S.N. Karon

An Interactive Plotting and Data-Analysis Package for the IBM PC

The Personal Computer Resource Center at Lincoln Laboratory has developed a plotting program called EasyPlot to display and analyze technical data on IBM Personal Computers. The program offers researchers a fast, interactive, graphical user interface that creates an intuitive environment for quickly viewing, analyzing, and plotting data. This article describes the program and some interesting design techniques that enhance program speed.

In February 1987, the Personal Computer Resource Center at Lincoln Laboratory began the project of writing an easy-to-use, scientific plotting package to run on IBM Personal Computers. One year later, the program, called EasyPlot, made its debut. Since then, EasyPlot has generated a large base of users and, with their feedback, has grown into a powerful plotting and data-analysis package. This article describes the four software layers that make up the program, and discusses the programming techniques that create a fast, interactive, graphics-based user interface. It also describes some of the unique plotting features made possible by the fast interactive graphics.

EasyPlot is built from four custom-written software layers. The top layer is the plotting package. It scales and labels axes automatically, plots multiple curves on a graph, stacks graphs in memory, and lets the user interact with the data and the graph. The lower three layers form the graphics system. They draw objects on the screen, manage windows, and provide the pull-down-menu interface. The bottom layer—the screen-drawing routines—consists of over 3,500 lines of assembly code. The other three layers consist of nearly 30,000 lines of C code.

The Plotting Package

EasyPlot reads ASCII and spreadsheet files in any row-and-column format. It also generates curves and surfaces from mathematical equations entered interactively. EasyPlot can display data in x-y plots (linear, semilog, and log-log), polar plots (full circle, hemisphere, and quadrant), contour maps, color maps, and 3-D surfaces. Figure 1 shows three plots of the same data—a 3-D surface, a contour map, and a color map—in the screen-display interface. Once data is on the screen, EasyPlot offers a number of analysis tools that, at the click of a button, let the user view and interact with the data. The user can smooth data; calculate best-fit polynomials and splines; compute means, standard deviations, and histograms; perform numerical integrations; run FFTs (including Hamming and Kaiser windows); and apply mathematical transformations to data. The mathematical-transformation feature includes the capability to perform math on multiple curves, such as adding two or more curves together or dividing one curve by another. Figure 2 shows an example of curve generation and transformation. For printed output, EasyPlot produces journal-quality graphs on several devices, including Postscript printers. Figure 3 shows four samples of hard-copy output.

As the program has grown, each new feature has been carefully integrated into the program to maintain consistency with other features and to preserve the program's ease of use. Computing a complex FFT, for example, is as easy as setting the range on an axis. Features have been designed to work with all the graph formats whenever possible. The equation parser, for example, handles polar and surface equations as well as x-y equations. To compute an x-y

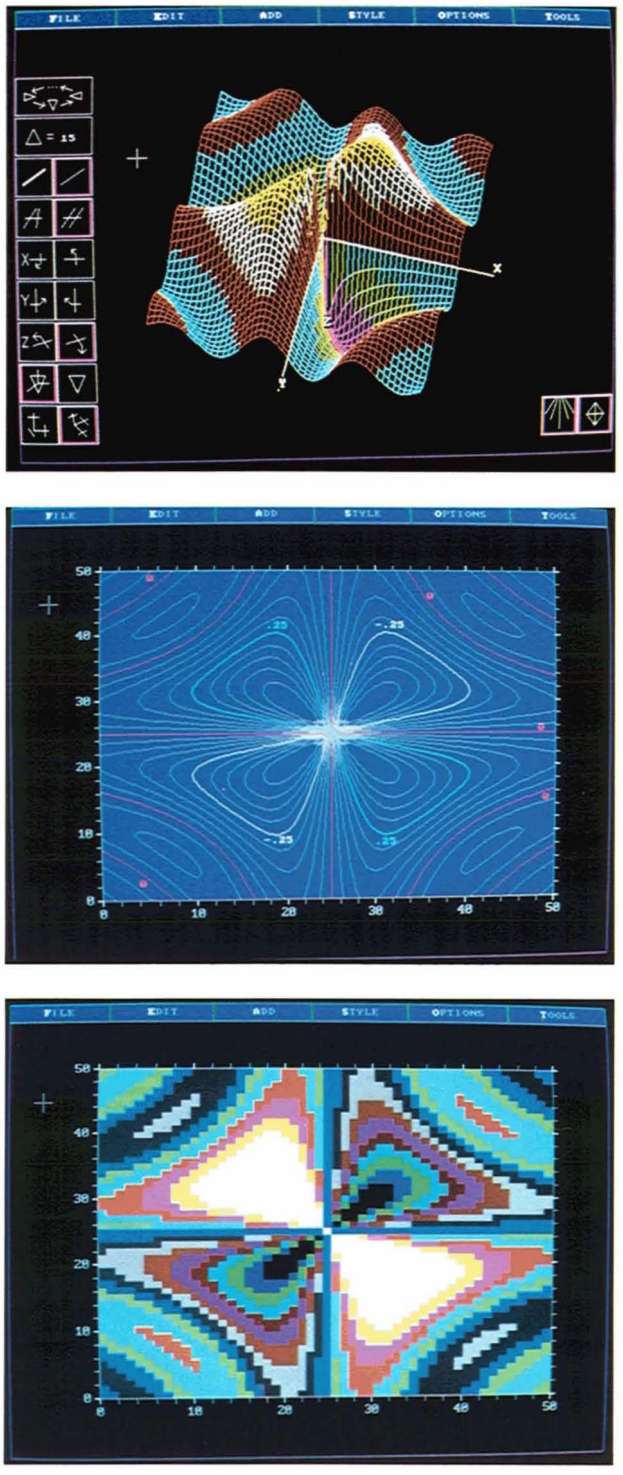


Fig. 1—(top) By clicking the 3-D buttons on the side of the screen, the user can rotate a 3-D surface to any orientation, turn perspective on or off, and remove hidden lines. The program can also animate the surface by rotating it automatically. (center) Contour plot of the surface data shown in top figure. (bottom) Color map of the surface data shown in top figure.

curve, the user enters a function of x or y at the keyboard:

```
y = 2x^3 - 3x.
```

To compute a polar curve, the user enters a function of r and t (where t represents θ) instead of x and y:

```
r = cos(3t).
```

To compute a surface, the user enters a function of x and y:

```
z = 3sin(x) + 5cos(y).
```

The program automatically distinguishes between equations and files, so the user can focus more on what to plot and less on how to plot it.

EasyPlot's fast, interactive interface makes a number of unique and powerful features possible. For viewing data, a zoom feature lets the user draw a rectangle anywhere on the graph and zoom on that rectangle (Fig. 4). The user can zoom repeatedly and magnify to any level. After zooming, a scroll-bar feature lets the user quickly scroll the graph from one portion of data to another (Fig. 5). A digitizing feature displays the graph coordinates of the cursor at all times (Fig. 5). For labeling, an annotation feature lets the user type annotations anywhere on the graph. If an annotation pertains to a specific data point or curve, the user can draw lines directly on the graph to point from the note to what it describes (Fig. 2[b]). The user can drag annotations, lines, and legends around the screen to fine-tune their positions. These features combine to create an intuitive, hands-on method for working with data and graphs.

The plotting package, or top layer of EasyPlot, has grown significantly since the first version became available at Lincoln Laboratory. User requests guided the growth of the program, and as a result every feature serves a need in science and engineering.

Drawing Objects on the Screen

With a graphics-based interface, the computer must draw objects on the screen extremely fast. Sluggish graphics erode the usefulness of an interactive program. To overcome this diffi-

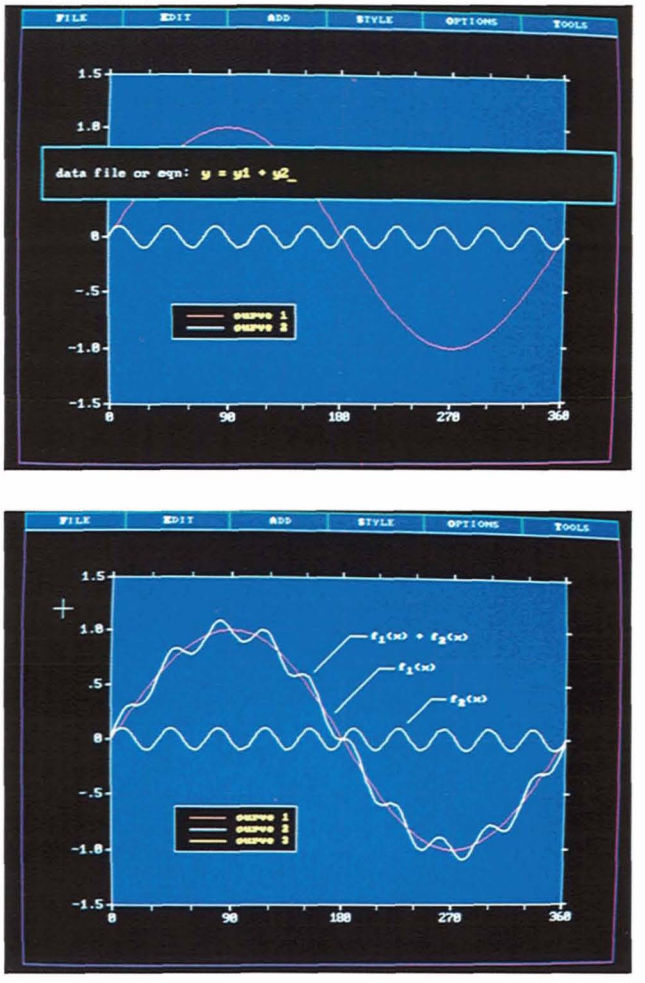


Fig. 2—EasyPlot can generate curves from equations and can perform mathematical transformations on curves. (top) Curves 1 and 2 were generated with equations y = sin(x) and y = 0.1 sin(10x). (bottom) Curve 3 was created by adding curve 1 and curve 2 in the top figure with the transformation equation y = y1 + y2.

culty, EasyPlot employs a set of optimized routines for drawing lines, text, and other objects on the screen. These assembly-code routines provide some of the fastest graphics attainable on the PC, and function as the foundation, or bottom layer, of EasyPlot.

The line-drawing routine is an implementation of Bresenham's algorithm [1]. The algorithm provides a computationally simple method for drawing a straight line on a raster device. An inner loop iterates once for each pixel along a line; since there are many iterations, the overall performance of the algorithm depends on the efficiency of the inner loop. To minimize the number of instructions in the loop, the processor's internal registers are utilized to hold as many variables as possible and to limit data movement that does not contribute to the computations. Careful design eliminated many instructions that only move variables between registers and temporary storage in memory. As a result the loop was reduced to 23 assembly-code instructions.

Seven of the 23 instructions use variables that remain constant for each line. Without performance considerations, the constants would be computed, stored in memory, and accessed from inside the loop. Accessing memory, however, is three to four times slower than accessing a register or having the constant as part of the instruction. Placing the constants in registers was impossible because of the PC processor's limited number of registers. Instead, instructions using constants were assembled with dummy immediate data (immediate data becomes part of the actual instruction), and code was added to write the constants over the dummy data before entering the loop. With this locally self-modifying code, the number of clock cycles for each pass through the loop was reduced from 195 to 118, which resulted in a 65% increase in speed.

Managing the Screen

Though sophisticated in their implementation, the screen-drawing routines do nothing more than turn pixels on and off. Their efforts last only until other objects are drawn in the same video space. When the user pulls down and releases a menu, everything underneath the menu must be redrawn. Records must be kept on every object drawn so that they can be redrawn when necessary. In EasyPlot, all the line segment and characters are objects, as are circles, rectangles, and their filled counterparts. Other shapes are constructed from these objects. A triangle, for example, is constructed from (and recorded as) three line segments. A windowing package that maintains these records is the second software layer in EasyPlot.

The windowing package allows graphic ob-

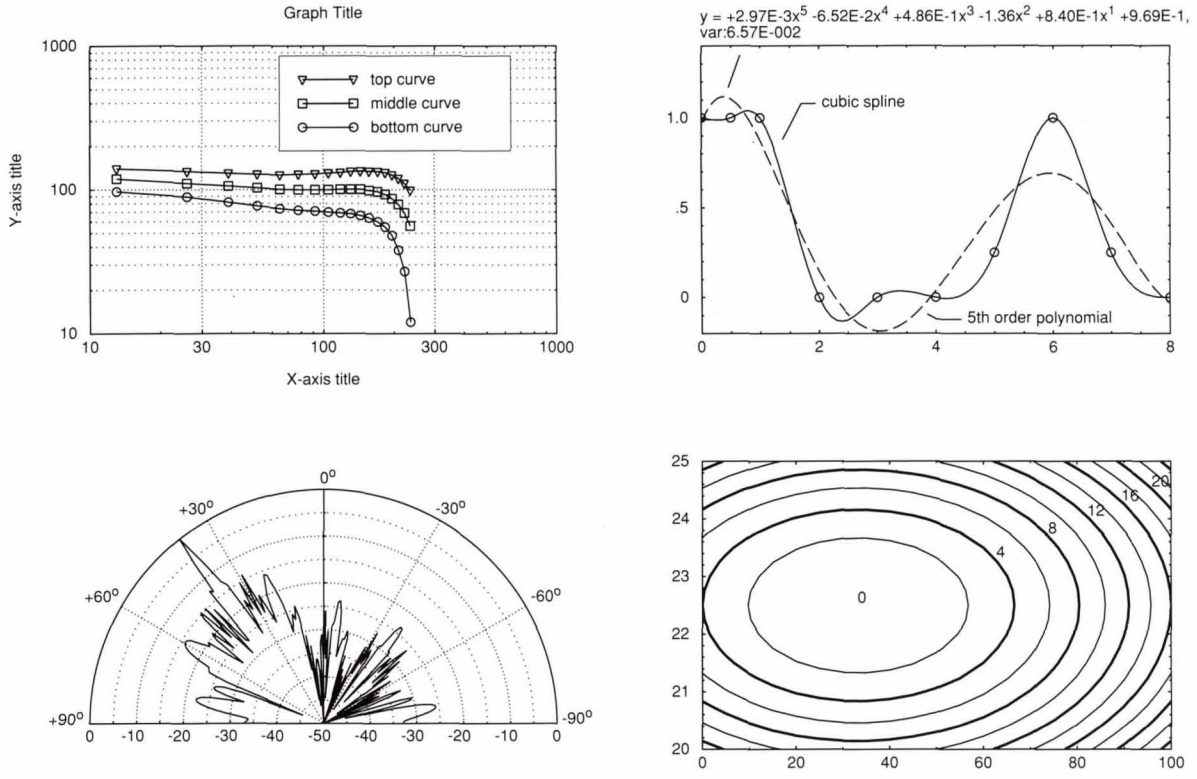


Fig. 3—Examples of EasyPlot printouts. The user can select any size within an 8 1/2" by 11" page for printed graphs.

jects to be grouped into logical units, or windows. The application programs (in this case the plotting package and pull-down menu layers of EasyPlot) can place logically connected objects such as the characters in a word into one window. Once grouped in a window, the objects can be drawn, hidden, or moved as a unit. Windows can be large or small and can overlap. A window obscures other windows only if the application package draws area-filling objects in it.

The windowing package maintains lists of objects that have been drawn in each window. With these lists, it can display and hide windows upon request. It displays a window by redrawing the objects belonging to that window, and hides a window by redrawing all the other windows that overlap the one being hidden. The pull-down menu system, for example, draws each menu button in a separate window. When the user pulls down a menu, the windowing package redraws the buttons' windows and the menu becomes visible. When the user releases the menu, the windowing package redraws all the windows overlapping the menu and the menu disappears.

If the screen-drawing routines are fast enough, the redraws appear to be instantaneous. Even with fast drawing routines, however, redrawing windows with large numbers of objects can result in visible delays. To maintain performance under these conditions, EasyPlot's windowing package looks at the coordinates of each object and redraws only those objects which actually overlap the area being redrawn.

With many objects on the screen, a simple search through object lists to look at each item's location produces noticeable delays. To alleviate this problem, EasyPlot's windowing package breaks a window into 16 segments. Seventeen object lists are maintained for the window: one for each of the 16 segments and one for a default

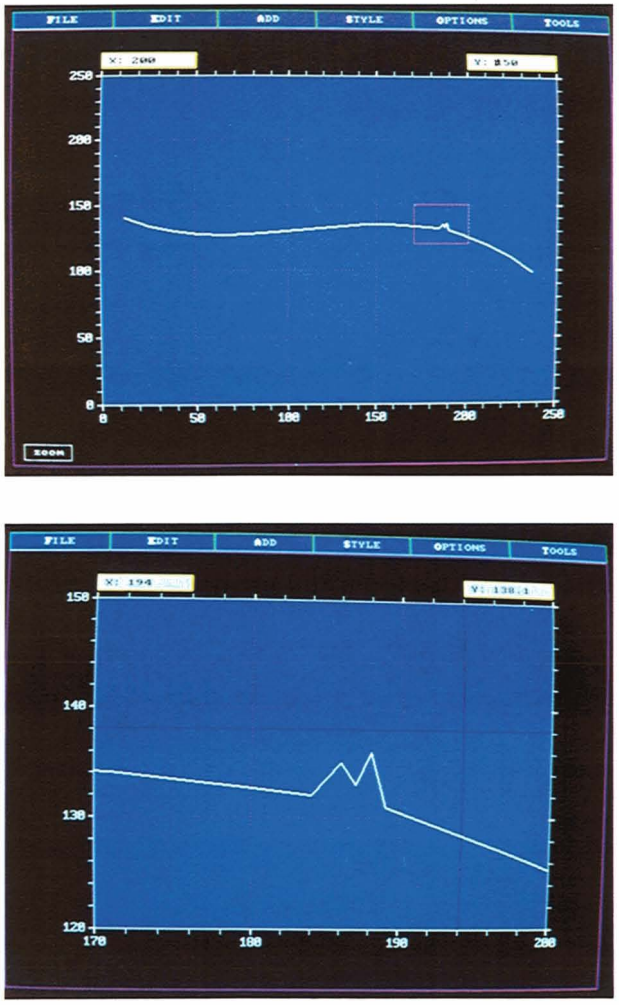


Fig. 4—(top) To zoom on data, the user draws a rectangle on the screen. (bottom) The data in the selected rectangle now fills the screen.

list. If an object lies entirely within a segment, it is recorded in the list for that segment. If it spans more than one segment, it is placed in the default list. Thus, only the default list and lists of those segments which overlap the area being redrawn need to be searched for objects to be drawn. EasyPlot divides windows containing user data into segments because large data sets can result in several thousand graphic objects in one window. Since most of the redrawing results from pulling down menus, and each menu overlaps a small portion of the graph, this technique can easily reduce redraw time by a factor of three or four.

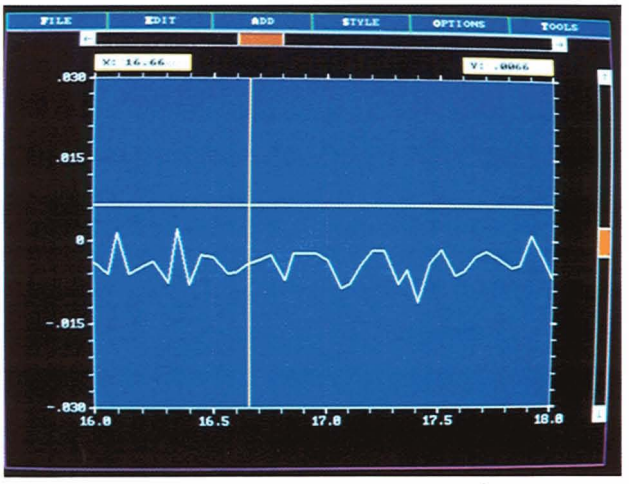


Fig. 5—Vertical and horizontal scroll bars let the user scroll quickly from one portion of data to another. A cross-hair feature displays the cursor's coordinates.

The Pull-Down Menus

The third software layer in EasyPlot is the pull-down-menu system. Pull-down menus provide quick access to all of the program's features. When the user selects one of the buttons at the top of the screen, a menu appears or is pulled down (Fig. 6). Each button in the menu is labeled to describe its function; the user never

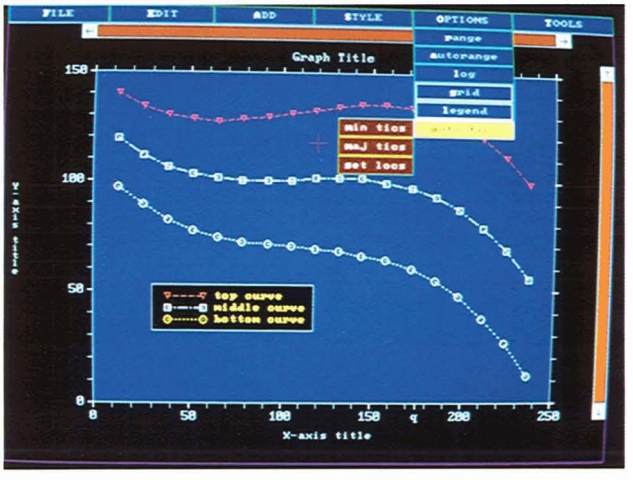
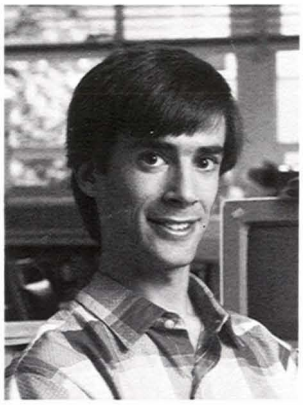


Fig. 6—The pull-down menus are always available at the top of the screen. Each menu button is labeled to describe its function.

needs to memorize commands or wade through pages of text menus. The user selects a menu with a mouse, with keyboard cursor keys, or by typing the first letter of the menu label. When the user runs a button, results appear immediately on the graph.

A unique letter sequence can be typed at the keyboard to select each button. EasyPlot's menu system can read the same letter sequences from a file to invoke menu buttons in batch mode. By providing EasyPlot a file containing batch commands, the user can automatically plot and print any sequence of graphs.



STUART KARON was an assistant staff member in the Computer and Telecommunications Systems Group. He received a B.S. in electrical engineering and an M.S. in computer science from Brown University. MIT recently licensed EasyPlot to Spiral Software of Brookline, Mass., to market and continue developing the program.

Summary

With a unique combination of power, flexibility, and ease of use, EasyPlot has become a tool used daily by researchers at Lincoln Laboratory. It accepts a wide range of input formats, and prints high-quality graphs with a variety of devices. EasyPlot is more than a convenient tool for making graphs, however. It creates an intuitive, interactive environment for viewing, analyzing, and plotting data.

Reference

1. J.D. Foley and A. Van Dam, *Fundamentals of Interactive Computer Graphics* (Addison-Wesley, Reading, MA, 1984), p. 433.