ANL-6161
Physics and Mathematics
(TID-4500, 15th Ed.)
AEC Research and
Development Report

ARGONNE NATIONAL LABORATORY
Argonne, Illinois

FORTRAN PROGRAMMING TECHNIQUES FOR GRAPH PLOTTING ON THE IBM-704 COMPUTER

by

Charles Erwin Cohn

Reactor Engineering Division

May, 1960

Operated by The University of Chicago
under
Contract W-31-109-eng-38

TABLE OF CONTENTS

FORTRAN PROGRAMMING TECHNIQUES FOR GRAPH PLOTTING ON THE IBM-704 COMPUTER

Charles Erwin Cohn

INTRODUCTION

In many cases, the numerical results of digital computer problems have increased usefulness if they are presented in graphical form. Relationships, trends, and unusual features in the results are more easily seen on a graph than in a list of numbers. For example, the flux and power results from reactor statics codes and the residuals from various types of least squares fits all greatly benefit from graphical presentation.

Ordinarily, graphs are produced either by laborious manual plotting or by elaborate automatic graph-plotting equipment. However, the IBM-717 peripheral line printer used with the IBM-704 computer is well suited to the task of graph plotting, and has the advantage that the graphs are obtained with no additional operations. On this printer there are 119 usable character spaces in a line (with program control) so that a tape record containing one character and filled in with blanks will allow any printer character to be printed in any space. This allows points to be placed along the horizontal axis. Points are placed vertically by paper spacing. In addition, legends, coordinates, and grid and border markings may also be printed. Thus, graphs which are completely ready for publication may be prepared.

To illustrate the possibilities, this report will describe the programming techniques which have been used to produce a number of different kinds of graphs. Two examples of the graphs produced are included. These are half-size photographic reductions of the graphs as they came off the printer. It was found that all necessary programming could be done using FORTRAN II, with the exception of the placement of characters along a line in proportion to a specified abscissa. Therefore, the subroutine AN J501 (see Appendix) was programmed in symbolic machine language for this purpose. The use of FORTRAN has the advantage of making these techniques available to the nonprofessional programmer.

The major drawback of the technique is the limited resolution afforded by the construction of the printer. About 100 spaces are available horizontally, and about 100 lines give a graph of page-size proportions. Thus, the resolution is about 1% in both directions, which is adequate for many purposes.

Other routines(1,2,3,4) for this type of graph plotting are in existence. These are subroutines which set up the entire graph. With one exception(4) they are not adapted to FORTRAN, and all use the on-line printer. These routines have the disadvantage of being limited to one particular graph size and format. On the other hand, the programming techniques to be presented here are good for many different types of graphs, so that a type can be chosen which is best suited to the job at hand.

The remainder of this report will consider first, the method of plotting the graph when the coordinates of all points are known; and secondly, some possible methods of setting up scales and assigning the coordinates of the points. The coding sequences to be presented are intended to be incorporated as parts of a larger program which uses the graph plotting for output.

PLOTTING THE GRAPH

The subroutine AN J501 writes a record on the output tape, which has the plotted character as the 20th to the 120th character, depending on the absolute value of an integer coordinate. The remainder of the record contains blanks, except the first character which contains a + to cause space suppression. The subroutine is called by the statement

```
CALL GRAPH(M,N,1HX)
```

here M is the coordinate, N is the number of the tape upon which the record is written, and X is the Hollerith character to be plotted. This character appears as the 20th character on the record for M = 0, as the 120th character for M = 200, and at proportionally intermediate positions for intermediate values. Since the first character of each record is used for printer carriage control, 18 characters at the left of the graph remain available for legends, coordinates, etc. A basic routine which will plot a graph is as follows:

```
 1  WRITE ØUTPUT TAPE 3,13
 2  WRITE ØUTPUT TAPE 3,11
 3  DØ9I=1,102
 4  WRITE ØUTPUT TAPE 3,12
 5  DØ8J=1,N
 6  IF(I-LØCY(J))8,7,8
 7  CALL GRAPH(LØCX(J),3,1H*)
 8  CØNTINUE
 9  CØNTINUE
10  WRITE ØUTPUT TAPE 3,11
```

```
11 FØRMAT(120H[18 spaces] [102 dots])
12 FØRMAT(120H[18 spaces] . [100 spaces].)
13 FØRMAT(1H1)
```

This coding plots N points with the character *. The coordinate of each point is a pair of integers LØCX(J), LØCY(J). The former is used by the GRAPH subroutine to determine the horizontal position of the point, while the latter specifies the number of the line (going from the top) on which the point appears. Assigning these integers to each point will be considered in the next section.

The detailed operation of this coding sequence is as follows: Statement 1 writes a record to restore the printer. Statement 2 writes a record to print a row of dots forming the upper border of the graph. Statements 3 through 9 form a DØ loop which produces the body of the graph, line by line. Statement 4 writes a record which spaces the paper and prints a dot at the left and right edges of the graph, these dots forming the vertical borders. Then, for each line, the DØ loop consisting of statements 5 through 8 is entered. Here the array LØCY of line numbers assigned to each point is scanned. If any of these is equal to the current line number I, the GRAPH subroutine is called for and writes a record to plot the point on that line. After all lines have been processed, statement 10 writes a record to print the lower border of the graph.

Many elaborations of this basic routine are possible. For example, if it is desired to print a title or other legend in the middle of the graph, it can be done as follows:

```
    IF(I-LØC)n,100,n
100 WRITE ØUTPUT TAPE 3,101, [List]
101 FØRMAT(1H+ etc.)
```

This may be placed either between statements 4 and 5 of the basic routine, in which case n would be 5, or between 8 and 9, in which case n would be 9. The number of the line on which the title is to be printed is given by LØC, and is normally specified as a constant. [List] refers to a problem number, etc. The FØRMAT statement produces a + as the first character of the tape record in order to overprint the title on the desired line. If the approximate shape of the curve is known, the title can be placed in a location where it will not be likely to be overprinted by points. Of course, titles can also be placed above or below the graph.

If more than one curve or set of data is to be printed, the basic routine can be modified by repeating statements 5 through 8 once for each extra curve, making appropriate changes in variables and using a different plotting character in each CALL GRAPH statement to distinguish the curves. For a larger number of curves, a more convenient form would be:

```
 1  WRITE ØUTPUT TAPE 3,23
 2  WRITE ØUTPUT TAPE 3,21
 3  DØ19I=1,102
 4  WRITE ØUTPUT TAPE 3,22
 5  DØ18K=1,M
 6  NK=N(K)
 7  DØ17J=1,NK
 8  IF(I-LØCY(J,K))17,9,17
 9  GØ TØ(10,12,14,16),K
10  CALL GRAPH(LØCX(J,K),3,1H*)
11  GØ TØ 17
12  CALL GRAPH(LØCX(J,K),3,1H+)
13  GØ TØ 17
14  CALL GRAPH(LØCX(J,K),3,1H$)
15  GØ TØ 17
16  CALL GRAPH(LØCX(J,K),3,1HX)
17  CØNTINUE
18  CØNTINUE
19  CØNTINUE
20  WRITE ØUTPUT TAPE 3,21
21  FØRMAT(120H[18 spaces] [102 dots])
22  FØRMAT(120H[18 spaces].[100 spaces].)
23  FØRMAT(1H1)
```

The above has been coded for up to four curves, but can be extended for more in an obvious manner. In this example, if there are always four curves, the variable M, the number of curves, can be replaced by the constant 4, but M may also vary between 1 and 4. Here N(K) is the number of points for the K'th curve, and LØCX and LØCY are now two-dimensional arrays. Now, for each line, the entire LØCY array is scanned, and when a correspondence with the current line number is found, a computed GØ TØ selects the CALL GRAPH statement containing the correct plotting character for the given curve.

One precaution that should be observed in the choice of plotting characters is to choose those that are symmetrically disposed within their allotted spaces. For example, a dot is not suitable for use as a plotting character because it appears near the bottom of its space. Thus, a curve plotted with dots would appear artificially low compared to curves plotted with other characters.

Note: In these coding sequences two or more CØNTINUE statements in sequence have occasionally been used for expository convenience. The extra statements have no effect on the object program, but can be eliminated if desired.

ASSIGNING COORDINATES AND SCALES

This section will deal with the problem of assigning coordinates to the points, and assigning and printing scale markings. There are obviously a large number of ways of doing this. Therefore, we shall not attempt to consider all possible methods, but only those which have been actually used and proven. Since both coordinates of each point are supplied to the graph-plotting routine in the form of integers, the techniques here described can be used to set up both axes. However, there is one difference. The FORTRAN built-in subroutines which convert floating-point numbers to integers do not round off, but merely truncate. Since rounding off is desirable for maximum plotting smoothness, the GRAPH subroutine has provisions to accomplish it. For the Y axis, the following function subprogram will convert a floating-point number to an integer with rounding:

```
  FUNCTION LEVEL (X)
  LEVEL=XINTF(X)
  IF(MØDF(X,1.0)-0.5)2,1,1
1 LEVEL=LEVEL+1
2 RETURN
```

There are, as mentioned, a large number of ways of assigning coordinates and scales, and the choice of a particular one depends on the nature of the data and the features of the graph that are to be emphasized. However, there are a couple of general considerations which apply to all cases. First, to take fullest advantage of the available resolution, the plotted points should be spread over practically the full extent of the graph. Thus, it is desirable to have the scaling of the graph determined by the range of the data to be plotted. Secondly, if scale markings are to be placed on the graph, the scaling must be such as to give an integral number of spaces between the markings.

The simplest graph-plotting applications are those where the independent variable goes in discrete steps. These include plots of residuals from least squares fits and of flux and power plots from reactor statics codes, where the independent variable can be taken as the number of the point. In such cases the programming is simplified by having the independent variable run vertically. This is necessary in any case if the number of points is greater than 101.

For illustration, a simple routine for plotting the residuals from a least squares fit will be shown. Given is an array RESID(I) with N members containing the residuals.

```
 1  RESMAX=0.0
 2  DØ3I=1,N
 3  RESMAX=MAX1F(RESMAX,ABSF(RESID(I)))
 4  RESMAX=RESMAX*1.05
 5  BØUND=100.0/RESMAX
 6  WRITE ØUTPUT TAPE 3,12,RESMAX
 7  WRITE ØUTPUT TAPE 3,13
 8  DØ10I=1,N
 9  WRITE ØUTPUT TAPE 3,14,I
10  CALL GRAPH(LEVEL(100.0+RESID(I)*BØUND),3,1H*)
11  WRITE ØUTPUT TAPE 3,13
12  FØRMAT(15H1RESIDUAL PLØT,F10.4,11H FULL SCALE)
13  FØRMAT(120H[18 spaces] [102 dots])
14  FØRMAT(I18,102H .[49 spaces].[49 spaces].)
```

Statements 1 through 5 establish the scale and the remainder are used to plot the graph. Here full scale on the graph is made 5% greater than the largest absolute residual in order to keep all points well inside the edge. This graph contains a vertical dotted line down the center to serve as a zero axis, because both positive and negative residuals will occur. The variable RESMAX is printed out as the full-scale coordinate of the X axis. If it is desired to round it off, or to print other scale markings, this can be done with methods yet to be shown.

Two typical scaling methods will now be shown. The first to be presented will illustrate logarithmic plotting on the vertical axis with scale marks placed at every decade. This is the method used in the first sample graph. Assume the array Y(I) contains the base 10 logarithms of the ordinates of the N points to be plotted, with YMAX and YMIN being the maximum and minimum, respectively, of this array. Also assume that the array LØCX(I) contains the abscissae of the points already in integer form ready for plotting. Then the coding becomes:

```
 1  SPAN=YMAX-YMIN
 2  NPDEC=XFIXF(100.0/SPAN)
 3  SCALE=FLØATF(NPDEC)
 4  INIDEC=2+XFIXF(MØDF(YMAX,1.0)*SCALE)
 5  ØRIGIN=INTF(YMAX)+FLØATF(INIDEC)/SCALE
 6  INDEX=XINTF(YMAX)+1
 7  DØ8I=1,N
 8  LØCY(I)=LEVEL((ØRIGIN-Y(I))*SCALE)
 9  WRITE ØUTPUT TAPE 3,23
10  WRITE ØUTPUT TAPE 3,24
11  DØ21I=1,102
12  IF(XMØDF(I-INIDEC,NPDEC))13,15,13
13  WRITE ØUTPUT TAPE 3,25
14  GØ TØ 17
15  INDEX=INDEX-1
16  WRITE ØUTPUT TAPE 3,26,INDEX
17  DØ20J=1,N
18  IF(I-LØCY(J))20,19,20
19  CALL GRAPH(LØCX(J),3,1H*)
20  CØNTINUE
21  CØNTINUE
22  WRITE ØUTPUT TAPE 3,24
23  FØRMAT(1H1)
24  FØRMAT(120H[18 spaces] [102 dots])
25  FØRMAT(120H[18 spaces].[100 spaces].)
26  FØRMAT(5H 1.0(I3,112H)----------[100 spaces]-)
```

In this program, statement 1 determines the range of logarithms covered. Statement 2 establishes the integral number of lines occupied by one decade NPDEC, statement 3 obtaining this in floating-point form as

SCALE. Statement 4 obtains the line number INIDEC on which the first decade mark is to appear. Here the 2 is provided to space the points away from the extreme top edge. Statement 5 obtains ØRIGIN, the value of Y corresponding to the line just above the top line of the graph, while statement 6 obtains INDEX, an integer one greater than the power of ten to be printed at the first decade mark. The DØ loop consisting of statements 7-8 then assigns line numbers LØCY to the various points. The graph is then plotted in the manner previously described. Statement 12 tests each line to determine whether it is an even decade. If it is, INDEX is decreased by one and printed as a scale division in place of the normal border dots. It will be noted that a dash is used as a scale marker since it is centered in its space. If desired, a scale line could, of course, be printed across the whole graph.

When the range of the data is less than two decades, the previous scheme is not too good, since in many cases only one decade mark would be printed and this would not be adequate to indicate the scale of the data. In fact, for very small spans no decade marks at all might appear. Therefore, another method, in which coordinates are printed only at the top and bottom of the graph, was developed. This can be done for either a linear or logarithmic scale, but will be shown here for the latter. This is the system used on the second sample graph.

The simplest way to do this would be to print the value of the maximum ordinate at the top line of the graph, and the value of the minimum ordinate at the bottom line. It would be desirable to terminate these numbers after just enough decimal places to give a range of the desired magnitude. For example, the maximum and minimum might be 0.99936 and 0.99512, respectively. Then it would be desirable to make the graph limits 1.000 and 0.995, giving a range of 0.005. However, it would not be desirable to make the limits 1.00 and 0.99, because this would give a range of 0.01 and the points would be bunched in the upper half of the graph. The desired effect can be obtained by dividing the maximum and minimum by 10^L, where L is the characteristic of the logarithm of the difference between the maximum and minimum. The quotients are then truncated to floating-point integers, 1.0 being added to the one for the maximum. Then these integers are multiplied by 10^L to obtain the limits. If the minimum is smaller than the difference, this procedure will give a zero lower limit, resulting again in points bunched at the top of the graph. Therefore, in such cases L is decreased by one and the process repeated until a finite lower limit is obtained. Further refinements might be required for linear plotting, where negative ordinates are admissible.

The following routine will perform the above functions. Input is assumed to be the same as in the previous example.

```
 1  DIFF=TENLØG(10.**YMAX-10.**YMIN)
 2  IF(DIFF)3,5,5
 3  L=XINTF(DIFF-1.0)
 4  GØ TØ 6
 5  L=XINTF(DIFF)
 6  TØPLIM=(INTF((10.**YMAX)/(10.**L))+1.0)*(10.**L)
 7  BØTLIM=INTF((10.**YMIN)/(10.**L))*(10.**L)
 8  IF(BØTLIM)9,9,11
 9  L=L-1
10  GØ TØ 6
11  TØPLØG=TENLØG(TØPLIM)
12  SCALE=(TØPLØG-TENLØG(BØTLIM))/102.0
13  DØ14I=1,N
14  LØCY(I)=LEVEL((TØPLØG-Y(I))/SCALE)
15  WRITE ØUTPUT TAPE 3,28
16  WRITE ØUTPUT TAPE 3,29
17  DØ25I=1,102
18  WRITE ØUTPUT TAPE 3,30
19  DØ22J=1,N
20  IF(I-LØCY(J))22,21,22
21  CALL GRAPH(LØCX(J),3,1H*)
22  CØNTINUE
23  IF(I-1)25,24,25
24  WRITE ØUTPUT TAPE 3,31,TØPLIM
25  CØNTINUE
26  WRITE ØUTPUT TAPE 3,31,BØTLIM
27  WRITE ØUTPUT TAPE 3,29
28  FØRMAT(1H1)
29  FØRMAT(120H[18 spaces] [102 dots])
30  FØRMAT(120H[18 spaces].[100 spaces].)
31  FØRMAT(1H+F8.6,9H---------)
```

Here the function subprogram TENLØG is designed to adapt the existing library natural-log function to give base 10 logs and to assure the return of zero for unity argument.

```
  FUNCTIØN TENLØG(X)
  IF(X-1.0)3,2,3
2 TENLØG=0.0
  GØ TØ 4
3 TENLØG=0.4342944819*LØGF(X)
4 RETURN
```

In the preceding program, statement 1 finds the logarithm of the difference between the maximum and minimum. Statements 2 through 5 determine the characteristic, L Then the upper and lower limits are determined by statements 6 and 7. Statements 8, 9, and 10 check for a zero lower limit and make appropriate corrections. Then the array LØCY is obtained and the graph is plotted. Statements 23 and 24 write the top coordinate on the output tape, while statement 26 writes the bottom coordinate.

These routines have been written for one curve, but they can be extended to more than one curve by methods previously described. In such cases, of course, YMAX and YMIN are the overall maximum and minimum. These techniques can also be used to establish the horizontal coordinates. Vertical coordinate lines at variable horizontal locations can be easily plotted with the GRAPH subroutine, but scale numbers at variable locations along the horizontal axis cannot be produced with these methods. In the sample graphs, the abscissae occur at fractional-inch intervals, but it was desired to show a centimeter scale. With such discrete intervals, assignment of horizontal coordinates is very simple. However, since the inch and centimeter scales are incommensurable, producing the centimeter scale on the printer would not be satisfactory. Therefore, it was decided to have a separate centimeter scale hand drawn and to place it on each graph and make a photographic image. This is economical because of the large number of graphs to be processed. The graphs shown here have not yet been so treated.

The examples shown here obviously cover only a small fraction of the possibilities inherent in these techniques. Such things as plotting in polar coordinates and other unusual coordinate systems have not been touched upon. Ingenious programmers will undoubtedly find many uses for these methods.

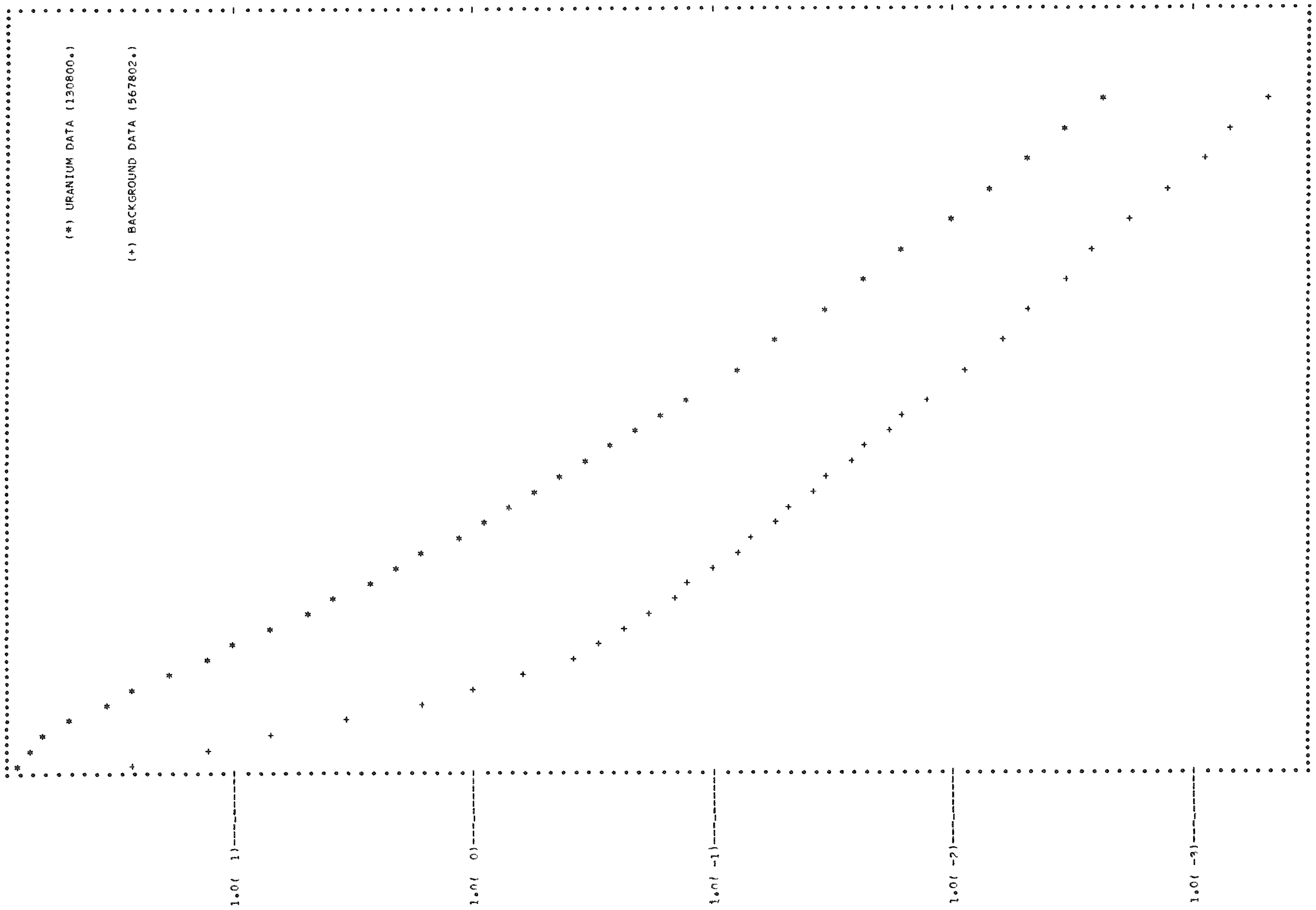
TIMING AND OPERATING CONSIDERATIONS

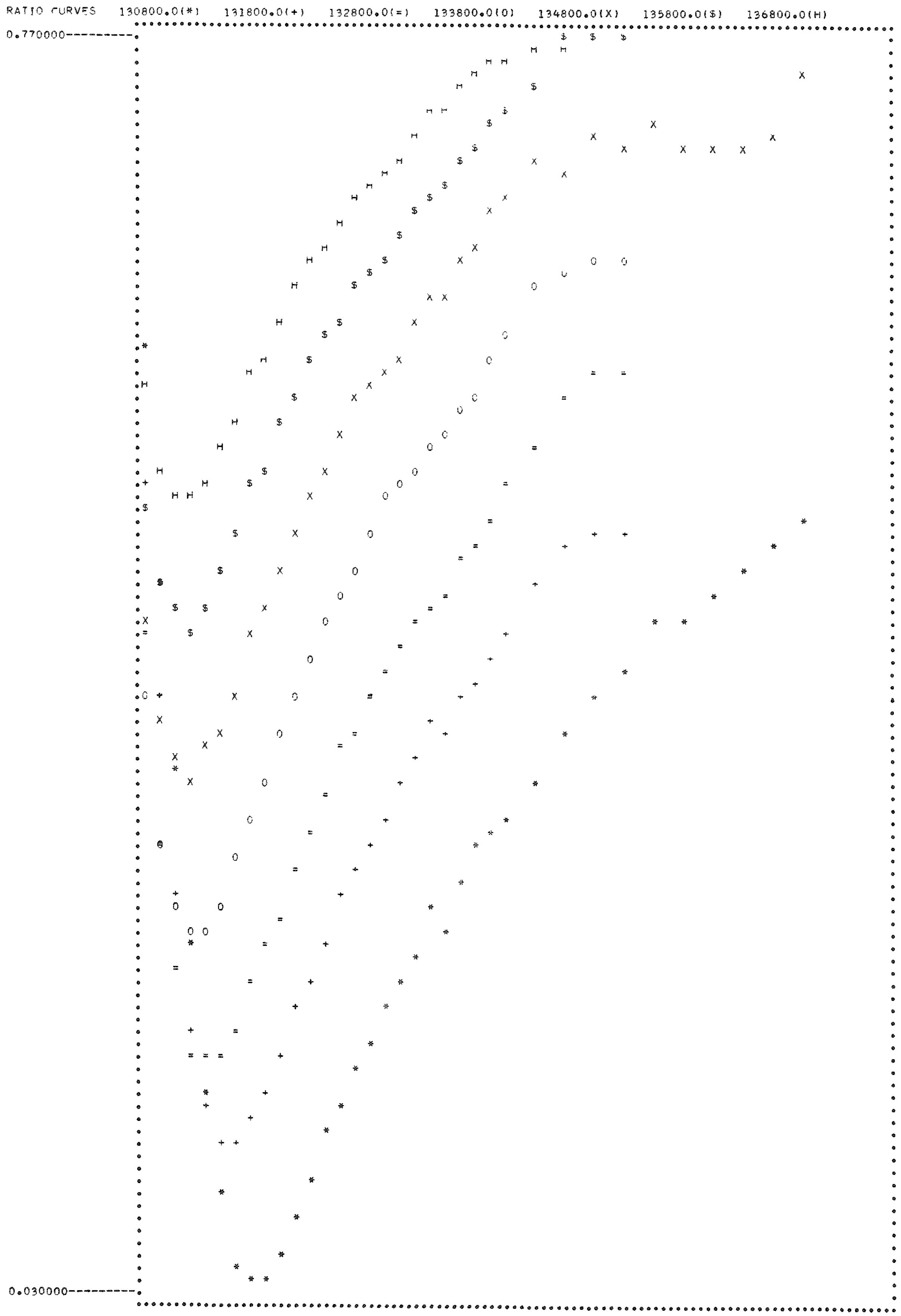
Graphs such as those shown here can be set up and written on tape in about 10 seconds after all necessary data are in storage. This time is primarily taken up by tape motions.

Actual machine operations are unaffected by the use of graph plotting. The output tapes are printed on Program Control in the usual manner. The only precaution to be observed is that the automatic form skip must be off if the graphs extend over more than one sheet of the forms.

REFERENCES

1. Lockheed Aircraft Corporation, Program CL PLT1, SHARE Distribution 131, October 2, 1956.
2. Lockheed Aircraft Corporation, Program CL PLT2, SHARE Distribution 236, April 22, 1957.
3. Westinghouse Electric Corporation, Program WH20, SHARE Distribution 284, June 21, 1957.
4. MURA-520, Program MU PPP5, September 1, 1959.


(*) URANIUM DATA (130800.)
(+) BACKGROUND DATA (567802.)
1.0(1)
1.0(0)
1.0(-1)
1.0(-2)
1.0(-3)


RATIO CURVES
130800.0(*)
131800.0(+)
132800.0(=)
133800.0(O)
134800.0(X)
135800.0($)
136800.0(H)
0.770000
0.030000

APPENDIX

AN J501, FORTRAN II Graph-plotting Subroutine (SHARE Distribution 763)

PURPOSE

FORTRAN II subroutine to write a BCD record on tape to print a selected character in a selected position in the field represented by the last 101 spaces on the off-line printer. The position of the character is specified by a fixed-point number. The record produced will not space the paper on the printer.

RESTRICTIONS

This subroutine requires one on-line tape unit and a peripheral printer or peripheral printer simulator.

USAGE

The subroutine is called through use of the FORTRAN statement

```
CALL GRAPH (M, N, 1HX)
```

Here M is the fixed-point variable whose absolute value specifies the position of the plotted character. The range of the variable is from 0 to 200, with 0 representing the left edge of the plotting field, 200 representing the right edge, and intermediate values representing proportionally intermediate positions. N is the logical address (from 1 to 10) of the tape unit on which the record is to be written, and X is the character which it is desired to plot. Since the subroutine does not space the paper, more than one point may be plotted on a line by successive entries into the subroutine. Line spacing must be handled by appropriate WRITE OUTPUT TAPE statements.

Since only the last 101 spaces on the printer are available for plotting, the first 19 may be used for other purposes. The first space will always be used for carriage control, so the next 18 spaces are available for coordinates, legends, etc. Legends and reference marks can be printed on the graph by use of WRITE OUTPUT TAPE statements with appropriate Hollerith formats.

Fifty-one storage locations are required.

METHOD

The subroutine is programmed in SAP language. It works essentially by adding a constant to M and then dividing to find the number of six-character BCD words of blanks which should precede the plotted character. (The first character of the first word is made a + to suppress spacing on the printer.) When the character to be plotted is gotten from the main program, it appears as the first character in a BCD word with the rest of the word filled in with blanks. This word is written on tape as the last word of the record after the character has been rotated into the proper position as controlled by the remainder from the aforementioned division.

Although the plotting range is 101 spaces or 100 intervals, the range of M is made from 0 to 200 in order to allow a shifting and rounding operation to be performed. This is desirable to increase the accuracy of plotting, since FORTRAN does not round when truncating a floating-point number to fixed point. An input-output delay order is included to wait for tape disconnect before returning control to the calling program as a precaution in case the calling program uses the MQ immediately.

CODING INFORMATION

Timing: The time in milliseconds required for one pass through the subroutine is equal to 12.4 + 0.067 (M + 26).

Error stops: There are two error stops possible with this subroutine.

1. The program stops with HPR115,7_8 in the storage register if the absolute value of M is greater than 200. The offending value then appears in the decrement field of the accumulator, and the index registers are as set by the calling program. Pushing START returns control to the calling program and no tape writing is done.

2. A divide check stop is possible on a DVH order. However, the programming is such that the conditions for a divide check should never be realized. Thus, such a stop would be an indication of machine malfunction.

Cards: The symbolic deck consists of 64 cards numbered J5010001-64. The binary deck consists of four cards numbered ANJ50101-4, comprising a program card required by the FORTRAN II BSS Loader and three relocatable cards containing the subroutine.

```
ANJ501 REM 0064 07209                                            J5010001
       REM SUBROUTINE GRAPH                                      J5010002
       FUL                      PROGRAM CARD                     J5010003
       ORG 0                                                     J5010004
       FOR 0,0,4                                                 J5010005
       OCT 675125473144         CHECKSUM                         J5010006
       PZE 51,0,0               NO. OF WDS IN SUBR.              J5010007
       PZE 0                    NO COMMON STORAGE                J5010008
       BCD 1GRAPH               NAME OF SUBROUTINE               J5010009
       PZE 1,0,0                ENTRY POINT                      J5010010
       REL                                                       J5010011
       ORG 0                    START PROGRAM                    J5010012
       HTR 0                    SAVE IR1                         J5010013
 GRAPH SXD GRAPH-1,1                                             J5010014
       CLA 1,4                                                   J5010015
       STA COORD                PLANT LOC OF COORDINATE          J5010016
       CLA 2,4                                                   J5010017
       STA TAPE                 PLANT LOC OF TAPE UNIT NUMBER    J5010018
       CLA 3,4                                                   J5010019
       STA CHRCTR               PLANT LOC OF CHARACTER           J5010020
  TAPE CLA **                   GET TAPE NUMBER                  J5010021
       ARS 18                                                    J5010022
       ADD ADDRS                FORM TAPE ADDRESS                J5010023
       STA SELECT               PLANT TAPE ADDRESS               J5010024
 COORD CLA **                   GET COORDINATE                   J5010025
       SSP                                                       J5010026
       CAS MAX                  IS COORDINATE TOO LARGE          J5010027
       TRA ERROR                YES                              J5010028
       NOP                      NO                               J5010029
       ADD GAP                                                   J5010030
       LRS 19                                                    J5010031
       RND                      ROUND COORDINATE                 J5010032
       LRS 35                                                    J5010033
       DVH SIX                  FIND NO OF BCD WDS PRECEDING MARK J5010034
       STQ NOWRDS               STORE NUMBER OF WORDS            J5010035
       CHS                                                       J5010036
       ADD SIX                                                   J5010037
       LRS 35                                                    J5010038
       MPY SIX                                                   J5010039
       LLS 35                                                    J5010040
       STA SHIFT                                          J5010041
CHRCTR LDQ **             GET CHARACTER                   J5010042
 SHIFT RQL **             PLACE CHARACTER                 J5010043
       STQ LASTWD                                         J5010044
       LXA NOWRDS,1                                       J5010045
SELECT WRS **             SELECT TAPE                     J5010046
       CPY FRSTWD                                         J5010047
  COPY CPY BLANKS                                         J5010048
       TIX COPY,1,1                                       J5010049
       CPY LASTWD                                         J5010050
       LXD GRAPH-1,1      RESTORE IR1                     J5010051
       IOD                WAIT FOR TAPE DISCONNECT        J5010052
       TRA 4,4            RETURN                          J5010053
 ERROR HPR 77,7           ERROR STOP                      J5010054
       TRA 4,4            ERROR RETURN                    J5010055
 ADDRS PZE 128,0,0                                        J5010056
   SIX PZE 6,0,0                                          J5010057
   MAX PZE 0,0,200                                        J5010058
   GAP PZE 0,0,26                                         J5010059
FRSTWD BCD 1+                                             J5010060
BLANKS BCD 1                                              J5010061
NOWRDS BSS 1                                              J5010062
LASTWD BSS 1                                              J5010063
       END 0                                              J5010064
```