Producing Graphics in S-PLUS

Aidan Palmer

May 26, 1997

1 Introduction

This handout explains how to create and embellish simple graphics in S-PLUS. The reader is assumed to have a working knowledge of S-PLUS, including some understanding of how to use data frames.

2 The Education Dataset

To follow along with the handout, copy the file `/afs/andrew/stat/data/S-Tutorials/education.dat` into your home directory, then start S-PLUS.

This file contains data about school expenditures for each of the 50 states and the District of Columbia. The variables are school expenditures in 1970 (`SE70`), each state's citizens' average income in 1968 (`PI68`), school age population per capita in 1969 (`Y69`), urban population per capita in 1970 (`Urban70`), and two variables for the state's location: `Region` (general) and `Locale` (specific). We are interested in exploring the relationships between `SE70` and the other variables.

Read this data set into a data frame and attach to it (data frames are explained in detail in another handout).

```
> edu <- read.table("education.dat", header=T)
> attach(education)
```

3 Opening a Graphics Window

Create a graphic output window by using the function `motif()` (on a UNIX machine) or `win.start()` (on Windows). When you enter plot commands in the normal S-PLUS window the output will appear in this new window.

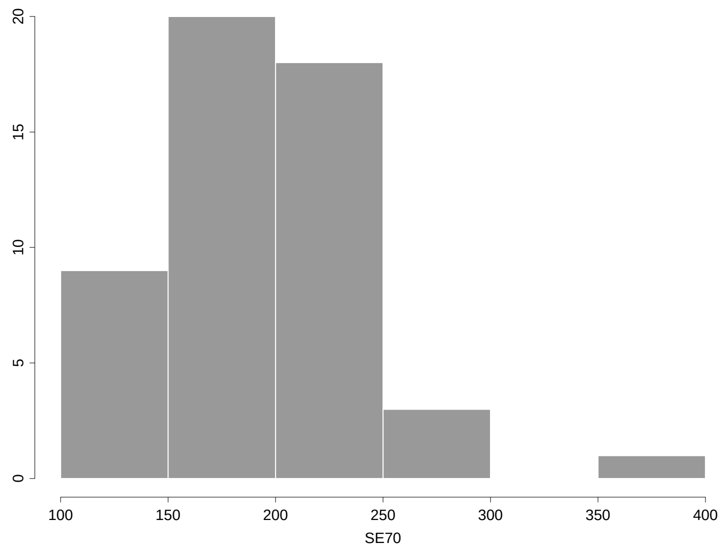
The graphics window is just like any other window: it can be moved, re-sized, etc. The window contains menus with several options. The ones you will likely be most interested in are the Color Scheme option (allows you to adjust the color scheme) and the Print option. The Print option will print the contents of the window to whatever printer you are connected to or have selected (on UNIX, Print will also save the contents as a postscript file in your working directory, with a name like `ps.out.0001.ps`).

4 Some Common Plotting Functions

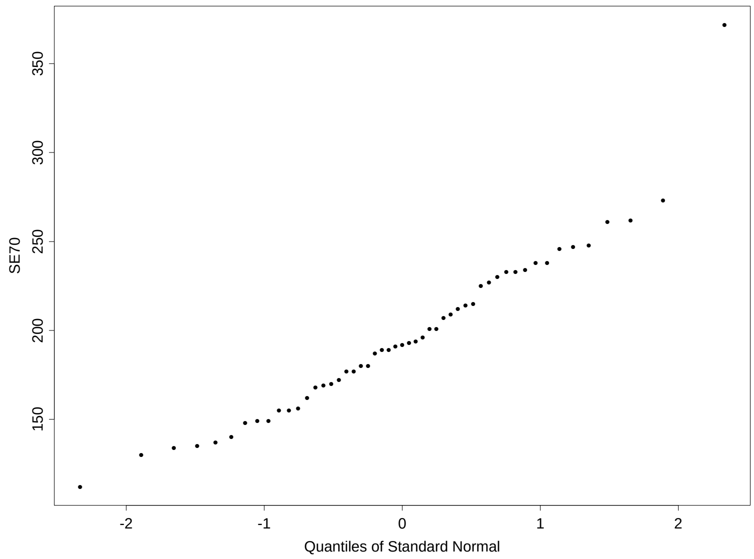
Functions that create a new picture are called "high-level" in S-PLUS terminology. Some useful high-level functions are `hist` (produces a histogram), `qqnorm` (produces a normal quantile plot) and `plot` (described in more detail below).

Let's begin our analysis of the education data by examining the distribution of the response variable, `SE70`.

```
> hist(SE70)
```

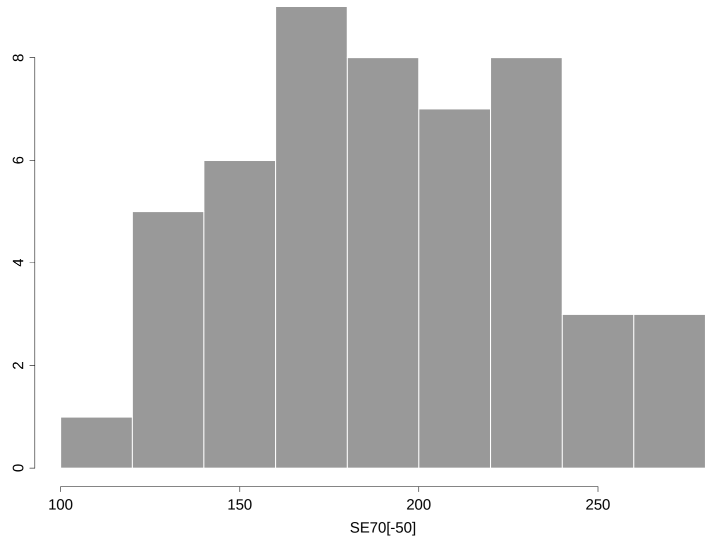



```
> qqnorm(SE70)
```



From these two plots it appears that school expenditures may be close to normally distributed, except for one outlier which had school expenditures over 350. If you look through the data, you will see that the outlier is Alaska. Let's see the histogram again, without Alaska. Alaska is listed 50th in the file, so we want to plot `SE70[-50]` (school expenditures of every state but the 50th).

```
> hist(SE70[-50])
```

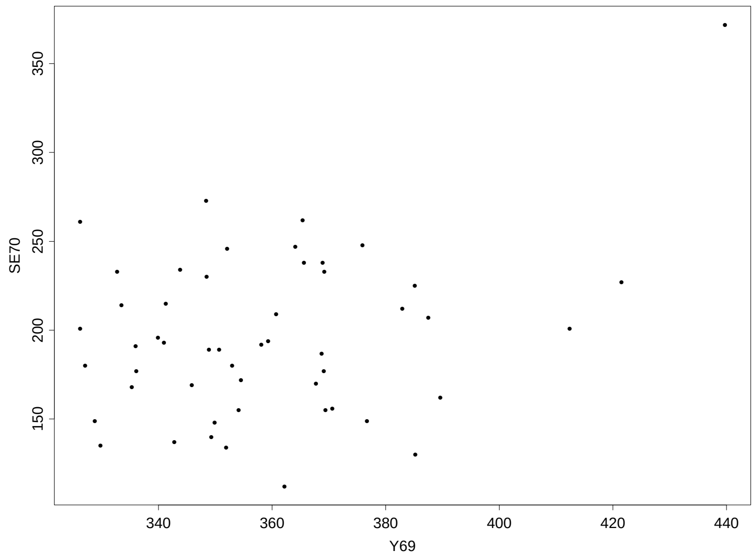


Judging by this histogram, SE70 does not have a particularly normal distribution after all. Of course, you should check this by doing another normal quantile plot.

The next step is to examine the relationship between SE70 and the other variables. The function `plot` will come in handy here. This function can be called in several different ways, and since S-PLUS uses Object Oriented Programming, `plot` will produce a different sort of plot depending on the class of object it receives. The most common use is `plot(x,y)`, with `x` a numeric vector, which results in a scatterplot of `x` against `y`. If `x` is a factor, this will instead produce boxplots.

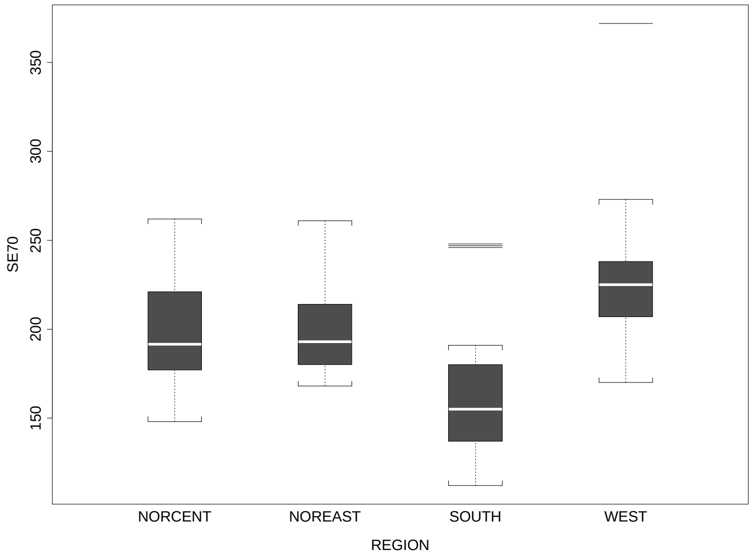
We can expect that school expenditures will be related to the number of students, so let's plot Y69 (students) against SE70.

```
> plot(Y69, SE70)
```



It does not appear from this scatterplot that there is any particular relationship between number of students and school expenditures (note that Alaska is the point in the upper right-hand corner).

We can also plot the `Region` vector (a factor) against school expenditures.



This shows that school expenditures are generally lower in the South.

The function `plot` can also be called with a single argument, and the output will be a plot of an index against that vector (if the argument is a vector) or a series of diagnostic plots (if the argument is a model, see the handout on models for an example).

> **Question:** How do you do the above boxplots, excluding Alaska? How do you plot only the North East and North Central data?

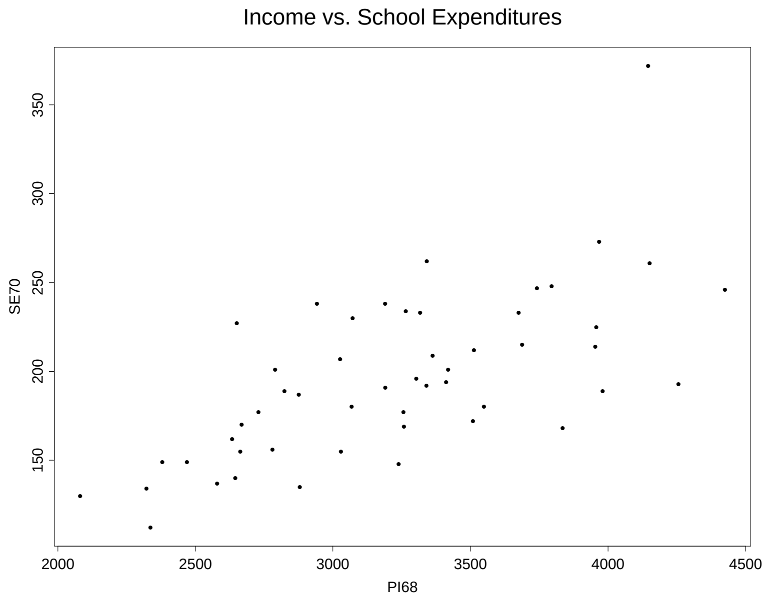
5 Embellishing Plots

Once you have created a plot, you can embellish it in various ways. S-PLUS documentation refers to function of this sort as "low-level" graphics functions.

5.1 Adding Titles

S-PLUS allows you to add titles at several places in the plot: a main title (above the picture) and x-axis and y-axis titles.

```
> plot(PI68, SE70)
> title(main="Income vs. School Expenditures")
```



You can add x-axis (`xlab=`) and y-axis (`ylab=`) titles in a similar way. Note that `plot` usually produces x-axis and y-axis labels, so there is rarely a need to add your own. In fact, if you add a new title to a plot where one already exists, S-PLUS will put the new title on the plot *without removing the existing one.* This means that if you made a mistake with one title and wish to change it, will need to create the plot all over again and then add the correct title.

5.2 Adding a Line

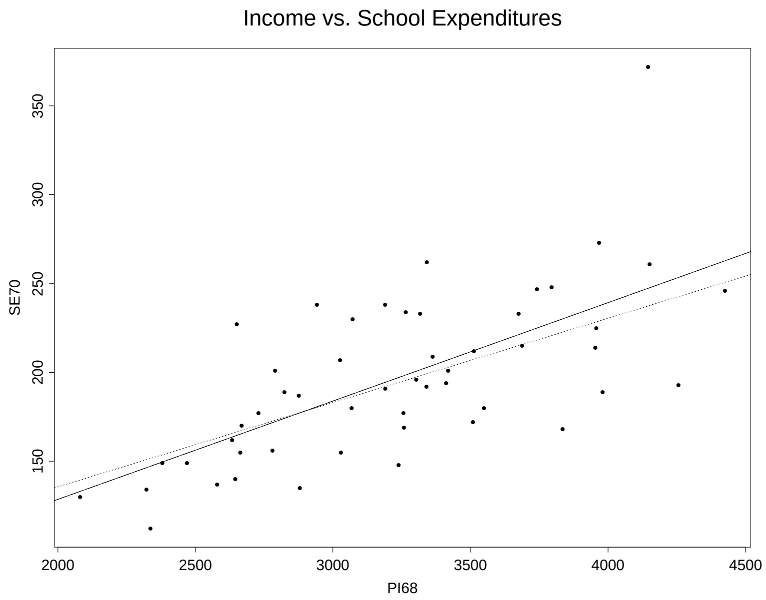
The function `abline` adds a line of the form `y = a + b*x` to a plot. It also has an `lty=` field to allow different line types (dashed, dotted, etc.).

Let's add two lines to the plot of income and school expenditures, one including Alaska and the other not. We can find these lines using regression (explained in the handout on models). The first command below says "`SE70` modelled by `PI68`" and the second says "`SE70` modelled by `PI68` using all points except the 50th".

```
> lm(SE70 ~ PI68)
Call:
lm(formula = SE70 ~ PI68)

Coefficients:
 (Intercept)       PI68
    17.71003 0.05537594

Degrees of freedom: 51 total; 49 residual
Residual standard error: 34.9384
> lm(SE70 ~ PI68, subset=-50)
Call:
lm(formula = SE70 ~ PI68, subset = -50)

Coefficients:
 (Intercept)       PI68
    40.56264 0.04747211

Degrees of freedom: 50 total; 48 residual
Residual standard error: 29.93981
```

Now we can use the first equation to plot a solid line (line type 1, the default), and the second to plot a dotted line (line type 2).

```
> abline(17.71003, 0.05537594, lty=1)
> abline(40.56264, 0.04747211, lty=2)
```



As you can see, the slope is smaller when Alaska is removed from the model, but the two lines are very similar.

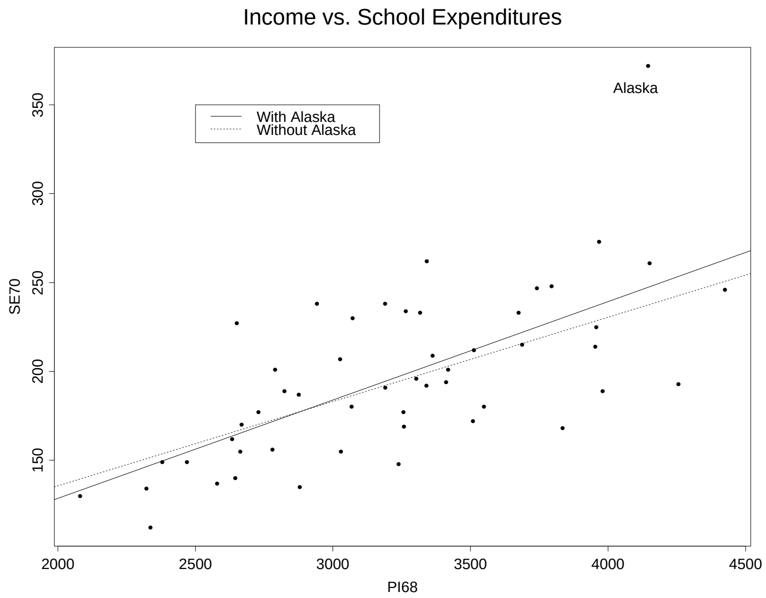
5.3 Adding Text

S-PLUS allows you to add text to a graph in a few different ways. The simplest is with the function `text`. The format is `text(x,y,"string")`, and this plots the text `string` at the coordinates (x,y).

S-PLUS also has a somewhat more elegant function, `legend`. This function takes x and y coordinates, a vector of labels, and a vector of colors or line types corresponding to those labels.

For instance, we could add a legend to the graph which labels the two line types, and `text` to write "Alaska" approximately underneath the Alaska data point.

```
> legend(2500, 350, c("With Alaska","Without Alaska"), lty=c(1,2))
> text(4100, 360, "Alaska")
```



If you try this on your computer, it may appear that the legend box is not large enough to hold the words. This happened to me, but it turned out fine on paper.

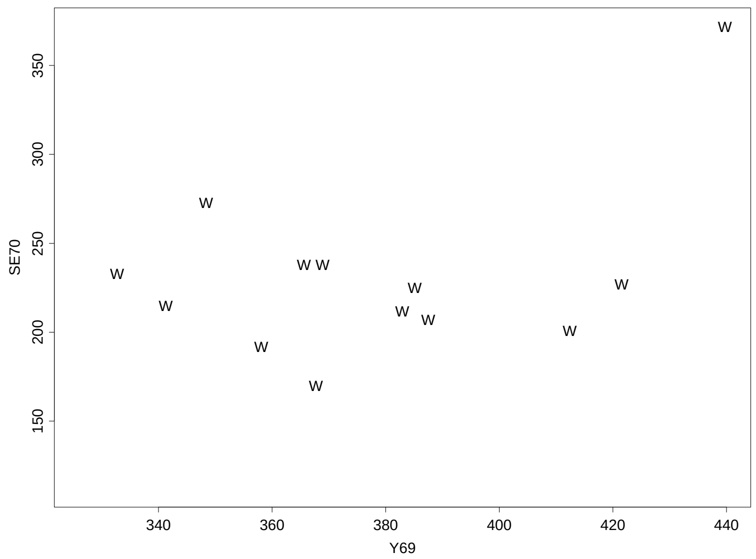
6 Plotting Points

Another way to create or embellish a plot is by adding points. Suppose we were interested in returning to the students vs. expenditures plot (where there did not appear to be any clear relationship) and breaking the data down by region. To plot the points from the West (and label them with `W`'s instead of `*`'s) we could use:

```
> plot(Y69[Region=="WEST"], SE70[Region=="WEST"], pch="W")
```

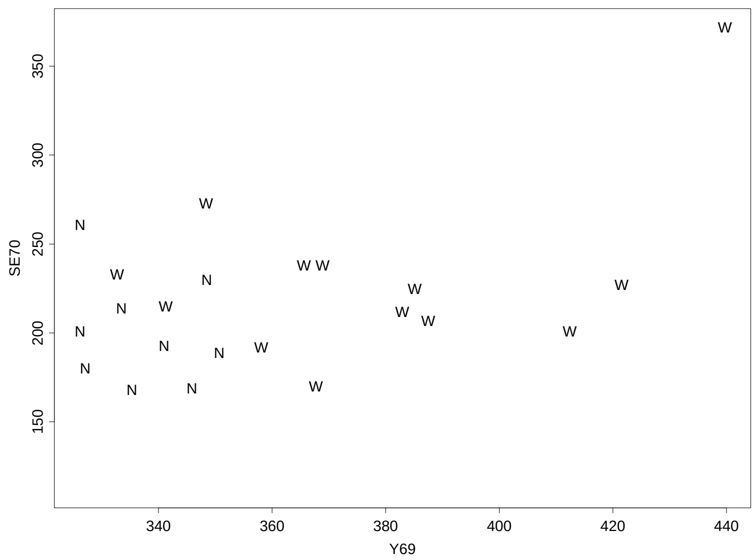
(The `pch=` field tells S-PLUS what character to use for the points.) This plot is OK for displaying the West points, but if we tried to add points from any other region, some might be outside the range of this graph. This happens because `plot` chose a range for the axes to fit the West points as well as possible, without any consideration for the other points. It is better to first create an empty plot which fits the full data, and then fill in the West points.

```
> plot(Y69, SE70, type="n")
> points(Y69[Region=="WEST"], SE70[Region=="WEST"], pch="W")
```



Now points from any other section can be added to this graph and they will fit. For instance, the Northeast:

```
> points(Y69[Region=="NOREAST"], SE70[Region=="NOREAST"], pch="N")
```



This shows that the Northeastern states had fewer students per capita, but about the same range of expenditures as the non-Alaska Western states.

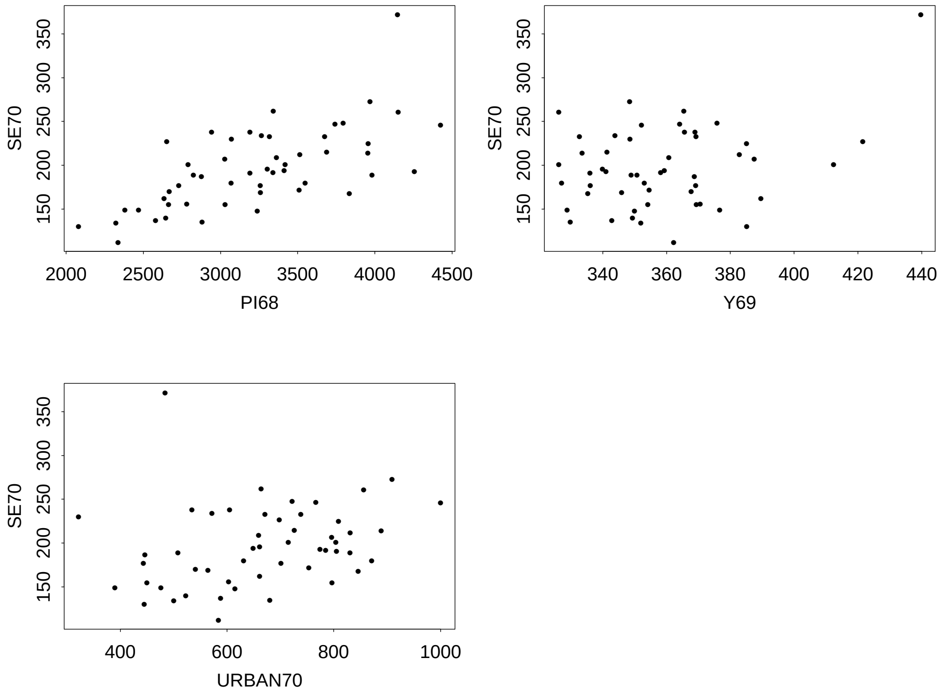
7 Multiple Plots per Page

To see multiple plots on the same page, first break the graphics window into multiple sections. Let's try a 2 by 2 grid of smaller windows.

```
> par(mfrow=c(2,2))
```

This has no obvious immediate effect. However, the next plot will appear in the upper left hand frame of a 2 by 2 matrix of plotting frames. Let's fill these frames with plots of predictors and SE70, two of which we have already seen.

```
> plot(PI68, SE70)
> plot(Y69, SE70)
> plot(Urban70, SE70)
```



The most recent plot (in this case `Urban70` vs. `SE70`) is the active one with respect to embellishments. Titles, lines, etc. will appear on that plot. Because of this you should finish working with one plot before creating any others.

The next plot will appear in the lower right-hand frame. If you then created a fifth plot, the first four would disappear and the fifth would go in the upper left-hand frame. Suppose you have these three graphs, and want to add a set of four more. If you simply tried to add four plots, the first would go on the current page and then that page would disappear and the last three plots would be displayed. If you want the four new plots to go together, you have several options. The first is to execute the `par(mfrow=c(2,2))` command again. This would take effect with the first of the four plots, causing it to appear in the upper left-hand frame of an otherwise blank window. A second option is to use the `frame()` command twice, to move the working frame two frames forward. When you type `frame()` the first time, S-PLUS will make the lower right-hand frame the active one, and you will be no longer be able to embellish the third picture. When you use the function a second time, S-PLUS will advance the active frame to the upper left-hand frame of a blank page and you will then have room for four plots on that page.

8 Using Color

Most plotting functions have a `col=` field, allowing you to select a color or colors for a plot. This field takes a numeric argument and selects the matching color from the active color scheme. For lines in the "white on black" (default) color scheme, 0 is black and 1 is white. You can edit the color scheme (via the menu in the graphics window) and add some more colors. For instance, if you wrote "red" and "yellow" after "white" on the list of colors for lines, `abline(a,b,col=2)` would produce a red line and `abline(a,b,col=3)` would produce a yellow line.