EASYPLOT, A HIGH LEVEL PLOTTING SYSTEM

By

JOHN RAYMOND FORREST

Bachelor of Science

Oklahoma State University

Stillwater, Oklahoma

1968

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
MASTER OF SCIENCE
July, 1972

EASYPLOT, A HIGH LEVEL PLOTTING SYSTEM

Thesis Approved:

G. E. Hedrick
Thesis Adviser

Donald D Fisher

John P. Chandler

D Durham
Dean of the Graduate College

PREFACE

This paper is a discussion of a high level plotting program called Easyplot. In addition to a description of the computer program and a guide to its use a study of the relative merits of popular interpolation and smoothing techniques is presented. The interpolation and smoothing techniques used in Easyplot are also derived in detail.

The author wishes to express his appreciation to his major advisor, Dr. George E. Hedrick for his assistance in the prepatation of this thesis. A note of appreciation is also extended to Dr. John P. Chandler for his valuable suggestions and contributions to this thesis.

The author also wishes to acknowledge the work of John Jensen in coding subroutine SMOTH which is used in creating smoothed curves. Thanks are also extended to Mrs. Grace Provence for typing the rough draft and to Mrs. Tom Lee for preparation of the final copy.

TABLE OF CONTENTS

Chapter		Page
I.	INTRODUCTION	1
	Motivation for Easyplot	1
	The Organization of Easyplot	3
	The Applications of Easyplot	4
	A Brief Survey of Plotting Literature	6
II.	A COMPARISON OF INTERPOLATION AND SMOOTHING METHODS	9
	Methods of Interpolation	9
	Smoothing Techniques	13
III.	NUMERICAL METHODS	22
	A Smooth Curve Fitting Method	22
	Smoothing Using Cubic Spline Functions	30
IV.	PROGRAM DESCRIPTION	38
	General Organization	38
	The Interface Routines	39
	The Plotting Program	43
V.	EASYPLOT USER'S GUIDE	48
	Introduction	48
	Subroutine GRAPH	48
	Subroutine LETTER	55
	Subroutine ORIGIN	57
	Subroutine KRPLT	58
	Debugging Aids	61
VI.	SUMMARY AND SUGGESTED EXTENSIONS	63
SELECTED BIBLIOGRAPHY		65
APPENDIX A - JOB CONTROL LANGUAGE FOR EASYPLOT		67
APPENDIX B - PLOTTING SYMBOLS		70
APPENDIX C - TEST PROGRAMS		72
APPENDIX D - FLOW CHARTS		85

LIST OF FIGURES

Figure		Page
1.	A Comparison of the Use of the Calcomp Routines and Easyplot	7
2.	"Comparison of Several Methods of Smooth Curve Fitting" (3)	11
3.	"A Condition for Determining the Slope of the Tangent to the Curve" (3)	23
4.	Illustration of an Unnatural Consequence of Equation (III.5)	25
5.	Job Control Language for the IBM S/360	69
6.	Symbols Available on the Calcomp Plotter	71
7.	Test Program Number One	73
8.	Parameter Dump for Test Program Number One	74
9.	Plot Produced by Test Program Number One	75
10.	Test Program Number Two	76
11.	Parameter Dump for Test Program Number Two	77
12.	Plot Produced by Test Program Number Two	78
13.	Test Program Number Three	79
14.	Parameter Dump for Test Program Number Three	80
15.	Plot Produced by Test Program Number Three	81
16.	Test Program Number Four	82
17.	Parameter Dump for Test Program Number Four	83
18.	Plot Produced by Test Program Number Four	84
19.	Flow Chart for Subroutine GRAPH	86
20.	Flow Charts for Subroutines DP and KRPLT	87

Figure Page

21. Flow Chart for Easyplot Main Program 88

22. Flow Chart for Subroutine PLTDAT 89

23. Flow Charts for Subroutines INTERP and LGAXIS 90

24. Flow Chart for Subroutine SMOTH 91

CHAPTER I

INTRODUCTION

This paper is a presentation of a high level plotting program designed to eliminate much of the trivia involved in using a typical digital incremental plotter. The plotting system called Easyplot incorporates the latest techniques in numerical analysis for plotting interpolated and smoothed curves. A discussion of popular interpolation and smoothing techniques is presented in Chapter II, and derivations of the techniques used in Easyplot are presented in Chapter III.

Motivation for Easyplot

The generation of a graph through the use of a digital computer and a Calcomp Digital Incremental Plotter presents many problems. Frequently, the generation of a graph is only one result of a complex program. Thus, a specific plotting algorithm may be merged into a program with many other functions. This plotting algorithm may require a considerable amount of computer storage which could place undesirable restrictions upon the remainder of the program. In many cases the plotting logic may be so entwined in the logic of the program that a slight change in the specifications of a graph may cause a major revision of the program. Another problem encountered is that the data to be plotted may have to be scaled since the Calcomp plotter requires data to be in units of inches. Thus, if the data is needed for further

calculations, it must be saved in some manner. Finally, the most serious problem is duplication of effort and unnecessary effort by programmers using the plotter. When a plotting algorithm is incorporated into the logic of a program it is generally difficult to adapt that logic to another program with similar plotting requirements. Duplication of effort is particularly evident when many programmers are using the plotter. Since most graphs can be plotted with a general format except for minor changes such as the axis length, labeling, and other trivia, much unnecessary effort results when the programmer must create a graph without a generalized plotting program.

The plotting program called Easyplot is an attempt to alleviate these problems. The programmer can create an entire graph through the use of Easyplot by making one subroutine call, without having to scale his data, plot axes, etc. In addition Easyplot has the capability to plot logarithmic axes, interpolated curves, and smoothed curves. The routines which calculate functions for interpolated and smoothed curves are independent of the plotting portion of the program. Therefore, these routines can be used for applications other than curve plotting. Easyplot is designed to require a minimal amount of storage in the user's program. Thus, the amount of storage available for other programming logic is not significantly affected by the plotting routines.

Easyplot was designed to conform closely with another plotting program called Simplotter developed by D. G. Scranton (19). The functions performed by Easyplot and Simplotter are virtually the same. In addition the names of the four problem-oriented subroutines and their parameters are the same. This close conformity permits programs written for one system to be run on the other. Since a plotting

program performing the same functions as Easyplot already exists, one must ask why there is a need for Easyplot? The answer is portability. Simplotter was written specifically for use on an IBM 360 computing system. Since Calcomp plotters and the standard Calcomp supplied plotting routines may be used with other computing systems, it is desirable that a plotting program be coded in a standard language which is acceptable to most computing systems. Easyplot is coded in American National Standard FORTRAN. Thus, it can be implemented on many different computing systems with only minor changes. In addition to portability, secondary reasons for the existence of Easyplot are the methods used for interpolation and smoothing. The methods used in Easyplot are generally better than those in Simplotter. A comparison of frequently used interpolation and smoothing techniques given in Chapter II supports the previous assertion.

The Organization of Easyplot

Easyplot is organized in two parts which shall be called the interface routines and the plotting program. The interface routines are called by the user from a FORTRAN program. The parameters passed to these routines describe the specifications of the plots desired; however, the parameters are not examined by these routines and no plotting is done by them. These routines create two output files which are passed to the plotting program. The separation of the user's program and the plotting program results in two significant advantages to the user. First, the interface routines require a relatively small amount of storage (about 8000 bytes on an IBM System 360). Second, the fact that the plotting program is executed separately from the user's

program permits simultaneous debugging of the user's plotting logic and his program logic.

There are four interface routines which may be called by a user. Each of the routines will cause the plotting program to perform specific plotting functions. The plotting program inputs the files created by the interface routines and calls the standard Calcomp plotting routines in order to create the desired plots. The plotting program includes subroutines which calculate and plot interpolated and smoothed curves for a set of data points.

The Applications of Easyplot

The four subroutines called by a user are named GRAPH, LETTER, ORIGIN, and KRPLT. It takes only one call to create an entire graph complete with horizontal and vertical axes, labels, and a set of points plotted in one of several different ways. If several different sets of data are to be plotted on one graph a subroutine call must be made for each data set. The first call to GRAPH, LETTER, or KRLPT for one particular graph is called a primary call. Additional data sets may be plotted on the graph created by a primary call by issuing superposition calls to one of the three previously mentioned subroutines.

The purposes of the four subroutines are:

Subroutine GRAPH is the general purpose routine. Graphs may be created with linear, semi-log, or log-log axes. The axes are plotted horizontally and vertically with numbered tick marks every inch for linear axes. Logarithmic axes always consist of an integral number of cycles. Each cycle has nine tick marks corresponding to an integer times the power of ten associated with the cycle. The logarithms

plotted may be natural logarithms or base 10 logarithms. The user specifies a label for the graph which is plotted in the upper right corner. Scale factors for the axes and data points may be specified by the user or calculated by Easyplot. If scale factors are calculated by Easyplot they will be based upon the maximum and minimum x and y coordinates of all data sets to be plotted. Each data set may be plotted with a variety of different options. The points may be plotted individually with a plotting symbol indicated by the user, and they may be connected by line segments. A closed or open interpolated curve may be plotted through the points. A closed curve is a curve with first and last points connected such as a circle while the first and last points are not connected in an open curve. If the user desires to plot a curve which will approximate a set of points but not necessarily pass through each point, a smoothing option is available which will produce the desired result.

Subroutine LETTER permits the user to plot labels in any position on a graph. It is usually used in the superposition made; however, a new graph may be created by a primary call to LETTER.

Subroutine ORIGIN allows the user to control the position of the origin and suppress the plotting of either or both axes. In the normal operating mode the origin is positioned automatically when a primary call is made to GRAPH, LETTER, or KRPLT. However, if the user takes control of the origin through this subroutine, the position of the origin cannot be changed except by another call to ORIGIN.

Subroutine KRPLT assembles data for histograms and ideograms and plots the results. A histogram is essentially a bar graph. The abcissa is divided into intervals and a count is associated with each

interval. The count is the number of events occurring in an interval. An ideogram is an extension of the histogram. The abcissa is divided into intervals; however, each event has an associated error. Thus, the event is represented by a normal or Gaussian curve with standard deviation equal to the error associated with the event. The curve has unit area and is centered on the coordinate of the event. The sums of the contributions of each Gaussian curve at the interval boundaries are accumulated. A histogram is plotted as a bar graph, and an ideogram is plotted as an interpolated curve (see Appendix C for examples).

To illustrate the ease with which a plot can be created with Easyplot compared to using the Calcomp routines, a very simple example program is presented in Figure 1. These two programs will produce nearly identical plots with linear axes, labels and a set of data points connected with line segments.

A Brief Survey of Plotting Literature

A survey of published literature concerning computer applications reveals that there has not been a great deal of information published concerning plotting programs. There are many programming systems which incorporate graphical output as a secondary result; however, these systems are not applicable to general plotting requirements. A specific example of this is a system called PEG which was created by Lyle B. Smith (21). The PEG system is primarily used for data-fitting problems using several different interpolation and least squares fitting and smoothing techniques. The system allows the user to interact with the computer through a cathode ray tube display terminal. The user may view graphical displays of intermediate results in the determination of

Sample Plot Using Calcomp Routines

```
C  DIMENSION X AND Y ARRAYS AND ARRAYS FOR LABELS
C
      DIMENSION X (27),Y(27),XL(5),YL(5),GL(5),GL(5)
C  READ IN THE LABELS
C
      READ(5,11) XL,YL,GL,DL
   11 FORMAT(20A4)
C  INITIALIZE THE PLOT ROUTINES
      CALL PLOTS
C  SET THE ORIGIN TO THE RIGHT HAND OF THE DRUM
     CALL PLOTC(1., -11.0, -3)
C  RELOCATE THE ORIGIN ONE-HALF INCH FROM THE EDGE
      CALL PLOTC(1., 0.5, -3)
C  GENERATE DATA POINTS TO BE PLOTTED
      XX = 0.
      DO 10 I = 1,25
      X(I) = XX
      Y(I) = SIN(XX**2) * EXP(-XX)
   10 XX = XX + .25
C  SCALE THE VALUES AND DRAW THE AXES
      CALL SCALE (X, 10., 25, 1)
      CALL AXIS (0.0, 0.0, XL, -8, 10.0, 0.0, X(26), X(27))
      CALL SCALE (Y, 6.0, 25, 1)
      CALL AXIS (0.0, 0.0, YL, 19, 6.0, 90.0, Y(26), Y(27))
C  PLOT THE POINTS CONNECTED BY LINE SEGMENTS
      CALL LINE (X, Y, 25, 1, 1, 3)
C  PLOT THE GRAPH LABELS
      CALL SYMBOL (6., 5., 0.14, GL, 0.0, 20)
      CALL SYMBOL (6., 4.75, 0.14, DL, 0.0, 20)
      STOP
      END
```

Sample Program Using Easyplot

```
      DIMENSION X(25), Y(25), XL(5), YL(5), GL(5), DL(5)
      READ(5,11) XL, YL, GL, DL
   11 FORMAT (20A4)
C  GENERATE DATA POINTS TO BE PLOTTED
      XX = 0.
      DO 10 I = 1,25
      X(I) = XX
      Y(I) = SIN(XX**2) * EXP(-XX)
   10 XX = XX + .25
      CALL GRAPH (25, X, Y, 3, 3, 10. 6., 0,0,0,0, XL, YL, GL, DL, 0)
      STOP
      END
```

Figure 1. A Comparison of the Use of the Calcomp Routines and Easyplot

a fitted curve. This allows the user to use his own judgement in determining what further refinements or changes the system should make in order to arrive at a suitable approximating curve. Smith (22) discusses several other similar systems which incorporate graphical output in the solution of numerical analysis problems. Some of these routines do produce permanent graphical output; however, they do not have the flexibility that a general plotting program should have. This is to be expected since these systems are designed to find numerical solutions to problems with graphical output as a side effect.

Although there is little published literature dealing with routines whose only function is the production of graphical output there is a great deal of information concerning numerical methods which may be used in such routines. In particular, there have been many articles written concerning interpolation and smoothing methods for approximating data. A survey of these methods is given in the next chapter.

CHAPTER II

A COMPARISON OF INTERPOLATION AND SMOOTHING METHODS

Methods of Interpolation

The purpose of the interpolation method used in Easyplot is to connect a set of points with a smooth, continuous, and "natural-looking" curve. A mathematical method of interpolation which produces a continuous curve with continuous first derivatives and no unnatural "wiggles" is desired. Since several methods satisfy these conditions, the criterion of natural appearance was the final determining factor in choosing the method used in Easyplot.

The manual drawing of a curve by a well-trained scientist or engineer is probably the best standard of comparison for the natural appearance of a curve. Since a manually drawn curve is sketched in portions which are based on a relatively small number of points, making no assumptions about any functional relationship involving the entire set of data, it seems that such an approach would be necessary for a numerical method also. A method which assumes a functional form for the entire set of data frequently results in a curve which is quite different from a manually drawn curve. The difficulty usually manifests itself in "unnatural wiggles" (3).

The most common interpolation method is the use of polynomials. Either a single polynomial of order n-1, where n is the number of data

points, is fit to the entire set of data, or a piecewise function composed of lower order polynomials is fit to successive subsets of the data. Fitting one polynomial to the entire set of data is not practical since this would not allow multi-valued curves, and high order polynomials tend to oscillate between the points to which they are fit (3, 13). An example of this behavior is shown in Figure 2.

The use of low order polynomials for interpolation functions is very common and in most cases produces satisfactory results. If a multi-valued curve is permissible, second order polynomials are most often used. Each subset of three points is made single-valued by rotating the coordinate system so that the x coordinates are monotone increasing. In addition to the increased complexity introduced by a rotation of the coordinates, the chief objection is the fact that the resulting curve is not necessarily smooth.

A second method considered is a Fourier series approximation which exemplifies the use of orthogonal functions. The curve is approximated by a function of the form

$$f(x) = a_0 + \sum_{i=1}^{n} (a_k \cos kx + b_k \sin kx), \qquad \text{(II.1)}$$

where 2n is the number of data points. The coefficients a_k and b_k are then uniquely determined by 2n simultaneous equations. This method cannot be applied to multi-valued functions except in a piecewise manner, and becomes cumbersome for a large number of points. The method exhibits oscillations in many cases (Figure 2) and is best suited to data having a periodic relation.

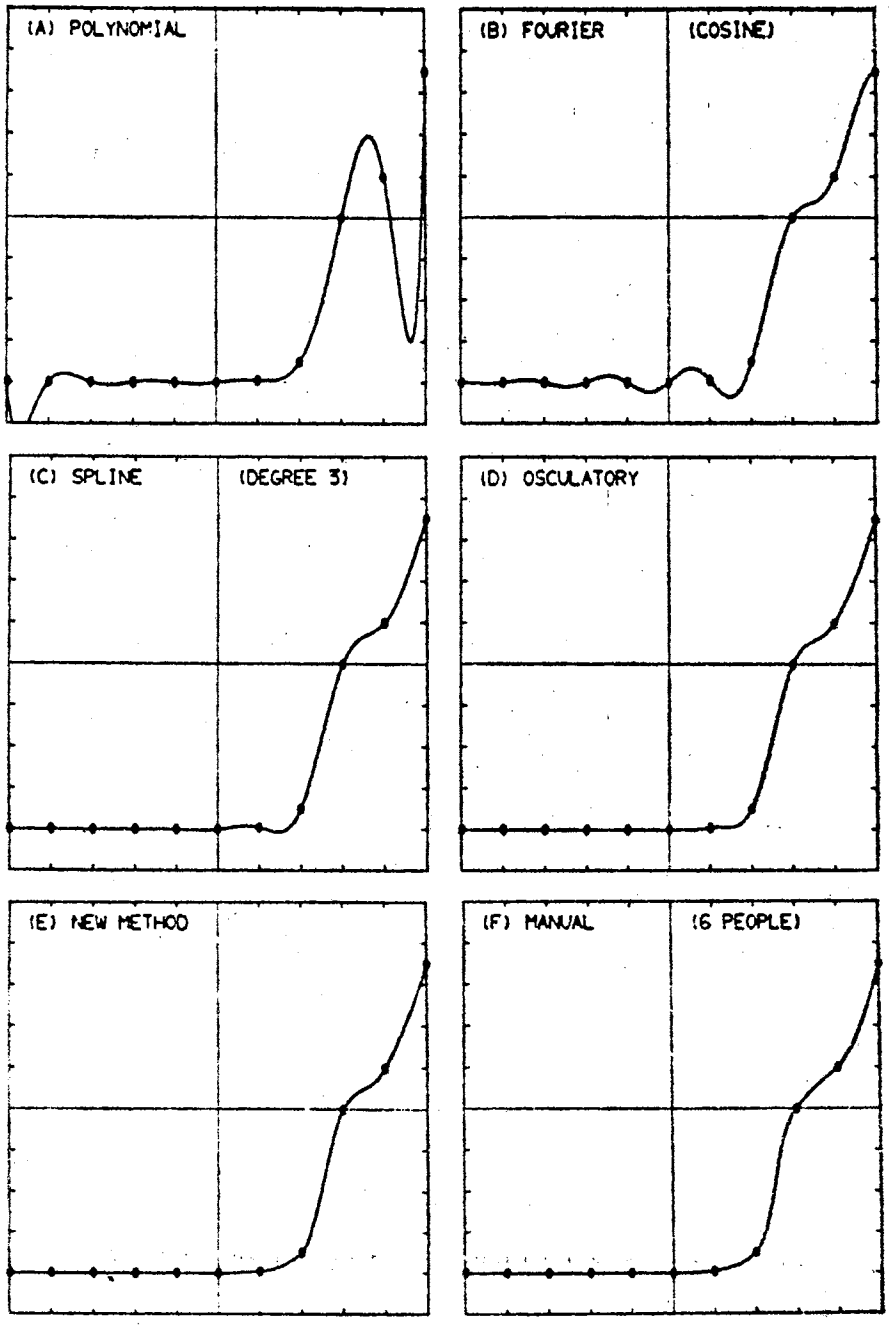


Figure 2. "Comparison of Several Methods of Smooth Curve Fitting." (3)

A third method is spline interpolation. A spline function of degree m is a piecewise function composed of polynomials. Each polynomial is of order m or less and is applicable to successive intervals of the given data points. The polynomials are determined so that the function is continuous and has continuous derivatives of order one to m-1. The spline function of degree three is most commonly used and has the property of least mean-squared curvature. This property is expressed mathematically as

$$\int_{Xo}^{Xn} f''(x)^2 dx \leq \int_{Xo}^{Xn} g''(x)^2 dx, \qquad \text{(II.2)}$$

where f(x) is a cubic spline function and g(x) is any other twice continuously differentiable function (6). The spline function is smooth and continuous; however, it was not used in Easyplot because in some cases unnatural waves result (Figure 2), and a spline function is single-valued. The unnatural waves result because the determination of the coefficients of the polynomials depends upon the entire set of data rather than a few points in each region.

A fourth method is called osculatory interpolation (1). This is a local method in which a cubic polynomial is used to interpolate between each pair of points. The polynomials are uniquely determined by first calculating an approximation to the slope of the interpolation curve at each point; thus, the smoothness of the curve is assured. The slope of the curve at a point is the slope of the quadratic polynomial determined by the point and its two neighboring points. This method is very similar to Akima's method which is used in Easyplot, and is explained in detail in Chapter III. The difference is in the method of determining the slope of the curve at each point. Both methods produce smooth

continuous curves, and each portion of the curve is determined only by its neighboring points which is generally necessary if a natural curve is desired. The osculatory method can also be applied to multi-valued curves by rotating each set of three points so that they are single-valued with respect to the x coordinate. The slope determination depends upon five points in Akima's method and only three in the osculatory method.

The choice between these two methods is based upon a comparison between curves produced by the methods and a manually drawn curve for several sets of data. An example is given in Figure 2. Akima's method produced curves which were more nearly like the manually drawn curves; therefore, his method was chosen for use in Easyplot.

Smoothing Techniques

Smoothing differs from interpolation in that the approximating curve does not necessarily pass through the given data points. Smoothing is particularly desirable when dealing with experimental data which is known to contain errors, or possibly when the "true" form of the curve underlying the data is unknown. Curve fitting using smoothing techniques includes three categories: graphing, model testing, and interpolation. The application of smoothing techniques to graphing is of prime importance since Easyplot is a plotting routine. The technique to be used in Easyplot must have several properties: 1) it must work well for any set of data, when no prior knowledge of an underlying functional relation is available; 2) the method should allow for varying amounts of smoothing. When no knowledge of the standard deviation of each point is available, the extent to which the entire curve is

smoothed should be changeable; 3) the method should allow for individual weighting of the data points; that is, if an estimate of the standard deviation of each point is known, it must be used by the method to control how far from each point the curve is allowed to deviate; 4) the method must be stable and accurate for a large number of points.

The method chosen for use in Easyplot incorporates all of these properties. It was introduced by Christian H. Reinsch (16) and involves the use of spline functions. A spline function of order m is a piecewise function consisting of polynomials of order m. The polynomials are determined such that the function and its derivatives of order one through m-1 are continuous throughout their ranges. The points at which one polynomial ends and another begins are called joints, nodes, or most commonly, knots.

A brief formulation of Reinsch's (16) method follows. Let the set of points (x_i, y_i), i=0,...,n be given and let the points be ordered such that

$$x_0 < x_1 < \cdots < x_n.$$

The smoothing function f(x) is chosen from the class of all twice continuously differentiable functions g(x) such that

$$\int_{Xo}^{Xn} f''(x)^2 dx \leq \int_{Xo}^{Xn} g''(x)^2 dx, \qquad \text{(II.3)}$$

subject to the constraint

$$\sum_{i=0}^{n} ((g(x_i)-y_i)/\delta y_i)^2 \leq S, \qquad \text{(II.4)}$$

where δy_i is an estimate of the standard deviation of y_i, and S is a constant which may be used to implicitly rescale all of the δy_i's.

The function f(x) turns out to be a cubic spline function. The derivation of an algorithm for determining f(x) is given in Chapter III.

The parameter S allows the amount of smoothing of the entire curve to be varied while the δy_i's determine how far from each point the curve is allowed to deviate. According to Reinsch (16) the method is particularly well suited to smoothing data for which no a priori knowledge of a functional relationship is available. The x coordinates of the nodes of the cubic spline are the x_i's which means that for a large number of points a large system of simultaneous equations must be solved. Since the matrices involved are either diagonal or tri-diagonal, when they are combined as shown in Chapter III, a positive definite symmetric band matrix results. This leads to a straight forward algorithm for determining the coefficients of f(x). Thus, the method does not use an excessive amount of storage, and f(x) is easily determined even for a large number of points.

It is appropriate to discuss some other smoothing techniques. One of the other techniques may produce a better approximation to a particular set of data than Reinsch's (16) spline method; however, in most of these cases the underlying functional relationship governing the data is known. In such cases it is best to try to fit some form of that function to the data using a least squares or some other criterion.

Given a set of irregularly spaced points, (x_i, y_i), i=1,...,N, where x is the independent variable and y_i is an observed value associated with x_i. The problem is to find a reasonably good function F(x) which will approximate the given points. If the errors in the data are independent and follow a normal distribution, then R. A. Fisher's "Principle of Maximum Likelihood" (14) leads to the criterion that

$$\sum_{i=1}^{N} ((F(x_i,\bar{a}) - y_i)/ \delta y_i)^2 \qquad \text{(II.5)}$$

should be minimized by choosing $\bar{a}$ appropriately. The $\bar{a}$ are the coefficients of the smoothing function F and δy_i is the standard deviation of y_i (14). If the errors are not known to follow a Gaussian distribution then the three most common criteria for determining the best fit are:

(i) The Gershgorin norm - $$\|E\|_1 = \sum_{i+1}^{N} |e_i|; \qquad \text{(II.6)}$$

(ii) The Euclidean or least squares norm

$$\|E\|_2 = \sum_{i-1}^{N} (e_i^2)^{\frac{1}{2}}; \qquad \text{(II.7)}$$

(iii) The Chebyshev or minimax norn

$$\|E\|_{oo} = \max|e_i|, \quad i=1,\ldots,N, \qquad \text{(II.8)}$$

where $e_i = F(x_i) - y_i$. The best fit is obtained when the norm is minimized. The Gershgorin norm is not generally used in this application and will not be discussed here. The least squares and minimax norms are both popular; however, since the least squares criterion is easier to apply it will be used in each of the methods to be discussed.

The first and probably most common method is to express the approximating function F(x) in terms of polynomials $f_k(x)$:

$$F(x) = b_o f_o(x) + b_1 f_1(x) + \ldots = b_n f_n(x), \qquad \text{(II.9)}$$

where $f_k(x)$ is a polynomial of degree k, and n=N-1, where N is the number of points given. The most important aspect of this method is the choice of the polynomials $f_k(x)$. The most common but probably least desirable choice is the use of the monomials $f_k(x) = x^k$. This

is a poor choice due to the ill-conditioned normal equations which result from the least squares criterion. In most cases the method will not give adequate accuracy for $n > 3$ (4).

The computational difficulties can be avoided by using orthogonal polynomials. A set of orthogonal polynomials satisfy

$$\sum_{i=1}^{n} p_j(x_i)p_1(x_i) = 0 \text{ if } j \neq 1 \text{ and } \neq 0 \text{ of } j=1, \qquad \text{(II.10)}$$

where x_i, $i=1,\ldots,n$ is a given set of points over which the polynomials $p_j(x_i)$ are orthogonal. The polynomials can be generated with a three-term recurrence relation whose constants are determined so that (II.10) is satisfied for a given set of x_i's (21). Forsythe (1957) has shown that a simple recurrence relation can be used to calculate the appropriate constants. Using orthogonal polynomials results in a diagonal coefficient matrix when the least squares criterion is used to find the coefficients of (II.9); thus, the coefficients are easily found and n, the degree of the polynomial fit, can be increased by solving a single equation. Other methods involving orthogonal polynomials are discussed by Berztiss (4).

The use of polynomials solves the problem of computational difficulties; however, Berztiss (4) states that many practical sets of data cannot be adequately approximated by polynomials of a reasonably low degree.

The classical method of smoothing involves least square polynomial fitting to replace each given point. A low order polynomial is fitted to an odd number of points, and the y coordinate of the middle point is replaced by the polynomial value at that point. Each point except points near the ends of the data is replaced in this manner. Points

near the ends are replaced in the same manner except that the point replaced may not be the middle point, or the number of points and the degree of the polynomial are reduced so that the middle point may always be used. This procedure may be repeated as many times as desired in order to increase the amount of smoothing.

If the ordinary method of least squares fitting of a polynomial to a set of points is used, the number of calculations necessary would make its use impractical. However, if the x coordinates are equally spaced, a simple formula can be developed for calculating the new y coordinates (13). A third degree polynomial is most commonly used. It is usually fitted to either five or seven points, however, any odd number of points can be used. This method of smoothing is used in Simplotter where cubic polynomials are fitted to 13 consecutive points whenever possible (20). The characteristics of this method are such that the amount of smoothing increases with the number of points used in the smoothing formula and decreases as the order of the fitted polynomial increases. The most serious drawback to this method is that the points must be equally spaced; however, this problem can be solved by appropriately weighting the unequally spaced points.

After the smoothing technique has been applied as many times as desired, a curve can be plotted by interpolating between the new points. Before the introduction of spline functions this method was almost exclusively used when smoothing was desired.

Another method which is particularly applicable to smoothing data which appears to be periodic in nature, is the truncated Fourier series. Given (x_i, y_i), $i=1,\ldots,n$, y_i is approximated by

$$f(x) = a_o + \sum_{j=1}^{m} (a_j \cos jx + b_j \sin jx), \qquad \text{(II.11)}$$

where the a_j's and b_j's are determined using the least squares criterion. An efficient algorithm for the calculation of these constants using equally spaced points is given by Goertzel (10). Another algorithm for unequally spaced points is given by Bjorck and Golub (5). Fourier approximations generally do not produce results comparable to other methods, when the data does not exhibit periodic behavior.

The last method to be discussed is another variation of the use of spline functions. In this method the nodes of the function do not correspond to the data points. For this application it is convenient to define the spline function in a different form. An elementary spline function is defined as

$$(x-\bar{x}_j)_+^m = \begin{cases} 0 \text{ if } x \leq \bar{x}_j \\ (x-\bar{x}_j)^m \text{ if } x > \bar{x}_j, \end{cases} \qquad \text{(II.12)}$$

where m is the degree of the spline function $S_m(x)$ and $x_j, j=1,\ldots J$ are the joints or nodes of the spline. Such a spline function is represented as

$$S_m(x) = a_o + a_1 s + \ldots + a_m x^m + \sum_{i=1}^{J} c_i (x-\bar{x}_i)_+^m. \qquad \text{(II.13)}$$

Given a set of points (x_i, y_i), i=1, ...,N, the function $S_m(x)$ is determined using the least squares criterion. That is the quantity

$$D = \sum_{i=1}^{N} (y_i - S_m(x_i))^2 \qquad \text{(II.14)}$$

is minimized. D is minimized by determining , $\partial D/\partial a_i, \partial D/\partial c_i$, i=1,...m, and equating them to zero. This results in a system of m+1+J

simultaneous equations. When m+1+J becomes larger than seven, the matrix of coefficients is likely to become ill-conditioned.

A method of obtaining a more accurate solution to the least squares problem involves the use of Householder transformations and is discussed by Golub (11). DeBoor and Rice (7, 8) discuss the use of an orthonormal basis to represent cubic splines. Their choice of an orthonormal basis for the space of all cubic splines leads to a more accurate solution of the system of equations.

The greatest difficulty in this method is determining the best choice for the nodes of the spline function. Since the plotting routine Easyplot is designed to eliminate as much work as possible for the user, it is not desirable to have the user choose the nodes. This means that the routine must be able to determine the best set of nodes for a particular set of data. A direct search method of doing this has been implemented by Lyle B. Smith (21) in his data-fitting system; however, this is an unnecessary complication, since Reinsch's (16) method does not require that an optimum set of nodes be found. Implementation of a direct search technique would also complicate the use of the plotting routine.

Each of the methods discussed allows for individual weighting of the data points by altering the least squares criterion to

$$\sum_{i=1}^{N} w_i (y_i - S_m(x_i))^2, \qquad \text{(II.15)}$$

where w_i is the weighting factor for y_i. The polynomial and Fourier series methods are well suited only for sets of data with specific characteristics, while smoothing with spline functions works well for general data (16, 21). Reinsch's (16) method is particularly well

suited for graphical purposes. One of the chief advantages of spline smoothing over the classical replacement method is computational efficiency. Increasing the amount of smoothing desired does not change the calculation time for Reinsch's spline technique, while each application of the classical technique significantly increases computation time. In addition the spline technique does not require interpolation after the points have been smoothed.

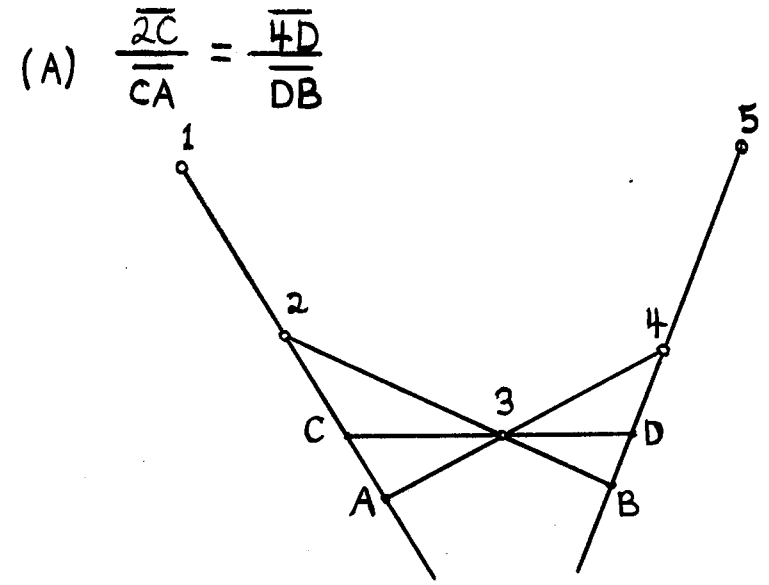
CHAPTER III

NUMERICAL METHODS

A Smooth Curve Fitting Method

The interpolation procedure used in Easyplot was invented by Hiroshi Akima (3). The method is based on the assumption that no a priori knowledge of a functional form relating the data is available. The method developed by Akima is an attempt to mathematically approximate a manually drawn interpolation curve. Since a scientist or engineer drawing an interpolation curve would base each portion on nearby points, the method developed here will use this approach. This is done by determining the slope of the curve at each point based only on the positions of two neighboring points on each side of the point in question. The key to Akima's method is the determination of this slope.

Consider five points, 1, 2, 3, 4, and 5, as in Figure 3. The slope of the curve at point 3 is represented by the line segment $\overline{CD}$. For a smooth curve Akima states that it is reasonable to assume that the slope $\overline{CD}$ should approach that of $\overline{23}$ as the slope of $\overline{12}$ approaches $\overline{23}$, and similarly the slope of $\overline{CD}$ should approach that of $\overline{34}$ as the slope of $\overline{45}$ approaches $\overline{34}$. It is also desirable that if the angle formed by $\overline{12}$ and $\overline{23}$ is equal to that formed by $\overline{34}$ and $\overline{45}$, then the angles formed by pairs ($\overline{23}$, $\overline{3C}$) and ($\overline{34}$, $\overline{3D}$) must be equal. These



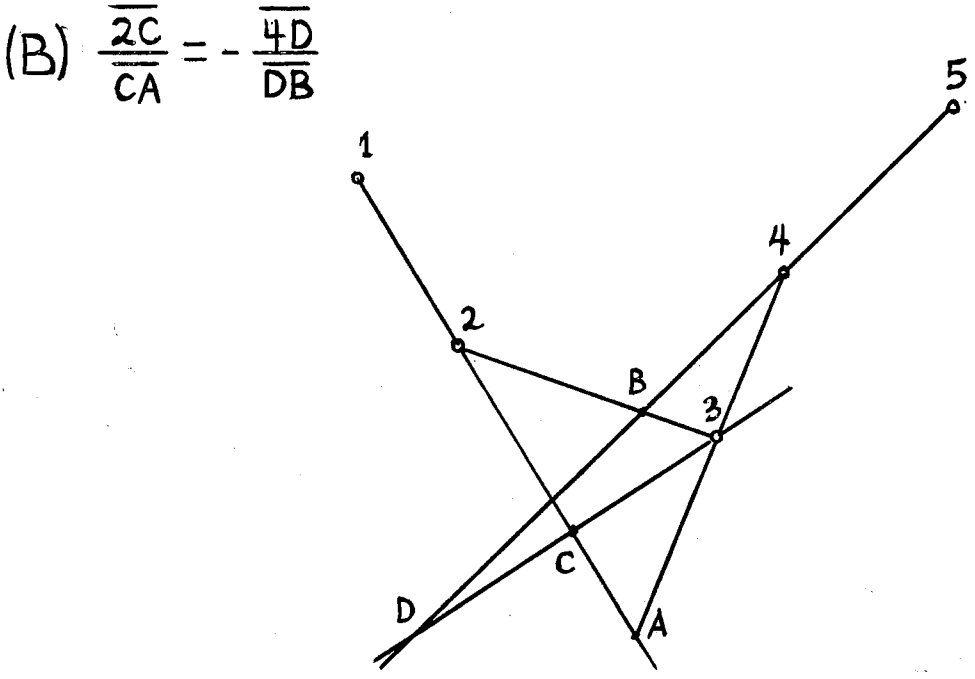


Figure 3. "A Condition for Determining the Slope of the Tangent to the Curve." (3)

conditions are not sufficient to determine uniquely the slope of $\overline{CD}$. The following additional intuitive condition is assumed which incorporates the previous conditions and uniquely determines the slope of $\overline{CD}$.

$$|\overline{2C}/\overline{CA}| = |\overline{4D}/\overline{DB}| \qquad \text{(III.1)}$$

An analytic expression for the slope of $\overline{CD}$, which shall be called t, can be obtained by expressing condition (III.1) and the slopes of all the line segments previously mentioned in terms of the coordinates of the points. The following quantities are defined in order to simplify the expression for t.

$$a_i = x_{i+1} - x_i, \ (i=1, 2, 3, 4), \qquad \text{(III.2)}$$

$$b_i = y_{i+1} - y_1, \ (i=1, 2, 3, 4), \qquad \text{(III.3)}$$

$$S_{ij} = a_i b_j - a_j b_i. \qquad \text{(III.4)}$$

The algebraic manipulations necessary to derive the following expression for t can be found in Akima (3). The result is

$$t = (w_2 b_2 + w_3 b_3)/(w_3 a_2 + w_3 a_3), \qquad \text{(III.5)}$$

where

$$w_2 = |S_{13} S_{34}|^{\frac{1}{2}}, \qquad \text{(III.6)}$$

$$w_3 = |S_{12} S_{24}|^{\frac{1}{2}}. \qquad \text{(III.7)}$$

Let the slopes of the four line segments $\overline{12}$, $\overline{23}$, $\overline{34}$, and $\overline{45}$ be m_1, m_2, m_3, and m_4 respectively. Equation (III.5) exhibits the desired results. Namely, that if $m_1 = m_2$, $m_3 \neq m_1$, and $m_4 \neq m_3$, then $t=m_1=m_2$, and if $m_3=m_4$, $m_1 \neq m_2$, and $m_4 \neq m_2$, then $t=m_3=m_4$. In certain situations equation (III.5) is undefined, requiring that t be determined in a different manner. If $m_1=m_2=m_3 \neq m_4$ or $m_1 \neq m_2=m_3=m_4$ t is logically assumed to be $t=m_2=m_3$. If $m_1=m_2 \neq m_3=m_4$, the slope is determined by $t = (m_2+m_3)/2$.

Despite its desirable properties, equation (III.5) has a serious drawback which results from condition (III.1). If $m_2=m_4$, $m_3\neq m_1$, and $m_4\neq m_3$, then $t=m_2$, and similarly if $m_1=m_3$, $m_2\neq m_1$, and $m_4\neq m_2$, then $t=m_1$. Inspection of the geometric interpolation of this result (Figure 4) shows that it is not desirable for a smooth natural curve.

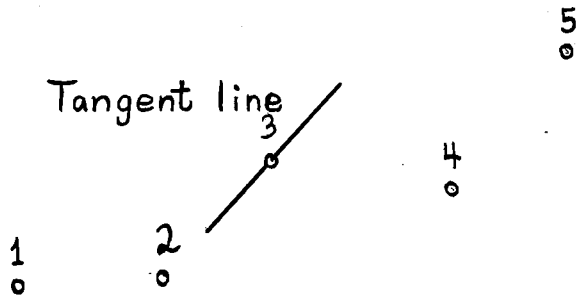


Figure 4. Illustration of an Unnatural Consequence of Equation (III.5). If $m_2=m_4$, $m_3\neq m_1$, and $m_4\neq m_3$, then $t=m_2=m_4$.

This result occurs because $S_{13}=0$ when $m_1=m_3$ and $S_{24}=0$ when $m_2=m_4$; therefore, S_{13} and S_{24} are eliminated from (III.6) and (III.7) by redefining w_2 and w_3 to be

$$w_2 = |S_{34}| , \tag{III.8}$$

$$w_3 = |S_{12}| . \tag{III.9}$$

These definitions are arrived at intuitively taking into account the observation previously stated, and the fact that the method must be invariant under a rotational transformation of the coordinate system.

Thus far the method has been based solely on geometric considerations, namely the slopes m_1, m_2, m_3, and m_4. The invariance under a rotation is maintained, since the quantity $|S_{ij}|$ is invariant and equal to twice the area of the triangle bounded by the vectors (a_i, b_j) and (a_j, b_i). Equation (III.5) can be rewritten as

$$t = (|m_4-m_3|m_2 + |m_2-m_1|m_3)/(|m_4-m_3| + |m_2-m_1|), \qquad \text{(III.10)}$$

showing that t depends on the slopes of the four line segments, which means that t is invariant under a linear scale transformation and a rotational transformation. When equations (III.8) and (III.9) are used in (III.5), t can no longer be represented as a function of only m_1, m_2, m_3, and m_4. This means that the method is no longer invariant under a linear scale transformation of the coordinate system.

The method of determining the slope of the interpolation curve at each point has been uniquely determined, and it is necessary to express the curve between points in some mathematical form. A polynomial is chosen since,

> polynomials are simple in form, can be calculated by elementary operations, are free from singular points, are unrestricted as to range of values, may be differentiated or integrated without difficulty, and the coefficients to be determined enter linearly (15).

Since four conditions, two points and the slope at each point, exist, a third degree polynomial is uniquely determined between each pair of points. The functional form of the interpolation curve must be applicable to multi-valued sets of data. This means that the standard polynomial form of y as a function of x and the expression for the slope dy/dx are inadequate, since polynomials are single-valued. The slope t is equal to tan θ where θ is the angle between the tangent line

and the x-axis. The form dy/dx can be eliminated by expressing the slope in terms of cos θ and sin θ. From equation (III.5) it follows that

$$t = \tan \theta = b_0/a_0, \tag{III.11}$$

where

$$a_0 = w_2 a_2 + w_3 a_3, \tag{III.12}$$

$$b_0 = w_2 b_2 + w_3 b_3, \tag{III.13}$$

and from (III.8) and (III.9)

$$w_2 = |s_{34}|, \tag{III.14}$$

$$w_3 = |s_{12}|. \tag{III.15}$$

From elementary trigonometry it follows that

$$\cos \theta = a_0 (a_0^2 + b_0^2)^{-\frac{1}{2}} \tag{III.16}$$

$$\sin \theta = b_0 (a_0^2 + b_0^2)^{-\frac{1}{2}} \tag{III.17}$$

The fact that polynomials are single-valued can be circumvented by introducing a parameter z which varies from 0 to 1 as the curve is traversed from (x_1, y_1) to (x_2, y_2). Thus x and y can be represented by

$$x = p_1 + p_2 z + p_3 z^2 + p_4 z^3, \tag{III.18}$$

$$y = q_1 + q_2 z + q_3 z^2 + q_4 z^3. \tag{III.19}$$

Four conditions exist for each point; they are:

$x=x_1$, $y=y_1$, $dy/dz=r\cos \theta_1$, and $dy/dz=r\sin \theta_1$ at $z=0$,

$x=x_2$, $y=y_2$, $dy/dz=r\cos \theta_2$, and $dy/dz=r\sin \theta_2$ at $z=1$,

where $r = ((x_2-x_1)^2 + (y_2-y_1)^2)^{\frac{1}{2}}$.

The p and q constants are determined by solving two sets of four simultaneous equations. The results are

$$p_1 = x_1, \tag{III.20}$$

$$p_2 = r \cos \theta_1, \tag{III.21}$$

$$p_3 = 3(x_2-x_1) - r(\cos \theta_2 + 2 \cos \theta_1), \tag{III.22}$$

$$p_4 = -2(x_2-x_1) + r(\cos \theta_2 + \cos \theta_1), \tag{III.23}$$

$$q_1 = y_1, \tag{III.24}$$

$$q_2 = r \sin \theta_1, \tag{III.25}$$

$$q_3 = 3(y_2-y_1) - r(\sin \theta_2 + 2 \sin \theta_1), \tag{III.26}$$

$$q_4 = -2(y_2-y_1) + r(\sin \theta_2 + \sin \theta_1). \tag{III.27}$$

If the interpolation curve is closed, that is the first and last points are connected, the formulation is complete. For an open curve, that is the first and last points are not connected, it is necessary to estimate two additional points at each end of the data in order to determine the slopes at the end points. For the sake of simplicity it is assumed that the estimated points lie on a curve expressed by

$$x = g_o + g_1 z + g_2 z^2, \tag{III.28}$$

$$y = h_o + h_1 z + h_2 z^2, \tag{III.29}$$

where z is a parameter introduced to preserve the multi-valued property of the curve. Assuming that

$x=x_i$ and $y=y_i$ at $z=i$ $(i=1, 2, 3, 4, 5)$.

Substituting the coordinates and corresponding z's into (III.28) and (III.29) results in ten simultaneous equations from which the g's and h's can be eliminated. The result is

$$(x_5-x_4) - (x_4-x_3) = (x_4-x_3) - (x_3-x_2) = (x_3-x_2) - (x_2-x_1), \quad \text{(III.30)}$$

$$(y_5-y_4) - (y_4-y_3) = (y_4-y_3) - (y_3-y_2) = (y_3-y_2) - (y_2-y_1), \quad \text{(III.31)}$$

where points 4 and 5 are the estimated points.

The algorithm for mathematically approximating a manually drawn interpolation curve is now complete. The method is based on local determination of the slope of the curve at each point and approximating the curve between each pair of points with a pair of cubic polynomials. The method is invariant under a rotation of the coordinate system; however, it is not invariant under a linear scale transformation. Therefore, if data is to be scaled it should be done before the slopes are calculated. The primary application of the method is plotting multi-valued curves.

Given the points (x_i, y_i), i=1, 2, 3, 4, 5, and 6, the curve between points 3 and 4 is computed by the algorithm:

1) Compute

$$a_i = x_{i+1}-x_i \quad (i=1, 2, 3, 4, 5), \quad \text{(III.32)}$$

$$b_i = y_{i+1}-y_i \quad (i=1, 2, 3, 4, 5). \quad \text{(III.33)}$$

2) Compute w_2 and w_3 from (III.14) and (III.15) for points 3 and 4 using (III.32) and (III.33).

3) Compute a_0 and b_0 from (III.12) and (III.13) for points 3 and 4 using the values computed in steps 1 and 2.

4) Compute the sin θ and cos θ for points 3 and 4 using (III.16) and (III.17).

5) Compute the coefficients of the interpolation equations (III.18) and (III.19) using the values computed in step 4 and equations (III.20) thru (III.27).

Smoothing Using Cubic Spline Functions

The cubic spline smoothing method derived here was introduced by Christian H. Reinsch (16). This is well suited for curve plotting and should be used when no a priori knowledge of a functional relationship involving the data is known (16).

A spline function of order m is a piecewise function composed of polynomials of order m. A point at which one polynomial ends and the next begins is called a joint or node. The coefficients of the polynomials are determined so that the function and its first m-1 derivatives are continuous. The special case of the cubic spline function is of interest, since it is said to have the property of "least mean-square curvature", that is, of all twice differentiable functions g(x) passing through a given set of points, the cubic spline function minimizes the integral

$$\int g''(x)^2 dx. \tag{III.34}$$

Given a set of points (x_i, y_i), i=0, ...,n such that

$$x_o < x_1 < \ldots < x_n,$$

a smoothing function f(x) is to be constructed which will minimize the integral

$$\int_{x_o}^{x_n} g''(x)^2 dx \tag{III.35}$$

among all twice continuously differentiable functions g(x) subject to the constraint

$$\sum_{i=0}^{n} ((g(x_i)-y_i)/\delta y_i)^2 \leq S, \tag{III.36}$$

where δy_i, i=0, ..., n and S are given constants. The δy_i's are weights for the individual points and should be an estimate of the standard deviation. S is a constant which is used to implicitly rescale the δy_i's. A change in the value of S results in a change in the amount of smoothing of the entire curve while each δy_i controls the distance that the curve is allowed to deviate from its associated y_i. If S is chosen to be zero, f(x) will be a cubic spline interpolating function. According to Reinsch (16), the best results are obtained if S is within the interval defined by

$$N - (2N)^{\frac{1}{2}} \leq S \leq N + (2N)^{\frac{1}{2}}, \quad N = n+1. \tag{III.37}$$

The derivation of the algorithm for constructing the cubic spline function f(x) begins by combining (III.35) and (III.36) giving the functional

$$\int_{x_o}^{x_n} g''(x)^2 dx + p\left(\sum_{i=0}^{n} ((g(x_i)-y_i)/\delta y_i)^2 + z^2 - S\right) \tag{III.38}$$

where z is a slack variable introduced to change (III.35) into an equality constraint, and p is a Lagrange multiplier used to impose the constraint upon (III.34).

Expression (III.38) is minimized if the Euler-Lagrange equation

$$\partial F/\partial g - d/dx(\partial F/\partial g') + d^2/dx^2(\partial F/\partial g'') = 0 \tag{III.39}$$

where $F = g''(x)^2$, is satisfied. Since the solution f(x) is a piecewise function, it must satisfy the Weierstrass-Erdmann corner conditions (17). The application of (III.39) to (III.38) results in

$$f''''(x) = 0, \quad x_i < x < x_{i+1}, \quad i=0, \ldots, n-1. \tag{III.40}$$

The application of the corner conditions of the knots of f(x) results in

$$f^{(k)}(x_i)_- - f^{(k)}(x_i)_+ = \begin{cases} 0 & \text{if } k=0,\ 1,\ i=1,\ \ldots,\ n-1 \\ 0 & \text{if } k-2,\ i=0,\ \ldots,\ n \\ 2p(f(x_i)-y_i)/\delta y_i^2 & \text{if } k=3,\ i=0,\ldots,\ n, \end{cases} \tag{III.41}$$

where $f^{(k)}(x_i)_{\pm} = \lim_{h \to 0} f^{(k)}(x_i \pm h)$.

If (III.40) is integrated four times the result is a cubic polynomial which means that f(x) is a piecewise function composed of cubic polynomials. (III.41) implies that the function and its first and second derivatives are continuous; therefore, f(x) fits the definition of a cubic spline function.

It is convenient to represent f(x) as

$$f(x) = a_i + b_i(x-x_i) + c_i(x-x_i)^2 + d_i(x-x_i)^3,\ x_i \leq x \leq x_{i+1}. \tag{III.42}$$

Since f(x) is continuous,

$$a_i + b_i(x_{i+1}-x_i) + c_i(x_{i+1}-x_i)^2 + d_i(x_{i+1}-x_i)^3 = a_{i+1}. \tag{III.43}$$

where i=0, ..., n-1. Define $h_i = x_{i+1}-x_i$ and (III.43) may be rewritten as

$$b_i = (a_{i+1}-a_i)/h_i - c_i h_i - d_i h_i^2. \tag{III.44}$$

Since f'(x) is continuous,

$$b_i + 2c_i h_i + 3d_i h_i^2 = b_{i+1},\ i=0,\ \ldots,\ n-1. \tag{III.45}$$

Since f'' is continuous and equal to zero at the end points,

$$c_o = c_n = 0,\ 2c_i + 6d_i h_i = 2c_{i+1},\ i=0,\ \ldots,\ n-1 \tag{III.46}$$

or

$$d_i = (c_{i+1}-c_i)/3h_i. \tag{III.47}$$

Using (III.47) to substitute for d_i in (III.45) and (III.44) and eliminating b_i and b_{i+1} from (III.45) by substituting (III.44) results

in

$$c_{i+2}h_{i+1}/3 + c_{i+1}(2h_i/3+2h_{i+1}/3) + c_ih_i/3 = a_{i+2}/h_{i+1}+$$

$$a_{i+1}(-1/h_i - 1/h_{i+1}) + a_i/h_i, \quad i=0, \ldots, n-2. \tag{III.48}$$

Equation (III.48) may be rewritten in matrix notation as

$$Tc = Q^ta, \tag{III.49}$$

where $c = (c_1, c_2, \ldots, c_{n-1})^t$, since $c_o=c_n=0$,

$a = (a_o, a_1, \ldots, a_n)^t$,

T is a positive definite tridiagonal matrix of order n-1:

$$t_{i,i}=2(h_{i-1}+h_i)/3, \; t_{i,i+1}=h_{i+1,i}=h_i/3,$$

and Q is a tridiagonal matrix with n+1 rows and n-1 columns:

$$q_{i-1,i}=1/h_{i-1}, q_{i,1}=-1/h_{i-1}-1/h_i, \; q_{i+1,i}=1/h_i.$$

Since $f'''(x)$ is constant over each interval, using (III.41) results in

$$3d_i - 3d_{i+1} = p(f(x_i)-y_i)/\delta y_i^2, \quad i=0, \ldots, n. \tag{III.50}$$

Substituting (III.45) into (III.50) results in

$$c_{1+2}/h_{i+1} + c_{i+1}(-1/h_i-1/h_{i+1}) + c_i/h_i = p(y_i-a_i)/\delta y_i^2, \quad i=0,$$

$\ldots$, n-1. (III.51)

(III.51) can be rewritten in matrix notation as

$$Qc = pD^{-2}(y-a), \tag{III.52}$$

where $D = \text{diag}\,(y_o, y_1, \ldots, y_n)$,

and $y = (y_o, y_1, \ldots, y_n)$. Premultiplying (III.52) by Q^tD^2 and substituting Tc for Q^ta results in

$$(Q^tD^2Q + pT)c = pQ^ty, \tag{III.53}$$

and solving (III.52) for a results in

$$a = y - p^{-1}D^2Qc. \tag{III.54}$$

If the value of p can be determined, all of the coefficieints of (III.42) can be calculated by solving (III.53) for c, (III.54) for a, and b and d are obtained from (III.44) and (III.46). Expression (III.38) is minimized with respect to p and z by setting the partial derivatives of (III.38) with respect to p and z equal to zero. The result is

$$pz = 0, \tag{III.55}$$

$$\sum_{i=0}^{n} ((f(x_i)-y_i)/\delta y_i)^2 = S - z^2. \tag{III.56}$$

The left hand side of (III.56) can be rewritten as

$$F(p)^2 = \sum_{i=0}^{n} ((a_i-y_i)/\delta y_i)^2. \tag{III.57}$$

F(p) is introduced for notational convenience. F(p) is expressed as the Euclidean norm of a vector by

$$F(p) = \left|\left|D^{-1}\ (a-y)\right|\right|_2. \tag{III.58}$$

If c is eliminated from (III.54) by substituting (III.53) and a eliminated from (III.58) by substituting (III.54) the result is

$$F(p) = \left|\left|DQ(Q^tD^2Q + pT)^{-1}Q^ty\ \right|\right|_2. \tag{III.59}$$

It is concluded from (III.55) that either p=0 or z=0. p=0 implies that $F(0) \leq S^{\frac{1}{2}}$. From (III.53) it is concluded that c=0, and thus, from (III.46) it is seen that d=0. Therefore, if p=0 then the cubic spline reduces to a straight line.

Since a straight line is not the desired result, $p \neq 0$ and z=0, thus, (III.36) reduces to an equality constraint and p is determined

from

$$F(p)=S^{\frac{1}{2}}. \tag{III.60}$$

At this point it is necessary to analyze (III.60) to determine if there is more than one root and which root minimizes (III.38). According to Reinsch (16) it can be shown that the second derivative of F(p) is always positive for $p \leq 0$; therefore, F(p) is convex and (III.60) has exactly one positive root. There is also at least one negative root; however, Reinsch states that it can be shown that a negative root will not minimize (III.38).

Newton's method is used to find the positive root of (III.60). The convexity of F(p) guarantees convergence for a starting value of p=0. Newton's method is expressed as

$$p_{i+1} = p_i - (F(p_i)^2 - S^{\frac{1}{2}}F(p_i))/(F(p_i)dF(p_i)/dp). \tag{III.61}$$

For notational convenience in deriving an expression for $dF(p_i)/dp$ the abbreviation

$$u = p^{-1}c = (Q^tD^2Q + pT)^{-1}Q^ty \tag{III.62}$$

is introduced. The expression $F(p)^2 = u^tQ^tD^2Qu$ results from combining (III.59) and (III.62) and

$$\begin{aligned} FdF/dp &= u^tQ^tD^2(du/dp) \\ &= -u^tQ^tD^2QT(Q^tD^2Q + pT)^{-2}Q^ty \\ &= -u^tQ^tD^2Q(Q^tD^2Q + pT)^{-1}Tu \\ &= -u^t(Q^tD^2Q + pT)(Q^tD^2Q + pT)^{-1}Tu + pu^tT(Q^tD^2Q + pT)^{-1}Tu \\ &= pu^tT(Q^tD^2Q + pT)^{-1}Tu - u^tTu. \end{aligned} \tag{III.63}$$

Since $Q^tD^2Q + pT$ is a positive definite symmetric matrix Cholesky decomposition can be used to calculate R where

$$R^tR = Q^tD^2Q + pT \qquad \text{(III.64)}$$

Cholesky's method results in an upper triangular matrix R, which permits u to be calculated by a forward substitution with R^t, followed by a backward substitution with R.

This derivation leads to the following algorithm for calculating the coefficients of the spline function.

1) Start with p=0.

2) Use Cholesky decomposition to calculate R^tR for $Q^tD^2Q + pT$.

3) Use a forward and a backward substitution to compute u from

$R^tRu = Q^ty$.

4) Calculate $F(p)^2$ from $u^tQ^tD^2Qu$.

5) If $F(p)^2$ is greater than S then compute a new value for p using (III.61).

From (III.63)

$$FdF/dp = pu^tT(R^tR)^{-1}Tu - u^tTu. \qquad \text{(III.65)}$$

is derived. Compute w from $R^tw = Tu$ by a forward substitution, thus

$$FdF/dp = pw^tw - u^tTu. \qquad \text{(III.66)}$$

Having computed a new p, return to step 2.

6) If $F(p)^2$ is less than or equal to S then compute $a = y - D^2Qu$, $c = pu$ and b, d from (III.44) and (III.46).

The attempt to arrive at a smoothing function through the minimization of expression (III.38) results in a cubic spline function with knots at the given values x_i, i=0, ..., n. The amount of smoothing for the entire function is determined by the given constant S, and the amount that each point is smoothed is governed by its standard

deviation δy_i. The determination of the coefficients of the cubic spline hinges upon finding the value of the Lagrange multiplier p which minimizes (III.38). Due to the properties of symmetry and positive definiteness possessed by the matrix $Q^t D^2 Q + pT$, p can be determined in a straight forward iterative manner (16).

CHAPTER IV

PROGRAM DESCRIPTION

General Organization

The Easyplot system consists of two separate program packages. The first consists of seven subroutines coded in American National Standard (ANS) FORTRAN. These subroutines provide an interface between the user of Easyplot and the second portion of the plotting system. The second portion is the plotting program and is also coded in ANS FORTRAN. Its function is to create a file of plot commands which will produce the plots the user has indicated by the parameters passed to the interface subroutines. The interface routines create two data files which contain the parameter information passed in the call lists. The files created by the interface routines are input by the plotting program which is executed as a separate step or job. Portability was the primary consideration in the coding of Easyplot. It is designed to run on any computing system having a standard FORTRAN compiler and a Calcomp Digital Incremental Plotter with the standard Calcomp plotting routines. Only minor modifications such as changing the input and output unit numbers are necessary in order to adapt the program to a different computing system. A flow chart of each significant routine is included in Appendix D.

The Interface Routines

General Comments

The interface portion consists of seven subroutines: GRAPH, GRAPHS, LETTER, LETTRS, ORIGIN, DP, KRPLT, and KRPLTS. The routines create files on units with variable names ITAPE and ITAPE1. The output data is written without format control in binary code. The subroutines GRAPHS, LETTRS, and KRPLTS serve only to permit the user to call GRAPH, LETTER, and KRPLT with a shortened call list. A superposition data set requires no scaling parameters and only one labeling parameter; however, all parameters must be represented in a call list. The subroutines GRAPHS, LETTRS, and KRPLTS initiate calls to GRAPH, LETTER, and KRPLT respectively with complete parameter lists. The call lists for these six subroutines are given in Chapter V.

Subroutine GRAPH

This subroutine creates two files. The file identified by the variable name ITAPE1 is used for scaling purposes. It consists of one logical record for each primary call made to GRAPH, LETTER, or KRPLT. Each record contains the maximum and minimum x and y coordinates of all data sets which will be plotted on a graph. If scaling factors are supplied by the user, the values written on ITAPE1 have no significance. The second file, identified by ITAPE, is contributed to by subroutines ORIGIN, LETTER, KRPLT, and DP as well as GRAPH. ITAPE contains the parameters passed to these routines.

The following algorithm summarizes the computations performed by GRAPH.

1) If GRAPH is called by LETTER or KRPLT then the maximum and minimum x and y coordinates are updated and control is transferred to step 9.

2) Backspace ITAPE one logical record so that the end of file indicator is overwritten.

3) If the call is a primary call, output the primary call indicator on one logical record and output all parameters except the arrays X and Y on the next record. The first record is one word in length and the second is 29 words in length. Transfer to step 5.

4) Output the superposition indicator followed by the parameters used for a superposition call. The first record is one word in length, and the second is eight words in length.

5) If X and Y arrays contain single precision data they are output on one logical record; otherwise, transfer to step 11.

6) If scaling factors were not supplied by the user, the X and Y arrays are searched in order to update the maxima and minima of all previous data sets to be plotted on the graph initialized by the last primary call.

7) If smoothing is specified with standard deviations for each point, the array STDEV is output on ITAPE forming one logical record.

8) Output the end of file indicator on ITAPE. This is a record one word in length.

9) If this call is a superposition call backspace ITAPE1 one record.

10) Output the updated maxima and minima and return.

11) Call subroutine DP and transfer to step 7.

Subroutine DP

This subroutine is called only if the arrays X and Y passed to

GRAPH contain double precision data. The routine outputs the X and Y arrays as single precision data. This data comprizes one logical record on unit ITAPE. The maximum and minimum x and y values are updated and then control is returned to GRAPH.

Subroutine ORIGIN

This subroutine backspaces ITAPE one record and outputs three logical records. The first record is one word in length and is the code indicating the following record was created by ORIGIN. The second record is three words in length and consists of the three parameters. The third record is one word in length and contains the end of file indicator.

Subroutine LETTER

This subroutine backspaces ITAPE one record and outputs three records. The first record is the one word identification code. The second record is 51 words in length and contains all parameters including 20 words for the labeling information in the array called LABL. The third record is the end of file indicator. If this is a primary call, subroutine GRAPH is called in order to initialize a new set of maxima and minima for the graph initiated by this call to LETTER.

Subroutine KRPLT

KRPLT builds data sets for histograms and ideograms and outputs the data sets on ITAPE. This subroutine performs three functions. The first is to initialize the array containing the interval boundaries and the array containing the event counts. The second is to record an

event, and the third is to output the arrays and the remaining parameters.

The initialization call zeroes the HTS array and calculates the interval boundaries storing them in ABCISA. The first location of ABCISA contains the number of intervals plus two. The second location contains the value of AMIN. Each succeeding location is incremented by (AMAX-AMIN)/NOINT, where NOINT is the number of intervals, AMAX is the upper bound of the last interval, and AMIN is the lower bound of the first interval.

The building call for a histogram finds the interval in which the event occurs and adds one to the corresponding location in HTS. A linear search is used to determine that EVENT is greater than or equal to ABCISA(I-1), and less than ABCISA(I), then HTS(I) is incremented by one.

The building call for an ideogram adds the value of a Gaussian curve evaluated at ABCISA(I) to HTS(I). The Gaussian curve is given by

$$1/(\sigma(2\pi)^{\frac{1}{2}})e^{-(x-a)^2/2\sigma^2}, \qquad \text{(IV.1)}$$

where σ is the standard deviation, x is ABCISA(I), and a is the coordinate of the event.

The finalization call creates four logical records on ITAPE. The first is a one word code indicating the following records were created by KRPLT. The second is 28 words in length and contains the scaling and labeling parameters. The third record contains the arrays ABCISA and HTS starting with their second locations. The order in which they are written depends upon whether ABCISA is to be plotted on the horizontal or vertical axis. The maximum and minimum values of the array

HTS are determined and subroutine GRAPH is called to update the maxima and minima record on ITAPE1 if the call to KRPLT was a superposition call. If the call to KRPLT was a primary call a new record is created on ITAPE1. Finally the end of file indicator is output on ITAPE before returning to the user's program.

The Plotting Program

The plotting program inputs the files created by the interface routines and calls Calcomp plotting routines which create a file of plotting commands. The main program inputs the code number of ITAPE indicating which interface subroutine created the next records on ITAPE. The code number triggers a branch to the section which processes the data created by the indicated routine. Several subroutines are included in the plotting program. Subroutine PLTDAT takes care of plotting a set of points according to the criteria indicated by the value of MODE. PLTDAT calls subroutine INTERP which calculates the coefficients of the interpolation curve between a pair of points. If smoothing is desired, PLTDAT calls subroutine SMOTH which calculates all coefficients for the smoothed curve in one call. Subroutine PARMD produces all printed output for the plotting routine. The printer output is a formatted parameter dump giving the value of all parameters that were passed to the interface routines. Due to the impracticality of printing all of the points on each data set, only the first four are printed. Subroutine LGAXIS plots either a horizontal or vertical logarithmic axis.

The following Calcomp supplied subroutines are called by the plotting program.

1) PLOTS initializes the plot command data set.

2) PLOTC moves the ink pen on the plotter to a specified position. The pen may be up or down and the new position may become the origin.

3) SCALE calculates a scale factor for an array of numbers.

4) AXIS plots a linear axis at any angle.

5) SYMBOL plots an array of characters.

6) NUMBER plots a floating point number.

7) LINE plots a set of points which may or may not be connected by line segments.

A detailed description of the Calcomp routines and their parameters is given in the System/360 User's Manual (23).

Main Program Description

The main program is divided into sections which process data output by a particular interface routine. The appropriate section is identified by the code record which precedes each group of parameters of ITAPE.

The section processing data output by a primary call to GRAPH begins the plotting of each new graph. The functions performed by this section are:

1) Inputs the maximum and minimum x and y coordinates from ITAPE1.

2) If LATCH=0 then a new origin is established and the graph border is plotted.

3) If scaling factors are not specified by the user the values from ITAPE1 are passed to subroutine SCALE and appropriate scaling factors are returned.

4) If the axes are linear they are plotted by subroutine AXIS.

5) If the axes are logarithmic and the number of cycles is not supplied by the user then the number of cycles and the minimum power of ten are calculated. The base of the logs determines how the calculation is performed. The logarithmic axes are plotted by subroutine LGAXIS.

6) The 20 character labels in the arrays GLAB and DATLAB are plotted in the upper right corner of the graph.

7) Subroutine PARMD is called to print the graph parameters excluding the data points.

8) Subroutine PLTDAT is called to plot the data points.

The section processing data output by a superposition call to GRAPH checks the value of MODE to determine whether there is a data set label. If so, the label is plotted and subroutine PLTDAT is called to plot the data set. The section processing data output by subroutine ORIGIN moves the origin and prints its position. The section processing data output by subroutine LETTER branches to the section which begins plotting new graphs if the call to LETTER was a primary call. After the axes and border for the new graph are plotted the characters in the array LABL are plotted. If the call to LETTER was a superposition call the characters in LABL are plotted on the current graph.

The section processing data from subroutine KRPLT branches to the section for primary calls to graph if the call to KRPLT was a primary call. Upon returning subroutine PLTDAT is called if an ideogram is to be plotted. Histograms are plotted in this section without calling PLTDAT.

Subroutine PLTDAT

This routine reads a set of data points from ITAPE and plots the points according to the specifications indicated by MODE. If MODE is negative all points with x or y coordinates equal to zero are eliminated from the arrays containing the point coordinates. The coordinates are scaled before they are plotted. The routine performs three different plotting functions. The first is to plot an interpolated curve. The second is to plot a smoothed curve, and the third is to plot the data set points.

The interpolated curve is plotted in sections. Subroutine INTERP calculates the coefficients of a pair of cubic polynomials which determine the curve between adjacent points in the data set. The curve is represented by

$$x = p_1 + p_2 z + p_3 z^2 + p_4 z^3, \qquad \text{(IV.2)}$$

$$y = q_1 + q_2 z + q_3 z^2 + q_4 z^3, \qquad \text{(IV.3)}$$

where z is a parameter varying from 0 to 1 as the curve is traversed from one point to the next. The curve is plotted by incrementing z and evaluating (IV.2) and (IV.3). The points obtained are connected by straight line segments. The increment in z is given by

$$dz = 1/10R \qquad \text{(IV.4)}$$

where R is the distance between the two original points.

The smoothed curve is a cubic spline function with knots (see Chapter II for definition) at the x coordinates of the points approximated by the curve. The portion of the curve between each pair of points is represented by a cubic polynomial in x. The coefficients for all of the polynomials are calculated by one call of subroutine SMOTH.

The array of standard deviations STDEV and the parameter S determine the amount of smoothing exhibited by the curve. If MODE is 31 or 32, S is set to the number of points and STDEV contains the scaled standard deviations supplied by the user. Otherwise, the standard deviation is the same for every point and is given by

$$\delta y = \text{YSIZE}/200. + \text{MOD}(\text{MODE}-1,10)*0.025 , \qquad \text{(IV.5)}$$

where YSIZE is the length of the y-axis. S is given by

$$S = N - (2N)^{\frac{1}{2}} + 2*\text{MOD}(\text{MODE}-1,10)*(2N)^{\frac{1}{2}}/9. \qquad \text{(IV.6)}$$

These formulas were determined by experimentation and observation. The curve is plotted by calculating points along the curve and connecting them with line segments. The points are determined by incrementing x and calculating a corresponding y value. The increment in x decreases as the rate of change of curvature increases and vice versa.

The data points are plotted by subroutine LINE. They may be connected by line segments if desired.

Subroutines INTERP and SMOTH

The algorithms for interpolation and smoothing are derived and outlined in Chapter III. These algorithms are implemented in the subroutines INTERP and SMOTH.

Subroutine LGAXIS

This routine plots logarithmic axes. The axis plotted may be horizontal or vertical and has an integral number of cycles. Each cycle has a tick mark at the values 1 through 9 times the power of 10 corresponding to the cycle. Tick marks 1, 3, 5, and 8 are labeled. The axis label is also plotted by this routine.

CHAPTER V

EASYPLOT USER'S GUIDE

Introduction

The plotting capabilities of Easyplot are utilized through subroutine calls from a FORTRAN program. Four routines are available each of which performs specific functions. They may be used separately or in combination. Subroutine GRAPH is the most general routine and performs most commonly needed plotting applications. One call to GRAPH can produce a complete plot including vertical and horizontal axes which may be linear, semi-log, or log-log; data points plotted with symbols, open or closed interpolation, or smoothing; automatic or specified scaling of data; and labeling of the axes and each set of data. More than one set of data can be plotted on one graph by additional calls to GRAPH.

Subroutine LETTER is used for additional labeling. Subroutine ORIGIN is used to control the position of the origin and to suppress the plotting of either or both axes. Subroutine KRPLT is used to gather data for histograms and ideograms and plot the results.

Subroutine GRAPH

Introductory Notes

The user must build two arrays which contain the x and y

coordinates of the points which are to be plotted. The parameters in the call list pass information to GRAPH determining the manner in which the points are to be plotted. There are two types of calls to GRAPH. The first is called a primary call. The primary call positions the origin two inches past the border of the previous graph and makes the necessary preparations for plotting a new set of axes. The second type of call is a superposition call. The set of points passed in a superposition call is plotted on the graph initialized by the previous primary call. The origin is not moved. A superposition call is made either by invoking GRAPHS or invoking GRAPH with the parameter XSIZE having a value of 0.0. A call to GRAPH with a value for XSIZE other than 0.0 indicates a primary call. The actual plotting of a graph does not take place until all superposition calls have been made. This means that if scaling is not specified by the user in the primary call, the points in the primary data set and all superposition data sets are used in determining appropriate scale factors.

Call List

Primary or Superposition Call: CALL GRAPH (NPTS, X, Y, KS, MODE, XSIZE, YSIZE, XSF, XMIN, YSF, YMIN, XLAB, YLAB, GLAB, DATLAB, STDEV).

Superposition Call: CALL GRAPHS (NPTS, X, Y, KS, MODE, DATLAB).

Parameter Definitions

Except for character strings used for labeling, the parameter names follow the standard FORTRAN naming convention for data type. Parameters beginning with the letters I thru N must be integer type variables or constants in the call list. All other parameters must be

real type variables or constants.

NPTS - Integer number whose magnitude is the number of data points in the arrays X and Y. A positive value indicates X and Y contain real single precision data. A negative value indicates X and Y contain double precision data.

X - An array containing the horizontal components of the data points (x-coordinates).

Y - An array containing the vertical components of the data points (y-coordinates).

KS - An integer number identifying the plotting symbol (if any). Refer to Appendix B for the symbols allowed and their code numbers.

MODE - A number which specifies the method to be used in plotting the data in X and Y.

= 1 Points plotted with the symbol KS and connected with an open (see Chapter I) smooth interpolated curve.

= 2 Points not plotted but they are connected with an open smooth interpolated curve.

= 3 Points plotted with the symbol KS and connected with straight line segments.

= 4 Points are not plotted with the symbol KS but are connected with straight line segments.

= 5 Points are plotted with the symbol KS and connected with a closed (see Chapter I) interpolated curve.

= 6 Points not plotted but connected with a closed interpolated curve.

= 7 Points are plotted with the symbol KS.

Note: If more than 20 points are to be plotted with option 7, they should be ordered by the x coordinate in order to decrease the required plotting time. If the number of points is about 200 or more, further significant savings in plotting time can be attained in the following manner. The range of the x coordinates is divided into strips of equal width. The points in each strip are sorted by their y coordinates in order to minimize the amount of vertical movement necessary. The width of the strips is important in minimizing the vertical movement and should be determined by

$$w = (3*XSIZE*YSIZE/NPTS)^{\frac{1}{2}}.$$

= 11-20 The points are plotted with the symbol KS and a smoothed curve is plotted which approximates the data. The amount of smoothing is increased as MODE varies from 11 to 20.

= 21-30 The points are not plotted; however, a smoothed curve is plotted which approximates the data. The amount of smoothing is increased as mode varies from 21 to 30.

= 31 The points are plotted with the symbol KS and a smoothed curve is plotted approximating them.

= 32 The points are not plotted but are approximated by a smoothed curve.

Note: If the value of MODE is not defined, the data set will not be plotted; however the plotting of axes and labels will not be affected.

Interpolation and Smoothing. If MODE is 31 or 32 the parameter STDEV must contain an approximation of the standard deviation of each data point. The magnitude of the standard deviation will locally determine the smoothness of the curve. That is, the smaller the standard deviation of a point the closer the curve must come to that point. The standard deviations must be positive and there must be one for each point. The curve drawn with the smoothing option must be single-valued; therefore, the points must be ordered so that the x coordinates are monotone increasing.

Interpolated curves may be multi-valued regardless of whether they are open or closed. Interpolated curves pass through all points of a data set in the order in which the points are loaded into the X and Y arrays. Interpolation cannot be used with less than three points.

XLAB - Alphanumeric label for the x-axis. It must be an array with a dimension of five and contain 20 characters including blanks.

YLAB - Same as XLAB except for the y-axis.

GLAB - Alphanumeric label which is printed in the upper right hand corner of the graph. It must be an array with a dimension of five and contain 20 characters including blanks.

DATLAB-Same as GLAB except it is plotted immediately below GLAB.

XSIZE -It defines the length of the x-axis in inches and the sign indicates whether the axis is linear or logarithmic.

XSIZE positive means the axis is linear.

XSIZE negative means the axis is logarithmic.

XSIZE equal to zero indicates a superposition call.

YSIZE -Same as XSIZE except that a value of zero has no meaning, and it applies to the y-axis.

Scaling Parameters. XSF, XMIN, YSF, and YMIN are used to determine the scaling factors for the axes. Their meanings differ depending upon whether a linear or log axis is to be drawn. YSF and YMIN refer to the y-axis and are completely independent of XSF and XMIN; however, their meanings are analogous. Thus, only XSF and XMIN will be defined.

Linear Axis. XSIZE is positive. The axis is drawn with a tick mark every inch and a number below each mark.

XSF - The scale factor in units per inch for the x-axis. If XSF is positive the value of XMIN is printed below the first tick mark, and the number at each following tick mark is increased by XSF. If XSF is negative or zero Easyplot will calculate an appropriate minimum value and scaling factor depending on the minimum and maximum x values for the primary and all superposition sets of data. The values are chosen so that all data points will appear on the graph. The scaling factor will be one of the following numbers: (1.0, 2.0, 4.0, 5.0, 8.0) x $10^{\pm i}$, i=integer.

Logarithmic axis. XSIZE is negative. The axis is drawn with an integer number of cycles. Each cycle has nine tick marks corresponding to the values 1 through 9 times the power of ten associated with the cycle. It is assumed that the relevant array already contains logarithms, that is, the routine does not take the log for the user.

XSF - Specifies the base of the log and the number of cycles on the axis. If XSF is positive or zero, the X array contains base 10 logs, and if XSF is negative, the X array contains natural logs.

|XSF| > 1.0 - XSF truncated to an integer specifies the number of cycles and XMIN truncated to an integer is the power ten printed under the first tick mark.

|XSF|<1.0 - The number of cycles and the minimum power of ten are chosen by the routines so that all points will appear on the graph.

Note on Scaling. If scaling is specified by the user some points may fall outside the axes or the border of the graph which is drawn ½ inch beyond the ends of the axes. The points will be plotted on the axis or border, whichever is appropriate. No interpolation is done between a pair of points if one or both of them lie outside of the range of the graph as defined previously. Similarly any portion of a smoothed curve will not be plotted if it is outside the range of the graph.

STDEV - An array containing the standard deviation of each value in the Y array. This parameter is used only when smoothing is desired and then only if MODE is 31 or 32. If this parameter is not used its position can be filled in the call list with a dummy variable or constant.

Plotting Superposition Data Sets. When XSIZE=0.0 the data in the X and Y arrays are plotted on the axes produced by the last primary call to one of the plotting routines. The scaling and labeling parameters are not used for a superposition call; however, if the call is to GRAPH rather than GRAPHS all parameters must be represented in the call list. Unused parameters may be represented with zeroes or any dummy variable or constant.

Key Facility. A key facility is provided when a superposition call is made which permits the user to label the superposition data set. If the absolute value of MODE is greater than 100, the label contained in DATLAB will be plotted immediately below the last label which

has been plotted in the upper right hand corner of the graph. The symbol KS will also be plotted immediately to the right of the label. The absolute value of MODE modulo 100 will be used as described under the MODE parameter to determine how the data are to be plotted.

Point Elimination. If MODE is negative all data points whose x or y value is zero will be ignored by GRAPH. The remaining points will be plotted according to the absolute value of MODE.

Program Restrictions.

1. The number of points in a single set of data must not exceed 5000.

2. When the smoothing option is used, the number of points must not exceed 1000.

3. Units numbered 14 and 15 are used by Easyplot and cannot be used by the user.

4. XSIZE and YSIZE should be at least 2.5 inches since the graph labels in the upper right hand portion of the plot start 3.0 inches to the left of the right border of the graph.

5. If the absolute value of YSIZE is greater than 10.0 inches, Easyplot will assume a value of exactly 10.0 inches.

Subroutine LETTER

The purpose of subroutine LETTER is to permit additional labeling for any graph. Up to 80 alphanumeric characters can be printed starting at any position and plotted at any angle. It is primarily used in the superposition mode; however, it can be used to define a primary graph.

Call List

Superposition Call. CALL LETTER (XO, YO, HEIGHT, LABL, THETA, NCHAR, 0,0,0,0,0,0,0,0,0,0) or CALL LETTRS (XO, YO, HEIGHT, LABL, THETA, NCHAR).

Primary Call. CALL LETTER (XO, YO, HEIGHT, LABL, THETA, NCHAR, XSIZE, YSIZE, XSF, XMIN, YSF, YMIN, XLAB, YLAB, GLAB, DATLAB).

Parameter Definitions

Except for character strings, the parameter names follow the standard FORTRAN naming convention for data type. Parameters beginning with the letters I through N must be integer type variables or constants. All other parameters must be real type variables or constants.

XO, YO - The coordinates in inches of the lower left corner of the first character of the label.

HEIGHT - Height in inches of the characters in the label.

LABL - An array containing the characters of the label. The characters must be packed four to a word with a maximum of 80 characters allowed.

THETA - The angle in degrees at which the label is to be plotted.

NCHAR - The number of characters, including blanks, which are to be plotted. The maximum allowed value is 80. If the user desires to plot more than 80 characters, it is necessary to make more than one call to LETTER.

The remaining parameters listed for a primary call have the same definitions as the parameters for GRAPH which have the same names.

Subroutine ORIGIN

The origin is automatically positioned two inches past the border of the previous graph every time a primary call is made to subroutine GRAPH, LETTER, or KRPLT. Subroutine ORIGIN permits the user to position the origin anywhere he likes and prevents Easyplot from automatically repositioning the origin when primary calls are encountered. Once control is assumed by the user the origin cannot be moved except by another call to ORIGIN.

Call List

CALL ORIGIN (AA, BB, LATCH).

Parameter Definitions

AA, BB - The coordinates in inches of the new origin relative to the present origin.

LATCH - An integer value which determines the action to be taken by ORIGIN.

- = 0 This is the normal mode of operation for automatic origin positioning. AA and BB are not used in this case.
- = 1-4 The user assumes control of the origin and the plotting of axes becomes optional. AA and BB are used as defined above.
- = 1 Both axes are plotted.
- = 2 The x-axis is not plotted.
- = 3 The y-axis is not plotted.

= 4 Neither axis is plotted.

Note: If LATCH has a value other than 0 through 4, 0 is assumed.

Subroutine KRPLT

Subroutine KRPLT is used to gather data for histograms or ideograms and plot the results. A histogram is a frequency count of events, and an ideogram is an extension of a histogram which allows an error to be associated with each event.

For a histogram the abcissa is divided into equal intervals, and a count of the number of events occurring in each interval is accumulated. A plot of the abcissa versus the count for each interval is produced.

For an ideogram the abcissa is divided into intervals as for the histogram; however, each event is represented by a Gaussian curve centered at the event's coordinate. The Gaussian or normal distribution curve has an area of one, and the standard deviation of the curve is the error associated with the event. The contributions of each Gaussian curve at the boundaries of each interval are accumulated. Thus, a single event contributes to all intervals rather than only one as for a histogram. A plot of the interval boundaries versus their corresponding sums is produced. The points are connected with an interpolated curve.

There are three types of calls to KRPLT all of which are necessary to produce either a histogram or an ideogram. The first call must be an initialization call which specifies the boundaries of the abcissi intervals and the number of intervals. The second type of call is a building call which must be made once for each event. A running total of the counts for a histogram or sums for an ideogram is maintained.

The last call must be the finalization call which begins the production of the plot. The final call may be a superposition call or a primary call.

Call List

Initialization Call. CALL KRPLT (K, ABCISA, HTS, NOINT, AMIN, AMAX, 0,0,0,0,0,0,0) or CALL KRPLTS (K, ABCISA, HTS, NOINT, AMIN, AMAX).

Building Call. CALL KRPLT (K, ABCISA, HTS, EVENT, SIGMA, 0,0,0,0,0,0,0,0) or CALL KRPLTS (K, ABCISA, HTS, EVENT, SIGMA, 0).

Finalization Call. CALL KRPLT (K, ABCISA, HTS, XSIZE, YSIZE, XSF, XMIN, YSF, YMIN, XLAB, YLAB, GLAB, DATLAB) for a primary data set. CALL KRPLT (K, ABCISA, HTS, 0,0,0,0,0,0,0,0,0,0) or CALL KRPLTS (K, ABCISA, HTS, 0.0.0) for a superposition data set.

Parameter Definitions

The parameter names follow the standard FORTRAN naming convention for data type. Parameters beginning with the letters I through N must be integer type variables or constants. All other parameters must be real type variables or constants.

K - The value of K determines the type of call and whether a histogram or an ideogram is being plotted.

= 0 Indicates an initialization call.

= 1 Indicates a building call for an ideogram.

= 2 Indicates a building call for a histogram.

= -1 Indicates a final call for an ideogram which is to be plotted with the abcissa in the horizontal direction.

= -2 Indicates a final call for a histogram which is to be plotted with the abcissa in the horizontal direction.

= -3 Indicates a final call for an ideogram which is to be plotted with the abcissa in the vertical direction,

= -4 Indicates a final call for a histogram which is to be plotted with the abcissa in the vertical direction.

NOINT - The number of intervals on the abcissa. The maximum allowable is 1000.

AMIN - Beginning value of the first interval.

AMAX - Ending value of the last interval.

EVENT - The coordinate of the event to be recorded.

SIGMA - The standard deviation associated with each event. Used only for ideograms.

ABCISA- Work array which must be dimensioned at least two greater than the number of intervals. The first location contains the number of intervals plus two and the remaining locations contain the bounds of the intervals starting with AMIN and ending with AMAX. The array is initialized during the initialization call and must not be altered by the user until after the final call.

HTS - Work array which must be dimensioned at least two greater than the number of intervals. The array is initialized to zeroes during the initialization call and must not be altered by the user until after the finalization call. For a histogram HTS(1) is not used and HTS(I) where $I \geq 1$ contains the number of events occurring in the interval bounded by ABCISA(I-1) and ABCISA(I). For an ideogram HTS(1) is not used and HTS(I) for

I > 1 contains the sum of all Gaussians evaluated at ABCISA(I).

If K has a value that is not defined no action is taken, and the message, "FIRST PARAMETER IN CALL TO KRPLT IS INVALID", is printed. If the number of intervals in an initialization call is negative or zero the message, "THE NUMBER OF INTERVALS FOR KRPLT IS NEGATIVE", is printed.

The remaining parameters in the finalization primary call have the same definitions as the corresponding parameters for subroutine GRAPH.

Debugging Aids

Easyplot is designed to allow simultaneous debugging of program logic and plotting logic. The four subroutines called by a user of Easyplot perform few operations with the parameters in the call lists. Their main function is to output the parameters to a file which is passed to another step or job which actually performs the plotting functions. Since the plotting package is organized in this manner, it is very unlikely that a user's program will abnormally terminate as a result of passing invalid parameters to one of the routines. Thus, the program will terminate normally unless the user's program logic causes an abnormal termination. The plotting program may be executed regardless of the cause of termination of the user's program. This separation of the plotting action and user's program allows simultaneous debugging.

The plotting program has an additional debugging aid consisting of a parameter dump. The program prints the values of all parameters passed to each of the four routines in a convenient format. Since it would be impractical to print all of the points in the X and Y arrays,

only the first four are printed for each set of data. Examples of the information printed in the parameter dump are given with the test programs in Appendix C.

CHAPTER VI

SUMMARY AND SUGGESTED EXTENSIONS

The Easyplot program is an extremely versatile plotting package which can save a great deal of time and money for most programming installations utilizing the Calcomp plotter. Only very specialized plotting applications such as multi-dimensional contour plots are beyond the scope of Easyplot. Two-dimensional graphs are plotted in a generalized format and can be produced easily. In addition the graph format may be altered with the specialized Easyplot routines ORIGIN and LETTER, and histograms and ideograms may be built and plotted.

The numerical methods for interpolation and smoothing chosen for use in Easyplot have been shown to be well suited for general plotting applications. A comparison of popular interpolation and smoothing techniques given in Chapter II indicates that Akima's (3) and Reinsch's (16) techniques are among the best methods for this purpose. Various tests using these techniques have verified that they produce acceptable curves.

The smoothing technique used in Easyplot is applicable only to single-valued curves. The plotting program could be extended to smooth multi-valued sets of points by using a new smoothing technique involving parametric curve fitting (12). Another possible extension would be the inclusion of a sorting routine to be used in conjunction with

plotting large numbers of individual points. Significant savings in plotting time can be realized by ordering the points according to the scheme used by Scranton (20).

SELECTED BIBLIOGRAPHY

(1) Ackland, T. G. "On Osculatory Interpolation, Where the Given Values of the Function Are at Unequal Intervals." J. Inst. Actuar. 49 (1915), 369-375.

(2) Akima, Hiroshi. "A Method of Smooth Curve Fitting." ESSA Tech. Rep. ERL 101-ITS 73. U. S. Government Printing Office, Washington, D. C., 1969.

(3) Akima, Hiroshi. "A New Method of Interpolation and Smooth Curve Fitting Based on Local Procedures." Journal of the Association for Computing Machinery, Vol. 17, No. 4 (October, 1970), 589-602.

(4) Berztiss, J. T. "Least Squares Fitting of Polynomials to Irregularly Spaced Data." SIAM Review, Vol. 6, No. 3 (July, 1964), 203-227.

(5) Bjorck, A., and G. H. Golub. "Iterative Refinements of Linear Least Squares Solutions by Householder Transformations." Report No. CS-83, Computer Science Department, Stanford University (January, 1968).

(6) Carnahan, B., H. A. Luther, and J. Wilkes. Applied Numerical Methods. New York: John Wiley and Sons, Inc., 1969.

(7) deBoor, C., and J. R. Rice. "Least Squares Cubic Spline Approximation I-Fixed Knots." Report No. CSD-TR-20, Computer Sciences Department, Purdue University (April, 1968).

(8) deBoor, C., and J. R. Rice. "Least Squares Cubic Spline Approximation II-Variable Knots." Report No. CSD-TR-21, Computer Sciences Department, Purdue University (April, 1968).

(9) Forsythe, G. E. "Generation and Use of Orthogonal Polynomials for Data-Fitting with a Digital Computer." SIAM Journal, Vol. 5, No. 2 (June, 1957), 74-88.

(10) Goertzel, G. "An Algorithm for the Evaluation of Finite Trigonometric Series." American Mathematical Monthly, Vol. 65, 1958, 34-35.

(11) Golub, G. H. "Numerical Methods for Solving Linear Least Squares Problems." Numerische Mathematik 7, 1965, 206-216.

(12) Grossman, M. "Parametric Curve Fitting." The Computer Journal, Vol. 14, No. 2 (May, 1971), 169-172.

(13) Hildebrand, F. B. Introduction to Numerical Analysis. New York: McGraw-Hill, 1956.

(14) Kendall, M. G., and A. Stuart. Advanced Theory of Statistics, Vol. 2, 2nd Ed. New York: Hafner Publication Co., 1967.

(15) Milne, W. E. Numerical Calculus. Princeton: Princeton University Press, 1949.

(16) Reinsch, C. H "Smoothing by Spline Functions." Numerische Mathematik, 10 (October, 1967), 177-183.

(17) Rund, Hanno. The Hamilton-Jacobi Theory in the Calculus of Variations. New York: D. Van Nostrand Co., Inc., 1966.

(18) Savitsky, A., and M. J. E. Galay. "Smoothing and Differentiation of Data by Simplified Least Squares Procedures." Analytical Chemistry, Vol. 36, No. 8 (July, 1964), 1627.

(19) Scranton, D. G. "SIMPLOTTER for B. P. S. FORTRAN." Computer Utilization Bulletin #23, Ames Laboratory USAEC, c.a. 1965.

(20) Scranton, D. G., and E. G. Manchester. "The Use of SIMPLOTTER, a High Level Plotting System." Report No. IS-2305, Ames Laboratory USAEC, 1971.

(21) Smith, L. B. "The Use of Man-Machine Interaction in Data-Fitting Problems." Technical Report No. CS131, Computer Science Department, Stanford University, 1969.

(22) Smith, L. B. "A Survey of Interactive Graphical Systems for Mathematics." Report No. SLAC-PUB-540, USAEC, Stanford University, 1969.

(23) System/360 User's Manual. Stillwater: Oklahoma State University, 1970.

(24) Whittaker, E., and G. Robinson. The Calculus of Observations, a Treatise of Numerical Mathematics. London: Blackie and Son Limited, 1944.

APPENDIX A

JOB CONTROL LANGUAGE FOR EASYPLOT

The job control statements necessary to run any FORTRAN job on an IBM System/360 plus several additional statements needed specifically for Easyplot are shown in Figure 5. All of the statements need not be coded exactly as shown; however, there are several parts which cannot be changed. Statements 4, 5, 6, 12, and 13 must be included exactly as shown. The following parameters must be included as shown.

Statement 8 - The name FT14F001 and the parameter RECFM=VS.

Statement 9 - The name FT15F001 and RECFM=VS.

Statement 15 - The name FT14F001 must be used, and the name of the data set must be the same as the name used in statement 8.

Statement 16 - The name FT15F001 must be used, and the data set name must be the same as the one in statement 9.

Statement 17 - The name PLOTOUT and the DISP parameter with a KEEP disposition must be used. Also, the DSN must be in the form shown.

```
1.  //   JOB
2.  //   EXEC FORTGLCLG
3.  //FORT.SYSIN DD *

        The user's FORTRAN program which calls the Easyplot interface
        routines is placed here.

4.  //LKED.PLOTS DD DSN=OSU.ACT11098.PROG,DISP=SHR
5.  //LKED.SYSIN DD *
6.  INCLUDE PLOTS(GRAPH)
7.  /*
8.  //GO.FT14F001 DD UNIT=SYSDA,SPACE=(TRK,(10,10)),DSN=&&PLTDATA,
    //      DCB=(RECFM=VS,BLKSIZE=3500),DISP=(NEW,PASS)
9.  //GO.FT15F001 DD UNIT=SYSDA,SPACE=(TRK,(1,1)),DISP=(NEW,PASS),
    //    DCB=(RECFM=VS,BLKSIZE=240),DSN=&&MAXMIN
10. //   EXEC PGM=EASYPLOT,REGION=127K
11. //STEPLIB DD DSN=OSU.ACT11098.PROG,DISP=SHR
12. //FT06F001 DD SYSOUT=A
13. //FT14F001 DD DSN=&&PLTDATA,DISP=(OLD,DELETE)
14. //FT15F001 DD DSN=&&MAXMIN,DISP=(OLD,DELETE)
15. //PLOTOUT DD UNIT=PLT,SPACE=(TRK,(10,10)),DISP=(,KEEP),
   //    DSN=PLOT.ACTxxxxx.yyyyyyyy
```

xxxxx represents a five digit account number.

yyyyyyyy represents a one to eight letter name which identifies the plot data set.

Figure 5. Job Control Language for the IBM S/360

APPENDIX B

PLOTTING SYMBOLS

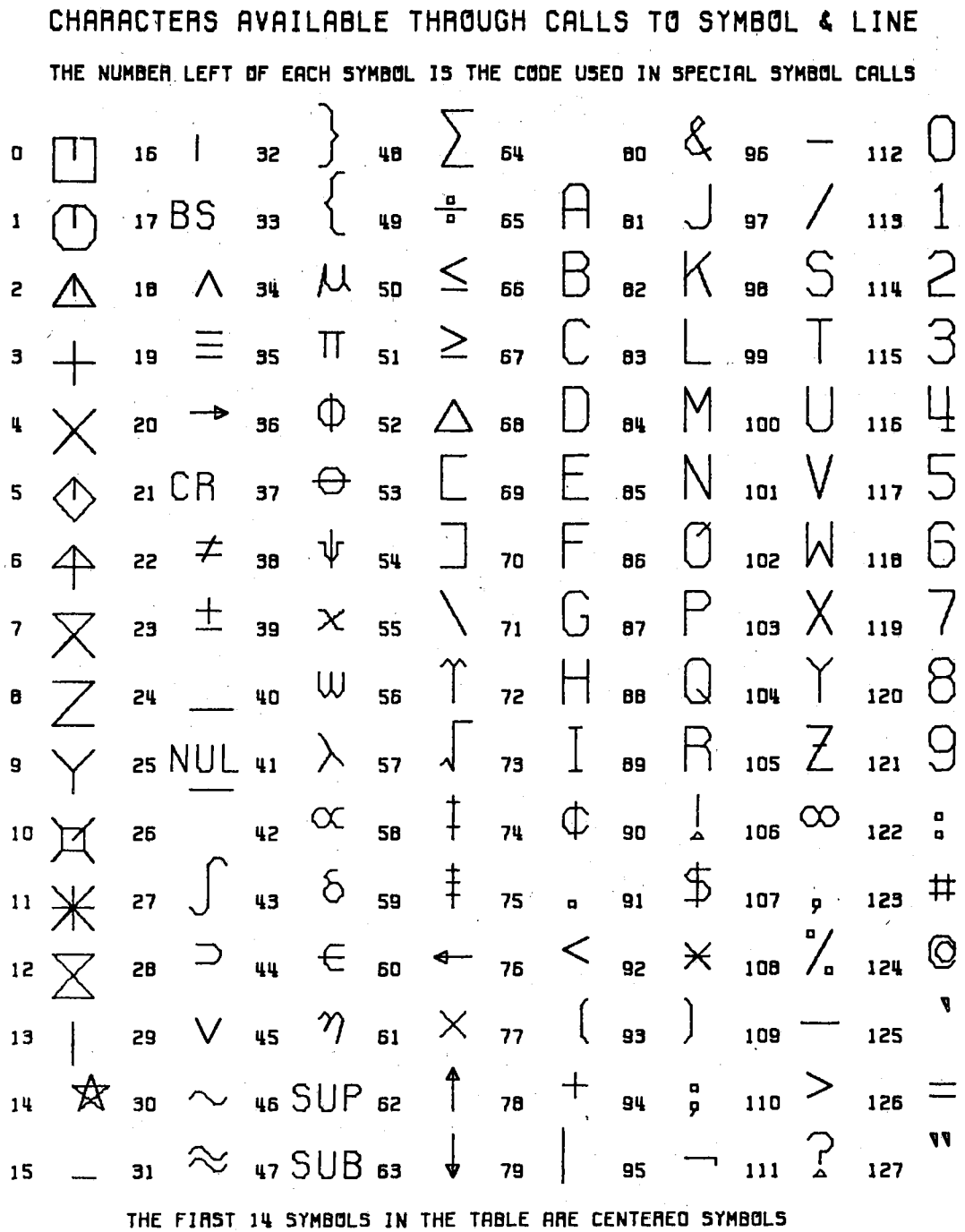


Figure 6. Symbols Available on the Calcomp Plotter

APPENDIX C

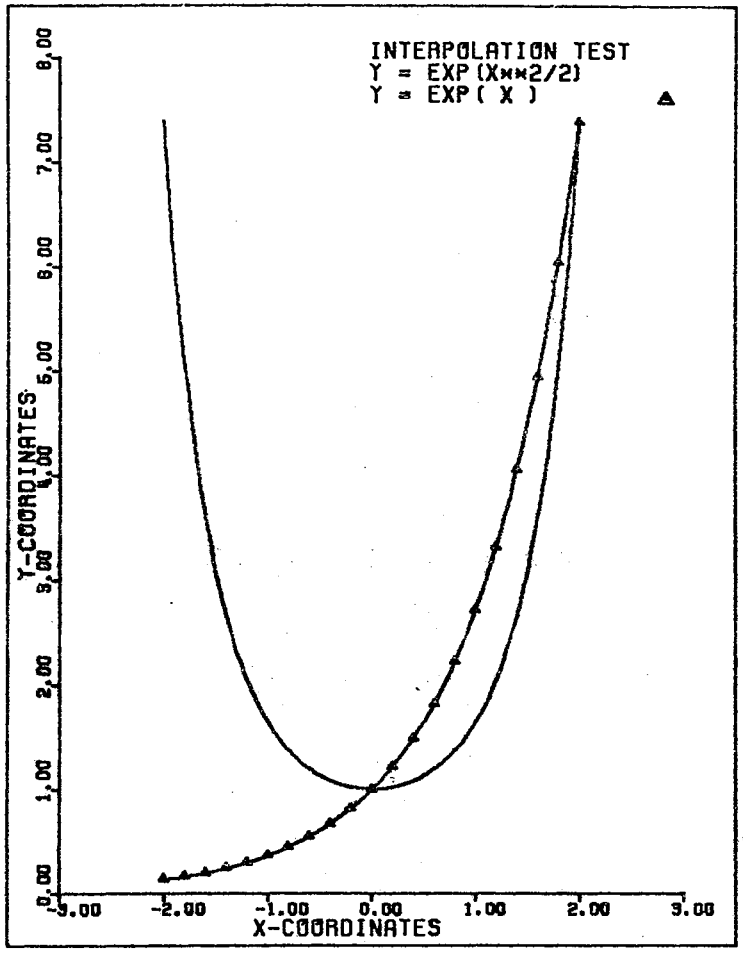
TEST PROGRAMS

```
FORTRAN IV G LEVEL  19                MAIN              DATE = 72122          14/38/15          PAGE 0001

            C     TEST PROGRAM NUMBER 1
            C
            C     AUTHOR. JOHN FORREST
            C     DATE.   MAY, 1972
            C
            C          THIS PROGRAM ILLUSTRATES THE INTERPOLATION AND LOGARITHMIC
            C     AXES FACILITIES OF GRAPH.
            C     THE FIRST GRAPH CONTAINS TWO INTERPOLATED CURVES WHICH ARE
            C     Y=EXP(X**2/2.) AND Y=EXP(X).  THE SECOND GRAPH HAS A VERTICAL
            C     LOGARITHMIC AXES WITH THE BASE E LOG OF THE Y VALUES PLOTTED.
            C
0001              DIMENSION  X(21),Y(21),Y1(21),XL(5),YL(5),GL(5),DL(5)
0002              IN = 5
0003              READ(IN,10) XL,YL,GL,DL
            C     CALCULATE VALUES FOR Y AND Y1.
0004              DX = -2.0
0005              DO 1 I=1,21
0006              X(I) = DX
0007              Y(I) = EXP ( DX**2/2.)
0008              Y1(I) = EXP( DX )
0009            1 DX = DX + 0.2
            C     PLOT THE INTERPOLATED CURVES.
0010              CALL GRAPH (21,X,Y,0,2,6.0,8.0,1.,-3.,0,0,XL,YL,GL,DL,0)
0011              READ(IN,10) DL
0012              CALL GRAPHS(21,X,Y1,2,101,DL)
            C     TAKE THE NATURAL LOG OF Y AND Y1.
0013              DO 2 I=1,21
0014              Y(I) = ALOG(Y(I))
0015            2 Y1(I) = ALOG(Y1(I))
0016              READ(IN,10) GL,DL
            C     PLOT AN INTERPOLATED CURVE THROUGH Y AND PLOT Y1 WITH THE POINTS
            C     CONNECTED BY LINE SEGMENTS.
0017              CALL GRAPH (21,X,Y,3,1,6.0,-8.0,1.,-3.,-.5,0,XL,YL,GL,DL,0)
0018              CALL GRAPHS(21,X,Y1,4,3,0)
0019           10 FORMAT(20A4)
0020              STOP
0021              END
```

Figure 7. Test Program Number One

```
EASYPLOT PARAMETER DUMP....CALCOMP PLOTTER VERSION    (J. R. FORREST, COMPUTING AND INFORMATION SCIENCES DEPT., O.S.U.)

EACH GRAPH IS REPRESENTED BY ONE BLOCK OF DATA WHICH INCLUDES
          1.  A SUMMARY OF PARAMETERS PASSED VIA CALL LISTS.
          2.  ACTUAL SCALE FACTORS USED IN THE PLOT.
          3.  FIRST FOUR POINTS OF EACH DATA SET.
              ( *    INDICATES A HISTOGRAM WITH HORIZONTAL ABCISSA,
                **   INDICATES A HISTOGRAM WITH VERTICAL ABCISSA,
                ***  INDICATES AN IDEOGRAM WITH HORIZONTAL ABCISSA,
                **** INDICATES AN IDEOGRAM WITH VERTICAL ABCISSA )


        XSIZE       YSIZE     XSF        XMIN        YSF        YMIN        X-AXIS LABEL          Y-AXIS LABEL          GRAPH LABEL
        6.000       8.000   0.100E 01 -0.300E 01   0.0          0.0        X-COORDINATES         Y-COORDINATES       INTERPOLATION TEST
 ACTUAL SCALE FACTORS        0.100E 01 -0.300E 01  0.100E 01   0.0

    DATA NO OF  SYM OPT
    SET  POINTS BOL ION        (X1,Y1)                 (X2,Y2)                 (X3,Y3)                 (X4,Y4)          DATA SET LABEL
      1    21     0   2 -0.200E 01, 0.739E 01 -0.180E 01, 0.505E 01 -0.160E 01, 0.360E 01 -0.140E 01, 0.266E 01 Y = EXP(X**2/2)
      2    21     2 101 -0.200E 01, 0.135E 00 -0.180E 01, 0.165E 00 -0.160E 01, 0.202E 00 -0.140E 01, 0.247E 00 Y = EXP( X )


        XSIZE       YSIZE     XSF        XMIN        YSF        YMIN        X-AXIS LABEL          Y-AXIS LABEL          GRAPH LABEL
        6.000      -8.000   0.100E 01 -0.300E 01 -0.500E 00   0.0          X-COORDINATES         Y-COORDINATES       BASE E LOG AXIS TEST
 ACTUAL SCALE FACTORS        0.100E 01 -0.300E 01  0.200E 01 -0.100E 01

    DATA NO OF  SYM OPT
    SET  POINTS BOL ION        (X1,Y1)                 (X2,Y2)                 (X3,Y3)                 (X4,Y4)          DATA SET LABEL
      1    21     3   1 -0.200E 01, 0.200E 01 -0.180E 01, 0.162E 01 -0.160E 01, 0.128E 01 -0.140E 01, 0.980E 00 Y=X**2/2 & Y=X
      2    21     4   3 -0.200E 01,-0.200E 01 -0.180E 01,-0.180E 01 -0.160E 01,-0.160E 01 -0.140E 01,-0.140E 01
```

Figure 8. Parameter Dump for Test Program Number One



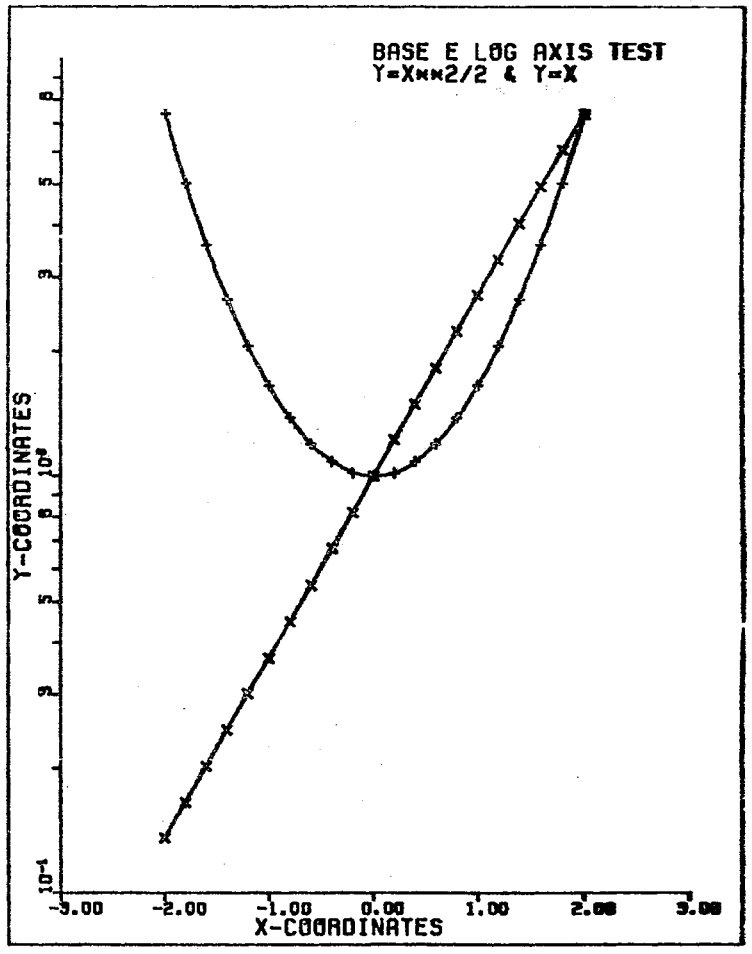


Figure 9. Plot Produced by Test Program Number One

```
      C     TEST PROGRAM NUMBER 2
      C
      C     AUTHOR. JOHN FORREST
      C     DATE.   MAY, 1972
      C
      C          THIS PROGRAM ILLUSTRATES THE USE OF ORIGIN, LETTER, AND THE
      C     SMOOTHING OPTION OF GRAPH.  AN ARBITRARY SET OF POINTS ARE PLOTTED
      C     AND A SMOOTHED CURVE APPROXIMATING THE POINTS IS PLOTTED.
      C
0001        DIMENSION X(256),Y(256),XL(5),YL(5),GL(5),DL(5),LABL(20)
      C     READ IN THE LABELS AND INITIALIZE THE X ARRAY.
0002        IN = 5
0003        READ(IN,10) XL,YL,GL,DL
0004     10 FORMAT(20A4)
0005        DO 1 I=1,256
0006      1 X(I) = I
0007        READ(IN,20) Y
0008     20 FORMAT(20F4.0)
      C     POSITION THE ORIGIN AT (0.,0.7).
0009        CALL ORIGIN(0.,-11.,1)
0010        CALL ORIGIN(0.,.7,1)
      C     PLOT THE SET OF POINTS WITH THE SYMBOL +.
0011        CALL GRAPH(256,X,Y,3,7,6.5,4.0,0,0,0,0,0,XL,YL,GL,DL,0)
      C
      C     MOVE THE ORIGIN UP 5 INCHES AND SUPPRESS THE X-AXIS.
0012        CALL ORIGIN(0.,5.,2)
      C     READ A NEW SET OF LABELS.
0013        READ(IN,10) YL,GL,DL
      C     PLOT A SMOOTHED CURVE WITH THE POINT NEGLECT OPTION.
0014        CALL GRAPH (256,X,Y,0,-30,6.5,4.0,0,0,0,0,0,XL,YL,GL,DL,0)
0015        READ(IN,10) LABL
      C     PLOT A SPECIAL LABEL USING LETTER.
0016        CALL LETTRS(1.0,1.0,0.21,LABL,0.0,17)
0017        STOP
0018        END
```

Figure 10. Test Program Number Two

```
EASYPLOT PARAMETER DUMP....CALCOMP PLOTTER VERSION    (J. R. FORREST, COMPUTING AND INFORMATION SCIENCES DEPT., O.S.U.)

EACH GRAPH IS REPRESENTED BY ONE BLOCK OF DATA WHICH INCLUDES
          1.  A SUMMARY OF PARAMETERS PASSED VIA CALL LISTS.
          2.  ACTUAL SCALE FACTORS USED IN THE PLOT.
          3.  FIRST FOUR POINTS OF EACH DATA SET.
              ( *    INDICATES A HISTOGRAM WITH HORIZONTAL ABCISSA,
                **   INDICATES A HISTOGRAM WITH VERTICAL ABCISSA,
                ***  INDICATES AN IDEOGRAM WITH HORIZONTAL ABCISSA,
                **** INDICATES AN IDEOGRAM WITH VERTICAL ABCISSA )


THE ORIGIN HAS BEEN REPOSITIONED TO  ( 0.0      ,-0.110E 02)   LATCH =  1


THE ORIGIN HAS BEEN REPOSITIONED TO  ( 0.0      , 0.700E 00)   LATCH =  1


        XSIZE      YSIZE    XSF       XMIN      YSF       YMIN       X-AXIS LABEL       Y-AXIS LABEL       GRAPH LABEL
        6.500      4.000  0.0         0.0       0.0       0.0        HORIZONTAL AXIS    VERTICAL AXIS      THE ORIGIN IS CON-
 ACTUAL SCALE FACTORS      0.400E 02  0.0       0.800E 02 0.0

   DATA NO OF  SYM OPT
   SET  POINTS BOL ION        (X1,Y1)                (X2,Y2)                (X3,Y3)                (X4,Y4)          DATA SET LABEL
     1   256     3   7  0.100E 01, 0.500E 01  0.200E 01, 0.100E 02  0.300E 01, 0.110E 02  0.400E 01, 0.150E 02 TROLLED BY THE USER


THE ORIGIN HAS BEEN REPOSITIONED TO  ( 0.0      , 0.500E 01)   LATCH =  2


        XSIZE      YSIZE    XSF       XMIN      YSF       YMIN       X-AXIS LABEL       Y-AXIS LABEL       GRAPH LABEL
        6.500      4.000  0.0         0.0       0.0       0.0        HORIZONTAL AXIS    Y-COORDINATES      SMOOTHED CURVE WITH
 ACTUAL SCALE FACTORS      0.400E 02  0.0       0.800E 02 0.0

   DATA NO OF  SYM OPT
   SET  POINTS BOL ION        (X1,Y1)                (X2,Y2)                (X3,Y3)                (X4,Y4)          DATA SET LABEL
     1   256     0 -30  0.100E 01, 0.500E 01  0.200E 01, 0.100E 02  0.300E 01, 0.110E 02  0.400E 01, 0.150E 02 ZERO NEGLECT OPTION
LOCATION   0.100E 01, 0.100E 01  LABEL - MAXIMUM SMOOTHING
```

Figure 11. Parameter Dump for Test Program Number Two

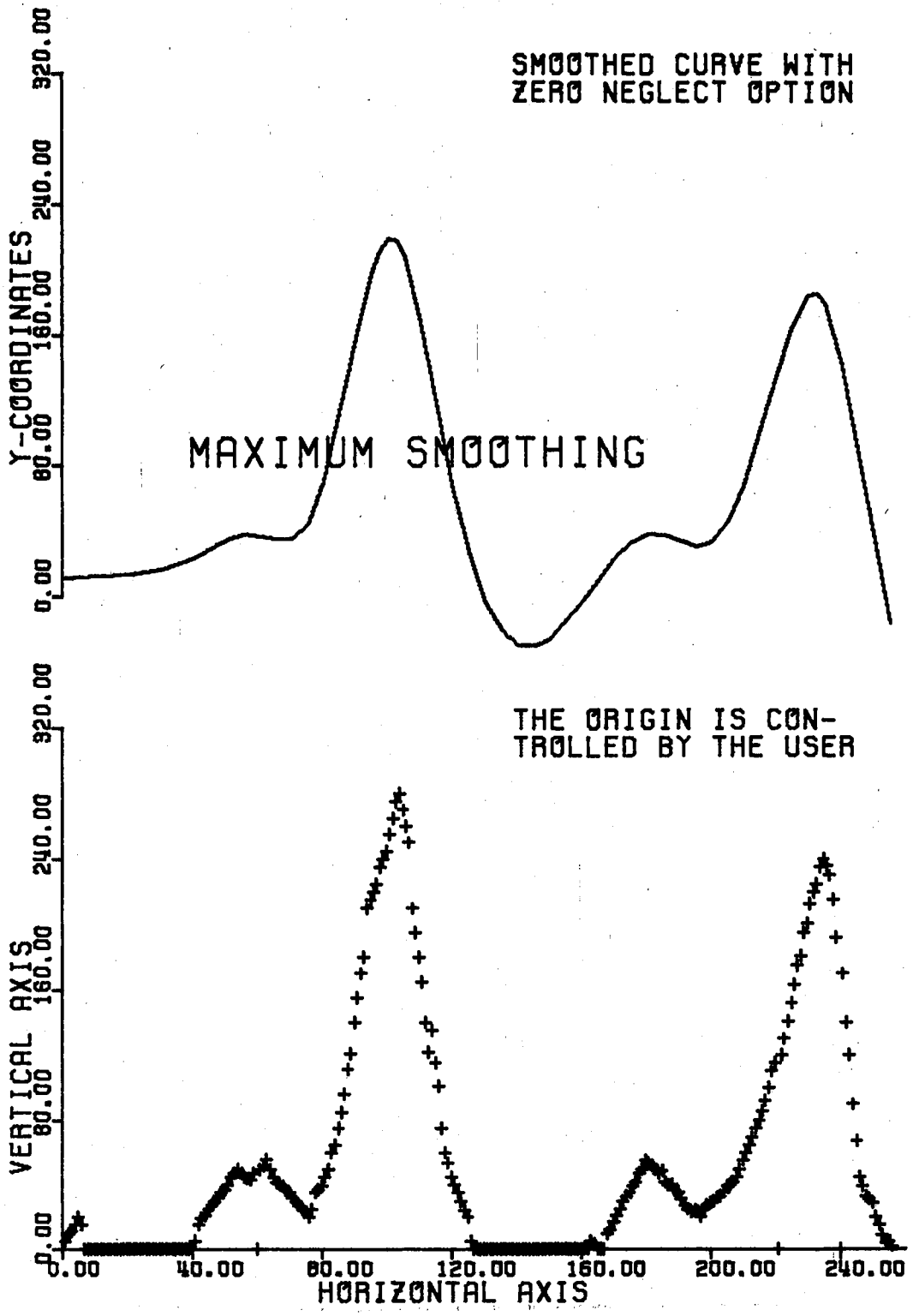


Figure 12. Plot Produced by Test Program Number Two

```
FORTRAN IV G LEVEL 19          MAIN          DATE = 72122          14/39/58          PAGE 0001

      C     TEST PROGRAM NUMBER 3
      C
      C     AUTHOR. JOHN FORREST
      C     DATE.   MAY, 1972
      C
      C          THIS PROGRAM ILLUSTRATES THE USE OF KRPLT TO BUILD AND PLOT
      C     HISTOGRAMS AND IDEOGRAMS.  A HISTOGRAM AND AN IDEOGRAM ARE PLOTTED
      C     WITH VERTICAL AND HORIZONTAL ABCISSI.
      C
0001        DIMENSION  ABCHST(42),CNTHST(42),ABCIDE(102),CNTIDE(102),
     1                 XL(5),YL(5),GL(5),DL(5)
0002        IN = 5
      C     INITIALIZE THE ARRAYS FOR THE HISTOGRAM AND IDEOGRAM
0003        CALL KRPLTS (0, ABCHST, CNTHST, 40, 0., 400.)
0004        CALL KRPLTS (0, ABCIDE, CNTIDE, 100, 0., 400.)
      C     READ IN THE LABELS
0005        READ(IN,4) XL,YL,GL,DL
0006      4 FORMAT( 20A4)
      C     INPUT THE EVENTS AND THEIR ERRORS AND BUILD THE ARRAYS.
      C     THE ERRORS ARE NOT USED FOR THE HISTOGRAM.
0007        DO 5 I=1,20
0008        READ(IN,10) EVENT,STDEV
CC09     10 FORMAT(2F8.0)
0010        CALL KRPLTS (1, ABCIDE, CNTIDE, EVENT, STDEV, 0)
0011      5 CALL KRPLTS (2, ABCHST, CNTHST, EVENT, 0, 0)
      C     NORMALIZE THE AREA UNDER THE IDEOGRAM TO THE AREA OF THE
      C     HISTOGRAM.
0012        DO 6 I=1,102
0013      6 CNTIDE(I) = CNTIDE(I)*10.
      C     PLOT THE HISTOGRAM AND IDEOGRAM WITH HORIZONTAL ABCISSA.
0014        CALL KRPLT (-1,ABCIDE,CNTIDE,5.0,8.0,0,0,0,0,XL,YL,GL,DL)
0015        CALL KRPLTS(-2,ABCHST,CNTHST,0,0,0)
      C     PLOT THE HISTOGRAM AND IDEOGRAM WITH VERTICAL ABCISSA.
0016        CALL KRPLT (-3,ABCIDE,CNTIDE,8.0,5.0,0,0,0,0,XL,YL,GL,DL)
0017        CALL KRPLTS(-4,ABCHST,CNTHST,0,0,0)
0018        STOP
CC19        END
```

Figure 13. Test Program Number Three

```
EASYPLOT PARAMETER DUMP....CALCOMP PLOTTER VERSION    (J. R. FORREST, COMPUTING AND INFORMATION SCIENCES DEPT., O.S.U.)

EACH GRAPH IS REPRESENTED BY ONE BLOCK OF DATA WHICH INCLUDES
          1.  A SUMMARY OF PARAMETERS PASSED VIA CALL LISTS.
          2.  ACTUAL SCALE FACTORS USED IN THE PLOT.
          3.  FIRST FOUR POINTS OF EACH DATA SET.
              ( *    INDICATES A HISTOGRAM WITH HORIZONTAL ABCISSA,
                **   INDICATES A HISTOGRAM WITH VERTICAL ABCISSA,
                ***  INDICATES AN IDEOGRAM WITH HORIZUNTAL ABCISSA,
                **** INDICATES AN IDEOGRAM WITH VERTICAL ABCISSA )


        XSIZE      YSIZE    XSF        XMIN       YSF        YMIN       X-AXIS LABEL        Y-AXIS LABEL        GRAPH LABEL
        5.000      8.000    0.0        0.0        0.0        0.0        INTERVAL AXIS       COUNT AXIS              HISTOGRAM AND
 ACTUAL SCALE FACTORS       0.800E 02  0.0        0.400E 00  0.0

    DATA NO OF  SYM OPT
    SET  POINTS BOL ION        (X1,Y1)                 (X2,Y2)                 (X3,Y3)                 (X4,Y4)          DATA SET LABEL
***   1   101     0   0  0.0        , 0.333E 00  0.400E 01, 0.355E 00  0.800E 01, 0.385E 00  0.120E 02, 0.431E 00        IDEOGRAM TEST
*     2    41     0   0  0.0        , 0.0        0.100E 02, 0.0        0.200E 02, 0.100E 01  0.300E 02, 0.0


        XSIZE      YSIZE    XSF        XMIN       YSF        YMIN       X-AXIS LABEL        Y-AXIS LABEL        GRAPH LABEL
        8.000      5.000    0.0        0.0        0.0        0.0        INTERVAL AXIS       COUNT AXIS              HISTOGRAM AND
 ACTLAL SCALE FACTORS       0.400E 00  0.0        0.800E 02  0.0

    DATA NO OF  SYM OPT
    SET  POINTS BOL ION        (X1,Y1)                 (X2,Y2)                 (X3,Y3)                 (X4,Y4)          DATA SET LABEL
****  1   101     0   0  0.333E 00, 0.0        0.355E 00, 0.400E 01  0.385E 00, 0.800E 01  0.431E 00, 0.120E 02        IDEOGRAM TEST
**    2    41     0   0  0.0        , 0.0        0.0        , 0.100E 02  0.100E 01, 0.200E 02  0.0        , 0.300E 02
```

Figure 14. Parameter Dump for Test Program Number Three

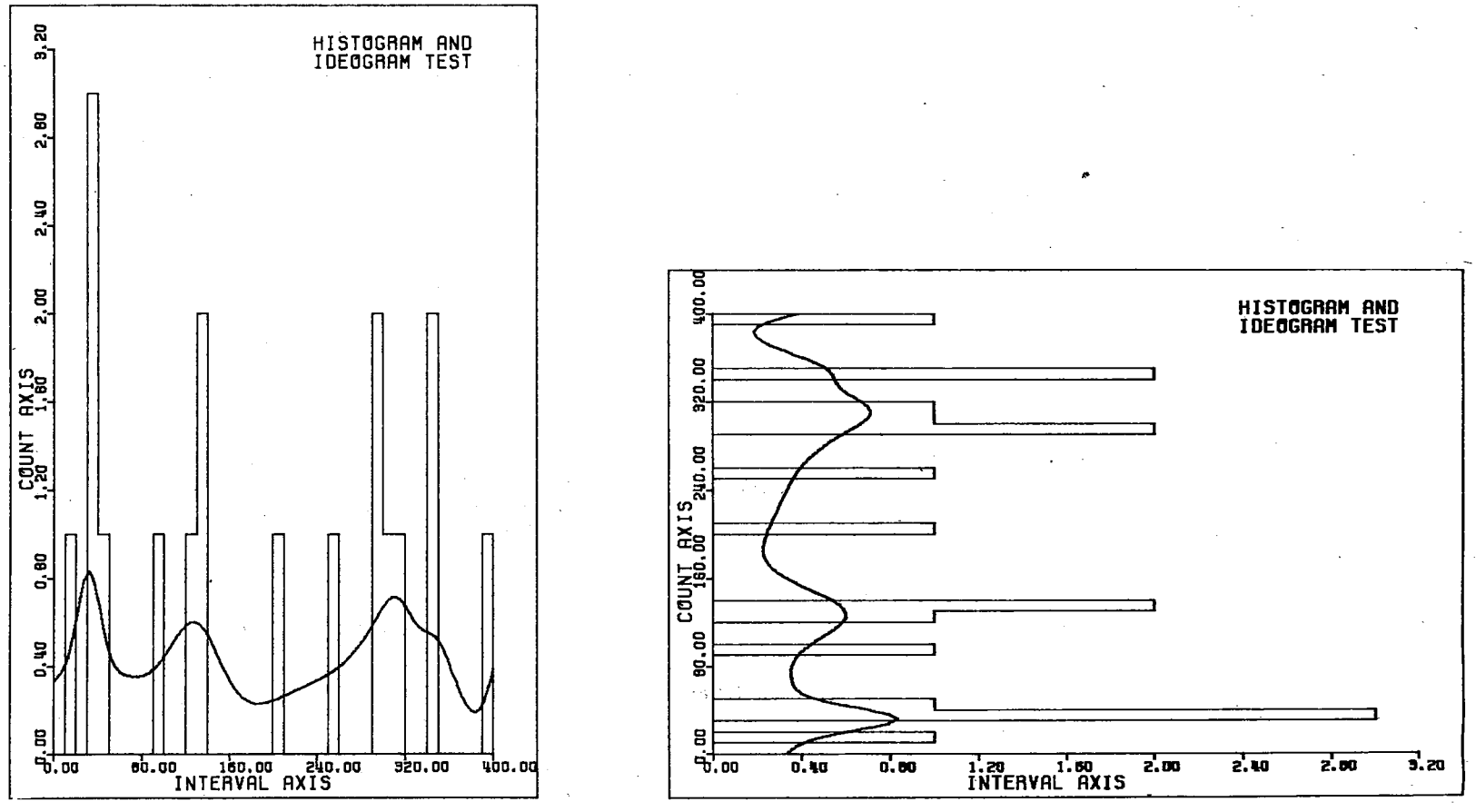


Figure 15. Plot Produced by Test Program Number Three

```
      C     TEST PROGRAM NUMBER 4
      C
      C     AUTHOR.  JOHN FORREST
      C     DATE.    MAY, 1972
      C
      C          THIS PROGRAM TEST THE SMOOTHING OPTION OF GRAPH WITH
      C     STANDARD DEVIATIONS GIVEN FOR EACH POINT.  THE DATA POINTS
      C     COME FROM A SINE FUNCTION WITH NORMALLY DISTRIBUTED ERRORS ADDED.
0001        DIMENSION X(101),Y(100),XL(5),YL(5),GL(5),DL(5),STD(100)
0002        IRAN = 9876543
0003        IN = 5
      C     READ IN LABELING INFORMATION
0004        READ(IN,10) XL,YL,GL,DL
0005     10 FORMAT(20A4)
0006        DX = 0.
      C     CALCULATE THE SINE VALUES AND ADD THE ERRORS. S IS THE STANDARD
      C     DEVIATION
0007        S = .15
0008        DO 1 I=1,70
0009        X(I) = DX
0010        CALL GAUSS (IRAN,S,0.,DY)
0011        STD(I) = S
0012        DX = DX + 0.25
0013      1 Y(I) = SIN(X(I)) + DY
      C     CALL GRAPH TO MAKE THE PLOT
0014        CALL GRAPH ( 70,X,Y,3,31, 9.,6.0,0,0,0,0,0,XL,YL,GL,DL,STD)
0015        STOP
0016        END
```

Figure 16. Test Program Number Four

```
EASYPLOT PARAMETER DUMP....CALCOMP PLOTTER VERSION    (J. R. FORREST, COMPUTING AND INFORMATION SCIENCES DEPT., U.S.U.)

EACH GRAPH IS REPRESENTED BY ONE BLOCK OF DATA WHICH INCLUDES
        1.  A SUMMARY OF PARAMETERS PASSED VIA CALL LISTS.
        2.  ACTUAL SCALE FACTORS USED IN THE PLOT.
        3.  FIRST FOUR POINTS OF EACH DATA SET.
            ( *    INDICATES A HISTOGRAM WITH HORIZONTAL ABCISSA,
              **   INDICATES A HISTOGRAM WITH VERTICAL ABCISSA,
              ***  INDICATES AN IDEOGRAM WITH HORIZONTAL ABCISSA,
              **** INDICATES AN IDEOGRAM WITH VERTICAL ABCISSA )


      XSIZE      YSIZE     XSF        XMIN       YSF        YMIN       X-AXIS LABEL          Y-AXIS LABEL          GRAPH LABEL
      9.000      6.000   0.0          0.0        0.0        0.0        ANGLE X IN RADIANS      SINE(X)           SMOOTHING TEST WITH
ACTUAL SCALE FACTORS      0.200E 01   0.0        0.500E 00 -0.150E 01

  DATA NO OF  SYM OPT
  SET  POINTS BOL ION        (X1,Y1)              (X2,Y2)              (X3,Y3)              (X4,Y4)          DATA SET LABEL
    1    70     3  31  0.0       , 0.565E-01  0.250E 00, 0.219E 00  0.500E 00, 0.695E 00  0.750E 00, 0.618E 00 STANDARD DEVIATIONS
```

Figure 17. Parameter Dump for Test Program Number Four

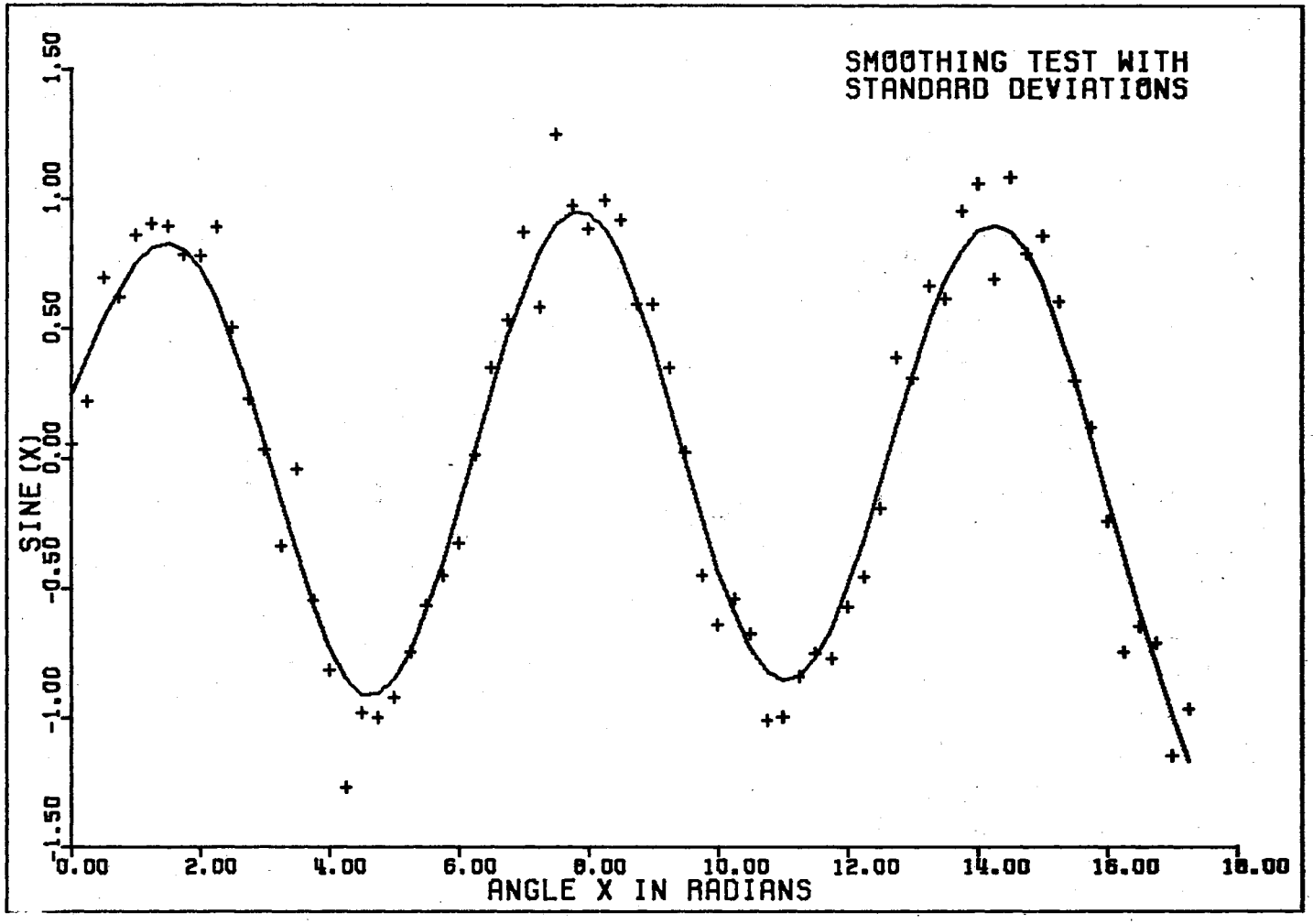


Figure 18. Plot Produced by Test Program Number Four

APPENDIX D

FLOW CHARTS

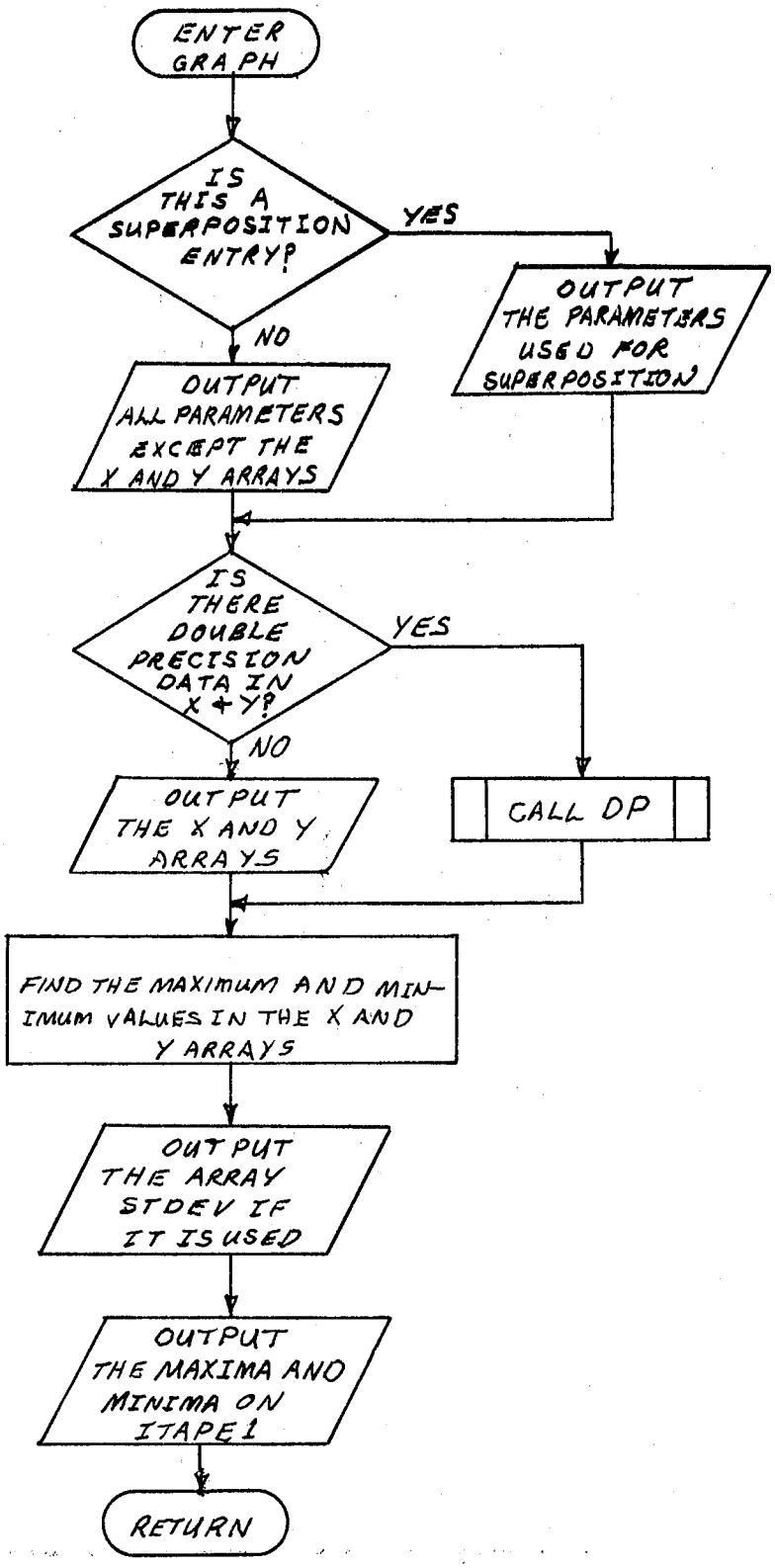


Figure 19. Flow Chart for Subroutine GRAPH

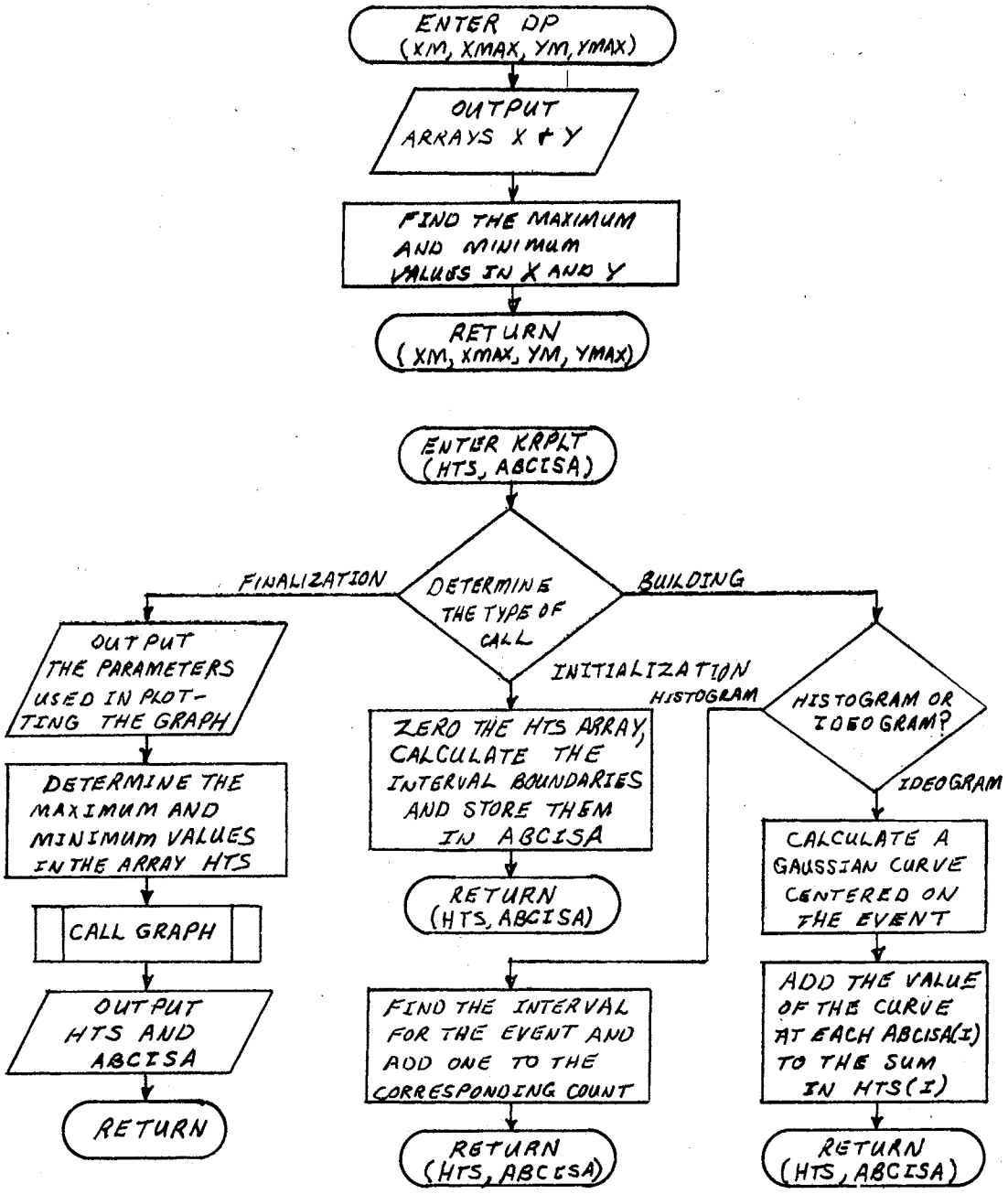


Figure 20. Flow Charts for Subroutines DP and KRPLT

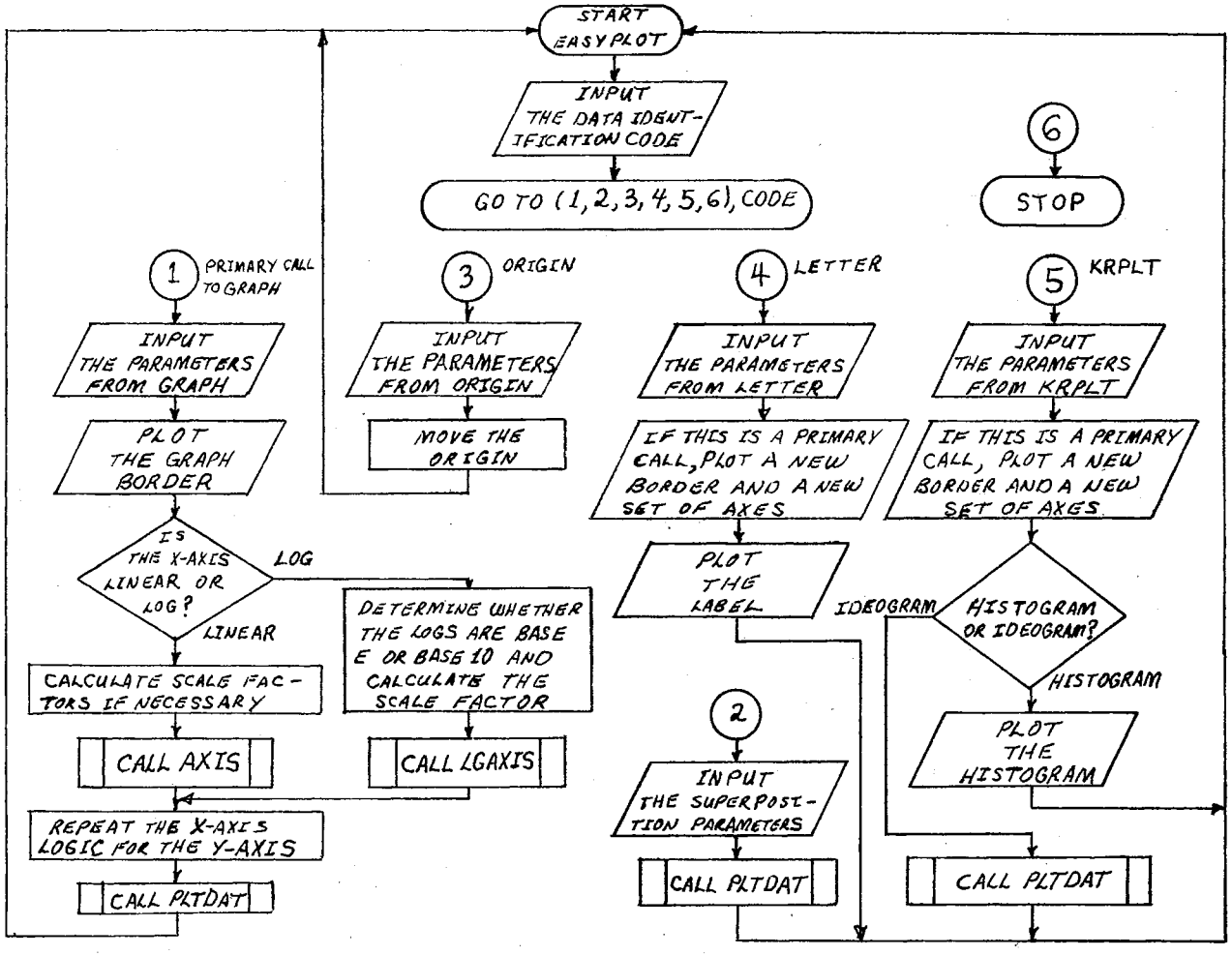


Figure 21. Flow Chart for Easyplot Main Program

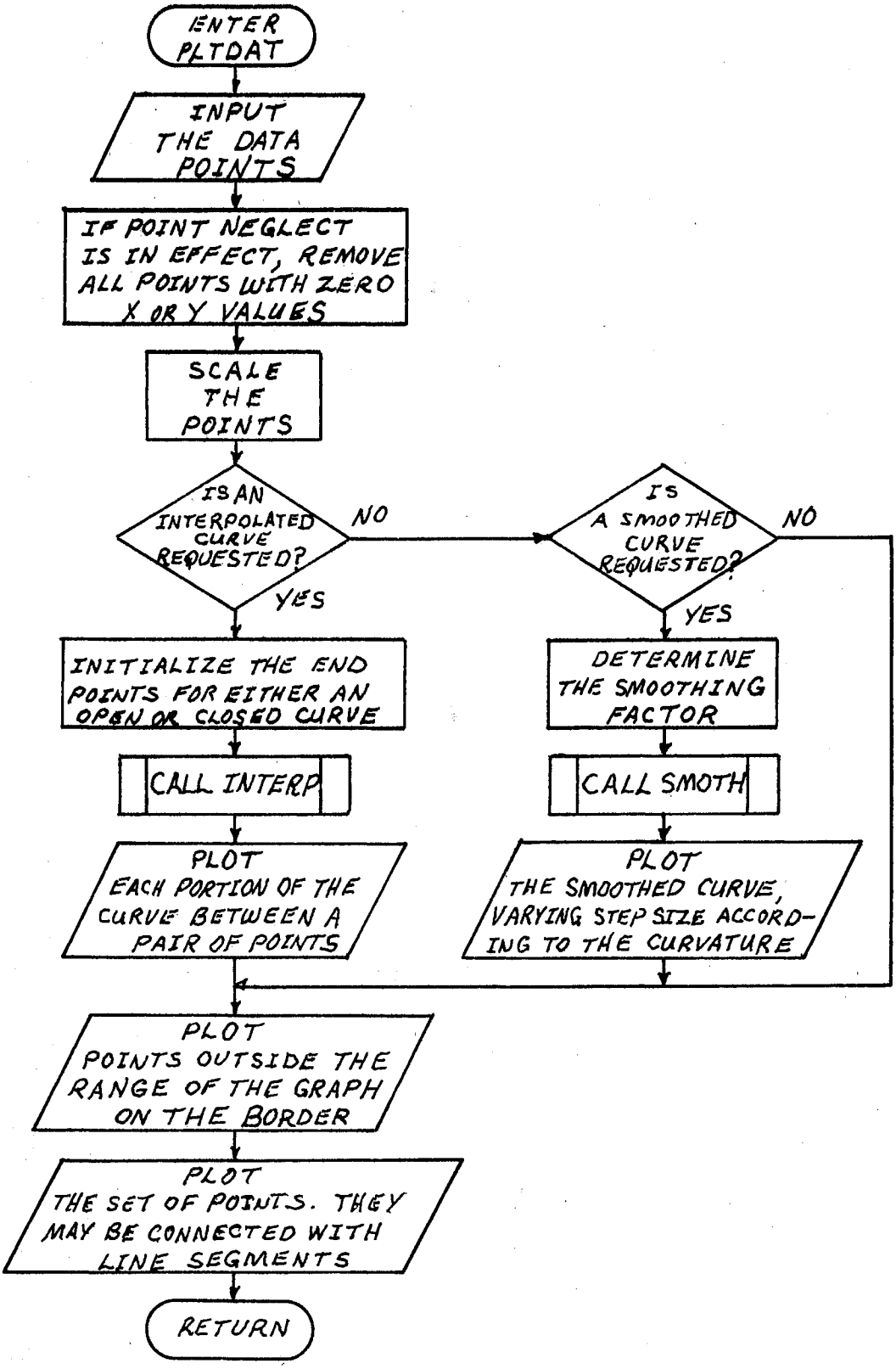


Figure 22. Flow Chart for Subroutine PLTDAT

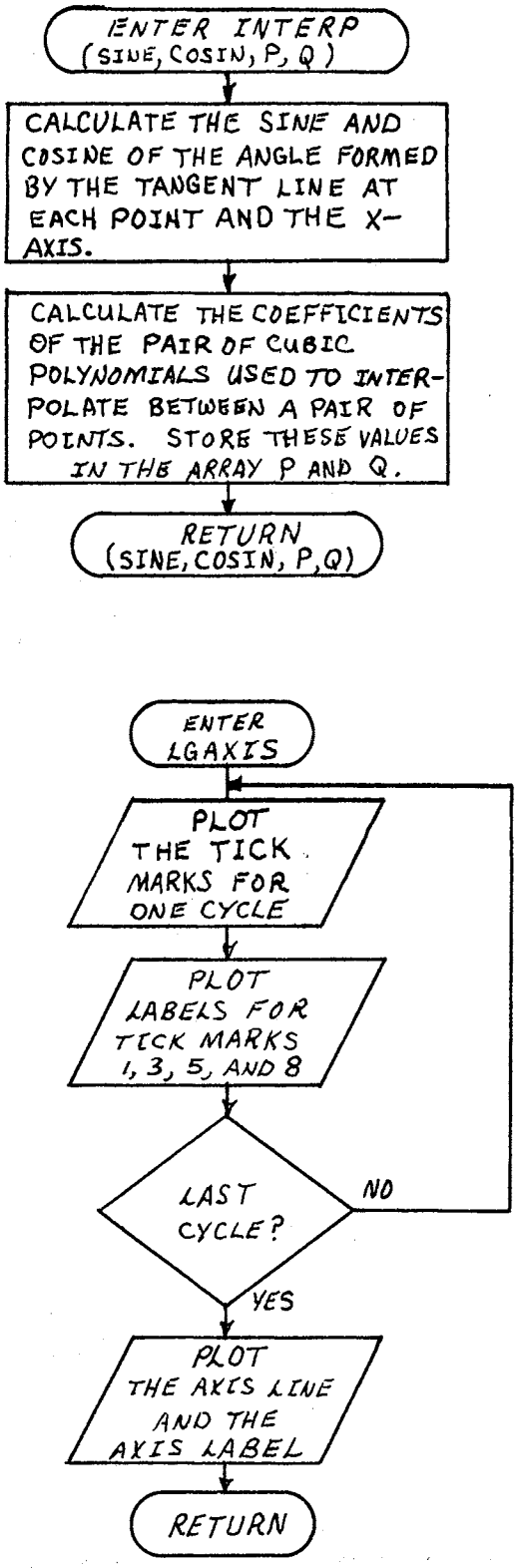


Figure 23. Flow Charts for Subroutines INTERP and LGAXIS

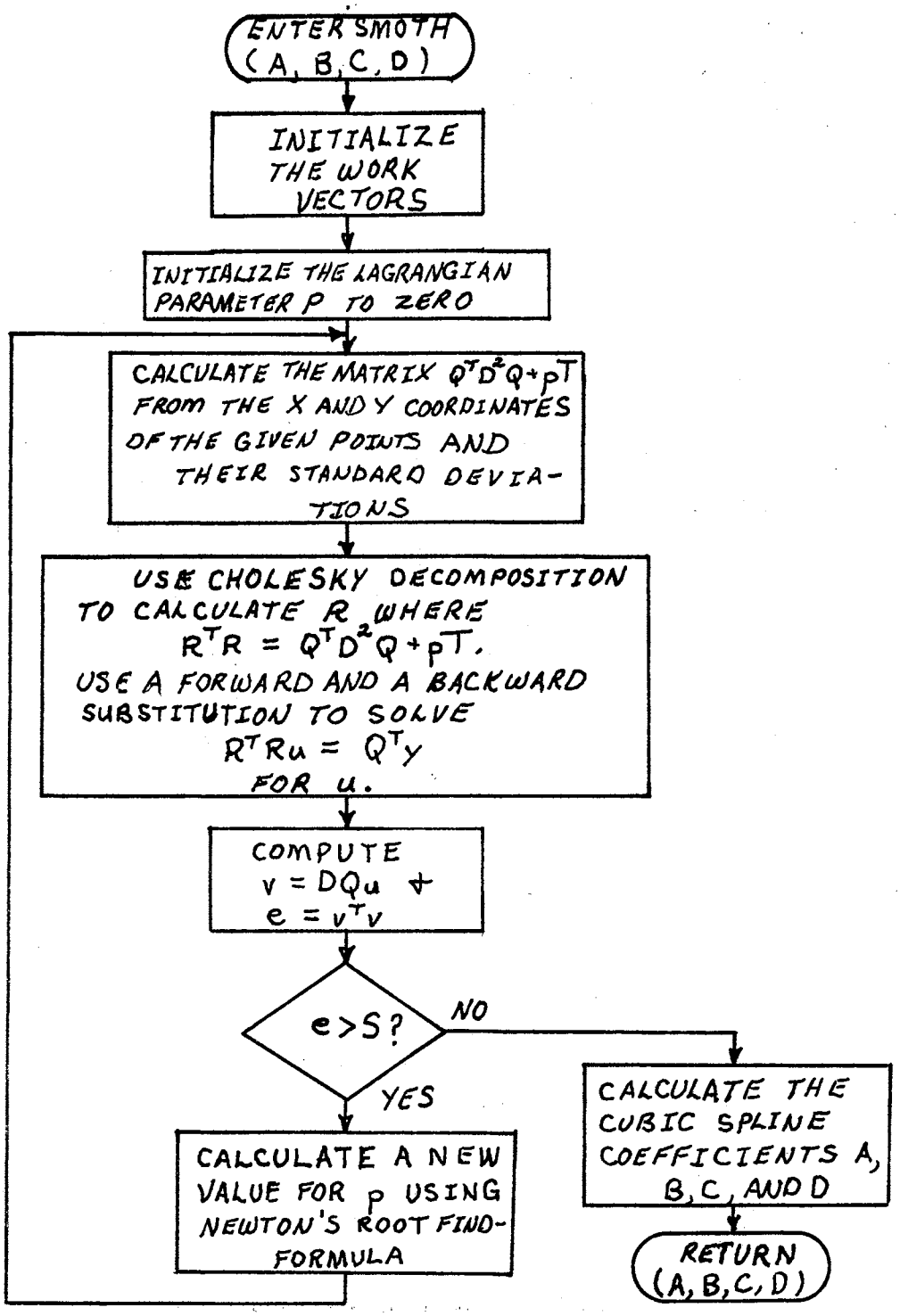


Figure 24. Flow Chart for Subroutine SMOTH

VITA

John Raymond Forrest

Candidate for the Degree of

Master of Science

Thesis: EASYPLOT, A HIGH LEVEL PLOTTING SYSTEM

Major Field: Computing and Information Sciences

Biographical:

Personal Data: Born in Corpus Christi, Texas, February 9, 1946, the son of Mr. and Mrs. Robert C. Forrest.

Education: Graduated from Central High School, Muskogee, Oklahoma, in May, 1964; received a Bachelor of Science degree in Physics from Oklahoma State University in May, 1968; completed requirements for the Master of Science degree at Oklahoma State University in July, 1972.

Professional Societies: Association for Computing Machinery and Sigma Pi Sigma professional physics society.