R for visualization

R has a powerful environment for visualization of scientific data

- It provides publication quality graphics
- Easily reproducible
- Lots of packages and functions with built-in graphics support
- Postscript, PDF, jpeg, png, SVG

Graphics: History

- **Low-Level Capability**
  - Base Graphics (Has Low and High Level functions)
  - Grid Graphics

- **High-Level Capability**
  - Lattice Graphics
  - ggplot2

Graphics: History

Base Graphics

- Oldest and most commonly used

- Uses a "pen-on-paper" model. You can only draw on top of the object. Cannot erase, modify, or delete what has already been drawn.

- Base graphics are fast.

- Lots of documentation and "google" support

Graphics: History

Lattice package

- Developed by Deepayan Sarkar.

- Easy to create conditioned plots with automatic creation of axes, legends, and other annotations

- An improvement over Base graphics.

Graphics: History

ggplot2

- Developed in 2005 by Hadley Wickham

- ggplot2 is an implementation of Leland Wilkinson's *Grammar of Graphics*--a general scheme for data visualization which breaks up graph into semantic components such as scales and layers.

- ggplot2 can serve as a replacement for the base graphics in R

Graphics: Base

FUNCTION NAME	PURPOSE
points(x,y)	Adds points to an existing plot
lines(x,y)	Adds lines to an existing plot
arrows(x,y)	Draws arrows on an existing plot
text(x,y,labels,...)	Adds text to an existing plot
abline(a,b)	Adds a line of slope b and intercept a
polygon(x,y,...)	Draws a polygon
legend(x,y,legend)	Adds a legend to the plot
title("title")	Adds a title to the plot
axis	Adds an axis to the current plot
mtext	Write text in one of the four margins
segments	Draws line segments on an existing plot

Graphics: Base

FUNCTION NAME	PURPOSE
plot(x,y)	Generic x-y plots
barplot(x)	Creates a barplot of a table object
boxplot(x)	Creates a boxplot of numeric vector
hist(x)	Histogram of numeric data
pie(x)	Pie chart of a table object
dotchart(x)	Dot Plot of a vector or matrix
qqnorm(x)	Normal qqplot of numeric vector
qqline	Draws the qqline
pairs(x)	Scatterplot of matrix or data frame
stripchart	1D Scatterplot
coplot(x ~ y | f)	Conditioned plot by factor

Graphics: Base

FUNCTION NAME	PURPOSE
add=TRUE	Adds a new plot on top of another (kind of)
axes=FALSE	Suppresses axis crea^on – you then make your own
xlab="STRING"	Makes the X label
ylab="STRING"	Makes the y-label
main="STRING"	Gives the plot a main ^tle
sub="STRING"	Gives a sub^tle
type="p"	Plot individual points
type="l"	Plot lines
type="b"	Plot points connected by lines
type="o"	Plot points overlaid by lines
type="n"	Suppresses ploVng but sets up device. Good for

Graphics: Base

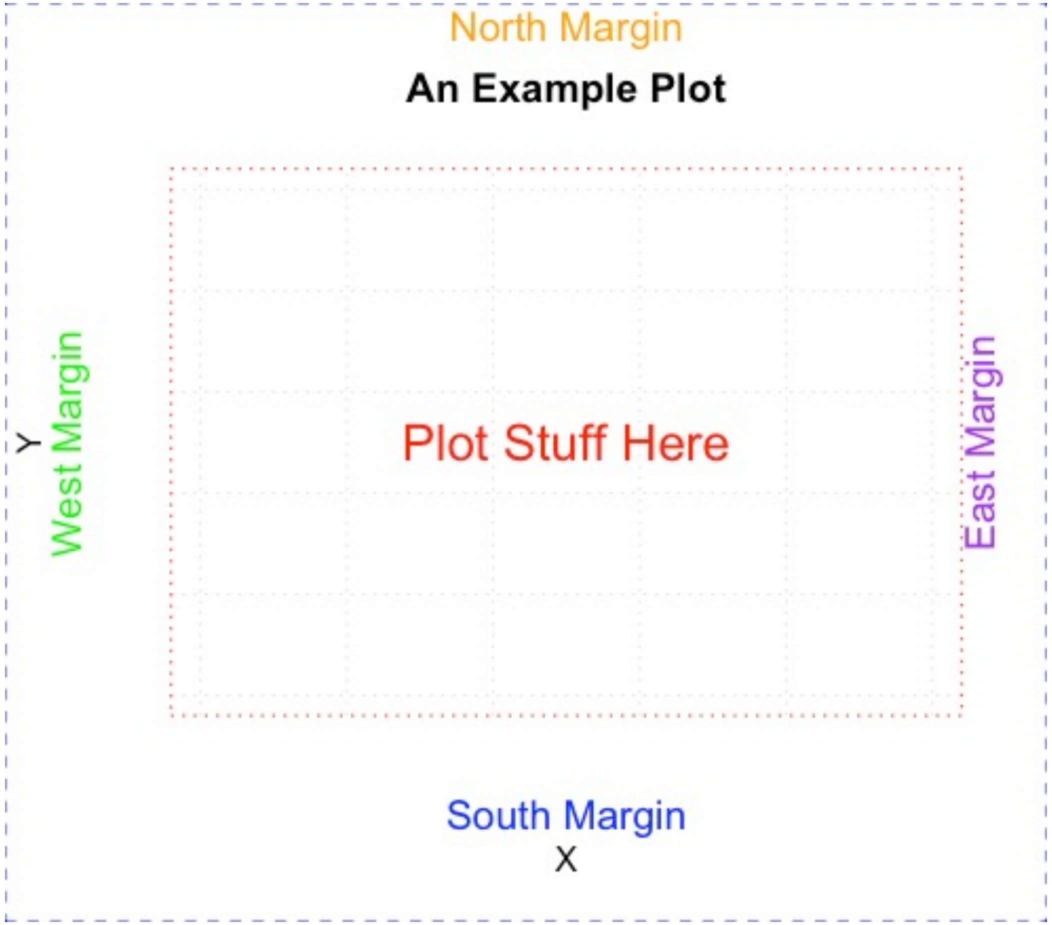
FUNCTION NAME	PURPOSE
mar	Specifies margins around plot area
col	Specify color of plot symbols
pch	Specify type of symbol example(pch)
lwd	Specify size of plot symbols
cex	Control font sizes (see also cex.main, cex.axis, cex.lab)
las	Direc^on of axis labels in rela^on to axis
lty	If lines are used this specifies line type (dashed, etc)
type="l"	Plot lines
type="b"	Plot points connected by lines
type="o"	Plot points overlaid by lines
type="n"	Suppresses ploVng but sets up device. Good for when

Graphics: Base

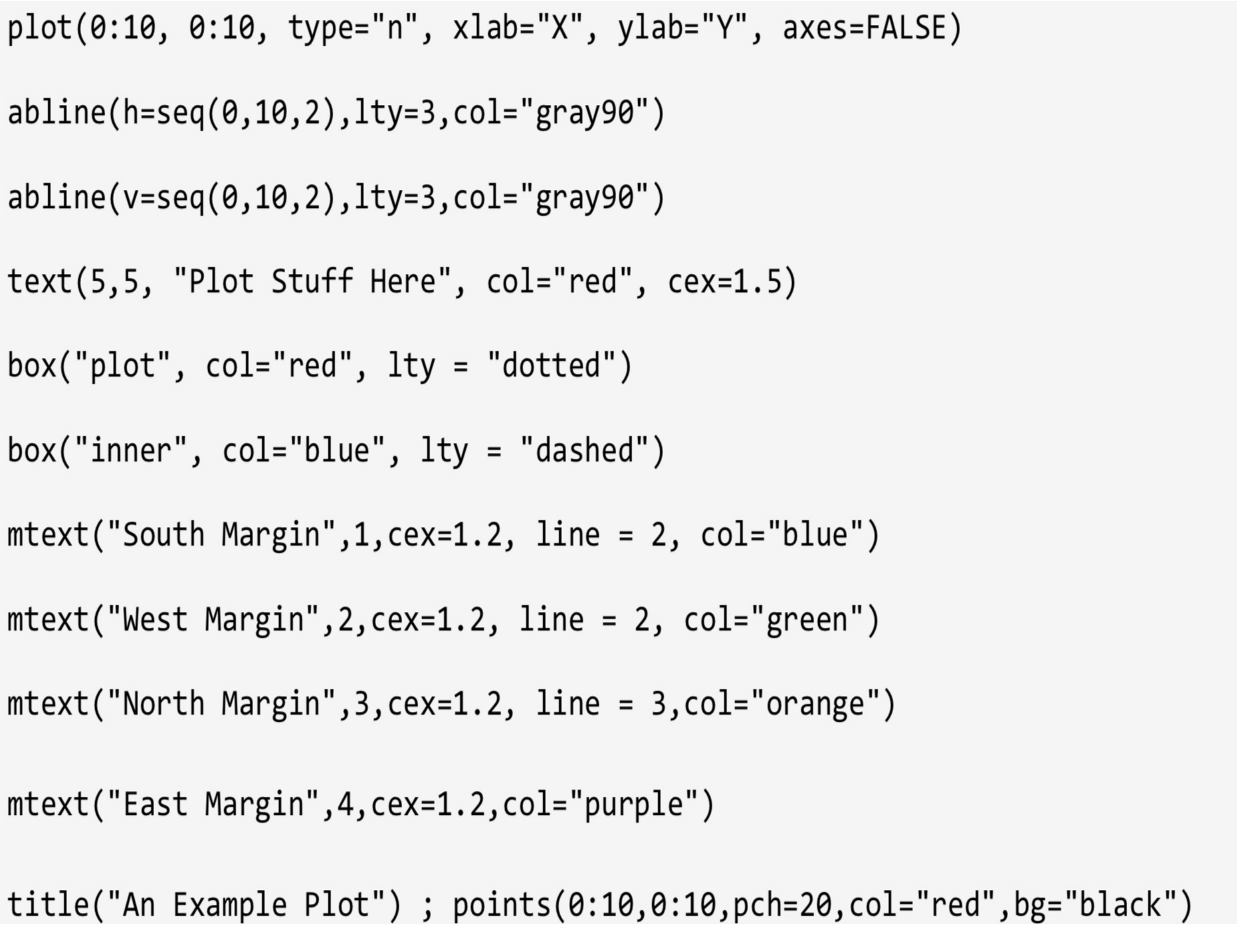
FUNCTION	RESULT OUTPUT
`pdf("file.pdf")`	Creates a PDF file called "file.pdf"
`png("file.png")`	Creates a PNG file
`jpeg("file.jpg")`	Creates a JPG file
`bmp(""file.bmp")`	Creates a BMP file
`postscript("file.ps")`	Creates a Postscript file
`win.meta("file.wmf")`	Creates a Windows meta file

Graphics: Base

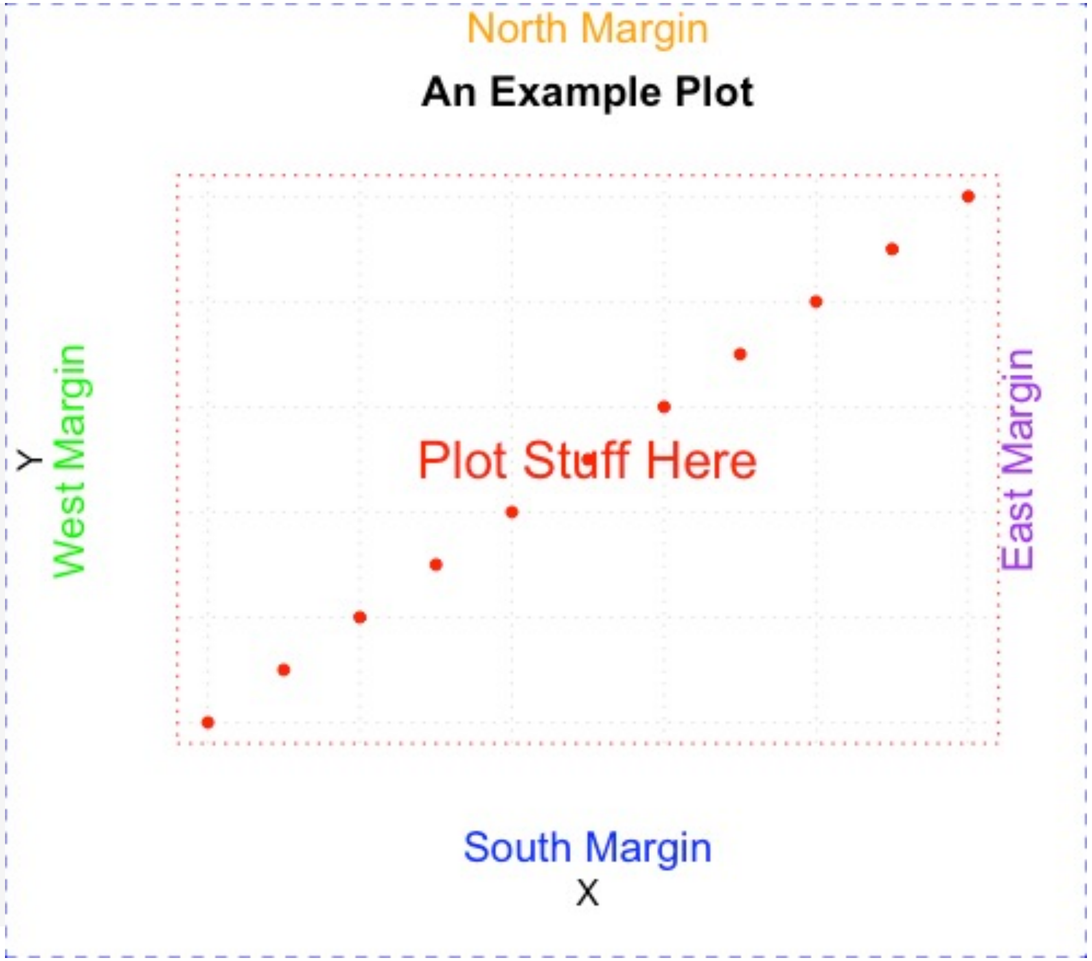
```
plot(0:10, 0:10, type="n", xlab="X", ylab="Y", axes=FALSE)

abline(h=seq(0,10,2),lty=3,col="gray90")

abline(v=seq(0,10,2),lty=3,col="gray90")

text(5,5, "Plot Stuff Here", col="red", cex=1.5)

box("plot", col="red", lty = "dotted")

box("inner", col="blue", lty = "dashed")

mtext("South Margin",1,cex=1.2,col="blue")

mtext("West Margin",2,cex=1.2,col="green")

mtext("North Margin",3,cex=1.2,col="orange")

mtext("East Margin",4,cex=1.2,col="purple")

title("An Example Plot")
```



Graphics: Base

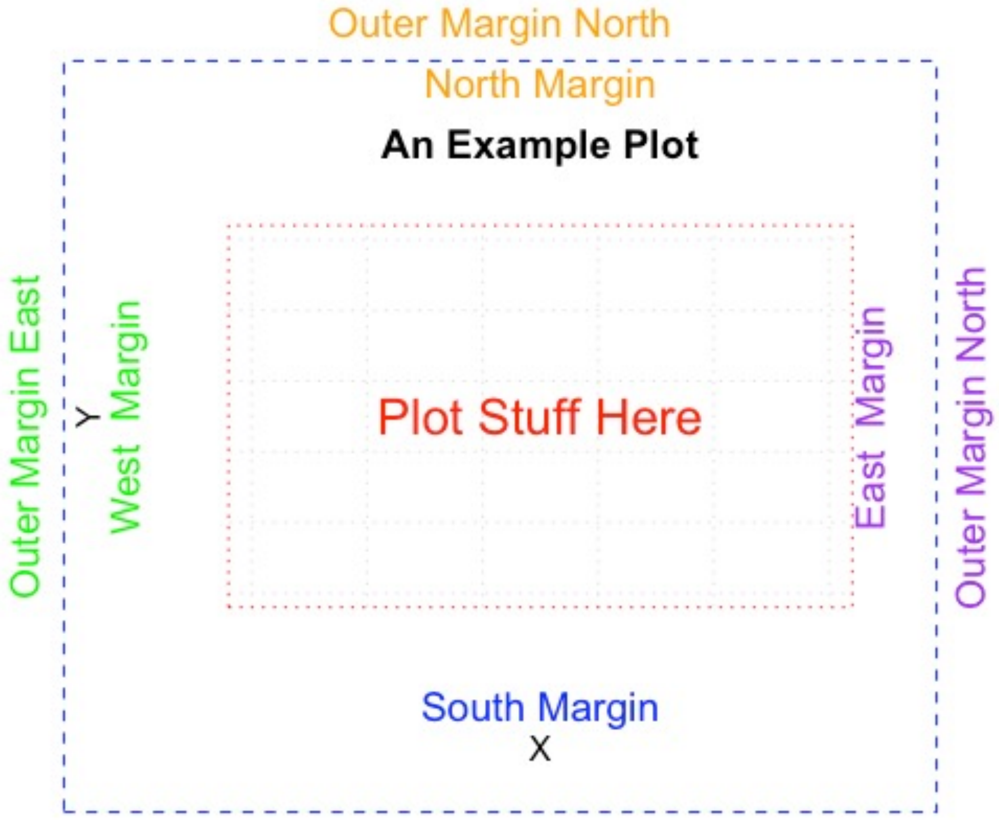


```
plot(0:10, 0:10, type="n", xlab="X", ylab="Y", axes=FALSE)

abline(h=seq(0,10,2),lty=3,col="gray90")

abline(v=seq(0,10,2),lty=3,col="gray90")

text(5,5, "Plot Stuff Here", col="red", cex=1.5)

box("plot", col="red", lty = "dotted")

box("inner", col="blue", lty = "dashed")

mtext("South Margin",1,cex=1.2, line = 2, col="blue")

mtext("West Margin",2,cex=1.2, line = 2, col="green")

mtext("North Margin",3,cex=1.2, line = 3,col="orange")

mtext("East Margin",4,cex=1.2,col="purple")

title("An Example Plot") ; points(0:10,0:10,pch=20,col="red",bg="black")
```



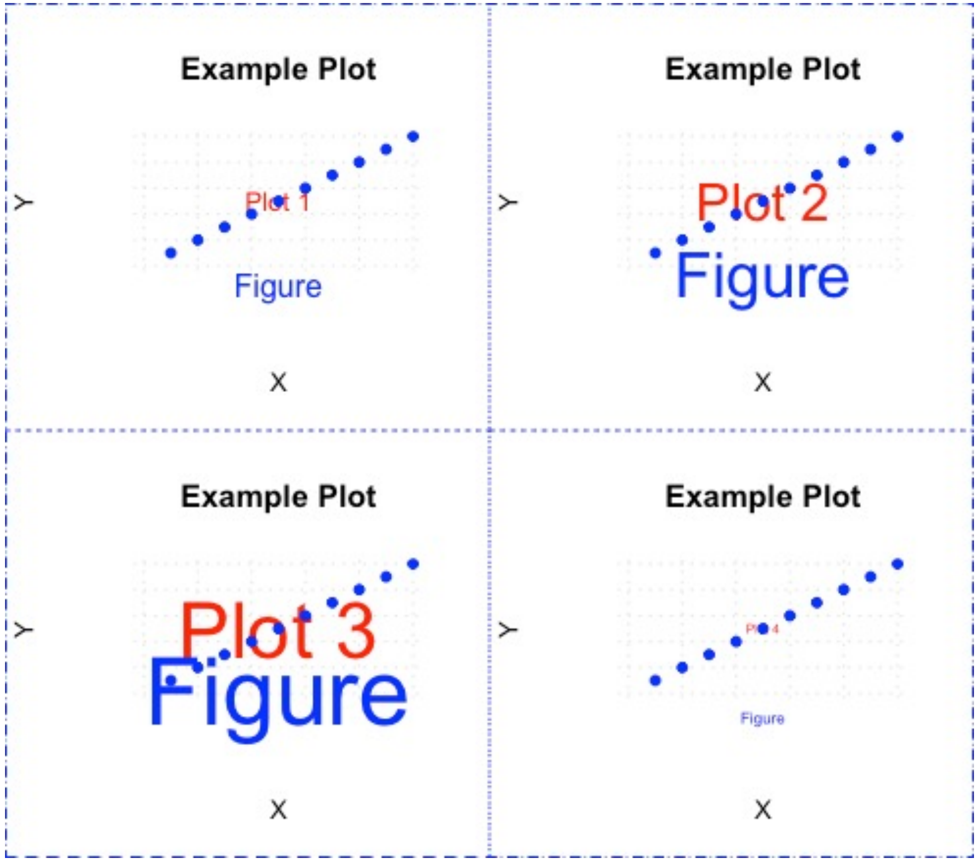
Graphics: Base

```
par(oma=c(2,2,2,2))

plot(0:10, 0:10, type="n", xlab="X", ylab="Y", axes=FALSE)

abline(h=seq(0,10,2), v=seq(0,10,2), lty=3,col="gray90")

text(5,5, "Plot Stuff Here", col="red", cex=1.5)

box("plot", col="red", lty = "dotted")

box("inner", col="blue", lty = "dashed")

mtext("South Margin",1, cex=1.2, line = 2, col="blue")
mtext("West  Margin",2, cex=1.2, line = 2, col="green")
mtext("North Margin",3, cex=1.2, line = 3, col="orange")
mtext("East  Margin",4, cex=1.2, col="purple")

title("An Example Plot")

mtext("Outer Margin East", 2, line=0.4, cex=1.2,col="green", outer=TRUE)

mtext("Outer Margin North", 3, line=0.4, cex=1.2,col="orange", outer=TRUE)

mtext("Outer Margin North", 4, line=0.4, cex=1.2,col="purple", outer=TRUE)
```



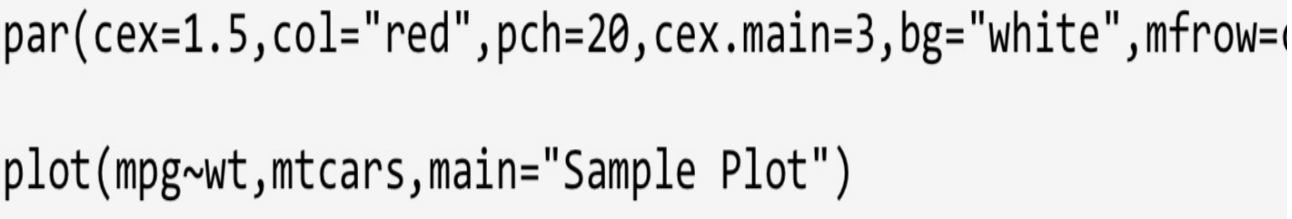
Graphics: Base

```
myPlot <- function(sometext, size1, size2) {
   plot(0:10, 0:10, type="n", xlab="X", ylab="Y",axes=F)
   abline(h=seq(0,10,2), v=seq(0,10,2), lty=3,col="gray90")
   text(5,5, sometext, col="red", cex=size1)
   box("inner", col="blue", lty = "dashed")
   box("figure", lty="dotted", col="blue")
   mtext("Figure", 1, cex=size2, col="blue")
   points(10:1,10:1,col = "blue", pch=20, cex=1.2)
   title("Example Plot")
}


par(mfrow=c(2,2))

myPlot("Plot 1",1,1)

myPlot("Plot 2",2,2)

myPlot("Plot 3",3,3)

myPlot("Plot 4",0.5,0.5)
```

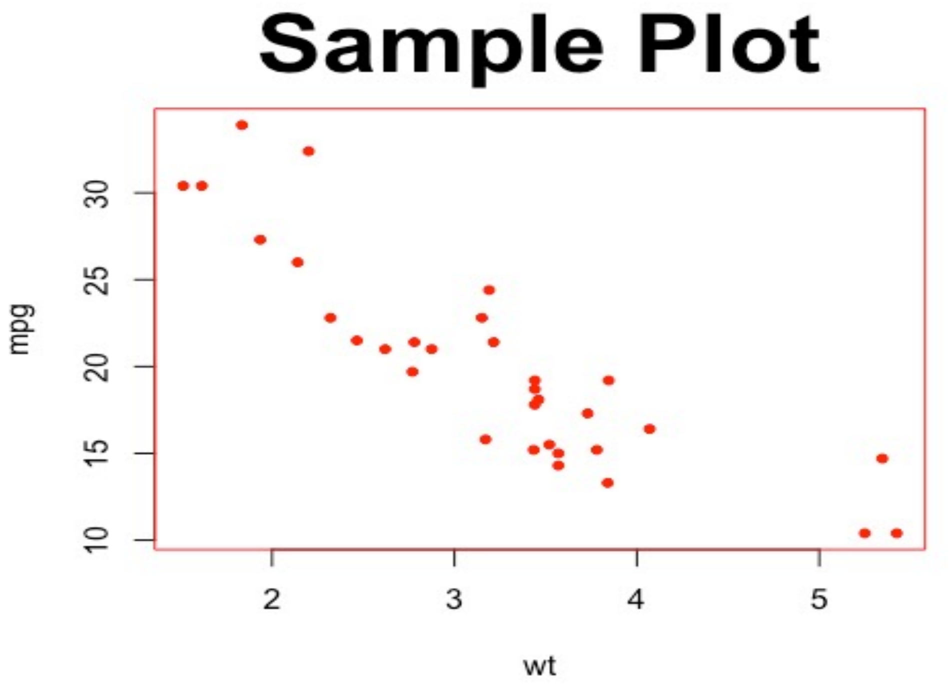


Graphics: Base

- 1) Specify all your color, font, and aesthetic choices using the par command and then plot stuff that will "inherit" the settings from par.

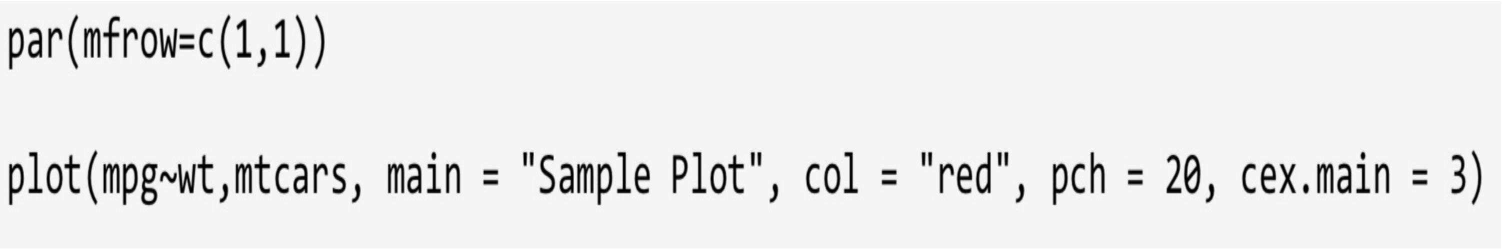


```
par(cex=1.5,col="red",pch=20,cex.main=3,bg="white",mfrow=(

plot(mpg~wt,mtcars,main="Sample Plot")
```

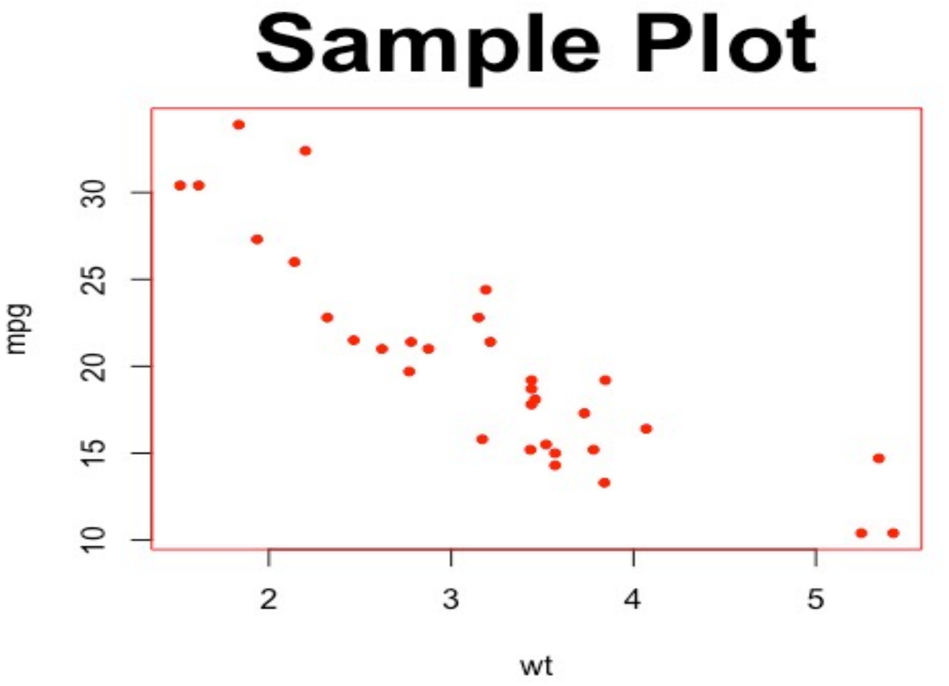


Graphics: Base

- 2) Use par only to set "major" layout aesthetics such as margin widths, number of plots per window, etc and then use options to the plot commands to specify colors, fonts, anc

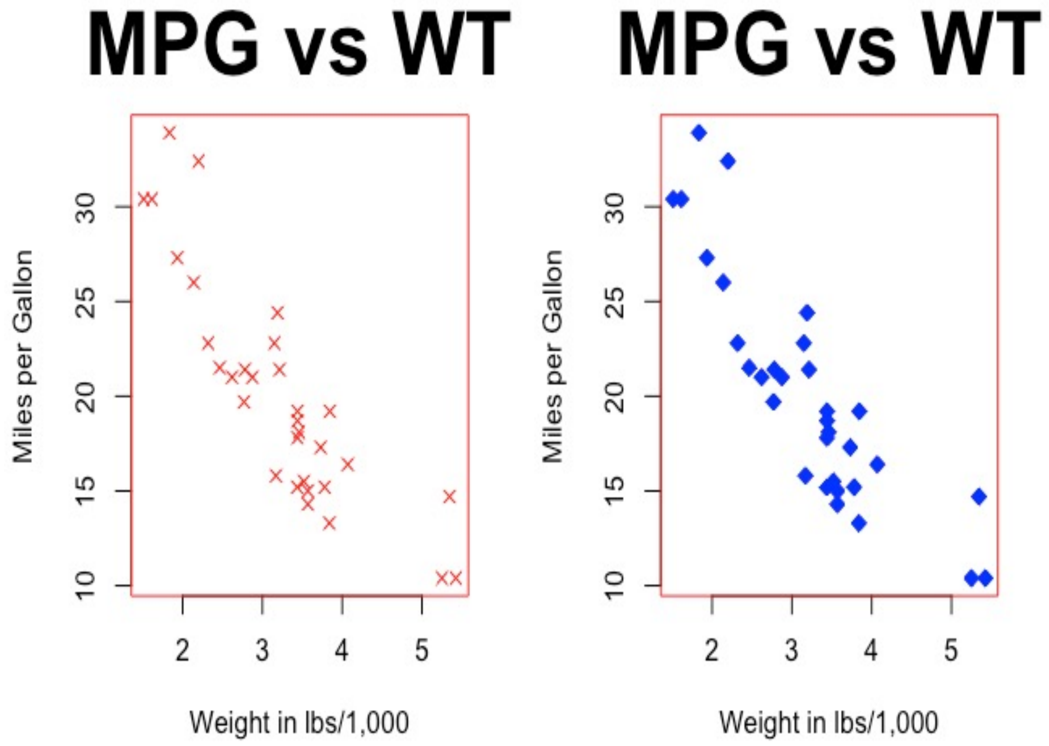


```
par(mfrow=c(1,1))

plot(mpg~wt,mtcars, main = "Sample Plot", col = "red", pch = 20, cex.main = 3)
```

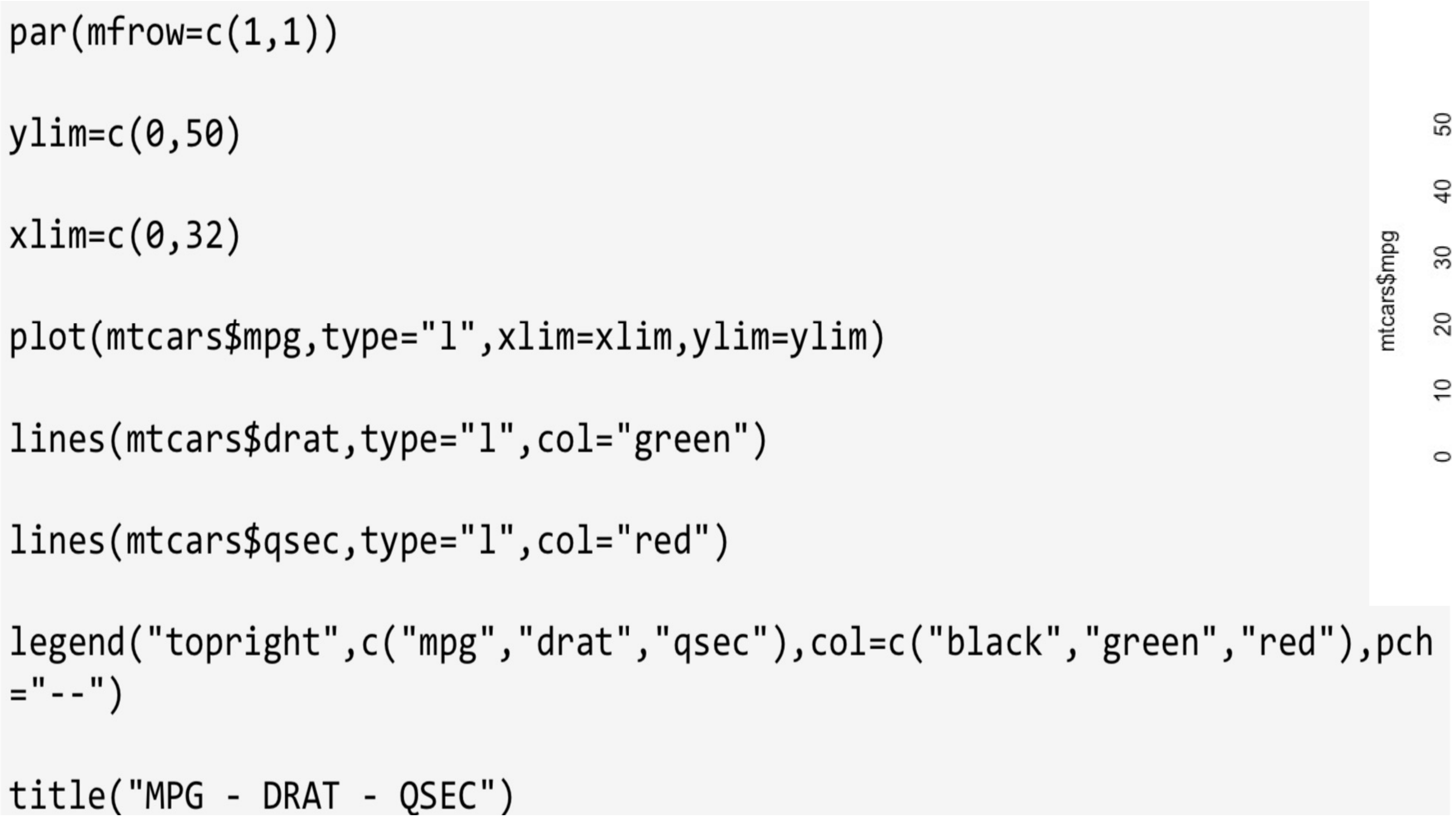


Graphics: Base

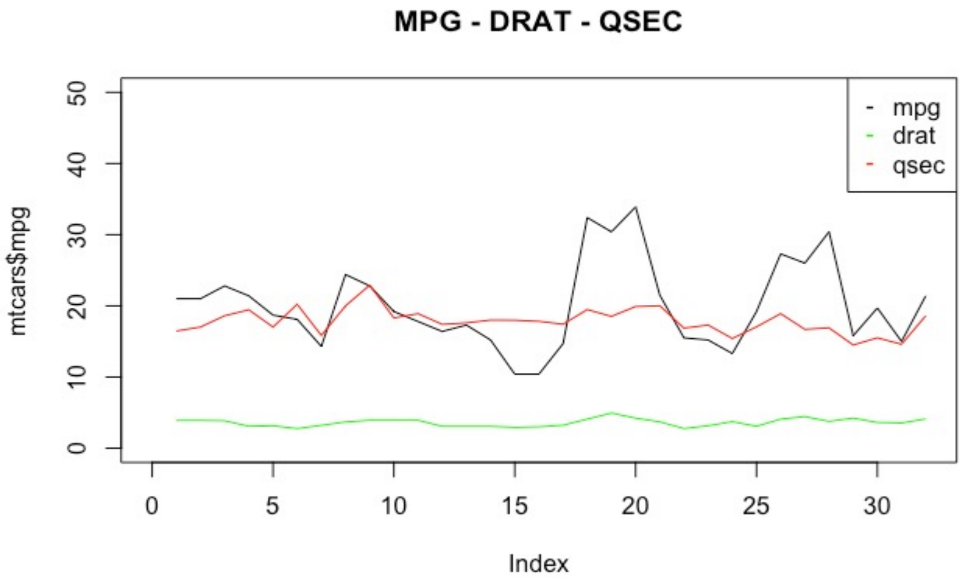
```
par(mfrow=c(1,2))

plot(mpg ~ wt, main="MPG vs WT", col="red",
      xlab="Weight in lbs/1,000",
      ylab="Miles per Gallon",
      pch=4, data = mtcars)

plot(mpg ~ wt, main="MPG vs WT", col="blue",
      xlab="Weight in lbs/1,000",
      ylab="Miles per Gallon",
      pch=23,
      bg="blue", data = mtcars)
```



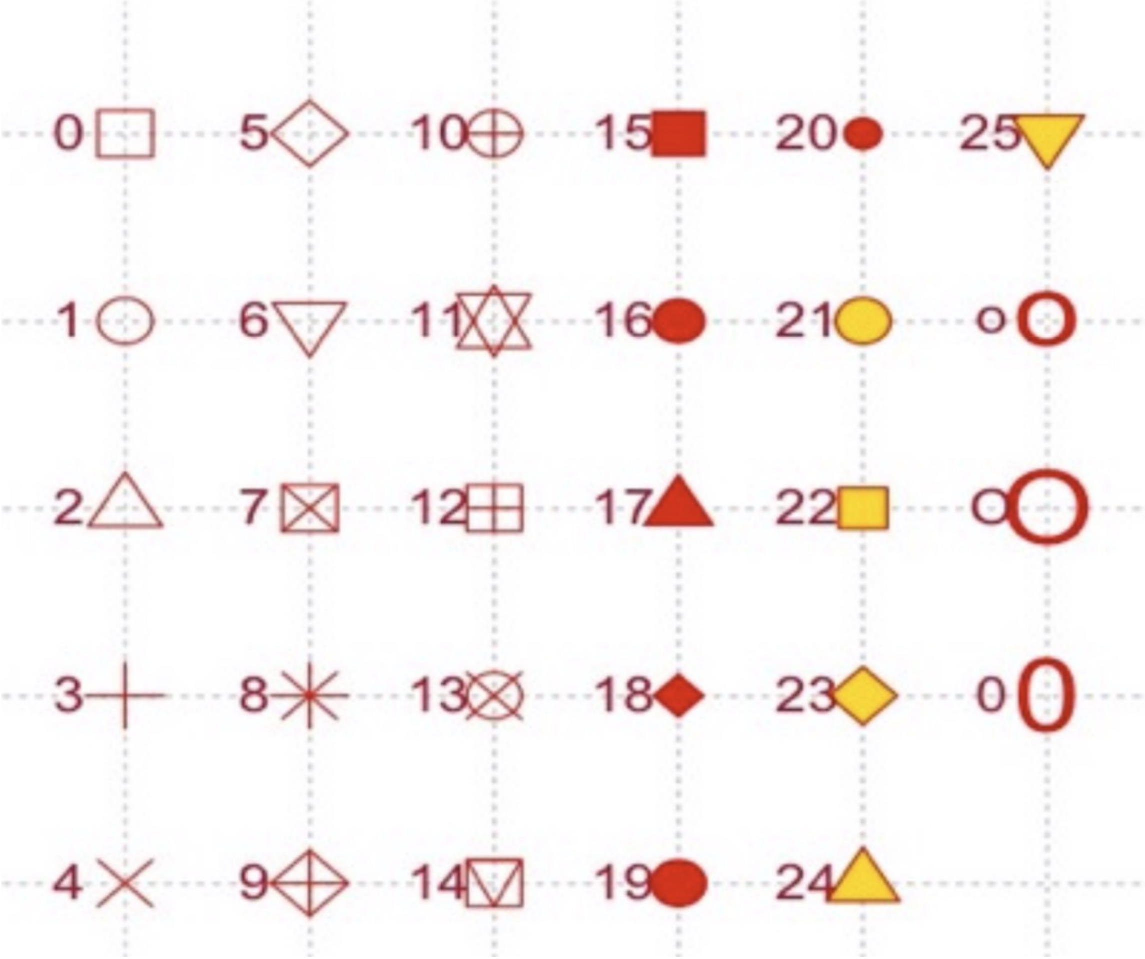
Graphics: Base

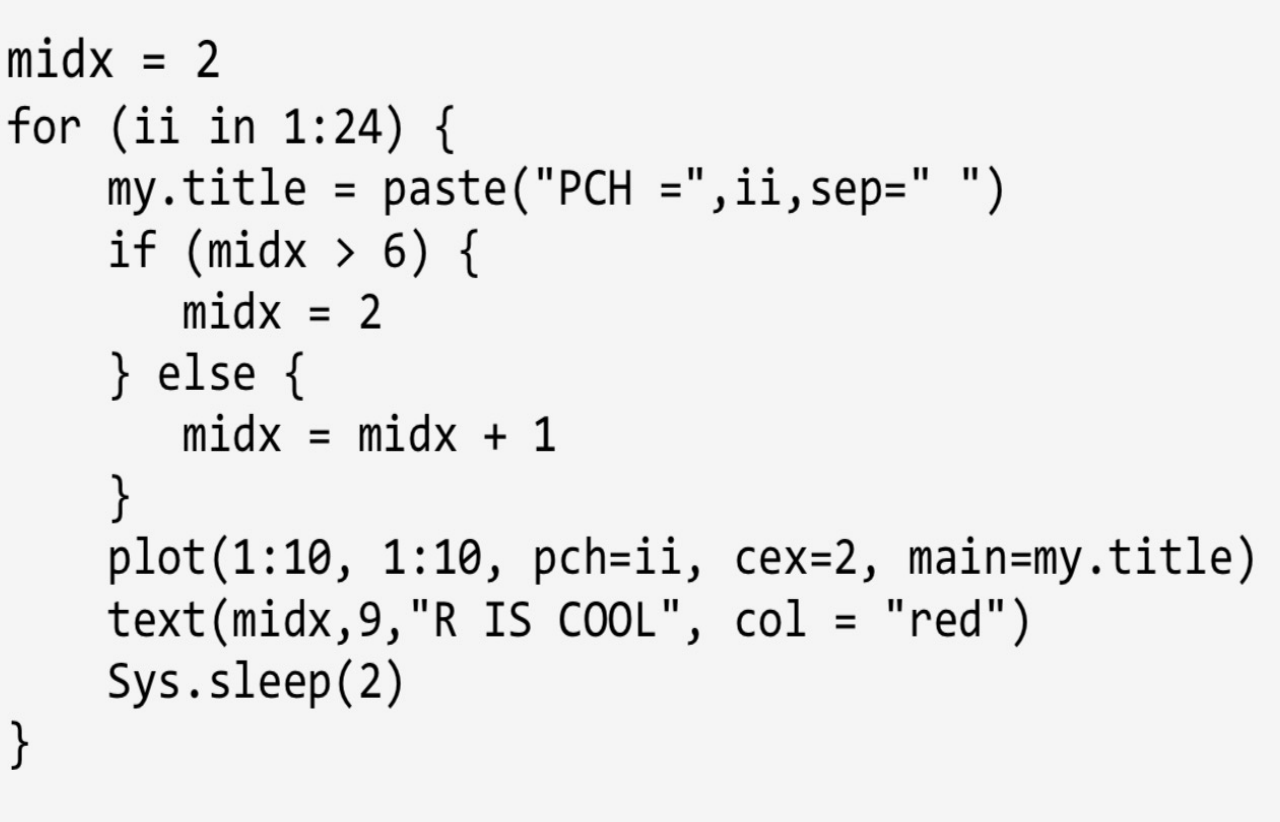


```
par(mfrow=c(1,1))

ylim=c(0,50)

xlim=c(0,32)

plot(mtcars$mpg,type="l",xlim=xlim,ylim=ylim)

lines(mtcars$drat,type="l",col="green")

lines(mtcars$qsec,type="l",col="red")

legend("topright",c("mpg","drat","qsec"),col=c("black","green","red"),pch
="--")

title("MPG - DRAT - QSEC")
```



Point type

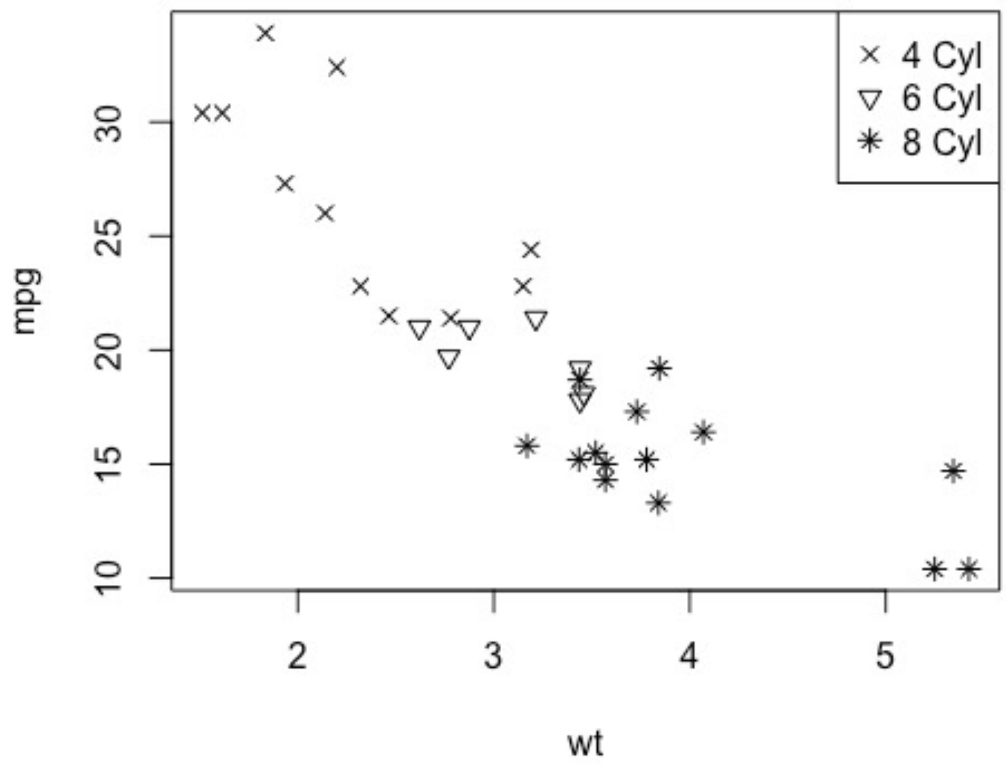




```
midx = 2
for (ii in 1:24) {
    my.title = paste("PCH =",ii,sep=" ")
    if (midx > 6) {
       midx = 2
    } else {
       midx = midx + 1
    }
    plot(1:10, 1:10, pch=ii, cex=2, main=my.title)
    text(midx,9,"R IS COOL", col = "red")
    Sys.sleep(2)
}
```

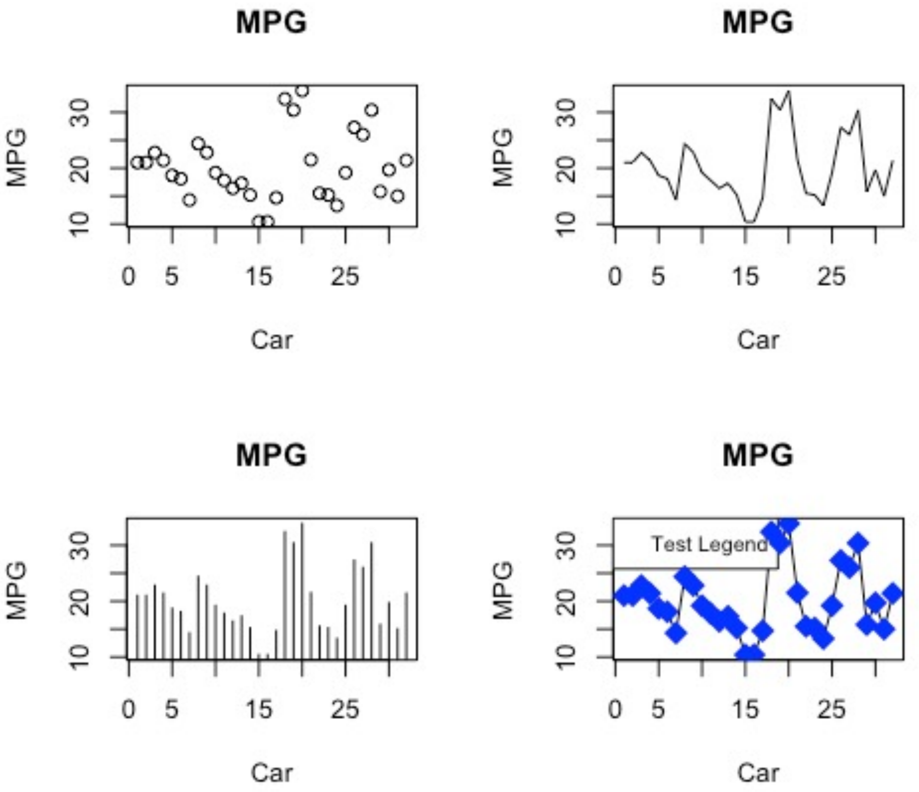
Point type

```
four.cyl = subset(mtcars, cyl == 4)

six.cyl = subset(mtcars, cyl == 6)

eight.cyl = subset(mtcars, cyl == 8)

plot(mpg~wt,data=mtcars,type="n")

points(mpg~wt,data=four.cyl,pch=4)

points(mpg~wt,data=six.cyl,pch=6)

points(mpg~wt,data=eight.cyl,pch=8)

legend("topright",c("4 Cyl","6 Cyl","8 Cyl"),
        pch=sort(unique(mtcars$cyl)))
```



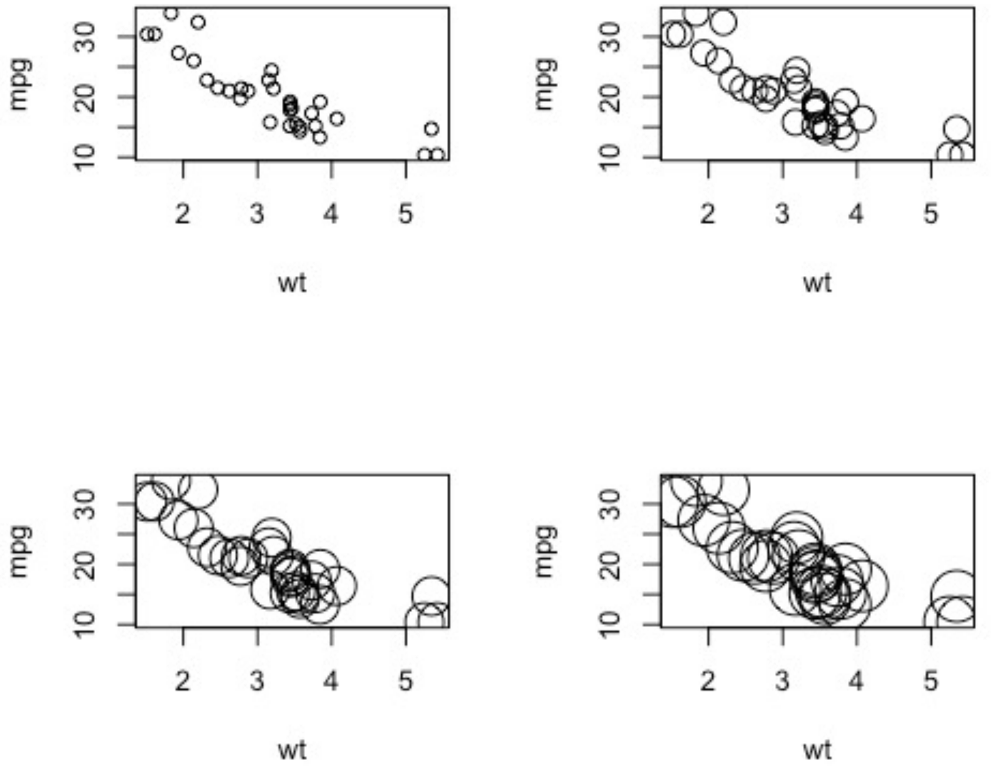
Line type

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

plot(mtcars$mpg,main="MPG",xlab="Car",ylab="MPG",type="p")

plot(mtcars$mpg,main="MPG",xlab="Car",ylab="MPG",type="l")

plot(mtcars$mpg,main="MPG",xlab="Car",ylab="MPG",type="h")

plot(mtcars$mpg,main="MPG",xlab="Car",ylab="MPG",type="o")

legend("topleft",legend=c("Test Legend"),cex=0.8)

points(mtcars$mpg, cex = 2.0,col = "blue",pch=18)
```



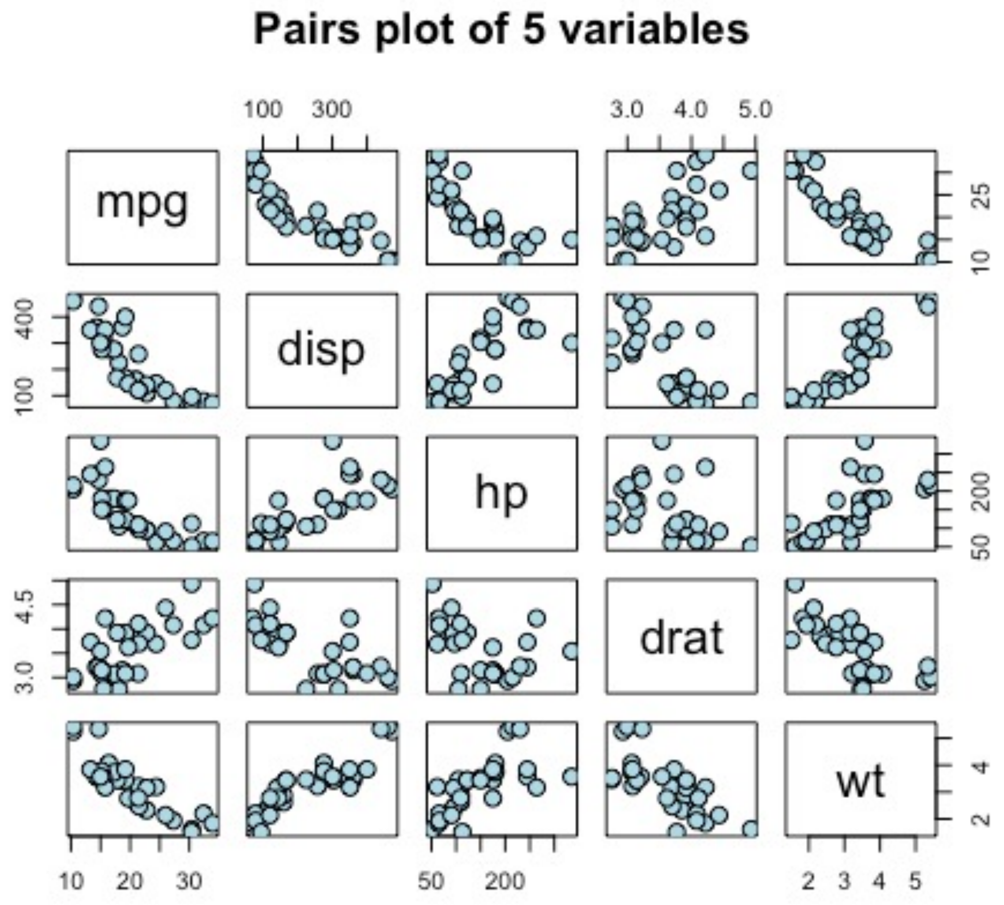
Font size

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

plot(mpg~wt,data=mtcars,pch=21,col="black",cex=1)

plot(mpg~wt,data=mtcars,pch=21,col="black",cex=1.5)

plot(mpg~wt,data=mtcars,pch=21,col="black",cex=2.5)

plot(mpg~wt,data=mtcars,pch=21,col="black",cex=3.5)

# OR MORE SIMPLY

par(mfrow=c(2,2))
my.font <- function(size) {
    plot(mpg~wt,data=mtcars,pch=21,col="black",cex=size)
}

sapply(seq(1,4,1), my.font)
```



Scatterplot

```
pairs(mtcars[,c(1,3:6)], cex = 1.5, pch=21,
      bg="light blue",main="Pairs plot of 5 variables")
```

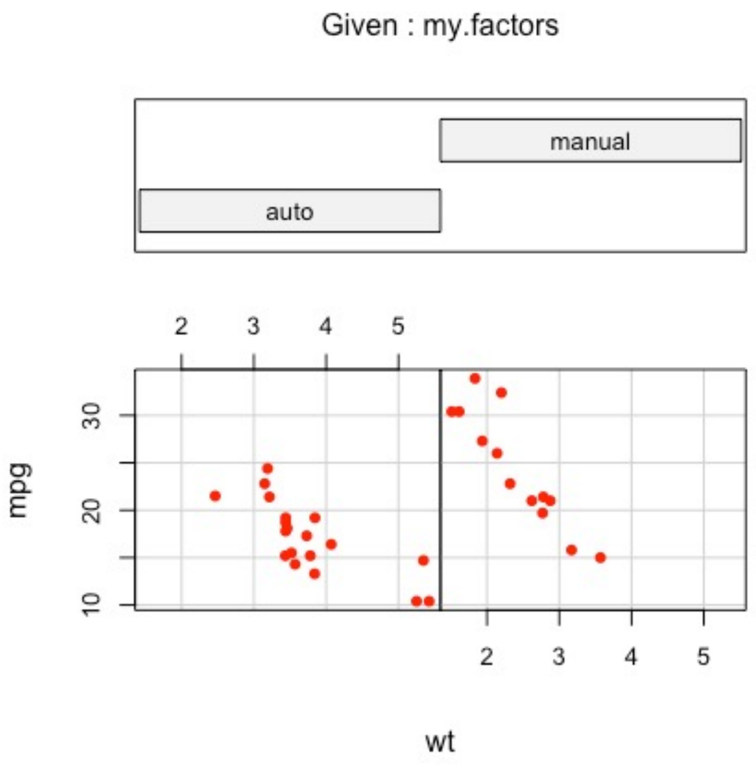


Coplot

Conditioned plots let us display a relationship between continuous variables on a per category basis. This implies that there is at least one continuous variable that will be plotted in groups or categories.

Coplot

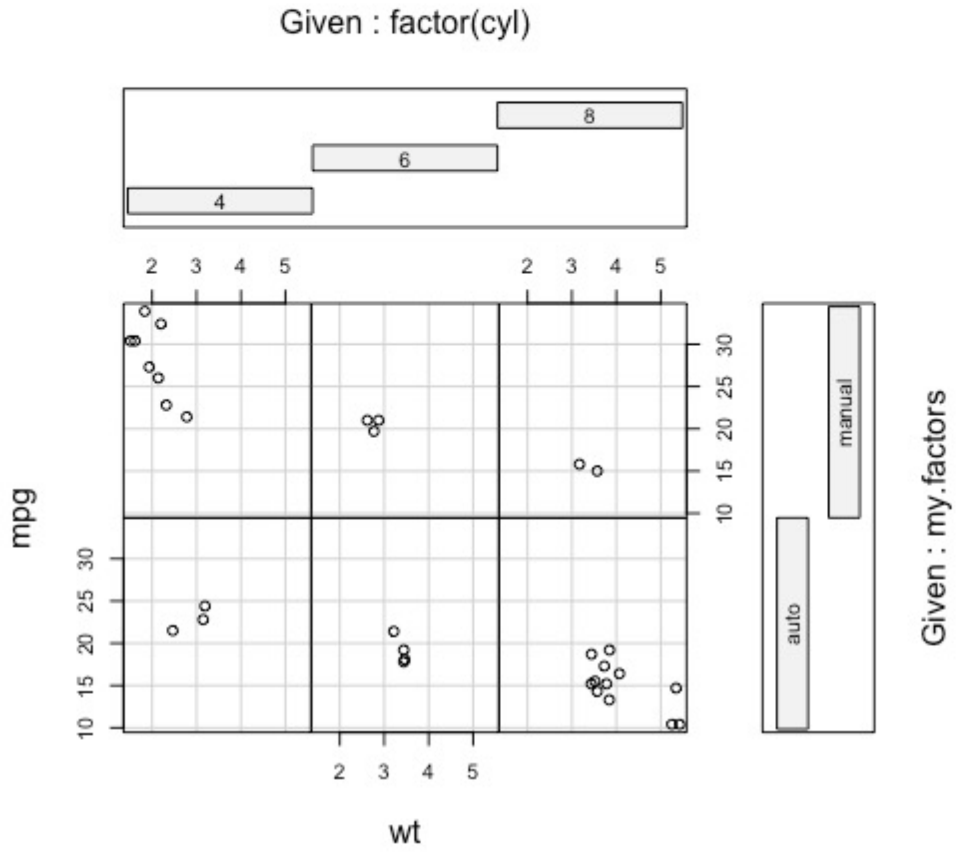
```
my.factors = factor(mtcars$am,labels=c("auto","manual"))

coplot(mpg~wt | my.factors,data=mtcars,col="red",
       pch=16,main="MPG vs Weight")
```



Coplot

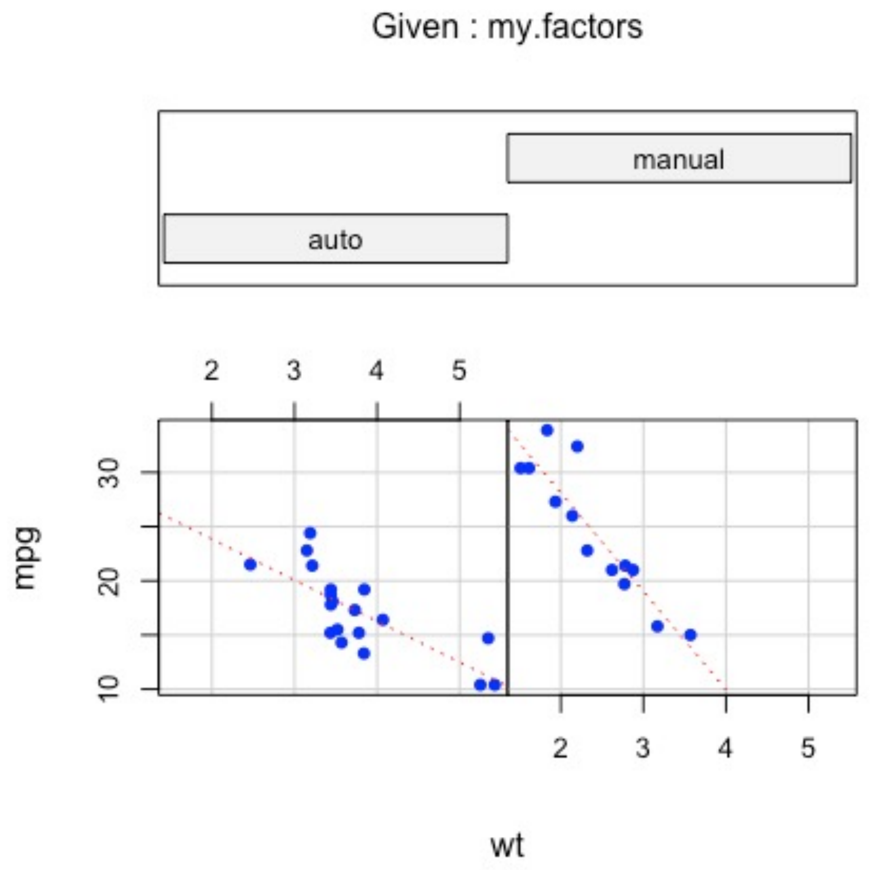
Note that you can plot conditionally on more than a single factor:

```
coplot(mpg~wt|factor(cyl)*my.factors,data=mtcars)
```

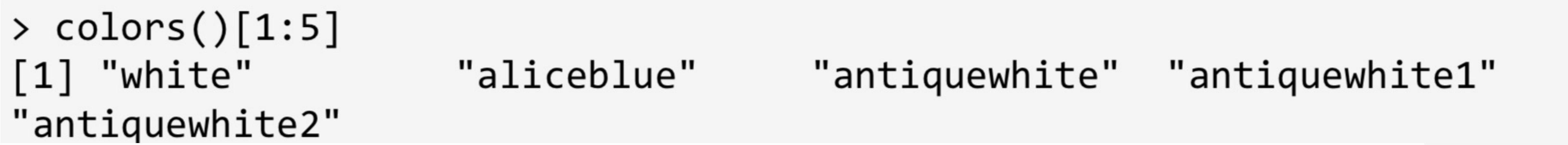


Coplot

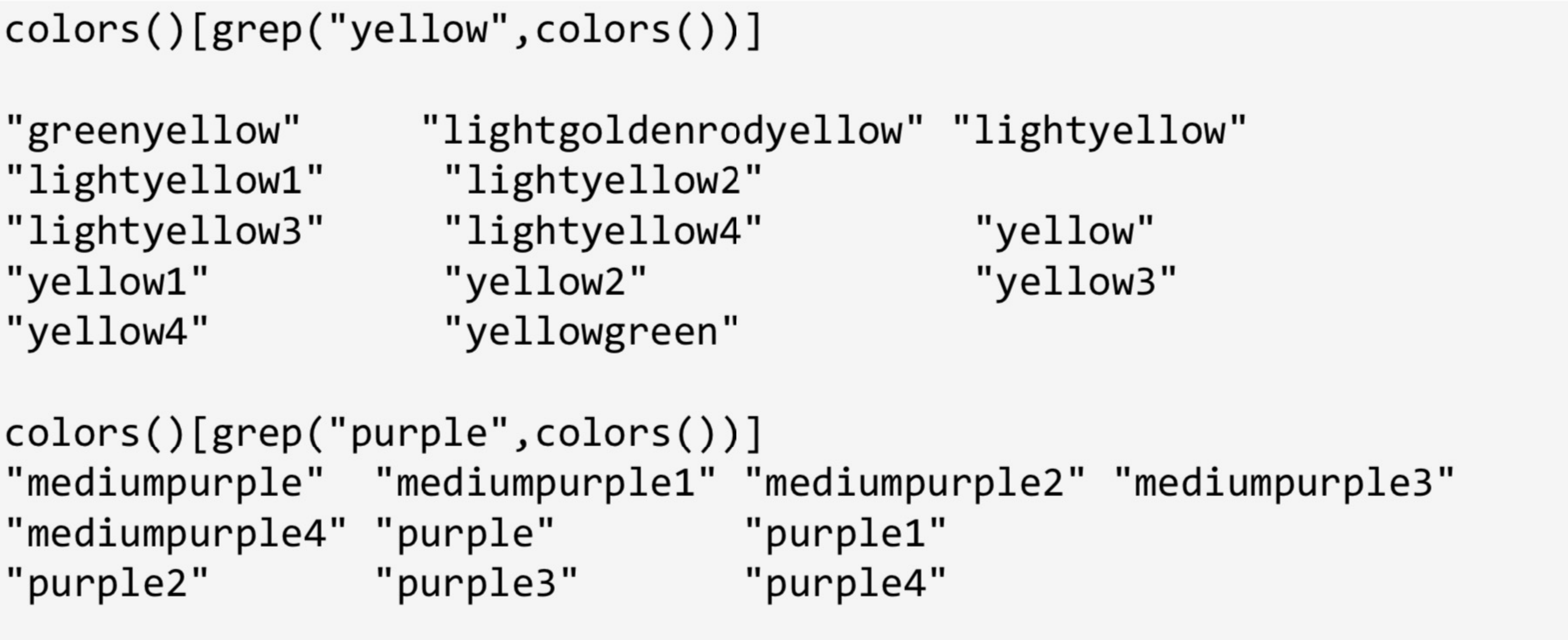
```
my.coplot <- function(x,y,...) {
   points(x,y,col="blue",pch=16)
   abline(lm(y~x),col="red",lty=3)
}


coplot(mpg~wt|my.factors, data=mtcars, col="red",
        pch=16,main="MPG vs Weight",
        panel=my.coplot)
```



Colors



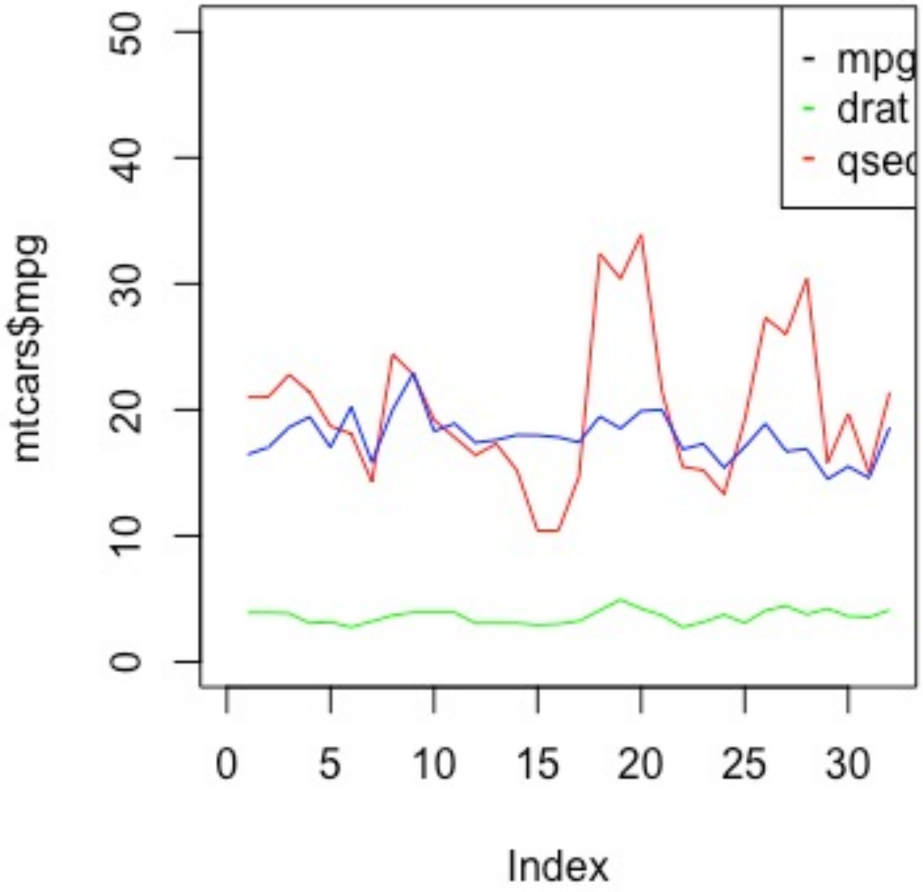
```
> colors()[1:5]
[1] "white"          "aliceblue"      "antiquewhite"  "antiquewhite1"
"antiquewhite2"
```



```
colors()[grep("yellow",colors())]

"greenyellow"     "lightgoldenrodyellow" "lightyellow"
"lightyellow1"     "lightyellow2"
"lightyellow3"     "lightyellow4"          "yellow"
"yellow1"          "yellow2"               "yellow3"
"yellow4"          "yellowgreen"

colors()[grep("purple",colors())]
"mediumpurple"  "mediumpurple1" "mediumpurple2" "mediumpurple3"
"mediumpurple4" "purple"        "purple1"
"purple2"       "purple3"       "purple4"
```

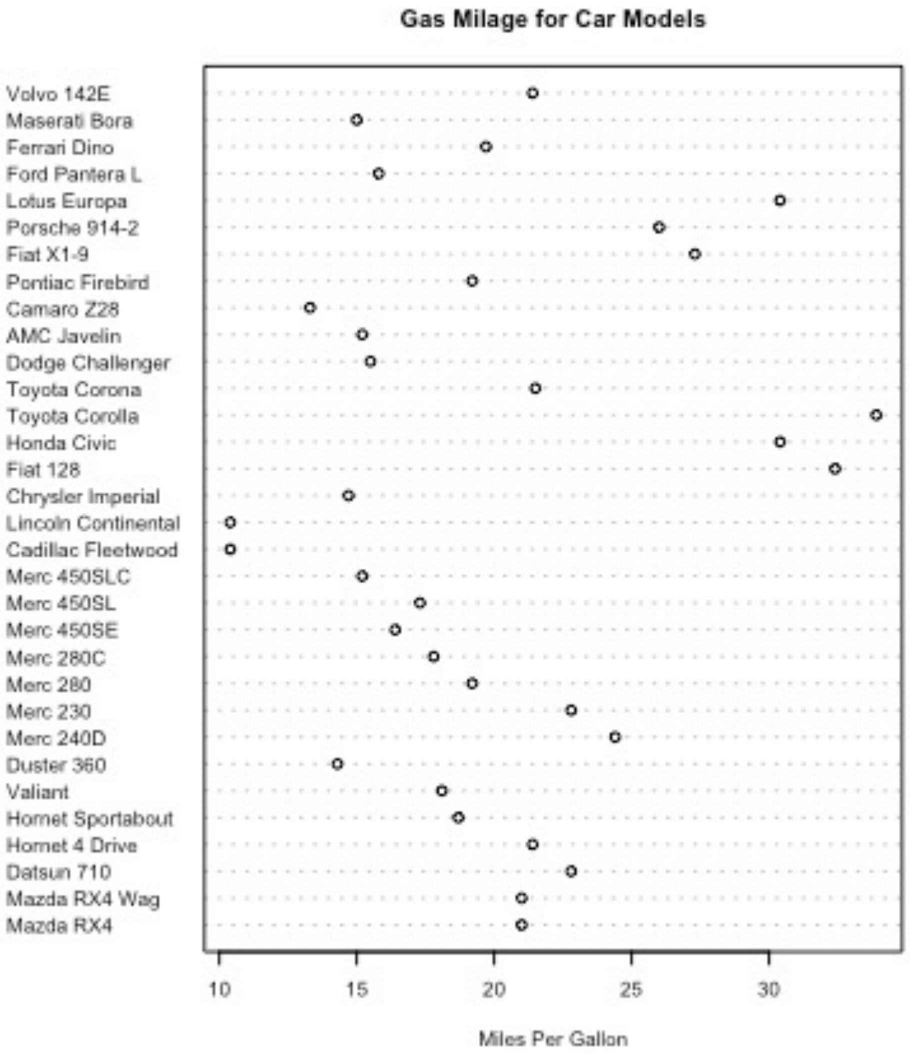
Colors

```
my.cols = rainbow(3)

my.cols
[1] "#FF0000FF" "#00FF00FF" "#0000FFFF"

par(mfrow=c(1,1))

ylim=c(0,50)

xlim=c(0,32)

plot(mtcars$mpg,type="l",xlim=xlim,ylim=ylim,col=my.cols[1])

lines(mtcars$drat,type="l",col=my.cols[2])

lines(mtcars$qsec,type="l",col=my.cols[3])

legend("topright",c("mpg","drat","qsec"),
        col=c("black","green","red"),pch="--")
```



dotplots

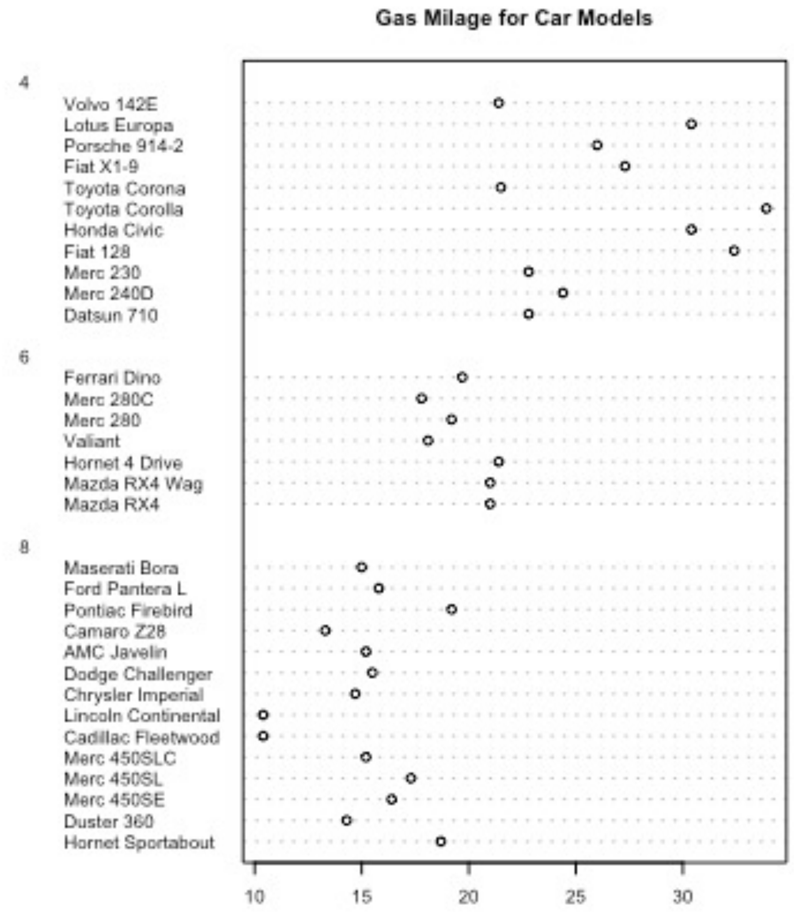
```
dotchart(mtcars$mpg,labels=row.names(mtcars),
    cex=.5,
    main="Gas Milage for Car Models",
    xlab="Miles Per Gallon")
```



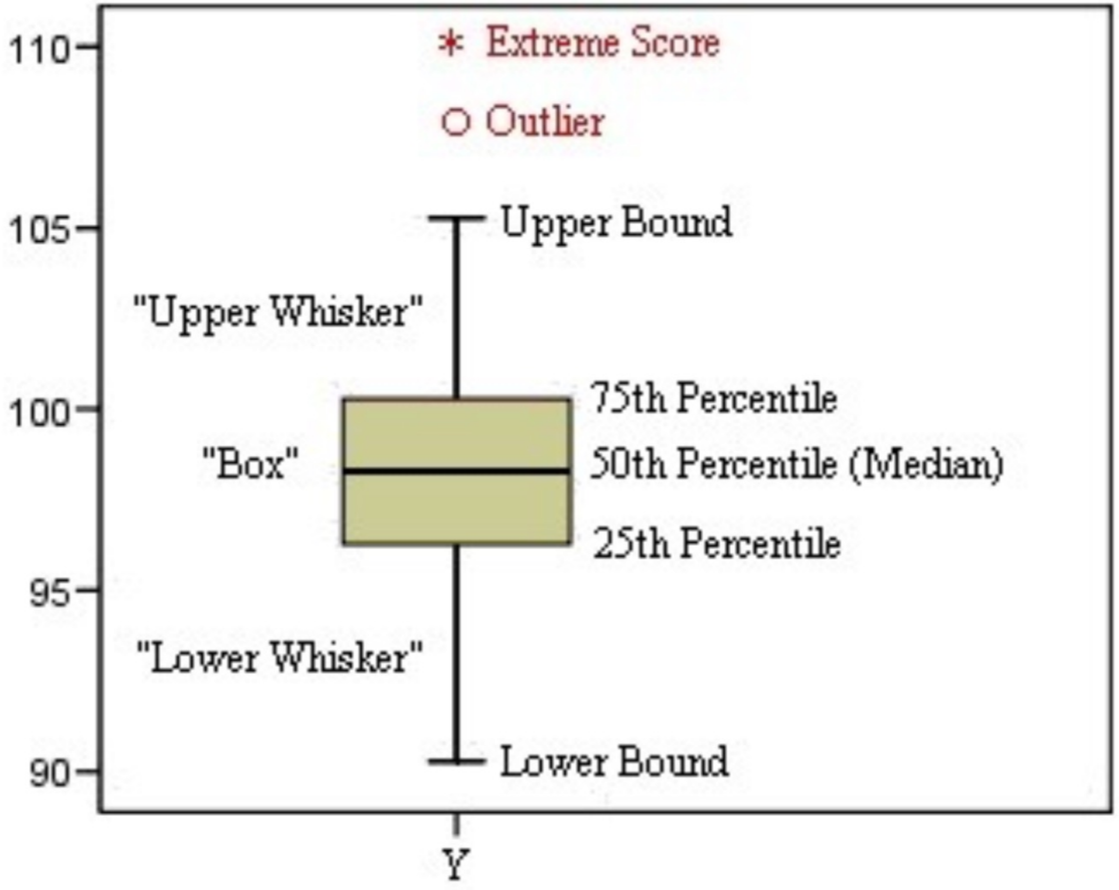
dotplots

You could also add a groups= option to designate a factor specifying how the elements of x are to be grouped. If so, the option gcolor= controls the color of the groups label. cex controls the size of the labels.

```
dotchart(mtcars$mpg, labels = row.names(mtcars),
           cex=.5, main="Gas Milage for Car Models",
           groups=factor(mtcars$cyl))
```



boxplot

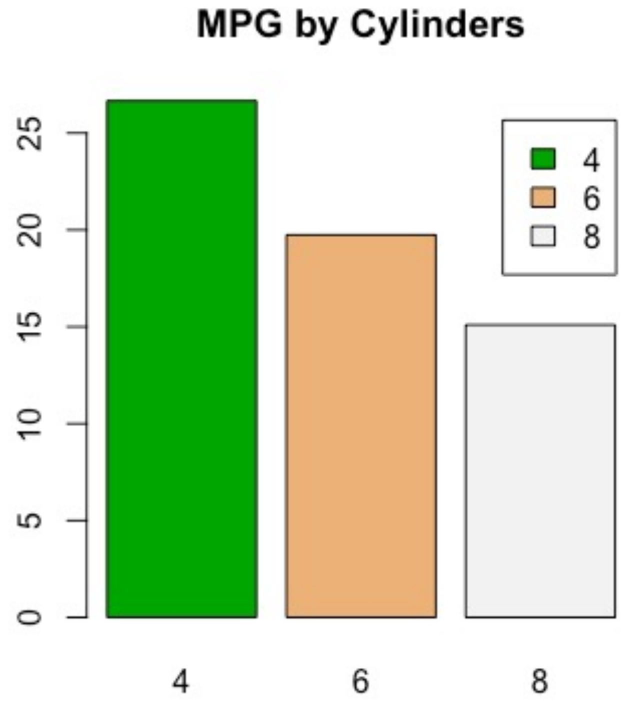


http://psystats.wikispaces.com/Boxplots

barplot

Barplots are also useful for showing statistical summaries of a continuous variable(s) across a number of groups. As an example, with the mtcars data set, let's get the mean of MPG across all cylinder types. We get:

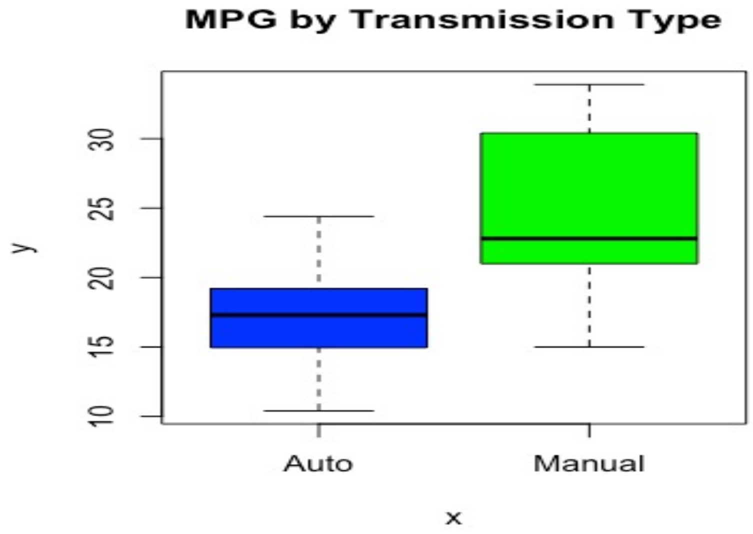
```
my.table = ( tapply(mtcars$mpg,mtcars$cyl,mean) )
       4        6        8
26.66364 19.74286 15.10000



barplot(my.table, col=terrain.colors(3),
        legend=TRUE, main="MPG by Cylinders")
```



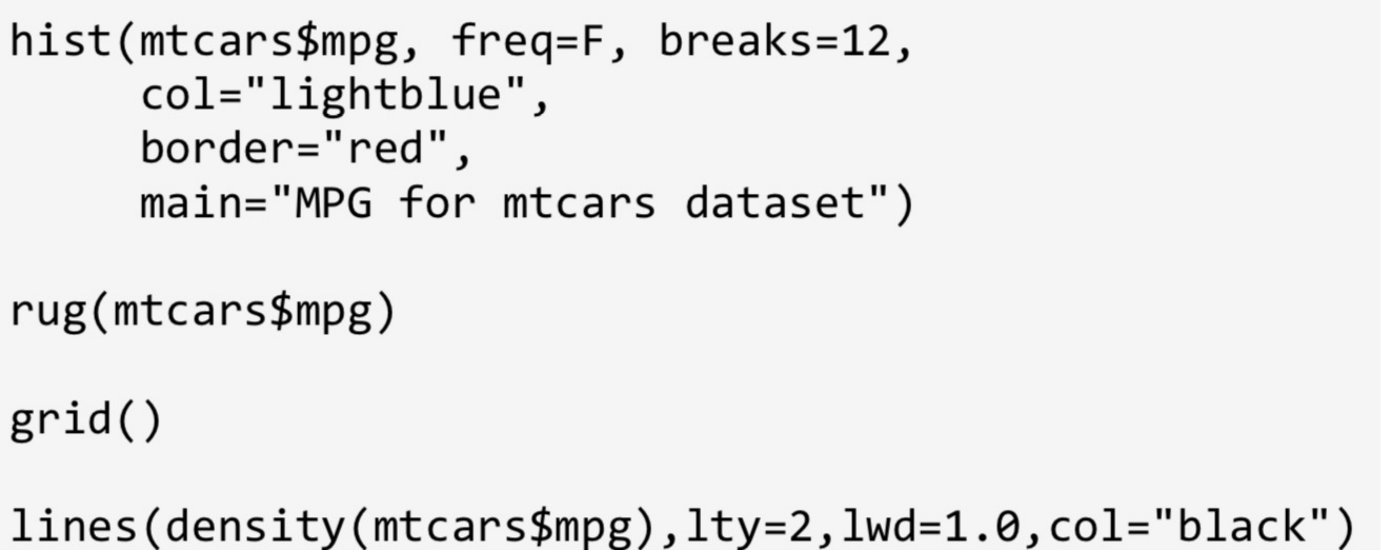
Boxplot

If you are doing a X/Y scatterplot and one of them is a categorical variable then the resulting plot will be a boxplot. Let's look at the mtcars dataset now.

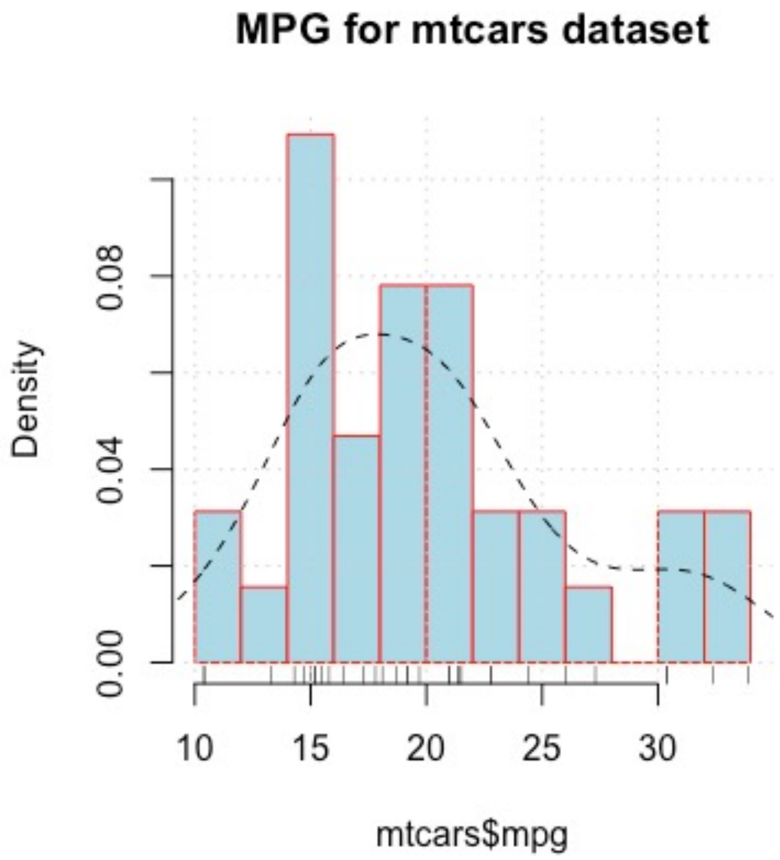
```
transmissions = factor(mtcars$am, labels=c("Auto","Manual"))

plot(transmissions,mtcars$mpg,
       main="MPG by Transmission Type", col=c("blue","green"))
```



Histogram



```
hist(mtcars$mpg, freq=F, breaks=12,
     col="lightblue",
     border="red",
     main="MPG for mtcars dataset")

rug(mtcars$mpg)

grid()

lines(density(mtcars$mpg),lty=2,lwd=1.0,col="black")
```



Summary

- R color, font size, point
- Boxplot, scatterplot, histogram, coplot, dotplot
- Next lecture Friday 12pm NOT Thursday