[4.4] Plot data and functions simultaneously

I frequently need to plot both data and a function on the same graph. This is useful when curve fitting and testing interpolation methods. To do this within a program is not difficult, but the method is not immediately obvious. The program below, *plotdemo(),* shows one way to do it.

```
plotdemo(xyspec,xyn,fplot,fpn)
prgm

©Demo program to plot xy-data & 1 function
©24dec99 dburkett@infinet.com
©xyspec: data points matrix or name
©xyn: xy data plot number
©fplot: function to plot; expression
©fpn: function plot number

local umode,gmode

©Save user's modes
getmode("ALL")→umode
stogdb gmode

©Set graph modes as needed
setmode({"Graph","Function","Split Screen","Full"})

©Clear previous graph contents
clrgraph
clrdraw
fnoff
plotsoff

©Get number of data points
rowdim(xyspec)→xyrows

©Convert xy data points to lists
mat▸list(submat(xyspec,1,1,xyrows,1))→xlist
mat▸list(submat(xyspec,1,2,xyrows,2))→ylist

©Plot the data and zoom to fit
newplot xyn,1,xlist,ylist
zoomdata

©Plot the function & trace
expr(string(fplot)&"→y"&string(exact(fpn))&"(x)")
trace

©Restore user's modes
delvar xlist,ylist
setmode(umode)
rclgdb gmode
disphome

endprgm
```

In this program, *xyspec* is the matrix (or name of the matrix) of the data points to plot. *fplot* is the function to plot, which must have an argument of *x*. *xyn* and *fpn* specify which plot number to use for the data and function plots, respectively. For example, this call:

```
plotdemo(xydata,1,sin(x),2)
```

plots the points in *xydata* as plot number 1, and the the function sin(x) as graph function y2(x).

Beyond all the house-keeping (saving the user's modes, and so on), the critical features of the program are these:

- Convert the matrix data to lists, to use the *newplot()* function. Note that these lists must be global variables to use *newplot(),* so the variables are deleted before the program exits.
- Use *zoomdata after* plotting the data, but *before* plotting the function, to scale the graph to the data.
- Use *expr()* on a string that assigns the input function to a y-function for plotting.
- Plot the function *after* plotting the data.
- Display the graph screen using *trace* so that the data points and function can be traced.

Control returns to the program after the user is done viewing the graph. Both the data points and the function can be traced, as usual, with the [UP], [DOWN], [LEFT] and [RIGHT] cursor movement keys. The user must press [ENTER] or [ESC] to continue running the program. You might consider displaying a dialog box with this information before showing the graph.

Since *newplot()* can only use global variables, *xlist* and *ylist* are deleted before the program exits.

The function is plotted by building a string, for example

```
"sin(x)→y1(x)"
```

then this string is evaluated with *expr().* The number of the y-function is passed as the *fpn* parameter, so if *fpn* = 3, then the string is actually

```
"sin(x)→y3(x)"
```

The function

```
string(exact(fpn))
```

is used to convert *fpn* to a string which is a simple integer with no decimal points or exponential notation, regardless of the current display mode settings.

Unfortunately, this program leaves the plot definition behind for *xlist* and *ylist*. Since the variables are deleted, the plot will not display, but an error will occur the next time the graph window is shown, unless the plot definition is manually deleted. There is no known method to delete the plot definition within the program.

(Credit to Olivier Miclo)