**R Graphics**

Producing high-quality graphics is one of the fundamental parts of any statistical computing. Graphics are often the starting point for statistical analysis. The particular plot function you need will depend on the number of variables you want to plot and the pattern you wish to highlight. One of the most attractive aspects of the R system is its capacity to produce state-of-the –art statistical graphics.

**Plots with Two Variables**

With two variables (typically the *response variable* on the y axis and the *explanatory variable* on the x axis), the kind of plot should produce depends upon the nature of explanatory variable. When the explanatory variable is a continuous variable, such as length or weight or altitude, then the appropriate plot is a scatterplot. In cases where the explanatory variable is categorical, such as genotype or color or gender, then the appropriate plot is either a box-and-whisker plot (when you want to show the scatter in the raw data) or a barplot (when you want to emphasize the effect sizes).

The most frequently used plotting functions for two variables in R are the following:

* ***plot(x,y)***
* ***boxplot(y)***
* ***barplot(y)***
* ***pie(y)***
* ***hist(y)***
* ***stem(y)***
* ***ts.plot(y)***
* ***stripchart(y)***
* ***curve(expression)***

**Scatterplots**

The *plot* function draws axes and adds a scatterplot of points. Two extra functions, points and lines, add extra points or lines to an existing plot. There are two ways of specifying plot, points and lines and you should choose whichever you prefer: Customize graphs (line style, symbols, color, etc) can be drawn by specifying graphical parameters. To see the total sum of arguments that one can call using plot we check

|  |  |
| --- | --- |
| plot.default {graphics} | R Documentation |

The Default Scatterplot Function

Description

Draw a scatter plot with decorations such as axes and titles in the active graphics window.

**Usage**

plot(x, y = NULL, type = "p", xlim = NULL, ylim = NULL,

log = "", main = NULL, sub = NULL, xlab = NULL, ylab = NULL,

ann = par("ann"), axes = TRUE, frame.plot = axes,

panel.first = NULL, panel.last = NULL, asp = NA, ...)

**Example**

> plot(c(1,2,3), c(3,5,7))



Note that the points in the graph in this figure are denoted by empty circles. If we want to use a different character type, specify a value for the named argument ***pch*** (for point character).

R graphics are wonderful in that one begins with a blank slate. There are a good number of options that can be combined in a number of ways to create figures that suit specific needs. These include scatterplots, line graphs, bar graphs, histograms, boxplots, and more. It is perhaps most useful for getting started to work through the basic functions/options that bring considerable flexibility to creating figures in R.

*We can build up a graph in stages by issuing a series of commands*

**The Coordinate System**

We want to establish the dimensions of the figure before plotting anything especially if we are building up from the blank figure. The most important point but perhaps is obvious that both variable x and y must be of the same length.

**Plot Types**:

We now want to plot these series, but the plot function allows for different types of plots. The different types that one can include within the generic plot function include:

***type="p"*** : This is the default and it plots the x and y coordinates as points.

***type= ”l''*** : This plots the x and y coordinates as lines.

***type= "b"*** for both,

***type=``n''*** : This plots the x and y coordinates as nothing (it sets up the coordinate space only).

***type=``o''*** : This plots the x and y coordinates as points and lines overlaid (i.e., it overplots").

***type=``h''*** : This plots the x and y coordinates as histogram-like vertical lines.

***type=``s''*** : This plots the x and y coordinates as stair-step like lines.

**Axes:**

It is possible to turn off the axes, to adjust the coordinate space by using the ***xlim*** and ***ylim*** options, and to create the labels for the axes.

***axes=*** Allows us to control whether the axes appear in the figure or not. We may select ***axes=F***. Then create own labels.

***xlim=, ylim=*** allows us to select desired x and y limits

***xlab="", ylab=""*** Creates labels for the x- and y-axis

**Style:**

There are a number of options to adjust the style in the figure, including changes in the line type, line weight, color, point style, and more.

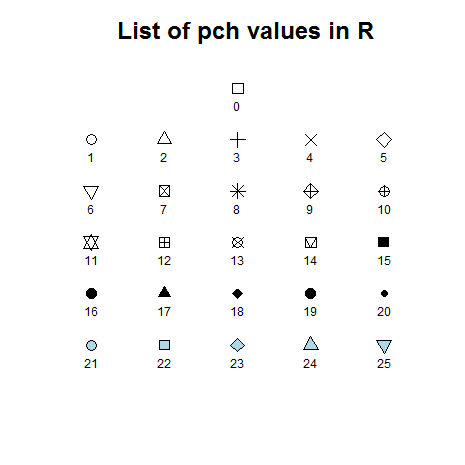
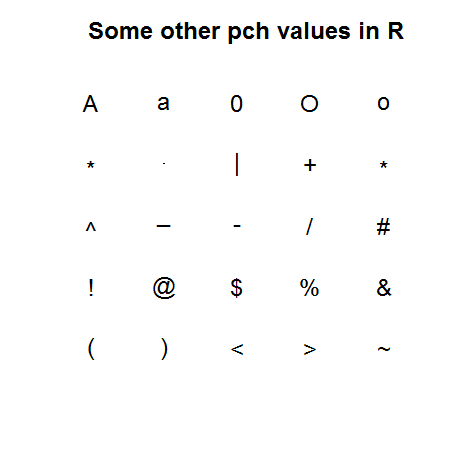
***lty=*** Selects the type of line (solid, dashed, short-long dash, etc.)

>plot(c(0,1), c(0,0), type="l", axes=FALSE, xlab=NA, ylab=NA, lty=1)

***lwd=*** Selects the line width (fat or skinny lines)

>plot(c(0,1), c(0,0), type="l", axes=FALSE, xlab=NA, ylab=NA, lwd=2)

***pch=*** Selects the plotting symbol, can either be a numbered symbol (pch=1) or a letter (pch="R")

 ***col=*** Selects the color of the lines/points in the figure

>colors()

> plot(c(1,2,3), c(3,5,7),col=4)

> plot(c(1,2,3), c(3,5,7),col="blue")

***bg=*** `background' color

***col=*** color of lines and data symbols

***col.axis=*** color of axis tick labels

***col.lab=*** color of axis labels

***col.main=*** color of plot title

***col.sub=*** color of plot sub-title

***fg=*** `foreground' color

> par(bg=4)

> plot(c(1,2,3), c(3,5,7),col=2)

**Graphic Parameters, *par()***

The function ***par()*** is used to set or get graphical parameters. This function contains many possible settings and allows us to adjust almost any feature of a graph. Graphic parameters are reset to the defaults with each new graphic device. To extract a graphic parameter, par("tag ") or par()$tag . similarly To set a graphic parameter, par(tag=value ) Most elements of par() can be set as additional arguments to a plot command, however there are some that can only be set by a call to par(), for example *mar, oma, mfrow, mfcol* etc.

par allows us to plot multiple (x, y)'s in a single graphic. This is accomplished by selecting par(new=T) following each call to plot.

**Add-on Functions**

There are also a number of add-on functions that one can use once the basic coordinate system has been created using plot. Some of these are listed below, and their implementation appears in the following sample code.

***arrows(x1, y1, x2, y2)*** Create arrows within the plot (useful for labeling particular data points, series, etc)

***text(x1, x2, "text")*** Create text within the plot

***lines(x, y)*** Create a plot that connects lines

***points(x, y)*** Create a plot of points

***polygon()*** Create a polygon of any shape (rectangles, triangles, etc.

***legend(x, y, at = c("", "",), labels=c("", "")))*** Create a legend to identify the components in the figure

***mtext( )*** Insert text in the figure and outer margins

***title( )*** Add figure title or outer title

***abline( )*** Add horizontal and vertical lines or a single line

***box( )*** Draw a box around the current plot

***rug()*** Add a 1-D plot of the data to the figure

***rect()*** Draw a rectangle

***segments( )*** Draw line segments

***trans3d()*** Add 2-D components to a 3-D plot

***main=*** an overall title for the plot

***sub=*** a sub title for the plot

**Text and Symbol Size**

The following options can be used to control text and symbol size in graphs.

|  |  |
| --- | --- |
| ***cex*** | Number indicating the amount by which plotting text and symbols should be scaled relative to the default.  cex=1 is default, cex=1.5 is 50% larger, cex=0.5 is 50% smaller, etc. |
| ***cex.axis*** | magnification of axis annotation relative to cex |
| ***cex.lab*** | magnification of x and y labels relative to cex |
| ***cex.main*** | magnification of titles relative to cex |

**Example:**

Link below provides the number of Atlantic hurricane from 1870 to 2010

<http://people.sc.fsu.edu/~jburkardt/datasets/time_series/hurricanes.txt>

We will import the subject data and plot a graph

>hurricane<-"http://people.sc.fsu.edu/~jburkardt/datasets/time\_series/hurricanes.txt"

>data=read.table(hurricane)

>x<-data$V1

>y<-data$V2



Now we will use some options to update the above graph

> plot(x,y, xlab="Year", ylab="Number of hurricanes", main="Number of Atlantic Hurricanes\n (1870 - 2010)",col=2)



**Example:**

The Duncan data frame has 45 rows and 4 columns. Data on the prestige and other characteristics of 45 U. S. occupations in 1950. The data is in the library “car” we will access the data as below

>library(car)

>data(Duncan)

>attach(Duncan)

> head(Duncan, n=5)

type income education prestige

accountant prof 62 86 82

pilot prof 72 76 83

architect prof 75 92 90

author prof 55 90 76

chemist prof 64 86 90

> plot(education)

> plot(prestige)



>plot(education, prestige)



**Outer Margin and Multiple Graphs**

>oma=c(1,1,1,1)

>par(mfrow=c(3,1))#creates 3 figures with 3 rows and 1 column

**Saving Graphs**

Since R runs on so many different operating systems, and supports so many different graphics formats, it's not surprising that there are a variety of ways of saving your plots, depending on what operating system you are using, what you plan to do with the graph, and whether you're connecting locally or remotely.

Format Driver Notes

JPG jpeg Can be used anywhere, but doesn't resize

PNG png Can be used anywhere, but doesn't resize

WMF win.metafile Windows only; best choice with Word; easily resizable

PDF pdf Best choice with pdflatex; easily resizable

Postscript postscript Best choice with latex and Open Office; easily resizable