# GRAPHICS IN APL

Alfred M. Bork
Physics Computer Development Project
University of California
Irvine, California

This document describes an experimental graphic facility within APL. The terminals are assumed to be inexpensive timeshared graphic terminals equipped with an APL character set. We first describe functions in a grpahic workspace, and then APL primitives for graphing.

## User Plotting Functions - Workspace DRAW

The following functions are available as a group called SEE in the DRAW workspace:

| | |
|---|---|
| DRAW | NOSCALE |
| TEK | CENTER |
| ARDS | SET |
| TERMINAL | INT |
| ERASE | VS |
| SCALE | DASH |
| AXES | |

1. DRAW produces a curve on the screen and determines where the curve is to appear.

   It imitates, at least partially, the APL/360 PLOT function. PLOT produces point plots or histograms on the typewriter. Its general form is

   ```
   20  50  PLOT  X  VS  Y
   ```

   VS is an APL function, combining the two vectors X and Y ($\rho X = \rho Y$) into an array suitable for use in PLOT. The numbers in front have an effect somewhat like windowing - they determine the "size" of the plot, the number of lines and the number of characters in each line.

   The corresponding graphic function is DRAW. It follows the general specification for PLOT. DRAW only plots one curve each time, in either two or three dimensions. It seems natural to let the left argument of DRAW specify a "window," a section of the screen on which the picture is to appear. It can use the function VS to combine arrays for plotting, or 1 by N, 2 by N, or 3 by N arrays can be used directly as the right argument.

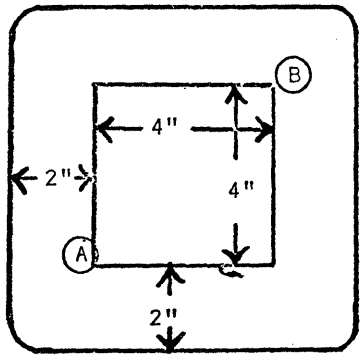
   We need four numbers for a window, the coordinates of points A and B in inches from the lower left corner, as in the diagram, so DRAW can be preceded by a four-vector, literals or a variable.

   If the left argument to DRAW is a scalar, the window currently in effect applies; the value of the scalar is ignored. The initial default window is the largest square possible touching the lower and right edges.

   DRAW can have a third argument on the right for three-dimensional plotting. Thus

   ```
   2  2  6  6  DRAW  VX  VS  VY  VS  VZ
   ```

   plots the three velocity components VX, VY, and VZ in a window as shown:




We must have $\rho VX = \rho VY = \rho VZ$. VS is extended to allow 3-D arguments to DRAW.

The distances for the first arugment of DRAW are measured in inches from the lower left corner. After a DRAW, the cursor goes to the next writeable line. DRAW does not erase the screen, so it can be used to overplot curves.

2. SCALE, NOSCALE, and CENTER determine the placement of the picture within the window.

SCALE determines the user coordinates for the smallest and largest value, the corners of the current window. The general form is

SCALE A

For a 2D plot, A is a four vector; the first two components are the maximum and minimum values of the horizontal variable, and the next 2 of the vertical variable. For a 3D curve, ρA = 6; the last two components determine the scale for the third, or Z, axis.

So for a 2D graph where the minimum values are -1.5, and the maximum 3, for both variables, we have

SCALE ¯1.5 3 ¯1.5 3

The default for SCALE is to scale the data to occupy the full window, finding the maxima and minima.

NOSCALE returns to this default case after the use of SCALE. It has no arguments.

CENTER places the origin of the coordinate system in the center of the windown, and then scales to fit the window. It has no arguments.

SCALE, NOSCALE, and CENTER do not return a value.

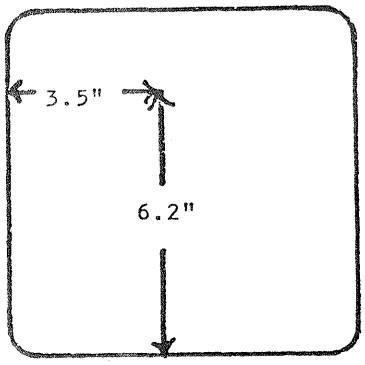
3. ERASE, HOME, and SET control utility functions on the CRT screen.

Screen control functions perform operations on the CRT, as in these examples:

ERASE - erases screen, sets cursor at upper left corner

HOME - sets cursor at upper left corner

3.5 SET 6.2 - the cursor is set to the position on the screen shown, with measurements in inches.




4. AXES draws axes corresponding to the current scaling and windowing conventions. It has no arguments, and returns no value.

5. DASH causes the next curve only to be dashed.

6. INT establishes an interval. It is often useful in plotting to establish a vector of equally spaced values. The function for this is INT, as in this example:

A ← INT -6 6 100

This function sets up a vector of 100 equally spaced values between -6 and 6, and assigns it to A.

7. ARDS, TEK and TERMINAL set the type of terminal in use. On initial release the APL Graphics facility supports three terminals: Tektronix 4002, Tektronix 4010, and ARDS

100. As these terminals have different graphic coding conventions, it is necessary for APL to know which is in use in order to draw curves.

In later versions of APL Graphic software terminal specification may use a system command. However the initial system employs the following functions to set terminal type:

ARDS - sets terminal as ARDS
TEK 4002 - sets terminal as Tektronix 4002
TEK 4010 - sets terminal as Tektronix 4010
TERMINAL - queries the user as to which terminal he is using, and takes appropriate action. Intended for use in graphics programs which do not suppose a highly knowledgeable graphics user.

The default terminal if no terminal is selected is the Tektronix 4010.

8. Later Features

Eventually we will allow the user to define what graphic terminal he is using, perhaps with a ")TERM" system command. This affects both code translation and graphic data.

We will also allow the user to "store" a picture, the actual graphic data; this may be done with a new data type, "graphic."

We will later allow for the possibility of graphic input, through tablet, light pen, joystick, mouse, etc.

DRAW may also eventually be called upon to construct functions in the complex plane, assuming that "complex" is defined as an APL data type.

## Underlying APL Primitives for Graphics

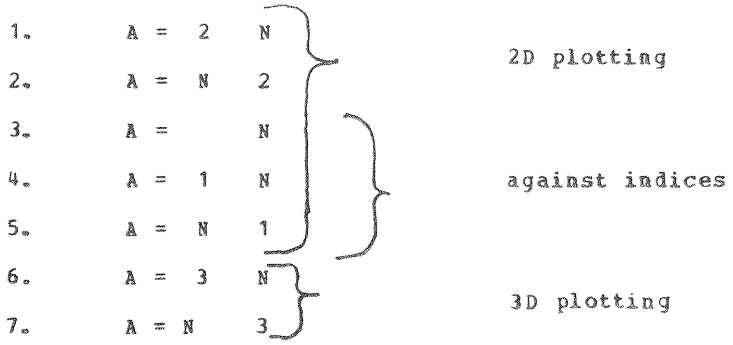
1. ⍞ - Quad backspace zero

This is the basic graphic output function. Its use is in the form

⍞ --- A

For ASCII terminals incapable of drawing APL characters the expression "$QO" can be used.

The following are legal possibilities for xxA:




This leaves several ambiguous cases. If ⍴A = 2 3, we interpret this as a 2D plot of 3 points. If ⍴A = 3 2, we understand a 3D plot of 2 points. If ⍴A is 2 1 or 3 1, a point is plotted. If ⍴A is 1 2, or 1 3, 2 or 3 points are plotted.

On initial implementation cases 1, 2, 6, and 7 are available.

Scaling, windowing, and the terminal currently in effect control the conversion of the arrays to graphic form. The graphic data is set to the terminal; the first bytes of data set the graphic mode and the last return to character mode. The screen is not erased by this operator.

This primitive is available and known to the user; the character ⍞ is legal.

If a single number is assigned to ⍞, an ASCII control charcter is sent. The correspondence between integers and control characters is in ascending code order. Other As give a RANK ERROR.

At a later time ⍞ will be used for input, both for interrogating the terminal (as with the TEK 4010) and for graphic input from tablets, joysticks, mouses, etc.

2. ⍒ - Quad backspace S

Memory inset - for controlling graphic conversion. ⍒ transfers data entered with SETPOINT, SCALE, AND DRAW (window data) to the code for generating graphic data. We can do this with a command of the form:

⍒ C,A

where S is a new APL primitive, A is a scalar or vector, or a variable with scalar or vector value, and C is an integer specifying the function as follows:

C = 1 terminal type 1 = ARDS, 2 = Tek 4002, 3 = Tek 4010

C = 2
- lower left window, x
- lower left window, y,
- upper right window, x
- upper right window, y

ρA = 4 for C = 2

C = 3
- scale, min x
- scale, max y
- scale, min y,
- scale, max y
- scale, min z
- scale, max z

ρA = 4 or 6

z arguments are optional

C = 4 setpoint - x + y values. ρA = 2

C = 5 Draw axes

C = 6 Controls scaling. If A = 1, use maximum scaling.
If A = 2, use centered coordinates with maximum scaling.
If A = 3, return to previously set window.

C = 7 Dash next curve.

C = 8 Erase screen

Higher values of C presently give a Value Error; some of them may be used for future extensions. For terminals without APL characters, $QS can be used.

Examples:

1. Changing the window

⍒ 2 2 2 4 4

2. Scaling for coordinates, 2D plot

⍒ 3 3.4 7.1 4.1 6.3

⍒ is a legal character, but it is expected that it would not normally be employed directly.

7. The system is implemented in APL under the Universal Timesharing System for the Xerox Sigma Implementation details are available in a separate document.