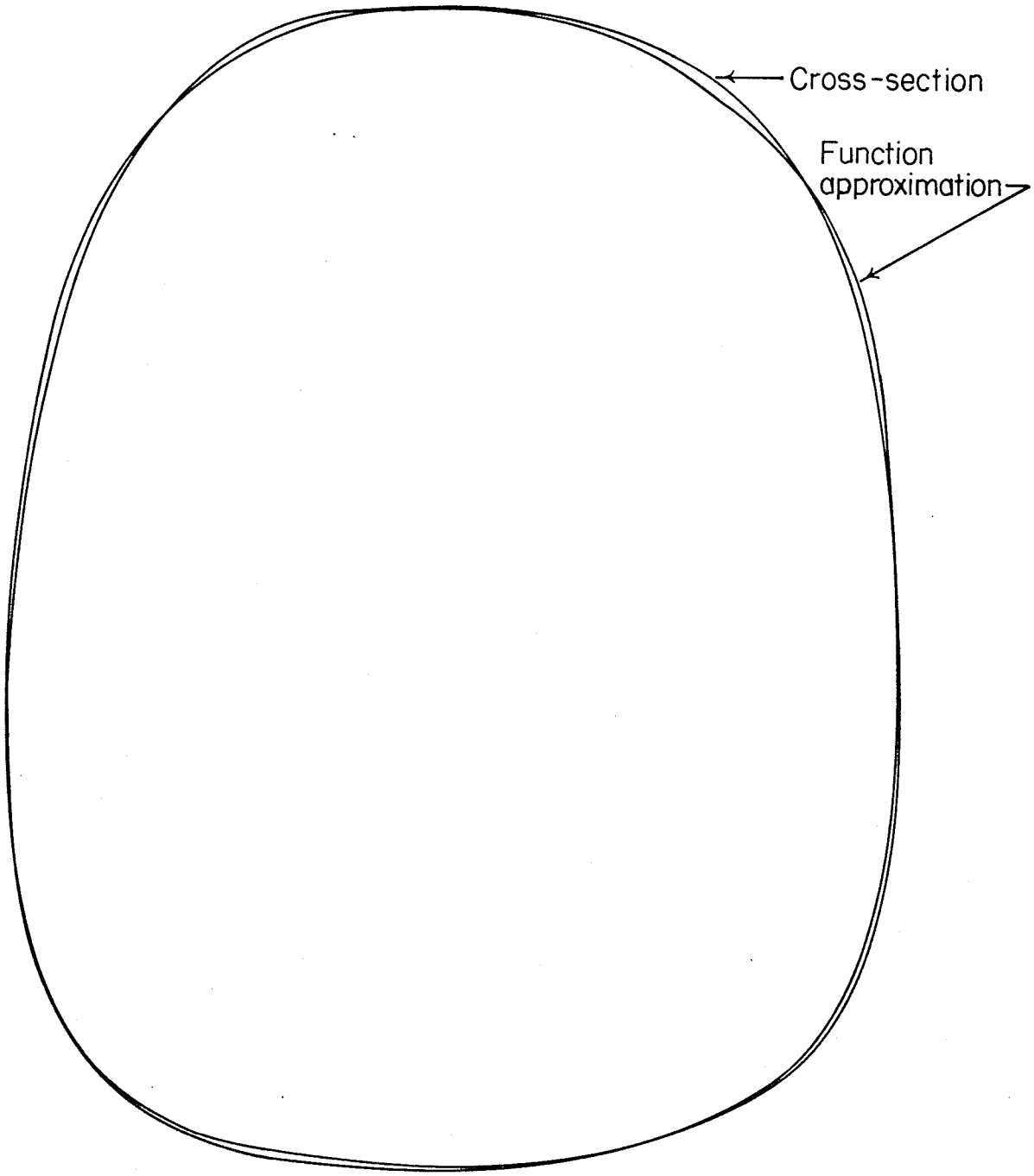


Figure 15a
Polynomial Function Fit – Thigh Cross-section

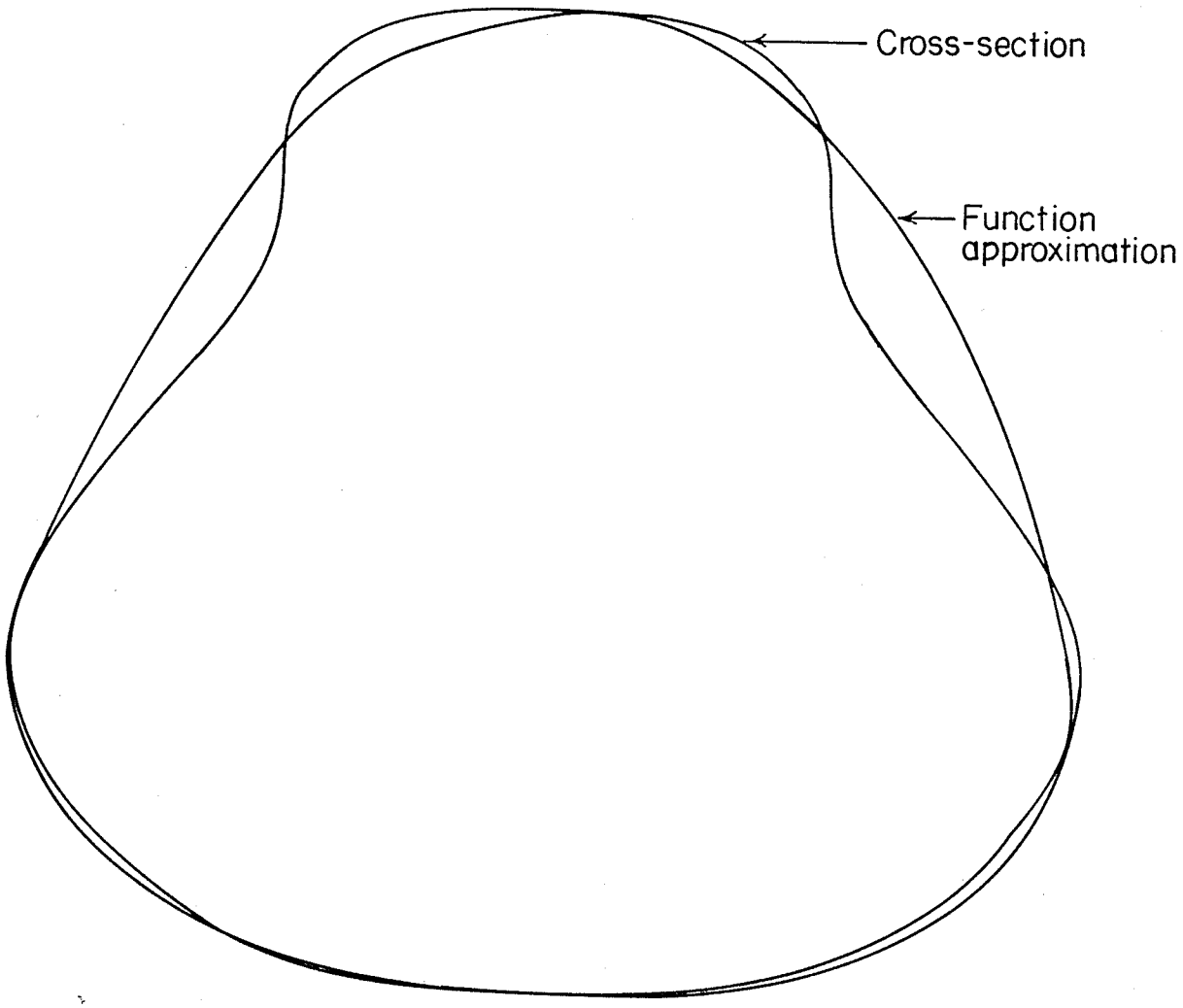


Figure 15-b

Polynomial Function Fit — Knee Cross-section

found to give the best fit to the various cross-section curvatures encountered. Figure 16 indicates the type of reproduction that is possible to obtain using this technique. In this illustrated cross-section, the data points are obscured by the spline points. The spline function fits well within the tolerance over the original curve. As shown in Figure 17, the spline function accurately fits shapes of rapidly changing curvature. The spline function is virtually superimposed over the original shape.

The distinct advantage of spline functions over the previously mentioned curve fitting techniques is that only a relatively small number of data points are required to define the function, such that the function will accurately fit a cross-section. Due to their specific characteristics, the previously described methods require a large number of points to properly fit a cross-section. In addition, the previous methods operated over large segments, whereas spline functions operated over small segments, and hence did not require iterations at segment boundaries.

Another primary objective of this thesis was to keep the system size to a minimum. This was accomplished by saving only the data points that were essential to accurately replicate the shape. This resulted in concentrations of data points at locations of rapidly changing curvature, and a sparse collection in areas of relatively constant curvature. The clustering technique described in Section 2.2.4 was made possible through the exclusive use of spline functions. Since the previous methods required a large

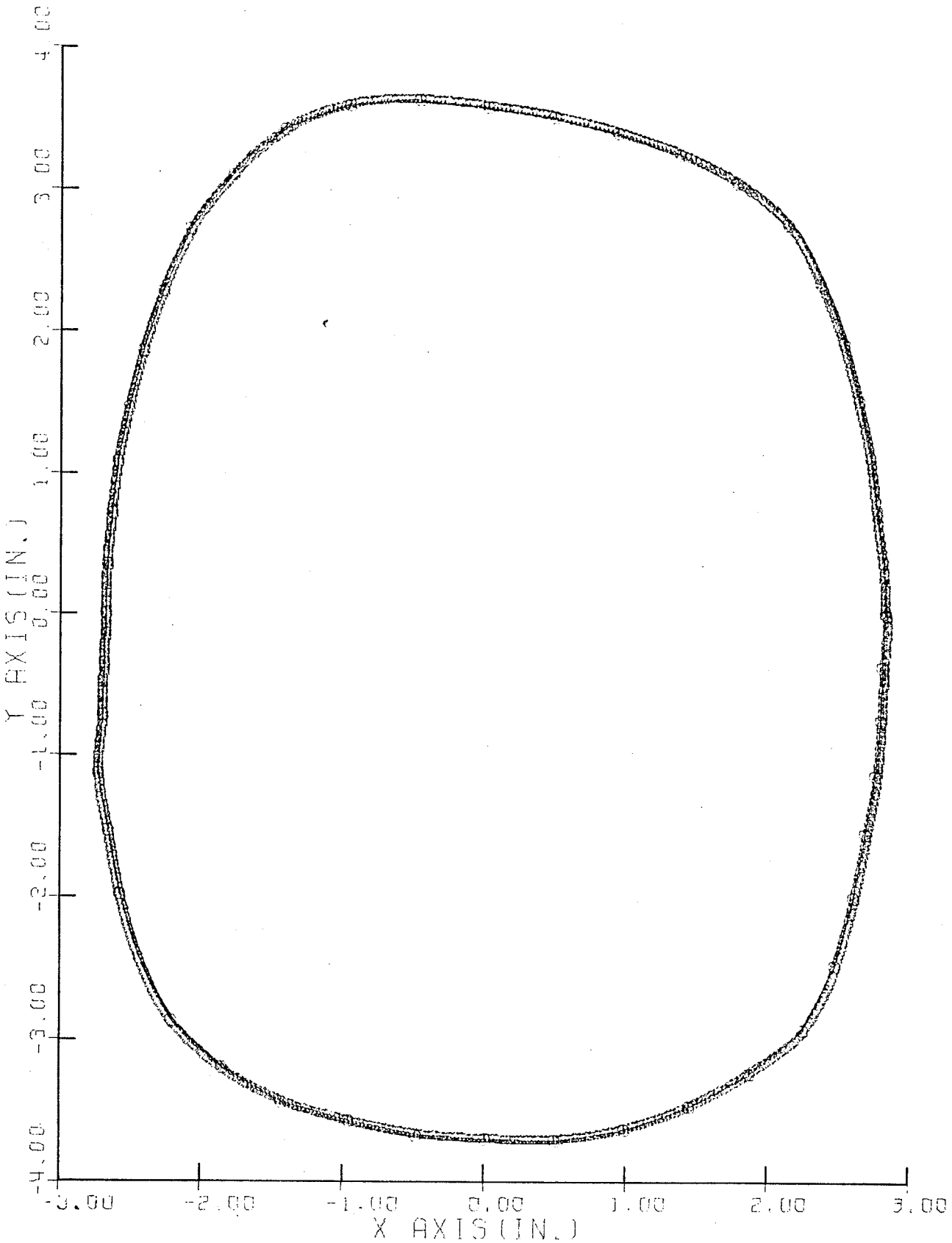


Figure 16. Spline Function Fit Thigh Cross-Section

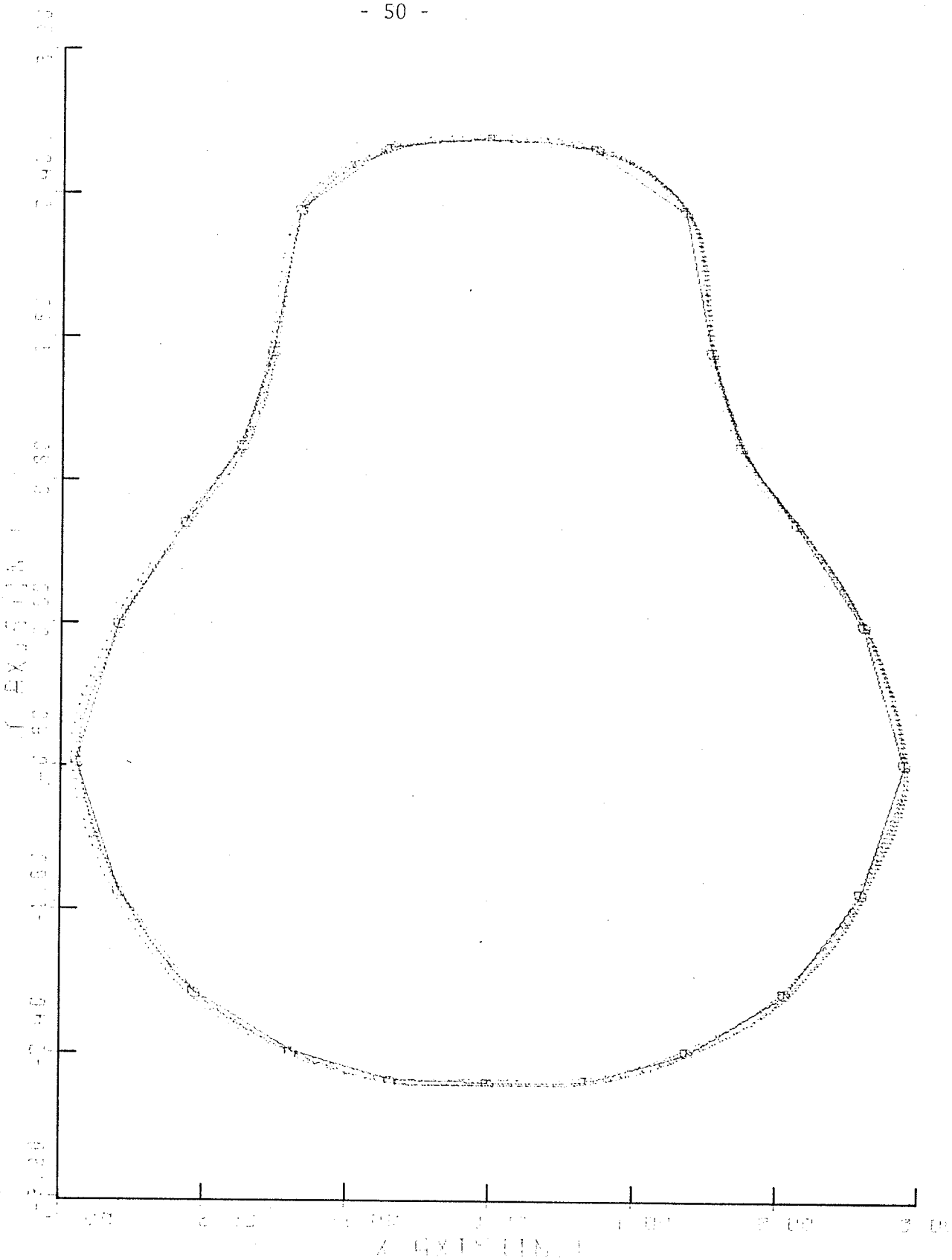


Figure 17. Spline Fit in Knee Region

number of data points for reasonable shape replication, the omission of any of these data points was impossible. Thus clustering was feasible only with the implementation of spline functions (Figure 18).

With reference to the accuracy of the limb cross-section by the spline technique, it must be concluded that the limb cross-section configuration is a spline function. Splines are by their nature minimum energy functions (7), leading to the conclusion that the limb cross-section is one of a minimum energy configuration. The human body (8) observes minimum energy principles in its mechanism of movement. Results obtained here confirm this observation in terms of limb configuration.

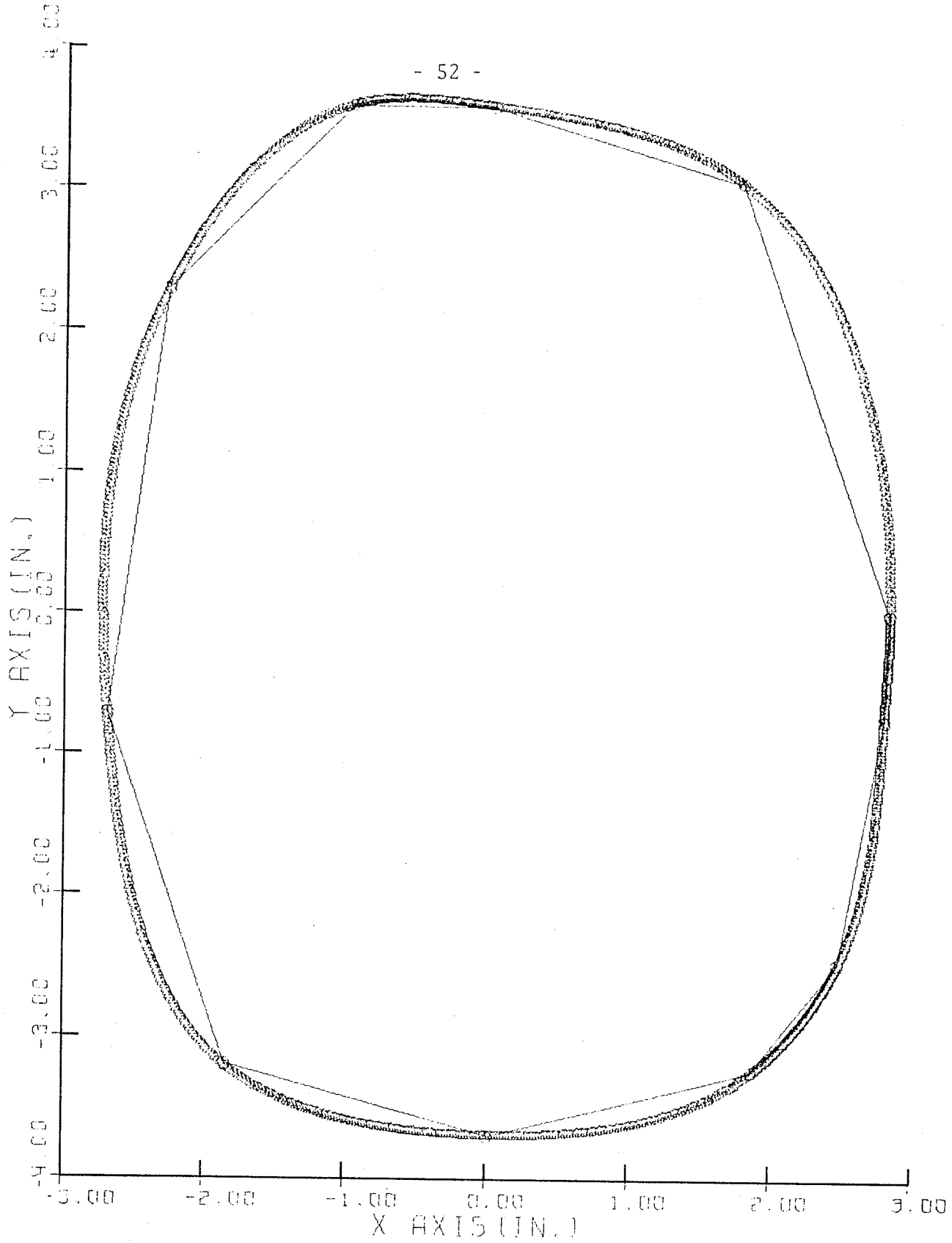


Figure 18 Clustering with Spline Functions

4 CONCLUSIONS

This investigation has shown that two main contributions can be made to the field of shape reproduction. The first contribution is the rapid, accurate reproduction of a limb shape done strictly by computer and machine, requiring virtually no human intervention in the actual machining process. Secondly, what would originally have been a sophisticated numerical control complex is replaced by a single dedicated computer.

In addition, a method of data point reduction has been developed that allows only a small number of critical points to be stored in order to duplicate complicated surface shapes and features. This would be an ideal method of storing and performing statistical analysis on anthropometric data for any future artificial limb studies. Standard limb forms could possible be obtained such that with a few individual changes made to the standard, the production of individual limbs would be even further speeded up, and the cost further reduced.

Finally, the system cost has been reduced to a minimum. Cost reduction was made possible through the exclusive use of spline functions and the clustering technique devised for this problem. By implementing these procedures on a small computer, this kind of system now falls within the price range of the small shop or research group.

APPENDIX I

The most important area of this proposed new system is the generation of smooth curves representative of limb cross-sections. These generated curves must give suitable accuracy when interpolating between the given points in two orthogonal directions. A more detailed explanation of the origin of these data points is now presented.

Assume that the shape-sensing package is at the location and is ready to probe the object (5). The limb (object) is placed in the device (Figure A1-1) and the device is then activated. The limb is then removed and the reading of each probe in the device is taken in turn. This is done for each section around (and down) the instrument (Figure A1-2). The readings given from each probe will only be in terms of a voltage signal difference from the undisturbed probe position. The undisturbed position is such that the tip of the probe is at the geometric centre of the device (Figure A1-3). Knowing the probe angle (from some predetermined zero angle position), it is easy to calculate the relative position of the object with respect to the ring geometric centre using the relations:

$$x = r\cos(\theta) \qquad y = r\sin(\theta)$$

(for purposes of analysis (see Appendix III), it is easier to deal with cartesian coordinates as opposed to cylindrical polar coordinates).

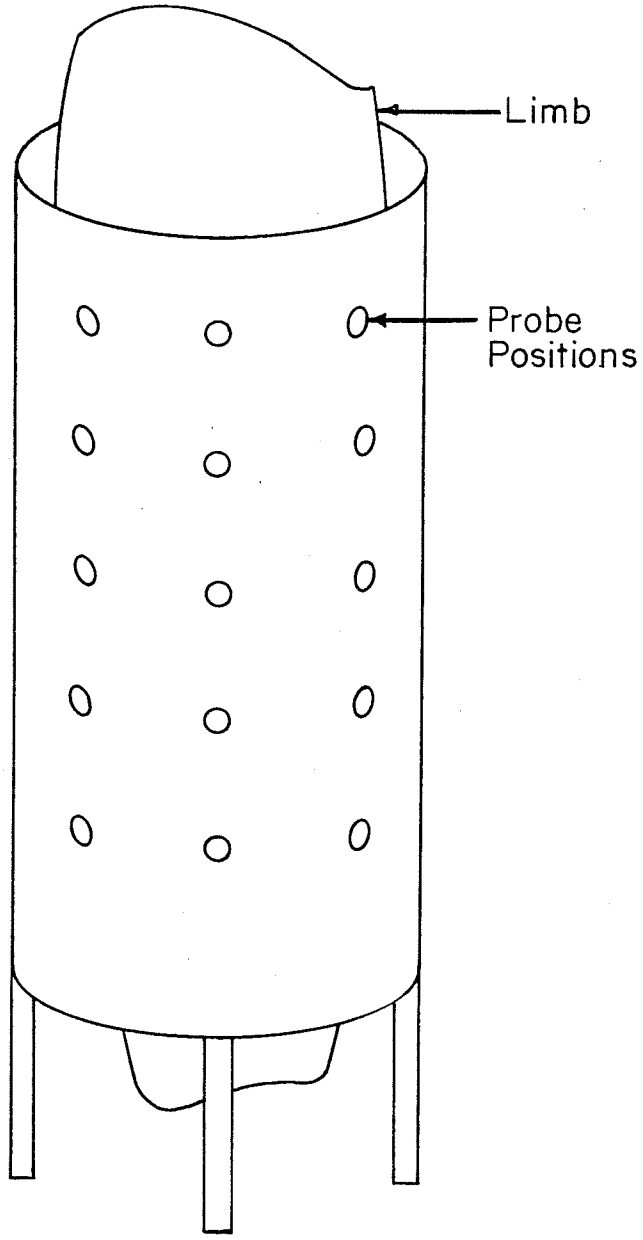


Figure AI-1. Sensing Device

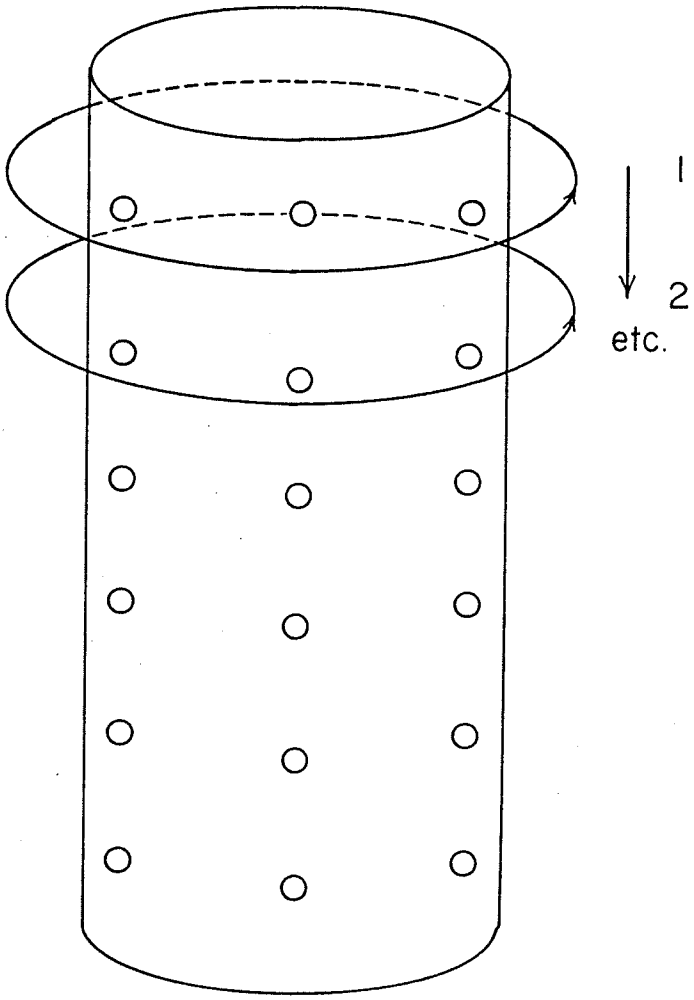


Figure A1-2 Sampling Sequence

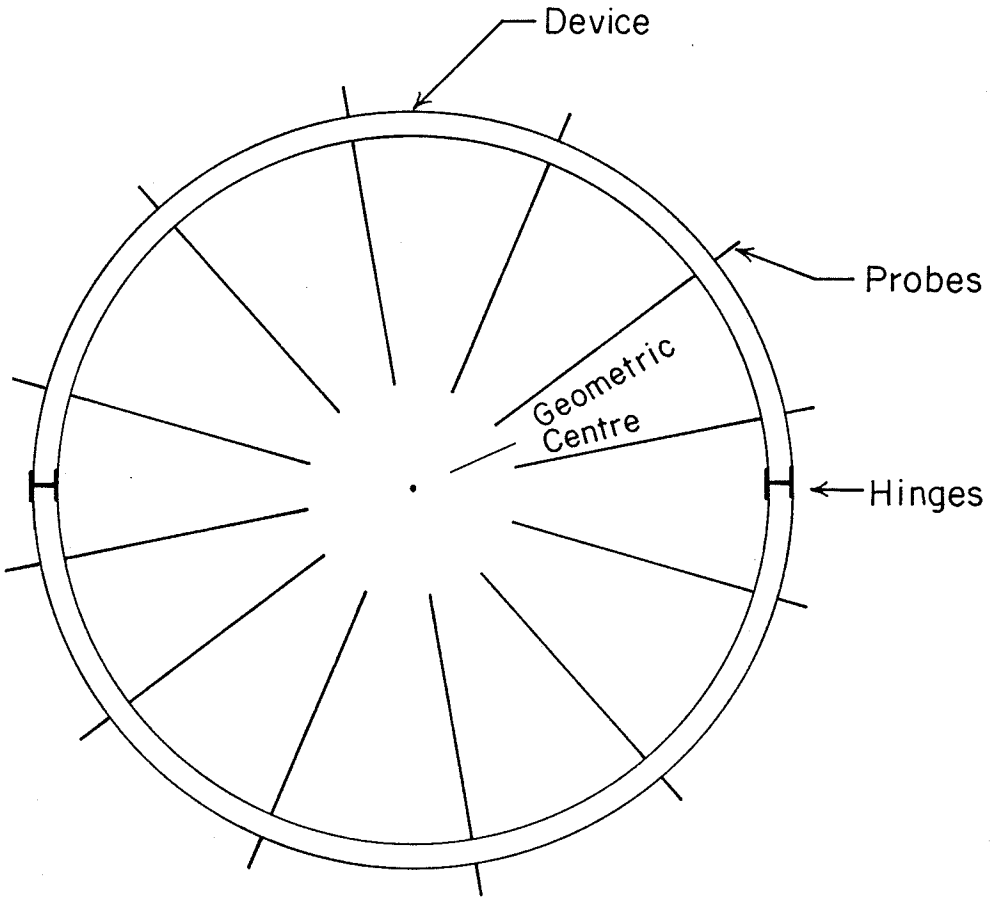


Figure AI-3 Top View of Sensing Device

Suppose that a leg cross-section has the form as shown in Figure A1-4. If the measuring device samples the leg as in Figure A1-5, it is noted that the device is obviously limited to sampling a finite number of points from the curve. From the sampled points, after performing the above-mentioned conversions into x-y coordinates, the shape as indicated in Figure A1-6 is obtained.

In order that a numerical control device be able to operate and carve out this likeness, it is necessary that the tool movement be completely defined for motion between the data points. Should the movement between data points follow a straight line, the curve as shown in Figure A1-7 would result. But Figure A1-7 by no means resembles Figure A1-4. Thus a linear interpolation between the data points does not produce a good result. It has been found that spline functions give the best reconstruction to the sampled shape, the results of which are contained in Appendix III.

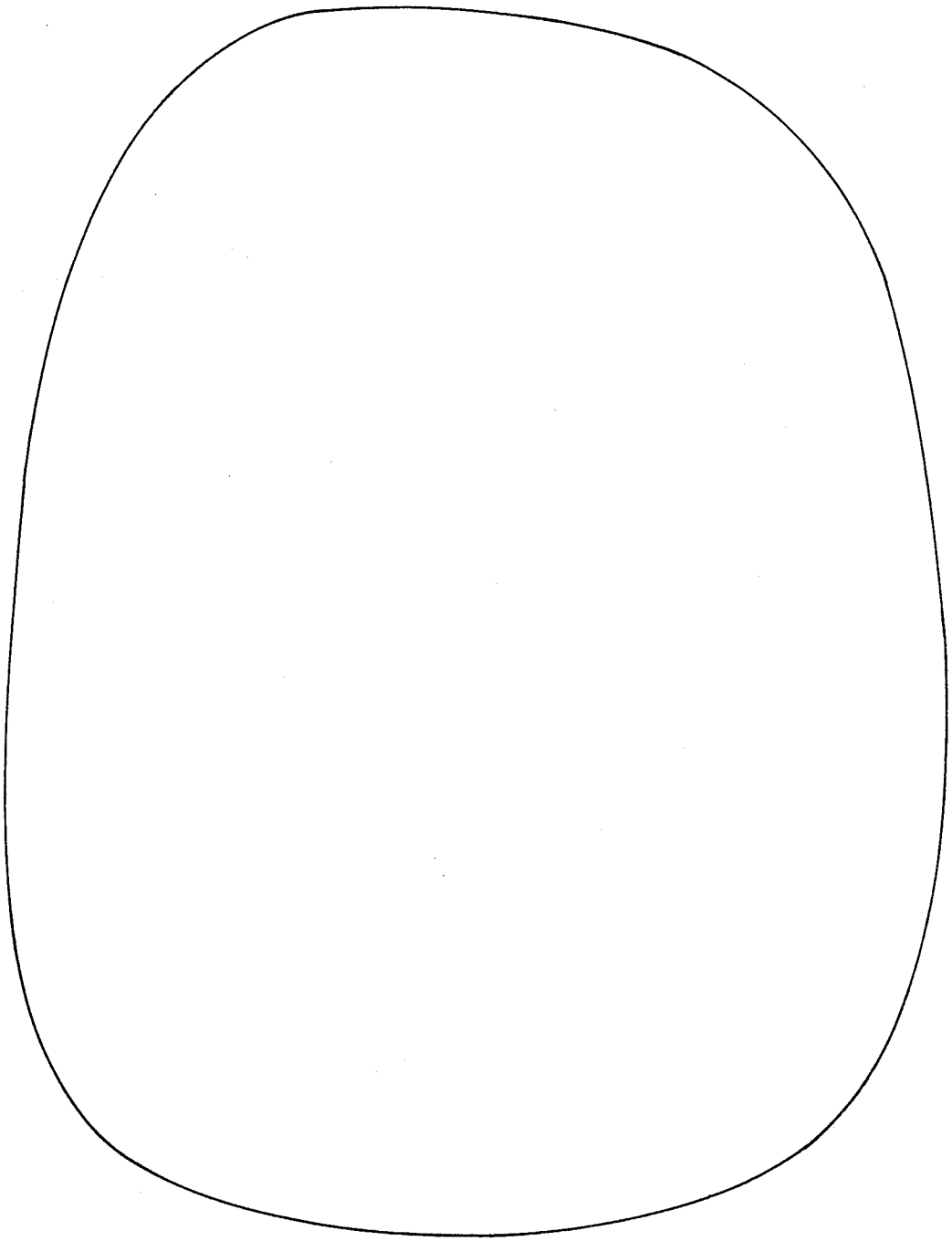


Figure AI-4. Cross-Section Limb Representation

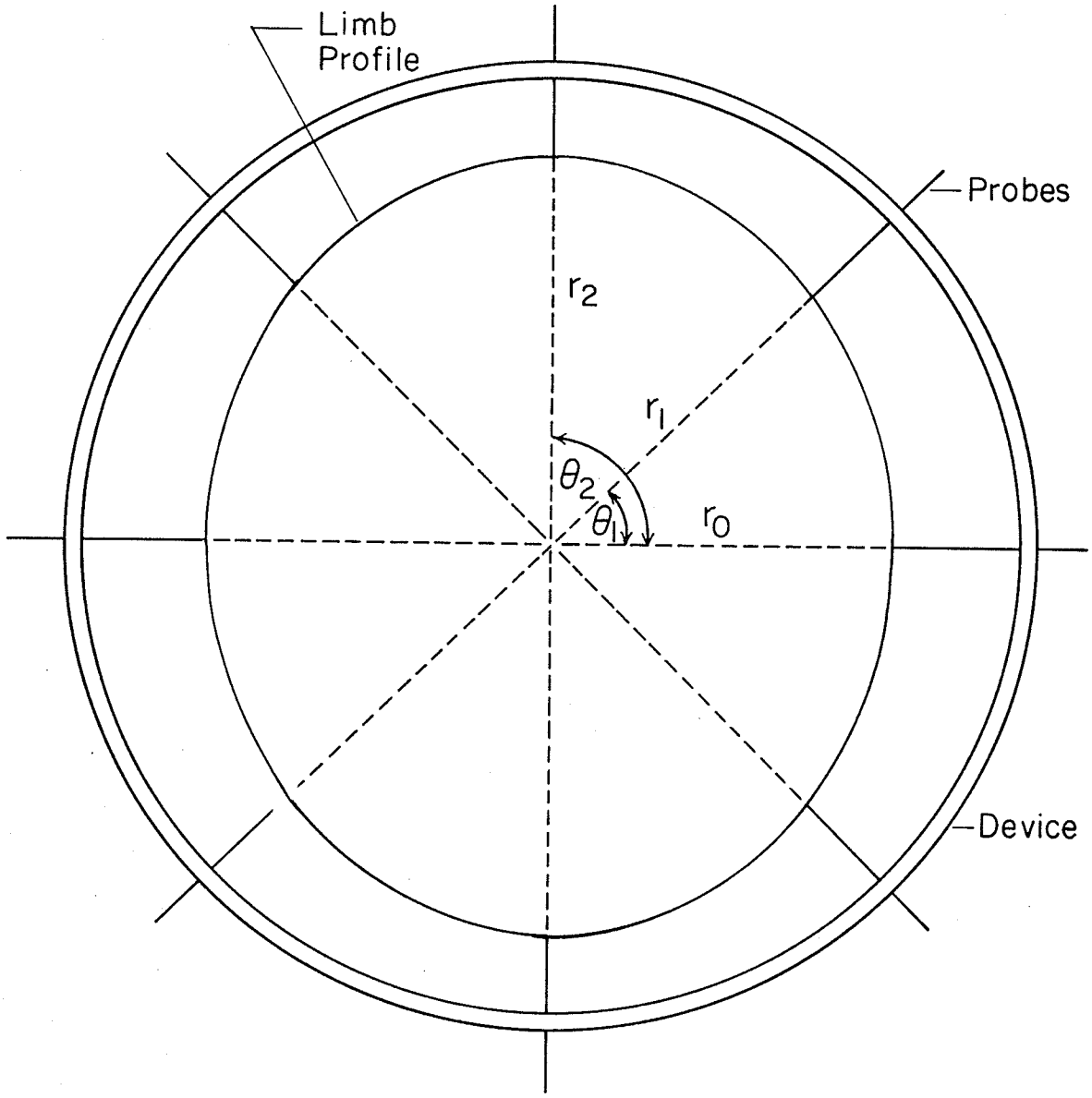


Figure A1-5 Measuring Device Sampling Leg Cross-Section

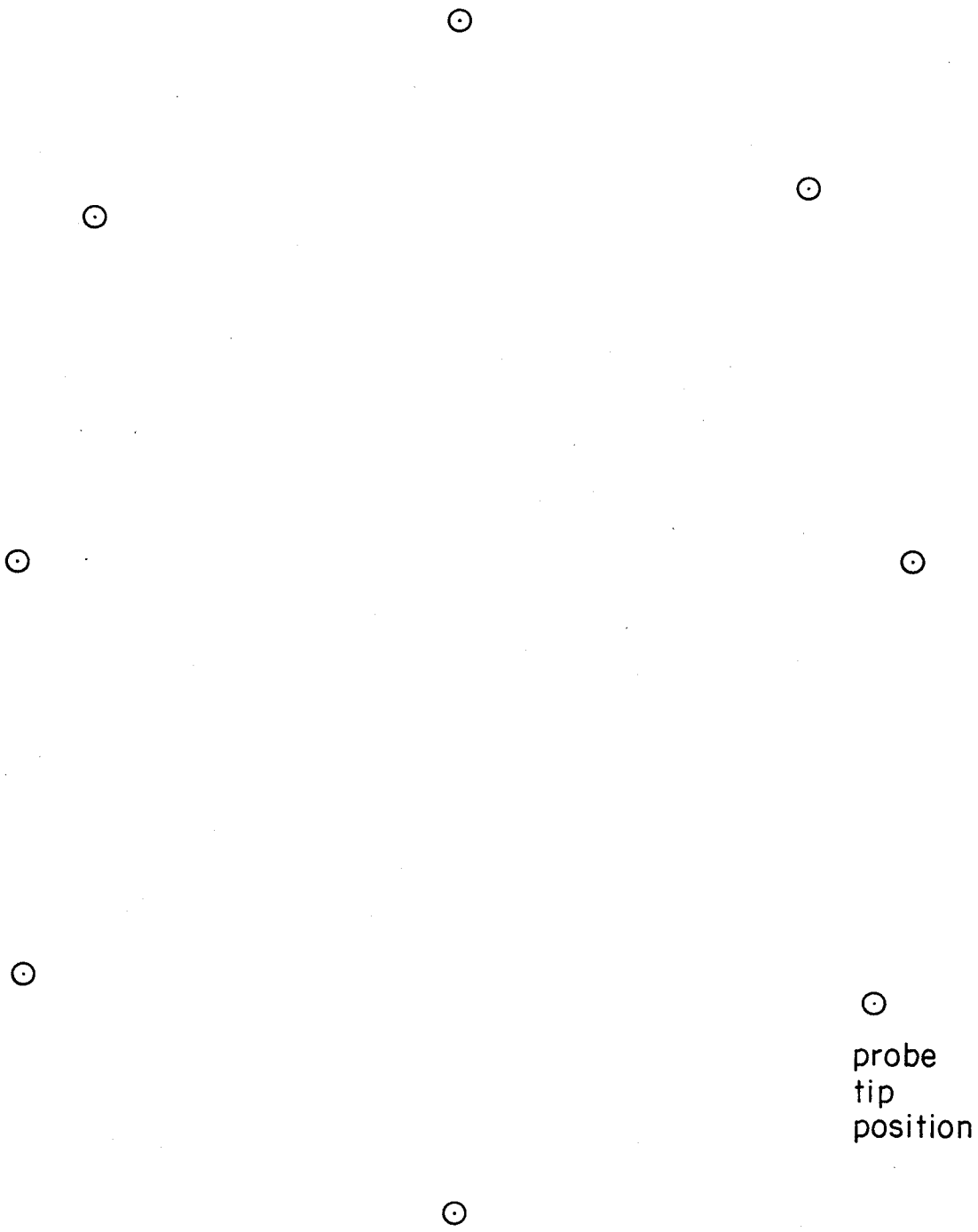


Figure A1-6 Initial Results From Shape Sensing Device

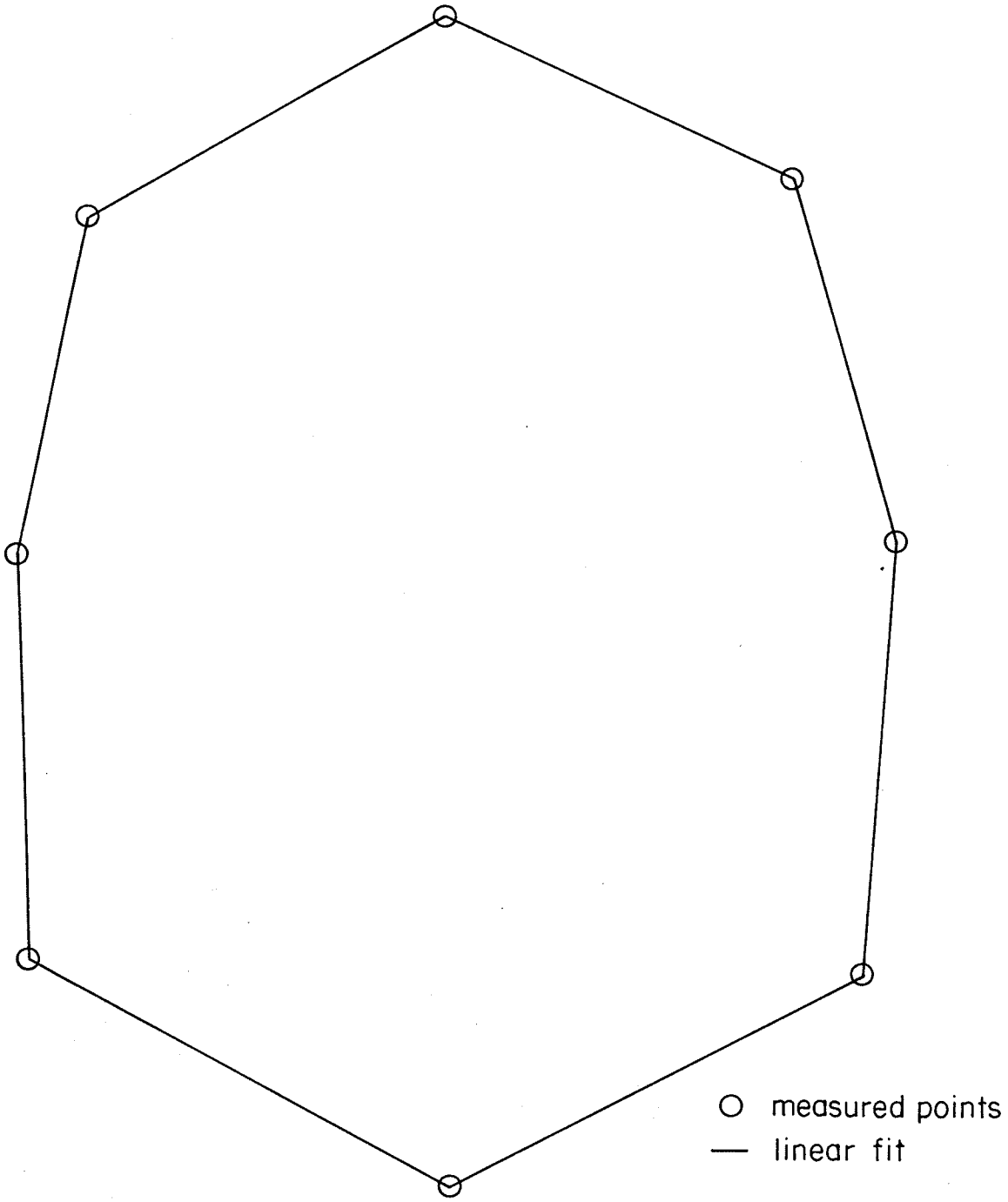


Figure A1-7 Linear Interpolation

APPENDIX II

This appendix contains what is hoped to be a clear, simple derivation of spline functions as they apply to two dimensional coordinate spaces.

In the mathematical spline, an interval $a \leq x \leq b$ is first considered. This interval is subdivided by a mesh of points which corresponds to the location of the weights of the draftsman's spline.

$$\Delta: \quad a = x_0 < x_1 < \cdots\cdots < x_n = b$$

There are n+1 points, and n subintervals. The subintervals between the points need not be the same length. An associated set of ordinates is prescribed.

$$Y: \quad y_0, y_1, y_2, \cdots\cdots y_n$$

A function $S_\Delta(x)$ is sought which must:

a) be a continuous function and have continuous first and second derivatives on the closed interval [a,b] (includes endpoints)

b) coincide with a cubic in each subinterval $x_{j-1} \leq x \leq x_j$ $(j=1,2, \cdots\cdots n)$, that is, the function will be of degree three in x on any subinterval

c) satisfy the condition $S_\Delta(x_j) = y_j$ $(j=0,1, \cdots\cdots n)$ That is, at each point x_j (which is a mesh point), the value of y_j at the point will be the y that was previously described.

The function $S_\Delta(x)$ which fulfils the three conditions mentioned is called a spline on Δ (or a spline with respect to the mesh Δ), which interpolates to the values

y_j at the mesh locations.

It is usual to call M_j the "moment", which is the second derivative of the spline function at the mesh point, i.e. $M_j \equiv S''_\Delta(x)$ $j=0,1, \cdots\cdots n$. However this is not the true moment of the draftsman's spline in the general context of the meaning.

The spline function $S_\Delta(x)$ can be found by deriving a relationship for $S''_\Delta(x)$ between any two adjacent mesh locations, then integrating $S''_\Delta(x)$ twice and match boundary conditions. However, before determining the relationship for $S''_\Delta(x)$, consider Figure A2-1.

From the formula for a straight is obtained

$$y-y_1=m(x-x_1)$$

$$y=m(x-x_1)+y_1$$

$$y=((y_2-y_1)/(x_2-x_1))(x-x_1)+y_1$$

expanding this

$$y(x_2-x_1)=(y_2-y_1)(x-x_1)+y_1(x_2-x_1)$$

$$y(x_2-x_1)=y_2x-y_1x-y_2x_1+y_1x_1+y_1x_2-y_1x_1$$

$$\therefore\ y=y_2(x-x_1)/(x_2-x_1)+y_1(x_2-x)/(x_2-x_1)$$

Thus on the closed interval $[x_{j-1},x_j]$, and considering the linearity of the second derivative (the second derivative is linear since the spline function is of order three in all subintervals), the following analysis holds (see also Figure A2-2):

$$S''_\Delta(x)-S''_\Delta(x_{j-1})=m(x-x_{j-1})$$

$$S''_\Delta(x)-S''_\Delta(x_{j-1})=((S''_\Delta(x_j)-S''_\Delta(x_{j-1})/(x_j-x_{j-1}))(x-x_{j-1})$$

Now since $S''_\Delta(x_{j-1})\equiv M_{j-1}$, and $S''_\Delta(x_j)\equiv M_j$, by substituting and

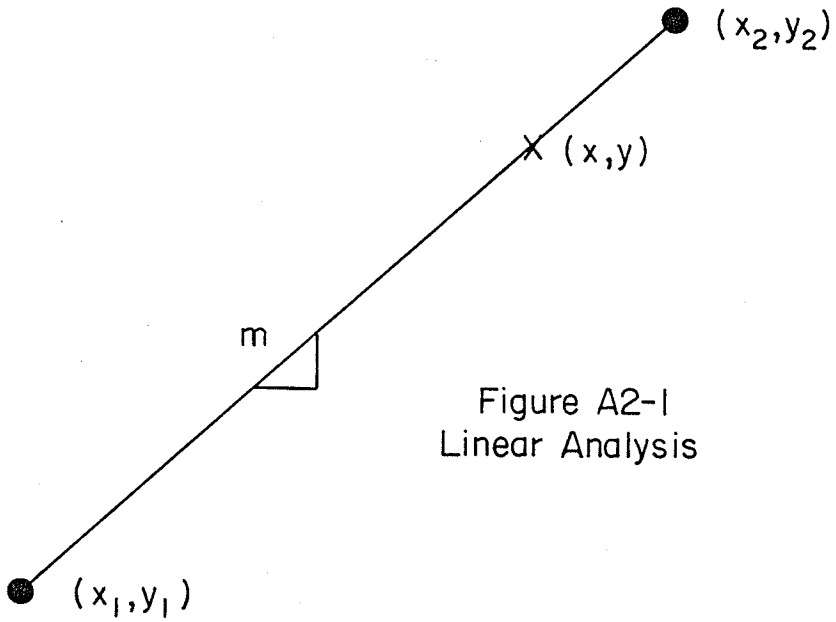


Figure A2-1
Linear Analysis

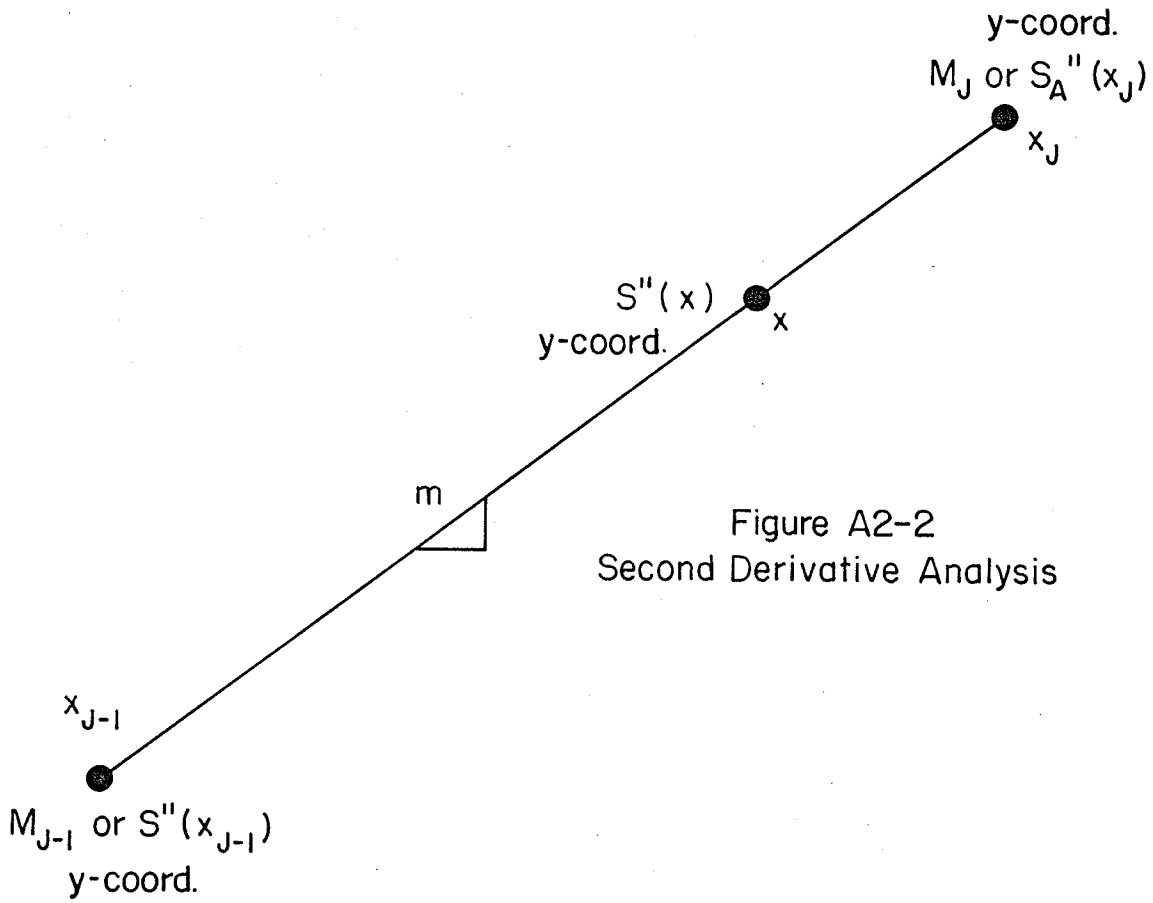


Figure A2-2
Second Derivative Analysis

cross multiplying

$$(x_j-x_{j-1})(S''_\Delta(x)-M_{j-1})=(M_j-M_{j-1})(x-x_{j-1})$$

expanding

$$(x_j-x_{j-1})S''_\Delta(x)-x_jM_{j-1}+x_{j-1}M_{j-1}$$

$$=xM_j-xM_{j-1}-x_{j-1}M_j+x_{j-1}M_{j-1}$$

$$(x_j-x_{j-1})S''(x)=M_{j-1}(x_j-x)+M_j(x-x_{j-1})$$

$$S''_\Delta(x)=M_{j-1}(x_j-x)/(x_j-x_{j-1})+M_j(x-x_{j-1})/(x_j-x_{j-1})$$

Let $h_j=x_j-x_{j-1}$

$$\therefore S''_\Delta(x)=M_{j-1}(x_j-x)/h_j+M_j(x-x_{j-1})/h_j \quad \ldots\ldots \quad (1)$$

Integrate equation (1) twice and evaluate the constants of integration.

$$S'_\Delta(x)=-M_{j-1}(x_j-x)^2/2h_j+M_j(x-x_{j-1})^2/2h_j+C_1 \quad ..(a)$$

$$S_\Delta(x)=M_{j-1}(x_j-x)^3/6h_j+M_j(x-x_{j-1})^3/6h_j+C_1x+C_2 \quad ..(b)$$

Boundary conditions for the interval are:

$$S_\Delta(x_j)=y_j \ , \quad S_\Delta(x_{j-1})=y_{j-1}$$

Putting these boundary conditions into (b) gives

$$y_j=M_{j-1}(x_j-x_j)^3/6h_j+M_j(x_j-x_{j-1})^3/6h_j+C_1x_j+C_2 \quad .. \quad (c)$$

$$y_{j-1}=M_{j-1}(x_j-x_{j-1})^3/6h_j+M_j(x_{j-1}-x_{j-1})^3/6h_j$$
$$+C_1x_{j-1}+C_2 \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (d)$$

Subtract (d) from (c)

$$y_j-y_{j-1}=M_jh_j^2/6-M_{j-1}h_j^2/6+C_1(x_j-x_{j-1})$$

$$(y_j-y_{j-1})/(x_j-x_{j-1})-((M_j-M_{j-1})/(x_j-x_{j-1}))h_j^2/6=C_1$$

or $\therefore C_1=(y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6$

Putting this value back into equation (a) gives

$$S'_\Delta(x)=-M_{j-1}(x_j-x)^2/2h_j+M_j(x-x_{j-1})^2/2h_j$$
$$+(y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6 \quad \ldots\ldots \quad (2)$$

From equation (c)

$$C_2=y_j-M_jh_j^2/6-x_j((y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6)$$

Putting this value into equation (b) gives

$$S_\Delta(x)=M_{j-1}(x_j-x)^3/6h_j+M_j(x-x_{j-1})^3/6h_j$$
$$+((y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6)x+y_j-M_jh_j^2/6$$
$$-x_j((y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6)$$

Expanding terms and then collecting yields

$$\therefore S_\Delta(x)=M_{j-1}(x_j-x)^3/6h_j+M_j(x-x_{j-1})^3/6h_j$$
$$+(y_{j-1}-M_{j-1}h_j^2/6)(x_j-x)/h_j$$
$$+(y_j-M_jh_j^2/6)(x-x_{j-1})/h_j \quad \ldots\ldots\ldots\ldots \quad (3)$$

From equation (2), the following expressions for the one-sided limits of the derivative are obtained (Figure A2-3)

$$S'_\Delta(x)=-M_{j-1}(x_j-x)^2/2h_j+M_j(x-x_{j-1})^2/2h_j$$
$$+(y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6$$

In the first interval, from j-1 to j, the following is obtained

$$S'_\Delta(x_j^-)=-M_{j-1}(x_j-x_j)^2/2h_j+M_j(x_j-x_{j-1})^2/2h_j$$
$$+(y_j-y_{j-1})/h_j-(M_j-M_{j-1})h_j/6$$
$$=M_jh_j/2+(y_j-y_{j-1})/h_j-M_jh_j/6+M_{j-1}h_j/6$$
$$=M_jh_j/3+M_{j-1}h_j/6+(y_j-y_{j-1})/h_j$$

In the second interval, from j to j+1, the following is obtained

$$S'_\Delta(x_j^+)=-M_{j-1}(x_{j+1}-x_j)^2/2h_{j+1}+M_{j+1}(x_j-x_j)^2/2h_{j+1}$$
$$+(y_{j+1}-y_j)/h_{j+1}-(M_{j+1}-M_j)h_{j+1}/6$$
$$=-M_jh_{j+1}/2+(y_{j+1}-y_j)/h_{j+1}-M_{j+1}h_{j+1}/6+M_jh_{j+1}/6$$
$$=-M_jh_{j+1}/3-M_{j+1}h_{j+1}/6+(y_{j+1}-y_j)/h_{j+1}$$

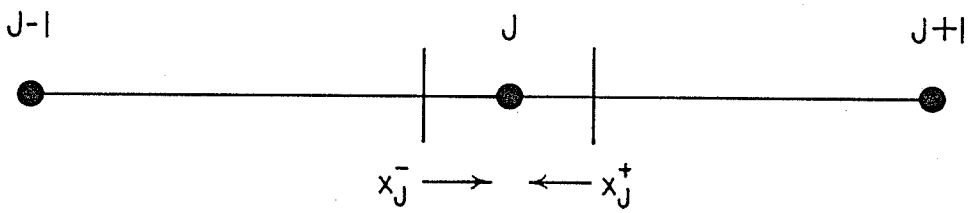


Figure A2-3 One Side Limits

So

$$\left.\begin{aligned} S'_{\Delta}(x_j^-) &= M_{j-1}h_j/6+M_jh_j/3+(y_j-y_{j-1})/h_j \\ S'_{\Delta}(x_j^+) &= -M_jh_{j+1}/3-M_{j+1}h_{j+1}/6+(y_{j+1}-y_j)/h_{j+1} \end{aligned}\right\} \quad \ldots\ (4)$$

Due to the fact that the nature of the cubic function in the closed interval is such that the second derivative is made so as to be piecewise analytic and continuous in this interval, the first derivative equation, i.e. (4), is "forced" to be continuous. This is the continuity requirement. In light of equations (1) and (3), the functions $S''_{\Delta}(x)$ and $S_{\Delta}(x)$ are continuous on [a,b]. By equating both parts of (4), the following is obtained

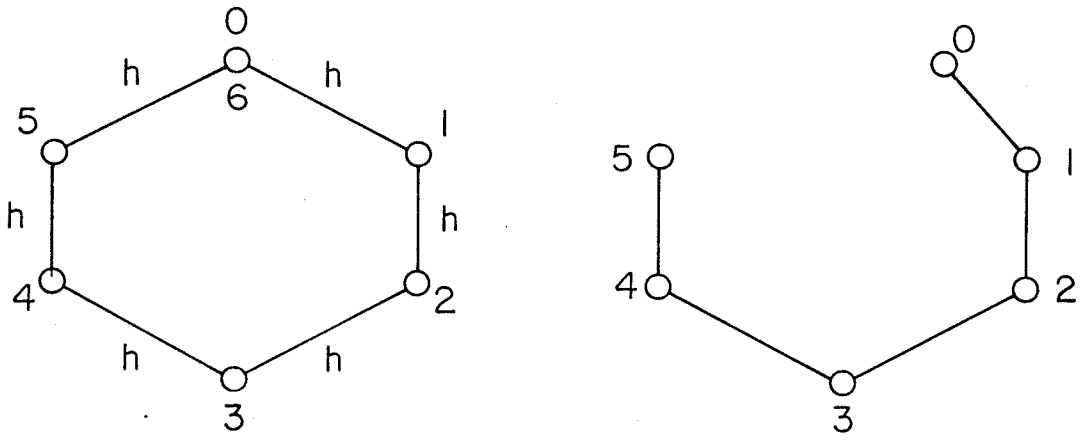
$$M_{j-1}h_j/6+M_jh_j/3+(y_j-y_{j-1})/h_j = -M_jh_{j+1}/3-M_{j+1}h_{j+1}/6+(y_{j+1}-y_j)/h_j$$

or

$$M_{j-1}h_j/6+M_j(h_j+h_{j+1})/3+M_jh_{j+1}/6 = (y_{j+1}-y_j)/h_{j+1}-(y_j-y_{j-1})/h_j \quad \ldots\ldots\ (5)$$

By definition, a spline is said to be periodic of period (b-a) if the condition $S^{(p)}(a^+)=S^{(p)}(b^-)$, (p=0,1,2) is satisfied.

For the periodic spline, equation (5) for j=1,2,...n-1 gives n-1 simultaneous equations in the quantities $M_1, M_2, \ldots M_n$. It is required that equation (5) be valid also for j=n, which is how the simultaneous equations in $M_1, M_2, \ldots M_n$ are obtained, and not $M_1, M_2, \ldots M_{n-1}$. Here $y_n=y_0$, $M_n=M_0$, and the following relations are prescribed $y_{n+1}=y_1$, $M_{n+1}=M_1$, $h_{n+1}=h_1$. From Figure A2-4a and Figure A3-4b, it turns out that for the periodic case (Figure A2-4a)



(a) Periodic (b) Non-periodic

Figure A2-4

Period and Non-periodic Conditions

mesh point 6 corresponds with mesh point 0.

Consider the periodic case. Equation (5) yields, using Figure A2-4(a+b) as an example,

$$M_0h_1/6+M_1(h_1+h_2)/3+M_2h_2/6=(y_2-y_1)/h_2-(y_1-y_0)/h_1$$

. . . . .

. . . . .

. . . . .

$$M_5h_6/6+M_6(h_6+h_7)/3+M_7h_7/6=(y_7-y_6)/h_7-(y_6-y_5)/h_6$$

or, writing only M's,

$$M_0 \quad M_1 \quad M_2 = \quad \text{(i)}$$
$$M_1 \quad M_2 \quad M_3 = \quad \text{(ii)}$$
$$M_2 \quad M_3 \quad M_4 = \quad \text{(iii)}$$
$$M_3 \quad M_4 \quad M_5 = \quad \text{(iv)}$$
$$M_4 \quad M_5 \quad M_6 = \quad \text{(v)}$$
$$M_5 \quad M_6 \quad M_7 = \quad \text{(vi)}$$

with $M_7=M_1$, $M_6=M_0$ due to periodicity, then have

$$M_0 \quad M_1 \quad M_2 = \quad \text{(i)}$$
$$M_1 \quad M_2 \quad M_3 = \quad \text{(ii)}$$
$$M_2 \quad M_3 \quad M_4 = \quad \text{(iii)}$$
$$M_3 \quad M_4 \quad M_5 = \quad \text{(iv)}$$
$$M_4 \quad M_5 \quad M_0 = \quad \text{(v)}$$
$$M_5 \quad M_0 \quad M_1 = \quad \text{(vi)}$$

There are now six simultaneous equations in six unknowns, which can be solved.

For the case of the nonperiodic spline, it is necessary to specify the "end conditions", which results in two additional conditions, and so permits the solution

of the n+1 quantities $M_0, M_1, \ldots\ldots M_n$. Many kinds of end conditions can be specified. One method of specification would be to state the slope of the spline at the end points, that is, at a and b. For $S'_\Delta(a)=y'_0$ and $S'_\Delta(b)=y'_n$, the following is obtained from equation (4)

$$S'_\Delta(x_j^+)=-M_j h_{j+1}/3-M_{j+1}h_{j+1}/6+(y_{j+1}-y_j)/h_{j+1}$$

at a, j=0, so

$$S'_\Delta(a)=-M_0 h_1/3-M_1 h_1/6+(y_1-y_0)/h_1=y'_0$$

rearrange

$$(y_1-y_0)/h_1-y'_0=h_1(M_0/3+M_1/6)$$

or

$$2M_0+M_1=((y_1-y_0)/h_1-y'_0)6/h_1 \quad \ldots\ldots\ldots\ldots\ldots \quad (e)$$

also

$$S'_\Delta(x_j^-)=M_{j-1}h_j/6+M_j h_j/3+(y_j-y_{j-1}0/h_j$$

at b, j=n, so

$$S'_\Delta(b)=M_{n-1}h_n/6+M_n h_n/3+(y_n-y_{n-1})/h_n=y'_n$$

rearrange

$$(M_{n-1}+2M_n)h_n/6=y'_n-(y_n-y_{n-1})/h_n$$

so

$$M_{n-1}+2M_n=((y'_n-(y_n-y_{n-1})/h_n)6/h_n \quad \ldots\ldots\ldots \quad (f)$$

Equations (e) and (f) specify one possible set of end conditions.

Another kind of end condition can be obtained from the equation

$$M_0-\lambda M_1=0 \qquad 1>\lambda\geq 0$$

This is the same as placing a simple (imaginary) support at a point

$$x_{imag.}=(x_0-\lambda x_1)/(1-\lambda)$$

and then requiring that the curvature of the resulting equation be the arc of a cubic in the region $x_{imag.}\leq x\leq x_1$ (λ is often taken to be =1/2).

This kind of boundary condition is employed when the value of M_0 is unknown. The method essentially results in a "reasonable" guess at M_0 using M_1.

The general equation for end conditions, of which the two previously discussed cases were simple degeneracies, is written in the form

$$2M_0+\lambda_0 M_1=d_0 \ , \mu_n M_{n-1}+2M_n=d_n \quad \ldots\ldots\ldots\ldots \quad (6)$$

The following notation is used

$$\lambda_j=h_{j+1}/(h_j+h_{j+1}), \mu_j=1-\lambda_j \quad (j=1,2, \ldots\ldots n-1)$$

The continuity requirement is equation (5),which is

$$M_{j-1}h_j/6+M_j(h_j+h_{j+1})/3+M_{j+1}h_{j+1}/6 = (y_{j+1}-y_j)/h_{j+1}-(y_j-y_{j-1})/h_j$$

since $\mu_j=1-\lambda_j$ and $\lambda_j=h_{j+1}/(h_j+h_{j+1})$

then $\mu_j=1-h_{j+1}/(h_j+h_{j+1})=(h_j+h_{j+1}-h_{j+1})/(h_j+h_{j+1})$

$$\mu_j=h_j/(h_j+h_{j+1})$$

$$\therefore\ h_j=\mu_j(h_j+h_{j+1})$$

$$\lambda_j=h_{j+1}/(h_j+h_{j+1})$$

$$h_{j+1}=\lambda_j(h_j+h_{j+1})$$

So in equation (5) have

$$\mu_j M_{j-1}\mu_j(h_j+h_{j+1})/6+M_j(h_j+h_{j+1})/3+M_{j+1}\lambda_j(h_j+h_{j+1})/6 = (y_{j+1}-y_j)/h_{j+1}-(y_j-y_{j-1})/h_j$$

or

$$\mu_j M_{j-1}+2M_j+\lambda_j M_{j+1}=6((y_{j+1}-y_j)/h_{j+1}-(y_j-y_{j-1})/h_j)/(h_j+h_{j+1}) \quad \ldots\ldots\ldots\ldots \quad (7)$$

For the nonperiodic spline, the defining equations (6) and (7) are now written as

$$\begin{bmatrix} 2 & \lambda_0 & 0 & \ldots & 0 & 0 & 0 \\ \mu_1 & 2 & \lambda_1 & \ldots & 0 & 0 & 0 \\ 0 & \mu_2 & 2 & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 2\lambda_{n-2} & 0 & \\ 0 & 0 & 0 & \ldots & \mu_{n-1} & 2\lambda_{n-1} & \\ 0 & 0 & 0 & \ldots & 0 & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-2} \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1} \\ d_n \end{bmatrix} \quad \ldots\ldots \quad (8)$$

where d_j (j=1,2, ... ,n-1) represents the right hand member of (7).

For the case of the periodic spline, the equations are

$$\begin{bmatrix} 2 & \lambda_1 & 0 & \ldots & 0 & 0 & \mu_1 \\ \mu_2 & 2 & \lambda_2 & \ldots & 0 & 0 & 0 \\ 0 & \mu_3 & 2 & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 2\lambda_{n-2} & 0 & \\ 0 & 0 & 0 & \ldots & \mu_{n-1} & 2\lambda_{n-1} & \\ \lambda_n & 0 & 0 & \ldots & 0 & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \\ M_3 \\ \vdots \\ M_{n-2} \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n-2} \\ d_{n-1} \\ d_n \end{bmatrix} \quad \ldots\ldots \quad (9)$$

with $M_0=M_n$, $\lambda_n=h_1/(h_n+h_1)$

$\mu_n=1-\lambda_n$

In many situations it is easier to use the slopes $m_j=S'_\Delta(x_j)$ than the moments $M_j=S''_\Delta(x_j)$. On the closed interval (really a subinterval) x_{j-1},x_j , the following equation holds

$$S_\Delta(x)=m_{j-1}((x_j-x)^2(x-x_{j-1}))/h_j^2 -m_j((x-x_{j-1})^2(x_j-x))/h_j^2$$
$$+y_{j-1}(x_j-x)^2(2(x-x_{j-1})+h_j)/h_j^3 +y_j(x-x_{j-1})^2(2(x_j-x)+h_j)/h_j^3 \quad \ldots (10)$$

The first derivative of equation (10) is

$$S'_\Delta(x)=m_{j-1}(x_j-x)(2x_{j-1}+x_j-3x)/h_j^2$$
$$-m_j(x-x_{j-1})(2x_j+x_{j-1}-3x)/h_j^2 +(y_j-y_{j-1})6(x_j-x)(x-x_{j-1})/h_j^3 \quad \ldots (11)$$

To obtain the second derivative of (10), first expand (11)

$$S'_\Delta(x)=m_{j-1}(2x_{j-1}x_j+x_j^2-3xx_j-2xx_{j-1}-xx_j-3x^2)/h_j^2$$
$$-m_j(2xx_j+xx_{j-1}-3x^2-2x_jx_{j-1}-x_{j-1}^2+3xx_{j-1})$$
$$+(y_j-y_{j-1})6(xx_j-x_jx_{j-1}-x^2+xx_{j-1})/h_j^3$$

then

$$S''_\Delta(x)=m_{j-1}(-3x_j-2x_{j-1}-x_j-6x)-m_j(2x_j+x_{j-1}-6x+3x_{j-1})/h_j^2$$
$$+(y_j-y_{j-1})6(x_j-2x+x_{j-1})/h_j^3$$
$$\therefore\ S''_\Delta(x)=-2m_{j-1}(2x_j+x_{j-1}-3x)/h_j^2 -2m_j(2x_{j-1}+x_j-3x)/h_j^2$$
$$+6(y_j-y_{j-1})(x_j+x_{j-1}-2x)/h_j^3 \quad \ldots\ldots (12)$$

The limiting values from the two sides of x_j are (as before)

$$S''_\Delta(x_j^-)=-2m_{j-1}(2x_j+x_{j-1}-3x_j)/h_j^2 -2m_j(2x_{j-1}+x_j-3x_j)/h_j^2$$
$$+6(y_j-y_{j-1})(x_j+x_{j-1}-2x_j)/h_j^3$$
$$=-2m_{j-1}(x_{j-1}-x_j)/h_j^2 -2m_j(-2x_j-2x_{j-1})/h_j^2$$
$$+6(y_j-y_{j-1})(x_{j-1}-x_j)/h_j^3$$
$$=2m_{j-1}/h_j +4m_j/h_j -6(y_j-y_{j-1})/h_j^2$$

and

$$S''_\Delta(x_j^+)=-2m_j(2x_{j+1}+x_j-3x_j)/h_j^2 -2m_{j+1}(2x_j+x_{j+1}-3x_j)/h_{j+1}^2$$
$$+6(y_{j+1}-y_j)(x_{j+1}+x_j-2x_j)/h_j^3$$
$$=-2m_j2(x_{j+1}-x_j)/h_{j+1}^2 -2m_{j+1}(x_{j+1}-x_j)/h_{j+1}^2$$
$$+6(y_{j+1}-y_j)(x_{j+1}-x_j)/h_j^3$$

$$= -4m_j/h_{j+1} - 2m_{j+1}/h_{j+1} + 6(y_{j+1}-y_j)/h_j^2$$

So:

$$\left.\begin{aligned} S''(x_j^-) &= 2m_{j-1}/h_j + 4m_j/h_j - 6(y_j-y_{j-1})/h_j^2 \\ S''(x_j^+) &= -4m_j/h_{j+1} - 2m_{j+1}/h_{j+1} + 6(y_{j+1}-y_j)/h_j^2 \end{aligned}\right\} \quad \ldots\ldots (13)$$

Imposing the continuity condition on $S''_\Delta(x)$ at x_j (j=1,2,...n-1) (i. e. equating the right hand sides of equation (13)) results in

$$2m_{j-1}/h_j + 4m_j/h_j - 6(y_j-y_{j-1})/h_j^2 = -4m_j/h_{j+1} - 2m_{j+1}/h_{j+1} + 6(y_{j+1}-y_j)/h_j^2$$

The continuity requirement for slopes m_j is thus

$$m_{j-1}/h_j + 2(1/h_j+1/h_{j+1})m_j + m_{j+1}/h_{j+1} = 3(y_j-y_{j-1})/h_j^2 + 3(y_{j+1}-y_j)/h_{j+1}^2 \quad \ldots\ldots (14)$$

Equation (14) can be written in a more convenient form as

$$\lambda_j m_{j-1} + 2m_j + \mu_j m_{j+1} = 3\lambda_j(y_j-y_{j-1})/h_j + 3\mu_j(y_{j+1}-y_j)/h_{j+1} \quad \ldots (15)$$

where $\lambda_j = h_{j+1}/(h_j+h_{j+1})$ $\quad \mu_j = 1-\lambda_j$

The equation system for the nonperiodic spline using slopes is therefore

$$\begin{bmatrix} 2 & 0 & 0 & \ldots\ldots & 0 & 0 & 0 \\ 1 & 2 & 1 & \ldots\ldots & 0 & 0 & 0 \\ 0 & 2 & 2 & \ldots\ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & & 2_{n-2} & 0 & \\ 0 & 0 & 0 & & n-1 & 2_{n-1} & \\ 0 & 0 & 0 & & 0 & n & 2 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix}$$

where the general end conditions $2m_0+\mu_0 m_1=c_0$, $\lambda_n m_{n-1}+2m_n=c_n$ are employed. As before, $\lambda_j=h_{j+1}/(h_j+h_{j+1})$, $\mu_j=1-\lambda_j$ (for j=1,2, ... n-1)

and c_j (j=1,2, ... n-1) represents the right hand member of (15).

For the periodic case, the equations are

$$\begin{bmatrix} 2 & \mu_1 & 0 & \cdots & 0 & 0 & 0 \\ \lambda_2 & 2 & \mu_2 & \cdots & 0 & 0 & 0 \\ 0 & \lambda_3 & 2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2 & \mu_{n-2} & 0 \\ 0 & 0 & 0 & \cdots & \lambda_{n-1} & 2 & \mu_{n-1} \\ \mu_n & 0 & 0 & \cdots & 0 & \lambda_n & 2 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \vdots \\ m_{n-2} \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix}$$

To reiterate, the equation that defines and produces the spline points is (from equation (10))

$$S_\Delta(x) = m_{j-1}(x_j-x)^2(x-x_{j-1})/h_j^2 - m_j(x-x_{j-1})^2(x_j-x)/h_j^2$$
$$+y_{j-1}(x_j-x)^2(2(x-x_{j-1})+h_j)/h_j^3 + y_j(x-x_{j-1})^2(2(x_j-x)+h_j)/h_j^3$$

Appendix III contains a computer program which uses this equation to generate a two-space spline which approximates a leg cross-section.

APPENDIX III

The theory of splines present in Appendix II is now applied to the following problem. Suppose that an arbitrary leg cross-section, which has the profile as shown in Figure A3-1 is being dimensionally sampled. This profile cross-section comes from the region indicated in Figure A3-2. Assume that the shape-sensing equipment took samples at the positions around the cross-section as indicated in Figure A3-2. Superimposing a fine mesh on this arbitrarily drawn leg profile, the set of x-y coordinates as entered in Table I are obtained. This set of coordinates becomes the data set which is used to generate spline functions. These functions produce x and y values which lie on a smooth curve between the data points. These x and y interpolated values are given by the following formulae (as explained in Appendix II):

$$x(\text{interpolated}) = m_{j-1}(s_j-s)^2(s-s_{j-1})/h_j^2$$

$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2 + x_{j-1}(s_j-s)^2(2(s-s_{j-1}) + h_j)/h_j^3$$

$$+ x_j(s-s_{j-1})^2(2(s_j-s) + h_j)/h_j^3$$

$$y(\text{interpolated}) = m_{j-1}(s_j-s)^2(s-s_{j-1})/h_j^2$$

$$-m_j(s-s_{j-1})^2(s_j-s)/h_j^2 + y_{j-1}(s_j-s)^2(2(s-s_{j-1}) + h_j)/h_j^3$$

$$+ y_j(s-s_{j-1})^2(2(s_j-s) + h_j)/h_j^3$$

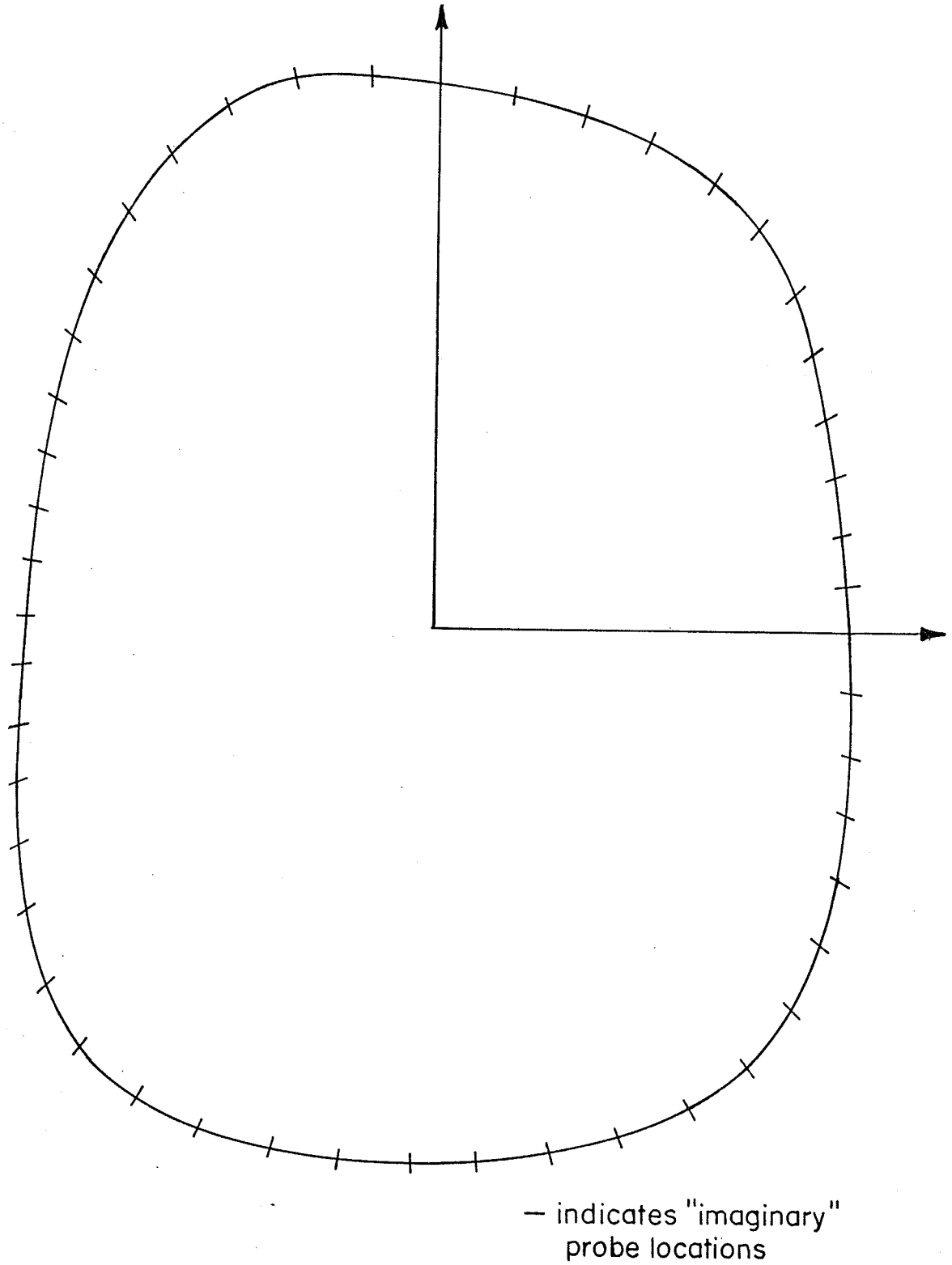


Figure A3-1 Limb Cross-Section

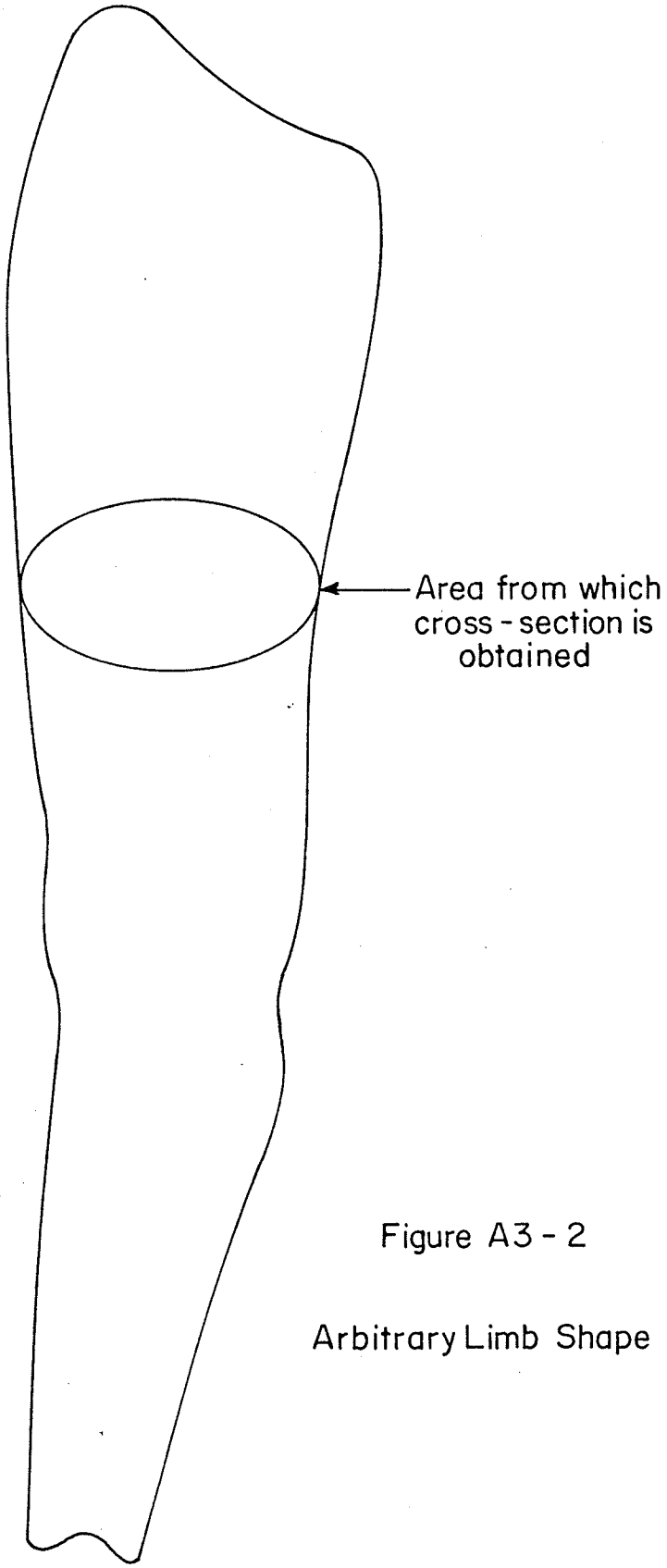


Figure A3 - 2

Arbitrary Limb Shape

TABLE I

From the indicated positions (corresponding to probe locations), the following data points were obtained:

θ	r(±.05)	θ	r(±.05)
0.0	7.20	187.5	6.90
7.5	7.20	195.0	7.10
15.0	7.25	202.5	7.50
22.5	7.45	210.0	7.80
30.0	7.70	217.5	8.25
37.5	8.05	225.0	8.75
45.0	8.45	232.5	9.20
52.5	8.85	240.0	9.40
60.0	8.95	247.5	9.45
67.5	8.95	255.0	9.40
75.0	8.95	262.5	9.40
82.5	9.00	270.0	9.40
90.0	9.10	277.5	9.50
97.5	9.30	285.0	9.55
105.0	9.45	292.5	9.55
112.5	9.40	300.0	9.50
120.0	9.10	307.5	9.40
127.5	8.70	315.0	8.90
135.0	8.20	322.5	8.35
142.5	7.75	330.0	7.90
150.0	7.40	337.5	7.60
157.5	7.15	345.0	7.35
165.0	6.95	352.5	7.20
172.5	6.85	360.0	7.20
180.0	6.80		

where all terms (for this, the periodic case) have been explained previously.

A computer program which produces these x and y values, and then plots the results (Figure A3-3) follows. Also included is a listing of the interpolated values which the plotter used to create Figure A3-3.

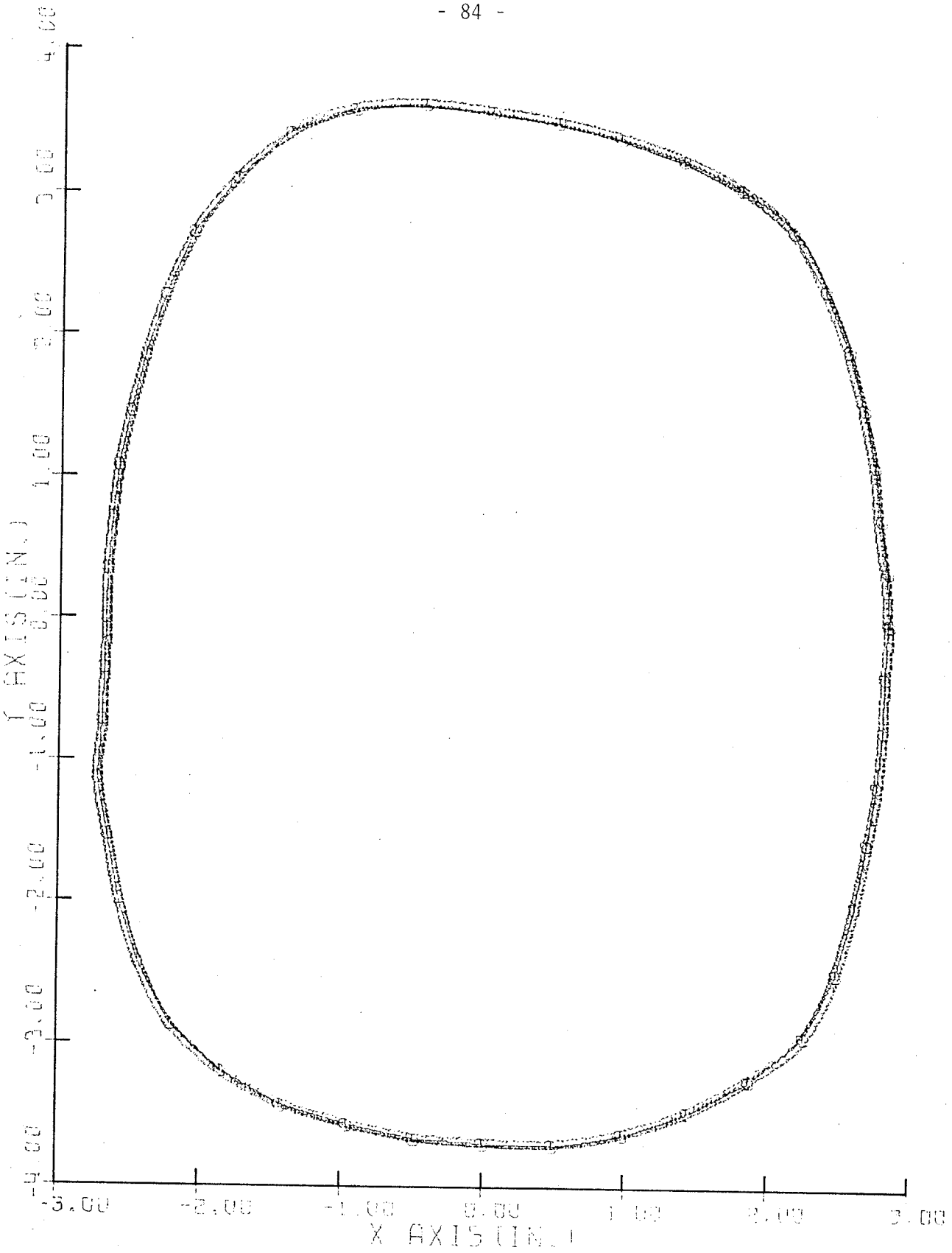


Figure A3-3. Cross-Section Reproduction

```
C
C     ***********************************************************
C
C          PLOT PROGRAM
C
C          FUNCTION:
C
C          THIS PROGRAM PLOTS THE PROFILE OF A LIMB CROSS-SECTION.
C          THE INPUT DATA IS IN POLAR(R-@) COORDINATES.  THE OUTPUT
C          IS IN CARTESIAN(X-Y) COORDINATES, FOR BOTH THE PLOT
C          AND THE NUMERIC LISTINGS.
C
C          TO USE THIS PROGRAM:
C
C          DO DATA INPUTS AS FOLLOWS.....
C               CARD I
C               -NUMBER OF DATA POINTS          I14
C               CARD II(NUMBER IS DATA DEPENDENT)
C               -PROBE POSITION(IN DEGREES)     12F6.1
C               CARD III(NUMBER IS DATA DEPENDENT)
C               -PROBE DISPLACEMENT(INCHES)     12F6.1
C
C          ***********************************************************
C
      REAL RADIUS(50),RADIUN(50),ANGLE(50),ANGLN(50)
      DIMENSION IBUF(1000),XSPLN(60),YSPLN(60)
      REAL MAT(60,60),MM(50),NN(50),X(60),Y(60)
      REAL A(60),V(60),H(50),S(50)
        REAL X1(60),Y1(60)
C
C          ***********************************************************
C
C          INPUT-OUTPUT FORMAT SPECIFICATIONS
C
  100 FORMAT('1','THE SPLINE POINTS ARE GENERATED',
     1' BETWEEN THE FOLLOWING COORDINATES:')
  101 FORMAT(///'    FROM X= ',F5.2,' ,Y= ',F5.2,
     1'  TO X= ',F5.2,' ,Y= ',F5.2)
  102 FORMAT(' ',' (R= ',F5.2,' IN.,@=',F5.1,
     1' DEG.) (R= ',F5.2,' IN.,@=',F5.1,' DEG.)')
  103 FORMAT(////T22,'X SPLINE     Y SPLINE')
  104 FORMAT(' ',T17,2F12.4)
  200 FORMAT(I14)
  201 FORMAT(12F6.1)
C
C          ***********************************************************
C
C          DATA INPUTS AND CONVERSIONS(POLAR TO CARTESIAN)
C
      READ(5,200) N
      DO 1 I=1,N
      DO 1 J=1,N
    1 MAT(I,J)=0.0
      READ(5,201) (RADIUN(I),I=1,N)
      READ(5,201) (ANGLN(I),I=1,N)
      DO 2 I=1,N
      ANGLE(I)=ANGLN(I)*3.14/180.
    2 RADIUS(I)=RADIUN(I)/2.54
```

```
      DO 4 I=1,N
      IF(ANGLE(I).GT.3.14/2.) GO TO 5
      X(I)=RADIUS(I)*COS(ANGLE(I))
      Y(I)=RADIUS(I)*SIN(ANGLE(I))
      GO TO 4
    5 IF(ANGLE(I).GT.3.14) GO TO 6
      ANGLE(I)=3.14-ANGLE(I)
      X(I)=-RADIUS(I)*COS(ANGLE(I))
      Y(I)=RADIUS(I)*SIN(ANGLE(I))
      GO TO 4
    6 IF(ANGLE(I).GT.3.*3.14/2.) GO TO 7
      ANGLE(I)=ANGLE(I)-3.14
      X(I)=-RADIUS(I)*COS(ANGLE(I))
      Y(I)=-RADIUS(I)*SIN(ANGLE(I))
      GO TO 4
    7 IF(ANGLE(I).GT.2.*3.14) GO TO 4
      ANGLE(I)=2.*3.14-ANGLE(I)
      X(I)=RADIUS(I)*COS(ANGLE(I))
      Y(I)=-RADIUS(I)*SIN(ANGLE(I))
    4 CONTINUE
C
C     ***************************************************************
C
C     CALL LIBRARY PLOT ROUTINES
C
      CALL PLOTS(IBUF,1000)
      CALL PLOT(0.0,1.0,-3)
      CALL SCALE(X,6.0,N,1)
      CALL SCALE(Y,8.0,N,1)
      CALL AXIS(0.0,0.0,11HX AXIS(IN.),-11,6.0
     1,0.0,X(N+1),X(N+2))
      CALL AXIS(0.0,0.0,11HY AXIS(IN.),11,8.0
     1,90.0,Y(N+1),Y(N+2))
      CALL LINE(X,Y,N,1,1,1)
C
C     ***************************************************************
C
C     CALCULATE CHORDAL AND CUMULATIVE CHORDAL DISTANCES
C
      N1=N-1
      N2=N-2
      S(1)=0.0
      DO 8 I=2,N
      ICT=I+1
      IF(ICT.LE.N) GO TO 8
      ICT=I-N+2
    8 H(I)=SQRT((X(ICT)-X(I))**2+(Y(ICT)-Y(I))**2)
      DO 9 I=1,N1
      ICT1=I+1
    9 S(ICT1)=S(I)+H(I)
C
C     ***************************************************************
C
C     COMPUTE THE MATRIX, THE X AND Y VECTORS
C
      DO 10 I=1,N1
      MAT(1,I)=(1./H(I)+1./H(I+1))*2.
      ICT2=I+2
```

```
      IF(ICT2.LT.N) GO TO 11
      ICT2=I-I+1
   11 Y1(I)=3.0/(H(I)**2*H(I+1)**2)*(H(I)**2*Y(ICT2)
     1-Y(I+1)*(H(I)**2-H(I+1)**2)-H(I+1)**2*Y(I))
   10 X1(I)=3.0/(H(I)**2*H(I+1)**2)*(H(I)**2*X(ICT2)
     1-X(I+1)*(H(I)**2-H(I+1)**2)-H(I+1)**2*X(I))
      DO 12 I=1,N2
      MAT(I+1,I)=1.0/H(I+1)
   12 MAT(I,I+1)=1.0/H(I+1)
      MAT(1,N1)=1.0/H(1)
      MAT(N1,1)=1.0/H(N)
C
C     ****************************************************************
C
C     TRIANGULARIZE THE MATRIX THEN MODIFY THE VECTORS
C
      DO 13 I=1,N2
      J=N1-I
      K=J+1
      VALU1=MAT(J,K)/MAT(K,K)
      Y1(J)=Y1(J)-Y1(K)*VALU1
      X1(J)=X1(J)-X1(J)*VALU1
      DO 14 L=1,K
   14 MAT(J,L)=MAT(J,L)-MAT(K,L)*VALU1
      VALU2=MAT(1,K)/MAT(K,K)
      Y1(1)=Y1(1)-Y1(K)*VALU2
      X1(1)=X1(1)-X1(K)*VALU2
      DO 15 M=1,K
   15 MAT(1,M)=MAT(1,M)-MAT(K,M)*VALU2
      DO 16 II=1,N1
      IF(ABS(MAT(J,II)).LE..000001) MAT(J,II)=0.0
   16 CONTINUE
   13 CONTINUE
C
C     ****************************************************************
C
C     CALL THE ROUTINE TO SOLVE THE EQUATIONS
C
      CALL LINYR(MAT,X1,N1,NN)
      CALL LINYR(MAT,Y1,N1,MM)
C     ****************************************************************
C
C     CALCULATE THIRTY SPLINE POINTS BETWEEN ANY TWO
C     ADJACENT DATA POINTS
C
      DO 17 I=1,N1
      LS1=I-1
      IF(LS1.EQ.0) LS1=N-I
      DO 18 J=1,30
      K=J
      DEL=K*H(I)/30+S(I)
      XSPLN(J)=NN(LS1)*(S(I+1)-DEL)**2*(DEL-S(I))/H(I)**2
     1-NN(I)*(DEL-S(I))**2*(S(I+1)-DEL)/H(I)**2
     1+X(I)*(S(I+1)-DEL)**2*(2*(DEL-S(I))+H(I))/H(I)**3
     1+X(I+1)*(DEL-S(I))**2*(2*(S(I+1)-DEL)+H(I))/H(I)**3
   18 YSPLN(J)=MM(LS1)*(S(I+1)-DEL)**2*(DEL-S(I))/H(I)**2
     1-MM(I)*(DEL-S(I))**2*(S(I+1)-DEL)/H(I)**2
     1+Y(I)*(S(I+1)-DEL)**2*(2*(DEL-S(I))+H(I))/H(I)**3
```

```
     1+Y(I+1)*(DEL-S(I))**2*(2*(S(I+1)-DEL)+H(I))/H(I)**3
C
C     ***************************************************************
C
C          MODIFY THE VALUES FOR OUTPUT
C          PUT THE RESULTS ON THE PLOTTER
C          PRODUCE A LISTING OF THE PLOTTED POINTS
C
          XSPLN(J+1)=X(N+1)
          XSPLN(J+2)=X(N+2)
          YSPLN(J+1)=Y(N+1)
          YSPLN(J+2)=Y(N+2)
          CALL LINE(XSPLN,YSPLN,J,1,-1,2)
          PRINT100
          WRITE(6,101) X(I),Y(I),X(I+1),Y(I+1)
          WRITE(6,102) RADIUS(I),ANGLN(I),RADIUS(I+1),ANGLN(I+1)
          PRINT103
          DO 19 L=1,30
   19     WRITE(6,104) XSPLN(L),YSPLN(L)
   17     CONTINUE
          CALL PLOT(15.0,0.0,999)
          STOP
          END
```

SPLINE POINTS ARE CALCULATED BETWEEN THE FOLLOWING COORDINATES:

FROM X= 2.83 ,Y= 0.0 TO X= 2.81 ,Y= 0.37
R= 2.83 IN.,@= 0.0 DEG.) (R= 2.83 IN.,@= 7.5 DEG.)

X SPLINE	Y SPLINE
2.8349	C.0123
2.8350	0.0247
2.8351	C.0370
2.8350	C.0494
2.8349	C.0617
2.8347	C.0741
2.8344	C.0864
2.8340	C.0988
2.8336	C.1111
2.8331	C.1235
2.8325	C.1358
2.8318	C.1482
2.8311	C.1605
2.8303	0.1728
2.8295	C.1852
2.8286	C.1975
2.8276	C.2099
2.8266	C.2222
2.8255	C.2345
2.8243	C.2468
2.8231	C.2592
2.8219	C.2715
2.8206	C.2838
2.8193	C.2961
2.8179	C.3084
2.8165	C.3207
2.8150	C.3330
2.8135	C.3453
2.8120	C.3575
2.8104	C.3698

APPENDIX IV

This appendix contains the program listing and graphical output which produces a three dimensional lower limb shape. The data used to produce the image is given in Table 2. This data is "imaginary" in the sense that it was not obtained from any human subject.

The method of generating the shape is as follows:

a) twenty "slices" along the limb are measured (from top to bottom) by obtaining thirteen (including end points - periodic spline) data points at each "slice" (Figure A4-1).

b) at each "slice", a spline function is generated between two points. These points are consecutive in the z direction.

c) the first spline point generated between the two consecutive points (say points 11 and 12), on all slices in the circumferential direction, is used to construct a spline function in the z (longitudinal) direction (Figure A4-2).

d) this procedure of producing spline points circumferentially between consecutive slice data points and then generating longitudinal splines continues around the entire data set which describes the limb. When the entire data set has been processed, a limb shape results.

This completes the procedure of duplicating the entire limb image. Computer listings and graphical outputs from this procedure follow (Figure A4-3).

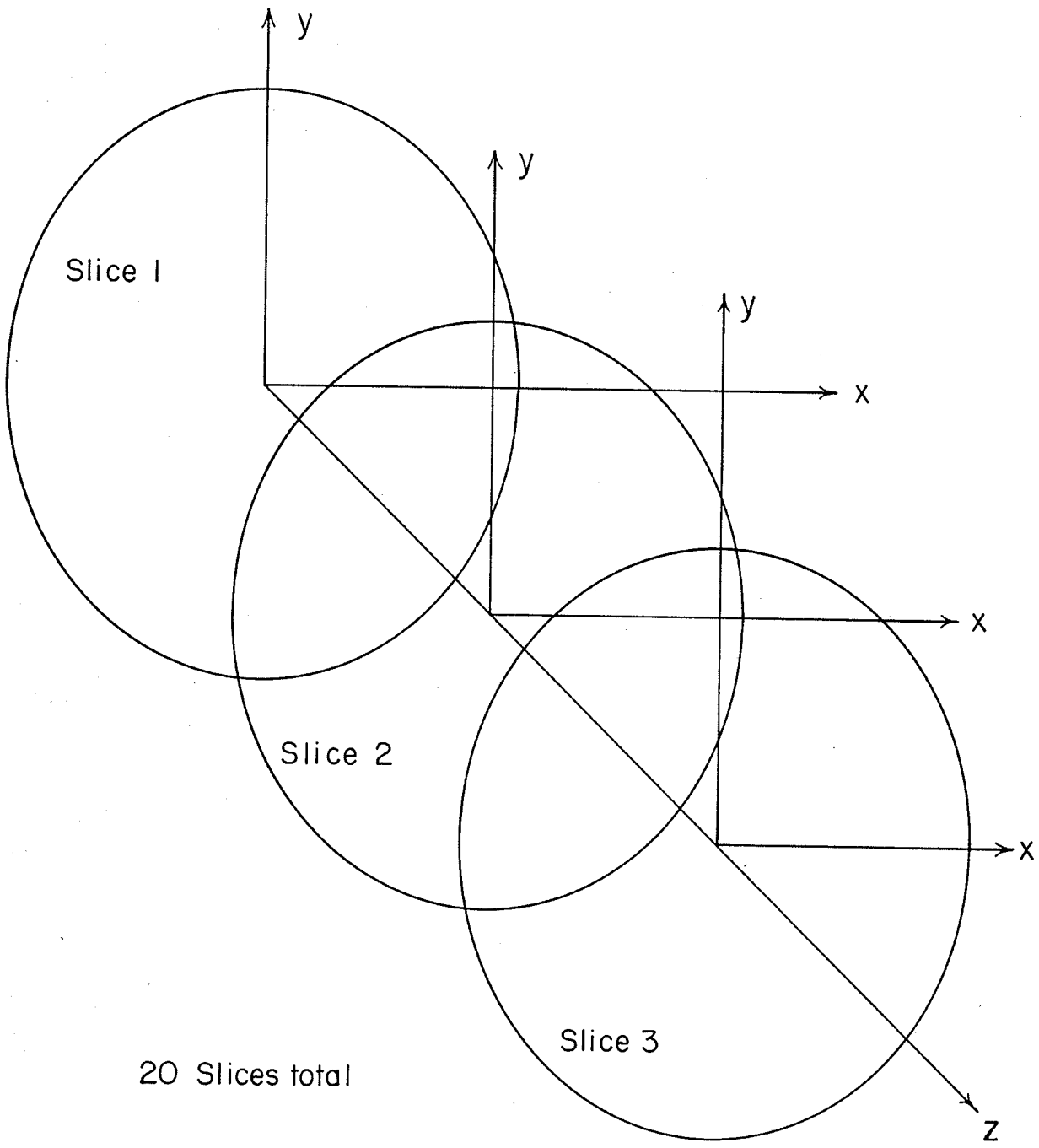
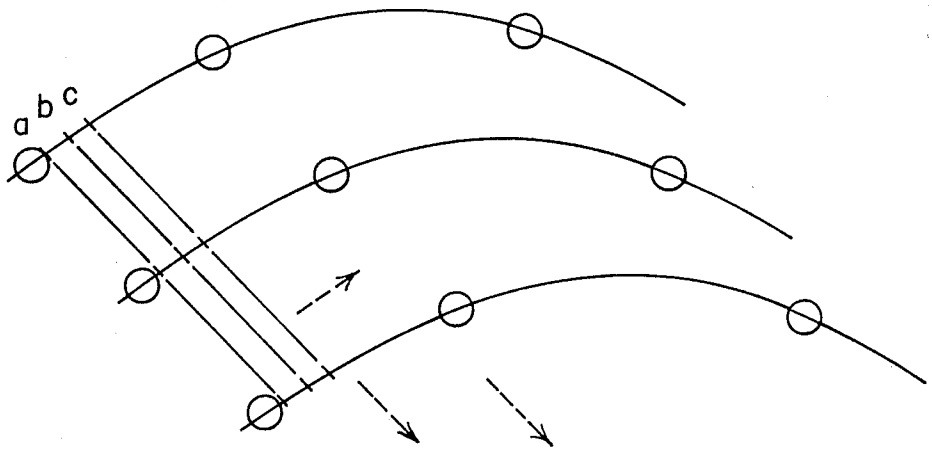


Figure A4-1 Limb "Slices" in the 'z' (Longitudinal) Direction



○ data points on slices

a,b,c spline points generated in circumferential direction

—— spline points generated in longitudinal direction

Figure A4-2 Generating Splines in the Longitudinal and Circumferential Direction

TABLE 2

The following values, when curve-fitted using spline functions produced the entire limb shape plotted in Figure A4 - 3.

	θ													
r	0	30	60	90	120	150	180	210	240	270	300	330	360	Slice
	6.7	7.3	8.6	8.8	8.9	7.4	6.7	7.4	9.0	8.9	9.0	7.4	6.7	1
	6.7	7.3	8.6	8.8	8.9	7.4	6.7	7.4	9.0	8.9	9.0	7.4	6.7	2
	6.7	7.3	8.6	8.8	8.9	7.4	6.7	7.4	9.0	8.9	9.0	7.4	6.7	3
	6.6	7.1	8.4	8.7	8.8	7.2	6.6	7.3	8.8	8.8	8.8	7.3	6.6	4
	6.4	6.8	8.1	8.4	8.7	6.9	6.2	7.0	8.4	8.3	8.4	7.0	6.4	5
	6.1	6.3	7.7	8.0	8.3	6.5	5.7	6.6	7.9	7.8	7.9	6.7	6.1	6
	5.7	5.6	7.1	7.4	7.7	6.0	5.3	6.1	7.3	7.1	7.4	6.2	5.7	7
	5.2	4.9	6.4	6.8	7.0	5.5	4.9	5.6	6.7	6.6	6.8	5.7	5.2	8
	4.6	4.2	5.4	5.7	6.0	4.7	4.4	5.1	6.2	6.0	6.4	5.2	4.6	9
	4.5	4.1	5.3	5.6	5.9	4.6	4.3	5.0	6.1	5.9	6.3	5.1	4.5	10
	4.5	4.1	5.2	5.5	5.8	4.5	4.2	4.9	6.0	5.9	6.2	5.0	4.5	11
	4.5	4.1	5.2	5.4	5.7	4.5	4.2	4.9	6.1	6.0	6.3	5.1	4.5	12
	4.5	4.1	5.1	5.3	5.6	4.4	4.2	5.0	6.3	6.3	6.6	5.4	4.5	13
	4.6	4.2	4.9	5.2	5.3	4.3	4.3	5.1	6.7	6.8	7.1	5.9	4.6	14
	4.6	4.2	4.8	5.0	5.0	4.1	4.2	5.2	7.1	7.4	7.7	6.5	4.6	15
	4.6	4.1	4.6	4.8	4.7	3.8	4.1	4.8	7.1	8.0	8.6	6.7	4.6	16
	4.5	4.0	4.4	4.6	4.3	3.6	3.9	4.5	6.9	8.1	9.0	6.5	4.5	17
	4.5	3.9	4.2	4.4	3.9	3.4	3.7	4.2	6.5	7.9	8.9	6.4	4.5	18
	4.4	3.8	4.1	4.1	3.8	3.3	3.4	4.0	6.2	7.8	8.8	6.3	4.4	19
	4.4	3.8	4.0	3.9	3.7	3.2	3.3	3.9	6.1	7.7	8.7	6.3	4.4	20

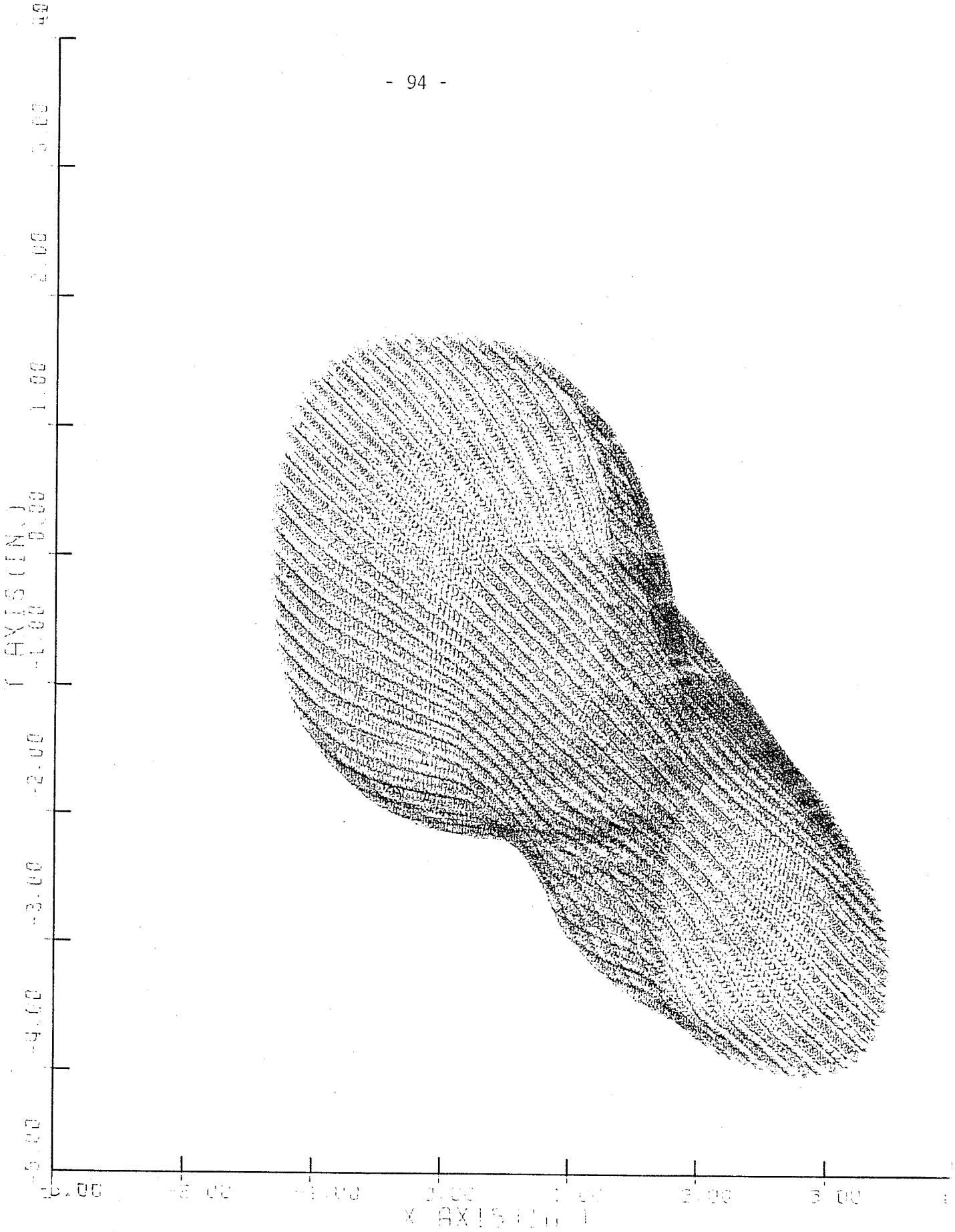


Figure A4-3. Total Limb Reproduction

```
C
C     ***********************************************************
C
C     PICTURE3 PROGRAM
C
C     FUNCTION:
C
C     THIS PROGRAM PRODUCES AN AXONOMETRIC(3 DIMENSIONAL)
C     PLOT OF A LIMB.  THE DATA INPUT IS IN POLAR(R-a)
C     COORDINATES.  THE OUTPUT IS IN CARTESIAN(X-Y-Z)
C     COORDINATES.  NO NUMERIC LISTING OF PLOTTED POINTS
C     IS PROVIDED, DUE TO THE LARGE NUMBER OF POINTS
C     INVOLVED.
C
C     NOTE THAT THIS PROGRAM IS CONSTRUCTED TO PLOT ONLY
C     FOR TWENTY "SLICES" IN THE Z(ALONG THE LIMB)
C     DIRECTION.  FOR DIFFERENT NUMBERS OF "SLICES",
C     THE PROGRAM WILL HAVE TO BE CHANGED
C
C     DO DATA INPUTS AS FOLLOWS.....
C          CARD I
C          -NUMBER OF DATA POINTS          1I4
C          CARD II
C          -PROBE DISPLACEMENT(INCHES)     13F6.1
C          CARD III
C          -PROBE POSITION(DEGREES)        13F6.1
C
C     ***********************************************************
C
      DIMENSION XI(30,30),YI(30,30),S(30,30)
      DIMENSION IBUF(1000),XPL(300),YPL(300)
      REAL X(30,30),Y(30,30),XSP(30,30),YSP(30,30)
      REAL M(30,30),N(30,30),ML(30,30),NL(30,30)
      REAL HL(30,30),Z(30,30),H(30,30)
      REAL A(30,30),SL(30,30),B1(30),B2(30)
      REAL XNSP(30,30),ZNSP(30,30),YNSP(30,30)
      REAL XPRNT(30,30),YPRNT(30,30),ZPRNT(30,30)
      REAL RADIUN(20),ANGLN(20),RADIUS(20),ANGLE(20)
C
C     ***********************************************************
C
C     INITIALIZE COUNTER VALUES AND PLOT ROUTINES
C
      NN=13
      MM=20
      NM=MM-1
      LLL=0
      CALL PLOTS(IBUF,1000)
      CALL PLOT(0.0,1.0,-3)
      XPL(134)=-3.0
      XPL(135)=4.0
      YPL(134)=-5.0
      YPL(135)=1.0
      PCEF=0.0
      CALL AXIS(0.0,0.0,11HX AXIS(IN.),-11,
     17.0,0.0,XPL(134),XPL(135))
      CALL AXIS(0.0,0.0,11HY AXIS(IN.),11,
     19.0,90.0,YPL(134),YPL(135))
```

```
C
C     ****************************************************************
C
C
C          DATA INPUTS AND CONVERSIONS(POLAR TO CARTESIAN)
C
  200  FORMAT(I14)
  201  FORMAT(13F6.1)
       DO 1 K=1,MM
       READ(5,200) NUT
       READ(5,201) (RADIUN(I),I=1,NUT)
       READ(5,201) (ANGLN(I),I=1,NUT)
       DO 11 I=1,NUT
       ANGLE(I)=ANGLN(I)*3.14/180.
  11   RADIUS(I)=RADIUN(I)/2.54
       DO 40 I=1,NUT
       IF(ANGLE(I).GT.3.14/2.) GO TO 50
       XI(K,I)=RADIUS(I)*COS(ANGLE(I))
       YI(K,I)=RADIUS(I)*SIN(ANGLE(I))
       GO TO 40
  50   IF(ANGLE(I).GT.3.14) GO TO 60
       ANGLE(I)=3.14-ANGLE(I)
       XI(K,I)=-RADIUS(I)*COS(ANGLE(I))
       YI(K,I)=RADIUS(I)*SIN(ANGLE(I))
       GO TO 40
  60   IF(ANGLE(I).GT.3.*3.14/2.) GO TO 70
       ANGLE(I)=ANGLE(I)-3.14
       XI(K,I)=-RADIUS(I)*COS(ANGLE(I))
       YI(K,I)=-RADIUS(I)*SIN(ANGLE(I))
       GO TO 40
  70   IF(ANGLE(I).GT.2.*3.14) GO TO 40
       ANGLE(I)=2.*3.14-ANGLE(I)
       XI(K,I)=RADIUS(I)*COS(ANGLE(I))
       YI(K,I)=-RADIUS(I)*SIN(ANGLE(I))
  40   CONTINUE
C
C     ****************************************************************
C
C          CALL THE ROUTINE TO PRODUCE AXONOMETRIC DATA
C
       CALL OFFSET(XI,YI,POFF,K,MM,X,Y)
C
C     ****************************************************************
C
C          CALL THE ROUTINE WHICH CALCULATES THE SLOPES AT
C          EACH DATA POINT(FOR ANY GIVEN Z "SLICE")
C
       CALL CALC(K,LLL,X,Y,MM,F,S,M,N)
       POFF=POFF+.25
  1    CONTINUE
       LLL=1
       DO 6 K=1,3
       Z(K,1)=1.
       DO 6 J1=2,MM
  6    Z(K,J1)=Z(K,J1-1)+1.
       DO 5 J=1,IM
       DO 3 L=1,8
       DO 2 K=1,MM
```

```
C
C     ***********************************************************
C
C          CALL THE ROUTINE WHICH CALCULATES A SINGLE SPLINE
C          POINT ON ANY Z "SLICE"
C
      CALL SPLINE(X,Y,J,L,NN,K,F,S,N,M,XSP,YSP)
      XSP(L,K)=XSP(K,L)
      YSP(L,K)=YSP(K,L)
    2 CONTINUE
C
C     ***********************************************************
C
C          CALL THE ROUTINE TO CALCULATE THE SLOPES AT
C          CORRESPONDING SPLINE POINTS ON ALL Z "SLICES"
C          (FOR X AND Z VALUES)
C
      CALL CALC(L,LLL,XSP,Z,MM,FL,SL,ML,NL)
C
C     ***********************************************************
C
C          CALL THE ROUTINE WHICH CALCULATES X AND Z
C          SPLINE POINTS IN THE LONGITUDINAL DIRECTION
C
      CALL SPLION(XSP,Z,L,MM,FL,SL,ML,NL,XNSP,ZNSP)
      LOUT=MM-1
      DO 8 INC=1,LOUT
      DO 7 KCONT=1,7
      XPRNT(INC,KCONT)=XNSP(INC,KCONT)
      ZPRNT(INC,KCONT)=ZNSP(INC,KCONT)
    7 CONTINUE
    8 CONTINUE
C
C     ***********************************************************
C
C          CALL THE ROUTINE TO CALCULATE THE SLOPES AT
C          CORRESPONDING SPLINE POINTS ON ALL Z "SLICES"
C          (FOR Y AND Z VALUES)
C
      CALL CALC(L,LLL,YSP,Z,MM,FL,SL,ML,NL)
C
C     ***********************************************************
C
C          CALL THE ROUTINE WHICH CALCULATES Y AND Z
C          SPLINE POINTS IN THE LONGITUDINAL DIRECTION
C
      CALL SPLION(YSP,Z,L,MM,FL,SL,ML,NL,YNSP,ZNSP)
      DO 10 INC=1,LOUT
      DO 9 KCONT=1,7
      YPRNT(INC,KCONT)=YNSP(INC,KCONT)
    9 CONTINUE
   10 CONTINUE
C
C     ***********************************************************
C
C          CALL THE ROUTINE TO ASSEMBLE THE CALCULATED
C          SPLINE POINTS FOR PLOTTING
C          PLOT THE POINTS
```

```
C
      CALL CURPLT(XPRNT,YPRNT,MM,XPL,YPL)
      CALL LINE(XPL,YPL,133,1,1,1)
    3 CONTINUE
    5 CONTINUE
      CALL PLOT(15.0,0.0,999)
      STOP
      END
```

```
C
C     ***********************************************************
C
C          SUBROUTINE CALC
C
C          THIS ROUTINE CALCULATES THE "KNOTS"(SLOPES)
C          AT THE DATA POINTS OF ANY LONGITUDINAL(Z) SLICE
C
      SUBROUTINE CALC(K,LLL,X,Y,NN,H,S,M,N)
      REAL X(30,30),Y(30,30),H(30,30),S(30,30)
      REAL M(30,30),A(30,30),N(30,30),B1(30),B2(30)
C
C     ***********************************************************
C
C          INITIALIZE VALUES
C
      DO 4 I=1,30
      DO 4 J=1,30
    4 A(I,J)=0.0
      NM=NN-1
      NM2=NN-2
      S(K,1)=0.0
C
C     ***********************************************************
C
C          CALCULATE CHORDAL AND CUMULATIVE CHORDAL DISTANCES
C
      DO 950 I=1,NN
      IND=I+1
      IF(IND.LE.NN) GO TO 950
      IND=I-NN+2
  950 H(K,I)=SQRT((X(K,IND)-X(K,I))**2+
     1(Y(K,IND)-Y(K,I))**2)
      DO 951 I=1,NM
      KK=I+1
  951 S(K,KK)=S(K,I)+H(K,I)
C
C     ***********************************************************
C
C          COMPUTE THE MATRIX, THE X AND Y VECTORS
C
      DO 250 I=1,NM
      HP=H(K,I)
      HP1=H(K,I+1)
      A(I,I)=(1./HP+1./HP1)*2
      IND=I+2
      IF(IND.LE.NN) GO TO 75
      IND=NP1-I+1
   75 B1(I)=3./(HP**2*HP1**2)*(HP**2*Y(K,IND)
     1-Y(K,I+1)*(HP**2-HP1**2)-HP1**2*Y(K,I))
  250 B2(I)=3./(HP**2*HP1**2)*(HP**2*X(K,IND)
     1-X(K,I+1)*(HP**2-HP1**2)-HP1**2*X(K,I))
      DO 251 J=1,NM2
      A(J+1,J)=1./H(K,J+1)
  251 A(J,J+1)=1./H(K,J+1)
      IF(LLL.NE.1) GO TO 5
      A(1,NM)=1./H(K,1)
      A(NM,1)=1./H(K,NN)
```

```
C
C     ***********************************************************
C                          - 100 -
C     TRIANGULARIZE THE MATRIX THEN MODIFY THE VECTORS
C
  5   DO 100 I=1,NM2
      KK=NM-I
      KP1=KK+1
      FACT1=A(KK,KP1)/A(KP1,KP1)
      B1(KK)=B1(KK)-B1(KP1)*FACT1
      B2(KK)=B2(KK)-B2(KP1)*FACT1
      DO 101 J=1,KP1
 101  A(KK,J)=A(KK,J)-A(KP1,J)*FACT1
      FACT2=A(1,KP1)/A(KP1,KP1)
      B1(1)=B1(1)-B1(KP1)*FACT2
      B2(1)=B2(1)-B2(KP1)*FACT2
      DO 102 JT=1,KP1
 102  A(1,JT)=A(1,JT)-A(KP1,JT)*FACT2
      DO 104 J=1,NM
      IF(ABS(A(KK,J)).LE.0.00001) A(KK,J)=0.0
 104  CONTINUE
 100  CONTINUE
C     ***********************************************************
C
C     CALL THE ROUTINE TO SOLVE THE EQUATIONS
C
      CALL SOLVE(K,NM,A,B2,N)
      CALL SOLVE(K,NM,A,B1,M)
      RETURN
      END
```

```
C
C     ***********************************************************
C
C     SUBROUTINE SPLICN
C
C     USED TO CALCULATE SETS OF SPLINE POINTS
C     (X AND Z, Y AND Z) IN THE LONGITUDINAL DIRECTION
C
      SUBROUTINE SPLICN(XSP,Z,L,MM,HL,SL,ML,NL,XNSP,ZNSP)
      REAL XSP(30,30),Z(30,30),HL(30,30),SL(30,30)
      REAL XNSP(30,30),ZNSP(30,30),ML(30,30),NL(30,30)
      NM=MM-1
      DO 1 J=1,IM
      INX=J-1
      IF(INX.LT.0.0) INX=NM-J
      DO 2 I=1,7
      SP=I*HL(L,J)/7.+SL(L,J)
      XNSP(J,I)=HL(L,INX)*(SL(L,J+1)-SP)**2*(SP-SL(L,J))
     1/HL(L,J)**2-NL(L,J)*(SP-SL(L,J))**2*(SL(L,J+1)
     1-SP)/HL(L,J)**2+XSP(L,J)*(SL(L,J+1)-SP)**2*(2*
     1(SP-SL(L,J))+HL(L,J))/HL(L,J)**3+XSP(L,J+1)*(SP-
     1SL(L,1))**2*(2*(SL(L,J+1)-SP)+HL(L,J))/HL(L,J)**3
      ZNSP(J,I)=ML(L,INX)*(SL(L,J+1)-SP)**2*(SP-SL(L,J))
     1/HL(L,J)**2-ML(L,J)*(SP-SL(L,J))**2*(SL(L,J+1)-SP)
     1/HL(L,J)**2+Z(L,J)*(SL(L,J+1)-SP)**2*(2*(SP
     1-SL(L,J))+HL(L,J))/HL(L,J)**3+Z(L,J+1)*(SP-SL
     1(L,J))**2*(2*(SL(L,J+1)-SP)+HL(L,J))/HL(L,J)**3
2     CONTINUE
1     CONTINUE
      RETURN
      END
```

```
C
C     ***********************************************************
C
C     SUBROUTINE SPLINE
C
C     THIS GENERATES A SINGLE SPLINE POINT
C     IN ANY GIVEN Z "SLICE" BETWEEN ANY TWO
C     GIVEN DATA POINTS FOR A PARTICULAR SLICE
C
      SUBROUTINE SPLINE(X,Y,J,L,NN,K,H,S,N,M,XSP,YSP)
      REAL H(30,30),S(30,30),N(30,30),XSP(30,30)
      REAL YSP(30,30),X(30,30),Y(30,30),M(30,30)
      INX=J-1
      IF(INX.EQ.0) INX=NN-J
      SP=L*H(K,J)/2.0+S(K,J)
      XSP(K,L)=N(K,INX)*(S(K,J+1)-SP)**2*(SP-S(K,J))
     1/H(K,J)**2-N(K,J)*(SP-S(K,J))**2*(S(K,J+1)-SP)
     1/H(K,J)**2+X(K,J)*(S(K,J+1)-SP)**2*(2*(SP-S(K,J)
     1)+H(K,J))/H(K,J)**3+X(K,J+1)*(SP-S(K,J))**2*(2
     1*(S(K,J+1)-SP)+H(K,J))/H(K,J)**3
      YSP(K,L)=M(K,INX)*(S(K,J+1)-SP)**2*(SP-S(K,J))
     1/H(K,J)**2-M(K,J)*(SP-S(K,J))**2*(S(K,J+1)-SP)
     1/H(K,J)**2+Y(K,J)*(S(K,J+1)-SP)**2*(2*(SP-S(K,J))
     1+H(K,J))/H(K,J)**3+Y(K,J+1)*(SP-S(K,J))**2*(2
     1*(S(K,J+1)-SP)+H(K,J))/H(K,J)**3
      RETURN
      END
```

```
C
C     ***********************************************************
C
C          SUBROUTINE SOLVE
C
C          USED TO SOLVE THE SYSTEM OF LINEAR EQUATIONS
C          WHICH DETERMINE THE KNOTS OF THE SPLINE FUNCTION
C
      SUBROUTINE SOLVE(K,NM,A,B,X)
      REAL A(30,30),B(30),X(30,30)
      X(K,1)=B(1)/A(1,1)
      DO 25 I=2,NM
      KOUNT=I-1
      X(K,I)=B(I)/A(I,I)
      DO 30 J=1,KOUNT
30    X(K,I)=X(K,I)-(A(I,J)*X(K,J))/A(I,I)
25    CONTINUE
      RETURN
      END
```

```
C
C     ***********************************************************
C
C     SUBROUTINE OFFSET
C
C     THIS MODIFIES THE DATA SO THAT THE PLOTTED
C     SPLINE POINTS APPEAR TO FORM A 3-D SHAPE
C
      SUBROUTINE OFFSET(XI,YI,POFF,K,NN,X,Y)
      DIMENSION XI(30,30),YI(30,30),X(30,30),Y(30,30)
      DO 1 I=1,NN
      X(K,I)=(XI(K,I)+POFF)*.50
    1 Y(K,I)=(YI(K,I)-POFF)*.50
      RETURN
      END
```

```
C
C     ***********************************************************
C
C          SUBROUTINE CURPLT
C
C          THIS ROUTINE PREPARES AN ENTIRE LONGITUDINAL
C          SPLINE STRIP(Z-DIRECTION) FOR PLOTTING
C
      SUBROUTINE CURPLT(XPRNT,YPRNT,MM,XPL,YPL)
      DIMENSION XPL(300),YPL(300),XPRNT(30,30),YPRNT(30,30)
      DO 3 K=1,7
      XPL(K)=XPRNT(1,K)
    3 YPL(K)=YPRNT(1,K)
      LP1=7
      LP=133
      LCNT=0
      J=2
      DO 1 I=8,LP
      LCNT=LCNT+1
      XPL(I)=XPRNT(J,I-LP1*(J-1))
      YPL(I)=YPRNT(J,I-LP1*(J-1))
      IF(LCNT.NE.7) GO TO 1
      LCNT=0
      J=J+1
    1 CONTINUE
      RETURN
      END
```

APPENDIX V

This appendix contains the computer program, with numerical and graphical results, which reduces a "large" data set to a "small" data set. The "large" data set is the same set of points as found in Appendix III.

The reduction procedure is as follows:

a) assuming that the data set is labelled as shown in Figure A5-1, point 1 is dropped, and a spline function is generated.

b) the spline function is closely checked to see whether or not it passes within 1/16" of point 1 (the excluded point - Figure A5-2).

c) if the distance between the spline function and the excluded point 1 is greater than 1/16", the point is retained as a member of the presently under creation "small" data set. Then point 2 is excluded and an entire spline function is generated, with the same checking at point 2 being carried out between points 1 and 3 (Figure A5-3).

d) if the distance between the spline function and the excluded point 1 is less than or equal to 1/16", the point is excluded from the new "small" data set. Point 2 is then excluded, and a spline function is generated. Checking is now done at points 1 and 2 between points 11 and 3 (Figure A5-4).

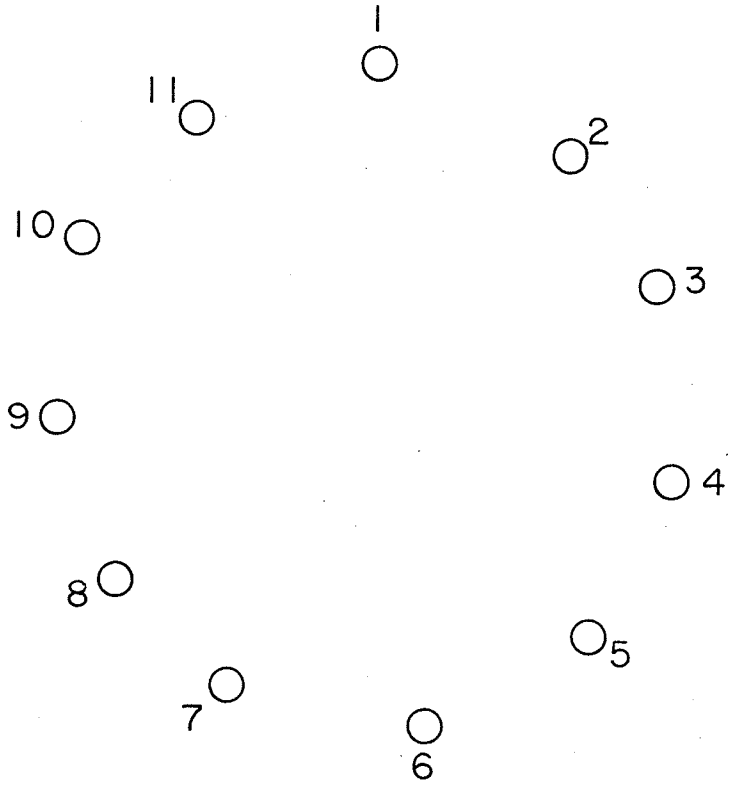
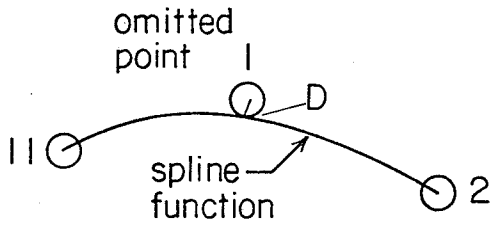


Figure A5-1 Labelled Data Set

(Note: not used in the production of numerical and graphical results in this appendix.)



if:
D > 1/16" save pt. 1
D ≤ 1/16" omit pt. 1

Figure A5-2
Point Exclusion

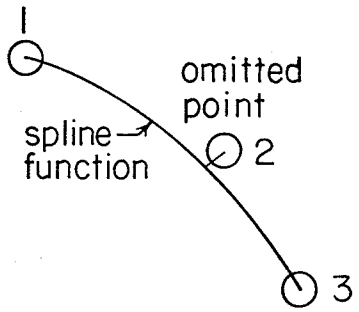


Figure A5-3
Omit Point 2

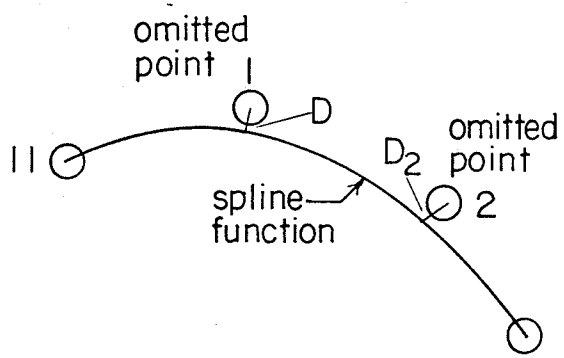


Figure A5-4
Checking Points 1 and 2

e) if either D_1 or D_2 is greater than 1/16", then point 2 is included in the "small" data set. Otherwise, both points are dropped, and the effect of excluding examined in the manner just outlined in the preceeding steps.

f) this procedure is carried out for every point in the data set. Once this has been done, a new "small" data set remains. This data set can produce (within 1/16" limits) the same shape that the "large" data set can produce.

This completes the reduction procedure. A computer program listing of this reduction algorithm, with graphical and numerical output follows (Figure A5-5).

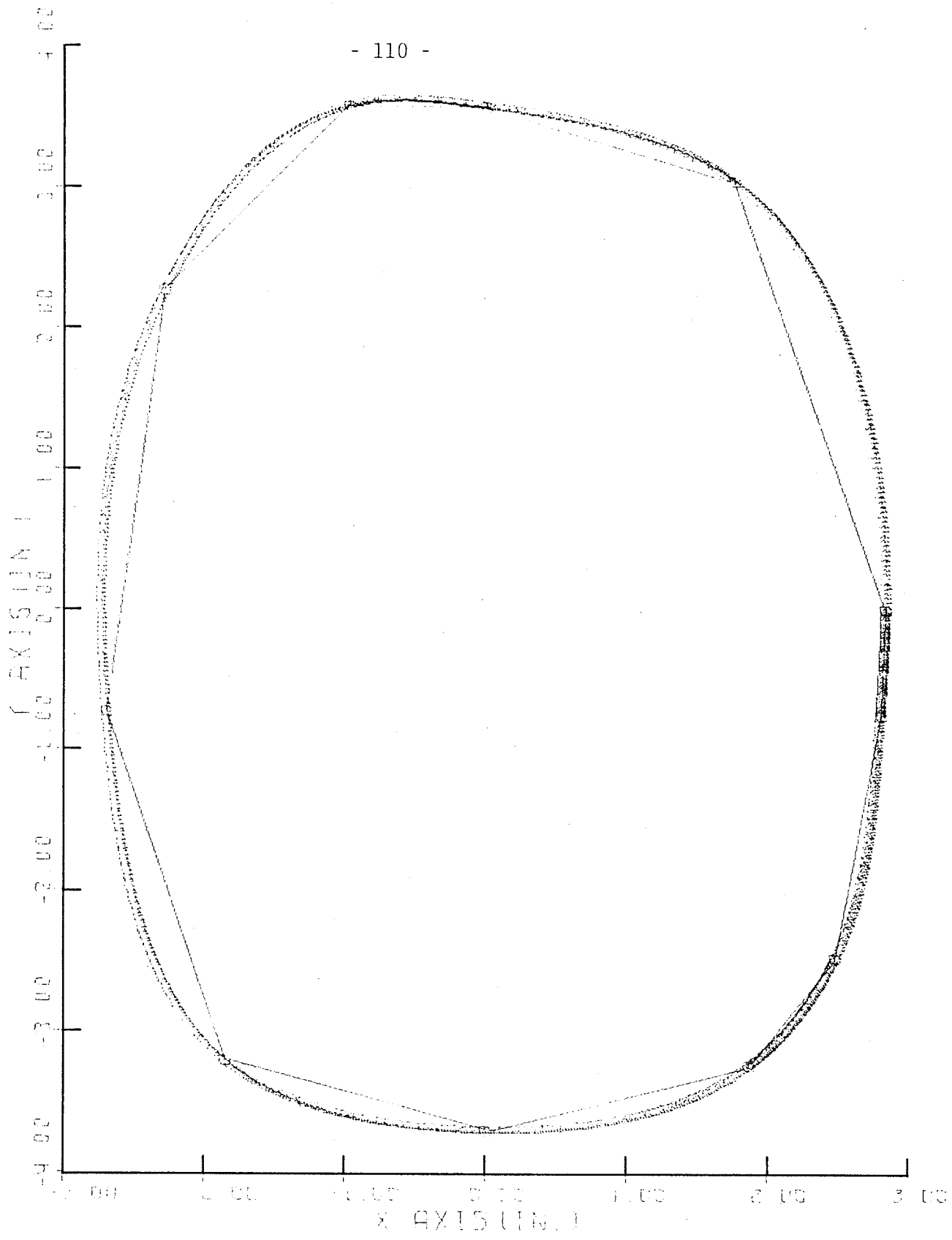


Figure A5-5 Data Reduction Results

```
C
C          ***************************************************
C                          - 111 -
C          REDUCE PROGRAM
C
C          FUNCTION:
C
C          THIS PROGRAM TAKES A DENSE SET OF DATA POINTS
C          OBTAINED FROM EXPERIMENTAL LIMB CROSS-SECTIONS, AND
C          REDUCES THE NUMBER OF DATA POINTS REQUIRED TO
C          PRODUCE AN ACCURATE CROSS-SECTIONAL REPLICA.  THE
C          TOLERANCE ON INTERPOLATED POINTS IS 1/16".
C
C          DO DATA INPUTS AS FOLLOWS.....
C              CARD I
C              -NUMBER OF DATA POINTS          1I4
C              CARD II
C              -PROBE DISPLACEMENT(INCHES)     13F6.1
C              CARD III
C              -PROBE POSITION(DEGREES)        13F6.1
C
C          ***************************************************
C
      REAL XN1(60,60),YN1(60,60),RADIUS(60),ANGLE(60)
      REAL A(60,60),M(60),N(60),X(60),Y(60),B1(60)
      REAL XX(60),YY(60),B2(60),H(60),S(60),MO,NO
      INTEGER ELIM,SAVE
C
C          ***************************************************
C
C          DATA INPUTS AND TRANSFORMATIONS
C
      READ(5,2000) NN
 2000 FORMAT(1I4)
 2001 FORMAT(12F6.1)
      READ(5,2001) (RADIUS(I),I=1,NN)
      READ(5,2001) (ANGLE(I),I=1,NN)
      DO 329 I=1,NN
      ANGLE(I)=ANGLE(I)*3.14/180.
  329 RADIUS(I)=RADIUS(I)/2.54
C
C          ***************************************************
C
C          CALL THE ROUTINE WHICH TRANSFORMS THE DATA FROM
C          R-@ COORDINATES TO X-Y COORDINATES
C
      CALL CHANGE(RADIUS,ANGLE,NN,X,Y)
C
C          ***************************************************
C
C          VECTOR AND COUNTER INITIALIZATIONS
C
      DO 555 I=1,NN
      DO 555 J=1,NN
  555 A(I,J)=0.0
      LLL=1
      NNN=2
  257 IF(NNN.EQ.1) GO TO 416
```

```
      IF(NNN.EQ.4) GO TO 259
      IF(NNN.GT.NN/4) GO TO 259
      NDIN=(NN-2)/NNN+2
      INDEX=(NN-2)/NNN+1
      GO TO 417
  416 NDIN=NN
      INDEX=NN-1
  417 NSUB=NNN-1
      NM=NDIN-1
      NM2=NDIN-2
      S(1)=0.0
C
C     ************************************************************
C
C     START THE PROCESS OF ELIMINATING ONE DATA POINT
C     AND CHECKING TO SEE WHETHER THE SPLINE FALLS
C     WITHIN 1/16" OF THE "DISCARDED" POINT
C
      DO 111 I=2,INDEX
      XX(I)=X(NNN*I-NSUB)
      YY(I)=Y(NNN*I-NSUB)
  111 CONTINUE
      XX(NDIN)=X(NN)
      YY(NDIN)=Y(NN)
      XX(1)=X(1)
      YY(1)=Y(1)
C
C     ************************************************************
C
C     CALCULATE THE CHORDAL AND CUMULATIVE CHORDAL DISTANCES
C
      DO 950 I=1,NDIN
      INE=I+1
      IF(INE.LE.NDIN) GO TO 950
      INE=I-NDIN+2
  950 H(I)=SQRT((XX(INE)-XX(I))**2
     1+(YY(INE)-YY(I))**2)
      DO 951 I=1,NM
      KK=I+1
  951 S(KK)=S(I)+H(I)
C
C     ************************************************************
C
C     COMPUTE THE MATRIX, THE X AND Y VECTORS
C
      DO 250 I=1,NM
      HP=H(I)
      HP1=H(I+1)
      A(I,I)=(1./HP + 1./HP1)*2.
      IND=I+2
      IF(IND.LE.NDIN) GO TO 75
      IND=NDIN-I+1
   75 B1(I)=3./(HP**2*HP1**2)*(HP**2*YY(IND)
     1-YY(I+1)*(HP**2-HP1**2)-HP1**2*YY(I))
  250 B2(I)=3./(HP**2*HP1**2)*(HP**2*XX(IND)
     1-XX(I+1)*(HP**2-HP1**2)-HP1**2*XX(I))
      DO 251 J=1,NM2
      A(J+1,J)=1./H(J+1)
```

```
  251  A(J,J+1)=1./H(J+1)
       A(1,NM)=1./H(1)
       A(NM,1)=1./H(NDIN)
C
C          ***************************************************
C
C          TRIANGULARIZE THE MATRIX THEN MODIFY THE VECTORS
C
       DO 100 I=1,NM2
       K=NM-I
       KP1=K+1
       FACT1=A(K,KP1)/A(KP1,KP1)
       B1(K)=B1(K)-B1(KP1)*FACT1
       B2(K)=B2(K)-B2(KP1)*FACT1
       DO 101 J=1,KP1
  101  A(K,J)=A(K,J)-A(KP1,J)*FACT1
       FACT2=A(1,KP1)/A(KP1,KP1)
       B1(1)=B1(1)-B1(KP1)*FACT2
       B2(1)=B2(1)-B2(KP1)*FACT2
       DO 102 JT=1,KP1
  102  A(1,JT)=A(1,JT)-A(KP1,JT)*FACT2
       DO 104 J=1,NM
       IF(ABS(A(K,J)).LE..000001) A(K,J)=0.0
  104  CONTINUE
  100  CONTINUE
C
C          ******** ** ***************************************
C
C          CALL THE ROUTINE TO SOLVE THE EQUATIONS
C
       CALL SOLVE(A,B1,NM,M)
       CALL SOLVE(A,B2,NM,N)
       II=1
C
C          ***************************************************
C
C          CHECK TO SEE IF THE INTERPOLATED SPLINE FALLS WITHIN
C          THE SPECIFIED TOLERANCE ONLY AT THE "DISCARDED"
C          DATA POINT
C
       DO 300 J=1,NM
       INX=J-1
       IF(INX.EQ.0) INX=NDIN-J
       MULT=20*NNN
       DO 302 K=1,NNN
       DIST1=100.
       DO 301 L=1,MULT
       SP=L*H(J)/(20*NNN)+S(J)
       XSP=N(INX)*(S(J+1)-SP)**2*(SP-S(J))/H(J)**2
      1-N(J)*(SP-S(J))**2*(S(J+1)-SP)/H(J)**2
      1+XX(J)*(S(J+1)-SP)**2*(2*(SP-S(J))+H(J))/H(J)**3
      1+XX(J+1)*(SP-S(J))**2*(2*(S(J+1)-SP)+H(J))/H(J)**3
       YSP=M(INX)*(S(J+1)-SP)**2*(SP-S(J))/H(J)**2
      1-M(J)*(SP-S(J))**2*(S(J+1)-SP)/H(J)**2
      1+YY(J)*(S(J+1)-SP)**2*(2*(SP-S(J))+H(J))/H(J)**3
      1+YY(J+1)*(SP-S(J))**2*(2*(S(J+1)-SP)+H(J))/H(J)**3
       D2X=(X(II+K)-XSP)**2
       D2Y=(Y(II+K)-YSP)**2
```

```
      DIST=SQRT(D2X+D2Y)
      IF(DIST.LE.DIST1) DIST1=DIST
  301 CONTINUE
      IF(DIST1.GE..0625) GO TO 304
      IF((NNN-1).EQ.K) GO TO 299
  302 CONTINUE
  299 II=II+NNN
      IF((II+NNN).GE.NN) GO TO 303
  300 CONTINUE
  303 NNN=NNN+1
      IF(LLL.EQ.2) GO TO 259
      GO TO 257
  304 NNN=NNN-1
      LLL=2
      GO TO 257
  259 CONTINUE
      WRITE(6,30) NNN
   30 FORMAT(' ',1I1)
      NNN=NNN-1
      NDIN=(NN-2)/NNN+2
      K=2
      J=1
      DO 14 I=1,NDIN
      XN1(I,J)=XX(I)
   14 YN1(I,J)=YY(I)
      ELIM=2
      ICOUNT=0
      SAVE=0
      GO TO 4
   12 ELIM=ELIM+1
   13  K=K+1
      J=J+1
      IF(K.EQ.46) GO TO 20
    4 DO 1 I=1,NDIN
      IF(I.GE.ELIM) GO TO 2
        XN1(I,K)=XN1(I,J)
      YN1(I,K)=YN1(I,J)
      GO TO 1
    2 XN1(I,K)=XN1(I+1,J)
      YN1(I,K)=YN1(I+1,J)
      IF(I.EQ.(NDIN-1)) GO TO 6
    1 CONTINUE
    6 NEW=NDIN-1
      NELIM=ELIM
C
C     ********** ********** ********** ********** ********** **********
C
C          CALL THE ROUTINE WHICH CHECKS TOLERANCES AT OTHER
C          "DROPPED" DATA POINTS
C
      CALL SPLSV(XN1,YN1,X,Y,ICOUNT
     1,ELIM,NELIM,SAVE,NNN,NEW,K,DIST1)
      K=K+1
      J=J+1
      IF(DIST1.GT..0625) GO TO 5
      NDIN=NDIN-1
      ICOUNT=ICOUNT+1
      SAVE=SAVE+1
```

```
      LOOPY1=NDIN-1
      DO 10 I=1,NDIN
      XN1(I,K)=XN1(I,J)
   10 YN1(I,K)=YN1(I,J)
      GO TO 13
    5 DO 7 I=1,NDIN
      XN1(I,K)=XN1(I,J-1)
    7 YN1(I,K)=YN1(I,J-1)
      SAVE=0
      GO TO 12
   20 DO 21 I=1,NDIN
   21 WRITE(6,22) XN1(I,K-1),YN1(I,K-1)
   22 FORMAT(' ',2F10.4)
      LXPAS=K-1
C
C     ***************************************************************
C
C        CALL THE ROUTINE WHICH PLOTS THE FINAL REDUCED DATA SET
C
      CALL CURGEN(XN1,YN1,NDIN,LXPAS)
      STOP
      END
```

```
C
C     ********** ********************************************************
C
C          SUBROUTINE SPLSV
C
C          THIS SUBROUTINE DETERMINES WHETHER DROPPING A DATA
C          POINT WILL ADVERSELY AFFECT THE TOLERANCE VALUE AT
C          A PREVIOUSLY ELIMINATED POINT
C
      SUBROUTINE SPLSV(XN,YN,X,Y,ICOUNT,ELIMGF
     1,NELIM,SAVE,NNN,NEW,K1,DIST1)
      REAL XN(60,60),YN(60,60),A(60,60),X(60)
      REAL Y(60),M(60),N(60),B1(60),B2(60)
      REAL XSP(300),YSP(300),H(60),S(60)
      INTEGER SAVE,ELIM,ELIMGF
      ELIM=ELIMGF+ICOUNT
      DO 555 I=1,NEW
      DO 555 J=1,NEW
  555 A(I,J)=0.0
      NM=NEW-1
      NM2=NEW-2
      S(1)=0.0
C
C     ********** ********************************************************
C
C          CALCULATE THE CHORDAL AND CUMULATIVE CHORDAL DISTANCES
C
      DO 950  I=1,NEW
      INE=I+1
      IF(INE.LE.NEW) GO TO 950
      INE=I-NEW+2
  950 H(I)=SQRT((XN(INE,K1)-XN(I,K1))**2
     1+(YN(INE,K1)-YN(I,K1))**2)
      DO 951  I=1,NM
      KK=I+1
  951 S(KK)=S(I)+H(I)
C
C     ********** ********************************************************
C
C          COMPUTE THE MATRIX, THE X AND Y VECTORS
C
      DO 250 I=1,NM
      HP=H(I)
      HP1=H(I+1)
      A(I,I)=(1./HP+1./HP1)*2.
      IND=I+2
      IF(IND.LE.NEW) GO TO 75
      IND=NEW-I+1
   75 B1(I)=3./(HP**2*HP1**2)*(HP**2*YN(IND,K1)
     1-YN(I+1,K1)*(HP**2-HP1**2)-HP1**2*YN(I,K1))
  250 B2(I)=3./(HP**2*HP1**2)*(HP**2*XN(IND,K1)
     1-XN(I+1,K1)*(HP**2-HP1**2)-HP1**2*XN(I,K1))
      DO 251 J=1,NM2
      A(J+1,J)=1./H(J+1)
  251 A(J,J+1)=1./H(J+1)
      A(1,NM)=1./H(1)
      A(NM,1)=1./H(NEW)
C
```

```
C     ***********************************************************
C
C          TRIANGULARIZE THE MATRIX THEN MODIFY THE VECTORS
C
      DO 100 I=1,NM2
      K=NM-I
      KP1=K+1
      FACT1=A(K,KP1)/A(KP1,KP1)
      B1(K)=B1(K)-B1(KP1)*FACT1
      B2(K)=B2(K)-B2(KP1)*FACT1
      DO 101 J=1,KP1
  101 A(K,J)=A(K,J)-A(KP1,J)*FACT1
      FACT2=A(1,KP1)/A(KP1,KP1)
      B1(1)=B1(1)-B1(KP1)*FACT2
      B2(1)=B2(1)-B2(KP1)*FACT2
      DO 102 JT=1,KP1
  102 A(1,JT)=A(1,JT)-A(KP1,JT)*FACT2
      DO 104 J=1,NM
      IF(ABS(A(K,J)).LE..000001) A(K,J)=0.0
  104 CONTINUE
  100 CONTINUE
C
C     ***********************************************************
C
C          CALL THE ROUTINE TO SOLVE THE EQUATIONS
C
      CALL SOLVE(A,B2,NM,N)
      CALL SOLVE(A,B1,NM,M)
      MULT=40*NNN+SAVE
      LOOP1=ELIM*NNN-2*(NNN-1+SAVE)
      LOOP2=ELIM*NNN
      INX=NELIM-2
      IF(INX.EQ.0) INX=NEW-1
C
C     ***********************************************************
C
C          GENERATE SPLINE VALUES
C
      DO 302 K=LOOP1,LOOP2
      DIST1=100
      DO 301 L=1,MULT
      SP=L*H(NELIM-1)/(40*NNN+SAVE)+S(NELIM-1)
      XSP(L)=N(INX)*(S(NELIM)-SP)**2*(SP-S(NELIM-1))
     1/H(NELIM-1)**2-N(NELIM-1)*(SP-S(NELIM-1))**2*(S(NELIM)
     1-SP)/H(NELIM-1)**2+XN(NELIM-1,K1)*(S(NELIM)-SP)**2
     1*(2*(SP-S(NELIM-1))+H(NELIM-1))/H(NELIM-1)**3+XN(NELIM
     1,K1)*(SP-S(NELIM-1))**2*(2*(S(NELIM)-SP)+H(NELIM-1))
     1/H(NELIM-1)**3
      YSP(L)=M(INX)*(S(NELIM)-SP)**2*(SP-S(NELIM-1))
     1/H(NELIM-1)**2-M(NELIM-1)*(SP-S(NELIM-1))**2*(S(NELIM)
     1-SP)/H(NELIM-1)**2+YN(NELIM-1,K1)*(S(NELIM)-SP)**2
     1*(2*(SP-S(NELIM-1))+H(NELIM-1))/H(NELIM-1)**3+YN(NELIM
     1,K1)*(SP-S(NELIM-1))**2*(2*(S(NELIM)-SP)+H(NELIM-1))
     1/H(NELIM-1)**3
      D2X=(X(K)-XSP(L))**2
      D2Y=(Y(K)-YSP(L))**2
      DIST=SQRT(D2X+D2Y)
      IF(DIST.LE.DIST1) DIST1=DIST
```

```
301   CONTINUE
      IF(DIST1.GE..0625) GO TO 304
302   CONTINUE
304   CONTINUE
      RETURN
      END
```

```
C
C     ********** ****** ***** ************************************
C
C          SUBROUTINE CURGEN
C
C          THIS ROUTINE GENERATES 100 SPLINE POINTS BETWEEN
C          THE REDUCED DATA SET POINTS.  THE SPLINE POINTS SO
C          PRODUCED ARE THEN PLOTTED AND PRINTED
C
      SUBROUTINE CURGEN(X,Y,NN,LXPAS)
      REAL X(60,60),Y(60,60),A(60,60),B1(60),B2(60)
      REAL H(60),S(60),M(60),N(60),XPL(60),YPL(60)
      DIMENSION IBUF(1000),XSP(120),YSP(120)
  799 FORMAT('1','THE SPLINE POINTS ARE GENERATED',
     1'BETWEEN THE FOLLOWING COORDINATES:')
  800 FORMAT(//,'   FROM X= ',F5.2,' ,Y= ',F5.2,
     1'   TO X= ',F5.2,' Y= ',F5.2)
  801 FORMAT(////T22,'X SPLINE    Y SPLINE')
  802 FORMAT(' ',T17,2F12.4)
C
C     **********************************************************
C
C          ASSIGN VECTOR VALUES AND INITIALIZE PLOT ROUTINES
C
       CALL PLOTS(IBUF,1000)
      DO 2 I=1,NN
      XPL(I)=X(I,LXPAS)
    2 YPL(I)=Y(I,LXPAS)
      CALL PLOT(0.0,1.5,-3)
      XPL(NN+1)=-3.0
      XPL(NN+2)=1.0
      YPL(NN+1)=-4.0
      YPL(NN+2)=1.0
      CALL AXIS(0.0,0.0,11HX AXIS(IN.),-11,
     16.0,0.0,XPL(NN+1),XPL(NN+2))
      CALL AXIS(0.0,0.0,11HY AXIS(IN.),11,
     18.0,90.0,YPL(NN+1),YPL(NN+2))
      CALL LINE(XPL,YPL,NN,1,1,1)
      DO 1 I=1,NN
      DO 1 J=1,NN
    1 A(I,J)=0.0
      NM=NN-1
      NM2=NN-2
      S(1)=0.0
C
C     **********************************************************
C
C          CALCULATE THE CHORDAL AND CUMULATIVE CHORDAL DISTANCES
C
      DO 950 I=1,NN
      INE=I+1
      IF(INE.LE.NN) GO TO 950
      INE=I-NN+2
  950 H(I)=SQRT((X(INE,LXPAS)-X(I,LXPAS))**2
     1+(Y(INE,LXPAS)-Y(I,LXPAS))**2)
      DO 951 I=1,NM
      KK=I+1
  951 S(KK)=S(I)+H(I)
```

```
C
C     ********** ****** *** ************************************************
C
C          COMPUTE THE MATRIX, THE X AND Y VECTORS
C
      DO 250 I=1,NM
      HP=H(I)
      HP1=H(I+1)
      A(I,I)=(1./HP+1./HP1)*2.
      IND=I+2
       IF(IND.LE.NN) GO TO 75
      IND=NN-I+1
   75 B1(I)=3./(HP**2*HP1**2)*(HP**2*Y(IND,LXPAS)
     1-Y(I+1,LXPAS)*(HP**2-HP1**2)-HP1**2*Y(I,LXPAS))
  250 B2(I)=3./(HP**2*HP1**2)*(HP**2*X(IND,LXPAS)
     1-X(I+1,LXPAS)*(HP**2-HP1**2)-HP1**2*X(I,LXPAS))
      DO 251 J=1,NM2
      A(J+1,J)=1./H(J+1)
  251 A(J,J+1)=1./H(J+1)
      A(1,NM)=1./H(1)
      A(NM,1)=1./H(NN)
C
C     ********* ***************************************************
C
C          TRIANGULARIZE THE MATRIX THEN MODIFY THE VECTORS
C
      DO 100 I=1,NM2
      K=NM-I
      KP1=K+1
      FACT1=A(K,KP1)/A(KP1,KP1)
      B1(K)=B1(K)-B1(KP1)*FACT1
      B2(K)=B2(K)-B2(KP1)*FACT1
      DO 101 J=1,KP1
  101 A(K,J)=A(K,J)-A(KP1,J)*FACT1
      FACT2=A(1,KP1)/A(KP1,KP1)
      B1(1)=B1(1)-B1(KP1)*FACT2
      B2(1)=B2(1)-B2(KP1)*FACT2
      DO 102 JT=1,KP1
  102 A(1,JT)=A(1,JT)-A(KP1,JT)*FACT2
      DO 104 J=1,NM
      IF(ABS(A(K,J)).LE..000001) A(K,J)=0.0
  104 CONTINUE
  100 CONTINUE
C
C     ******** ********************** ***************************************
C
C          CALL THE ROUTINE TO SOLVE THE EQUATIONS
C
      CALL SOLVE(A,B2,NM,N)
      CALL SOLVE(A,B1,NM,M)
C
C     ***************************************************************
C
C          GENERATE SPLINE VALUES
C
      DO 300 J=1,NM
      INX=J-1
      IF(INX.EQ.0) INX=NN-J
```

```
      DO 301 K=1,100
      L=K
      SP=L*H(J)/100.+S(J)
      XSP(K)=N(INX)*(S(J+1)-SP)**2*(SP-S(J))/H(J)**2
     1-N(J)*(SP-S(J))**2*(S(J+1)-SP)/H(J)**2
     1+X(J,LXPAS)*(S(J+1)-SP)**2*(2*(SP-S(J))+H(J))/H(J)**3
     1+X(J+1,LXPAS)*(SP-S(J))**2*(2*(S(J+1)-SP)+H(J))/H(J)**3
      YSP(K)=M(INX)*(S(J+1)-SP)**2*(SP-S(J))/H(J)**2
     1-M(J)*(SP-S(J))**2*(S(J+1)-SP)/H(J)**2
     1+Y(J,LXPAS)*(S(J+1)-SP)**2*(2*(SP-S(J))+H(J))/H(J)**3
     1+Y(J+1,LXPAS)*(SP-S(J))**2*(2*(S(J+1)-SP)+H(J))/H(J)**3
  301 CONTINUE
      XSP(K+1)=XPL(NN+1)
      XSP(K+2)=XPL(NN+2)
      YSP(K+1)=YPL(NN+1)
      YSP(K+2)=YPL(NN+2)
      CALL LINE(XSP,YSP,K,1,-1,2)
      PRINT799
      WRITE(6,800) X(J,LXPAS),Y(J,LXPAS),
     1X(J+1,LXPAS),Y(J+1,LXPAS)
      PRINT801
      DO 557 I=1,100
  557 WRITE(6,802) XSP(I),YSP(I)
  300 CONTINUE
      CALL PLOT(15.0,0.0,999)
      RETURN
      END
```

```
C     ***********************************************************
C
C          SUBROUTINE CHANGE
C
C          THIS ROUTINE CHANGES THE R-@ INPUT COORDINATE
C          VALUES TO X-Y COORDINATE VALUES
C
      SUBROUTINE CHANGE(RADIUS,ANGLE,NN,X,Y)
      REAL RADIUS(60),ANGLE(60),X(60),Y(60)
      DO 1 I=1,NN
      IF(ANGLE(I).GT.3.14/2.) GO TO 3
      X(I)=RADIUS(I)*COS(ANGLE(I))
      Y(I)=RADIUS(I)*SIN(ANGLE(I))
      GO TO 1
    3  IF(ANGLE(I).GT.3.14) GO TO 4
      ANGLE(I)=3.14-ANGLE(I)
      X(I)=-RADIUS(I)*COS(ANGLE(I))
      Y(I)=RADIUS(I)*SIN(ANGLE(I))
      GO TO 1
    4 IF(ANGLE(I).GT.3.*3.14/2.) GO TO 5
      ANGLE(I)=ANGLE(I)-3.14
      X(I)=-RADIUS(I)*COS(ANGLE(I))
      Y(I)=-RADIUS(I)*SIN(ANGLE(I))
      GO TO 1
    5 IF(ANGLE(I).GT.2.*3.14) GO TO 1
      ANGLE(I)=2.*3.14-ANGLE(I)
      X(I)=RADIUS(I)*COS(ANGLE(I))
      Y(I)=-RADIUS(I)*SIN(ANGLE(I))
    1 CONTINUE
      RETURN
      END
```

```
C
C     ***********************************************************
C
C     SUBROUTINE SOLVE
C
C     USED TO SOLVE THE SYSTEM OF LINEAR EQUATIONS
C     WHICH DETERMINE THE KNOTS OF THE SPLINE FUNCTION
C
      SUBROUTINE SOLVE(A,B,NM,X)
      REAL A(60,60),B(60),X(60)
      X(1)=B(1)/A(1,1)
      DO 25 I=2,NM
      KOUNT=I-1
      X(I)=B(I)/A(I,I)
      DO 30 J=1,KOUNT
   30 X(I)=X(I)-(A(I,J)*X(J))/A(I,I)
   25 CONTINUE
      RETURN
      END
```

```
8346     0.0
7634     3.0506
0029     3.5827
9653     3.5931
2837     2.2819
7001    -0.7231
8521    -3.2040
0030    -3.7008
8718    -3.2381
4786    -2.4767
7952    -0.7486
8346    -0.0000
```

SPLINE POINTS ARE GENERATED BETWEEN THE FOLLOWING COORDINATES:

OM X= 2.83 ,Y= 0.0 TO X= 1.76 Y= 3.05

X SPLINE	Y SPLINE
2.8351	0.0332
2.8353	0.0666
2.8354	0.1001
2.8353	0.1337
2.8351	0.1675
2.8347	0.2013
2.8341	0.2353
2.8333	0.2694
2.8323	0.3035
2.8311	0.3378
2.8298	0.3721
2.8283	0.4065
2.8266	0.4410
2.8247	0.4756
2.8226	0.5102
2.8203	0.5448
2.8178	0.5795
2.8151	0.6143
2.8122	0.6491
2.8091	0.6839
2.8059	0.7188
2.8024	0.7536
2.7987	0.7885
2.7948	0.8234
2.7907	0.8583
2.7864	0.8932
2.7818	0.9281
2.7771	0.9629
2.7721	0.9978
2.7670	1.0326
2.7616	1.0674
2.7560	1.1021
2.7501	1.1368
2.7441	1.1714
2.7378	1.2060
2.7313	1.2406
2.7245	1.2750
2.7175	1.3094
2.7103	1.3437
2.7029	1.3779
2.6952	1.4120
2.6873	1.4460
2.6792	1.4800
2.6708	1.5138
2.6621	1.5475
2.6532	1.5810
2.6441	1.6145
2.6347	1.6478
2.6251	1.6810
2.6152	1.7140
2.6051	1.7469
2.5947	1.7796
2.5841	1.8121
2.5732	1.8445
2.5620	1.8767
2.5506	1.9088

2.5270	1.9723
2.5148	2.0037
2.5023	2.0350
2.4895	2.0660
2.4765	2.0969
2.4632	2.1275
2.4496	2.1578
2.4358	2.1880
2.4216	2.2179
2.4072	2.2476
2.3925	2.2770
2.3776	2.3061
2.3623	2.3350
2.3468	2.3636
2.3309	2.3920
2.3148	2.4201
2.2984	2.4478
2.2817	2.4753
2.2647	2.5025
2.2474	2.5294
2.2298	2.5560
2.2119	2.5823
2.1937	2.6082
2.1751	2.6338
2.1563	2.6591
2.1372	2.6841
2.1178	2.7087
2.0980	2.7329
2.0780	2.7568
2.0576	2.7803
2.0369	2.8035
2.0159	2.8263
1.9946	2.8487
1.9730	2.8707
1.9510	2.8924
1.9287	2.9136
1.9061	2.9344
1.8831	2.9548
1.8599	2.9748
1.8363	2.9944
1.8123	3.0136
1.7880	3.0323
1.7634	3.0506

APPENDIX VI

This appendix contains a listing of a program written in BASIC which produces a limb cross-section using spline techniques. Also included are some of the points generated by this method.

```
10  DIM X[12],Y[12],A[12,12],M[12],L[12]
20  DIM B[12],C[12],H[12],S[12]
30  INPUT N
32  FOR K=1 TO N
33  INPUT X[K],Y[K]
34  NEXT K
40  FOR I=1 TO N
50  FOR J=1 TO N
60  LET A[I,J]=0
70  NEXT J
80  NEXT I
90  LET N1=N-1
100  LET N2=N-2
110  LET S[1]=0
120  FOR I=1 TO N
130  LET I2=I+1
140  IF I2 <= N THEN 160
150  LET I2=I-N+2
160  LET T1=(X[I2]-X[I])↑2
170  LET T2=(Y[I2]-Y[I])↑2
180  LET H[I]=SQR(T1+T2)
190  NEXT I
200  FOR I=1 TO N1
210  LET K=I+1
220  LET S[K]=S[I]+H[I]
230  NEXT I
240  FOR I=1 TO N1
250  LET H1=H[I]
260  LET H2=H[I+1]
270  LET A[I,I]=(1/H1+1/H2)*2
280  LET I3=I+2
290  IF I3 <= N THEN 310
300  LET I3=N-I+1
310  LET R1=H1↑2*H2↑2
320  LET R2=H1↑2*Y[I3]
330  LET R3=Y[I+1]*(H1↑2-H2↑2)
340  LET R4=H2↑2*Y[I]
350  LET B[I]=3/(R1)*(R2-R3-R4)
360  LET R5=H1↑2*X[I3]
370  LET R6=X[I+1]*(H1↑2-H2↑2)
380  LET R7=H2↑2*X[I]
390  LET C[I]=3/(R1)*(R5-R6-R7)
400  NEXT I
410  FOR J=1 TO N2
420  LET A[J+1,J]=1/H[J+1]
430  LET A[J,J+1]=1/H[J+1]
440  NEXT J
450  LET A[1,N1]=1/H[1]
460  LET A[N1,1]=1/H[N]
470  FOR I=1 TO N2
480  LET K2=N1-I
490  LET K1=K2+1
500  LET F1=A[K2,K1]/A[K1,K1]
510  LET B[K2]=B[K2]-B[K1]*F1
520  LET C[K2]=C[K2]-C[K2]*F1
530  FOR J=1 TO K1
540  LET A[K2,J]=A[K2,J]-A[K1,J]*F1
550  NEXT J
560  LET F2=A[1,K1]/A[K1,K1]
570  LET B[1]=B[1]-B[K1]*F2
```

```
580  LET C[1]=C[1]-C[K1]*F2
590  FOR J1=1 TO K1
600  LET A[I,J1]=A[I,J1]-A[K1,J1]*F2
610  NEXT J1
620  FOR J=1 TO N1
630  IF ABS(A[K2,J]) <= 1.00000E-06 THEN 650
640  GOTO 660
650  LET A[K2,J]=0
660  NEXT J
670  NEXT I
680  LET L[1]=C[1]/A[1,1]
690  FOR I=2 TO N1
700  LET K3=I-1
710  LET L[I]=C[I]/A[I,I]
720  FOR J=1 TO K3
730  LET L[I]=L[I]=(A[I,J]*L[J])/A[I,I]
740  NEXT J
750  NEXT I
760  LET M[1]=B[1]/A[1,1]
770  FOR I=2 TO N1
780  LET K3=I-1
790  LET M[I]=B[I]/A[I,I]
800  FOR J=1 TO K3
810  LET M[I]=M[I]-(A[I,J]*M[J])/A[I,I]
820  NEXT J
830  NEXT I
840  FOR J=1 TO N1
850  LET I3=J-1
860  IF I3=0 THEN 880
870  GOTO 890
880  LET I3=N-J
890  FOR K=1 TO 100
910  LET L=K
915  IF L=99 THEN 1050
920  LET S1=L*H[J]/100+S[J]
930  LET U1=L[I3]*(S[J+1]-S1)↑2*(S1-S[J])/H[J]↑2
940  LET U2=L[J]*(S1-S[J])↑2*(S[J+1]-S1)/H[J]↑2
950  LET U3=X[J]*(S[J+1]-S1)↑2*(2*(S1-S[J])+H[J])/H[J]↑3
960  LET U4=X[J+1]*(S1-S[J])↑2*(2*(S[J+1]-S1)+H[J])/H[J]↑3
970  LET U5=M[I3]*(S[J+1]-S1)↑2*(S1-S[J])/H[J]↑2
980  LET U6=M[J]*(S1-S[J])↑2*(S[J+1]-S1)/H[J]↑2
990  LET U7=Y[J]*(S[J+1]-S1)↑2*(2*(S1-S[J])+H[J])/H[J]↑3
1000  LET U8=Y[J+1]*(S1-S[J])↑2*(2*(S[J+1]-S1)+H[J])/H[J]↑3
1010  LET Z1=U1-U2+U3+U4
1020  LET Z2=U5-U6+U7+U8
1030  PRINT Z1,Z2
1040  NEXT K
1050  NEXT J
1060  END
```

```
RUN

?   12
?    2.8346,    0.0
?    1.7634,    3.0506
?    0.0029,    3.5827
?   -0.9653,    3.5931
?   -2.2837,    2.2819
?   -2.7001,   -0.7231
?   -1.8521,   -3.2040
?   -0.0030,   -3.7008
?    1.8718,   -3.2318
?    2.4786,   -2.4767
?    2.7952,   -0.7486
?    2.8346,    0.0
```

2.83458	3.31856E-02
2.83453	6.65028E-02
2.83444	9.99463E-02
2.8343	.13351
2.8341	.167188
2.83385	.200976
2.83354	.234867
2.83315	.268855
2.83269	.302935
2.83215	.337102
2.83153	.371349
2.83081	.405671
2.83	.440062
2.82908	.474516
2.82806	.509029
2.82693	.543593
2.82567	.578204
2.82429	.612856
2.82278	.647542
2.82114	.682258
2.81935	.716997
2.81742	.751755
2.81534	.786525
2.8131	.821301
2.8107	.856078
2.80813	.89085
2.80538	.925612
2.80246	.960358
2.79935	.995082
2.79605	1.02978
2.79255	1.06444
2.78886	1.09906
2.78495	1.13364
2.78084	1.16817
2.7765	1.20265
2.77194	1.23706
2.76716	1.2714
2.76214	1.30567
2.75688	1.33986
2.75137	1.37397
2.74561	1.40799
2.7396	1.44191
2.73332	1.47573
2.72678	1.50944
2.71996	1.54304
2.71287	1.57652
2.70549	1.60988
2.69782	1.64311
2.68986	1.6762
2.68159	1.70915
2.67302	1.74195
2.66414	1.7746
2.65494	1.80709
2.64542	1.83942
2.63557	1.87157
2.62539	1.90356
2.61487	1.93536
2.604	1.96697
2.59278	1.99839
2.58121	2.02961

2.56928	2.06063
2.55698	2.09144
2.5443	2.12204
2.53125	2.15241
2.51782	2.18256
2.504	2.21247
2.48978	2.24214
2.47517	2.27158
2.46014	2.30076
2.44471	2.32968
2.42886	2.35835
2.41259	2.38675
2.39589	2.41488
2.37876	2.44273
2.3612	2.47029
2.34318	2.49757
2.32472	2.52455
2.3058	2.55123
2.28643	2.5776
2.26658	2.60367
2.24627	2.62941
2.22548	2.65483
2.2042	2.67992
2.18244	2.70468
2.16019	2.7291
2.13743	2.75317
2.11417	2.77688
2.0904	2.80024
2.06612	2.82324
2.04132	2.84587
2.01598	2.86812
1.99012	2.88999
1.96372	2.91148
1.93678	2.93258
1.90928	2.95327
1.88124	2.97357
1.85263	2.99345
1.82346	3.01292
1.74584	3.061
1.728	3.07124
1.70989	3.08134
1.69151	3.09129
1.67287	3.1011
1.65398	3.11076
1.63485	3.12028
1.61549	3.12966
1.5959	3.1389
1.5761	3.148
1.5561	3.15697
1.53589	3.1658
1.5155	3.17449
1.49493	3.18305
1.47419	3.19148
1.45328	3.19979
1.43222	3.20796
1.41101	3.21601
1.38967	3.22393
1.3682	3.23173
1.34661	3.2394
1.32491	3.24696

APPENDIX VII

This appendix contains two programs written in Hewlett-Packard Assembler language (the machine language instructions for the computer). Program I is a frequency analyzer. Program II outputs data points to operate a numerical control machine.

```
ASMB,R,B,L,T
       NAM DATA
       ENT DATA
       EXT .ENTR
DATA   NOP
       JSB .ENTR
       DEF AREA
START  NOP
       CLF 0
       CLC 0
       LDA CW1
       OTA TBG1
       STC TBG1,C
       STF 0
       JMP *
TB1SB  NOP
       LDA CW2
       OTA TBG2
       STC TBG2,C
LOOPY  STC ADC
       LDA =D-280
       STA CNTRE
       LDA =D-10
       STA TEN
       LDA =D0
       STA OLD
       LDA =D-10
       STA PSCTR
       CLF ADC
       SFS ADC
       JMP *-1
       LIB ADC
       LDA =D-10
       STA TEN
       LDA OLD
       SSB
       JMP LOOP1
       STB OLD
       ISZ PSCTR
       JMP CON
       JMP PCTLP
CON    XOR 1
       BLF
       RBL,RBL
       ALF
       RAL,RAL
LP1    RBL
       RAL
       SSA
       JMP CMPR1
       ISZ TEN
       JMP LP1
       JMP CTRNO
CMPR1  SSB
       JMP CTRNO
       LDA COUNT,I
       ADA =D1
       STA COUNT,I
       LDA =D-1
       STA PSCTR
```

```
        JMP LOOP
CTRNO   LDA =D-10
        STA PSCTR
        JMP LOOP
PCTLP   XOR 1
        BLF
        RBL,RBL
        ALF
        RAL,RAL
LP2     RBL
        RAL
        SSA
        JMP CMPR2
        ISZ TEN
        JMP LP2
        JMP PUT1
CMPR2   SSB
        JMP PUT
PUT1    LDA =D-1
        STA PSCTR
        JMP LOOP
PUT     LDA =D-10
        STA PSCTR
        JMP LOOP
LOOP1   LDA =D0
        STA OLD
LOOP    JMP LOOPY
TB2SB   NOP
        CLF 0
        ISZ CNTRE
        JMP ADR
        JMP CHNG
ADR     LDA COUNT
        ADA =B1
        STA COUNT
        STF 0
        JMP TB1SB,I
CHNG    LDA =D-280
        STA ODOUT
        LDA OUTCT,I
        FMP =F2.4
        DST AREA,I
        LDA =B1
        ADA OUTCT
        LDA =B2
        STA AREA
        ISZ ODOUT
        JMP CHNG
        JMP DATA,I
TBG1    EQU 10B
TBG2    EQU 11B
ADC     EQU 12B
CW1     OCT 5
CW2     OCT 4
TEN     BSS 1
ODOUT   BSS 1
OLD     BSS 1
CNTRE   BSS 1
PSCTR   BSS 1
OUTCT   DEF SPACE
```

```
COUNT DEF SPACE
SPACE BSS 280
AREA  BSS 560
      END START
```

```
0001                    ASMB,R,B,L,T
ADC       000012
ADR    R  000130
CON    R  000045
CW1    R  000153
CW2    R  000154
LP1    R  000052
LP2    R  000101
OLD    R  000157
PUT    R  000115
TEN    R  000155
.DST   X  000003
.ENTR  X  000001
.FMP   X  000002
AREA   R  000614
CHNG   R  000135
CMPR1  R  000061
CMPR2  R  000110
CNTRE  R  000160
COUNT  R  000163
CTRNO  R  000071
DATA   R  000000
LOOP   R  000122
LOOP1  R  000120
LOOPY  R  000017
ODOUT  R  000156
OUTCT  R  000162
PCTLP  R  000074
PSCTR  R  000161
PUT1   R  000112
SPACE  R  000164
START  R  000003
TB1SB  R  000013
TB2SB  R  000123
TBG1      000010
TBG2      000011
** NO ERRORS*
```

```
DKEDATA  DC@X@DATA P.DST .ENTR.FMP <@A% @
D=TM"DDL@Z@DKTED@?@6RD=T@D=TQJZJ,
                          @
                           Z,1<M,*"Z,9,9DSD?6ZTSD@TQ,RD=4TQ,R @
                                                              ;@])?@
                                                                   6P,H
 PAGE 0002


**  NO ERRORS*
```

```
0001                        ASMB,R,B,L,T
0002  00000                       NAM DATA
0003                              ENT DATA
0004                              EXT .ENTR
0005  00000 000000  DATA    NOP
0006  00001 016001X          JSB .ENTR
0007  00002 000614R          DEF AREA
0008  00003 000000  START   NOP
0009  00004 103100          CLF 0
0010  00005 106700          CLC 0
0011  00006 062153R          LDA CW1
0012  00007 102610          OTA TBG1
0013  00010 103710          STC TBG1,C
0014  00011 102100          STF 0
0015  00012 026012R          JMP *
0016  00013 000000  TB1SB   NOP
0017  00014 062154R          LDA CW2
0018  00015 102611          OTA TBG2
0019  00016 103711          STC TBG2,C
0020  00017 102712  LOOPY   STC ADC
0021  00020 063674R          LDA =D-280
0022  00021 072160R          STA CNTRE
0023  00022 063675R          LDA =D-10
0024  00023 072155R          STA TEN
0025  00024 063676R          LDA =D0
0026  00025 072157R          STA OLD
0027  00026 063675R          LDA =D-10
0028  00027 072161R          STA PSCTR
0029  00030 103112          CLF ADC
0030  00031 102312          SFS ADC
0031  00032 026031R          JMP *-1
0032  00033 106512          LIB ADC
0033  00034 063675R          LDA =D-10
0034  00035 072155R          STA TEN
0035  00036 062157R          LDA OLD
0036  00037 006020          SSB
0037  00040 026120R          JMP LOOP1
0038  00041 076157R          STB OLD
0039  00042 036161R          ISZ PSCTR
0040  00043 026045R          JMP CON
0041  00044 026074R          JMP PCTLP
0042  00045 020001  CON     XOR 1
0043  00046 005700          BLF
0044  00047 005222          RBL,RBL
0045  00050 001700          ALF
0046  00051 001222          RAL,RAL
0047  00052 005200  LP1     RBL
0048  00053 001200          RAL
0049  00054 002020          SSA
0050  00055 026061R          JMP CMPR1
0051  00056 036155R          ISZ TEN
0052  00057 026052R          JMP LP1
0053  00060 026071R          JMP CTRNO
0054  00061 006020  CMPR1   SSB
0055  00062 026071R          JMP CTRNO
0056  00063 162163R          LDA COUNT,I
0057  00064 043677R          ADA =D1
```

```
0058  00065 172163R        STA COUNT,I
0059  00066 063700R        LDA =D-1
0060  00067 072161R        STA PSCTR
0061  00070 026122R        JMP LOOP
0062  00071 063675R CTRNO  LDA =D-10
0063  00072 072161R        STA PSCTR
0064  00073 026122R        JMP LOOP
0065  00074 020001  PCTLP  XOR 1
0066  00075 005700         BLF
0067  00076 005222         RBL,RBL
0068  00077 001700         ALF
0069  00100 001222         RAL,RAL
0070  00101 005200  LP2    RBL
0071  00102 001200         RAL
0072  00103 002020         SSA
0073  00104 026110R        JMP CMPR2
0074  00105 036155R        ISZ TEN
0075  00106 026101R        JMP LP2
0076  00107 026112R        JMP PUT1
0077  00110 006020  CMPR2  SSB
0078  00111 026115R        JMP PUT
0079  00112 063700R PUT1   LDA =D-1
0080  00113 072161R        STA PSCTR
0081  00114 026122R        JMP LOOP
0082  00115 063675R PUT    LDA =D-10
0083  00116 072161R        STA PSCTR
0084  00117 026122R        JMP LOOP
0085  00120 063676R LOOP1  LDA =D0
0086  00121 072157R        STA OLD
0087  00122 026017R LOOP   JMP LOOPY
0088  00123 000000  TB2SB  NOP
0089  00124 103100         CLF 0
0090  00125 036160R        ISZ CNTRE
0091  00126 026130R        JMP ADR
0092  00127 026135R        JMP CHNG
0093  00130 062163R ADR    LDA COUNT
0094  00131 043677R        ADA =B1
0095  00132 072163R        STA COUNT
0096  00133 102100         STF 0
0097  00134 126013R        JMP TB1SB,I
0098  00135 063674R CHNG   LDA =D-280
0099  00136 072156R        STA ODOUT
0100  00137 162162R        LDA OUTCT,I
0101  00140 016002X        FMP =F2.4
      00141 001701R
0102  00142 016003X        DST AREA,I
      00143 100614R
0103  00144 063677R        LDA =B1
0104  00145 042162R        ADA OUTCT
0105  00146 063703R        LDA =B2
0106  00147 072614R        STA AREA
0107  00150 036156R        ISZ ODOUT
0108  00151 026135R        JMP CHNG
0109  00152 126000R        JMP DATA,I
0110  00010          TBG1   EQU 10B
0111  00011          TBG2   EQU 11B
0112  00012          ADC    EQU 12B
```

```
PAGE 0005 #01


0113   00153 000005  CW1    OCT 5
0114   00154 000004  CW2    OCT 4
0115   00155 000000  TEN    BSS 1
0116   00156 000000  ODOUT  BSS 1
0117   00157 000000  OLD    BSS 1
0118   00160 000000  CNTRE  BSS 1
0119   00161 000000  PSCTR  BSS 1
0120   00162 000164R OUTCT  DEF SPACE
0121   00163 000164R COUNT  DEF SPACE
0122   00164 000000  SPACE  BSS 280
0123   00614 000000  AREA   BSS 560
       01674 177350
       01675 177766
       01676 000000
       01677 000001
       01700 177777
       01701 046314
       01702 146404
       01703 000002
0124                        END START
**  NO ERRORS*
```

```
ASMB,R,B,L,T
       NAM OUT
       ENT OUT
       EXT .ENTR
OUT    NOP
       JSB .ENTR
       DEF VALX
START  NOP
       STC DA1,C
       STC DA2,C
       STC DA3,C
       LDA =D-140
       STA COUNT
LOOP   LDA VALX,I
       OTA DA1
       SFS DA1
       JMP *-1
       LDA VALY,I
       OTA DA2
       SFS DA2
       JMP *-1
       LDA VALZ,I
       OTA DA3
       SFS DA3
       JMP *-1
       LDA =B2
       ADA VALX
       STA VALX
       LDA =B2
       ADA VALY
       STA VALY
       LDA =B2
       ADA VALZ
       STA VALZ
       ISZ COUNT
       JMP LOOP
       JMP OUT,I
VALX   BSS 280
VALY   BSS 280
VALZ   BSS 280
DA1    EQU 13B
DA2    EQU 14B
DA3    EQU 15B
COUNT  BSS 1
       END START
```

```
0001                    ASMB,R,B,L,T
DA1       000013
DA2       000014
DA3       000015
OUT     R 000000
.ENTR   X 000001
COUNT   R 001551
LOOP    R 000011
START   R 000003
VALX    R 000041
VALY    R 000471
VALZ    R 001121
**   NO ERRORS*
```

** NO ERRORS*

```
0001                        ASMB,R,B,L,T
0002  00000                       NAM OUT
0003                              ENT OUT
0004                              EXT .ENTR
0005  00000 000000  OUT     NOP
0006  00001 016001X             JSB .ENTR
0007  00002 000041R             DEF VALX
0008  00003 000000  START   NOP
0009  00004 103713            STC DA1,C
0010  00005 103714            STC DA2,C
0011  00006 103715            STC DA3,C
0012  00007 063552R            LDA =D-140
0013  00010 073551R            STA COUNT
0014  00011 162041R LOOP      LDA VALX,I
0015  00012 102613            OTA DA1
0016  00013 102313            SFS DA1
0017  00014 026013R            JMP *-1
0018  00015 162471R            LDA VALY,I
0019  00016 102614            OTA DA2
0020  00017 102314            SFS DA2
0021  00020 026017R            JMP *-1
0022  00021 163121R            LDA VALZ,I
0023  00022 102615            OTA DA3
0024  00023 102315            SFS DA3
0025  00024 026023R            JMP *-1
0026  00025 063553R            LDA =B2
0027  00026 042041R            ADA VALX
0028  00027 072041R            STA VALX
0029  00030 063553R            LDA =B2
0030  00031 042471R            ADA VALY
0031  00032 072471R            STA VALY
0032  00033 063553R            LDA =B2
0033  00034 043121R            ADA VALZ
0034  00035 073121R            STA VALZ
0035  00036 037551R            ISZ COUNT
0036  00037 026011R            JMP LOOP
0037  00040 126000R            JMP OUT,I
0038  00041 000000  VALX      BSS 280
0039  00471 000000  VALY      BSS 280
0040  01121 000000  VALZ      BSS 280
0041  00013         DA1       EQU 13B
0042  00014         DA2       EQU 14B
0043  00015         DA3       EQU 15B
0044  01551 000000  COUNT     BSS 1
      01552 177764
      01553 000002
0045                              END START
**  NO ERRORS*
```

REFERENCES

1. Ahlberg, J. H., Nilson, E. N., Walsh, J. L., "The Theory of Splines and Their Applications", Academic Press, 1970.

2. Koerner, Ilse, "The Gait of the Amputee", Journal of the Canadian Physiotherapy Association, December, 1968.

3. Leslie, W. H., "Numerical Control Programming Languages", North Holland Publishing Co.

4. Olesten, Nils, "Numerical Control", 1970.

5. Pizey, Gordon, "Three Dimensional Shape Sensing", Undergraduate Thesis, 1970.

6. P.O.R.D.U. Report No. 6, "A Pylon Prosthesis System for Shank (BK) Amputees", Manitoba Rehabilitation Hospital, 1965.

7. Schoenberg, I. J., "Approximations, with Special Emphasis on Spline Functions", Academic Press, 1969.

8. Steindler, Arthur, "Kinesiology of the Human Body", Charles C. Thomas, 1955.