A SIMPLE GRAPHICAL DISPLAY PROGRAM FOR PHYSICS*

P.J. Moffa
Department of Physics and Astronomy
University of Maryland, College Park, Maryland 20742

August 1977

U. of Md. TR #78-016
U. of Md. PP #78-041

UNIVERSITY OF MARYLAND
DEPARTMENT OF PHYSICS AND ASTRONOMY
COLLEGE PARK, MARYLAND

MASTER

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

ORO 5126-27
U. of Md. TR #78-016
U. of Md. PP #78-041

A SIMPLE GRAPHICAL DISPLAY PROGRAM FOR PHYSICS*

P.J. Moffa

Department of Physics and Astronomy
University of Maryland, College Park, Maryland 20742

August 1977

ABSTRACT

A simple subroutine based on the Tektronix Plot-10/Advanced Graphing II package to be used with a Tektronix CRT terminal is presented. It is specifically designed for the needs of a physicist, but contains enough options to allow for a wider use.

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

* Work supported by U.S. Energy Research and Development Administration.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

The purpose of this report is to describe a simple subroutine that can easily be incorporated into a working program when one desires the option of making two dimensional plots on a Tektronix CRT Terminal as part of the program. It utilizes a small fraction of the capabilities of the Tektronix Plot-10/Advanced Graphing II package, but should be general enough for most physics applications.

Specifically, it allows one to make a plot of a number of calculated curves and to compare these (if desired) with discrete sets of points (data). This can be done for any combination of linear and logarithmic axes, and with curves and data each containing different numbers of points. It also provides for the outputting in standard formats of up to three lines of information both above and below the plot. Figure 1 displays a simple example of the type of plots that can be produced.

The basic components of this plotting subroutine will now be discussed. The listing of the subroutine that made the plot in Figure 1 is contained in the Appendix. A very clear and complete description of all the subroutines used in it can be found in the Tektronix Plot-10/ Advanced Graphing II User's Manual, published by Tektronix, Inc., Beaverton, Oregon, Document No. 062-1530-00. All of the page references given in this report will refer to this manual. It is hoped, however, that for most applications, referring to the manual will be unnecessary.

Lines 3-13 of this subroutine are necessary to get the arrays to be plotted, the number of points in each array, and some parameters into the subroutine.

This could also have been done through common blocks, and in the calling line of the subroutine. However, as will be explained later, reading and writing from a disc seems to be the best way. We have now transmitted to the

plot routine a set of parameters, two two-dimensional arrays, (X,Y) and (T,D) and the number of points in each, namely NPT pairs in (X,Y) and NDT pairs in (T,D).

Both INITT and BINITT are necessary to initialize the system. The argument of INITT is the terminal transmission rate in characters per second (one tenth of the baud rate). NPTS must be called with the number of points in the first array to be plotted. XTYPE and YTYPE define the type of axes required. The default value is one and that gives a linear scale, while the value two requests a logarithmic scale. Hence, we have requested a linear x-axis vs. a logarithmic y-axis (see Figure 1). CHECK must now be called with the array to be plotted. Among other things, CHECK calculates the maximum and minimum values to be plotted and sets up the axes accordingly. If, for some reason, different limits on the axes are required, then the subroutines DLIMX and DLIMY should be used. They are called with the desired minimum and maximum values as follows: CALL DLIMX (XMIN, XMAX) and similarly for DLIMY.

The actual plotting is initiated by calling DSPLAY, again with the points to be plotted. In order to make additional plots, once the axes have been drawn, the subroutine CPLOT is utilized. In our example, DSPLAY has drawn a continuous line which is the default option. In order to obtain a discrete plot we need to call LINE. The values of the argument allowed are given on Page 62. The argument zero implies no line between points and one (default) signifies a continuous line. As described on Page 62, different dashed and dot-dashed lines can also be produced. Once one signifies no line, then the SYMBOL option should be used. This created the circles in Figure 1. Some of the options are: zero (default) - no symbol, one-circle, two - "X", four-square, five-star, etc. A full list of options is on page 65.

Note that before CPLOT is called with the array to be plotted both NPTS and CHECK must again be called. Hence, once the axes have been drawn by DISPLAY any number of other curves or discrete point sets may be plotted with the sequence: NPTS, CHECK, CPLOT. By using different LINE and SYMBL options different kinds of lines and points can be obtained.

We have now finished the actual plotting section of the subroutine. Lines 26-37 are needed to display information both above and below the plot. In order to do this the subroutines MOVABS and HLABEL have been used. The arguments of MOVABS are the integer (x,y) coordinates of the point on the screen where the cursor is desired. The usable screen is described by an (x,y) coordinate system with the origin in the lower left corner. In raster units (128 raster units equals one inch) the screen is 1023 units long in the x-direction, and 780 units high in the y-direction. It was found that the MOVABS routine would not function properly unless a call to one of the routines like HLABEL followed. HLABEL and VLABEL will label the x and y axes respectively. They are called with two integer arguments, the first being the number of symbols in the label and the second being the label in ASCII code. The ASCII code is given at the end of the manual. Instead of having to translate standard alpha-numeric symbols into this code, I chose to use standard format print statements. The first MOVABS call locates the cursor three lines above the plot at the left edge of the available screen area. The HLABEL call is a dummy call (it prints a blank). Now all the information necessary to describe the plot can be outputted on the three top lines. The example program also contains another MOVABS call that locates the cursor at the line below the graph. Here, for example, parameters that change from run to run can be printed out. Especially here, the use of HLABEL (or NOTATE, see p. 50) would be trouble-

some since the changing parameters would have to be translated into ASCII code for each run.

Lines 38 and 39 of the program are important. They cause the program to pause after the plot is finished. Otherwise, if there was more outputting in the rest of the program it would appear at the bottom of the screen if all three lines of space are not utilized there. With this READ statement included, the plot can be examined, a copy can be made, and then by inputting a zero the program will continue.

Now that the subroutine has been explained, I will discuss how to incorporate it into an existing program. First, I would strongly suggest that an option be put in at the plot calling location to allow a bypass of the call. In this way, the program can be run through batch processing or from other terminals. Now it can be seen why all the information necessary to plot should be written on a disc. If a run is not made on the Tektronix CRT terminal, but a plot is desired, the output can be saved in a file and plotted at a later time. This would require a trivial modification of subroutine PLOT in order to make it a main program. If the information were transmitted to PLOT through a COMMON or in the calling statement, this option would be lost.

In order to access the necessary subroutines used in PLOT, the library containing them must be incorporated into the program. This library is available on both the 1100/40 and the 1108. The statement LIB UOM*AG2PLOT10. must be added to the program map. This causes the mapping to take considerably longer and also causes the absolute that is produced to be larger. For this reason it has been suggested that the PLOT subroutine be added to a program that is already working. Finally, in order to avoid the "output interrupt" message from appearing on the graph, the message, @CON NOIM, should be typed upon logging in.

It is hoped that the subroutine described here is simple enough and yet versatile enough to be readily used in many physics and related applications.

The generous support of the University of Maryland Computer Science Center is gratefully acknowledged.

APPENDIX

```

1:      SUBROUTINE PLOT
2:      DIMENSION X(80),Y(80),T(80),D(80)
3:      REWIND 20
4:      READ(20,29)PAR1,PAR2,PAR3,PAR4
5:      READ(20,30)NPT,NDT
6:      29  FORMAT(8E15.5)
7:      30  FORMAT(2I5)
8:      DO 1 I=1,NPT
9:      READ(20,29) X(I),Y(I)
10:     1    CONTINUE
11:     DO 33 I=1,NDT
12:     READ (20,29)T(I),D(I)
13:     33   CONTINUE
14:     CALL INITT(120)
15:     CALL BINITT
16:     CALL NPTS(NPT)
17:     CALL XTYPE(1)
18:     CALL YTYPE(2)
19:     CALL CHECK(X,Y)
20:     CALL DSPLAY(X,Y)
21:     CALL NPTS(NDT)
22:     CALL LINE(0)
23:     CALL SYMBL(1)
24:     CALL CHECK(T,D)
25:     CALL CFLOT(T,D)
26:     CALL MOVABS(0,760)
27:     CALL HLABEL(1,0)
28:     PRINT 67
29:     67   FORMAT(28X,'THIS IS A',//,28X,'TEST PLOT')
30:     CALL MOVABS(0,60)
31:     CALL HLABEL(1,0)
32:     PRINT 7,PAR1
33:     PRINT8,PAR2,PAR3
34:     PRINT 9,PAR4
35:     7    FORMAT(3X,'PAR1= ',F15.5)
36:     8    FORMAT('  PAR2= ',F15.5,'  PAR3= 'F15.5)
37:     9    FORMAT('  PAR4= 'F15.5)
38:     READ 5,1DU
39:     5    FORMAT()
40:     REWIND 20
41:     RETURN
42:     END

```

THIS IS A
TEST PLOT

Figure 1.