Plots in R

There are three basic plotting functions in R: high-level plots, low-level plots, and the layout command `par`. Basically, a high-level plot function creates a complete plot and a low-level plot function adds to an existing plot, that is, one created by a high-level plot command.

High-Level Plot Functions

Some of the basic plot functions include:

Function	Description
`plot`	scatter/line plot
`hist`	histogram
`barplot`	barplot
`boxplot`	boxplot
`qqnorm`	normal-quantile

Download the example data set **States03** from `http://apps.carleton.edu/curricular/math/resources/rcomputing/`, then import it into your session.

If you are using `RStudio`, then at the menu, **Tools** > **Import Dataset** > **From Text File...** and navigate to the location of the file.

Or at the command line, use `read.csv` to import the data:

```
> States03 <- read.csv("States03.csv")
```

(exact path will vary depending on where you saved the file).

Basic single variable plots:

```
> barplot(table(States03$Region))
> hist(States03$Poverty)
```

To create a scatter plot, there are two approaches:

```
> plot(States03$Unemp, States03$Poverty, xlab = "Unemployment", ylab = "Poverty")
> plot(Poverty ~ Unemp, data = States03, xlab = "Unemployment", ylab = "Poverty")
```

In the first approach, provide the `plot` command with the x-variable, then the y-variable. In the second approach, if the data are contained in a data frame, then provide the names of the variable `Y` $\sim$ `X` along with the name of the data frame.

High-level functions may also take optional arguments that enhance the plot.

```
> hist(States03$Poverty, main = "Poverty", xlab = "percent",
       xlim = c(0, 24), ylim = c(0, 20))
> plot(1:19, 1:19, pch = 1:19, xlab = "x", ylab = "y")
> pie(rep(1, 8), col = 1:8)
```

Option	Description
pch	point character (pch=1, 2, ...)
lty	line type (lty=1, 2, ...)
lwd	line thickness (lwd= 1, 2,...)
col	color (col="red", "blue",...)
xlim	x-axis limits: xlim=c(min,max)
ylim	y-axis limits
xlab	x-axis label: xlab="my label"
ylab	y-axis label
main	main title
sub	sub title

To plot smooth curves, use the `curve` command. The first argument must be an expression in terms of x:

```
> curve(x^2, from = 0, to = 2)
> curve(cos(x), from = 0, to = pi)
> curve(cos(x), from = 0, to = pi, lty = 4, col = "red")
```

Low-level Plot Functions

Low-level plot functions can be executed only after a high-level plot has been created. For example,

```
> plot(Poverty ~ Unemp, data = States03, xlab = "Unemployment", ylab = "Poverty")
> abline(v = mean(States03$Unemp), lty = 2)  #vertical line at mean unemployment rate,
> text(30, 18, "mean unemployment rate")   #text at (30, 18)
> title("Data from 2003")
```

The `abline` function has several options:

`abline(3, 5)` adds the straight line $y = 3 + 5x$

`abline(v = 2)` adds the vertical line, $x = 2$

`abline(h = 0)` adds the horizontal line, $y = 0$

```
> plot(Poverty ~ ColGrad, data = States03, col = "blue", pch = 19, xlab = "College grad (%)",
   ylab = "Poverty (%)")
> points(Uninsured ~ ColGrad, data = States03, col = "red", pch = 19)
> mtext("Percent uninsured", side = 4)
> legend("bottomleft", legend = c("Y: Poverty","Y: Uninsured"), col = c("blue","red"),
  pch = c(16, 16))
```

You can also use different plotting symbols for different levels of a factor variable:

```
> range(States03$Poverty)
> range(States03$ColGrad)

> plot(Poverty ~ ColGrad, data = States03, pch=16, subse t =Region=="West",
     xlim = c(15,40), ylim = c(5, 20))
> points(Poverty ~ ColGrad, data = States03, pch=16, col = "red", subset = Region=="South")
> points(Poverty ~ ColGrad, data = States03, pch=16, col = "green", subset = Region=="Northeast")
> points(Poverty ~ ColGrad, data = States03,pch = 16, col = "blue", subset = Region=="Midwest")
> legend("topright", legend=c("West", "South", "Northeast", "Midwest"),
   pch = rep(16,4), col = c("black", "red", "green", "blue"))


> curve(cos(x), from = 0, to = 2*pi)
> curve(sin(x), add = TRUE, col = "blue", lty = 2)
```

Function	**Description**
lines	add a line plot
points	add points
text	add text
mtext	margin text
abline	add a straight line
qqline	add line to qqnorm
title	add a title

The par Command

The `par` command controls the layout of the graphics device. The option you will use most often will probably be `mfrow` (**m**ulti-**f**igure, by row), or `mfcol`. For example, to have a 3x2 layout where the plots are added by row, set

This setting will exist throughout the life of the graphics device unless you change it back to the default `mfrow=c(1,1)`.

You can also change the default color, plot character, etc. for the graphs created on the graphics device.

```
> par(mfrow = c(2, 2))   #2x2 layout
> curve(3*x^2)
> curve(cos(x))
> hist(States03$Population)
> qqnorm(States03$Population)
> qqline(States03$Population)
> par(mfrow = c(1, 1))   #reset to default layout
```

Misc.

- Type `colors()` at the command line to see the list of colors available to the plotting commands.
- You can export to some common file formats (**jpg, pdf, ps**). With the graph in focus, go to the menu, in Windows, **File** > **Save As...** and save to jpg, pdf, ps, png or bmp. On the Macintosh, **File** > **Save as** to pdf only.

  Or, at the command line, for instance

```
> postscript(file = "MyPlot.eps")    #open graphics device
> hist(States03$Births, main = "Number of births")  #create graph
> dev.off()                          #close graphics device
```

  The file MyPlot.eps will be located in your working directory.

  See the help file for `postscript`, `jpeg`, `png`, `tiff` or `pdf`.