N72-15179

NASA TECHNICAL MEMORANDUM

NASA TM X-2465

NASA TM X-2465

COMPUTER PROGRAMS FOR PLOTTING CURVES WITH VARIOUS DASHED-LINE SEQUENCES

by Robert N. Desmarais and Robert M. Bennett

Langley Research Center
Hampton, Va. 23365

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1972

1. Report No. NASA TM X-2465	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle COMPUTER PROGRAMS FOR PLOTTING CURVES WITH VARIOUS DASHED-LINE SEQUENCES		5. Report Date February 1972 6. Performing Organization Code
7. Author(s) Robert N. Desmarais and Robert M. Bennett		8. Performing Organization Report No. L-8064
9. Performing Organization Name and Address NASA Langley Research Center Hampton, Va. 23365		10. Work Unit No. 136-14-02-02 11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		13. Type of Report and Period Covered Technical Memorandum 14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

Two FORTRAN-callable subprograms have been written to draw a smooth curve through a set of input points as a solid line or as a general sequence of long and short dashes. Subroutine LINSEQ draws conventional curves whereas subroutine CONSEQ draws smooth closed curves (contours). The subprograms are based on an approximate calculation of the arc length along the curve and spline interpolation along the arc length. Options are provided for smoothing of the input data and for offsetting the plotted curve from the input data points. The method of calculation of the arc length and the generation of the line sequence are described. Usage descriptions of the main subprograms, sample calling programs illustrating the various features of the subprograms, and sample plots are given. The subroutines should be readily adaptable to almost any computer-driven incremental plotter.

17. Key Words (Suggested by Author(s)) Plotting Computer graphics Spline functions	18. Distribution Statement Unclassified – Unlimited

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 67	22. Price* $3.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

CONTENTS

Page

COMPUTER PROGRAMS FOR PLOTTING CURVES WITH VARIOUS DASHED-LINE SEQUENCES

By Robert N. Desmarais and Robert M. Bennett
Langley Research Center

SUMMARY

Two FORTRAN-callable subprograms have been written to draw a smooth curve through a set of input points as a solid line or as a general sequence of long and short dashes. Subroutine LINSEQ draws conventional curves whereas subroutine CONSEQ draws smooth closed curves (contours). The subprograms are based on an approximate calculation of the arc length along the curve and spline interpolation along the arc length. Options are provided for smoothing of the input data and for offsetting the plotted curve from the input data points. The method of calculation of the arc length and the generation of the line sequence are described. Usage descriptions of the main subprograms, sample calling programs illustrating the various features of the subprograms, and sample plots are given. The subroutines should be readily adaptable to almost any computer-driven incremental plotter.

INTRODUCTION

When plotting several curves on one graph, a unique sequence of long and short dashes is commonly used to identify each curve. When such curves are hand-drawn, a reasonable degree of skill and judgment is required by the draftsman to produce a result of good quality. For example, some effort should be made to see that the sequence is properly terminated at the end of the line and to maintain constant dash lengths along the arc of the curve. The modern digital computer has provided the capability of generating such a large volume of results that quality hand plotting and fairing of curves is prohibitively slow. Thus, automatic plotting hardware and software have been developed. Conventionally, however, only solid curves are generated by connecting closely spaced points with straight lines. Apparently, little work has been done in the past on automatic computer generation of dashed-line sequences analogous to hand plotting, partly because of the complication of maintaining a constant dash length along the arc of the curve.

The purpose of this paper is to describe a set of computer subprograms (called LINSEQ and CONSEQ) to generate dashed-line sequences for an automatic plotter. Subprogram LINSEQ draws a conventional (or open) curve which can be a multiple-valued function of either coordinate X or Y. Subprogram CONSEQ draws a curve that closes smoothly with continuous deflections and slopes such as would be used for contour lines. An option is provided for the smoothing of the input data such as is sometimes desired when plotting experimental data points. An option is also provided for offsetting the curve from the input data so that heavy curves can be drawn with a fine pen to reduce smearing of ink or so that nearly coincident curves with different line sequences can be distinguished.

The method of calculating the arc length, the method of generating the dashed-line sequence, and sample calling programs with input data and output plots are presented. Listings of each subprogram and usage descriptions of the principal subprograms are given. The spline-interpolation subprograms used for curve fairing can be used independently of LINSEQ and CONSEQ, and their usage descriptions are also given. The programs were written in FORTRAN for use on the Control Data series 6000 computer systems using the Langley Research Center versions of the RUN compiler and SCOPE 3.0 operating system. However, they should be adaptable to other computer-driven incremental plotters without undue difficulty.

This paper is intended to serve both as a user's guide and as a programer's manual. Sufficient information for the user is contained in the descriptions of subprograms LINSEQ and CONSEQ in appendix A and in the examples in appendix B. The detailed descriptions of the algorithms used, contained in the body of the text, and the program listings in appendix C can be skipped by those not interested in modifying the programs or converting them so they may be used with other equipment.

METHOD

In order to fair a dashed curve through a set of data points, it is necessary to have some means of interpolating the data to generate the plot vector file used by the incremental plotter. It is also necessary to incorporate into the interpolation algorithm some scheme for insuring that the dashes are equally spaced along the curve. Both of these objectives have been realized in the programs described herein by basing the interpolation on a pair of cubic-spline interpolation functions, with the approximate arc length from the first data point used as the independent variable and with X and Y as separate dependent variables.

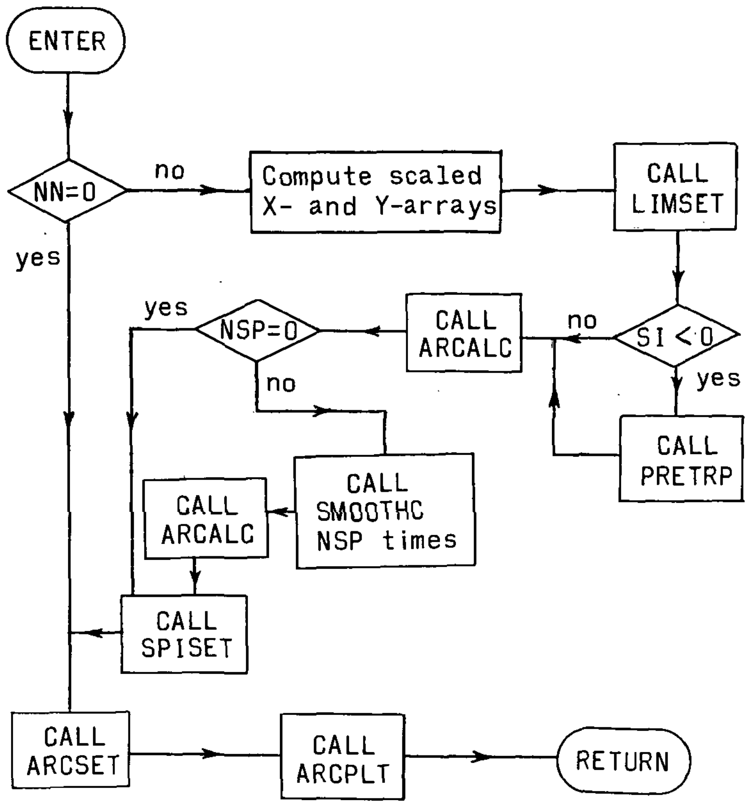
Two different curve-fairing subprograms are available. Subprogram LINSEQ fairs a set of N data points with a smooth curve passing through the points starting at (X_1,Y_1) and terminating at (X_N,Y_N) and having zero curvature at the two ends. Subprogram CONSEQ constructs a closed curve starting and ending at (X_1,Y_1) with no discontinuities in slope or curvature. The only essential difference between the two subprograms is that in LINSEQ the interpolation is performed by using natural cubic splines (i.e., splines for which the curvature vanishes at the two ends) and in CONSEQ the interpolation is performed by using periodic cubic splines (i.e., splines for which the slopes, deflections, and curvatures are matched at the two ends). Because LINSEQ and CONSEQ are so similar, only LINSEQ will be described in detail.

Organization of Subprogram LINSEQ

The curve-fairing subprogram LINSEQ is organized in such a manner that it primarily scales the input data, configures the basic line sequence, and calls several subprograms to perform specific tasks. The tasks performed by these subprograms are listed below and will be described in detail in succeeding portions of this report:

Subprogram	Task
PRETRP	Implements preinterpolation option
ARCALC	Computes arc length
SMOOTHC	Smooths input data if smoothing option is selected
SPISET	Initializes spline interpolation
ARCPLT	Generates a single dash of the curve
CHLSKYS	Solves simultaneous equations
SPIFUN	Spline interpolation (ordinates)
SPIDER	Spline interpolation (derivatives)
LIMPLT	Commands pen motion if within user-prescribed limits
CALPLT	Causes pen motion with pen up or down (This subprogram is not part of the LINSEQ package; it is the installation library routine that interfaces the user program with the plotting devices.)

The logical flow of subprogram LINSEQ is illustrated by the flow chart below:



The meanings of the tested parameters NN, SI, and NSP (further described in appendix A) are as follows:

NN	number of input points to be plotted (If NN = 0, it is assumed that the "no" branch was executed on a previous call.)
SI	approximate interval between interpolated points on the plotted curve in inches (If SI is set to a negative value, a preinterpolation to 2*NN-1 points is requested.)
NSP	number of iterated smoothing passes applied to the input data points

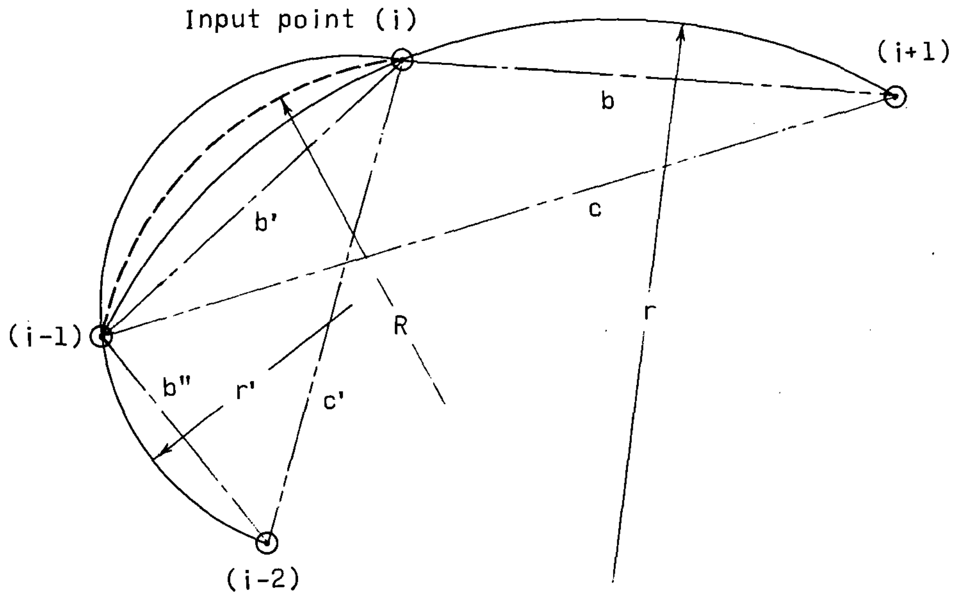
Note that LIMSET and ARCSET are initialization entry points of subroutines LIMPLT and ARCPLT, respectively. Also SPISET initializes certain parameters subsequently used by the spline interpolation functions SPIFUN and SPIDER called by ARCPLT.

Arc-Length Calculation

A set of data points (X_i, Y_i), with $i = 1, 2, \ldots, N$, is given. A third array S_i, the array of arc lengths, has to be computed; S_1 is set to zero and succeeding elements of the array are computed from the equation

$$S_i = S_{i-1} + \Delta S_i \qquad (i = 2, 3, \ldots, N)$$

where ΔS_i is the computed length of the arc of the curve connecting data point (X_{i-1}, Y_{i-1}), which is labeled (i-1) in the sketch below, to data point (X_i, Y_i), which is labeled (i) in the sketch below. This length ΔS_i is approximated by the arc of a circle of radius R connecting the two points. The radius R is obtained by averaging the curvature of the left and right circumscribed circles with radii r' and r, respectively, as shown in the following sketch:



Since any radius in this sketch may be infinite, it is more convenient to work with half-curvatures (h, h', and h_{av}). Let $h = \frac{1}{2r}$, $h' = \frac{1}{2r'}$, and $h_{av} = \frac{1}{2R}$. Then,

$$h_{av} = \frac{h' + h}{2}$$

and

$$\Delta S_i = \frac{\sin^{-1}(b' h_{av})}{h_{av}}$$

Since it is considered desirable to underestimate the arc length ΔS_i, the arcsine function is replaced by its two-term Taylor series; the resulting equation is

$$\Delta S_i = b'\left[1 + \frac{\left(b'h_{av}\right)^2}{6}\right]$$

In the sketch on the preceding page, let $A/2$ be the area of the triangle (i-1, i, i+1) and let $A'/2$ be the area of the triangle (i-2, i-1, i). Then, $h = \frac{A}{b'bc}$ and $h' = \frac{A'}{b''b'c}$ where

$$A = \left(X_i - X_{i-1}\right)\left(Y_{i+1} - Y_{i-1}\right) - \left(X_{i+1} - X_{i-1}\right)\left(Y_i - Y_{i-1}\right)$$

$$b = \sqrt{\left(X_{i+1} - X_i\right)^2 + \left(Y_{i+1} - Y_i\right)^2}$$

and

$$c = \sqrt{\left(X_{i+1} - X_{i-1}\right)^2 + \left(Y_{i+1} - Y_{i-1}\right)^2}$$

The terms A', b', and c' are computed the same as A, b, and c except that all subscripts are reduced by 1. Also, b'' is computed the same as b except that subscripts are reduced by 2.

This procedure for computing arc lengths to data points is implemented in subroutine ARCALC called by LINSEQ and by CONSEQ. The parameter IFLAG is used by LINSEQ and CONSEQ to determine the manner in which the ends are treated. For LINSEQ, the curvature is set to zero at the end points $\left(X_1,Y_1\right)$ and $\left(X_N,Y_N\right)$. For CONSEQ, there are no ends because the data are assumed to define a continuous closed curve.

Spline Interpolation

After the array S_i, with $i = 1, 2, \ldots, N$, has been constructed, the set of input data points $\left(X_i,Y_i\right)$, with $i = 1, 2, \ldots, N$, is interpolated by a pair of parametric interpolation functions

$$X = X(S)$$

and

$$Y = Y(S)$$

where S, the independent variable, is the arc length along the curve. The interpolation functions used are cubic splines (e.g., refs. 1 and 2). A cubic spline is a set of piecewise cubic polynomials joined at each data point, such that ordinate, slope, and second derivatives match where the polynomials are joined. Two additional conditions are required to define a cubic spline uniquely. In the spline subprogram called by LINSEQ, the additional conditions are obtained by setting the two end-point second derivatives to zero. Such a spline is called a natural cubic spline. In the spline subprogram called by CONSEQ, the additional conditions are obtained by causing the end-point first and second derivatives to match. Such a spline is called a periodic cubic spline (provided the ordinates also match as they do in this case).

The spline interpolation functions are computed by using a compact scheme based on an overrelaxation solution of a system of linear equations. This algorithm (described in detail in ref. 1) furnishes the second derivative at each data point. Nonconvergence of the overrelaxation solution would result in discontinuities in the first derivatives. Since such discontinuities would be unacceptable, the two spline initialization subroutines SPISET and SPISETP use the computed second derivatives to compute left and right first derivatives at the data points. Then the averaged first derivatives are passed to the spline evaluation subprograms SPIFUN and SPIFUNP. Now nonconvergence causes discontinuities in the second derivatives, but these discontinuities are so small that they cannot be detected in the faired curves generated by LINSEQ and CONSEQ.

Six subprograms are used to implement the spline interpolations. Subprograms SPISET, SPIFUN, and SPIDER are called by LINSEQ. Subprograms SPISETP, SPIFUNP, and SPIDERP are called by CONSEQ. These subprograms can also be used, independently of LINSEQ and CONSEQ, as general-purpose interpolation routines.

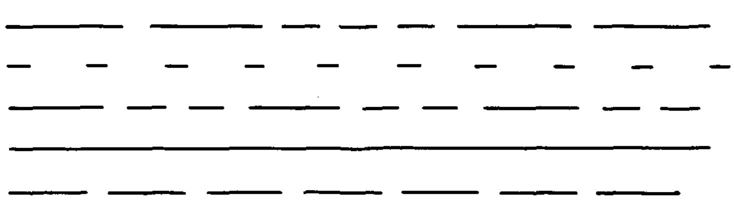
Dash Configuration and Offset

Six user-furnished parameters are used to configure a sequence of long and short dashes. These parameters and their meanings are listed below:

SI	approximate interpolation interval in inches (The actual interpolation interval DS is computed by the program.)
L1	number of intervals of length DS in the gap between dashes
L2	number of long dashes per cycle (A cycle consists of a single string of long dashes followed by a single string of short dashes.)
L3	number of intervals of length DS in each long dash

L4	number of short dashes per cycle
L5	number of intervals of length DS in each short dash

The dashed-line sequences below illustrate the use of the five line-sequence parameters (L1, L2, L3, L4, L5). For each line sequence, the interpolation interval SI is approximately 0.1 inch.



Note that if only a single-length dash is to be used, L4 (not L2) should be set to zero. Setting L1 to zero causes a solid line to be drawn.

The actual interpolation interval along the curve DS is computed in such a manner that the curve from point (X_1,Y_1) to (X_N,Y_N) starts and ends with a series of long dashes if LINSEQ is used. If CONSEQ is used, DS is computed so that there are an integral number of cycles starting at (X_1,Y_1).

The line-sequence parameter L1, L3, or L5, as appropriate, is passed to subroutine ARCPLT which then draws a single dash of the prescribed length. The line-sequence parameters L2 and L4 are used as loop delimiters in LINSEQ.

The offset parameter H is used to draw parallel curves. For example, if the pen diameter in inches was D and one wanted to draw a curve of width 5D, one could call LINSEQ five times with H set to -2D, -D, 0, D, and 2D. Whenever multiple calls to LINSEQ are used with the same data, the parameter NN (number of data points) should be set to zero for all but the first call. This procedure causes the first part of LINSEQ to be bypassed. A positive value of H causes the curve to be offset to the right. Occasionally, it is desirable to set H to ±D/2 to distinguish curves that coincide over a significant fraction of their lengths.

Use of Frame Limits and Plotter Conversion

The pen-motion instructions issued by LINSEQ or CONSEQ are transmitted through a short subprogram called LIMPLT. Subprogram LIMPLT scans the pen-motion instruction and transmits it only if the resulting pen motion would be within user-prescribed limits. These limits are prescribed by the parameters XLIM and YLIM in the calling sequence to LINSEQ. The parameters XLIM and YLIM are two word arrays (in the same units as the input X- and Y-arrays) used to delimit the frame within which pen motion is permitted. They are transmitted, in scaled form (inches), from LINSEQ to LIMPLT by a call to the entry point LIMSET.

The use of frame limits in this manner facilitates the plotting of a subinterval of the input data while retaining the influence of the points that are off-scale during the interpolation process. It can also guard somewhat against the pen hitting the mechanical stops on a mechanical plotter during the debug stages of a program.

Subroutine LIMPLT is very short and is the only subroutine of the LINSEQ/CONSEQ package that contains a pen-motion instruction to the plotting device. This feature makes conversion of LINSEQ from one plotter to another very simple. The version of LIMPLT described herein contains the installation-dependent plotting instruction.

```
CALL CALPLT(X,Y,IPLM)
```

which means that the pen will be moved to the point (X,Y) (expressed in inches) with the pen lowered if IPLM = 2 or with the pen raised if IPLM = 3. (The usage of CALPLT is further described in appendix B.) Suppose that one wished to convert this instruction for plotting on a cathode-ray tube using the instructions of the graphics display library for the Xerox Data Systems Sigma series computers. For this device the pen-down instruction is

```
CALL LINE(X,Y,MODE)
```

and the pen-up instruction is

```
CALL BEAM(X,Y,MODE)
```

where

MODE = 0	means long absolute
MODE = 1	means long incremental
MODE = 2	means short absolute
MODE = 3	means short incremental

Then the instruction

```
CALL CALPLT(X,Y,IPLM)
```

should be replaced by the two instructions

```
IF(IPLM.EQ.2) CALL LINE(X,Y,2)
IF(IPLM.EQ.3) CALL BEAM(X,Y,2)
```

Preinterpolation and Smoothing

The preinterpolation and smoothing options provide the user with some control over the manner in which the curve fairing is performed.

The purpose of preinterpolation is to compensate for the fact that LINSEQ uses the arc length as the interpolation abscissa although, in many cases, it is more desirable to use X as the interpolation abscissa. This option is selected by adjoining a minus sign to the interpolation-interval parameter SI. Preinterpolation is performed by adding to the X-array $\left(X_i\right.$, with $i = 1, 2, \ldots, N\left.\right)$ additional $N - 1$ abscissas $X_{i-1/2}$, with $i = 2, 3, \ldots, N$, obtained by averaging adjacent abscissas – that is,

$$X_{i-1/2} = \frac{X_{i-1} + X_i}{2} \qquad (i = 2, 3, \ldots, N)$$

and then computing the corresponding ordinates $Y_{i-1/2}$ by spline interpolating with X as the independent variable. Example 3 of appendix B shows how the quality of the faired curve can be improved by preinterpolation. If the X-array is not monotonically increasing, preinterpolation will be attempted with Y as the independent variable. If neither the X-array nor the Y-array is monotonically increasing, preinterpolation will not be performed; thus, the program itself usually inhibits preinterpolation when it is not appropriate. However, preinterpolation is not appropriate and is not inhibited by the program if the X-array is monotonic but dY/dX is infinite or nearly infinite within the interpolation interval. This behavior is illustrated by Example 4 of appendix B. The preinterpolation option is not available in the closed-curve subprogram CONSEQ.

The smoothing option is intended for use when the input data have random errors such as those resulting from a measurement process. The option is selected by assigning a value greater than zero to the parameter NSP (number of iterations) of subprogram LINSEQ. The value of NSP defines the number of iterations of the smoothing procedure in which a locally centered, five-point least-squares cubic is used as the smoothing formula for X and Y as functions of the arc length S. That is, at each point $\left(X_i, Y_i\right)$, the values are replaced by computed values

$$\overline{X}_i = f(S_i)$$

and

$$\overline{Y}_i = g(S_i)$$

where $i = 1, 2, \ldots, N$.

For example, the X-smoothing function $f(S_i) = a_0 + a_1S_i + a_2S_i^2 + a_3S_i^3$ is a cubic polynomial with the coefficients $(a_0,\ a_1,\ a_2,$ and $a_3)$ chosen to minimize the sum

$$\sum_{j=i-2}^{i+2} w_j\left[f(S_j) - X_j\right]^2$$

The weights w_j, which serve to localize the cubic, are taken to be

$$w_j = \frac{1}{l^2 + (S_j - S_i)^2}$$

The reference length l is equal to $K\,\overline{\Delta S}$ where K is the iteration number of smoothing passes 1 to NSP and where $\overline{\Delta S} = \dfrac{S_N - S_1}{N}$ represents the average value of the arc distance between consecutive data points. Note that if the length l is small, the cubic is strongly localized and there is very little smoothing. The Y-smoothing functions $g(S)$ are evaluated in the same manner. At the two ends, the five points are taken to be the five closest points rather than the ith point and two others on each side. The smoothing option is available both with LINSEQ and CONSEQ. The amount of smoothing is controlled by the number of iterations NSP. If NSP = 1, there will be very little smoothing. The effect of letting NSP be very large is to cause the interpolated curve to resemble a parametric cubic over its whole range. Example 5 of appendix B illustrates the effect of smoothing for some particularly rough input data.

The preinterpolation and smoothing options partially cancel each other and are not normally both used for the same curve.

CONCLUDING REMARKS

A set of computer subprograms (called LINSEQ and CONSEQ) to generate dashed-line sequences for an automatic plotter are described, and their usage is illustrated with examples. Subprogram LINSEQ draws a conventional (or open) curve which can be a multiple-valued function of either coordinate X or Y, whereas subprogram CONSEQ draws a curve that closes smoothly such as would be used for contour lines. The options provided for smoothing of the input data and for offsetting the drawn curve from the input data are also described. In addition, the method of calculating the arc length, the method of generating the dashed-line sequence, and the sample calling programs are presented. The subprograms should be adaptable, without undue difficulty, to computer-driven plotters other than the one used.

Langley Research Center,
National Aeronautics and Space Administration,
Hampton, Va., December 21, 1971.

APPENDIX A

SUBPROGRAM DESCRIPTIONS

This appendix contains usage descriptions of subprograms LINSEQ and CONSEQ, the plotting subprograms that are the subject of this paper. It also contains descriptions of the other subprograms called by LINSEQ and CONSEQ that are essentially independent subprograms suitable for other applications. (Since PRETRP, ARCALC, ARCPLT, and ARCPLTP are special-purpose subroutines that primarily communicate with LINSEQ and CONSEQ through labeled common, usage descriptions are not given.) In addition, since LINSEQ and CONSEQ are intended to be compatible with the NASA Langley Research Center graphic output library, a description of subprogram ASCALE has been included. ASCALE describes the use of the interleave factor, adjusted minimum, and scale factor used by LINSEQ and CONSEQ. The examples in appendix B also call CALCOMP, LEROY/BALLPT and CALPLT from the graphic output library, so these descriptions are included as well.

Subroutine LINSEQ

Language: FORTRAN

Purpose: To draw a solid or dashed smooth curve through a set of data points. Various dashed-line sequences with long and short dashes can be plotted.

Use: CALL LINSEQ(XX,YY,NN,K,XLIM,YLIM,NSP,H,SI,L1,L2,L3,L4,L5)

XX,YY	The names of the arrays containing the input values of X and Y to be plotted. LINSEQ expects the adjusted minimums in locations K*NN+1 and the scale factors in locations K*(NN+1)+1 (normally, NN+1 and NN+2, respectively, for K = 1) of XX and YY as described in ASCALE. XX and YY must thus be dimensioned at least K*NN+K+1 in the calling program (normally, NN+2 for K = 1).
NN	The number of input points to be plotted. For successive calls with the same XX- and YY-arrays (such as for different values of H or of L1 to L5), NN should be set to zero to bypass the arc length calculation and interpolation, thereby redrawing the previous curve with revised line parameters.
K	The interleave factor of XX and YY if mixed arrays. (Normally, K = 1.)
XLIM,YLIM	Two word arrays containing the lower and upper limits on X and Y, respectively, in units of XX and YY. These parameters allow the user to designate a rectangular frame, outside of which pen motion is not permitted.
NSP	The number of iterated smoothing passes applied to the input data points. A weighted, locally centered, five-point least-squares cubic is used. Smoothing does not affect XX and YY. (Normally, NSP = 0.)
H	The distance the drawn curve is offset normal to the input data in inches, positive to the right as the curve is drawn. (Normally, H = 0.)

SI	The approximate interval between interpolated points on the plotted curve in inches (0.05 inch is a reasonable value). The actual interpolation interval DS along the curve is computed from SI so that the curve starts and ends with a series of long dashes. If SI is set to a negative value, a preinterpolation of the input data to 2*NN-1 points is requested. Preinterpolation will be performed only if either the XX- or YY-array is monotonically increasing.
L1	The number of intervals of length DS in each gap between dashes.
L2	The number of long dashes per cycle.
L3	The number of intervals of length DS in each long dash.
L4	The number of short dashes per cycle.
L5	The number of intervals of length DS in each short dash.
Note:	If L1 = 0, L2 through L5 will be ignored and a solid curve will be drawn. If L4 = 0, L5 will be ignored and all dashes will be the same length. With these exceptions, L1 through L5 must be positive integers.

Restrictions: The subroutine contains five internal temporary storage arrays: S(101), X(101), Y(101), DX(101), and DY(101). These arrays are used to store certain plotting parameters. Thus, the value of NN is limited to 101 (51 if preinterpolation is used). These five arrays are in labeled common blocks ARCPAR, SCALEX, SCALEY, SLOPEX, and SLOPEY, respectively. If the user intends to process more than 101 input points, he should declare these blocks, with a larger dimension of at least NN, in a program or subprogram that will be loaded ahead of LINSEQ. It might also be noted that interpolation and computations are made for input XX and YY that are outside the frame specified by XLIM and YLIM. If input points are extremely far outside the frame, computational time will be unduly increased.

Method: The approximate arcs joining consecutive data points are computed. Then the input is interpolated by using natural cubic splines with distances along the arc as abscissas and with X and Y as ordinates. Because neither X nor Y is a privileged axis, this subroutine can be used to draw nonsingle-valued curves such as spirals. A drawback of this feature is that more data points are needed for acceptable fairing than would be required if conventional interpolation (i.e., with X as the abscissa) were used.

Storage:		
	LINSEQ	436_8
	ARCPLT	203_8
	PRETRP	161_8
	SPISET	300_8
	SPIFUN	127_8
	SPIDER	126_8
	SMOOTHC	220_8
	CHLSKYS	224_8
	ARCALC	304_8
	LIMPLT	71_8
	LABELED COMMON (ARCPAR,SCALEX, SCALEY,SLOPEX, SLOPEY, and CSPISET)	774_8
	TOTAL:	3634_8

Subprograms used: CALPLT and SQRT in addition to above list

Subroutine SPISET/Functions SPIFUN and SPIDER

Language: FORTRAN

Purpose: SPIFUN is a function subprogram to perform interpolation by using a natural cubic spline as the interpolation function. SPIDER is a function subroutine to compute the first derivative of a natural cubic spline. SPISET initializes certain parameters used by SPIFUN and SPIDER.

Use:
```
CALL SPISET(N,X,Y,D)
YP = SPIFUN(XP,N,X,Y,D)
DYDX = SPIDER(XP,N,X,Y,D)
```

N	The number of words in the input X- and Y-arrays.
X	The array of input abscissas.
Y	The array of input ordinates.
D	The array of N points used to store the first derivatives dY/dX at the data points. These derivatives are computed by SPISET and are required by SPIFUN and SPIDER.
XP	The abscissa at which interpolation or derivative evaluation is to be performed. If XP is not within the interval X(1) to X(N), linear extrapolation is performed by using the deflection and slope at the appropriate end.

Three additional seldom used parameters JMAX, IFRMS, and RMS are transmitted via labeled common in the following manner: COMMON/CSPISET/JMAX,IFRMS,RMS
The parameters JMAX and IFRMS are defaulted by a DATA statement.

JMAX	The number of iterations used to compute the derivatives (Default = 20.)
IFRMS	If 1, the RMS is computed; otherwise, the RMS is not computed (Default = 0.)
RMS	The root-mean-square value of the discontinuity in the second derivative at each data point.

Restrictions: The X-array must be monotonically increasing. The arrays X, Y, and D must be dimensioned N or larger in the calling program. SPISET must be called before the first call to SPIFUN or SPIDER.

Method: A natural cubic spline is completely defined for the interval (X_{i-1}, X_i) between data points by the four quantities Y_{i-1}, Y'_{i-1}, Y_i, and Y'_i. Subprogram SPIFUN or SPIDER tests XP to find which interval it is in and evaluates the ordinate or slope (respectively) of the cubic defined by the deflections and slopes at the two ends of the interval. Subprogram SPISET computes, and stores in D, the slopes required by SPIFUN and SPIDER. In order to save storage, the system of equations defining the array D is solved iteratively as in reference (a) of this subroutine. Since the D-array is only an approximation, there may be discontinuities in second derivatives at the data points. The root-mean-square value of this discontinuity can be tested by the user if desired.

Accuracy: The iterative scheme used to compute second derivatives is absolutely stable and converges at a rate of slightly more than two bits per iteration. The nominal number of iterations is 20. If the program is used for plotting, where high accuracy is not required, it is recommended that the user change the nominal number to 6.

Reference: (a) Greville, T. N. E.: Spline Functions, Interpolation, and Numerical Quadrature. Mathematical Methods for Digital Computers, Vol. II, Anthony Ralston and Herbert S. Wilf, eds., John Wiley & Sons, Inc., c.1967, pp. 156-168.

Storage:

SPISET	300_8
SPIFUN	127_8
SPIDER	126_8
COMMON	3_8

Subprograms used: None

Other coding information: These subprograms use reentrant code. Thus, independent calls to SPISET/SPIFUN/SPIDER can be used in the same program provided the actual parameter lists are distinct.

Subroutine SMOOTHC/Subroutine SMOOTHP

Language: FORTRAN

Purpose: To replace a set of empirical ordinates by smoothed values. Typically these subroutines are employed prior to interpolating data when it is suspected that noise may be present. Subroutine SMOOTHP is used when Y is a periodic function of X.

Use: CALL SMOOTHC(N,X,Y,T)

CALL SMOOTHP(N,X,Y,T)

N	The number of points in the X- and Y-arrays.
X	A one-dimensional array of input abscissas. If SMOOTHP is called, X(N)-X(1) is the length of the period.
Y	A one-dimensional array of input ordinates. On return, Y contains smoothed ordinates. Note that SMOOTHP sets Y(N) to Y(1).
T	A 25-word array of temporary storage. On input, T(1) must contain a number designating the degree to which the smoothing formula is locally weighted (see Method). Usually, T(1) is set to 1.

Restrictions: The array X must be monotonically increasing.

Method: Each abscissa Y(I), where I is the point index, is replaced by a new value computed by passing a least-squares cubic through the five points (X(J),Y(J)), with J = I-2, . . ., I+2. For the ends, Y(1), Y(2), Y(N-1), and Y(N), the five closest points are used when SMOOTHC is called.

The least-squares cubic is computed by using weights W(J) = 1./(S**2+(X(I)-X(J))**2) where S = T(1)*(X(N)-X(1))/N.

The user can select the amount of weighting by prescribing T(1) on input. In the limit as $S \rightarrow 0$, the effect of W(I) is to inhibit smoothing completely. In the limit as $S \rightarrow \infty$, the smoothing formula becomes the conventional five-point least-squares cubic described in reference (a) of these subroutines.

Reference: (a) Hildebrand, F. B.: Introduction To Numerical Analysis. McGraw-Hill Book Co., Inc., 1956, pp. 295-302.

Storage:	SMOOTHC	220_8
	SMOOTHP	244_8

Subprogram used: CHLSKYS

Subroutine LIMPLT

Language: FORTRAN

Purpose: To move plotter pen to new location with pen up or pen down if new location is within prescribed frame limits, but not to move pen if new location is outside prescribed frame limits.

Use:

Calling sequence: CALL LIMPLT(X,Y,IPEN)

X,Y	The floating-point values of page coordinates for pen movement (inches).
IPEN = 2	Move to point (X,Y) pen down if (X,Y) within frame limits.
IPEN = 3	Move to point (X,Y) pen up if (X,Y) within frame limits.

Entry point: CALL LIMSET(XLIM,YLIM,IPEN)

XLIM,YLIM	One-dimensional arrays of dimension two containing the X and Y minimum and maximum page-coordinate values (inches) in locations one and two, respectively.
IPEN = 0	For limits to be set with call to LIMSET.

Restrictions: A call to LIMSET with IPEN set to zero must be made prior to the use of LIMPLT.

Method: Point (X,Y) is checked and if $XLIM(1) \leqq X \leqq XLIM(2)$ and $YLIM(1) \leqq Y \leqq YLIM(2)$ are both satisfied, the basic pen-movement subroutine (CALPLT) is called with the value of IPEN passed to LIMPLT. If these conditions are not satisfied, the pen will not be moved but will go pen up to the next point that satisfies the conditions. Recycling can be accomplished with a subsequent call to LIMSET. This method permits a user to plot only the portion of a curve within the frame.

Accuracy: The limits of the frame are approached only as closely as the last point within the frame.

Subroutine CHLSKYS

Language: FORTRAN

Purpose: To solve a set of systems of linear algebraic equations $AX = B$, where A is a square symmetric coefficient matrix and B is a rectangular matrix of right-hand-side column vectors. On return, X (the matrix of solution vectors) is stored in B.

Use: CALL CHLSKYS(A,N,B,M,NX)

A	The square coefficient matrix. It is assumed symmetric by the program, so the user need only compute the diagonal and upper triangle of A.
N	The order of A and the number of rows of B.
B	The matrix of right-hand-side column vectors. On return, the solution vectors are stored in B.
M	The number of column vectors of B.
NX	The column size of A and B as given in the dimension statement of the calling program.

Restrictions: Reliable results can be guaranteed only if the matrix A is positive definite. The dimensions of A and B in the calling program should be A(NX,NN) where $NN \geqq N$ and B(NX,MM) where $MM \geqq M$.

Method: Cholesky's triangular decomposition is used (e.g., see ref. (a) of this subroutine). That is, A is expressed as

$$A = \bar{S}DS$$

where S is an upper triangular matrix with ones on the diagonal, D is the diagonal, and $\bar{S}$ is the transpose of S. Then, successively,

$$B' = S^{-1}B$$

$$B'' = D^{-1}B'$$

and

$$B''' = \bar{S}^{-1}B''$$

are computed. B''' is the matrix of solution vectors X. The decomposition of A and the three steps above are performed concurrently, so the matrices S, D, and $\bar{S}$ are never actually available in storage. The input matrix A is destroyed by the subprogram.

Accuracy: This algorithm can fail catastrophically if A is not positive definite. If it is suspected that $|A_{ij}A_{ji}|$ can be greater than $|A_{ii}A_{jj}|$ for some values of i and j, a different algorithm (employing pivoting) should be used. Loss of accuracy can also result from ill-conditioning. While there is no provision in the subprogram for condition-number computation, a useful estimate of the condition number is given by C1*C2 where C1 = A(1,1) before the call to CHLSKYS and C2 = A(N,N) after the call to CHLSKYS. Note that the loss of accuracy due to ill-conditioning is almost algorithm independent.

Reference: (a) Wilkinson, J. H.: The Solution Of Ill-Conditioned Linear Equations. Mathematical Methods for Digital Computers, Vol. II, Anthony Ralston and Herbert S. Wilf, eds., John Wiley & Sons, Inc., c.1967, pp. 65-93.

Storage: CHLSKYS 224_8

Subprograms used: None

Subroutine CONSEQ

Language: FORTRAN

Purpose: To draw a solid or dashed smooth closed curve (contour) through a set of data points. Various dashed-line sequences with long and short dashes can be plotted.

Use: CALL CONSEQ(XX,YY,NN,K,XLIM,YLIM,NSP,H,SI,L1,L2,L3,L4,L5)

XX,YY	The names of the arrays containing the input values of X and Y to be plotted. CONSEQ expects the adjusted minimums in locations K*(NN+1)+1 and the scale factors in locations K*(NN+1)+1 (normally, NN+1 and NN+2, respectively, for K = 1) of XX and YY as described in ASCALE. XX and YY must thus be dimensioned at least K*(NN+1)+1 (normally NN+2 for K = 1) in the calling program. CONSEQ closes the curve by taking the last point to be (XX(1),YY(1)). Thus (XX(1),YY(1)) must not be the same point as (XX(NN),YY(NN)).
NN	The number of input points to be plotted. For successive calls with the same XX- and YY-arrays (such as for different values of H or of L1 to L5), NN should be set to zero to bypass the arc length calculation and interpolation, thereby redrawing the previous curve with revised line parameters.
K	The interleave factor of XX and YY if mixed arrays. (Normally, K = 1.)
XLIM,YLIM	Two word arrays containing the lower and upper limits on X and Y, respectively, in units of XX and YY. These parameters allow the user to designate a rectangular frame, outside of which pen motion is not permitted.
NSP	The number of iterated smoothing passes applied to the input data points. A weighted, locally centered, five-point least-squares cubic is used. Smoothing does not affect XX and YY. (Normally, NSP = 0.)
H	The distance the drawn curve is offset normal to the input data in inches, positive to the right as the curve is drawn. (Normally, H = 0.)

SI	The approximate interval between interpolated points on the plotted curve in inches (0.05 inch is a reasonable value). The actual interpolation interval DS is computed from SI so that the curve contains an integral number of cycles.
L1	The number of intervals of length DS in each gap between dashes.
L2	The number of long dashes per cycle.
L3	The number of intervals of length DS in each long dash.
L4	The number of short dashes per cycle.
L5	The number of intervals of length DS in each short dash.
Note:	If L1 = 0, L2 through L5 will be ignored and a solid curve will be drawn. If L4 = 0, L5 will be ignored and all dashes will be the same length. With these exceptions, L1 through L5 must be positive integers.

Restrictions: The subroutine contains five internal temporary storage arrays: S(101), X(101), Y(101), DX(101), and DY(101). These arrays are used to store certain plotting parameters. The value of NN is limited to 100. These five arrays are in labeled common blocks ARCPAR, SCALEX, SCALEY, SLOPEX, and SLOPEY, respectively. If the user intends to process more than 100 input points, he should declare these blocks, with a larger dimension of at least NN+1, in a program or subprogram that will be loaded ahead of CONSEQ. It might also be noted that interpolation and computations are made for input XX and YY that are outside the frame specified by XLIM and YLIM. If input points are extremely far outside the frame, computational time will be unduly increased.

Method: The approximate arcs joining consecutive data points are computed. Then the input is interpolated by using periodic cubic splines with distances along the arc as abscissas and with X and Y as ordinates.

Storage:	CONSEQ	401_8
	ARCPLTP	203_8
	SPISETP	520_8

SPIFUNP	136_8
SPIDERP	141_8
SMOOTHP	244_8
CHLSKYS	224_8
ARCALC	304_8
LIMPLT	71_8
LABELED COMMON (ARCPAR,SCALEX, SCALEY,SLOPEX, SLOPEY, and CSPISET)	774_8
TOTAL	3704_8

(Note that CHLSKYS, ARCALC, LIMPLT, and LABELED COMMON are also used by LINSEQ.)

Subprograms used: CALPLT and SQRT in addition to above list

Subroutine SPISETP/Functions SPIFUNP and SPIDERP

Language: FORTRAN

Purpose: SPIFUNP is a function subprogram to perform interpolation by using a periodic cubic spline as the interpolation function. SPIDERP is a function subroutine to compute the first derivative of a periodic cubic spline. SPISETP initializes certain parameters used by SPIFUNP and SPIDERP.

Use:
```
CALL SPISETP(N,X,Y,D)
YP = SPIFUNP(XP,N,X,Y,D)
DYDX = SPIDERP(XP,N,X,Y,D)
```

N	The number of words in the input X- and Y-arrays.
X	The array of input abscissas. X(N)-X(1) is the length of the period.
Y	The array of input ordinates. Y(N) need not be furnished, as it will be set to Y(1) by the program.
D	The array of N points used to store the first derivatives dY/dX at the data points. These derivatives are computed by SPISETP and are required by SPIFUNP and SPIDERP.
XP	The abscissa at which interpolation or derivative evaluation is to be performed. If XP is not within the interval X(1) to X(N), the argument of the cubic is made to lie in that interval by adding or subtracting the appropriate multiple of the period.

Three additional seldom used parameters JMAX, IFRMS, and RMS are transmitted via labeled common in the following manner: COMMON/CSPISET/JMAX,IFRMS,RMS
The parameters JMAX and IFRMS are defaulted by a DATA statement.

JMAX	The number of iterations used to compute the derivatives (Default = 20.)
IFRMS	If 1, the RMS is computed; otherwise, the RMS is not computed (Default = 0.)

RMS	The root-mean-square value of the discontinuity in the second derivative at each data point.

Restrictions: The X-array must be monotonically increasing. The arrays X, Y, and D must be dimensioned N or larger in the calling program. SPISETP must be called before the first call to SPIFUNP or SPIDERP.

Method: A natural cubic spline is completely defined for the interval (X_{i-1},X_i) between data points by the four quantities Y_{i-1}, Y'_{i-1}, Y_i, and Y'_i. Subprogram SPIFUNP or SPIDERP tests XP to find which interval it is in and evaluates the ordinate or slope (respectively) of the cubic defined by the deflections and slopes at the two ends of the interval. Subprogram SPISETP computes, and stores in D, the slopes required by SPIFUNP and SPIDERP. In order to save storage, the system of equations defining the array D is solved iteratively as in reference (a) of this subroutine. Since the D-array is only an approximation, there may be discontinuities in second derivatives at the data points. The root-mean-square value of this discontinuity can be tested by the user if desired.

Accuracy: The iterative scheme used to compute second derivatives is absolutely stable and converges at a rate of slightly more than two bits per iteration. The nominal number of iterations is 20. If the program is used for plotting, where high accuracy is not required, it is recommended that the user change the nominal number to 6.

Reference: (a) Greville, T. N. E.: Spline Functions, Interpolation, and Numerical Quadrature. Mathematical Methods for Digital Computers, Vol. II, Anthony Ralston and Herbert S. Wilf, eds., John Wiley & Sons, Inc., c.1967, pp. 156-168.

Storage:

SPISETP	520_8
SPIFUNP	136_8
SPIDERP	141_8
COMMON	3_8

Subprograms used: None

Other coding information: These subprograms use reentrant code. Thus, independent calls to SPISETP/SPIFUNP/SPIDERP can be used in the same program provided the actual parameter lists are distinct.

Subroutine CALCOMP

Language: FORTRAN

Purpose: This is the normal mode processor. The necessary parameters and linkage are set up to output a tape for the CalComp Model 780/763 Electro-Mechanical Plotter.

Use: CALL CALCOMP

Restrictions: This call must be given before the first call to a plotting routine.

Storage: CALCOMP 5467_8 total for all subprograms used

Subprograms used: PLOTSW, PLT763, BLCR, STRC, TAPWRI, ENCOD1, STRCALL, CREATEF, BOUNDCK, TRUNCL, and LOCATE

Other coding information: The following is a list of messages, the circumstances under which they will appear, and the action taken:

NO PLOTTING DEVICE SPECIFIED	This message is printed in the output file and the job is ended in subroutine PLOTSW. This condition occurs if there is no initialization CALL CALCOMP in the program prior to using subroutines which generate plotting output.
PLOTTING COMMENCED	This message is printed in the dayfile when the first pen movement is encountered as a result of a program call to the plotting subroutines.
THE LAST CALCOMP BLOCK ADDRESS WAS xxx DATA PLOTTED = yyyyyy	These messages are printed in the dayfile when the plotting is completed. xxx is the value of the last block address on the CalComp plotter tape. This block address is at the end of the last valid data. yyyyyy is the approximate number of data points. Plotting is completed either as a result of a CALL CALPLT(x,y,999) or when the job is ended due to an error recognized by the CalComp subroutines.

Subroutine CALPLT

Language: FORTRAN

Purpose: To move the plotter pen to a new location with pen up or down and to signal the end of a job segment by incrementing the block address number.

Use: CALL CALPLT(X,Y,IPEN)

X,Y	The floating-point values for pen movement.
IPEN = 2	Pen down
IPEN = 3	Pen up

Negative IPEN will assign (X = 0, Y = 0) as the location of the pen after moving the X,Y (creating a new reference point) and will increase the block number by one. (The block number is the number that appears in the display at the top of the tape drive on the plotter and identifies the portion of the output tape that is being plotted. The block address 001 is written automatically as a result of the initialization processor call.) Each block address generally implies a separate page or plot.

IPEN = 999	Writes a terminating block address of 999 for peripheral handling of the plotter tape and all further processing is skipped. X and Y may be any values since they are ignored.

Restrictions: All X- and Y-coordinates must be expressed as floating-point values in inches (actual page dimensions) in deflection from the origin.

(A CALL TO CALPLT WITH EITHER NEGATIVE IPEN (USUALLY -3) OR A TERMINATING BLOCK ADDRESS (IPEN = 999) MUST BE GIVEN AS THE LAST PLOTTING INSTRUCTION BEFORE ENDING A PROGRAM WHICH USES ANY OF THE PLOTTER SUBROUTINES: THIS IS TO BE SURE THAT ALL PLOTTER INSTRUCTIONS ARE WRITTEN ON THE PLOTTER TAPE.)

Method: The main subroutine in the CalComp software package is the CALPLT subroutine. All other special-purpose subroutines eventually call CALPLT either directly or indirectly. Subroutine CALPLT moves the pen in a straight line between the present pen position and another pen location to which the programer wishes the pen to be moved.

In order to cause such instructions to be written, the programer specifies the coordinates of the point to which the pen is to be moved and whether the pen is to be moved in a raised or lowered position. This movement is accomplished by the FORTRAN instruction

```
CALL CALPLT(X,Y,IPEN)
```

Also, the subroutine provides "sequence numbers" on the tape, making it possible to afford identification of job segments. The block address 001 is written on the first call to CALPLT. Thereafter, if the programer defines a new origin or wishes to divide the job into several segments, he need only set the argument IPEN negative. The CALPLT routine then moves the pen to (X,Y); stores this location as (0,0), that is, a new origin; and increases the block address by one.

Storage: CALPLT 253_8

Subprograms used: PLOTSW, STRCALL, and LOCATE

Subroutine ASCALE

Language: FORTRAN

Purpose: To compute a scaling factor for an array of numbers to be plotted over a certain area and find the minimum data value within the array.

Use: CALL ASCALE(ARRAY,S,N,K,DV)

ARRAY	The name of the array containing the floating-point values to be scaled.
S	The length (floating-point value in inches) over which the data are to be plotted (usually the length of one of the axes).
N	The number of data values in ARRAY from which points are to be plotted in accordance with K.
K	The interleave factor which specifies the sequence in which data are stored. K = 1 indicates that values are stored sequentially. K = 2 indicates that values are stored in every other location in the array and so forth.
DV	The number of divisions per inch of the plotting paper to be used. (DV should be 10.0, 20.0, 25.0, or 25.4.)

Restrictions: The array must be dimensioned to include storage space for two extra elements per interleave factor. For example: N = 100, K = 1, DIMENSION ARRAY (102); N = 75, K = 3, DIMENSION ARRAY (231).

Method: This routine scans the elements in the array to find the minimum and maximum. ASCALE computes an adjusted minimum (origin value) and stores it in ARRAY((N*K)+1) and computes a scale factor and stores it in ARRAY((N*K)+1+K). The scale factor will be either 2, 4, 5, or 10 with a decimal exponent. The data in the array may be scaled to floating-point values in inches by using a formula similar to the following: SV = (AE-MV)/SF, where SV is the scaled value, AE is the present value of the array element, MV is either the minimum value or the value desired at the origin, and SF is the scale factor computed by the subroutine.

Storage: ASCALE 262_8

Subprograms used: ALOG, ALOG10

Other coding information: Example:

```
DIMENSION ORD(102),ABS(204)
CALL ASCALE(ORD,10.,100,1,10.)
CALL ASCALE(ABS,15.,100,2,10.)
```

Subroutine LEROY/BALLPT

Language: FORTRAN

Purpose: The parameters necessary to accommodate plotting with the liquid ink pen are set up by CALL LEROY. Once set, this mode will remain in effect as long as CALCOMP is in use or until a call to BALLPT is given.

The parameters for plotting with the ballpoint pen are reset by CALL BALLPT. This mode is automatically in effect with CALCOMP unless there has been a call to LEROY.

Use: CALL BALLPT

CALL LEROY

Restrictions: The CALL LEROY should be used only with CALCOMP. In addition to reducing the speed of the plotter for all plotting movements, the number of plot vectors in any annotation is considerably increased.

The CALL LEROY must be made prior to any plotting calls, but after the CALL CALCOMP.

Storage: LEROY/BALLPT 25_8

Subprograms used: None

APPENDIX B

SAMPLE CALLING PROGRAMS AND THEIR RESULTS

This appendix contains several examples illustrating the use of subroutine LINSEQ. For each example, the adjusted minimum was set to 0, the scale factor was set to 0.125, and the interpolation interval SI was set to 0.16 inch, except when otherwise noted. The curves were reduced to approximately one-half of their original size for insertion into this appendix, so the interpolation interval on the figures in this appendix is 0.08 inch. For each example, the five line-sequence parameters are listed in the form L = (L1,L2,L3,L4,L5).

In order to keep the calling sequences simple, only LINSEQ and CONSEQ have been used to construct the plots. For this reason, most of the plots do not include the usual additional information, such as axes, scales, and legends. The sample calling programs also contain four installation-dependent calls that do not actually cause pen motion. They are as follows:

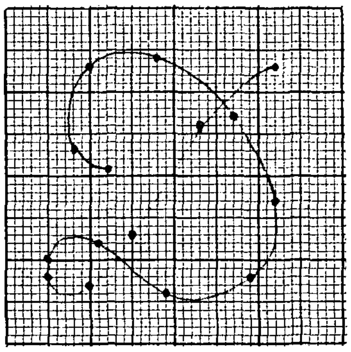
CALL CALCOMP	Device initialization
CALL LEROY	Slows down device to improve quality
CALL CALPLT(0.,0.,-3)	Sets origin on paper
CALL CALPLT(0.,0.,999)	Writes terminating block address on plotting tape

Examples 3, 4, and 5 also contain a call to a Langley subroutine LINPLT used to put plotting symbols at the input data points. The parameter list is (X,Y,N,K,-1,13,3,0) where X, Y, N, and K have the same meaning as in LINSEQ; -1 means don't connect the points; 13 means use the 13th plotting symbol (diamond with cross inside); 3 means use a large symbol; and the last parameter 0 is unused here.

Note that for the FORTRAN compiler used, the symbol $ separates distinct FORTRAN statements on the same coding line.

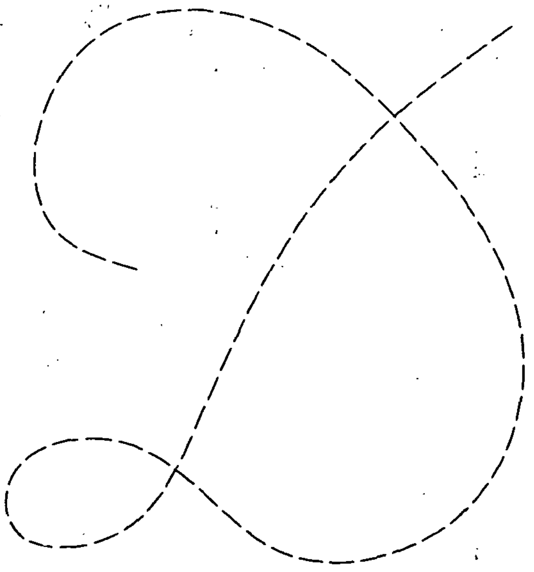
Example 1 – Plotting Empirical Data

Example 1 illustrates the use of LINSEQ to plot empirical data and shows that the input data need not represent a single-valued function. A letter D was drawn on graph paper and digitized at 15 points as shown in the figure and table on the following page.



X	Y
0.60	0.65
.15	.30
-.25	-.35
-.50	-.65
-.75	-.60
-.75	-.50
-.45	-.40
-.05	-.70
.45	-.60
.60	-.15
.35	.35
-.10	.70
-.50	.65
-.60	.15
-.40	.05

Result.- The information in the above table was read from cards and plotted by using the line-sequence parameters L = (1,1,3,0,0). The resulting plot is



Listing of program and data.- The listing of the program and data for Example 1 is as follows:

```
      DIMENSION X(101),Y(101),XLIM(2),YLIM(2)
*     EXAMPLE 1.  PLOTTING EMPIRICAL DATA
      ADJMIN=0.  $  SCAFAC=.125
      CALL CALCOMP  $  CALL LEROY  $  CALL CALPLT(0.,0.,-3)
      SI=.16  $  K=1
      READ 2, N
      X(N+1)=Y(N+1)=ADJMIN  $  X(N+2)=Y(N+2)=SCAFAC
      XLIM(1)=YLIM(1)=-1.2  $  XLIM(2)=YLIM(2)=1.2
      DO 1 I=1,N
    1 READ 3, X(I),Y(I)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,1,1,3,0,0)
      CALL CALPLT(0.,0.,999)
    2 FORMAT(I4)
    3 FORMAT(2F5.2)
      END

  15
  .60  .65
  .15  .30
 -.25 -.35
 -.50 -.65
 -.75 -.60
 -.75 -.50
 -.45 -.40
 -.05 -.70
  .45 -.60
  .60 -.15
  .35  .35
 -.10  .70
 -.50  .65
 -.60  .15
 -.40  .05
```

Example 2 – Various Features of LINSEQ

Example 2 illustrates the following aspects of the use of LINSEQ:

(1) The use of LINSEQ to distinguish several different curves on the same plot.

(2) The use of a user-written subroutine (GENSEQ here) to generate a "standard" set of line-sequence parameters. (This subroutine could be more complicated or simpler than GENSEQ.)

(3) The use of two-point calls to draw axes and the identification key for the figure. (Note that the lettering $U_1(X)$ through $U_7(X)$ was typed after the plots were generated.)

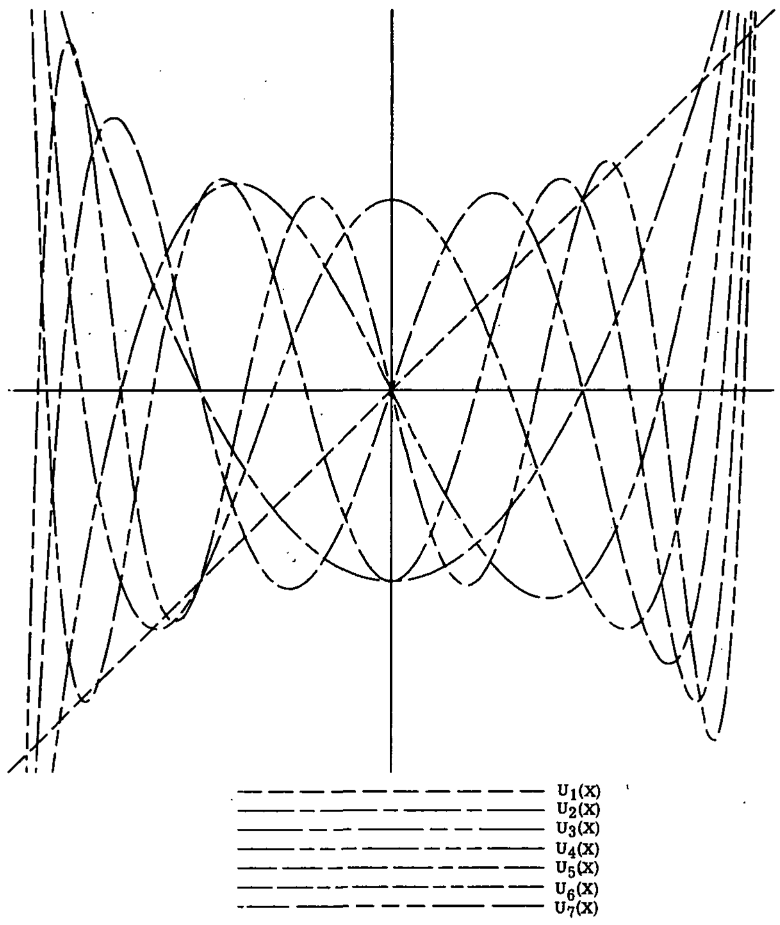
(4) The use of user-declared labeled common blocks to increase the capacity of LINSEQ to more than the nominal value of 101 points.

(5) The use of frame limits to limit the plotted curves to a predeclared frame.

Result.- The data plotted are Chebyshev polynomials of the second kind

$$\frac{1}{2} U_{L-1}(X) = \frac{\sin\left(L \cos^{-1} X\right)}{2 \sin\left(\cos^{-1} X\right)} \qquad (L = 2, 3, \ldots, 8)$$

digitized at 23L - 43 points (i.e., from 3 to 141 points depending on degree). The frame limit parameter YLIM was used to limit the curves to Y = ±1. The resulting plot is



Listing of program.- The program listing for Example 2 is as follows:

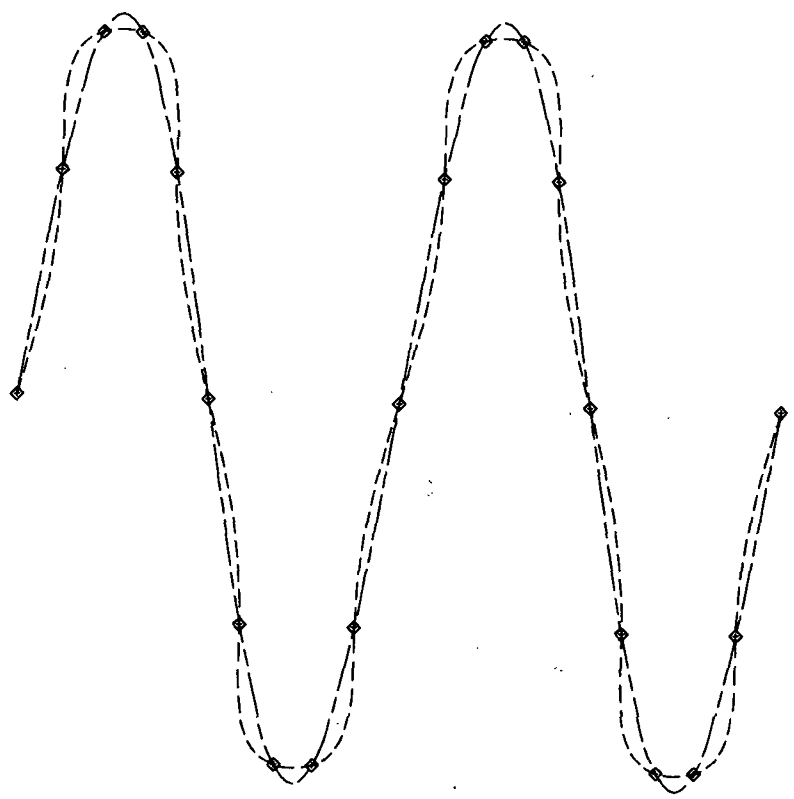
```
      DIMENSION X(143),Y(143),XLIM(2),YLIM(2)
*  EXAMPLE 2.  VARIOUS FEATURES OF LINSEQ
      DIMENSION U(4),V(4)
      COMMON/ARCPAR/ARCPAR(141)/SCALEX/SCALEX(141)/SCALEY/SCALEY(141)
      COMMON/SLOPEX/SLOPEX(141)/SLOPEY/SLOPEY(141)
      ADJMIN=0.  $  SCAFAC=.125
      CALL CALCOMP  $  CALL LEROY  $  CALL CALPLT(0.,0.,-3)
      SI=.08  $  K=1
      XLIM(1)=YLIM(1)=-1.00001  $  XLIM(2)=YLIM(2)=1.00001
      U(3)=V(3)=ADJMIN  $  U(4)=V(4)=SCAFAC
      U(1)=-1.  $  U(2)=1.  $  V(1)=V(2)=0
      CALL LINSEQ(U,V,2,K,XLIM,YLIM,0,0,SI,0,0,0,0,0)
      CALL LINSEQ(V,U,2,K,XLIM,YLIM,0,0,SI,0,0,0,0,0)
      PI=3.14159265358979  $  EPS=1.E-12
      DO 2 L=2,8  $  N=23*L-43  $  DO 1 I=1,N
      T=PI*(N-I)/(N-1.)+EPS  $  X(I)=COS(T)
    1 Y(I)=.5*SIN(L*T)/SIN(T)  $  CALL GENSEQ(L,L1,L2,L3,L4,L5)
      X(N+1)=Y(N+1)=ADJMIN  $  X(N+2)=Y(N+2)=SCAFAC
    2 CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,L1,L2,L3,L4,L5)
      YLIM(1)=-2.  $  U(1)=-.4  $  U(2)=.4  $  V(1)=-1.  $  DO 6 L=2,8
      CALL GENSEQ(L,L1,L2,L3,L4,L5)  $  V(1)=V(2)=V(1)-.05
    6 CALL LINSEQ(U,V,2,K,XLIM,YLIM,0,0,SI,L1,L2,L3,L4,L5)
      CALL CALPLT(0.,0.,999)
      END

      SUBROUTINE GENSEQ(L,L1,L2,L3,L4,L5)
*  GENERATES A SO-CALLED STANDARD LINE SEQUENCE FOR USE BY LINSEQ
      L1=2  $  L2=L3=L4=L5=0  $  IF(L-2)1,2,3
    1 L1=0  $  RETURN
    2 L2=1  $  L3=6  $  RETURN
    3 L3=16  $  L5=4  $  IF(L.GT.5)L3=12  $  L2=L/3  $  L4=L+1-3*L2
      RETURN
      END             SUBROUTINE GENSEQ
```

Example 3 – Preinterpolation

Example 3 illustrates the use of the preinterpolation option.

Result.- The function $Y = \sin(2\pi X)$ was generated at 21 equispaced abscissas on the interval (-1,1). These abscissas were plotted by using symbols. Then LINSEQ was called with line-sequence parameters $L = (1,1,2,0,0)$. The resulting curve that looks like a string of bowling pins (see short dashes) is much worse than one would expect from 10 data points per period. LINSEQ was called again with line-sequence parameters $L = (1,1,6,1,2)$ and with the preinterpolation option set (minus sign on interpolation interval SI). The curve that results from using preinterpolation (see long and short dashes) is very close to a true sine wave. These two curves are presented on the opposite page.



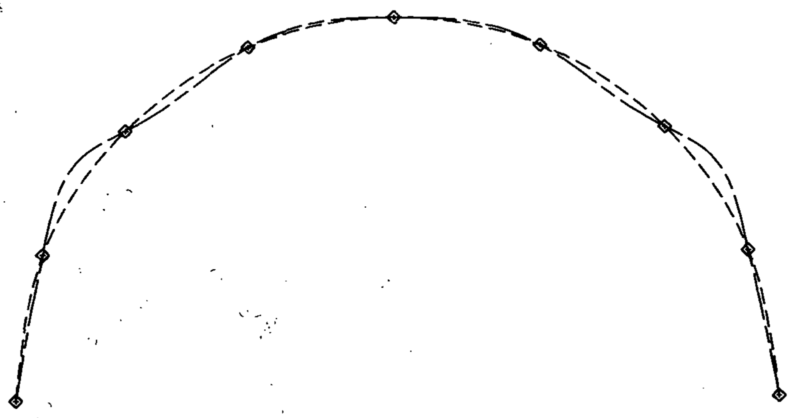
Listing of program.- The program listing for Example 3 is as follows:

```
      DIMENSION X(23),Y(23),XLIM(2),YLIM(2)
*    EXAMPLE 3.  PREINTERPOLATION
      ADJMIN=0.  $  SCAFAC=.125
      CALL CALCOMP  $  CALL LEROY  $  CALL CALPLT(0.,0.,-3)
      SI=.16  $  K=1  $  N=21
      X(N+1)=Y(N+1)=ADJMIN  $  X(N+2)=Y(N+2)=SCAFAC
      XLIM(1)=YLIM(1)=-1.2  $  XLIM(2)=YLIM(2)=1.2
      PI=3.14159265358979
      DO 1 I=1,N  $  X(I)=-1.+.1*(I-1)
    1 Y(I)=SIN(2.*PI*X(I))
      CALL LINPLT(X,Y,N,K,-1,13,3,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,1,1,2,0,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,-SI,1,1,6,1,2)
      CALL CALPLT(0.,0.,999)
      END
```

Example 4 – Incorrect Use of Preinterpolation

Example 4 shows how the preinterpolation option can be used incorrectly.

Result.- A semicircle was generated at nine points and plotted as symbols. LINSEQ was called with line-sequence parameters L = (1,1,2,0,0). The result is the short-dashed curve. Then LINSEQ was called again with L = (1,1,6,1,2) and the preinterpolation option set. The resulting curve (long and short dashes) is much worse than the result without preinterpolation because dY/dY is infinite at the two ends of the interval. Preinterpolation should not be used if Y or dY/dY is infinite or extremely large over the interpolation interval. These two curves are presented below:



Listing of program.- The program listing for Example 4 is as follows:

```
      DIMENSION X(23),Y(23),XLIM(2),YLIM(2)
*     EXAMPLE 4.   INCORRECT USE OF PREINTERPOLATION
      ADJMIN=0.   $   SCAFAC=.125
      CALL CALCOMP   $   CALL LEROY   $   CALL CALPLT(0.,0.,-3)
      SI=.16   $   K=1   $   N=9
      X(N+1)=Y(N+1)=ADJMIN   $   X(N+2)=Y(N+2)=SCAFAC
      XLIM(1)=YLIM(1)=-1.2   $   XLIM(2)=YLIM(2)=1.2
      PI=3.14159265358979
      DO 1 I=1,N   $   T=PI*(N-I)/(N-1.)   $   X(I)=COS(T)
    1 Y(I)=SIN(T)
      CALL LINPLT(X,Y,N,K,-1,13,3,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,1,1,2,0,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,-SI,1,1,6,1,2)
      CALL CALPLT(0.,0.,999)
      END
```

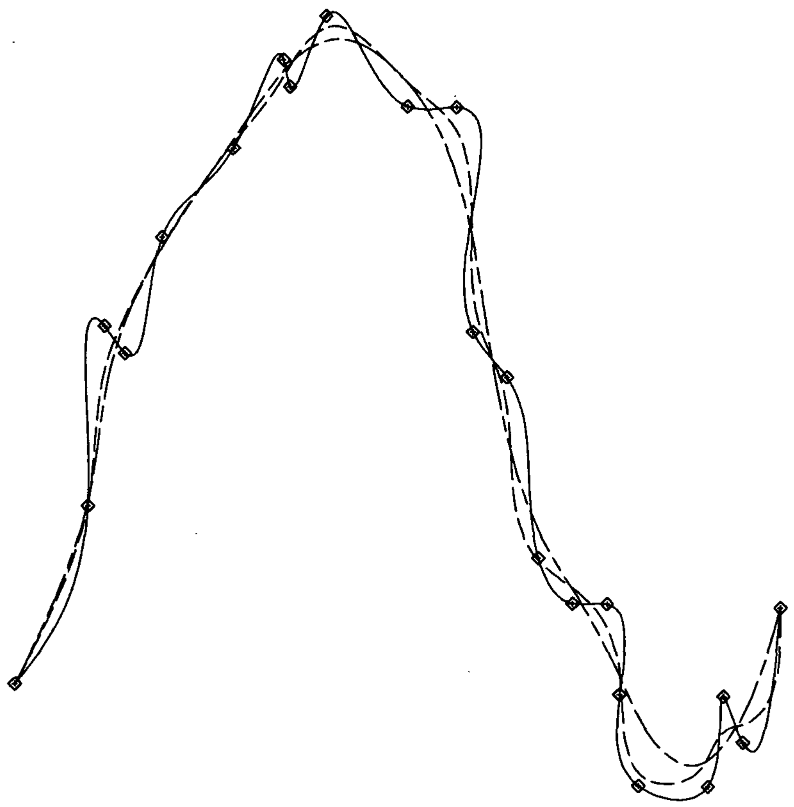
Example 5 – Smoothing

Example 5 illustrates the use of the smoothing option.

Result.- Twenty-two points from an empirical curve were read and plotted as symbols. LINSEQ was called three times with parameters set as shown below:

Solid curve	No smoothing (NSP = 0); L = (0,0,0,0,0)
Short dashes	One smoothing pass (NSP = 1); L = (1,1,3,0,0)
Long and short dashes	Eight smoothing passes (NSP = 8); L = (1,1,6,1,2)

Note that the curve with 8 smoothing passes (NSP = 8) closely resembles a cubic polynomial in the resulting plot presented below:



Listing of program and data.- The program listing for Example 5 is as follows:

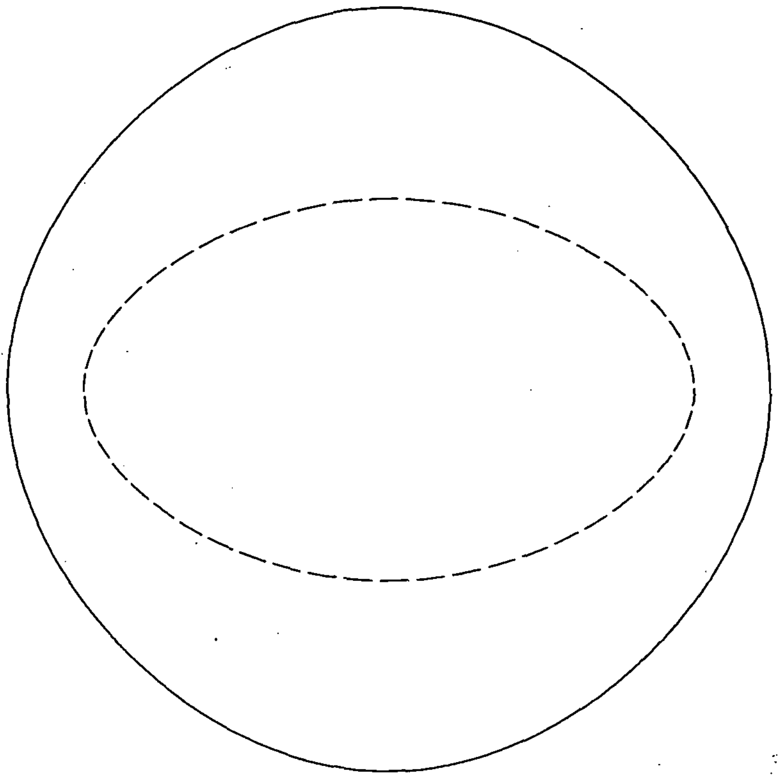
```
      DIMENSION X(101),Y(101),XLIM(2),YLIM(2)
*  EXAMPLE 5.  SMOOTHING
      CALL CALCOMP  $  CALL LEROY  $  CALL CALPLT(0.,0.,-3)
      SI=.16  $  K=1
      READ 2, N
      XLIM(1)=YLIM(1)=-20.  $  XLIM(2)=YLIM(2)=20.
      READ 3, X(1),Y(1)  $  XMIN=XMAX=X(1)  $  YMIN=YMAX=Y(1)
      DO 1 I=2,N  $  READ 3, X(I),Y(I)
      IF(X(I).GT.XMAX) XMAX=X(I)  $  IF(X(I).LT.XMIN) XMIN=X(I)
      IF(Y(I).GT.YMAX) YMAX=Y(I)  $  IF(Y(I).LT.YMIN) YMIN=Y(I)
    1 CONTINUE  $  X(N+1)=XMIN  $  Y(N+1)=YMIN
      X(N+2)=.0625*(XMAX-XMIN)  $  Y(N+2)=.0625*(YMAX-YMIN)
      CALL LINPLT(X,Y,N,K,-1,13,3,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,0,0,0,0,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,1,0,SI,1,1,3,0,0)
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,8,0,SI,1,1,6,1,2)
      CALL CALPLT(0.,0.,999)
    2 FORMAT(I4)
    3 FORMAT(2F5.2)
      END

  22
  1.0  5.0
  2.0  7.0
  2.2  9.0
  2.5  8.7
  3.0 10.0
  4.0 11.0
  4.7 12.0
  4.8 11.7
  5.3 12.5
  6.5 11.5
  7.2 11.5
  7.5  9.0
  8.0  8.5
  8.5  6.5
  9.0  6.0
  9.5  6.0
  9.7  5.0
 10.0  4.0
 11.0  4.0
 11.2  5.0
 11.5  4.5
 12.0  6.0
```

Example 6 – Closed Curves

Example 6 shows how CONSEQ can be used to draw closed curves.

Result.- A four-point curve resembling a circle defined at (-1,0), (0,1), (1,0), and (0,-1) was drawn with line-sequence parameters L = (0,0,0,0,0). A four-point curve resembling an ellipse defined at (-0.8,0), (0,0.5), (0.8,0), and (0,-0.5) was drawn with L = (1,1,3,0,0). These curves are presented below:



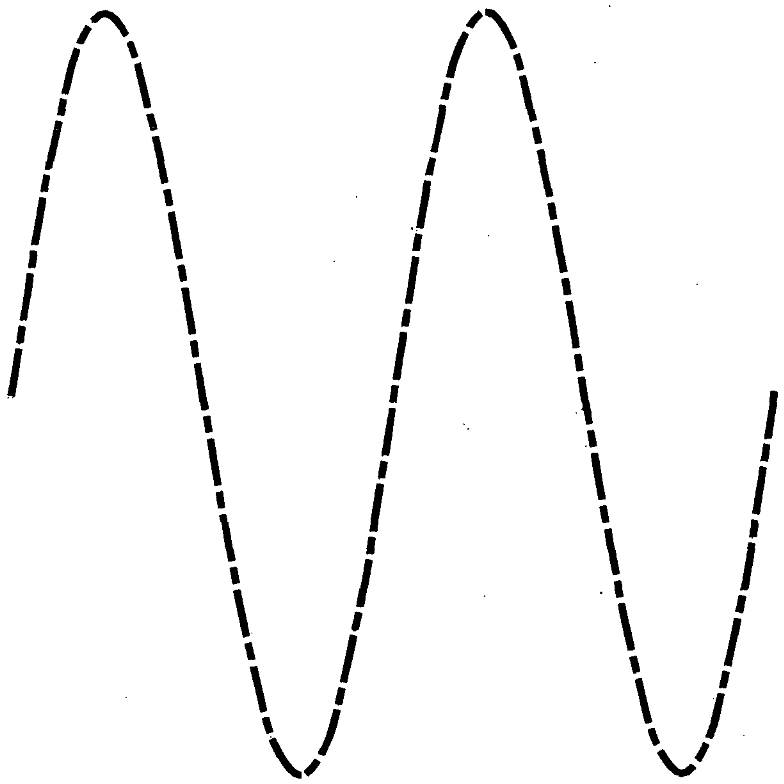
Listing of program.- The program listing for Example 6 is as follows:

```
      DIMENSION X(23),Y(23),XLIM(2),YLIM(2)
*  EXAMPLE 6.     CLOSED CURVES
      ADJMIN=0.  $  SCAFAC=.125
      CALL CALCOMP  $  CALL LEROY  $  CALL CALPLT(0.,0.,-3)
      SI=.16  $  K=1
      XLIM(1)=YLIM(1)=-1.2  $  XLIM(2)=YLIM(2)=1.2
      N=4
      X(N+1)=Y(N+1)=ADJMIN  $  X(N+2)=Y(N+2)=SCAFAC
* FOUR POINT CIRCLE
      X(1)=0    $  Y(1)=1.
      X(2)=1.   $  Y(2)=0
      X(3)=0    $  Y(3)=-1.
      X(4)=-1.  $  Y(4)=0
      CALL CONSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,0,0,0,0,0)
*  FOUR POINT ELLIPSE
      X(1)=-.8   $  Y(1)=0
      X(2)=0     $  Y(2)=.5
      X(3)=.8    $  Y(3)=0
      X(4)=0     $  Y(4)=-.5
      CALL CONSEQ(X,Y,N,K,XLIM,YLIM,0,0,SI,1,1,3,0,0)
      CALL CALPLT(0.,0.,999)
      END
```

Example 7 – Offset Feature

Example 7 illustrates the use of the offset parameter to draw a heavy line curve such as would be used for an oral presentation. LINSEQ was called seven times with values of the offset parameter H differing by slightly less than the pen width. Note that for all calls except the first, the third parameter of LINSEQ was set to zero.

Result.- The resulting figure plotted below is for the same input data points as Example 3:



Listing of program.- The program listing for Example 7 is as follows:

```
      DIMENSION X(23),Y(23),XLIM(2),YLIM(2)
*    EXAMPLE 7.  OFFSET FEATURE
      ADJMIN=0.  $  SCAFAC=.125
      CALL CALCOMP  $  CALL LERDY  $  CALL CALPLT(0.,0.,-3)
      SI=.16  $  K=1  $  N=21
      X(N+1)=Y(N+1)=ADJMIN  $  X(N+2)=Y(N+2)=SCAFAC
      XLIM(1)=YLIM(1)=-1.2  $  XLIM(2)=YLIM(2)=1.2
      PI=3.14159265358979
      DO 1 I=1,N  $  X(I)=-1.+.1*(I-1)
    1 Y(I)=SIN(2.*PI*X(I))
      H=-.06  $  DO 2 II=1,7
      CALL LINSEQ(X,Y,N,K,XLIM,YLIM,0,H,-SI,1,1,6,1,2)
      H=H+.02  $  N=0
    2 CONTINUE
      CALL CALPLT(0.,0.,999)
      END
```

APPENDIX C

SUBPROGRAM LISTINGS

This appendix contains FORTRAN IV listings of subprograms LINSEQ and CONSEQ and of the subprograms called by LINSEQ and CONSEQ (except system routines CALPLT and SQRT). Annotation is minimal, so the descriptions in appendix A should be consulted when the listings are read.

Listing of Subroutine LINSEQ

```
      SUBROUTINE LINSEQ(XX,YY,NN,K,XLIM,YLIM,NSP,H,SI,L1,L2,L3,L4,L5)
*     SUBROUTINE FOR PLOTTING SEQUENCE OF CURVED LINES USING LONG AND SHORT
*       DASHES
      DIMENSION XX(1),YY(1),XLIM(1),YLIM(1)
      COMMON/ARCPAR/S(101)/SCALEX/X(101)/SCALEY/Y(101)
      COMMON/SLOPEX/DX(101)/SLOPEY/DY(101)
      COMMON/CSPISET/JMAX,IFRMS,RMS
      JMAX=6
      IF(NN.EQ.0) GO TO 8
      N=NN
      KN=K*N+1
      KNK=KN+K
      RX=1./XX(KNK)
      RY=1./YY(KNK)
*     COMPUTE SCALED  FRAME LIMITS FOR LIMPLT
      X=(XLIM-XX(KN))*RX
      X(2)=(XLIM(2)-XX(KN))*RX
      Y=(YLIM-YY(KN))*RY
      Y(2)=(YLIM(2)-YY(KN))*RY
      CALL LIMSET(X,Y,0)
*     COMPUTE SCALED X,Y ARRAYS
      DO 2 I=1,N
      KI=K*I
      X(I)=(XX(KI)-XX(KN))*RX
    2 Y(I)=(YY(KI)-YY(KN))*RY
*     CALL PRETRP IF PRE-INTERPOLATION IS REQUESTED (SI.LT.0)
      IF(SI.LT.0)CALL PRETRP(N,SI)
      SS=SI
*     CALL ARCALC TO COMPUTE APPROXIMATE ARC LENGTH BETWEEN DATA POINTS
      CALL ARCALC(N,0)
      IF(NSP.EQ.0) GO TO 6
*     SMOOTHING OF X = X(S) AND Y = Y(S) IS PERFORMED BY CALLS TO SMOOTHC
      DO 4 I=1,NSP
      DX=I
      CALL SMOOTHC(N,S,X,DX)
      DX=I
    4 CALL SMOOTHC(N,S,Y,DX)
      CALL ARCALC(N,0)
*     CALLS TO SPISET INITIALIZE SPLINE DERIVATIVE ARRAYS DX AND DY FOR USE
*       BY SPIFUN AND SPIDER
    6 CALL SPISET(N,S,X,DX)
      CALL SPISET(N,S,Y,DY)
*     CALL TO ARCSET INITIALIZES PEN POSITION
    8 CALL ARCSET(H,N,0)
*     CONFIGURE DESIRED SEQUENCE OF LONG AND SHORT DASHES WITH CALLS TO
*       ARCPLT WHICH EVALUATES INTERPOLATED CURVE AND CAUSES PEN MOTION
      IF(L1.EQ.0)GO TO 18
      IF(L4.EQ.0)GO TO 20
```

```
*  CALCULATE REVISED POINT SPACING AND NUMBER OF CYCLES FOR INTEGER
*    NUMBER OF CYCLES PLUS TAIL OF LONG DASHES
      NIC=L2*(L3+L1)+L4*(L5+L1)
      NIT=L2*(L3+L1)-L1
      NIN=S(N)/SS
      NCYC=NIN/NIC
      NIN=NCYC*NIC+NIT
      DS=S(N)/NIN
      DL1=L1*DS
*  GENERAL LINE-SEQUENCE CYCLE LOOP IS 14-LOOP
      DO 14 ICYC=1,NCYC
      DO 12 I2=1,L2
      CALL ARCPLT(DS,L3,2)
   12 CALL ARCPLT(DL1,1,3)
      DO 14 I4=1,L4
      CALL ARCPLT(DS,L5,2)
   14 CALL ARCPLT(DL1,1,3)
*  LOOP FOR TERMINATING TAIL OF LONG DASHES IS 16-LOOP
      DO 16 I2=1,L2
      CALL ARCPLT(DS,L3,2)
   16 CALL ARCPLT(DL1,1,3)
      RETURN
*  PLOT SOLID LINE
   18 NIN=S(N)/SS
      DS=S(N)/NIN
      CALL ARCPLT(DS,NIN,2)
      RETURN
*  PLOT LINE OF LONG DASHES ONLY
   20 NIC=L3+L1
      NIN=S(N)/SS
      NCYC=NIN/NIC
      NIN=NCYC*NIC-L1
      DS=S(N)/NIN
      DL1=L1*DS
      DO 22 ICYC=1,NCYC
      CALL ARCPLT(DS,L3,2)
   22 CALL ARCPLT(DL1,1,3)
      RETURN
      END              SUBROUTINE LINSEQ
```

Listing of Subroutine ARCPLT

```
      SUBROUTINE ARCPLT(DS,L,IPEN)
*    CURVE IS CALCULATED FROM SPLINE AND ARC OF LONG OR SHORT DASH IS
*      PLOTTED PEN UP OR DOWN
      COMMON/ARCPAR/S(101)/SCALEX/X(101)/SCALEY/Y(101)
      COMMON/SLOPEX/DX(101)/SLOPEY/DY(101)
      IF(H.NE.0) GO TO 4
      DO 2 I=1,L
      SP=SP+DS
      XP=SPIFUN(SP,N,S,X,DX)
      YP=SPIFUN(SP,N,S,Y,DY)
    2 CALL LIMPLT(XP,YP,IPEN)
      RETURN
    4 DO 6 I=1,L
      SP=SP+DS
      XP=SPIFUN(SP,N,S,X,DX)
      XDOT=SPIDER(SP,N,S,X,DX)
      YP=SPIFUN(SP,N,S,Y,DY)
      YDOT=SPIDER(SP,N,S,Y,DY)
      HS=H/SQRT(XDOT*XDOT+YDOT*YDOT)
      XP=XP+HS*YDOT
      YP=YP-HS*XDOT
    6 CALL LIMPLT(XP,YP,IPEN)
      RETURN

      ENTRY ARCSET
*    ENTRY POINT ARCSET INITIALIZES H AND N, AND MOVES RAISED PEN TO
*      BEGINNING OF CURVE
      H=DS
      N=L
      SP=0
      CALL LIMPLT(X,Y,3)
      IF(H.EQ.0)  RETURN
      HS=H/SQRT(DX*DX+DY*DY)
      XP=X+HS*DY
      YP=Y-HS*DX
      CALL LIMPLT(XP,YP,3)
      RETURN
      END             SUBROUTINE ARCPLT
```

Listing of Subroutine PRETRP

```
      SUBROUTINE PRETRP(N,SI)
*   N X,Y POINTS ARE INTERPOLATED AS Y = Y(X) TO GIVE 2*N-1 POINTS IF X
*     IS MONOTONICALLY INCREASING.  IF NOT, POINTS ARE INTERPOLATED AS
*     X=X(Y) IF Y IS MONOTONICALLY INCREASING, OTHERWISE RETURN.
      COMMON/ARCPAR/S(101)/SCALEX/X(101)/SCALEY/Y(101)
      COMMON/SLOPEX/DX(101)/SLOPEY/DY(101)
      SI=-SI
      DO 1 I=2,N
      IF(X(I).LE.X(I-1)) GO TO 4
    1 CONTINUE
      DO 2 I=1,N
      S(I)=X(I)
    2 DY(I)=Y(I)
      CALL SPISET(N,S,DY,DX,RMSX)
      DO 3 I=2,N
      I1=I+I-2
      I2=I1+1
      X(I2)=S(I)
      Y(I2)=DY(I)
      X(I1)=.5*(S(I-1)+S(I))
    3 Y(I1)=SPIFUN(X(I1),N,S,DY,DX)
      N=2*N-1
      RETURN
    4 DO 5 I=2,N
      IF(Y(I).LE.Y(I-1)) RETURN
    5 CONTINUE
      DO 6 I=1,N
      S(I)=Y(I)
    6 DX(I)=X(I)
      CALL SPISET(N,S,DX,DY,RMSY)
      DO 7 I=2,N
      I1=I+I-2
      I2=I1+1
      Y(I2)=S(I)
      X(I2)=DX(I)
      Y(I1)=.5*(S(I-1)+S(I))
    7 X(I1)=SPIFUN(Y(I1),N,S,DX,DY)
      N=2*N-1
      RETURN
      END             SUBROUTINE PRETRP
```

Listing of Subroutine SPISET

```
      SUBROUTINE SPISET(N,X,Y,D)
*******************************************************************************
*  THIS SUBROUTINE COMPUTES DERIVATIVES OF A NATURAL CUBIC SPLINE FOR         *
*  SUBSEQUENT USE BY FUNCTION SUBPROGRAMS SPIFUN AND SPIDER.                  *
*  STATEMENT 3 IMPLEMENTS EQN (44), PAGE 163 OF RALSTON + WILF, VOL 2         *
*  WITH NOTATION CHANGES NOTED BELOW,                                         *
*                    D(I-1) FOR U(I)                                          *
*                    W      FOR OMEQA = 4.*(2.-SQRT(3.))                      *
*                    Y(I)   FOR G(I)                                          *
*                    X(I)   FOR B(I)                                          *
*  THE DO LOOPS IN THE SUBPROGRAM PERFORM THE FOLLOWING TASKS,                *
*    1 LOOP REPLACES X BY DIFFERENCES AND Y BY DIVIDED DIFFERENCES.           *
*    2 LOOP REPLACES X BY SUBDIAGONALS (EQN (43) OF REF), Y BY 3* SECOND      *
*       DIVIDED DIFFERENCES, AND PUTS STARTING SECOND DERIVATIVES IN D.       *
*    3 LOOP SOLVES EQN (41) OF REF BY OVERRELAXATION USING JMAX               *
*         ITERATIONS (DEFAULT VALUE OF JMAX IS 20).                           *
*    4 LOOP UNDOES 2 LOOP.                                                    *
*    5 LOOP REPLACES SECOND DERIVATIVES BY FIRST DERIVATIVES (AVERAGE OF      *
*         LEFT AND RIGHT).                                                    *
*    7 LOOP COMPUTES RMS JUMP IN SECOND DERIVATIVES IF IFRMS=1.               *
*    9 LOOP UNDOES 1 LOOP.                                                    *
*******************************************************************************
      DIMENSION X(1),Y(1),D(1)
      COMMON/CSPISET/JMAX,IFRMS,RMS
      DATA JMAX,IFRMS,R6,W/20,0,.16666666666667,1.07179676972449/
      D(1)=D(N)=0
      DO 1 II=2,N
      I=N+2-II
      X(I)=X(I)-X(I-1)
    1 Y(I)=(Y(I)-Y(I-1))/X(I)
      IF(N.EQ.2) GO TO 5
      DO 2 II=3,N
      I=N+3-II
      D(I-1)=2.*(Y(I)-Y(I-1))/(X(I)+X(I-1))
      Y(I)=1.5*D(I-1)
    2 X(I)=.5*X(I-1)/(X(I)+X(I-1))
      DO 3 J=1,JMAX
      DO 3 I=3,N
    3 D(I-1)=W*(Y(I)-X(I)*D(I-2)-(.5-X(I))*D(I))-(W-1.)*D(I-1)
      DO 4 I=3,N
      X(I)=X(I-1)*(.5/X(I)-1.)
    4 Y(I)=.33333333333333*Y(I)*(X(I)+X(I-1))+Y(I-1)
    5 SAVE=Y(2)-R6*X(2)*(2.*D(1)+D(2))
```

```
      DO 6 I=2,N
      C=R6*X(I)
      S1=Y(I)-C*(2.*D(I-1)+D(I))
      S2=Y(I)+C*(D(I-1)+2.*D(I))
      D(I-1)=.5*(SAVE+S1)
    6 SAVE=S2
      D(N)=SAVE
      IF(IFRMS.NE.1) GO TO 8
      RMS=0
      IF(N.EQ.2) GO TO 8
      DO 7 I=3,N
      C=2.*D(I-1)
    7 RMS=RMS+((C+D(I-2)-3.*Y(I-1))/X(I-1)+(D(I)+C-3.*Y(I))/X(I))**2
      RMS=2.*SQRT(RMS/N)
    8 DO 9 I=2,N
      Y(I)=Y(I)*X(I)+Y(I-1)
    9 X(I)=X(I)+X(I-1)
      RETURN
      END            SUBROUTINE SPISET
```

Listing of Function Subprogram SPIFUN

```
      FUNCTION SPIFUN(XP,N,X,Y,D)
      DIMENSION X(1),Y(1),D(1)
*  EVALUATES A NATURAL CUBIC SPLINE USING SLOPE ARRAY D CALCULATED BY
*     SPISET AND USING THE INPUT DATA ARRAYS X AND Y.
      IF(XP.LE.X(1)) GO TO 6
      DO 2 I=2,N
      IF(XP.LT.X(I)) GO TO 4
    2 CONTINUE
      SPIFUN=Y(N)+D(N)*(XP-X(N))
      RETURN
    4 C1=1./(X(I)-X(I-1))
      C2=X(I)-XP
      C3=XP-X(I-1)
      C4=C2*C1
      C5=C3*C1
      SPIFUN=C5*C5*((1.+2.*C4)*Y(I)-C2*D(I))
     , +C4*C4*((1.+2.*C5)*Y(I-1)+C3*D(I-1))
      RETURN
    6 SPIFUN=Y(1)-D(1)*(X(1)-XP)
      RETURN
      END              REAL FUNCTION SPIFUN
```

Listing of Function Subprogram SPIDER

```
      FUNCTION SPIDER(XP,N,X,Y,D)
      DIMENSION X(1),Y(1),D(1)
*  EVALUATES THE FIRST DERIVATIVE OF A NATURAL CUBIC SPLINE USING SLOPE
*     ARRAY D CALCULATED BY SPISET AND USING THE INPUT DATA ARRAYS X AND Y
      IF(XP.LE.X(1)) GO TO 6
      DO 2 I=2,N
      IF(XP.LT.X(I)) GO TO 4
    2 CONTINUE
      SPIDER=D(N)
      RETURN
    4 C1=1./(X(I)-X(I-1))
      C2=X(I)-XP
      C3=XP-X(I-1)
      C4=2.*C2-C3
      C5=2.*C3-C2
      SPIDER=C1*C1*(C3*(2.*(1.+C1*C4)*
     , Y(I)-C4*D(I))-C2*(2.*(1.+C1*C5)*Y(I-1)+C5*D(I-1)))
      RETURN
    6 SPIDER=D(1)
      RETURN
      END              REAL FUNCTION SPIDER
```

Listing of Subroutine SMOOTHC

```
      SUBROUTINE SMOOTHC(N,X,Y,T)
*  THE Y ARRAY IS SMOOTHED BY A LOCAL FIVE POINT LEAST SQUARES CUBIC
*     WEIGHTED BY W
      DIMENSION X(1),Y(1),T(1)
      IF(N.LT.5)RETURN
      S=(T*(X(N)-X(1))/N)**2
      DO 4 L=1,N
      K=MINO(N-4,MAXO(1,L-2))
      K4=K+4
      DO 1 I=1,20
    1 T(I)=0.
      DO 3 M=K,K4
      W=1./(S+(X(L)-X(M))**2)
      R=1.
      DO 3 I=1,4
      RR=1.
      DO 2 J=1,4
      T(I+J*4-4)=T(I+J*4-4)+R*RR*W
    2 RR=RR*X(M)
      T(I+16)=T(I+16)+R*Y(M)*W
    3 R=R*X(M)
      CALL CHLSKYS(T,4,T(17),1,4)
      M=1+MOD(L-1,5)
      IF(L.GT.5) Y(L-5)=T(M+20)
      T(M+20)=0.
      R=1.
      DO 4 J=1,4
      T(M+20)=T(M+20)+R*T(J+16)
    4 R=R*X(L)
      DO 5 L=1,5
      ML=1+MOD(M+L-1,5)
    5 Y(N-5+L)=T(ML+20)
      RETURN
      END            SUBROUTINE SMOOTHC
```

Listing of Subroutine LIMPLT

```
      SUBROUTINE LIMPLT(X,Y,IPEN)
*  CAUSES PEN MOTION IF (X,Y) IS IN PREDECLARED FRAME
      DIMENSION X(1),Y(1)
      IPLM=IPEN
      IF(X.GT.XX) GO TO 1
      IF(X.LT.XN) GO TO 1
      IF(Y.GT.YX) GO TO 1
      IF(Y.LT.YN) GO TO 1
      IF(IFLAG.EQ.1)IPLM=3
      CALL CALPLT(X,Y,IPLM)
      IFLAG=0
      RETURN
    1 IFLAG=1
      RETURN

      ENTRY LIMSET
*  ENTRY POINT LIMSET IS USED TO PASS FRAME LIMITS (INCHES)
      IFLAG=IPEN
      XN=X
      YN=Y
      XX=X(2)
      YX=Y(2)
      RETURN
      END              SUBROUTINE LIMPLT
```

Listing of Subroutine CHLSKYS

```
      SUBROUTINE CHLSKYS(A,N,B,M,NX)
      DIMENSION A(NX,1),B(NX,1)
*  CHOLESKY DECOMPOSITION IS USED TO SOLVE THE MATRIX EQUATION AX = B,
*    WHERE THE COEFFICIENT MATRIX, A, IS SYMMETRIC.  ON OUTPUT X IS
*    STORED IN B.
      IF(N.EQ.1) GO TO 6
      DO 2 I=2,N
      I1=I-1
      DO 2 J=I,N
      DO 2 L=1,I1
    2 A(I,J)=A(I,J)-A(L,I)*A(L,J)/A(L,L)
      DO 5 K=1,M
      DO 3 I=2,N
      I1=I-1
      DO 3 L=1,I1
    3 B(I,K)=B(I,K)-A(L,I)*B(L,K)/A(L,L)
      DO 4 I=2,N
      I1=I-1
      DO 4 L=1,I1
      NI=N-I1
      NL=N+1-L
    4 B(NI,K)=B(NI,K)-A(NI,NL)*B(NL,K)/A(NL,NL)
      DO 5 I=1,N
    5 B(I,K)=B(I,K)/A(I,I)
      RETURN
    6 A(1,1)=1./A(1,1)
      DO 7 L=1,M
    7 B(1,L)=A(1,1)*B(1,L)
      RETURN
      END              SUBROUTINE CHLSKYS
```

Listing of Subroutine ARCALC

```
      SUBROUTINE ARCALC(N,IFLAG)
*    ARCALC CALCULATES THE S(I) ARRAY OF APPROXIMATE ARC LENGTHS
      COMMON/ARCPAR/S(101)/SCALEX/X(101)/SCALEY/Y(101)
      R24=.0416666666666667
      S(1)=0
      S(2)=B1=SQRT((X(2)-X(1))**2+(Y(2)-Y(1))**2)
      IF(N.EQ.2) RETURN
      N1=N-1
      HS=H1=0
      IF(IFLAG.EQ.0) GO TO 1
      B2=SQRT((X(1)-X(N1))**2+(Y(1)-Y(N1))**2)
      C1=SQRT((X(2)-X(N1))**2+(Y(2)-Y(N1))**2)
      A1=(X(1)-X(N1))*(Y(2)-Y(N1))-(X(2)-X(N1))*(Y(1)-Y(N1))
      HS=H1=A1/(B2*B1*C1)
    1 DO 2 I=2,N1
      IP=I+1
      IN=I-1
      B=SQRT((X(IP)-X(I))**2+(Y(IP)-Y(I))**2)
      C=SQRT((X(IP)-X(IN))**2+(Y(IP)-Y(IN))**2)
      A=(X(I)-X(IN))*(Y(IP)-Y(IN))-(X(IP)-X(IN))*(Y(I)-Y(IN))
      H=A/(B1*B*C)
      S(I)=S(I-1)+B1*(1.+R24*((H1+H)*B1)**2)
      B1=B
      H1=H
    2 CONTINUE
      S(N)=S(N-1)+B1*(1.+R24*((H1+HS)*B1)**2)
      RETURN
      END             SUBROUTINE ARCALC
```

Listing of Subroutine CONSEQ

```
      SUBROUTINE CONSEQ(XX,YY,NN,K,XLIM,YLIM,NSP,H,SI,L1,L2,L3,L4,L5)
*   SUBROUTINE FOR PLOTTING SEQUENCE OF SMOOTH CLOSED CURVES (CONTOURS)
*     USING LONG AND SHORT DASHES
      DIMENSION XX(1),YY(1),XLIM(1),YLIM(1)
      COMMON/ARCPAR/S(101)/SCALEX/X(101)/SCALEY/Y(101)
      COMMON/SLOPEX/DX(101)/SLOPEY/DY(101)
      COMMON/CSPISET/JMAX,IFRMS,RMS
      JMAX=6
      IF(NN.EQ.0)GO TO 8
      N=NN+1
      SS=SI
      KN=K*NN+1
      KNK=KN+K
      RX=1./XX(KNK)
      RY=1./YY(KNK)
*   COMPUTE SCALED  FRAME LIMITS FOR LIMPLT
      X=(XLIM-XX(KN))*RX
      X(2)=(XLIM(2)-XX(KN))*RX
      Y=(YLIM-YY(KN))*RY
      Y(2)=(YLIM(2)-YY(KN))*RY
      CALL LIMSET(X,Y,0)
*   COMPUTE SCALED X,Y ARRAYS
      DO 2 I=1,NN
      KI=K*I
      X(I)=(XX(KI)-XX(KN))*RX
    2 Y(I)=(YY(KI)-YY(KN))*RY
      X(N)=X(1)
      Y(N)=Y(1)
*   CALL ARCALC TO COMPUTE APPROXIMATE ARC LENGTH BETWEEN DATA POINTS
      CALL ARCALC(N,1)
      IF(NSP.EQ.0) GO TO 6
*   SMOOTHING OF X = X(S) AND Y = Y(S) IS PERFORMED BY CALLS TO SMOOTHP
      DO 4 I=1,NSP
      DX=I
      CALL SMOOTHP(N,S,X,DX)
      DX=I
    4 CALL SMOOTHP(N,S,Y,DX,DY)
      CALL ARCALC(N,1)
*   CALLS TO SPISETP INITIALIZE SPLINE DERIVATIVE ARRAYS DX AND DY FOR US
*     BY SPIFUNP AND SPIDERP
    6 CALL SPISETP(N,S,X,DX)
      CALL SPISETP(N,S,Y,DY)
*   CALL TO ARCSETP INITIALIZES PEN POSITION
    8 CALL ARCSETP(H,N,0)
*   CONFIGURE DESIRED SEQUENCE OF LONG AND SHORT DASHES WITH CALLS TO
*     ARCPLTP WHICH EVALUATES INTERPOLATED CURVE AND CAUSES PEN MOTION
      IF(L1.EQ.0)GO TO 13
      IF(L4.EQ.0)GO TO 20
```

```
*  CALCULATE REVISED POINT SPACING AND NUMBER OF CYCLES FOR INTEGER
*    NUMBER OF CYCLES
      NIC=L2*(L3+L1)+L4*(L5+L1)
      NIN=S(N)/SS
      NCYC=NIN/NIC
      NIN=NCYC*NIC+L1
      DS=S(N)/NIN
      DL1=L1*DS
*  GENERAL LINE-SEQUENCE CYCLE LOOP IS 14-LOOP
      DO 14 ICYC=1,NCYC
      DO 12 I2=1,L2
      CALL ARCPLTP(DS,L3,2)
   12 CALL ARCPLTP(DL1,1,3)
      DO 14 I4=1,L4
      CALL ARCPLTP(DS,L5,2)
   14 CALL ARCPLTP(DL1,1,3)
      RETURN
*  PLOT SOLID LINE
   18 NIN=S(N)/SS
      DS=S(N)/NIN
      CALL ARCPLTP(DS,NIN,2)
      RETURN
*  PLOT LINE OF LONG DASHES ONLY
   20 NIC=L3+L1
      NIN=S(N)/SS
      NCYC=NIN/NIC
      NIN=NCYC*NIC
      DS=S(N)/NIN
      DL1=L1*DS
      DO 22 ICYC=1,NCYC
      CALL ARCPLTP(DS,L3,2)
   22 CALL ARCPLTP(DL1,1,3)
      RETURN
      END              SUBROUTINE CONSEQ
```

Listing of Subroutine ARCPLTP

```
      SUBROUTINE ARCPLTP(DS,L,IPEN)
*  CURVE IS CALCULATED FROM SPLINE AND ARC OF LONG OR SHORT DASH IS
*    PLOTTED PEN UP OR DOWN
      COMMON/ARCPAR/S(101)/SCALEX/X(101)/SCALEY/Y(101)
      COMMON/SLOPEX/DX(101)/SLOPEY/DY(101)
      IF(H.NE.0) GO TO 4
      DO 2 I=1,L
      SP=SP+DS
      XP=SPIFUNP(SP,N,S,X,DX)
      YP=SPIFUNP(SP,N,S,Y,DY)
    2 CALL LIMPLT(XP,YP,IPEN)
      RETURN
    4 DO 6 I=1,L
      SP=SP+DS
      XP=SPIFUNP(SP,N,S,X,DX)
      XDOT=SPIDERP(SP,N,S,X,DX)
      YP=SPIFUNP(SP,N,S,Y,DY)
      YDOT=SPIDERP(SP,N,S,Y,DY)
      HS=H/SQRT(XDOT*XDOT+YDOT*YDOT)
      XP=XP+HS*YDOT
      YP=YP-HS*XDOT
    6 CALL LIMPLT(XP,YP,IPEN)
      RETURN

      ENTRY ARCSETP
*  ENTRY POINT ARCSETP INITIALIZES H AND N, AND MOVES RAISED PEN TO
*    BEGINNING OF CURVE
      H=DS
      N=L
      SP=0
      CALL LIMPLT(X,Y,3)
      IF(H.EQ.0)  RETURN
      HS=H/SQRT(DX*DX+DY*DY)
      XP=X+HS*DY
      YP=Y-HS*DX
      CALL LIMPLT(XP,YP,3)
      RETURN
      END             SUBROUTINE ARCPLTP
```

Listing of Subroutine SPISETP

```
      SUBROUTINE SPISETP(N,X,Y,D)
************************************************************************
*  THIS SUBROUTINE COMPUTES DERIVATIVES OF A PERIODIC CUBIC SPLINE FOR  *
*  SUBSEQUENT USE BY FUNCTION SUBPROGRAMS SPIFUNP AND SPIDERP.          *
*  THE SUBROUTINE SETS Y(N) TO Y(1).  THE PERIOD IS X(N)-X(1).          *
*  STATEMENT 4 IMPLEMENTS EQN (44), PAGE 163 OF RALSTON + WILF, VOL 2   *
*  WITH NOTATION CHANGES NOTED BELOW,                                   *
*                  D(I-1)  FOR U(I)                                     *
*                  W       FOR OMEQA = 4.*(2.-SQRT(3.))                 *
*                  Y(I)    FOR G(I)                                     *
*                  B       FOR B(I)                                     *
*  THE DO LOOPS IN THE SUBPROGRAM PERFORM THE FOLLOWING TASKS,          *
*    2 LOOP PUTS STARTING SECOND DERIVATIVES IN D.                      *
*    5 LOOP SOLVES EQN (41) OF REF BY OVERRELAXATION USING JMAX         *
*        ITERATIONS (DEFAULT VALUE OF JMAX IS 20).                      *
*    8 LOOP REPLACES SECOND DERIVATIVES BY FIRST DERIVATIVES (AVERAGE OF *
*        LEFT AND RIGHT).                                               *
*   10 LOOP COMPUTES RMS JUMP IN SECOND DERIVATIVES IF IFRMS=1.         *
************************************************************************
      DIMENSION X(1),Y(1),D(1)
      COMMON/CSPISET/JMAX,IFRMS,RMS
      DATA JMAX,IFRMS,R6,W/20,0,.1666666666666667,1.07179676972449/
      Y(N)=Y(1)
      P=X(N)-X(1)
      IF(N.EQ.2) GO TO 12
      N1=N-1
      D(1)=2.*((Y(2)-Y(1))/(X(2)-X(1))-(Y(1)-Y(N-1))/(X(1)-X(N-1)+P))/
     , (X(2)-X(N-1)+P)
      DO 2 I=2,N1
    2 D(I)=2.*((Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))/(X(I)-X(I-1)))/
     , (X(I+1)-X(I-1))
      D(N)=D(1)
      DO 6 J=1,JMAX
      B=.5*(X(1)-X(N-1)+P)/(X(2)-X(N-1)+P)
      D(1)=W*(3.*((Y(2)-Y(1))/(X(2)-X(1))-(Y(1)-Y(N-1))/(X(1)-X(N-1)+P))
     , /(X(2)-X(N-1)+P)-B*D(N-1)-(.5-B)*D(2))-(W-1.)*D(1)
      DO 4 I=2,N1
      B=.5*(X(I)-X(I-1))/(X(I+1)-X(I-1))
    4 D(I)=W*(3.*((Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))/(X(I)-X(I-1)
     , ))/(X(I+1)-X(I-1))-B*D(I-1)-(.5-B)*D(I+1))-(W-1.)*D(I)
    6 D(N)=D(1)
```

```
      SAVE=(Y(2)-Y(1))/(X(2)-X(1))-R6*(X(2)-X(1))*(2.*D(1)+D(2))
      DO 8 I=1,N1
      YDIF=(Y(I+1)-Y(I))/(X(I+1)-X(I))
      XDIF=R6*(X(I+1)-X(I))
      S1=YDIF-XDIF*(2.*D(I)+D(I+1))
      S2=YDIF+XDIF*(D(I)+2.*D(I+1))
      D(I)=.5*(SAVE+S1)
    8 SAVE=S2
      D(N)=D(1)
      IF(IFRMS.NE.1) RETURN
      S1=1./(X(1)-X(N-1)+P)
      S2=1./(X(2)-X(1))
      RMS=(S1*(2.*D(1)+D(N-1)-3.*S1*(Y(1)-Y(N-1)))+S2*(D(2)+2.*
     , D(1)-3.*S2*(Y(2)-Y(1))))**2
      DO 10 I=2,N1
      S1=1./(X(I)-X(I-1))
      S2=1./(X(I+1)-X(I))
   10 RMS=RMS+(S1*(2.*D(I)+D(I-1)-3.*S1*(Y(I)-Y(I-1)))+S2*(D(I+1)+2.*
     , D(I)-3.*S2*(Y(I+1)-Y(I))))**2
      RMS=2.*SQRT(RMS/(N-1.))
      RETURN
   12 D(1)=D(2)=0
      RETURN
      END             SUBROUTINE SPISETP
```

Listing of Function Subprogram SPIFUNP

```
      FUNCTION SPIFUNP(XP,N,X,Y,D)
*  EVALUATES A PERIODIC CUBIC SPLINE USING SLOPE ARRAY D CALCULATED BY
*    SPISETP AND USING THE INPUT DATA ARRAYS X AND Y
      DIMENSION X(1),Y(1),D(1)
      XX=XP
      P=X(N)-X(1)
    2 IF(XX.GE.X(1)) GO TO 4
      XX=XX+P
      GO TO 2
    4 IF(XX.LT.X(N)) GO TO 6
      XX=XX-P
      GO TO 4
    6 DO 8 I=2,N
      IF(XX.LT.X(I)) GO TO 10
    8 CONTINUE
   10 C1=1./(X(I)-X(I-1))
      C2=X(I)-XX
      C3=XX-X(I-1)
      C4=C2*C1
      C5=C3*C1
      SPIFUNP=C5*C5*((1.+2.*C4)*Y(I)-C2*D(I))
     , +C4*C4*((1.+2.*C5)*Y(I-1)+C3*D(I-1))
      RETURN
      END              REAL FUNCTION SPIFUNP
```

Listing of Function Subprogram SPIDERP

```
      FUNCTION SPIDERP(XP,N,X,Y,D)
*  EVALUATES THE FIRST DERIVATIVE OF A PERIODIC CUBIC SPLINE USING SLOPE
*    ARRAY D CALCULATED BY SPISETP AND USING THE INPUT DATA ARRAYS X AND Y
      DIMENSION X(1),Y(1),D(1)
      XX=XP
      P=X(N)-X(1)
    2 IF(XX.GE.X(1)) GO TO 4
      XX=XX+P
      GO TO 2
    4 IF(XX.LT.X(N)) GO TO 6
      XX=XX-P
      GO TO 4
    6 DO 8 I=2,N
      IF(XX.LT.X(I)) GO TO 10
    8 CONTINUE
   10 C1=1./(X(I)-X(I-1))
      C2=X(I)-XX
      C3=XX-X(I-1)
      C4=2.*C2-C3
      C5=2.*C3-C2
      SPIDERP=C1*C1*(C3*(2.*(1.+C1*C4)*
     , Y(I)-C4*D(I))-C2*(2.*(1.+C1*C5)*Y(I-1)+C5*D(I-1)))
      RETURN
      END              REAL FUNCTION SPIDERP
```

Listing of Subroutine SMOOTHP

```
      SUBROUTINE SMOOTHP(N,X,Y,T)
      DIMENSION X(1),Y(1),T(1),SAVE(3)
*   THE Y ARRAY IS SMOOTHED BY A LOCAL FIVE POINT LEAST SQUARES CUBIC
*     WEIGHTED BY W (Y IS A PERIODIC FUNCTION OF X)
      IF(N.LT.5)RETURN
      P=X(N)-X(1)
      SAVE(2)=Y(2)
      SAVE(3)=Y(3)
      S=(T*(X(N)-X(1))/N)**2
      DO 6 L=1,N
      DO 1 I=1,20
    1 T(I)=0.
      DO 5 K=1,5
      M=L+K-3
      XM=X(M)
      YM=Y(M)
      IF(M.GT.0) GO TO 2
      MM=M+N-1
      YM=Y(MM)
      XM=X(MM)-P
    2 IF(M.LE.N) GO TO 3
      MM=M+1-N
      YM=SAVE(MM)
      XM=X(MM)+P
    3 W=1./(S+(X(L)-XM)**2)
      R=1.
      DO 5 I=1,4
      RR=1.
      DO 4 J=1,4
      T(I+J*4-4)=T(I+J*4-4)+R*RR*W
    4 RR=RR*XM
      T(I+16)=T(I+16)+R*YM*W
    5 R=R*XM
      CALL CHLSKYS(T,4,T(17),1,4)
      M=1+MOD(L-1,5)
      IF(L.GT.5) Y(L-5)=T(M+20)
      T(M+20)=0.
      R=1.
      DO 6 J=1,4
      T(M+20)=T(M+20)+R*T(J+16)
    6 R=R*X(L)
      DO 7 L=1,5
      ML=1+MOD(M+L-1,5)
    7 Y(N-5+L)=T(ML+20)
      RETURN
      END           SUBROUTINE SMOOTHP
```

REFERENCES

1. Greville, T. N. E.: Spline Functions, Interpolation, and Numerical Quadrature. Mathematical Methods for Digital Computers, Vol. II, Anthony Ralston and Herbert S. Wilf, eds., John Wiley & Sons, Inc., c.1967, pp. 156-168.

2. Ahlberg, J. H.; Nilson, E. N.; and Walsh, J. L.: The Theory of Splines and Their Applications. Academic Press, Inc., c.1967.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958