INTERACTIVE FORTRAN GRAPHICS PLOTTING ROUTINES FOR THE DIGITAL EQUIPMENT CORPORATION GT-44 SYSTEM

by

Lynden Edward Holsclaw

A Thesis Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

1 9 7 7

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Gaylen Edward Hohclaw

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

John V. Wait
JOHN V. WAIT
Professor of Electrical Engineering

June 1 '77
Date

PREFACE

A computer graphics system provides the user with the ability to generate immediate visual displays of solutions to a wide variety of problems. Development of the software package described in this paper was undertaken to provide X-Y plotting capability on the Digital Equipment Corporation GT-44 computer graphics system at The University of Arizona. Originally intended as a visual display mechanism to complement DARE-11, a program which solves differential equations, the package has been written completely in general, and is applicable to any problem requiring CRT displays of X-Y plots in various formats.

Special acknowledgment is due Dr. J. V. Wait and Dr. Granino Korn of The University of Arizona for their advice and assistance on this project.

TABLE OF CONTENTS

TABLE OF CONTENTS--*Continued*

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS--Continued

ABSTRACT

The software package described herein was written for use on the Digital Equipment Corporation GT-44 computer graphics system to produce X-Y point plots from real-number input arrays representing up to four functions and an independent variable.

The package features automatic scaling of the input arrays, combined with automatic selection of only that portion of the coordinate grid applicable to the data, to produce maximum resolution from a user-defined number of points per plot. A limited set of keyboard commands provide interactive run-time control of some of the plot characteristics, including a 1/2 reduction in plot size to allow photographing with a conventional oscilloscope camera.

Five major subroutines are included, providing a choice of various plot formats to best display a particular problem. Seven additional routines perform the necessary scaling, labeling, and grid pattern generation.

CHAPTER 1

INTRODUCTION

Development of the software package described in this study was first proposed as an adjunct to DARE-11, a simulation software system developed at The University of Arizona to solve differential equations (Korn and Wait, 1977).

DARE-11 has previously been implemented on the Digital Equipment Corporation GT-44 computer graphics system, which includes a PDP-11/40 central processor, VT-11 CRT graphics display subsystem, and associated peripherals. The existing software has had some graphics capability; however, major improvements have been needed. The requirements included providing the capability for real-time alterations to the display and the ability to replot data using various format parameters, as well as taking advantage of the increases speed and resolution afforded by CRT graphics as opposed to line printer plots. Additionally, FORTRAN was to be used, with the calling statements compatible with existing line printer plotting routines.

The requirements outlined above are satisfied by the FORTRAN subroutines described in Chapter 3. It should be emphasized at this point that although the routines were developed with application to DARE-11 in mind, generality in the software design was the foremost consideration. The result is a comprehensive utility package available for use in connection with any problem requiring X-Y CRT plots of a given set of data.

Control of the CRT display itself from a FORTRAN environment is made possible via calls to a set of previously written subroutines under the file name, GTFOR6 (Conley, 1975). These routines provide FORTRAN-callable commands to the display processor to generate absolute and relative X-Y points and vectors (lines), adjust intensities, flash portions of displays, enable light pen operations, etc. This set of routines is used extensively, and must therefore be linked with the plotting routines and the user object program in forming an executable program. This is further described in Chapter 5.

CHAPTER 2

SOFTWARE DESIGN CONSIDERATIONS

Within the constraint that the calling statements for the CRT routines be compatible with those for the line printer routines, considerable freedom existed in program design. Analysis of the design criteria revealed conflicting requirements in some cases, which necessitated making certain compromises to maintain generality. This chapter will discuss these points and present the overal design philosophy.

Data Scaling and Coordinate Grid Format

A method of formatting one coordinate grid usable with any set of data extending over an extremely wide range in both abscissa and ordinate posed the most difficult problem. The most straight-forward approach is to center the X and Y axes on the screen, scale the data to the 0 to 1023 addressable point range required by the display processor, and let the plot fall where it may. In many cases, this approach would be perfectly satisfactory; however, it is deficient in several aspects.

Consider the function, log (x), in which all data falls in the right half of the plane. At least half of the available plot area is unused, area that could be used for increased resolution. Worse yet is the situation in which all data lies within one quadrant.

The above approach also produces unsatisfactory results when the range of the data being plotted is small relative to its absolute magnitude. That is, the boundary values of the plot are fixed by the greatest absolute value of the data, and a relatively narrow range of values will be compressed toward the boundaries. When scaled to a range of 0 to 1023 (displaced from the range, -511 to + 511), such a set of data may produce only a few points to be displayed. In the extreme case, a set of, say, 50 data points spread over a range of 10 about a magnitude of 10^{20} would result in a single point after scaling.

Another approach considered was to have multiple routines which placed the axes to show one or two quadrants each. This would satisfy a much broader range of problems, including the frequent case of displaying functions of time from zero to some limit, by displaying quadrants I and IV. However, this requires prior knowledge of the data to determine which routine to call and does not solve the scaling problem.

Perhaps the best approach is to first scale and plot the input data to fit the available diplay area completely and retroactively generate the coordinate grid "over" the data. In this way, plot resolution, a primary advantage of CRT displays, is maximized. The zero axes of the grid, if they are displayed at all, are then positioned as dictated by the data itself. This is the approach that was adopted for this project.

Note that with this method, when labeled grid lines are generated, their positioning and labels are likewise determined by the data. It is this, along with the requirement that the number of grid lines be user-specified, that forces this approach to remain a compromise under certain situations. One such case occurs when the data require placing one or both of the axes off-center, usually at a point such that the remaining screen area cannot be evenly divided to display a specified number of lines. Another occurs when the axis is centered, but an even number of lines has been specified. This problem is further examined and exemplified in Chapter 5. In each case, the resulting display may contain one more line than was requested in the user program. Avoiding this would require re-positioning the plots, with some unavoidable degradation in resolution, which was considered an unacceptable sacrifice.

Plot Distinction

A second problem arose in distinguishing one plot from another, since it was desired to plot up to four functions against a common independent variable. Several alternatives for accomplishing this were considered, ranging from simply assuming the user would have sufficient knowledge of the data to enable recognizing individual function characteristics to passing a literal text string identifier and automatically displaying it near (but not on) the respective curve. The latter choice, although ideal, was quickly dismissed, as it would require additional parameters, and would thus be incompatible with the line printer routines. Utimately, the software step-wise adjustable intensity feature of the VT-11 display processor was utilized; successive functions in the calling statement are plotted in order with successively lower intensities. A legend showing the relative intensities and corresponding to the order in which the functions are plotted is displayed beneath the plot as an aid in identification. This, along with proper adjustment of the CRT manual intensity control as described in Chapter 5, allows easy and positive identification of the four possible plots.

Standardization and Portability

Insofar as possible, all routines have been written in A.N.S.I. standard FORTRAN, so as to be as portable as possible (American National Standards Institute, 1966). Structured programming techniques (Kreitzberg and Schneiderman, 1972) and self-documentation of the source files were given considerable attention. Detailed header comment blocks have been included with each routine to describe its function, arguments, local variables, etc. Also, provisions have been made to simplify modifications to the routines for different formats and display sizes as may be required for specific applications. A minimum number of such modifications should allow satisfactory execution of the package on a variety of computer graphics systems, although separate driver routines similar to GTFOR6 would be required.

CHAPTER 3

SUBROUTINE DESCRIPTIONS

This chapter describes in some detail each of the five user-callable subroutines, as well as those used in turn at further nested levels for specific purposes.

User-Callable Graph Subroutines

The following paragraphs provide descriptions that will be of interest to the general user of the subroutine package. Features common to all routines are discussed first. The reader is referred to Appendix A for source listings.

Common Features

Each GRAPH subroutine is called with from five to eight parameters, as follows:

1. The total (integer) number of arrays passed.
2. The integer number of points per array.
3. The integer number of grid lines desired. (Note this includes the boundary lines and zero axes.)
4. The real-number array representing the X-coordinates.
5. The real-number array representing the first set of Y-coordinates.

6. The real-number array representing the second set of Y-coordinates, if any.
7. The real-number array representing the third set of Y-coordinates, if any.
8. The real-number array representing the fourth set of Y-coordinates, if any.

The parameters are listed in order, except as provided in the descriptions for GRAPH3 and GRAPH4. Note that when calls are made with a negative value as the third parameter, the absolute value is taken as a number of grid lines to be displayed, but the entire plot is reduced in size by one-half to allow photographing.

Control of the CRT display itself is effected by generating entires in an integer display file array, IDISPL, which is maintained in common block DISFIL. Each GRAPH subroutine first initializes IDISPL(1) to zero, as required by the GTFOR6 package. Subsequent calls to GTFOR6 routines control the index and entries to IDISPL. When all entries for the plots, grid, labels, and text have been established by such calls, GTFOR6 routine DINIT is called, which initializes scanning of IDISPL by the display processor, causing the plot to be displayed. At this point, a message is output to the console and a routine continuation command is requested. The user may request a re-plot of the data reduced or expanded in size, a change in the vertical scale displayed in the case of GRAPH2 and GRAPH4, or termination.

When the request has been entered, DCLOSE is called to terminate scanning of IDISPL, followed by a re-plot or return of control to the calling program, as required.

GRAPH1

Subroutine GRAPH1 is the most straight-forward of the five routines. Up to four functions of a common independent variable may be plotted on a single coordinate system.

Horizontal scaling is determined by the minimum and maximum values in the "X" array. Vertical scaling is accomplished by finding the extremes among all of the "Y" arrays and scaling these arrays by a common factor, that is, the most positive function value determines the upper scaling limit, and the most negative value (which need not necessarily be from the same function) determines the lower limit. In this way, magnitude relationships among the functions are preserved, although individual plots will likely not fill the coordinate grid area.

A pictorial "graph paper" grid system is displayed with scale markings in "E"-format. The grid boundaries and zero axes (if shown) are intensified. Successive "Y" arrays in the calling statement are displayed with successively lower intensities, and a legend identifying each function is displayed beneath the plot.

A typical calling statement for this routine would appear as follows: CALL GRAPH1 (NARRYS, NPNTS, NLINES, X, FCN1, FCN2, FCN3, FCN4).

GRAPH2

Another system of scaling is employed in GRAPH2. Each "Y" array is individually auto-scaled as dictated by its own extreme values so as to fill the plot area; that is, the resolution of each plot is maximized. This technique is useful in situations where functions differ widely in amplitude range, and plotting by GRAPH1 would result in indiscernibly small plots of the lower magnitude functions. Using GRAPH2, each function is plotted full-size, and, although magnitude relationships are destroyed, absolute amplitudes of individual functions are still displayed by the vertical scale markings. Due to the limited screen area available, only one vertical scale is displayed at a time; however, the scale may be changed to reflect any corresponding function by keyboard commands at run time. A legend is displayed beneath the plot indicating which function the current scale markings correspond to.

The use of GRAPH2 is particularly effective to display time relationships (phase shift) among time-variant functions with like frequencies but dissimilar amplitudes, when magnitude relationships are not of immediate concern. Such relationships can now be derived from horizontal distances between plots at any vertical level, whereas such derivations using the technique of GRAPH1 can be made only at well-defined points, such as zero crossings or peak amplitude points, which may appear on a plot over a given range.

GRAPH2 is similar in all other aspects to GRAPH1, including calling statement format, grid generation, and plot identification. A typical calling statement would be: CALL GRAPH2 (NARRYS, NPNTS, NLINES, X, FCN1, FCN2, FCN3, FCN4).

GRAPH3

The purpose of GRAPH3 is to plot up to three functions against another function, such as might be done to produce a phase-diagram relating two state variables.

The calling statement differs slightly in that no independent variable array is passed; the fourth, fifth, and sixth arguments represent "Y" arrays to be plotted against the array named in the seventh. Note that if less than three functions are to be plotted, dummy arguments must be provided in order that the horizontal array appears as the seventh argument. This situation was unavoidable, as the calling statement was constrained as previously defined for the line printer routines.

Inasmuch as the function of GRAPH3 is identical to that of GRAPH1 except for the ordering of the input arguments, it is accomplished in exactly that manner; following a simple error check on the number of arrays provided, GRAPH3 simply calls GRAPH1 with the seventh original input argument moved to the fourth position, followed by the three vertical arrays. Thus, scaling and plot format are identical to those of GRAPH1.

A typical calling statement might appear as: CALL GRAPH3 (NARRYS, NPNTS, NLINGES, YFCN1, YFCN2, YFCN3, XFCN).

GRAPH4

The function of GRAPH4 is similar to that of GRAPH3, but the nested call is made to GRAPH2 to provide full-screen scaling of the functions. With calls to GRAPH4, up to three functions are plotted full-size against another function.

The use of GRAPH4 will probably be limited to the production of Lissajous-type patterns from functions of widely varying amplitudes. Since GRAPH2 is ultimately run, vertical scale selection and legend display are provided.

The calling statement format is the same as for GRAPH3: CALL GRAPH4 (NARRYS, NPNTS, NLINES, YFCN1, YFCN2, YFCN3, XFCN).

GRAPH5

GRAPH5 again provides the user the ability to generate up to four function plots of an independent variable, but now separate plots are displayed instead of a single plot with several functions superpositioned. The number of plots is automatically determined by the number of input arrays provided, and full screen utilization is ensured by varying the vertical dimension of each plot accordingly.

The calling statement format is the same as for GRAPH1 and GRAPH2. The coordinate grid(s) generated differ, of course, and, due to the restricted area available for labels, etc., only the extreme values of each array, horizontally and vertically, are shown by the scale markings.

This subroutine is useful to display phase relationships when simultaneous display of all magnitude ranges is desired.

A typical calling statement would appear as: CALL GRAPH5 (NARRYS, NPNTS, NLINES, X, FCN1, FCN2, FCN3, FCN4).

Support Subroutines

The following paragraphs describe the various support subroutines called from the user-callable graph routines.

MINMAX

To perform auto-scaling on the real-number input arrays to produce integer arrays in a specified range, it is necessary to search each array and find the minimum and maximum values. This is done in MINMAX using FORTRAN library routines, AMIN1 and AMAX1. The routine is called with four arguments in the following order:

1. The array to be searched.
2. The number of entries in the array.
3. The variable name for returning the minimum value.
4. The variable name for returning the maximum value.

As an example, to find the minimum and maximum values in an array named F2, consisting of 500 values, the calling statement might be: CALL MINMAX (F2, 500, F2MIN, F2MAX).

SCALE

The scaling subroutine itself scales a real-number input array to a specified integer range. The calling parameters, in order, are:

1. The lower bound of the integer scaling range.
2. The upper bound of the integer scaling range.
3. The real-number input array name.
4. The integer number of entries in the array.
5. The minimum real value in the input array.
6. The maximum real value in the input array.
7. The array name for returning the scaled array.

To continue the previous example, if it is now desired to scale the real array F2 to the integer range, 400 to 1000, the calling statement would be: CALL SCALE (400, 1000, F2, 500, F2MIN, F2MAX, IF2).

PLOT

After obtaining scaled integer arrays, the GRAPH routines next call PLOT once for each function plotted. This routine does the actual plot generation via calls to GTFOR6 routine POINTA. PLOT is called with parameters:

1. The scaled integer array representing the X-coordinates.
2. The scaled integer array representing the corresponding Y-coordinates.
3. The number of points (array entries) to be plotted.
4. The intensity (integer 0 —— 7) desired for the plot.

The example function above could now be plotted at maximum intensity against a previously scaled set of X-coordinates using the following calling statement: CALL PLOT (IX, IF2, 500, 7).

GRID1

Two types of coordinate grid displays are available. The first, called from GRAPH1, GRAPH2, GRAPH3, and GRAPH4, is generated by subroutine GRID1. One square grid system with numerically labeled lines and intensified borders and zero axes is displayed. The grid normally occupies that portion of the screen represented by the addressable points 149 through 1023 in each direction, leaving room at the left-hand side for vertical magnitude labels and at the bottom for horizontal magnitude labels and legends.

However, if the number of grid lines requested in the fifth parameter is a negative value, the entire plot is reduced in size by one-half, and the grid occupies points 404 through 767 in each direction.

The calling arguments, in order, are:

1. The minimum real-value X-coordinate.
2. The maximum real-value X-coordinate.
3. The minimum real-value Y-coordinate.
4. The maximum real-value Y-coordinate.
5. The integer number of grid lines to be displayed, including border lines and zero axes.

The following calling statement would be typical for producing a grid system for the single example function above: CALL GRID1 (XMIN, XMAX, F2MIN, F2MAX, 5).

GRID2

In the case of GRAPH5, GRID2 is called to generate up to four separate coordinate grids over corresponding plots. The vertical space used for each is adjusted according to the number of grids necessary, so as to make full use of the available display area. Control of the overall plot size is handled as in GRID1. The calling statement is similar, with the addition of four parameters:

6. The integer number representing the screen coordinate of the bottom edge of the plot being generated.

7. The vertical dimension, measured in addressable screen points, of each plot.
8. The screen coordinate representing the bottom and left-hand edges of the available area for the plots.
9. The screen coordinate representing the top and right-hand edges of the available area for the plots.

If, for example, three plots were being displayed using GRAPH5, reduced to one-half size, the calling statement to GRID2 for the first might appear as follows: CALL GRID2 (XMIN, XMAX, F1MIN, F1MAX, 5, 404, 114, 404, 767).

LEGEND

For each of the first four user-callable routines, a legend is displayed beneath the plot indicating which vertical input arrays are plotted and showing the corresponding intensity of each curve. This is effected by call to subroutine LEGEND, with arguments:

1. The total number of plots displayed.
2. The screen coordinate representing the bottom edge of the display area, used for positioning the legend.

A typical calling statement used when four plots are being displayed full-size would appear as: CALL LEGEND (4, 149).

NUMTXT

Although the GTFOR6 routine TEXT existed to display textual information on the graphics screen, it required literal ASCII character strings as input. No routine was available to generate visual display of a number from its binary representation in memory. For this purpose, NUMTXT, called by the GRID routines, was written.

To facilitate displaying numbers over a very wide range within a limited area on the screen, "E"-format was chosen. The literals '0' through '9' are stored in array NUM by DATA type declaration. The input number is repeatedly multiplied or divided by 10 until it is in the range 0.1 to 1.0. An exponent variable is adjusted accordingly. The resulting value is then multiplied by 10 and truncated, with the result used as an index for NUM, which in turn supplies the argument for TEXT. This is repeated according to the number of significant figures desired on the display, after which the exponent is displayed in a similar manner.

Two arguments are required in the calling statement:

1. The real-number value to be displayed.
2. The number of significant figures desired.

A typical calling statement might be: CALL NUMTXT (XMAX, 3).

CHAPTER 4

EXAMPLE PROGRAMS

Three test driver programs for the GRAPH subroutines were written for checkout and demonstration purposes. This chapter describes these programs and includes sample photographic results. Source listings for each program are provided in Appendix B.

Test Driver SINES1

The first test program generates arrays of points representing sine functions and displays them via calls to GRAPH1, GRAPH2, and GRAPH5. Various messages are output to the operator console at run time which provide step-by-step explanations of the displays appearing.

The first plot generated is of a single function, Y = 3.5 * SIN (2 * PI * T), and is plotted with 301 data points over the range T = -1.0 to +1.0. The result of the call to GRAPH1 is shown in Figure 1. Note that since the plot crosses both zero axes, they are shown intensified. As only one function is being displayed, it is scaled for maximum resolution within the available area.

Figure 2 shows the result of adding a constant of +5.0 to the function and re-plotting. The X-axis in this case is not shown at all, in order that maximum resolution

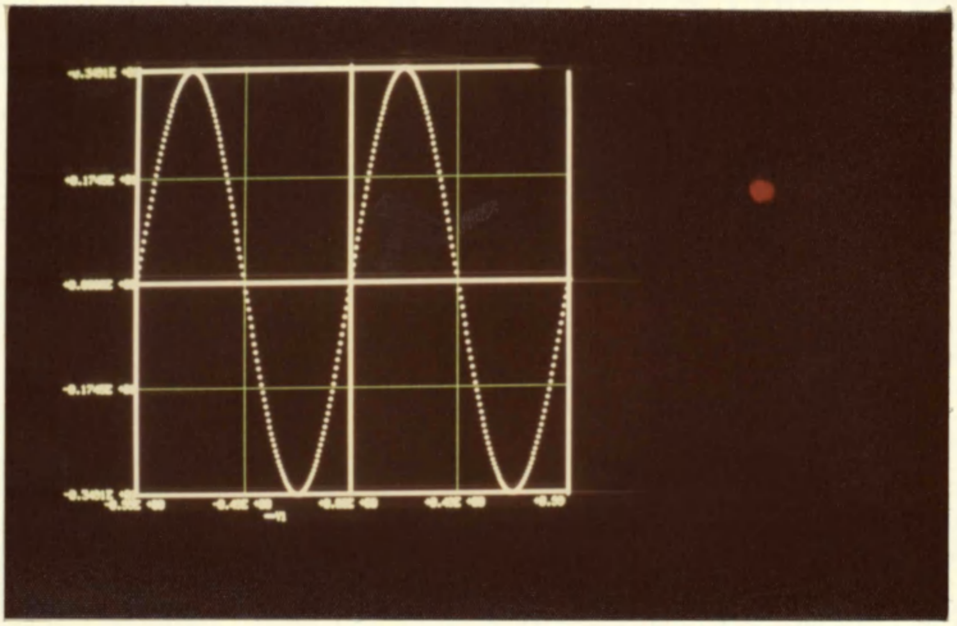

Figure 1. Plot of Y = 3.5 * SIN (2 * PI * T), over the range, T = -1.0 to +1.0, using GRAPH1.

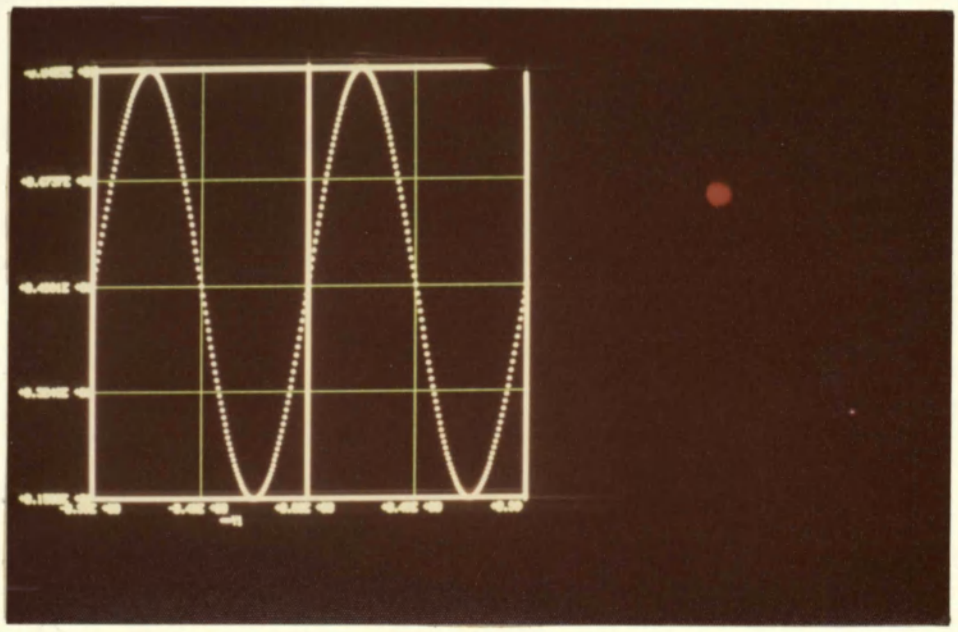

Figure 2. Plot of Y = 5.0 + 3.5 * SIN (2 * PI * T), over the range, T = -1.0 to +1.0, using GRAPH1.

be retained, and the vertical scale markings are changed accordingly.

For the third plot, the constant is changed to -9.0 and an arbitrary range of T = 5.8 to 7.8 is chosen. The resulting display is illustrated in Figure 3.

The next plot introduces a second function and exemplifies a situation where GRAPH1 produces less than satisfactory results. The data are meant to simulate the input and output signals of an amplifier with a voltage gain of 100 and some phase shift. The result of plotting with GRAPH1 is shown in Figure 4. Note that the input signal is too small to derive any information about either magnitude or phase shift.

The same data are more usefully displayed using GRAPH2; the results are shown in Figure 5. Now the phase shift is clearly visible, and absolute magnitudes are displayed by the vertical scale markings. Recall that with GRAPH2, the data may be re-plotted at run time with the vertical scale selected to correspond to any function plotted. Note the legend beneath the plot indicating the intensities of each plot and the vertical scale currently displayed.

Finally, the same data are plotted by call to GRAPH5. As shown in Figure 6, this results in separate plots which clearly show phase shift as well as the amplitude at the plot extremes in both directions.

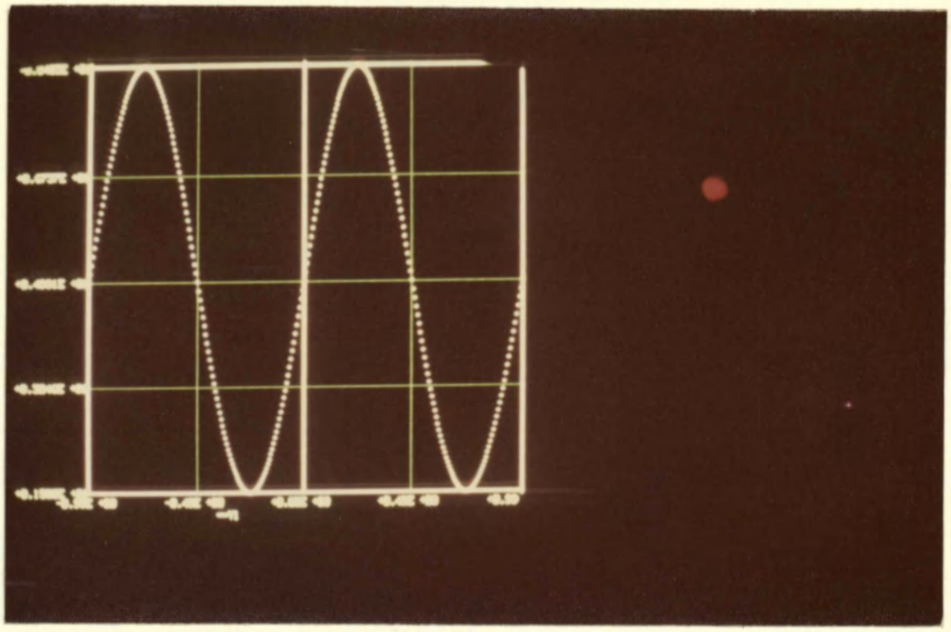

Figure 3. Plot of Y = -9.0 + 3.5 * SIN (2 * PI * T), over the range, T = 5.8 to 7.8, using GRAPH1.

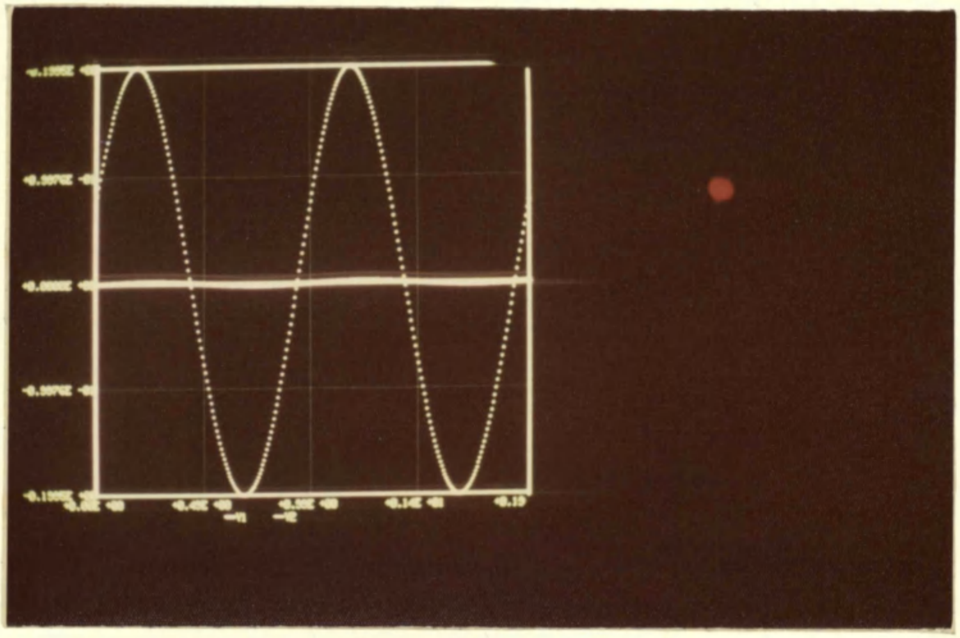

Figure 4. Functions of widely differing amplitudes plotted with GRAPH1.

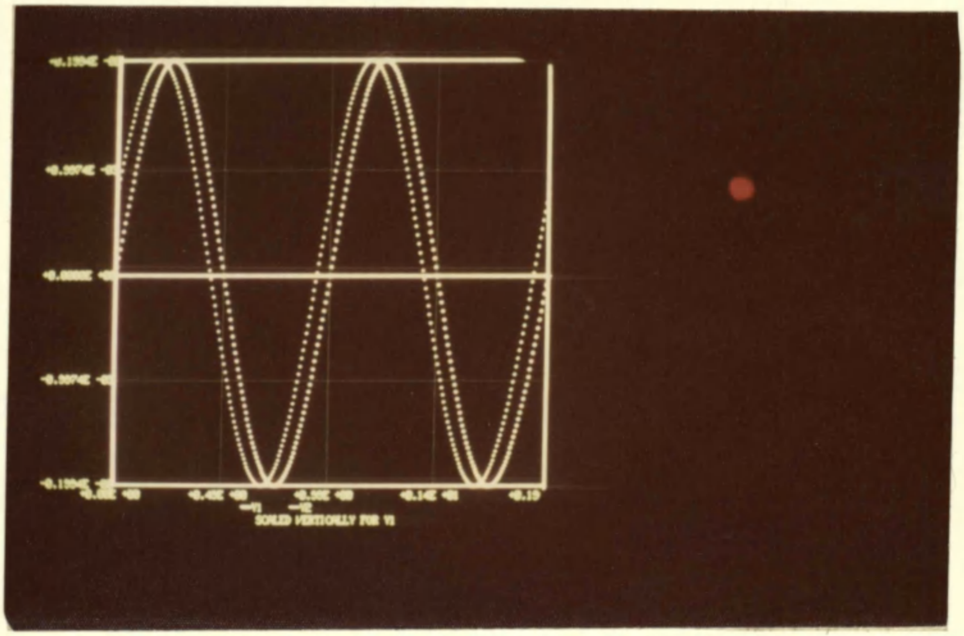

Figure 5. The data of Figure 4 plotted using GRAPH2.

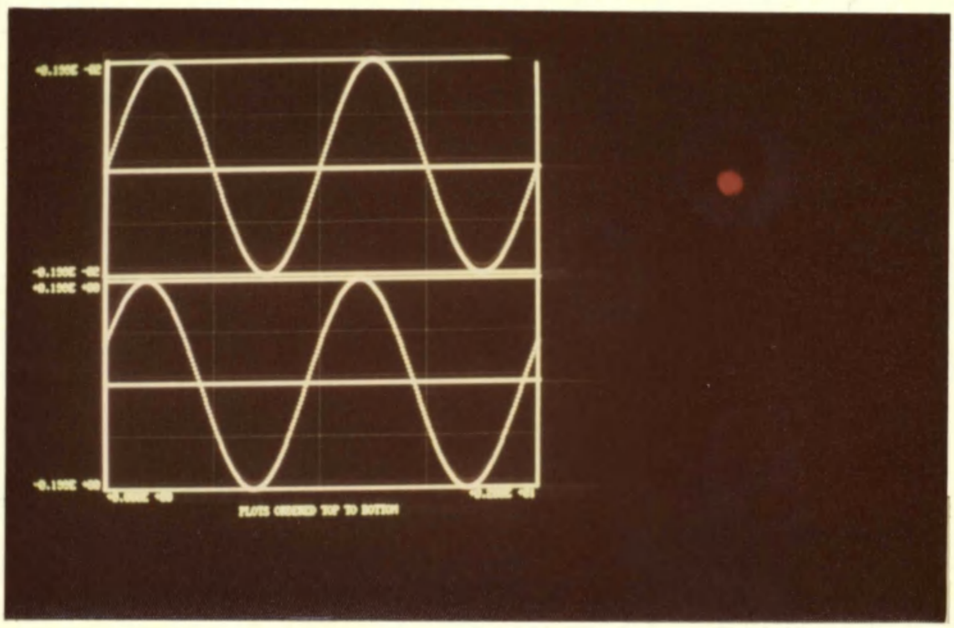

Figure 6. The data of Figure 4 plotted using GRAPH5.

Figure 7 illustrates the capability of reducing plot size at run time for photographing.

Test Driver SINES2

The second test program further exemplifies the use of GRAPH1, GRAPH2, and GRAPH5. In this program, 201 data points of four sine functions are generated for plotting as follows:

1. SIN1(T) = 1.0 * SIN (2 * PI * 1 * T + 0.0)
2. SIN2(T) = 2.0 * SIN (2 * PI * 2 * T + 0.0)
3. SIN3(T) = 2.0 * SIN (2 * PI * 2 * T + PI/2)
4. SIN4(T) = 3.0 * SIN (2 * PI * 3 * T + PI).

If all four functions are plotted simultaneously using GRAPH1, the result in Figure 8 is obtained.

The same data are also plotted using only 3 grid lines rather than 5 and reduced in size. The resulting plot is shown in Figure 9.

To obtain individual plots on the same screen, GRAPH5 is used. Figure 10 shows the result, and Figure 11 represents a run-time size reduction.

Test Driver LISS

The final test program was written to demonstrate GRAPH3 and GRAPH4. Again, 201 data points of various sine functions are generated, but they are not plotted against time, but against one of the functions.

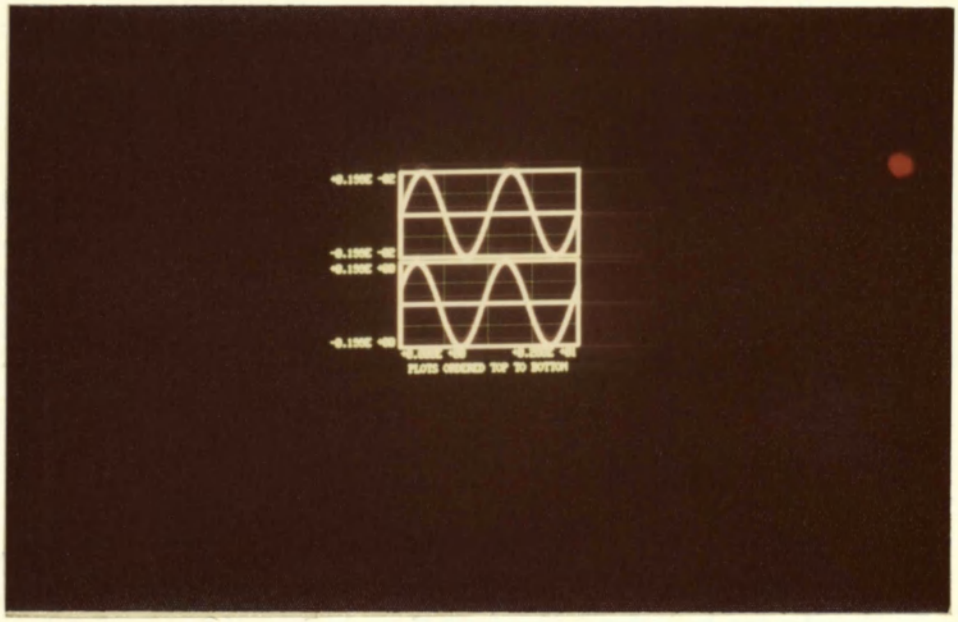

Figure 7. Plot size reduction at run time.

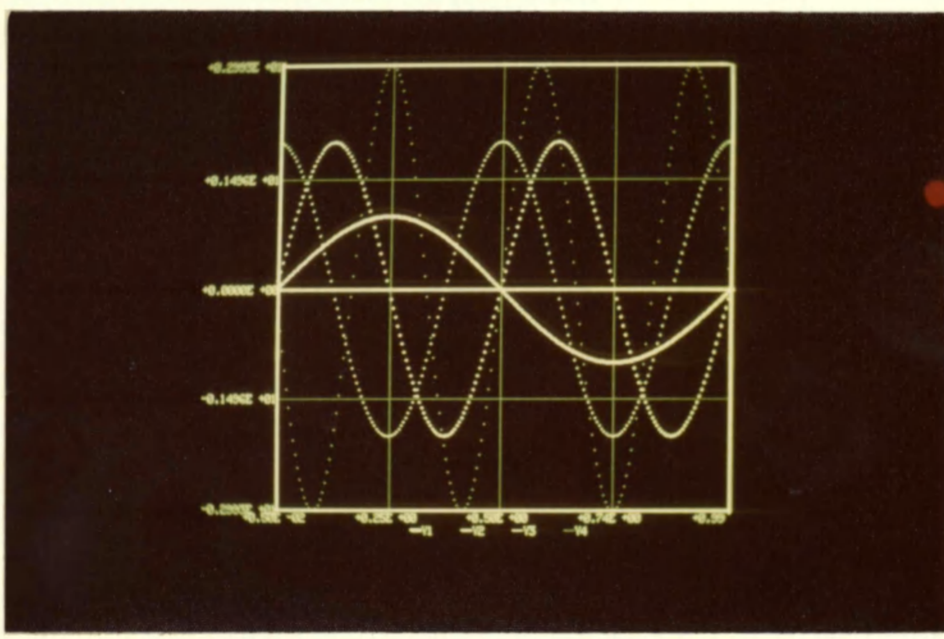

Figure 8. Simultaneous plotting of four functions by GRAPH1.

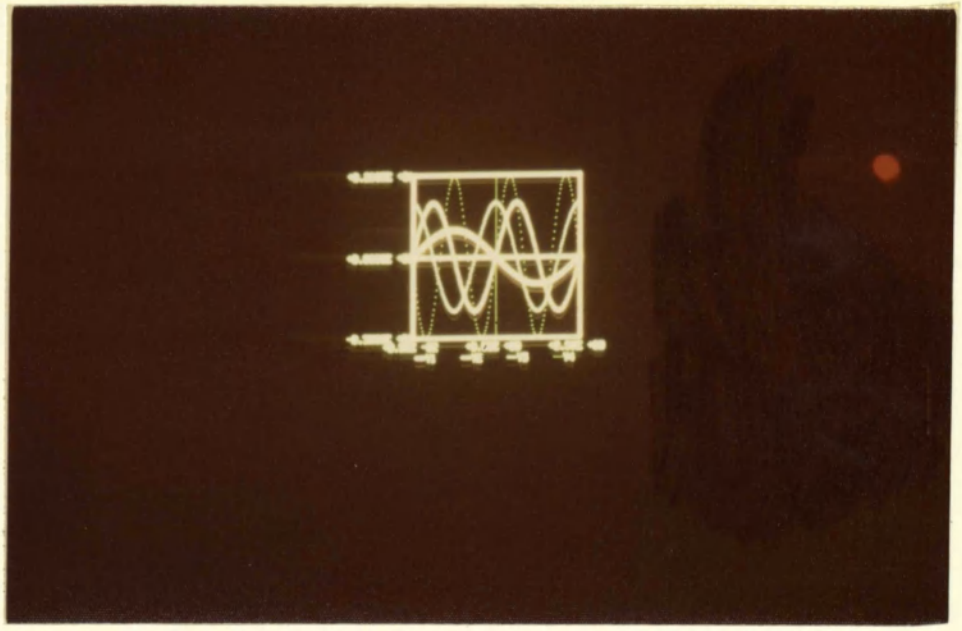

Figure 9. A re-plot of Figure 8 with differing format.

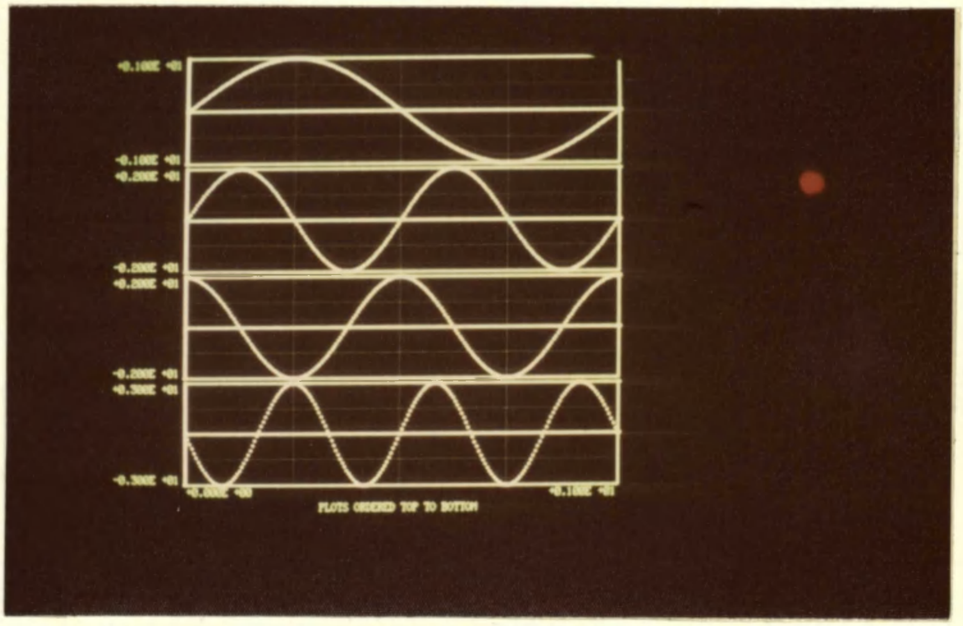

Figure 10. Individual plots using GRAPH5.

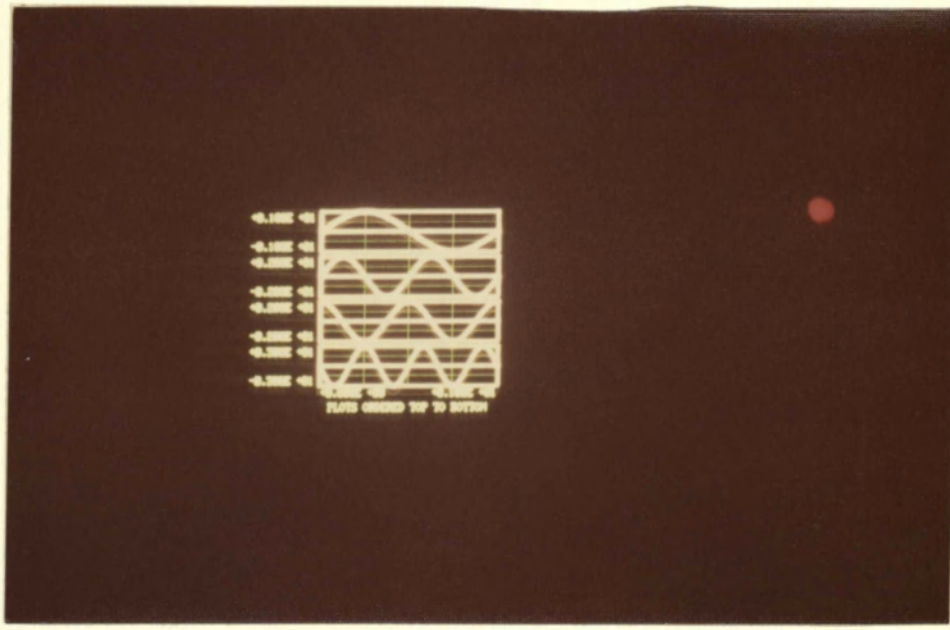

Figure 11. A run-time size reduction of Figure 10.

The first display plots one cycle of a sine function of peak magnitude 3 against itself, resulting in data points along the line, X = Y, as shown in Figure 12. Note the crowding of points toward the ends of the line as the sine functions go through their minimum and maximum points. That is, as each sine function goes through one complete cycle, actual plot generation begins at (0, 0) and proceeds up at 45 degrees to (3, 3), retraces and continues to (-3, -3), and finally retraces to (0, 0).

Figure 13 shows the result of shifting the phase of one function by 90 degrees and plotting against the original. Now plot generation begins at (0, 3) and proceeds clockwise for a full circle.

Lissajous patterns indicating relative frequencies may also be generated using GRAPH3. Figure 14 shows the result of plotting the function, Y = 3.0 SIN (2.0 * PI * 3.0 * T + PI/2) vertically versus the function, Y = 3.0 * SIN (2.0 * PI * 1.0 * T + 0.0).

In Figure 15, a third function has been introduced, this time with 5 time frequency and 90 degree phase shift, but only half amplitude.

Finally, GRAPH4 is called with the same data; the display of Figure 16 results. Note this is similar to Figure 15, but both functions are plotted full-size, having actually been plotted by a nested call to GRAPH2. As stated

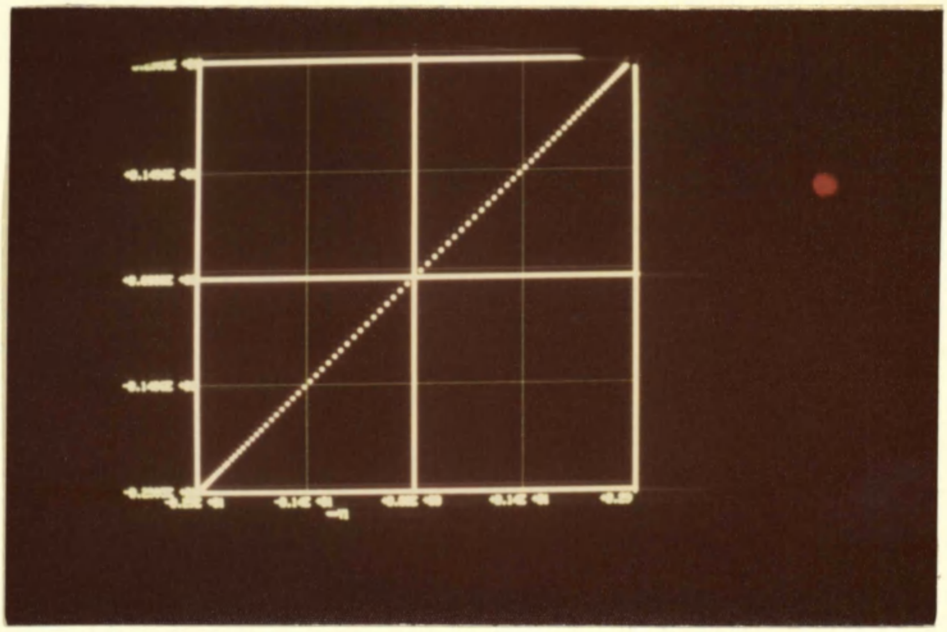

Figure 12. Sine function plotted vs. itself using GRAPH3.

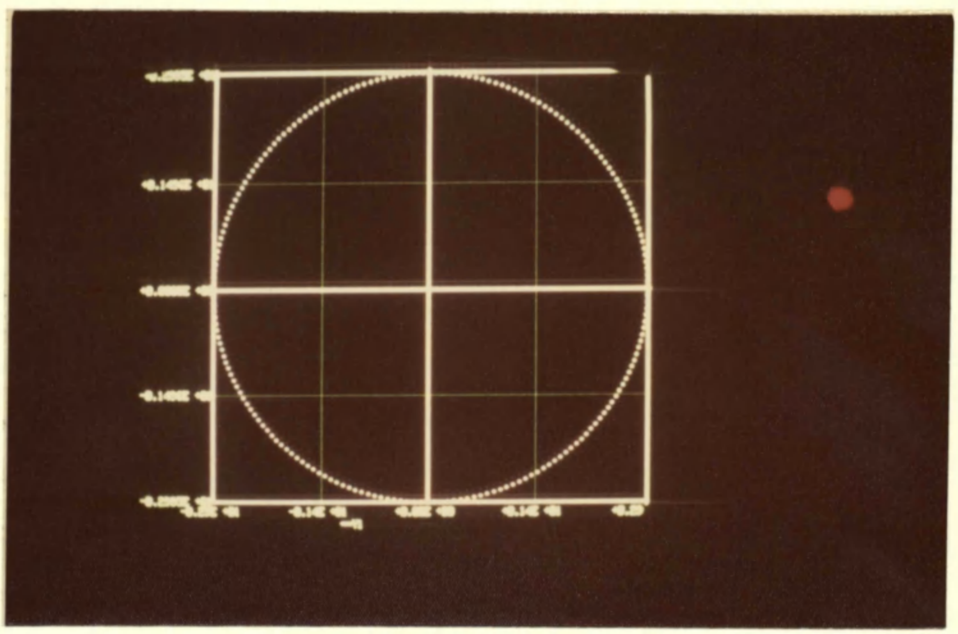

Figure 13. Result of 90 degree phase shift in one function.

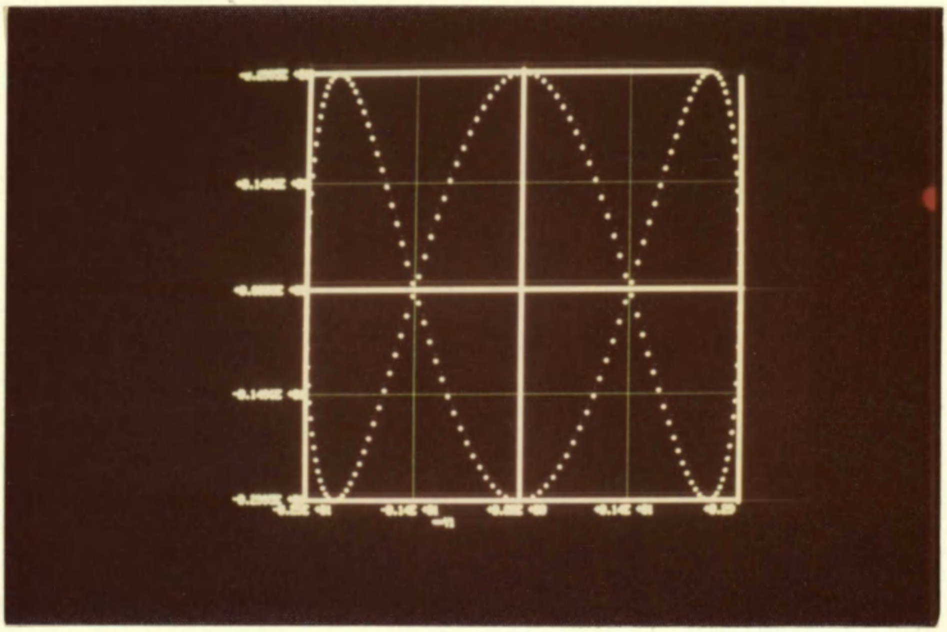

Figure 14. Lissajous pattern displaying frequency shift.

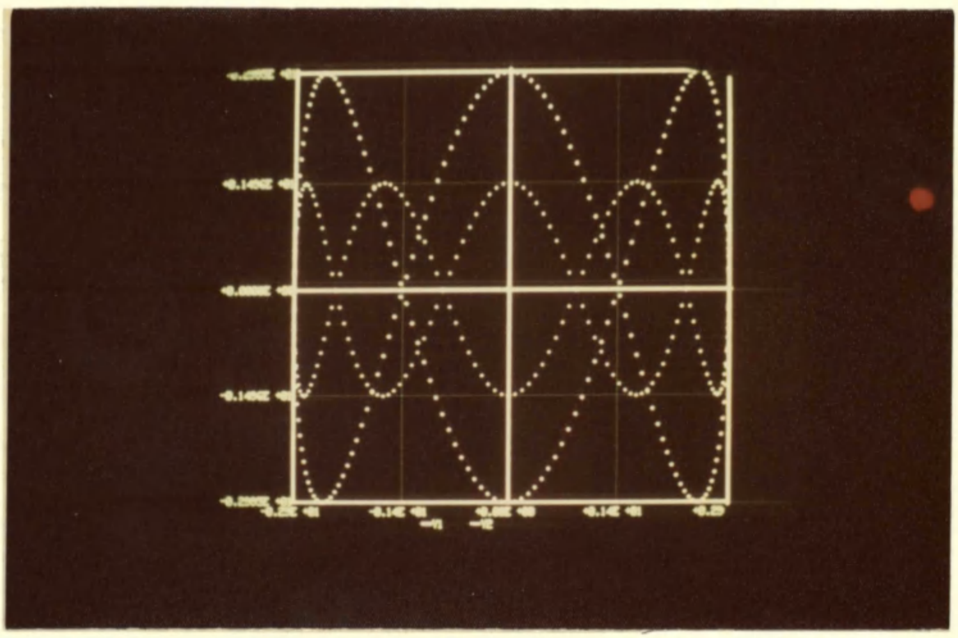

Figure 15. A third function with differing frequency and amplitude.

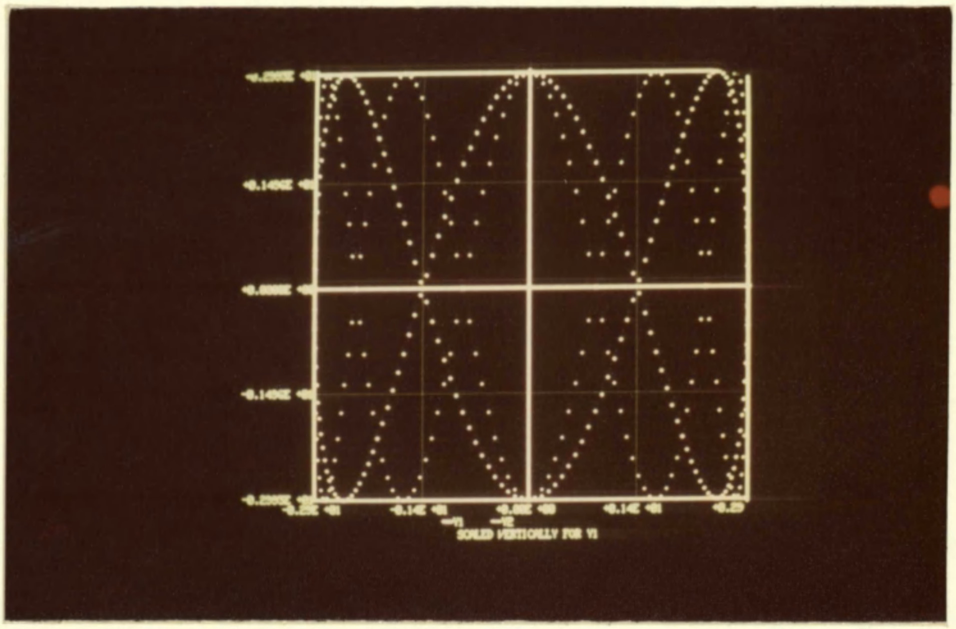

Figure 16. The data of Figure 15 plotted using GRAPH4.

previously, the utility of GRAPH4 is somewhat limited, due to the difficulty in distinguishing between plots.

CHAPTER 5

NOTES ON USING THE ROUTINES

Several comments regarding file creation and general implementation are in order at this point to enable the user to make effecgive use of the GRAPH subroutine package.

File Creation

The object code for all GRAPH subroutines is contained in file GRAPHS.OBJ. Should it become necessary to re-compile the source files, the user subroutine no-swap switch (/U) must be specified. The FORTRAN compiler command string thus appears as follows:

```
*GRAPHS[,TT: or LP:]=GRAPHS/U
```

where [,TT: or LP:] is an optional specifier to produce a compiler listing on the operator console or line printer.

In preparation for running the user program, GRAPHS.OBJ, GTFOR6.OBJ, the FORTRAN library, and the user's compiled program must be linked in generating the executable code. For example, to create the file, SINES1.SAV, the following command string was entered after initiating LINK:

```
*SINES1=SINES1,GRAPHS,GTFOR6/F.
```

General Usage

Several suggestions may be helpful in avoiding possible difficulties in using the routines.

When it is anticipated that a reduced size plot will be needed, specifying more than 3 grid lines will result in overlapping of the horizontal scale markings, as illustrated in Figure 17.

Another problem may be encountered when an even number of grid lines is specified and the plot crosses one or both zero axes. This is shown in Figure 18. Four lines were specified and were generated properly for the vertical lines; however, this cannot be accomplished for the horizontal lines, since the X-axis is placed by the data itself. Thus, an odd number of lines (ideally 5) is recommended when it is anticipated that the zero axes may be included in the display.

An upper limit of 875 points exists for each array plotted. This is due to the fact that the size of the integer arrays filled from scaling the input data arrays must be specified in their associated dimensioning statement. The figure of 875 was chosen because it represents the maximum number of addressable screen points in each direction available for plotting a true function. Thus, creating larger integer arrays for plotting would offer no advantage in improved resolution; the result would merely be multiple "overstrikes" of the same points.

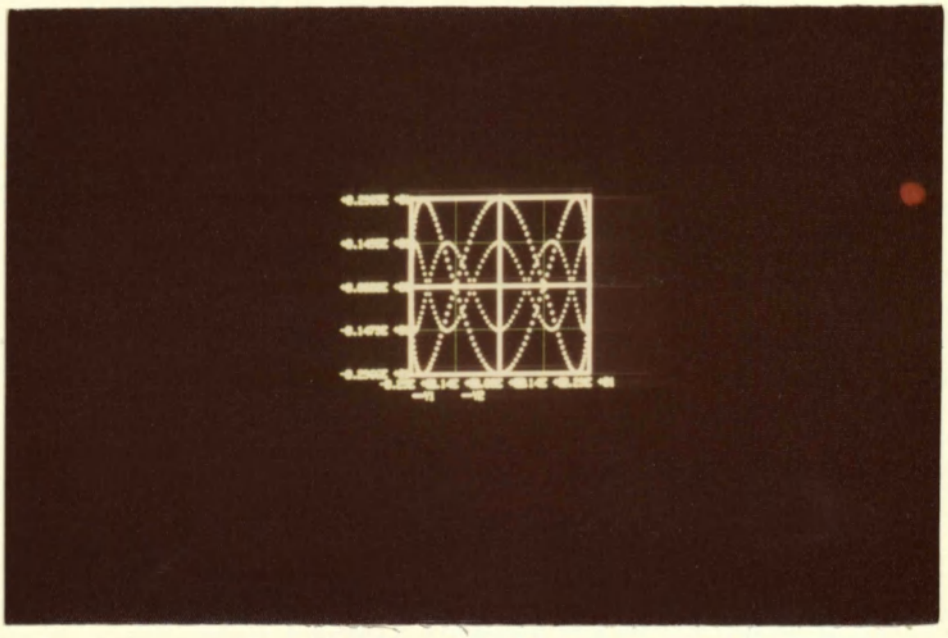

Figure 17. Overlapping text in a reduced size plot.

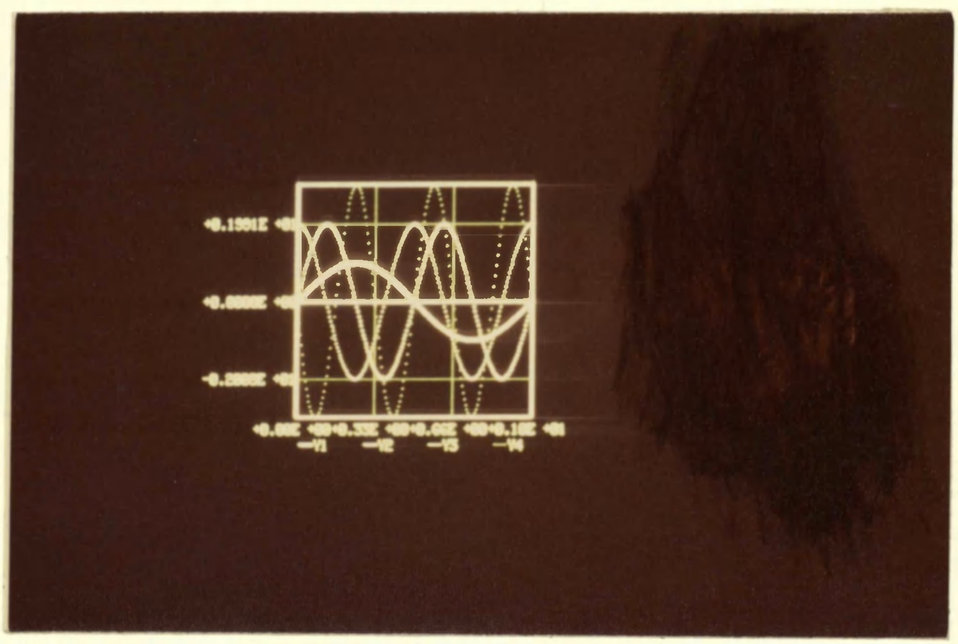

Figure 18. An even number of grid lines with zero axes included.

Because multiple curves on a single plot are distinguished by different intensities, it is vital to adjust the manual CRT intensity control carefully. The control should not be set at maximum intensity, as all curves plotted will saturate the CRT phosphor and become indistinguishable.

Space has been left at the bottom of the display screen for user titles, etc. These may consist of up to three lines of text over the entire addressable screen width. Note that these must be generated before calling the GRAPH subroutines, as the wait loop to hold the display for viewing occurs within the subroutine. In creating such title lines with the GTFOR6 package, it is recommended that a separate display file be specified which must be opened before and closed after the call to the GRAPH subroutine.

Modifications and Extensions

All subroutines were written with ease of modification built in. For example, the coordinate grid size is specified by variables MINCOR and MAXCOR. Changing their specified value at each definition will accomplish everything necessary to change plot size, as all scaling, labeling, and plotting is done with reference to these variables.

Several extensions to the package came to mind, but most were considered beyond the immediate scope of this problem. They included implementation of polar graphs, semi-log and log-log graphs, Smith charts, and representations of 3-dimensional graphs. Techniques similar to those employed herein could be used in developing such additions, with machine memory size being the only anticipated obstacle.

A possible improvement in plot distinction could be made by using the relative vector feature of the GTFOR6 package. In this way, solid, dashed, or dotted lines could be used to display individual curves.

For specific application to DARE-11, selection of convenient scale factors may enhance interpretation of results. For example, one might restrict scale factors to 1, 2, or 5 x $10^{\pm n}$. This would require only a new routine to replace SCALE.

APPENDIX A

SUBROUTINE LISTINGS

Source listings of the GRAPH subroutines are provided on the following pages.

```
      SUBROUTINE GRAPH1 (NARRYS, NPNTS, NLINES, X, Y1, Y2, Y3, Y4)
C
C       DESCRIPTION:
C            THIS ROUTINE PLOTS 1, 2, 3, OR 4 OF THE ARRAYS Y1 --- Y4
C       AS A FUNCTION OF X, ON ONE VERTICAL SCALE DETERMINED BY THE
C       MINIMUM AND MAXIMUM VALUES IN THE ARRAYS TO BE PLOTTED.
C
C       CALLED BY:
C            USER
C            'GRAPH3'
C
C       ARGUMENTS:
C            'NARRYS' --- NUMBER OF ARRAYS (1 --- 5) SUPPLIED; USED
C                 TO DETERMINE NUMBER OF PLOTS TO BE GENERATED.
C            'NPNTS' --- NUMBER OF POINTS PER ARRAY; ASSUMED TO BE
C                 THE SAME FOR ALL.
C            'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C                 AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C                 LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C                 FOR PHOTOGRAPHING.
C            'X' --- HORIZONTAL INPUT ARRAY.
C            'Y1' --- VERTICAL INPUT ARRAY 1.
C            'Y2' --- VERTICAL INPUT ARRAY 2.
C            'Y3' --- VERTICAL INPUT ARRAY 3.
C            'Y4' --- VERTICAL INPUT ARRAY 4.
C
C       LOCAL VARIABLES:
C            'YMIN' --- REAL ARRAY(5) --- MINIMUM VALUES IN VERTICAL
C                 INPUT ARRAYS AND OVERALL MINIMUM VALUE.
C            'YMAX' --- REAL ARRAY(5) --- MAXIMUM VALUES IN VERTICAL
C                 INPUT ARRAYS AND OVERALL MAXIMUM VALUE.
C            'NPLOTS' --- INTEGER; NUMBER OF PLOTS BEING GENERATED.
C            'XMIN' --- REAL; MINIMUM VALUE IN INPUT ARRAY 'X'.
C            'XMAX' --- REAL; MAXIMUM VALUE IN INPUT ARRAY 'X'.
C            'I' --- INTEGER; INDEXING VARIABLE.
```

```
C          'MINCOR' --- INTEGER; SCREEN COORDINATE FOR BOTTOM AND
C               LEFT BOUNDARIES.
C          'MAXCOR' --- INTEGER; SCREEN COORDINATE FOR TOP AND
C               RIGHT BOUNDARIES.
C          'INPCHR' --- INTEGER; USER-SELECTED INPUT CHARACTER TO
C               SPECIFY ROUTINE TERMINATION OR REPLOT.
C
C     COMMON BLOCKS:
C          'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C          'IARRYS':  'IX' --- HORIZONTAL INTEGER ARRAY.
C                     'IY' --- VERTICAL INTEGER ARRAY.
C
C     SUBROUTINES CALLED:
C          'MINMAX'
C          'SCALE'
C          'PLOT'
C          'GRID1'
C          'LEGEND'
C          'DINIT'
C          'DCLOSE'
C
      COMMON /DISFIL/ IDISPL(2000)
      COMMON /IARRYS/ IX(875), IY(875)
      DIMENSION X(1), Y1(1), Y2(1), Y3(1), Y4(1)
      DIMENSION YMIN(5), YMAX(5)
      IDISPL(1) = 0
      NPLOTS = NARRYS - 1
C
C          CHECK THAT 'NPLOTS' IS IN RANGE 1 --- 4.
C
      IF (NPLOTS .LT. 1 .OR. NPLOTS .GT. 4) GO TO 140
C
C          FIND MINIMUM AND MAXIMUM VALUES IN EACH ARRAY TO BE
C     PLOTTED.
C
```

```
      CALL MINMAX (X, NPNTS, XMIN, XMAX)
      GO TO (40, 30, 20, 10) NPLOTS
10    CALL MINMAX (Y4, NPNTS, YMIN(4), YMAX(4))
20    CALL MINMAX (Y3, NPNTS, YMIN(3), YMAX(3))
30    CALL MINMAX (Y2, NPNTS, YMIN(2), YMAX(2))
40    CALL MINMAX (Y1, NPNTS, YMIN(1), YMAX(1))
C
C          FIND MINIMUM AND MAXIMUM VALUES OVER ALL ARRAYS TO BE
C     PLOTTED.  IF JUST ONE PLOT, = YMIN(1), YMAX(1).
C
      IF (NPLOTS .GT. 1) GO TO 45
      YMIN(5) = YMIN(1)
      YMAX(5) = YMAX(1)
      GO TO 50
45    DO 50 I = 1, NPLOTS - 1
      YMIN(5) = AMIN1 (YMIN(I), YMIN(I + 1))
      YMAX(5) = AMAX1 (YMAX(I), YMAX(I + 1))
50    CONTINUE
C
C          SCALE AND PLOT ARRAYS, DISPLAY LEGEND.
C
      MINCOR = 149
      MAXCOR = 1023
      IF (NLINES .GE. 0) GO TO 55
C
C          RESET BOUNDARIES FOR REDUCED PLOT.
C
      MINCOR = 404
      MAXCOR = 767
55    CALL SCALE (MINCOR, MAXCOR, X, NPNTS, XMIN, XMAX, IX)
      GO TO (90, 80, 70, 60) NPLOTS
60    CALL SCALE (MINCOR, MAXCOR, Y4, NPNTS, YMIN(5), YMAX(5), IY)
      CALL PLOT (IX, IY, NPNTS, 1)
70    CALL SCALE (MINCOR, MAXCOR, Y3, NPNTS, YMIN(5), YMAX(5), IY)
      CALL PLOT (IX, IY, NPNTS, 3)
```

```
80      CALL SCALE (MINCOR, MAXCOR, Y2, NPNTS, YMIN(5), YMAX(5), IY)
        CALL PLOT (IX, IY, NPNTS, 5)
90      CALL SCALE (MINCOR, MAXCOR, Y1, NPNTS, YMIN(5), YMAX(5), IY)
        CALL PLOT (IX, IY, NPNTS, 7)
C
C            GENERATE COORDINATE GRID.
C
        CALL GRID1 (XMIN, XMAX, YMIN(5), YMAX(5), NLINES)
        CALL LEGEND (NPLOTS, MINCOR)
C
C            INITIATE DISPLAY.
C
        CALL DINIT (IDISPL)
C
C            WAIT FOLLOWS.
C
        WRITE (5, 1000)
1000    FORMAT (/, 5X, 16HCONTINUATION ---)
100     IF (NLINES .GE. 0) GO TO 110
        WRITE (5, 2000)
2000    FORMAT (/, 5X, 27HENTER '6' FOR EXPANDED PLOT, /, 5X,
     1  27HENTER '0' TO TERMINATE PLOT)
        READ (7, 3000) INPCHR
3000    FORMAT (I1)
        IF (INPCHR .NE. 6) GO TO 130
        GO TO 120
110     WRITE (5, 4000)
4000    FORMAT (/, 5X, 26HENTER '5' FOR REDUCED PLOT, /, 5X,
     1  27HENTER '0' TO TERMINATE PLOT)
        READ (7, 3000) INPCHR
        IF (INPCHR .NE. 5) GO TO 130
120     NLINES = - NLINES
        CALL DCLOSE (IDISPL)
        IDISPL(1) = 0
        GO TO 50
```

```
130       IF (INPCHR .EQ. 0) GO TO 150
          WRITE (5, 5000)
5000      FORMAT (/, 5X, 18HUNRECOGNIZED INPUT)
          GO TO 100
140       WRITE (5, 6000)
6000      FORMAT (/, 5X, 43HINVALID INPUT DATA TO 'GRAPH' SUBROUTINE.  ,
     1    21HEXECUTION TERMINATED.)
150       CALL DCLOSE (IDISPL)
          RETURN
          END



          SUBROUTINE GRAPH2 (NARRYS, NPNTS, NLINES, X, Y1, Y2, Y3, Y4)
C
C         DESCRIPTION:
C              THIS ROUTINE PLOTS 1, 2, 3, OR 4 OF THE ARRAYS
C         Y1 --- Y4 AS A FUNCTION OF X, WITH EACH ARRAY OPTIMALLY SCALED
C         TO FILL THE SCREEN.  THE VERTICAL SCALE MARKINGS USED INITIAL-
C         LY CORRESPOND TO Y1, BUT MAY BE CHANGED BY ENTERING '1', '2',
C         '3', OR '4' ON THE KEYBOARD TO SELECT A VERTICAL SCALE COR-
C         RESPONDING TO ANY OF THE ARRAYS PLOTTED.  THE SCALE MAY BE
C         CHANGED AS MANY TIMES AS DESIRED.
C
C         CALLED BY:
C              USER
C              'GRAPH4'
C
C         ARGUMENTS:
C              'NARRYS' --- NUMBER OF PLOTS (1 --- 5) SUPPLIED; USED
C                   TO DETERMINE NUMBER OF PLOTS TO BE GENERATED.
C              'NPNTS' --- NUMBER OF POINTS PER ARRAY; ASSUMED TO BE
C                   THE SAME FOR ALL.
```

```
C          'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C               AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C               LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C               FOR PHOTOGRAPHING.
C          'X' --- HORIZONTAL INPUT ARRAY.
C          'Y1' --- VERTICAL INPUT ARRAY 1.
C          'Y2' --- VERTICAL INPUT ARRAY 2.
C          'Y3' --- VERTICAL INPUT ARRAY 3.
C          'Y4' --- VERTICAL INPUT ARRAY 4.
C
C     LOCAL VARIABLES:
C          'YMIN' --- REAL ARRAY(4); MINIMUM VALUES IN Y-ARRAYS.
C          'YMAX' --- REAL ARRAY(4); MAXIMUM VALUES IN Y-ARRAYS.
C          'YARRAY' --- DATA ARRAY(4); TEXT FOR Y-ARRAY INDICATION.
C          'NPLOTS' --- INTEGER; NUMBER OF PLOTS BEING GENERATED.
C          'MINCOR' --- INTEGER; SCREEN COORDINATE FOR BOTTOM AND
C               LEFT BOUNDARIES.
C          'MAXCOR' --- INTEGER; SCREEN COORDINATE FOR TOP AND
C               RIGHT BOUNDARIES.
C          'XMIN' --- REAL; MINIMUM VALUE IN INPUT ARRAY 'X'.
C          'XMAX' --- REAL; MAXIMUM VALUE IN INPUT ARRAY 'X'.
C          'NSCALE' --- INTEGER; Y-ARRAY NUMBER FOR CORRESPONDING
C               VERTICAL SCALE DESIRED.
C          'INPCHR' --- USER-SELECTED INPUT CHARACTER TO
C               SPECIFY ROUTINE TERMINATION OR REPLOT.
C
C     COMMON BLOCKS:
C          'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C          'IARRYS':  'IX' --- HORIZONTAL INTEGER ARRAY.
C                     'IY' --- VERTICAL INTEGER ARRAY.
C
C     SUBROUTINES CALLED:
C          'MINMAX'
C          'SCALE'
C          'PLOT'
```

```
C             'GRID1'
C             'LEGEND'
C             'PRAMTR'
C             'POINTA'
C             'TEXT'
C             'DINIT'
C             'DCLOSE'
C
      COMMON /DISFIL/ IDISPL(2000)
      COMMON /IARRYS/ IX(875), IY(875)
      DIMENSION X(1), Y1(1), Y2(1), Y3(1), Y4(1)
      DIMENSION YMIN(4), YMAX(4)
      DIMENSION YARRAY(4)
      DATA YARRAY(1), YARRAY(2), YARRAY(3), YARRAY(4)
     1 /'Y1', 'Y2', 'Y3', 'Y4'/
      IDISPL(1) = 0
      NPLOTS = NARRYS - 1
C
C             CHECK THAT 'NPLOTS' IS IN RANGE 1 --- 4.
C
      IF (NPLOTS .LT. 1 .OR. NPLOTS .GT. 4) GO TO 180
C
C             FIRST PLOT WILL SHOW VERTICAL SCALE FOR Y1.
C
      NSCALE = 1
5     MINCOR = 149
      MAXCOR = 1023
      IF (NLINES .GE. 0) GO TO 10
C
C             RESET BOUNDARIES FOR REDUCED PLOT.
C
      MINCOR = 404
      MAXCOR = 767
10    CONTINUE
C
```

```
C          FIND MINIMUM AND MAXIMUM VALUES IN EACH ARRAY TO BE
C       PLOTTED, SCALE AND PLOT ARRAYS.
C
        CALL MINMAX (X, NPNTS, XMIN, XMAX)
        CALL SCALE (MINCOR, MAXCOR, X, NPNTS, XMIN, XMAX, IX)
        GO TO (50, 40, 30, 20) NPLOTS
20      CALL MINMAX (Y4, NPNTS, YMIN(4), YMAX(4))
        CALL SCALE (MINCOR, MAXCOR, Y4, NPNTS, YMIN(4), YMAX(4), IY)
        CALL PLOT (IX, IY, NPNTS, 1)
30      CALL MINMAX (Y3, NPNTS, YMIN(3), YMAX(3))
        CALL SCALE (MINCOR, MAXCOR, Y3, NPNTS, YMIN(3), YMAX(3), IY)
        CALL PLOT (IX, IY, NPNTS, 3)
40      CALL MINMAX (Y2, NPNTS, YMIN(2), YMAX(2))
        CALL SCALE (MINCOR, MAXCOR, Y2, NPNTS, YMIN(2), YMAX(2), IY)
        CALL PLOT (IX, IY, NPNTS, 5)
50      CALL MINMAX (Y1, NPNTS, YMIN(1), YMAX(1))
        CALL SCALE (MINCOR, MAXCOR, Y1, NPNTS, YMIN(1), YMAX(1), IY)
        CALL PLOT (IX, IY, NPNTS, 7)
C
C          DISPLAY GRID, LEGEND.
C
        CALL GRID1 (XMIN, XMAX, YMIN(NSCALE), YMAX(NSCALE), NLINES)
        CALL LEGEND (NPLOTS, MINCOR)
C
C          DISPLAY WHICH VERTICAL SCALE IS SHOWN.
C
        CALL PRAMTR (2, 6, IDISPL)
        CALL POINTA (440, (MINCOR - 80), 0, IDISPL)
        CALL TEXT ('SCALED VERTICALLY FOR ', 22, IDISPL)
        CALL TEXT (YARRAY(NSCALE), 2, IDISPL)
C
C          INITIATE DISPLAY.
C
        CALL DINIT (IDISPL)

C
```

```
C              WAIT FOLLOWS.
C
          WRITE (5, 1000)
1000      FORMAT (/, 5X, 16HCONTINUATION ---)
105       IF (NLINES .GE. 0) GO TO 110
          WRITE (5, 2000)
2000      FORMAT (/, 5X, 44HENTER '6' FOR EXPANDED PLOT WITH SELECTABLE ,
     1    14HVERTICAL SCALE, /, 5X, 30HENTER '7' TO CHANGE SCALE ONLY, /,
     1    5X, 27HENTER '0' TO TERMINATE PLOT)
          READ (7, 3000) INPCHR
3000      FORMAT (I1)
          IF (INPCHR .NE. 6) GO TO 130
          GO TO 120
110       WRITE (5, 4000)
4000      FORMAT (/, 5X, 43HENTER '5' FOR REDUCED PLOT WITH SELECTABLE ,
     1    14HVERTICAL SCALE, /, 5X, 30HENTER '7' TO CHANGE SCALE ONLY, /,
     1    5X, 27HENTER '0' TO TERMINATE PLOT)
          READ (7, 3000) INPCHR
          IF (INPCHR .NE. 5) GO TO 130
120       NLINES = - NLINES
          GO TO 140
130       IF (INPCHR .EQ. 7) GO TO 140
          IF (INPCHR .EQ. 0) GO TO 180
          WRITE (5, 5000)
5000      FORMAT (/, 5X, 18HUNRECOGNIZED INPUT)
          GO TO 105
140       WRITE (5, 6000)
6000      FORMAT (/, 5X, 22HSELECT VERTICAL SCALE:, /, 5X,
     1    57HENTER '1', '2', '3', OR '4' TO DISPLAY VERTICAL SCALE FOR,
     1    /, 5X, 21HCORRESPONDING Y-ARRAY)
150       READ (7, 3000) NSCALE
          IF (NSCALE .GE. 1 .AND. NSCALE .LE. NPLOTS) GO TO 160
          WRITE (5, 7000)
7000      FORMAT (/, 5X, 43HINVALID INPUT --- NO CORRESPONDING Y-ARRAY.,
     1    18H  PLEASE RE-ENTER.)
```

```
      GO TO 150
160   CALL DCLOSE (IDISPL)
      IDISPL(1) = 0
      GO TO 5
170   WRITE (5, 8000)
8000  FORMAT (/, 5X, 43HINVALID INPUT DATA TO 'GRAPH' SUBROUTINE.  ,
     1 21HEXECUTION TERMINATED.)
180   CALL DCLOSE (IDISPL)
      RETURN
      END



      SUBROUTINE GRAPH3 (NARRYS, NPNTS, NLINES, Y1, Y2, Y3, Y4)
C
C     DESCRIPTION:
C          THIS ROUTINE PLOTS 1, 2, OR 3 OF THE ARRAYS Y1 --- Y3
C     AS A FUNCTION OF Y4, ON ONE VERTICAL SCALE DETERMINED BY THE
C     MINIMUM AND MAXIMUM VALUES IN THE ARRAYS TO BE PLOTTED.
C
C     CALLED BY:
C          USER
C
C     ARGUMENTS:
C          'NARRYS' --- NUMBER OF ARRAYS (1 --- 4) SUPPLIED; USED
C               TO DETERMINE NUMBER OF PLOTS TO BE GENERATED.
C          'NPNTS' --- NUMBER OF POINTS PER ARRAY; ASSUMED TO BE
C               THE SAME FOR ALL.
C          'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C               AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C               LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C               FOR PHOTOGRAPHING.
C          'Y1' --- VERTICAL INPUT ARRAY 1.
C          'Y2' --- VERTICAL INPUT ARRAY 2.
```

```
C             'Y3' --- VERTICAL INPUT ARRAY 3.
C             'Y4' --- HORIZONTAL INPUT ARRAY.
C
C        LOCAL VARIABLES:
C             NONE
C
C        SUBROUTINES CALLED:
C             'GRAPH1'
C
C             CHECK THAT 'NARRYS' IS IN RANGE 1 --- 4.
C
         IF (NARRYS .LT. 1 .OR. NARRYS .GT. 4) GO TO 10
         CALL GRAPH1 (NARRYS, NPNTS, NLINES, Y4, Y1, Y2, Y3)
         GO TO 20
10       WRITE (5, 1000)
1000     FORMAT (///, 5X, 43HINVALID INPUT DATA TO 'GRAPH' SUBROUTINE.   ,
     1   21HEXECUTION TERMINATED.)
20       RETURN
         END


         SUBROUTINE GRAPH4 (NARRYS, NPNTS, NLINES, Y1, Y2, Y3, Y4)
C
C        DESCRIPTION:
C             THIS ROUTINE PLOTS 1, 2, OR 3 OF THE ARRAYS Y1 --- Y3
C        AS A FUNCTION OF Y4, WITH EACH ARRAY OPTIMALLY SCALED TO FILL
C        THE SCREEN.  THE VERTICAL SCALE MARKINGS USED INITIALLY COR-
C        RESPOND TO Y1, BUT MAY BE CHANGED BY ENTERING '1', '2', OR '3'
C        ON THE KEYBOARD TO SELECT A VERTICAL SCALE CORRESPONDING TO
C        ANY OF THE ARRAYS PLOTTED.  THE SCALE MAY BE CHANGED AS MANY
C        TIMES AS DESIRED.
C
```

```
C         CALLED BY:
C              USER
C
C         ARGUMENTS:
C              'NARRYS' --- NUMBER OF ARRAYS (1 --- 4) SUPPLIED; USED
C                   TO DETERMINE NUMBER OF PLOTS TO BE GENERATED.
C              'NPNTS' --- NUMBER OF POINTS PER ARRAY; ASSUMED TO BE
C                   THE SAME FOR ALL.
C              'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C                   AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C                   LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C                   FOR PHOTOGRAPHING.
C              'Y1' --- VERTICAL INPUT ARRAY 1.
C              'Y2' --- VERTICAL INPUT ARRAY 2.
C              'Y3' --- VERTICAL INPUT ARRAY 3.
C              'Y4' --- HORIZONTAL INPUT ARRAY.
C
C         LOCAL VARIABLES:
C              NONE
C
C         SUBROUTINES CALLED:
C              'GRAPH2'
C
C              CHECK THAT 'NARRYS' IS IN RANGE 1 --- 4.
C
          IF (NARRYS .LT. 1 .OR. NARRYS .GT. 4) GO TO 10
          CALL GRAPH2 (NARRYS, NPNTS, NLINES, Y4, Y1, Y2, Y3)
          GO TO 20
10        WRITE (5, 1000)
1000      FORMAT (///, 5X, 43HINVALID INPUT DATA TO 'GRAPH' SUBROUTINE.   ,
     1    21HEXECUTION TERMINATED.)
20        RETURN
          END
```

```
      SUBROUTINE GRAPH5 (NARRYS, NPNTS, NLINES, X, Y1, Y2, Y3, Y4)
C
C     DESCRIPTION:
C         THIS ROUTINE PLOTS 1, 2, 3, OR 4 OF THE ARRAYS
C     Y1 --- Y4 ON SEPARATE PLOTS WITHIN ONE SCREEN AS A FUNCTION
C     OF X, WITH EACH ARRAY OPTIMALLY SCALED TO FILL EACH COORDINATE
C     GRID.  DUE TO THE RESTRICTED SPACE FOR TEXT ON THE SCREEN, THE
C     GRIDS ARE LABELED ONLY WITH THE MINIMUM AND MAXIMUM VALUES OF
C     EACH ARRAY.
C
C     CALLED BY:
C         USER
C
C     ARGUMENTS:
C         'NARRYS' --- NUMBER OF ARRAYS (1 --- 5) SUPPLIED; USED
C              TO DETERMINE NUMBER OF PLOTS TO BE GENERATED.
C         'NPNTS' --- NUMBER OF POINTS PER ARRAY; ASSUMED TO BE
C              THE SAME FOR ALL.
C         'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C              AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C              LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C              FOR PHOTOGRAPHING.
C         'X' --- HORIZONTAL INPUT ARRAY.
C         'Y1' --- VERTICAL INPUT ARRAY 1.
C         'Y2' --- VERTICAL INPUT ARRAY 2.
C         'Y3' --- VERTICAL INPUT ARRAY 3.
C         'Y4' --- VERTICAL INPUT ARRAY 4.
C
C     LOCAL VARIABLES:
C         'HEIGHT' --- INTEGER; VERTICAL DIMENSION OF EACH PLOT,
C              MEASURED IN SCREEN POINTS.
C         'NPLOTS' --- INTEGER; NUMBER OF PLOTS BEING GENERATED.
C         'MINCOR' --- INTEGER; SCREEN COORDINATE FOR BOTTOM AND
C              LEFT BOUNDARIES.
C         'MAXCOR' --- INTEGER; SCREEN COORDINATE FOR TOP AND
```

```
C                RIGHT BOUNDARIES.
C           'IYO' --- SCREEN COORDINATE OF BOTTOM EDGE OF PLOT BEING
C                GENERATED.
C           'XMIN' --- REAL; MINIMUM VALUE IN INPUT ARRAY 'X'.
C           'XMAX' --- REAL; MAXIMUM VALUE IN INPUT ARRAY 'X'.
C           'YMIN' --- REAL; MINIMUM VALUE IN VERTICAL ARRAY BEING
C                PLOTTED.
C           'YMAX' --- REAL; MAXIMUM VALUE IN VERTICAL ARRAY BEING
C                PLOTTED.
C           'INPCHR' --- USER-SELECTED INPUT CHARACTER TO
C                SPECIFY ROUTINE TERMINATION OR REPLOT.
C
C     COMMON BLOCKS:
C           'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C           'IARRYS':  'IX' --- HORIZONTAL INTEGER ARRAY.
C                      'IY' --- VERTICAL INTEGER ARRAY.
C
C     SUBROUTINES CALLED:
C           'PRAMTR'
C           'MINMAX'
C           'SCALE'
C           'PLOT'
C           'GRID2'
C           'POINTA'
C           'TEXT'
C           'DINIT'
C           'DCLOSE'
C
      COMMON /DISFIL/ IDISPL(2000)
      COMMON /IARRYS/ IX(875), IY(875)
      DIMENSION X(1), Y1(1), Y2(1), Y3(1), Y4(1)
      INTEGER HEIGHT
      IDISPL(1) = 0
      CALL PRAMTR (2, 7, IDISPL)
      NPLOTS = NARRYS - 1
```

```
C
C           CHECK THAT 'NPLOTS' IS IN RANGE 1 --- 4.
C
      IF (NPLOTS .LT. 1 .OR. NPLOTS .GT. 4) GO TO 100
5     MINCOR = 149
      MAXCOR = 1023
      IF (NLINES .GE. 0) GO TO 10
C
C           RESET BOUNDARIES FOR REDUCED PLOT.
C
      MINCOR = 404
      MAXCOR = 767
10    CONTINUE
C
C           DETERMINE HEIGHT OF EACH PLOT, INITIALIZE BOTTOM EDGE
C     Y-COORDINATE.
C
      HEIGHT = ((MAXCOR - MINCOR) - 10 * (NPLOTS - 1)) / NPLOTS
      IYO = MINCOR
C
C           FIND MINIMUM AND MAXIMUM VALUES IN EACH ARRAY TO BE
C     PLOTTED.  SCALE, PLOT, AND GENERATE COORDINATE GRIDS.
C
      CALL MINMAX (X, NPNTS, XMIN, XMAX)
      CALL SCALE (MINCOR, MAXCOR, X, NPNTS, XMIN, XMAX, IX)
      GO TO (50, 40, 30, 20) NPLOTS
20    CALL MINMAX (Y4, NPNTS, YMIN, YMAX)
      CALL SCALE (IYO, (HEIGHT + IYO), Y4, NPNTS, YMIN, YMAX, IY)
      CALL PLOT (IX, IY, NPNTS, 7)
      CALL GRID2 (XMIN, XMAX, YMIN, YMAX, NLINES, IYO, HEIGHT,
     1 MINCOR, MAXCOR)
      IYO = IYO + HEIGHT + 10
30    CALL MINMAX (Y3, NPNTS, YMIN, YMAX)
      CALL SCALE (IYO, (HEIGHT + IYO), Y3, NPNTS, YMIN, YMAX, IY)
      CALL PLOT (IX, IY, NPNTS, 7)
```

```
      CALL GRID2 (XMIN, XMAX, YMIN, YMAX, NLINES, IYO, HEIGHT,
     1 MINCOR, MAXCOR)
      IYO = IYO + HEIGHT + 10
40    CALL MINMAX (Y2, NPNTS, YMIN, YMAX)
      CALL SCALE (IYO, (HEIGHT + IYO), Y2, NPNTS, YMIN, YMAX, IY)
      CALL PLOT (IX, IY, NPNTS, 7)
      CALL GRID2 (XMIN, XMAX, YMIN, YMAX, NLINES, IYO, HEIGHT,
     1 MINCOR, MAXCOR)
      IYO = IYO + HEIGHT + 10
50    CALL MINMAX (Y1, NPNTS, YMIN, YMAX)
      CALL SCALE (IYO, (HEIGHT + IYO), Y1, NPNTS, YMIN, YMAX, IY)
      CALL PLOT (IX, IY, NPNTS, 7)
      CALL GRID2 (XMIN, XMAX, YMIN, YMAX, NLINES, IYO, HEIGHT,
     1 MINCOR, MAXCOR)
C
C           DISPLAY MESSAGE.
C
      CALL POINTA (420, (MINCOR - 55), 0, IDISPL)
      CALL PRAMTR (2, 6, IDISPL)
      CALL TEXT ('PLOTS ORDERED TOP TO BOTTOM', 27, IDISPL)
C
C           DISPLAY XMIN, XMAX.
C
      CALL POINTA (MINCOR, (MINCOR - 25), 0, IDISPL)
      CALL NUMTXT (XMIN, 3)
      CALL POINTA ((MAXCOR - 135), (MINCOR - 25), 0, IDISPL)
      CALL NUMTXT (XMAX, 3)
C
C           INITIATE DISPLAY.
C
      CALL DINIT (IDISPL)
C
C           WAIT FOLLOWS.
C
      WRITE (5, 1000)
```

```
1000    FORMAT (/, 5X, 16HCONTINUATION ---)
60      IF (NLINES .GE. 0) GO TO 70
        WRITE (5, 2000)
2000    FORMAT (/, 5X, 27HENTER '6' FOR EXPANDED PLOT, /, 5X,
     1  27HENTER '0' TO TERMINATE PLOT)
        READ (7, 3000) INPCHR
3000    FORMAT (I1)
        IF (INPCHR .NE. 6) GO TO 90
        GO TO 80
70      WRITE (5, 4000)
4000    FORMAT (/, 5X, 26HENTER '5' FOR REDUCED PLOT, /, 5X,
     1  27HENTER '0' TO TERMINATE PLOT)
        READ (7, 3000) INPCHR
        IF (INPCHR .NE. 5) GO TO 90
80      NLINES = - NLINES
        CALL DCLOSE (IDISPL)
        IDISPL(1) = 0
        GO TO 5
90      IF (INPCHR .EQ. 0) GO TO 110
        WRITE (5, 5000)
5000    FORMAT (/, 5X, 18HUNRECOGNIZED INPUT)
        GO TO 60
100     WRITE (5, 6000)
6000    FORMAT (/, 5X, 43HINVALID INPUT DATA TO 'GRAPH' SUBROUTINE.  ,
     1  21HEXECUTION TERMINATED.)
110     CALL DCLOSE (IDISPL)
        RETURN
        END


        SUBROUTINE MINMAX (XARRAY, NPNTS, XMIN, XMAX)
C
C       DESCRIPTION:
```

```
C           THIS ROUTINE RETURNS THE MINIMUM AND MAXIMUM VALUES IN THE
C       INPUT ARRAY.
C
C       CALLED BY:
C           'GRAPH1'
C           'GRAPH2'
C           'GRAPH5'
C
C       ARGUMENTS:
C           'XARRAY' --- INPUT ARRAY.
C           'NPNTS' --- NUMBER OF POINTS IN THE INPUT ARRAY.
C           'XMIN' --- RETURNS MINIMUM VALUE.
C           'XMAX' --- RETURNS MAXIMUM VALUE.
C
C       LOCAL VARIABLES:
C           'I' --- INTEGER; INDEXING VARIABLE.
C
C       SUBROUTINES CALLED:
C           NONE
C
        DIMENSION XARRAY(1)
        XMIN = XARRAY(1)
        XMAX = XMIN
        DO 10 I = 2, NPNTS
        XMIN = AMIN1 (XMIN, XARRAY(I))
        XMAX = AMAX1 (XMAX, XARRAY(I))
10      CONTINUE
        RETURN
        END



        SUBROUTINE SCALE (IMIN, IMAX, XARRAY, NPNTS, XMIN, XMAX, IARRAY)
C
```

```
C         DESCRIPTION:
C              THIS ROUTINE SCALES THE INPUT ARRAY TO AN INTEGER RANGE FOR
C         USE BY 'PLOT'.
C
C         CALLED BY:
C              'GRAPH1'
C              'GRAPH2'
C              'GRAPH5'
C         ARGUMENTS:
C              'IMIN' --- LOWER BOUND OF INTEGER SCALING RANGE.
C              'IMAX' --- UPPER BOUND OF INTEGER SCALING RANGE.
C              'XARRAY' --- INPUT ARRAY TO BE SCALED.
C              'NPNTS' --- NUMBER OF POINTS IN INPUT ARRAY.
C              'XMIN' --- MINIMUM VALUE IN INPUT ARRAY.
C              'XMAX' --- MAXIMUM VALUE IN INPUT ARRAY.
C              'IARRAY' --- RETURNS SCALED INTEGER ARRAY.
C
C         LOCAL VARIABLES:
C              'I' --- INTEGER; INDEXING VARIABLE.
C
C         SUBROUTINES CALLED:
C              NONE
C
          DIMENSION XARRAY(1), IARRAY(1)
          DO 10 I = 1, NPNTS
          IARRAY(I) = (XARRAY(I) - XMIN) * (IMAX - IMIN) /
     1    (XMAX - XMIN) + (IMIN)
10        CONTINUE
          RETURN
          END


          SUBROUTINE PLOT (IXARRY, IYARRY, NPNTS, INTENS)
```

```
C
C        DESCRIPTION:
C             THIS ROUTINE PLOTS THE INTEGER ARRAYS SUPPLIED BY THE
C        'GRAPH' SUBROUTINES.
C
C        CALLED BY:
C             'GRAPH1'
C             'GRAPH2'
C             'GRAPH5'
C
C        ARGUMENTS:
C             'IXARRY' --- SCALED INTEGER HORIZONTAL ARRAY.
C             'IYARRY' --- SCALED INTEGER VERTICAL ARRAY.
C             'NPNTS' --- NUMBER OF POINTS IN INTEGER ARRAYS.
C             'INTENS' --- INTENSITY (0 --- 7) DESIRED.
C
C        LOCAL VARIABLES:
C             'I' --- INTEGER; INDEXING VARIABLE.
C
C        SUBROUTINES CALLED:
C             'PRAMTR'
C             'POINTA'
C
         COMMON /DISFIL/ IDISPL(2000)
         DIMENSION IXARRY(1), IYARRY(1)
         CALL PRAMTR (2, INTENS, IDISPL)
         DO 10 I = 1, NPNTS
         CALL POINTA (IXARRY(I), IYARRY(I), 1, IDISPL)
10       CONTINUE
         RETURN
         END
```

```
      SUBROUTINE GRID1 (XMIN, XMAX, YMIN, YMAX, NLINES)
C
C     DESCRIPTION:
C          THIS ROUTINE GENERATES ONE COORDINATE GRID FOR 'GRAPH1',
C     'GRAPH2', 'GRAPH3', AND 'GRAPH4'.
C
C     CALLED BY:
C          'GRAPH1'
C          'GRAPH2'
C          'GRAPH3'
C          'GRAPH4'
C
C     ARGUMENTS:
C          'XMIN' --- MINIMUM VALUE ALONG HORIZONTAL AXIS.
C          'XMAX' --- MAXIMUM VALUE ALONG HORIZONTAL AXIS.
C          'YMIN' --- MINIMUM VALUE ALONG VERTICAL AXIS.
C          'YMAX' --- MAXIMUM VALUE ALONG VERTICAL AXIS.
C          'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C               AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C               LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C               FOR PHOTOGRAPHING.
C
C     LOCAL VARIABLES:
C          'XCOORD' --- INTEGER; CURRENT SCREEN X-COORDINATE.
C          'YCOORD' --- INTEGER; CURRENT SCREEN Y-COORDINATE.
C          'XAXCOR' --- INTEGER; SCREEN Y-COORDINATE OF X-AXIS.
C          'YAXCOR' --- INTEGER; SCREEN X-COORDINATE OF Y-AXIS.
C          'REDUCE' --- LOGICAL; .TRUE. FOR REDUCED PLOT SIZE.
C          'MINCOR' --- INTEGER; SCREEN COORDINATE FOR BOTTOM AND
C               LEFT BOUNDARIES.
C          'MAXCOR' --- INTEGER; SCREEN COORDINATE FOR TOP AND
C               RIGHT BOUNDARIES.
C          'LSPAC' --- INTEGER; GRID LINE SPACING MEASURED IN SCREEN
C               POINTS.
C          'I' --- INTEGER; INDEXING VARIABLE.
```

```
C           'YLABEL' --- REAL; NUMERICAL VALUE FOR LABELING HORIZONTAL
C                GRID LINES.
C           'XLABEL' --- REAL; NUMERICAL VALUE FOR LABELING VERTICAL
C                GRID LINES.
C
C      COMMON BLOCKS:
C           'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C
C      SUBROUTINES CALLED:
C           'PRAMTR'
C           'POINTA'
C           'LINE'
C           'NUMTXT'
C
       COMMON /DISFIL/ IDISPL(2000)
       INTEGER XCOORD, YCOORD, XAXCOR, YAXCOR
       LOGICAL REDUCE
       REDUCE = .FALSE.
       MINCOR = 149
       MAXCOR = 1023
       IF (NLINES .GE. 0) GO TO 10
       REDUCE = .TRUE.
C
C           RESET BOUNDARIES FOR REDUCED PLOT.
C
       MINCOR = 404
       MAXCOR = 767
       NLINES = - NLINES
10     CONTINUE
C
C           DRAW BOUNDARY LINES.
C
       CALL PRAMTR (2, 7, IDISPL)
       CALL POINTA (MINCOR, MINCOR, 0, IDISPL)
       CALL LINE (0, (MAXCOR - MINCOR), 1, IDISPL)
```

```
      CALL LINE ((MAXCOR - MINCOR), 0, 1, IDISPL)
      CALL LINE (0, (MINCOR - MAXCOR), 1, IDISPL)
      CALL LINE ((MINCOR - MAXCOR), 0, 1, IDISPL)
C
C           SET LINE SPACING.
C
      IF (NLINES .LT. 2) GO TO 90
      LSPAC = (MAXCOR - MINCOR) / (NLINES - 1)
C
C           DRAW HORIZONTAL LINES.
C
      CALL PRAMTR (2, 1, IDISPL)
C
C           DETERMINE IF X-AXIS WILL BE DISPLAYED.  INITIALIZE
C     'YCOORD', 'XAXCOR' TO BE USED IF NO AXIS.
C
      YCOORD = MINCOR
      XAXCOR = 0
      IF (YMIN .GT. 0.0 .OR. YMAX .LT. 0.0) GO TO 30
C
C           AXIS WILL BE DISPLAYED.  FIND X-AXIS Y-COORDINATE.
C
      XAXCOR = FLOAT (MINCOR) + ABS (YMIN) / (YMAX - YMIN) *
     1 FLOAT (MAXCOR - MINCOR) + 0.5
C
C           MOVE TO BOTTOM OF PLOT AT EVEN SPACING FROM AXIS.
C
      YCOORD = XAXCOR
      DO 20 I = 1, NLINES
      IF ((YCOORD - LSPAC) .LT. MINCOR) GO TO 30
      YCOORD = YCOORD - LSPAC
20    CONTINUE
30    DO 40 I = 1, NLINES
      CALL POINTA (MINCOR, YCOORD, 0, IDISPL)
      IF (YCOORD .EQ. XAXCOR) CALL PRAMTR (2, 7, IDISPL)
```

```
      CALL LINE ((MAXCOR - MINCOR), 0, 1, IDISPL)
C
C          LABEL LINE.  IF LINE IS AXIS, SET LABEL = 0.0.
C
      YLABEL = FLOAT (YCOORD - MINCOR) / FLOAT (MAXCOR - MINCOR)
     1  * (YMAX - YMIN) + YMIN
      IF (YCOORD .EQ. XAXCOR) YLABEL = 0.0
      CALL POINTA ((MINCOR - 145), (YCOORD - 11), 0, IDISPL)
      CALL PRAMTR (2, 6, IDISPL)
      CALL NUMTXT (YLABEL, 4)
      CALL PRAMTR (2, 1, IDISPL)
      YCOORD = YCOORD + LSPAC
      IF (YCOORD .GT. MAXCOR) GO TO 50
40    CONTINUE
50    CONTINUE
C
C          DRAW VERTICAL LINES.
C          DETERMINE IF Y-AXIS WILL BE DISPLAYED.  INITIALIZE
C     'XCOORD', 'YAXCOR' TO BE USED IF NO AXIS.
C
      XCOORD = MINCOR
      YAXCOR = 0
      IF (XMIN .GT. 0.0 .OR. XMAX .LT. 0.0) GO TO 70
C
C          AXIS WILL BE DISPLAYED.  FIND Y-AXIS X-COORDINATE.
C
      YAXCOR = FLOAT (MINCOR) + ABS (XMIN) / (XMAX - XMIN) *
     1  FLOAT (MAXCOR - MINCOR) + 0.5
C
C          MOVE TO LEFT OF PLOT AT EVEN SPACING FROM AXIS.
C
      XCOORD = YAXCOR
      DO 60 I = 1, NLINES
      IF ((XCOORD - LSPAC) .LT. MINCOR) GO TO 70
      XCOORD = XCOORD - LSPAC
```

```
60    CONTINUE
70    DO 80 I = 1, NLINES
      CALL POINTA (XCOORD, MINCOR, 0, IDISPL)
      IF (XCOORD .EQ. YAXCOR) CALL PRAMTR (2, 7, IDISPL)
      CALL LINE (0, (MAXCOR - MINCOR), 1, IDISPL)
C
C          LABEL LINE.  IF LINE IS AXIS, SET LABEL = 0.0.
C
      XLABEL = FLOAT (XCOORD - MINCOR) / FLOAT (MAXCOR - MINCOR)
     1  * (XMAX - XMIN) + XMIN
      IF (XCOORD .EQ. YAXCOR) XLABEL = 0.0
      CALL POINTA ((XCOORD - 67), (MINCOR - 30), 0, IDISPL)
      CALL PRAMTR (2, 6, IDISPL)
      CALL NUMTXT (XLABEL, 2)
      CALL PRAMTR (2, 1, IDISPL)
      XCOORD = XCOORD + LSPAC
      IF (XCOORD .GT. MAXCOR) GO TO 90
80    CONTINUE
90    IF (REDUCE) NLINES = - NLINES
      RETURN
      END



      SUBROUTINE GRID2 (XMIN, XMAX, YMIN, YMAX, NLINES, IYO, HEIGHT,
     1  MINCOR, MAXCOR)
C
C     DESCRIPTION:
C          THIS ROUTINE GENERATES COORDINATE GRIDS FOR 'GRAPH5',
C     AND IS CALLED ONCE FOR EACH PLOT.
C
C     CALLED BY:
C          'GRAPH5'
C
```

```
C         ARGUMENTS:
C              'XMIN' --- MINIMUM VALUE ALONG HORIZONTAL AXIS.
C              'XMAX' --- MAXIMUM VALUE ALONG HORIZONTAL AXIS.
C              'YMIN' --- MINIMUM VALUE ALONG VERTICAL AXIS.
C              'YMAX' --- MAXIMUM VALUE ALONG VERTICAL AXIS.
C              'NLINES' --- NUMBER OF LABELED GRID LINES, HORIZONTALLY
C                   AND VERTICALLY, DESIRED.  IF NEGATIVE, ABS (NLINES)
C                   LINES ARE DISPLAYED, BUT PLOT IS REDUCED TO 1/2 SIZE
C                   FOR PHOTOGRAPHING.
C              'IYO' --- SCREEN COORDINATE OF BOTTOM EDGE OF PLOT BEING
C                   GENERATED.
C              'HEIGHT' --- VERTICAL DIMENSION OF EACH PLOT, MEASURED IN
C                   SCREEN POINTS.
C              'MINCOR' --- SCREEN COORDINATE FOR BOTTOM AND LEFT
C                   BOUNDARIES.
C              'MAXCOR' --- SCREEN COORDINATE FOR TOP AND RIGHT
C                   BOUNDARIES.
C
C         LOCAL VARIABLES:
C              'XLSPAC' --- INTEGER; GRID LINE SPACING IN X-DIRECTION.
C              'YLSPAC' --- INTEGER; GRID LINE SPACING IN Y-DIRECTION.
C              'XCOORD' --- INTEGER; CURRENT SCREEN X-COORDINATE.
C              'YCOORD' --- INTEGER; CURRENT SCREEN Y-COORDINATE.
C              'XAXCOR' --- INTEGER; SCREEN Y-COORDINATE OF X-AXIS.
C              'YAXCOR' --- INTEGER; SCREEN X-COORDINATE OF Y-AXIS.
C              'REDUCE' --- LOGICAL; .TRUE. FOR REDUCED PLOT SIZE.
C              'I' --- INTEGER; INDEXING VARIABLE.
C
C         COMMON BLOCKS:
C              'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C
C         SUBROUTINES CALLED:
C              'PRAMTR'
C              'POINTA'
C              'LINE'
```

```
C           'NUMTXT'
C           'TEXT'
C
      COMMON /DISFIL/ IDISPL(2000)
      INTEGER HEIGHT, XLSPAC, YLSPAC, XCOORD, YCOORD, XAXCOR, YAXCOR
      LOGICAL REDUCE
      REDUCE = .FALSE.
C
C           DRAW BOUNDARY LINES, DISPLAY YMIN, YMAX.
C
      CALL PRAMTR (2, 6, IDISPL)
      CALL POINTA (MINCOR, IYO, 0, IDISPL)
      CALL LINE (0, HEIGHT, 1, IDISPL)
      CALL LINE ((MAXCOR - MINCOR), 0, 1, IDISPL)
      CALL LINE (0, (-HEIGHT), 1, IDISPL)
      CALL LINE ((MINCOR - MAXCOR), 0, 1, IDISPL)
      CALL POINTA ((MINCOR - 145), IYO, 0, IDISPL)
      CALL NUMTXT (YMIN, 3)
      CALL POINTA ((MINCOR - 145), (IYO + HEIGHT - 23), 0, IDISPL)
      CALL NUMTXT (YMAX, 3)
C
C           SET LINE SPACING.
C
      IF (NLINES .GE. 0) GO TO 10
      NLINES = - NLINES
      REDUCE = .TRUE.
10    CONTINUE
      IF (NLINES .LT. 2) GO TO 90
      XLSPAC = (MAXCOR - MINCOR) / (NLINES - 1)
      YLSPAC = HEIGHT / (NLINES - 1)
C
C           DRAW HORIZONTAL LINES.
C
      CALL PRAMTR (2, 1, IDISPL)
C
```

```
C             DETERMINE IF X-AXIS WILL BE DISPLAYED.  INITIALIZE
C         'YCOORD', 'XAXCOR' TO BE USED IF NO AXIS.
C
          YCOORD = IYO
          XAXCOR = 0
          IF (YMIN .GT. 0.0 .OR. YMAX .LT. 0.0) GO TO 30
C
C             AXIS WILL BE DISPLAYED.  FIND X-AXIS Y-COORDINATE.
C
          XAXCOR = FLOAT (IYO) + ABS (YMIN) / (YMAX - YMIN) *
     1    FLOAT (HEIGHT) + 0.5
C
C             MOVE TO BOTTOM OF PLOT AT EVEN SPACING FROM AXIS.
C
          YCOORD = XAXCOR
          DO 20 I = 1, NLINES
          IF ((YCOORD - YLSPAC) .LT. IYO) GO TO 30
          YCOORD = YCOORD - YLSPAC
20        CONTINUE
30        DO 40 I = 1, NLINES
          CALL POINTA (MINCOR, YCOORD, 0, IDISPL)
          IF (YCOORD .EQ. XAXCOR) CALL PRAMTR (2, 7, IDISPL)
          CALL LINE ((MAXCOR - MINCOR), 0, 1, IDISPL)
          IF (YCOORD .EQ. XAXCOR) CALL PRAMTR (2, 1, IDISPL)
          YCOORD = YCOORD + YLSPAC
          IF (YCOORD .GT. (IYO + HEIGHT)) GO TO 50
40        CONTINUE
50        CONTINUE
C
C             DRAW VERTICAL LINES.
C             DETERMINE IF Y-AXIS WILL BE DISPLAYED.  INITIALIZE
C         'XCOORD', 'YAXCOR' TO BE USED IF NO AXIS.
C
          XCOORD = MINCOR
          YAXCOR = 0
```

```
      IF (XMIN .GT. 0.0 .OR. XMAX .LT. 0.0) GO TO 70
C
C           AXIS WILL BE DISPLAYED.  FIND Y-AXIS X-COORDINATE.
C
      YAXCOR = FLOAT (MINCOR) + ABS (XMIN) / (XMAX - XMIN) *
     1  FLOAT (MAXCOR - MINCOR) + 0.5
C
C           MOVE TO LEFT OF PLOT AT EVEN SPACING FROM AXIS.
C
      XCOORD = YAXCOR
      DO 60 I = 1, NLINES
      IF ((XCOORD - XLSPAC) .LT. MINCOR) GO TO 70
      XCOORD = XCOORD - XLSPAC
60    CONTINUE
70    DO 80 I = 1, NLINES
      CALL POINTA (XCOORD, IYO, 0, IDISPL)
      IF (XCOORD .EQ. YAXCOR) CALL PRAMTR (2, 7, IDISPL)
      CALL LINE (0, HEIGHT, 1, IDISPL)
      IF (XCOORD .EQ. YAXCOR) CALL PRAMTR (2, 1, IDISPL)
      XCOORD = XCOORD + XLSPAC
      IF (XCOORD .GT. MAXCOR) GO TO 90
80    CONTINUE
90    IF (REDUCE) NLINES = - NLINES
      RETURN
      END



      SUBROUTINE LEGEND (NPLOTS, MINCOR)
C
C     DESCRIPTION:
C           THIS ROUTINE DISPLAYS A LEGEND OF THE ARRAYS PLOTTED
C     BENEATH THE COORDINATE GRID(S).
C
```

```
C       CALLED BY:
C            'GRAPH1'
C            'GRAPH2'
C
C       ARGUMENTS:
C            'NPLOTS' --- NUMBER OF PLOTS, ORDERED FROM Y1 TO Y4.
C            'MINCOR' --- SCREEN COORDINATE FOR BOTTOM BOUNDARY.
C
C       LOCAL VARIABLES:
C            'TXT' --- DATA ARRAY(4); TEXT FOR LEGENDS.
C            'IX' --- INTEGER; X-COORDINATE FOR BEGINNING OF EACH
C                 LEGEND.
C            'INTENS' --- INTEGER; INTENSITY (1, 3, 5, OR 7) FOR
C                 LEGENDS.
C            'I' --- INTEGER; INDEXING VARIABLE.
C
C       COMMON BLOCKS:
C            'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C
C       SUBROUTINES CALLED:
C            'POINTA'
C            'PRAMTR'
C            'LINE'
C            'TEXT'
C
        COMMON /DISFIL/ IDISPL(2000)
        DIMENSION TXT(4)
        DATA TXT(1), TXT(2), TXT(3), TXT(4) /'Y1', 'Y2', 'Y3', 'Y4'/
        IX = 410
        INTENS = 7
        DO 10 I = 1, NPLOTS
        CALL POINTA (IX, (MINCOR - 43), 0, IDISPL)
        CALL PRAMTR (2, INTENS, IDISPL)
        CALL LINE (20, 0, 1, IDISPL)
        CALL POINTA ((IX + 20), (MINCOR - 55), 0, IDISPL)
```

```
      CALL PRAMTR (2, 6, IDISPL)
      CALL TEXT (TXT(I), 2, IDISPL)
      IX = IX + 100
      INTENS = INTENS - 2
10    CONTINUE
      RETURN
      END




      SUBROUTINE NUMTXT (XVALUE, NFIGS)
C
C     DESCRIPTION:
C          THIS ROUTINE PLACES CODE IN THE DISPLAY FILE 'IDISPL' TO
C     CAUSE THE GT-44 DISPLAY TO GENERATE CHARACTERS CORRESPONDING TO
C     A NUMERICAL VALUE.  DISPLAY IS IN 'E' FORMAT.  THE NUMBER IS DIS-
C     PLAYED AT THE POINT AND INTENSITY EXISTING AT THE TIME OF THE
C     CALL.
C
C     CALLED BY:
C          'GRID1'
C          'GRID2'
C
C     ARGUMENTS:
C          'XVALUE' --- NUMERICAL INPUT VALUE FOR WHICH CHARACTERS
C               ARE GENERATED.
C          'NFIGS' --- NUMBER OF SIGNIFICANT FIGURES DESIRED.
C
C     LOCAL VARIABLES:
C          'NUM' --- DATA ARRAY(10); TEXT FOR A GIVEN VALUE.
C          'SIGN' --- LOGICAL; .TRUE. IS POSITIVE.
C          'IEXP' --- INTEGER; VALUE OF EXPONENT.
C          'VALUE' --- LOCAL COPY OF 'XVALUE'.
C          'I' --- INTEGER; INDEXING VARIABLE.
```

```
C            'INDEX' --- INTEGER; POINTER FOR DATA ARRAY.
C
C       COMMON BLOCKS:
C            'DISFIL':  'IDISPL' --- DISPLAY PROCESSOR ARRAY.
C
C       SUBROUTINES CALLED:
C            'TEXT'
C
        COMMON /DISFIL/ IDISPL(2000)
        DIMENSION NUM(10)
        LOGICAL SIGN
        DATA NUM(1), NUM(2), NUM(3), NUM(4), NUM(5), NUM(6), NUM(7),
     1  NUM(8), NUM(9), NUM(10) /'1', '2', '3', '4', '5', '6', '7',
     1  '8', '9', '0'/
        IEXP = 0
C
C       SIGN POSITIVE = .TRUE.
C       SIGN NEGATIVE = .FALSE.
C
        VALUE = XVALUE
        SIGN = .TRUE.
        IF (VALUE .EQ. 0.0) GO TO 30
        IF (VALUE .LT. 0.0) SIGN = .FALSE.
        VALUE = ABS (VALUE)
 10     IF (VALUE .LT. 1.0) GO TO 20
        VALUE = VALUE / 10.0
        IEXP = IEXP + 1
        GO TO 10
 20     IF (VALUE .GE. 0.1) GO TO 30
        VALUE = VALUE * 10.0
        IEXP = IEXP - 1
        GO TO 20
 30     IF (SIGN) CALL TEXT ('+0.', 3, IDISPL)
        IF (.NOT. SIGN) CALL TEXT ('-0.', 3, IDISPL)
        DO 40 I = 1, NFIGS
```

```
      VALUE = VALUE * 10.0
      INDEX = INT (VALUE)
      IF (INDEX .EQ. 0) INDEX = 10
      CALL TEXT (NUM(INDEX), 1, IDISPL)
      VALUE = VALUE - FLOAT (INT (VALUE))
40    CONTINUE
      IF (IEXP .LT. 0) CALL TEXT ('E -', 3, IDISPL)
      IF (IEXP .GE. 0) CALL TEXT ('E +', 3, IDISPL)
      IEXP = IABS (IEXP)
      INDEX = IEXP
      IF (INDEX .GE. 10) GO TO 50
      IF (INDEX .EQ. 0) INDEX = 10
      CALL TEXT ('0', 1, IDISPL)
      GO TO 60
50    INDEX = INT (FLOAT (IEXP) / 10.0)
      IF (INDEX .EQ. 0) INDEX = 10
      CALL TEXT (NUM(INDEX), 1, IDISPL)
      INDEX = IEXP - INDEX * 10
      IF (INDEX .EQ. 0) INDEX = 10
60    CALL TEXT (NUM(INDEX), 1, IDISPL)
      RETURN
      END
```

APPENDIX B

EXAMPLE PROGRAM LISTINGS

Source listings of the example programs described herein are provided on the following pages

```
              PROGRAM SINES1
C
C     DESCRIPTION:
C          "SINES1" IS A TEST DRIVER AND DEMONSTRATION PROGRAM FOR THE
C     "GRAPH" SUBROUTINES.  CALLS ARE MADE TO "GRAPH1", "GRAPH2", AND
C     "GRAPH5" TO SHOW THE DIFFERENCES IN DISPLAY FORMAT AND RELATIVE
C     ADVANTAGES OF EACH.
C
C     LOCAL VARIABLES:
C          'SINE1' --- REAL ARRAY(301); SIN(T) VALUES
C          'SINE2' --- REAL ARRAY(301); SIN(T) VALUES
C          'T' --- REAL; ARGUMENT FOR SIN FUNCTIONS
C          'PI' --- REAL; CONSTANT = 3.1415927
C          'I' --- INTEGER; INDEXING VARIABLE
C
C     SUBROUTINES CALLED:
C          'GRAPH1'
C          'GRAPH2'
C          'GRAPH5'
C
      DIMENSION SINE1(301), SINE2(301), T(301)
      PI = 3.1415927
C
C          PRINT HEADING.
C
      WRITE (5, 1000)
1000  FORMAT (5(/), 10X, 'SINES1:  DEMONSTRATION PROGRAM FOR "GRAPH" ',
     1 'SUBROUTINES')
C
C          FIRST PLOT.
C
      WRITE (5, 2000)
2000  FORMAT (///, 2X, 'PLOT 1.', /, 5X, 'CREATE AN ARRAY OF ',
     1 '301 POINTS REPRESENTING THE FUNCTION:', /, 2X, 'V(T) = 3.5 * ',
     1 'SIN (2 * PI * T + 0.0) VOLTS FROM T = -1.0 TO 1.0,', /, 2X,
```

```
     1   'AND PLOT IT USING "GRAPH1".')
         DO 10 I = 1, 301
         T(I) = FLOAT (I - 1) / 150.0 - 1.0
         SINE1(I) = 3.5 * SIN (2.0 * PI * 1.0 * T(I) + 0.0)
10       CONTINUE
         CALL GRAPH1 (2, 301, 5, T, SINE1)
C
C             NEXT PLOT.
C
         WRITE (5, 3000)
3000     FORMAT (/, 2X, 'PLOT 2.', /, 5X, 'ADD A D.C. OFFSET OF +5.0 ',
     1   'VOLTS AND REPLOT.')
         DO 20 I = 1, 301
         SINE1(I) = SINE1(I) + 5.0
20       CONTINUE
         CALL GRAPH1 (2, 301, 5, T, SINE1)
C
C             NEXT PLOT.
C
         WRITE (5, 4000)
4000     FORMAT (/, 2X, 'PLOT 3.', /, 5X, 'REPLOT FROM 5.8 TO 7.8, ',
     1   'THIS TIME WITH AN OFFSET OF -9.6 VOLTS.')
         DO 30 I = 1, 301
         T(I) = FLOAT (I - 1) / 150.0 + 5.8
         SINE1(I) = -9.6 + 3.5 * SIN (2.0 * PI * 1.0 * T(I) + 0.0)
30       CONTINUE
         CALL GRAPH1 (2, 301, 5, T, SINE1)
C
C             NEXT PLOT.
C
         WRITE (5, 5000)
5000     FORMAT (/, 2X, 'PLOT 4.', /, 5X, 'ASSUME AN AMPLIFIER WITH A ',
     1   'GAIN OF 100.  WITH AN INPUT OF', /, 2X, '2.0 MILLIVOLTS ',
     1   'PEAK-TO-PEAK, PLOT INPUT AND OUTPUT ON THE SAME GRAPH.')
         DO 40 I = 1, 301
```

```
          T(I) = FLOAT (I - 1) / 150.0
          SINE1(I) = 0.002 * SIN (2.0 * PI * 1.0 * T(I) + 0.0)
          SINE2(I) = 0.2 * SIN (2.0 * PI * 1.0 * T(I) + PI / 8.0)
40        CONTINUE
          WRITE (5, 6000)
6000      FORMAT (5(/), 5X, '"NICE, BUT THE INPUT IS TOO SMALL TO SEE.  ',
     1    /, 2X, 'BESIDES, I WANT TO SEE THE AMOUNT OF PHASE SHIFT."',
     1    5(/), 5X, 'O.K., TERMINATE THIS, AND WE WILL TRY SOMETHING ',
     1    'ELSE...')
          CALL GRAPH1 (3, 301, 5, T, SINE1, SINE2)
          WRITE (5, 7000)
7000      FORMAT (/, 2X, 'PLOT 5.', /, 5X, 'THIS TIME, PLOT THE SAME ',
     1    'FUNCTIONS USING "GRAPH2".')
          CALL GRAPH2 (3, 301, 5, T, SINE1, SINE2)
C
C              NEXT PLOT.
C
          WRITE (5, 8000)
8000      FORMAT (/, 5X, 'PLOT 6.', /, 5X, 'YOU MIGHT ALSO LIKE TO USE ',
     1    '"GRAPH5".  NOW YOU CAN SEE', /, 2X, 'RELATIVE AMPLITUDES AS ',
     1    'WELL AS PHASES, BUT YOU SACRIFICE SOME', /, 2X, 'RESOLUTION.')
          CALL GRAPH5 (3, 301, 5, T, SINE1, SINE2)
          WRITE (5, 9000)
9000      FORMAT (/, 10X, 'END OF DEMONSTRATION; THANKS FOR WATCHING.')
          END
```

```
            PROGRAM SINES2
C
C     DESCRIPTION:
C          THIS PROGRAM GENERATES ARRAYS OF POINTS CORRESPONDING
C     TO SINE FUNCTIONS FOR T = 0.000 TO 1.000 AT 0.005 INTERVALS.
C     CALLS ARE THEN MADE TO THE 'GRAPH' SUBROUTINES FOR DEMON-
C     STRATION.
C
C     LOCAL VARIABLES:
C          'SIN1' --- REAL ARRAY(201) OF SIN(T) VALUES
C          'SIN2' --- REAL ARRAY(201) OF SIN(T) VALUES
C          'SIN3' --- REAL ARRAY(201) OF SIN(T) VALUES
C          'SIN4' --- REAL ARRAY(201) OF SIN(T) VALUES
C          'T' --- REAL, ARGUMENT FOR SINE FUNCTIONS
C          'PI' --- REAL, CONSTANT = 3.1415927
C          'I' --- INTEGER, INDEXING VARIABLE
C
C     SUBROUTINES CALLED:
C          'GRAPH1'
C          'GRAPH2'
C          'GRAPH5'
C
      DIMENSION SIN1(201), SIN2(201), SIN3(201), SIN4(201), T(201)
      PI = 3.1415927
      DO 10 I = 1, 201
      T(I) = FLOAT (I - 1) / 200.0
      SIN1(I) = 1.0 * SIN (2.0 * PI * 1.0 * T(I) + 0.0)
      SIN2(I) = 2.0 * SIN (2.0 * PI * 2.0 * T(I) + 0.0)
      SIN3(I) = 2.0 * SIN (2.0 * PI * 2.0 * T(I) + PI / 2.0)
      SIN4(I) = 3.0 * SIN (2.0 * PI * 3.0 * T(I) + PI)
10    CONTINUE
      WRITE (5, 1000)
1000  FORMAT (5(/), 10X, 'SINES2:  DEMONSTRATION PROGRAM FOR "GRAPH" ',
     1 'SUBROUTINES', //, 5X,
     1 'SIN1 = 1.0 SIN (2 * PI * 1 * T + 0.0)', /, 5X,
```

```
     1  'SIN2 = 2.0 SIN (2 * PI * 2 * T + 0.0)', /, 5X,
     1  'SIN3 = 2.0 SIN (2 * PI * 2 * T + PI/2)', /, 5X,
     1  'SIN4 = 3.0 SIN (2 * PI * 3 * T + PI)', //, 5X,
     1  'PLOTS GENERATED:', /, 10X,
     1  '1.  SIN1, SIN2, SIN3, SIN4 VS. T BY "GRAPH1"', /, 10X,
     1  '2.  SIN1, SIN2, SIN3, SIN4 VS. T BY "GRAPH2"', /, 10X,
     1  '3.  PLOT 1 REDUCED FOR PHOTOGRAPHING, 4 GRID LINES', /, 10X,
     1  '4.  PLOT 2 REDUCED FOR PHOTOGRAPHING, 3 GRID LINES', /, 10X,
     1  '5.  SIN1, SIN2, SIN3, SIN4 VS. T BY "GRAPH5"', /, 10X,
     1  '6.  PLOT 5 REDUCED FOR PHOTOGRAPHING', /////)
C
C           FIRST PLOT.
C
        WRITE (5, 2000)
2000    FORMAT (2X, 'PLOT 1 NOW DISPLAYED')
        CALL GRAPH1 (5, 201, 5, T, SIN1, SIN2, SIN3, SIN4)
C
C           NEXT PLOT.
C
        WRITE (5, 3000)
3000    FORMAT (2X, 'PLOT 2 NOW DISPLAYED')
        CALL GRAPH2 (5, 201, 5, T, SIN1, SIN2, SIN3, SIN4)
C
C           NEXT PLOT.
C
        WRITE (5, 4000)
4000    FORMAT (2X, 'PLOT 3 NOW DISPLAYED')
        CALL GRAPH1 (5, 201, -4, T, SIN1, SIN2, SIN3, SIN4)
C
C           NEXT PLOT.
C
        WRITE (5, 5000)
5000    FORMAT (2X, 'PLOT 4 NOW DISPLAYED')
        CALL GRAPH2 (5, 201, -3, T, SIN1, SIN2, SIN3, SIN4)
C
```

```
C             NEXT PLOT.
C
        WRITE (5, 6000)
6000    FORMAT (2X, 'PLOT 5 NOW DISPLAYED')
        CALL GRAPH5 (5, 201, 5, T, SIN1, SIN2, SIN3, SIN4)
C
C             NEXT PLOT.
C
        WRITE (5, 7000)
7000    FORMAT (2X, 'PLOT 6 NOW DISPLAYED')
        CALL GRAPH5 (5, 201, -5, T, SIN1, SIN2, SIN3, SIN4)
C
C             PRINT END MESSAGE.
C
        WRITE (5, 8000)
8000    FORMAT (/, 10X, 'END OF DEMONSTRATION; THANKS FOR WATCHING.')
        END
```

```
      PROGRAM LISS
C
C         DESCRIPTION:
C              TEST DRIVER AND DEMONSTRATION PROGRAM FOR "GRAPH3" AND
C         "GRAPH4" SUBROUTINES.  PLOTS SINE FUNCTIONS AGAINST SINE
C         FUNCTIONS OF VARYING FREQUENCY AND PHASE RELATIONSHIPS.
C
C         LOCAL VARIABLES:
C              'SINE1' --- REAL ARRAY(201); SIN(T) VALUES
C              'SINE2' --- REAL ARRAY(201); SIN(T) VALUES
C              'SINE3' --- REAL ARRAY(201); SIN(T) VALUES
C              'DUMMY' --- REAL ARRAY(201); DUMMY CALLING ARRAY
C              'PI' --- REAL; CONSTANT = 3.1415927
C              'I' --- INTEGER; INDEXING VARIABLE
C              'T' --- REAL; ARGUMENT FOR SIN FUNCTIONS
C
C         SUBROUTINES CALLED:
C              'GRAPHS3'
C              'GRAPHS4'
C
          DIMENSION SINE1(201), SINE2(201), SINE3(201), DUMMY(201)
          PI = 3.1415927
C
C              PRINT HEADING.
C
          WRITE (5, 500)
500       FORMAT (5(/), 10X, 'LISS:  DEMONSTRATION PROGRAM FOR "GRAPH3" ',
     1   'AND "GRAPH4"')
C
C              FIRST PLOT.
C
          WRITE (5, 1000)
1000      FORMAT (/, 2X, 'PLOT 1.', /, 5X, 'PLOT A SINE FUNCTION OF 1 ',
     1   'HERTZ AGAINST ITSELF')
          DO 10 I = 1, 201
```

```
      T = FLOAT (I - 1) / 200.0 - 0.5
      SINE1(I) = 3.0 * SIN (2.0 * PI * 1.0 * T + 0.0)
10    CONTINUE
      CALL GRAPH3 (2, 201, 5, SINE1, DUMMY, DUMMY, SINE1)
C
C           NEXT PLOT.
C
      WRITE (5, 2000)
2000  FORMAT (/, 2X, 'PLOT 2.', /, 5X, 'SHIFT PHASE OF ONE SINE',
     1 ' FUNCTION BY 90 DEGREES AND REPLOT.')
      DO 20 I = 1, 201
      T = FLOAT (I - 1) / 200.0 - 0.5
      SINE2(I) = 3.0 * SIN (2.0 * PI * 1.0 * T + PI / 2.0)
20    CONTINUE
      CALL GRAPH3 (2, 201, 5, SINE2, DUMMY, DUMMY, SINE1)
C
C           NEXT PLOT.
C
      WRITE (5, 3000)
3000  FORMAT (/, 2X, 'PLOT 3.', /, 5X, 'PLOT SINE FUNCTION OF ',
     1 'THREE TIMES FREQUENCY AGAINST ORIGINAL,', /, 2X,
     1 'TO FORM LISSAJOUS PATTERN.')
      DO 30 I = 1, 201
      T = FLOAT (I - 1) / 200.0 - 0.5
      SINE2(I) = 3.0 * SIN (2.0 * PI * 3.0 * T + PI / 2.0)
30    CONTINUE
      CALL GRAPH3 (2, 201, 5, SINE2, DUMMY, DUMMY, SINE1)
C
C           NEXT PLOT.
C
      WRITE (5, 4000)
4000  FORMAT (/, 2X, 'PLOT 4.', /, 5X, 'ADD ANOTHER FUNCTION ',
     1 'OF FIVE TIMES FREQUENCY, BUT HALF AMPLITUDE.')
      DO 40 I = 1, 201
      T = FLOAT (I - 1) / 200.0 - 0.5
```

```
      SINE3(I) = 1.5 * SIN (2.0 * PI * 5.0 * T + PI / 2.0)
40    CONTINUE
      CALL GRAPH3 (3, 201, 5, SINE2, SINE3, DUMMY, SINE1)
C
C          NEXT PLOT.
C
      WRITE (5, 5000)
5000  FORMAT (/, 2X, 'PLOT 5.', /, 5X, 'REPLOT USING "GRAPH4".')
      CALL GRAPH4 (3, 201, 5, SINE2, SINE3, DUMMY, SINE1)
C
C          PRINT END MESSAGE.
C
      WRITE (5, 6000)
6000  FORMAT (/, 10X, 'END OF DEMONSTRATION; THANKS FOR WATCHING.')
      END
```

LIST OF REFERENCES

American National Standards Institute. "U. S. A. Standard FORTRAN," U. S. A. S. Report, X3.7, New York, 1966.

Conley, Stevwn W. "FORTRAN Subroutines for Inter-Active Graphics on the GT-44 System," University of Arizona, Department of Electrical Engineering Internal Report #280, 1975.

Korn, Granino A., and John V. Wait. Digital Continuous-Time System Simulation, Prentice-Hall, Inc., 1977.

Kreitzberg, Charles B., and Ben Shneiderman. The Elements of FORTRAN Style, Harcourt Brace Jovanovich, Inc., 1972.