Class: 13rd April 2018

**Topics**

1. Plotting Data
2. Working with datasets
3. Correlations and Covariance with plotting
4. Data frames accessing
5. Reading files
6. Gist about packages and installing packages.

**Plotting operations in R**

The most used plotting function in R programming is the plot() function. It is a generic function, meaning, it has many methods which are called according to the type of object passed to plot().

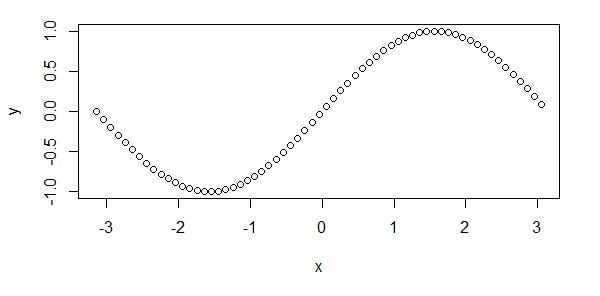
In the simplest case, we can pass in a vector and we will get a scatter plot of magnitude vs index. But generally, we pass in two vectors and a scatter plot of these points are plotted.

For example, the command plot(c(1,2),c(3,5)) would plot the points (1,3) and (2,5).

Here is a more concrete example where we plot a sine function form range -pi to pi.

x <- seq(-pi,pi,0.1)

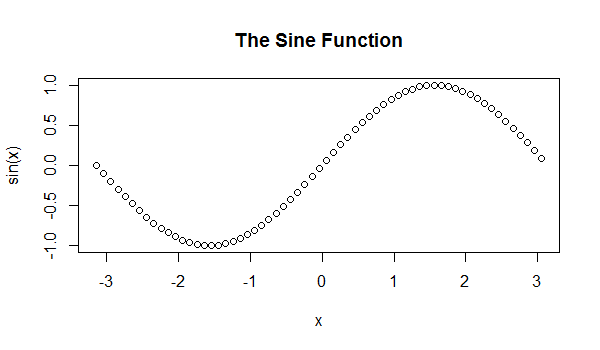
plot(x, sin(x))



**Adding Titles and Labelling Axes**

We can add a title to our plot with the parameter main. Similarly, xlab and ylab can be used to label the x-axis and y-axis respectively.

plot(x, sin(x),main="The Sine Function",ylab="sin(x)")



**Changing Color and Plot Type**

We can see above that the plot is of circular points and black in color. This is the default color.

We can change the plot type with the argument type. It accepts the following strings and has the given effect.

"p" - points

"l" - lines

"b" - both points and lines

"c" - empty points joined by lines

"o" - overplotted points and lines

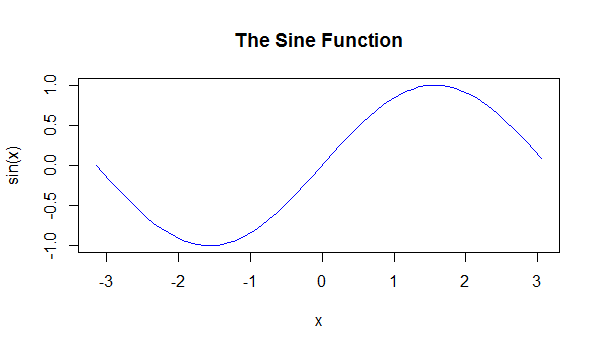
"s" and "S" - stair steps

"h" - histogram-like vertical lines

"n" - does not produce any points or lines

Similarly, we can define the color using col.

plot(x, sin(x),main="The Sine Function",ylab="sin(x)",type="l",col="blue")



**Overlaying Plots Using legend() function**

Calling plot() multiple times will have the effect of plotting the current graph on the same window replacing the previous one.

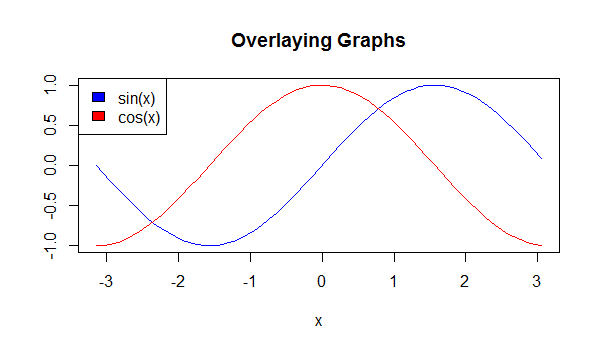
However, sometimes we wish to overlay the plots in order to compare the results.

This is made possible with the functions lines() and points() to add lines and points respectively, to the existing plot.

plot(x, sin(x),main="Overlaying Graphs",ylab="",type="l",col="blue")

lines(x,cos(x), col="red")

legend("topleft",c("sin(x)","cos(x)"),fill=c("blue","red"))



We have used the function legend() to appropriately display the legend. Visit legend() function to learn more.

Also visit plot() function to learn more about different arguments plot() function can take, and more examples.

**Working with datasets**

Datasets in R can be of csv or text files. You can access a dataset by using the following commands.

The default datasets files that come with R are as follow:

AirPassengers Monthly Airline Passenger Numbers 1949-1960

BJsales Sales Data with Leading Indicator

BJsales.lead (BJsales) Sales Data with Leading Indicator

BOD Biochemical Oxygen Demand

CO2 Carbon Dioxide Uptake in Grass Plants

ChickWeight Weight versus age of chicks on different diets

DNase Elisa assay of DNase

EuStockMarkets Daily Closing Prices of Major European Stock Indices, 1991-1998

Formaldehyde Determination of Formaldehyde

HairEyeColor Hair and Eye Color of Statistics Students

Harman23.cor Harman Example 2.3

Harman74.cor Harman Example 7.4

Indometh Pharmacokinetics of Indomethacin

InsectSprays Effectiveness of Insect Sprays

JohnsonJohnson Quarterly Earnings per Johnson & Johnson Share

LakeHuron Level of Lake Huron 1875-1972

LifeCycleSavings Intercountry Life-Cycle Savings Data

Loblolly Growth of Loblolly pine trees

Nile Flow of the River Nile

Orange Growth of Orange Trees

OrchardSprays Potency of Orchard Sprays

PlantGrowth Results from an Experiment on Plant Growth

Puromycin Reaction Velocity of an Enzymatic Reaction

Seatbelts Road Casualties in Great Britain 1969-84

Theoph Pharmacokinetics of Theophylline

Titanic Survival of passengers on the Titanic

ToothGrowth The Effect of Vitamin C on Tooth Growth in Guinea Pigs

UCBAdmissions Student Admissions at UC Berkeley

UKDriverDeaths Road Casualties in Great Britain 1969-84

UKgas UK Quarterly Gas Consumption

USAccDeaths Accidental Deaths in the US 1973-1978

USArrests Violent Crime Rates by US State

USJudgeRatings Lawyers' Ratings of State Judges in the US Superior Court

USPersonalExpenditure Personal Expenditure Data

UScitiesD Distances Between European Cities and Between US Cities

VADeaths Death Rates in Virginia (1940)

WWWusage Internet Usage per Minute

WorldPhones The World's Telephones

ability.cov Ability and Intelligence Tests

airmiles Passenger Miles on Commercial US Airlines, 1937-1960

airquality New York Air Quality Measurements

anscombe Anscombe's Quartet of 'Identical' Simple Linear Regressions

attenu The Joyner-Boore Attenuation Data

attitude The Chatterjee-Price Attitude Data

austres Quarterly Time Series of the Number of Australian Residents

beaver1 (beavers) Body Temperature Series of Two Beavers

beaver2 (beavers) Body Temperature Series of Two Beavers

cars Speed and Stopping Distances of Cars

chickwts Chicken Weights by Feed Type

co2 Mauna Loa Atmospheric CO2 Concentration

crimtab Student's 3000 Criminals Data

discoveries Yearly Numbers of Important Discoveries

esoph Smoking, Alcohol and (O)esophageal Cancer

euro Conversion Rates of Euro Currencies

euro.cross (euro) Conversion Rates of Euro Currencies

eurodist Distances Between European Cities and Between US Cities

faithful Old Faithful Geyser Data

fdeaths (UKLungDeaths) Monthly Deaths from Lung Diseases in the UK

freeny Freeny's Revenue Data

freeny.x (freeny) Freeny's Revenue Data

freeny.y (freeny) Freeny's Revenue Data

infert Infertility after Spontaneous and Induced Abortion

iris Edgar Anderson's Iris Data

iris3 Edgar Anderson's Iris Data

islands Areas of the World's Major Landmasses

ldeaths (UKLungDeaths) Monthly Deaths from Lung Diseases in the UK

lh Luteinizing Hormone in Blood Samples

longley Longley's Economic Regression Data

lynx Annual Canadian Lynx trappings 1821-1934

mdeaths (UKLungDeaths) Monthly Deaths from Lung Diseases in the UK

morley Michelson Speed of Light Data

mtcars Motor Trend Car Road Tests

nhtemp Average Yearly Temperatures in New Haven

nottem Average Monthly Temperatures at Nottingham, 1920-1939

npk Classical N, P, K Factorial Experiment

occupationalStatus Occupational Status of Fathers and their Sons

precip Annual Precipitation in US Cities

presidents Quarterly Approval Ratings of US Presidents

pressure Vapor Pressure of Mercury as a Function of Temperature

quakes Locations of Earthquakes off Fiji

randu Random Numbers from Congruential Generator RANDU

rivers Lengths of Major North American Rivers

rock Measurements on Petroleum Rock Samples

sleep Student's Sleep Data

stack.loss (stackloss) Brownlee's Stack Loss Plant Data

stack.x (stackloss) Brownlee's Stack Loss Plant Data

stackloss Brownlee's Stack Loss Plant Data

state.abb (state) US State Facts and Figures

state.area (state) US State Facts and Figures

state.center (state) US State Facts and Figures

state.division (state) US State Facts and Figures

state.name (state) US State Facts and Figures

state.region (state) US State Facts and Figures

state.x77 (state) US State Facts and Figures

sunspot.month Monthly Sunspot Data, from 1749 to "Present"

sunspot.year Yearly Sunspot Data, 1700-1988

sunspots Monthly Sunspot Numbers, 1749-1983

swiss Swiss Fertility and Socioeconomic Indicators (1888) Data

treering Yearly Treering Data, -6000-1979

trees Girth, Height and Volume for Black Cherry Trees

uspop Populations Recorded by the US Census

volcano Topographic Information on Auckland's Maunga Whau Volcano

warpbreaks The Number of Breaks in Yarn during Weaving

women Average Heights and Weights for American Women

**Example**:

> Titanic

, , Age = Child, Survived = No

Sex

Class Male Female

1st 0 0

2nd 0 0

3rd 35 17

Crew 0 0

, , Age = Adult, Survived = No

Sex

Class Male Female

1st 118 4

2nd 154 13

3rd 387 89

Crew 670 3

, , Age = Child, Survived = Yes

Sex

Class Male Female

1st 5 1

2nd 11 13

3rd 13 14

Crew 0 0

, , Age = Adult, Survived = Yes

Sex

Class Male Female

1st 57 140

2nd 14 80

3rd 75 76

Crew 192 20

**Assignments**:

Check some datasets to the R console just by typing the names.

NB: you can see all the data types just by typing data() in the console.

If you want to print a certain amount of data(here 5 columns), you can do like the following thing.

> head(CO2, 5)

Plant Type Treatment conc uptake

1 Qn1 Quebec nonchilled 95 16.0

2 Qn1 Quebec nonchilled 175 30.4

3 Qn1 Quebec nonchilled 250 34.8

4 Qn1 Quebec nonchilled 350 37.2

5 Qn1 Quebec nonchilled 500 35.3

Here 5 is the number of rows of the dataset.

To get how many columns and rows are there in the dataset, you can use the ncol(Dataset\_name) and nrow(Dataset\_name) functions respectively.

Only head() return 5/6 rows.

**Example**:

> nrow(CO2)

[1] 84

> ncol(CO2)

[1] 5

**Loading a dataframe**

To load a dataframe with .Rda extension or .Rdata extension, we can use the following command.

load(data\_frame\_name.Rda)

If you use the .Rda dataframe a lot in R, you can also save the data file into R as follow.

save(list='mydata',file="data\_frame\_name.Rda")

after this command, you will have a dataframe mydata in R.

**Building a dataframe**

We can build a dataframe using vector.

N <- 100

u <- rnorm(N)

x1 <- rnorm(N)

x2 <- rnorm(N)

y <- 1 + x1 + x2 + u

mydat <- data.frame(y,x1,x2)

here rnorm(n, mean = , sd = ) is used to generate n normal random numbers with arguments mean and sd; while runif(n, min = , max = ) is used to generate n uniform random numbers lie in the interval (min, max).

**Assignment:**

Create a dtaframe using the following data. (hint: use data.frame() function)

> employee <- c('John Doe','Peter Gynn','Jolie Hope')

> salary <- c(21000, 23400, 26800)

> startdate <- as.Date(c('2010-11-1','2008-3-25','2007-3-14'))

Plot a graph for salary. Find the average salary, maximum and minimum salary.

**Types of Graphs in R**

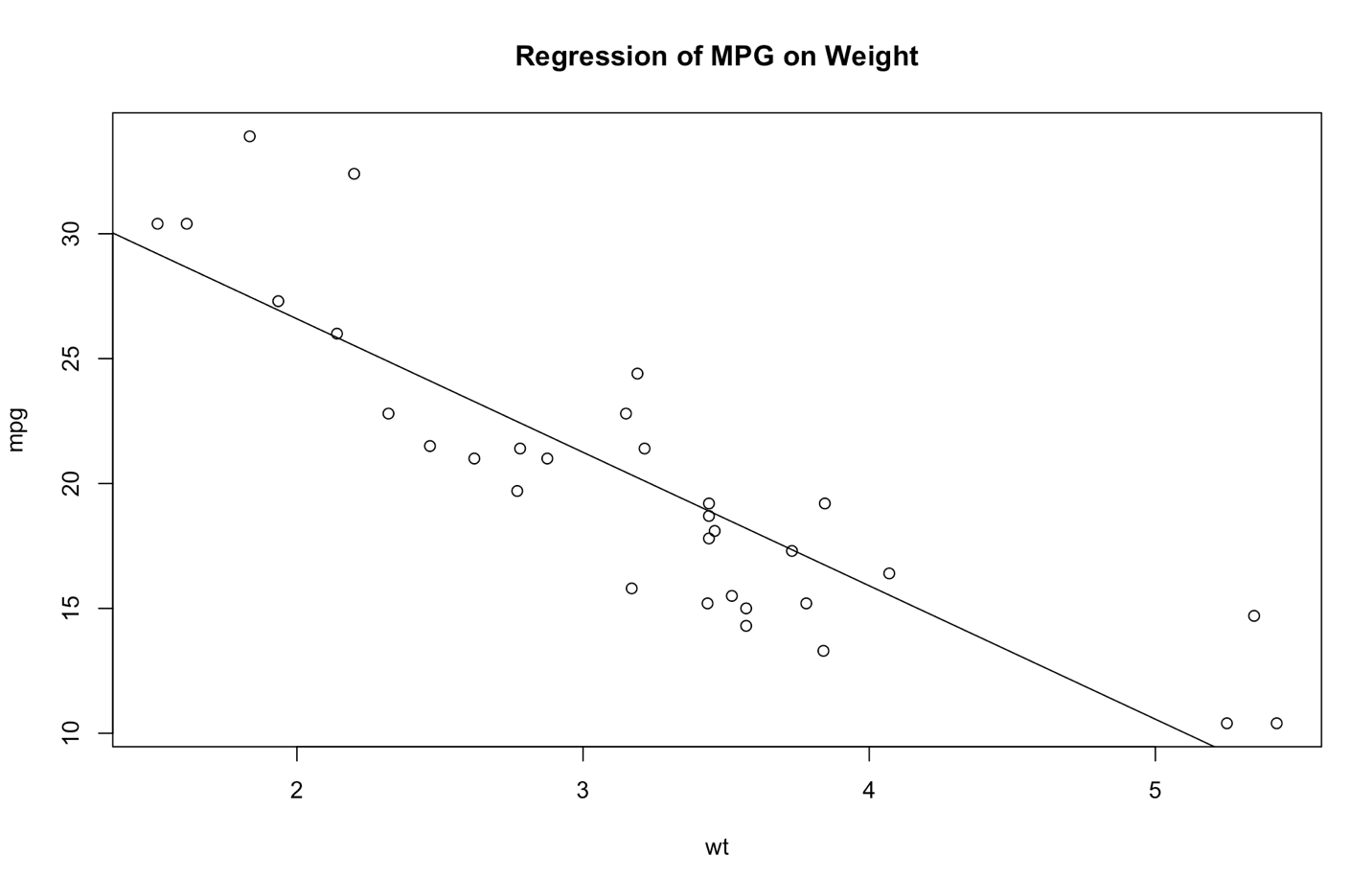
**1. Simple plotting**

> attach(mtcars) // setting latest/real time dataframe

> plot(wt, mpg) //form the same dataset

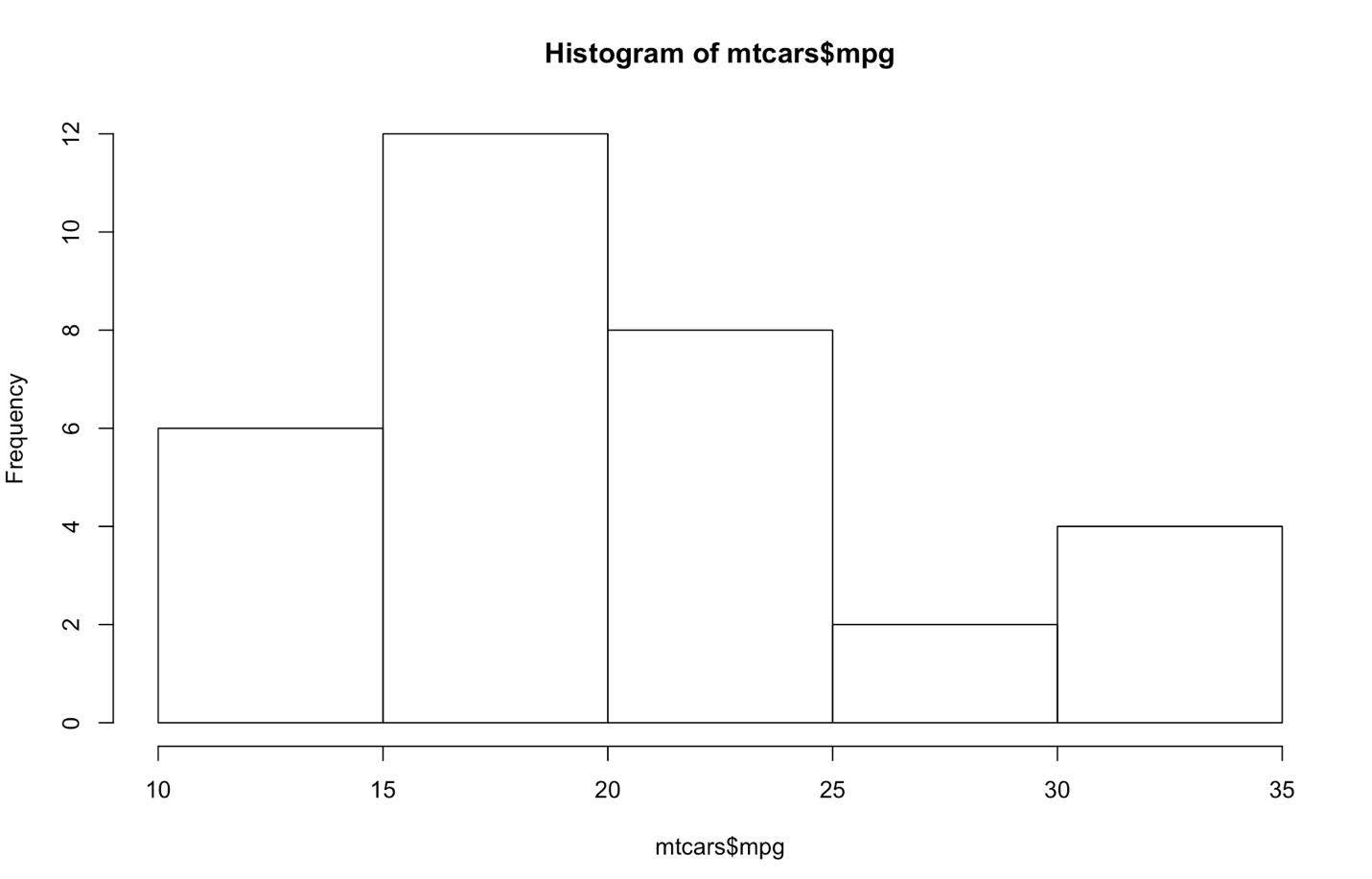
> abline(lm(mpg~wt)) //y and x axix respectively

> title("Regression of MPG on Weight") //top title.

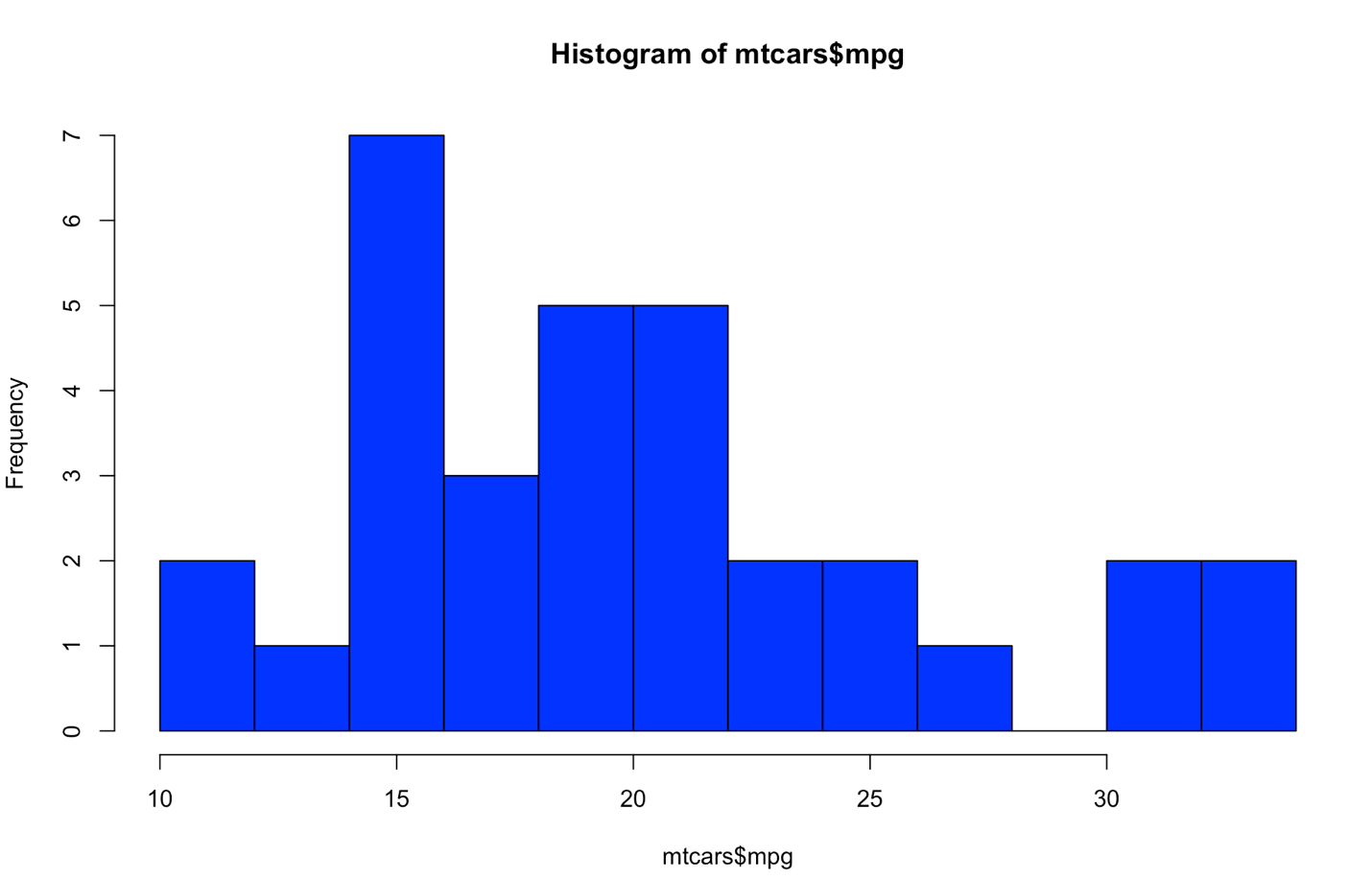


**2. Histogram**

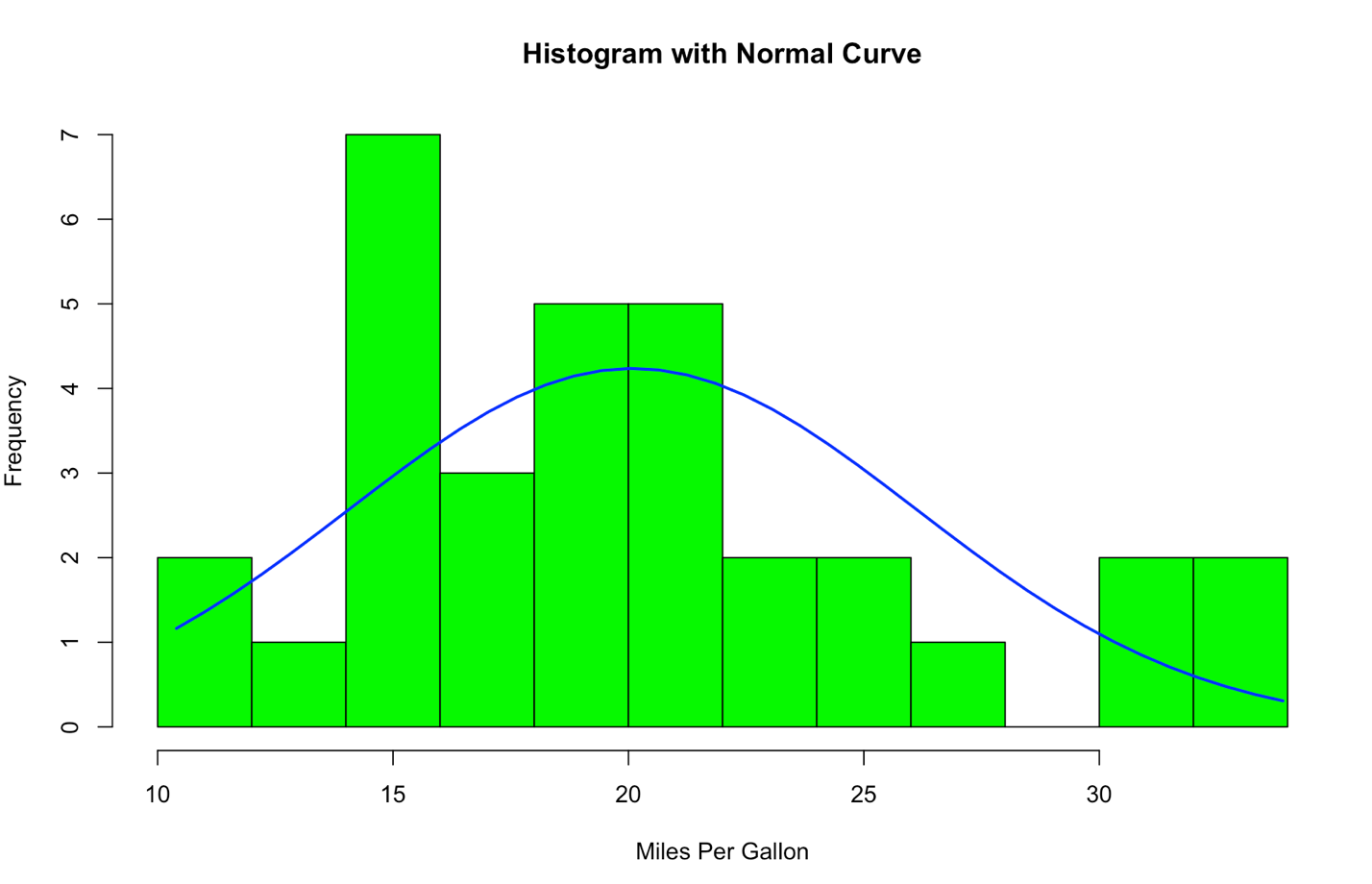
hist(mtcars$mpg) //from mtcars dataframe, mpg’s histogram

****

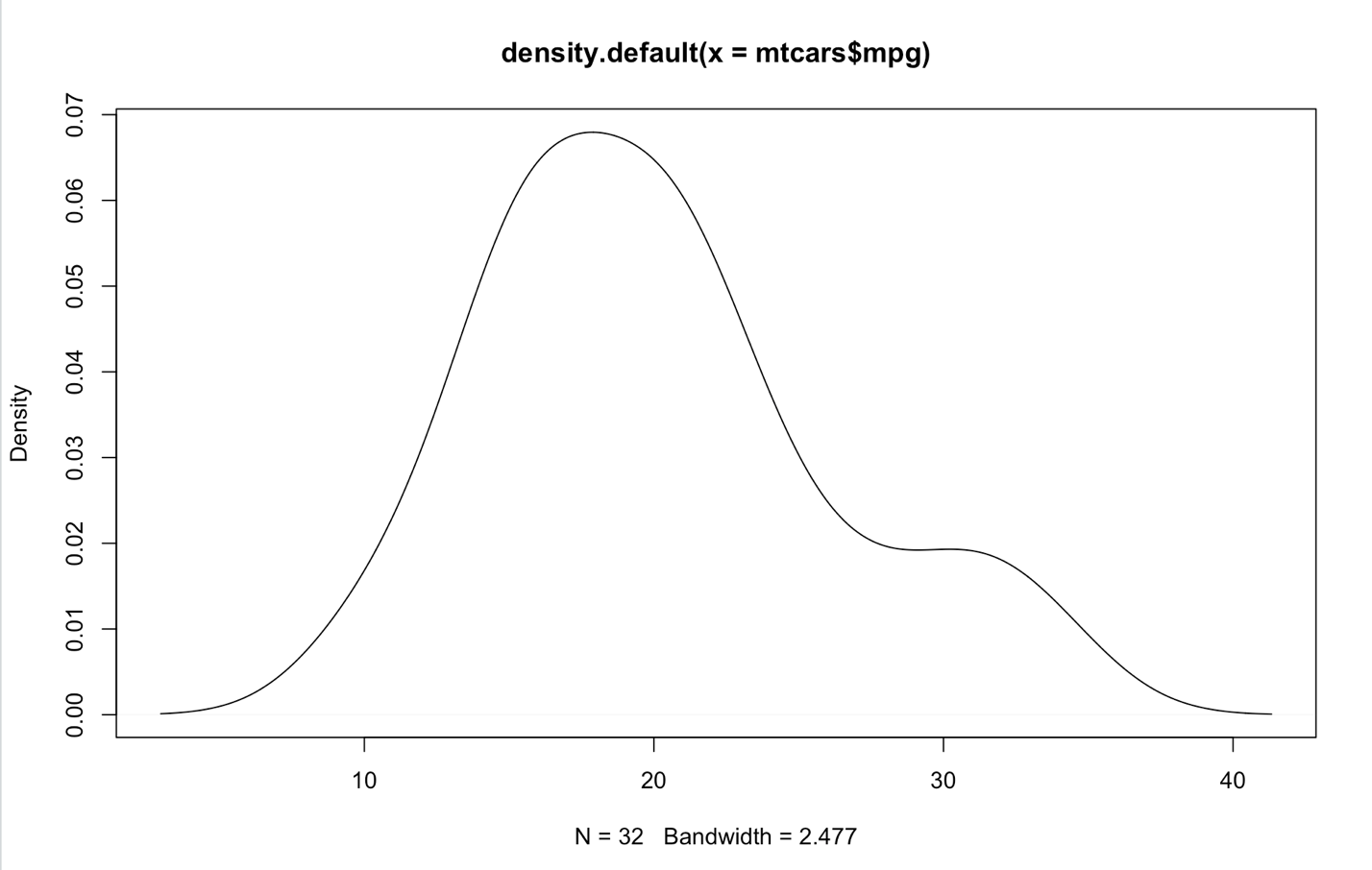
# Colored Histogram with Different Number of Bins  
hist(mtcars$mpg, breaks=12, col="blue")

****

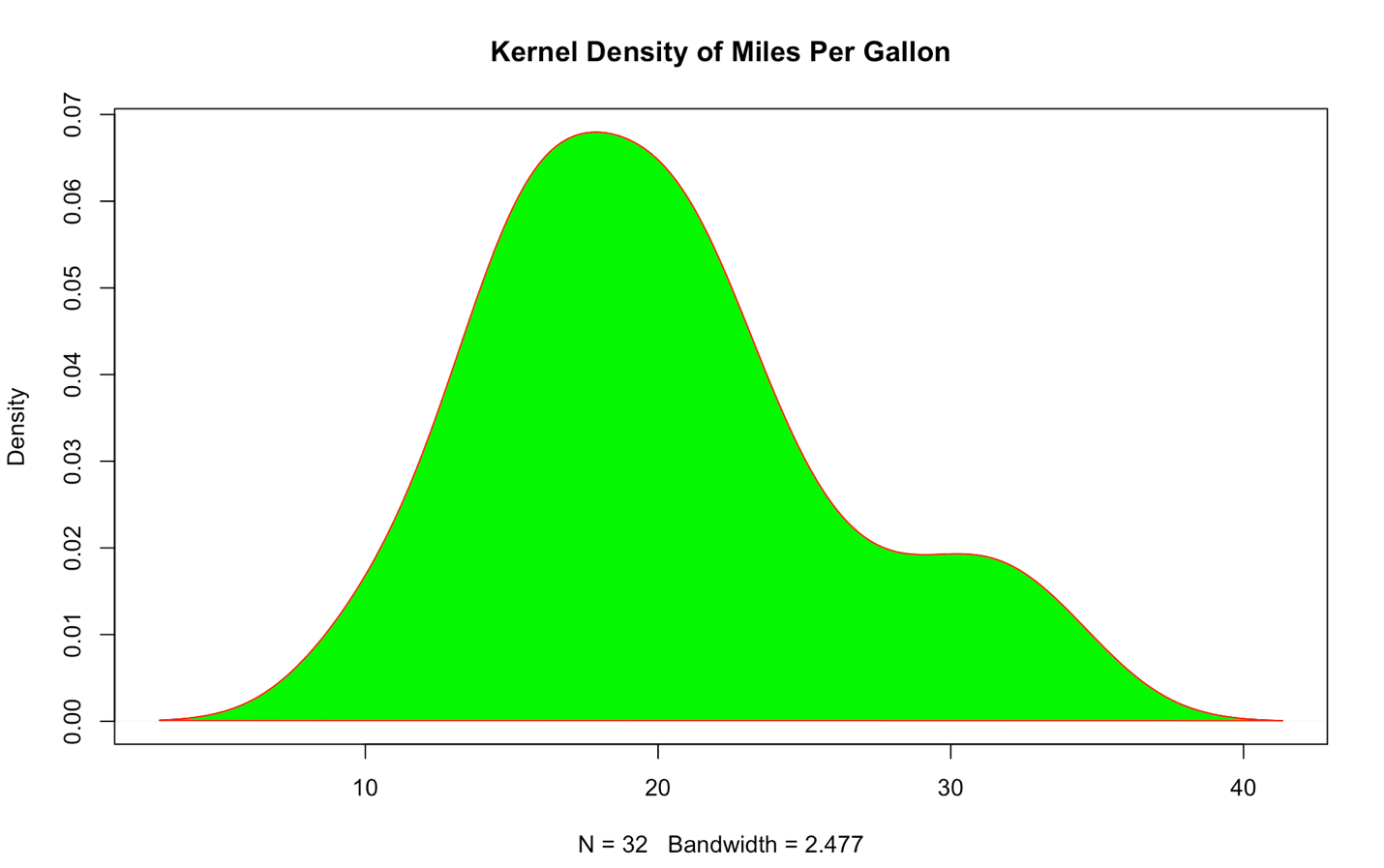
# Add a Normal Curve   
x <- mtcars$mpg   
h<-hist(x, breaks=10, col="green", xlab="Miles Per Gallon",   
   main="Histogram with Normal Curve")   
xfit<-seq(min(x),max(x),length=40)   
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))   
yfit <- yfit\*diff(h$mids[1:2])\*length(x)   
lines(xfit, yfit, col="blue", lwd=2)

****

# Kernel Density Plot  
d <- density(mtcars$mpg) # returns the density data   
plot(d) # plots the results

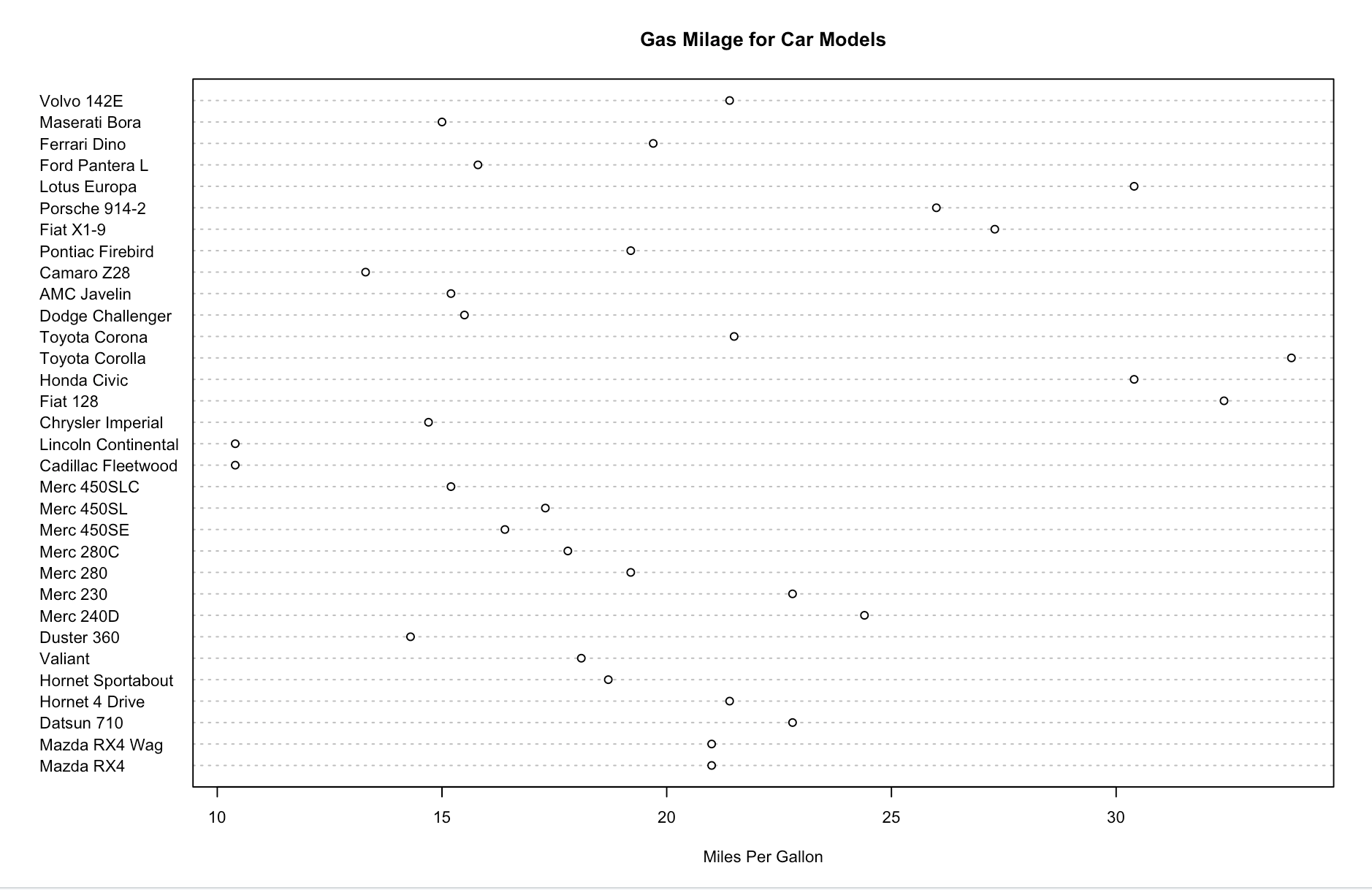
****

# Filled Density Plot  
d <- density(mtcars$mpg)  
plot(d, main="Kernel Density of Miles Per Gallon")  
polygon(d, col="green", border="red")

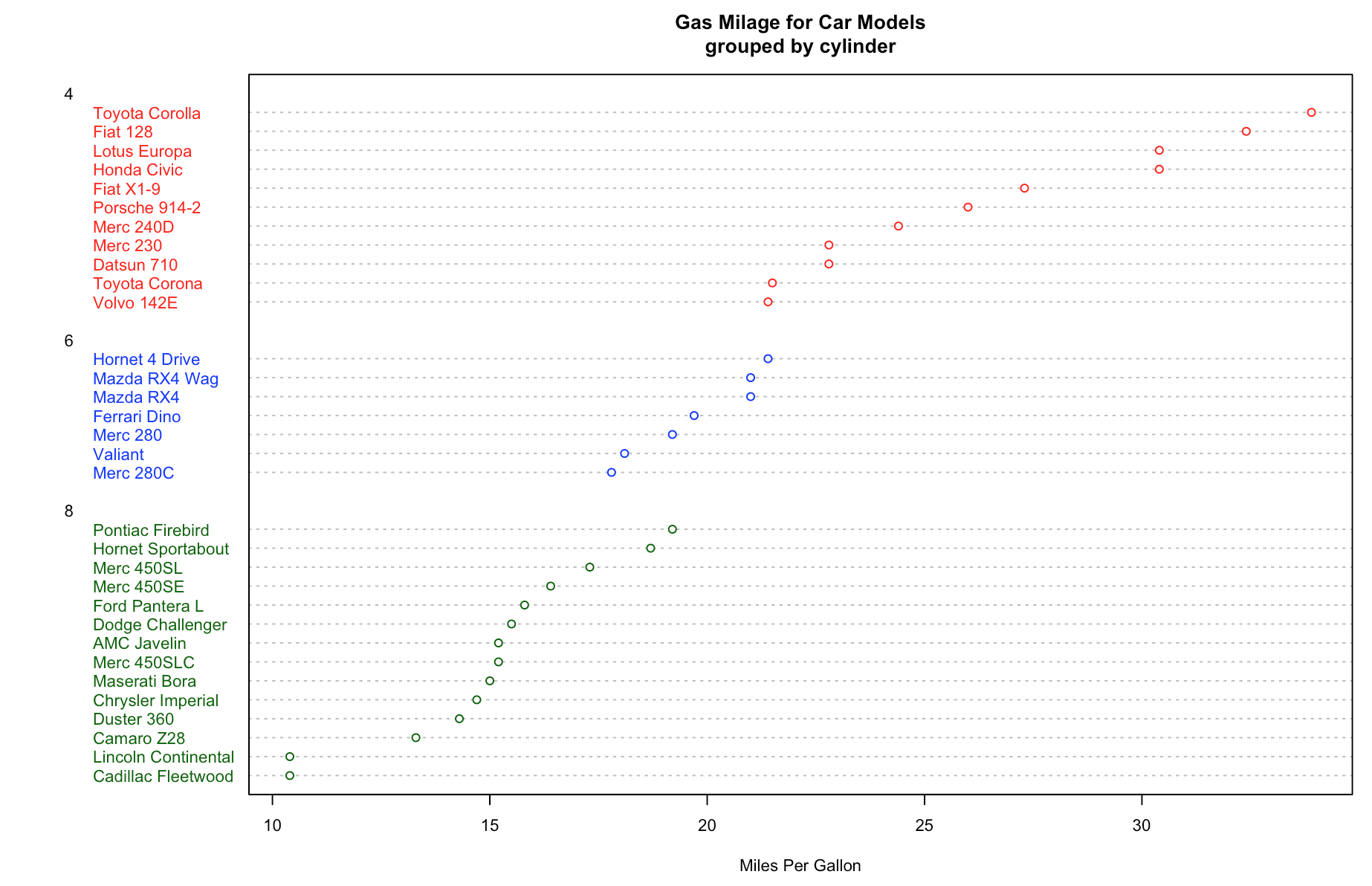
****

**3. Dot Plot**

# Simple Dotplot  
dotchart(mtcars$mpg,labels=row.names(mtcars),cex=.7,  
   main="Gas Milage for Car Models",   
   xlab="Miles Per Gallon")

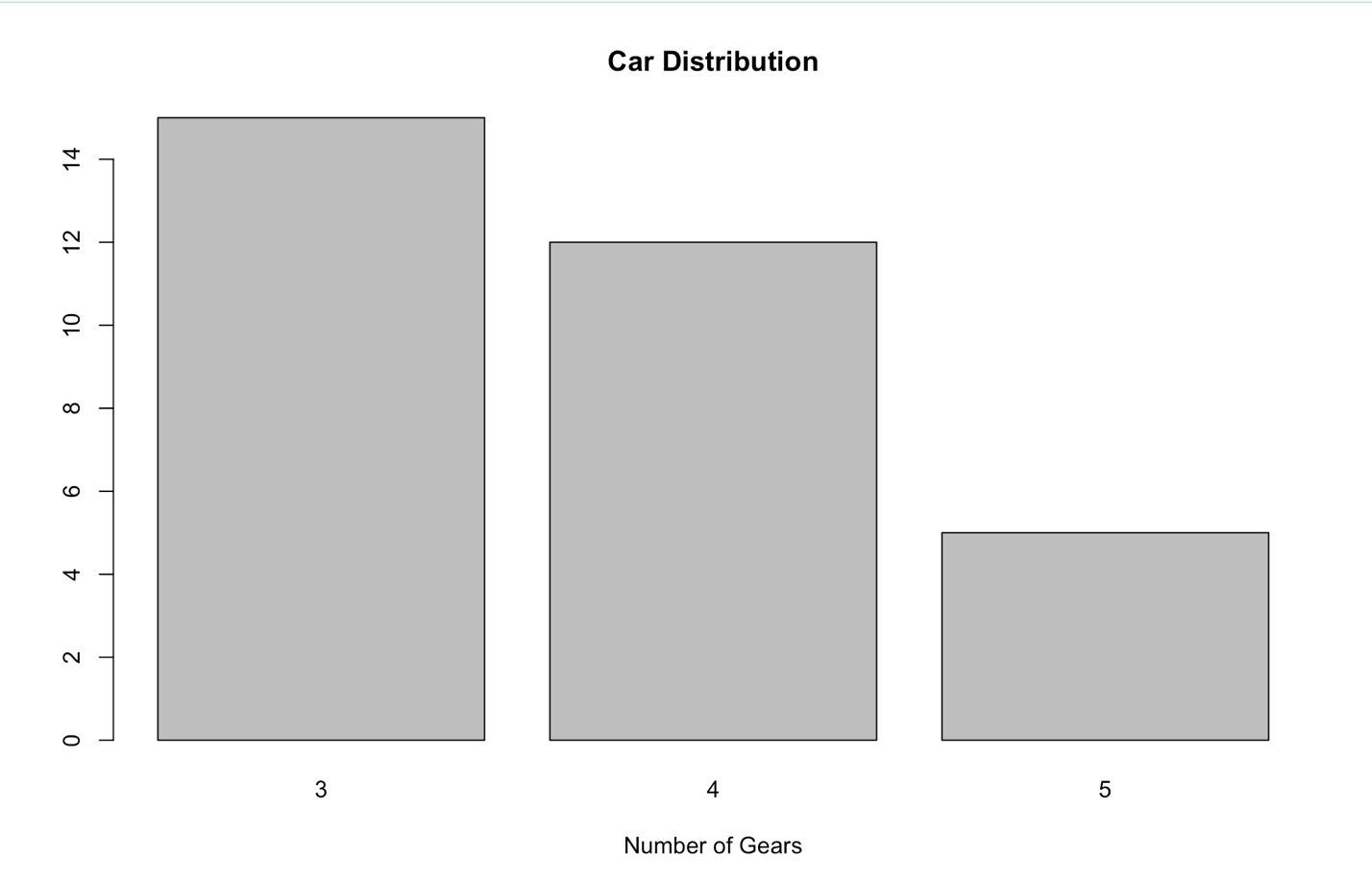


# Dotplot: Grouped Sorted and Colored  
# Sort by mpg, group and color by cylinder   
x <- mtcars[order(mtcars$mpg),] # sort by mpg  
x$cyl <- factor(x$cyl) # it must be a factor  
x$color[x$cyl==4] <- "red"  
x$color[x$cyl==6] <- "blue"  
x$color[x$cyl==8] <- "darkgreen"   
dotchart(x$mpg,labels=row.names(x),cex=.7,groups= x$cyl,  
   main="Gas Milage for Car Models\ngrouped by cylinder",  
   xlab="Miles Per Gallon", gcolor="black", color=x$color)

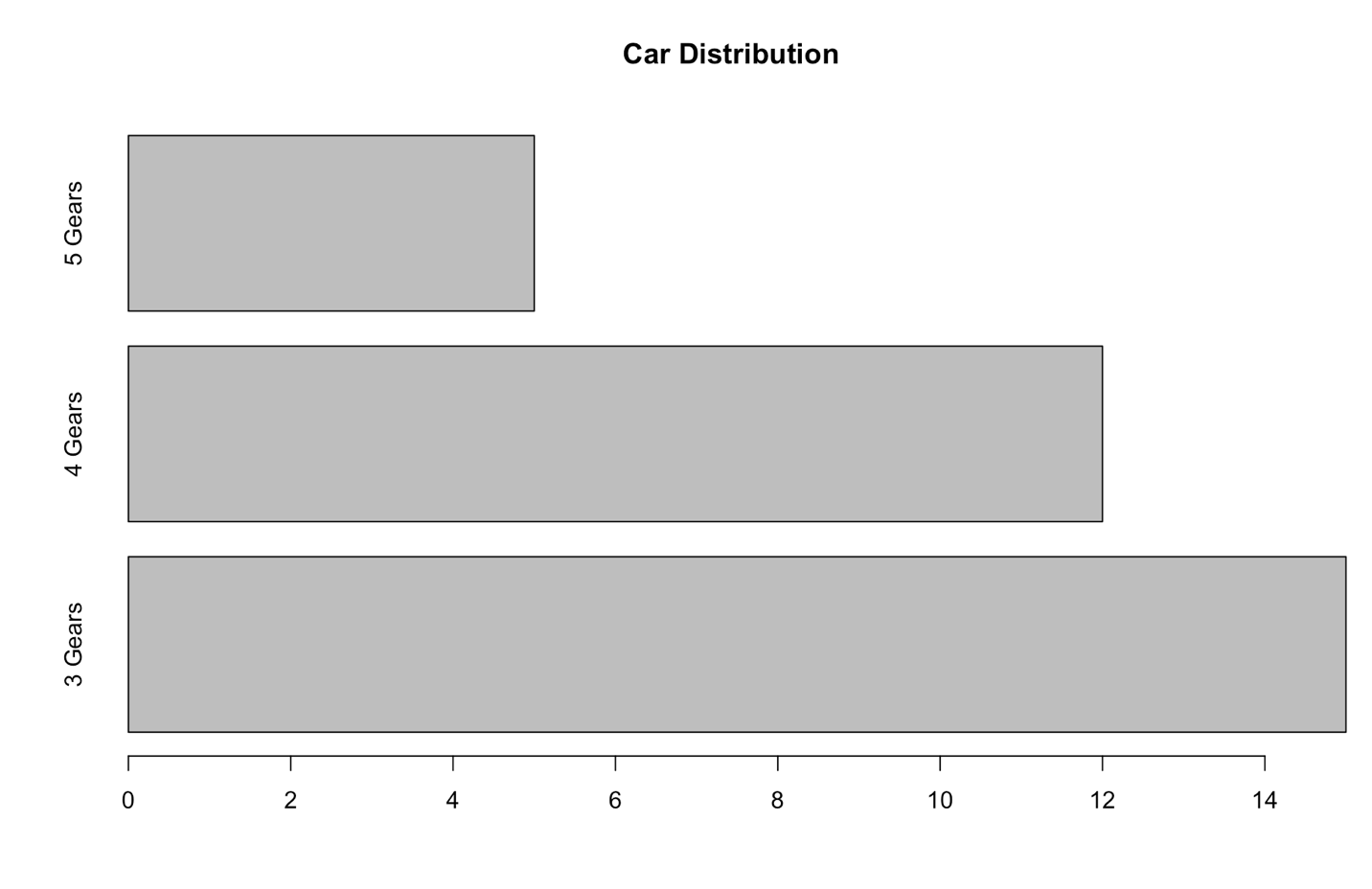


4**. Bar Plot**

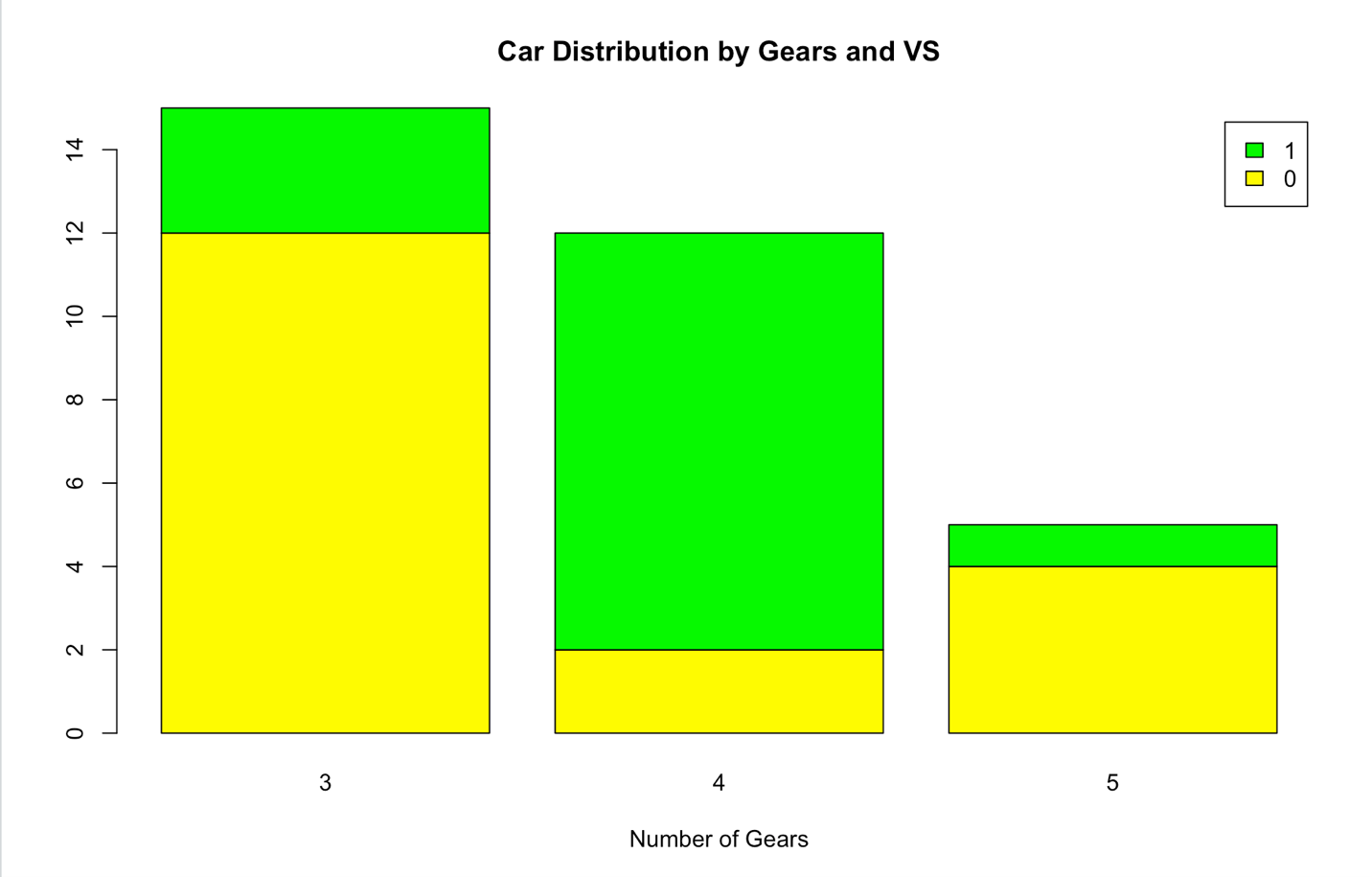
# Simple Bar Plot   
counts <- table(mtcars$gear)  
barplot(counts, main="Car Distribution",   
   xlab="Number of Gears")



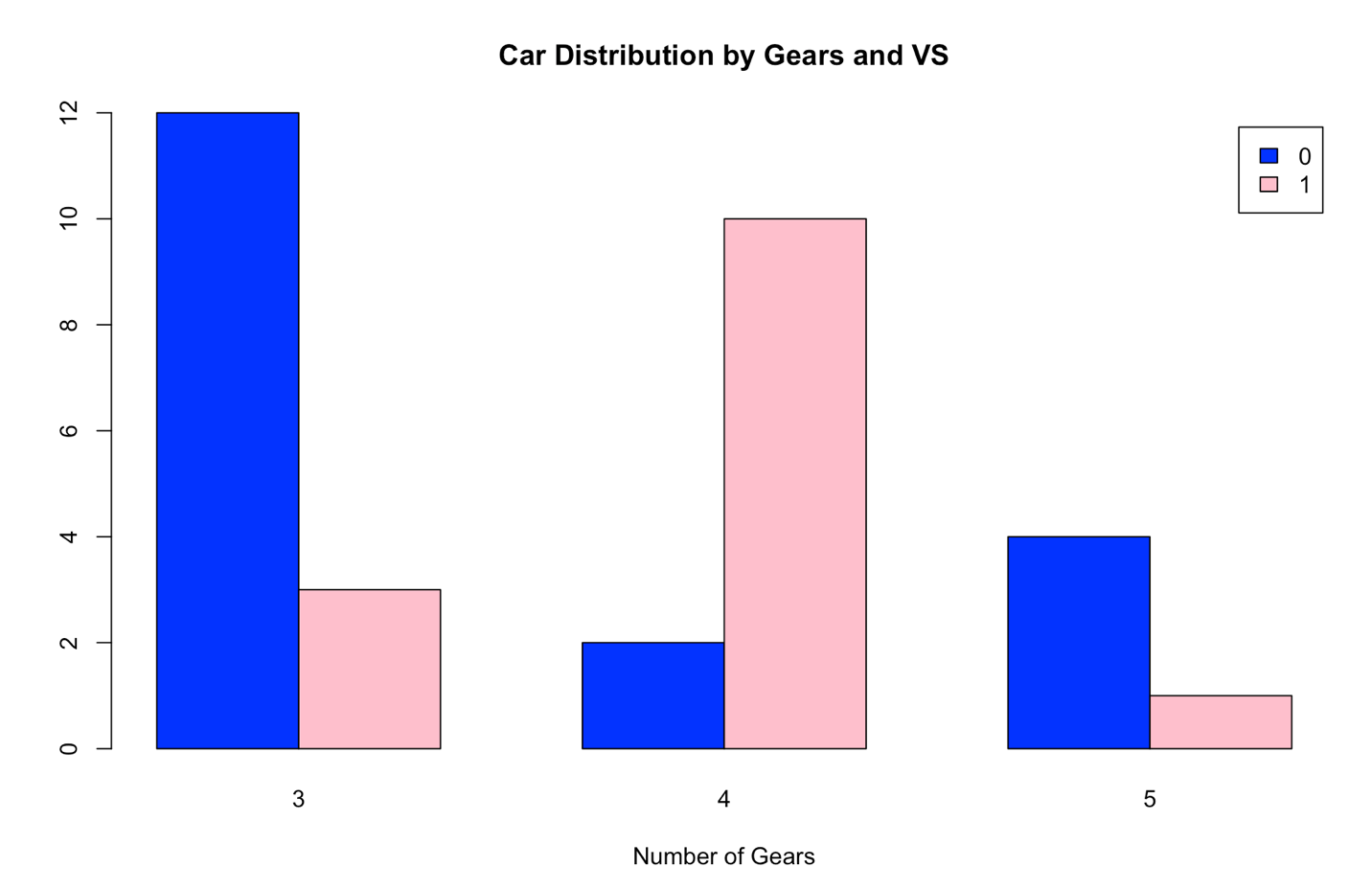
# Simple Horizontal Bar Plot with Added Labels   
counts <- table(mtcars$gear)  
barplot(counts, main="Car Distribution", horiz=TRUE,  
  names.arg=c("3 Gears", "4 Gears", "5 Gears"))



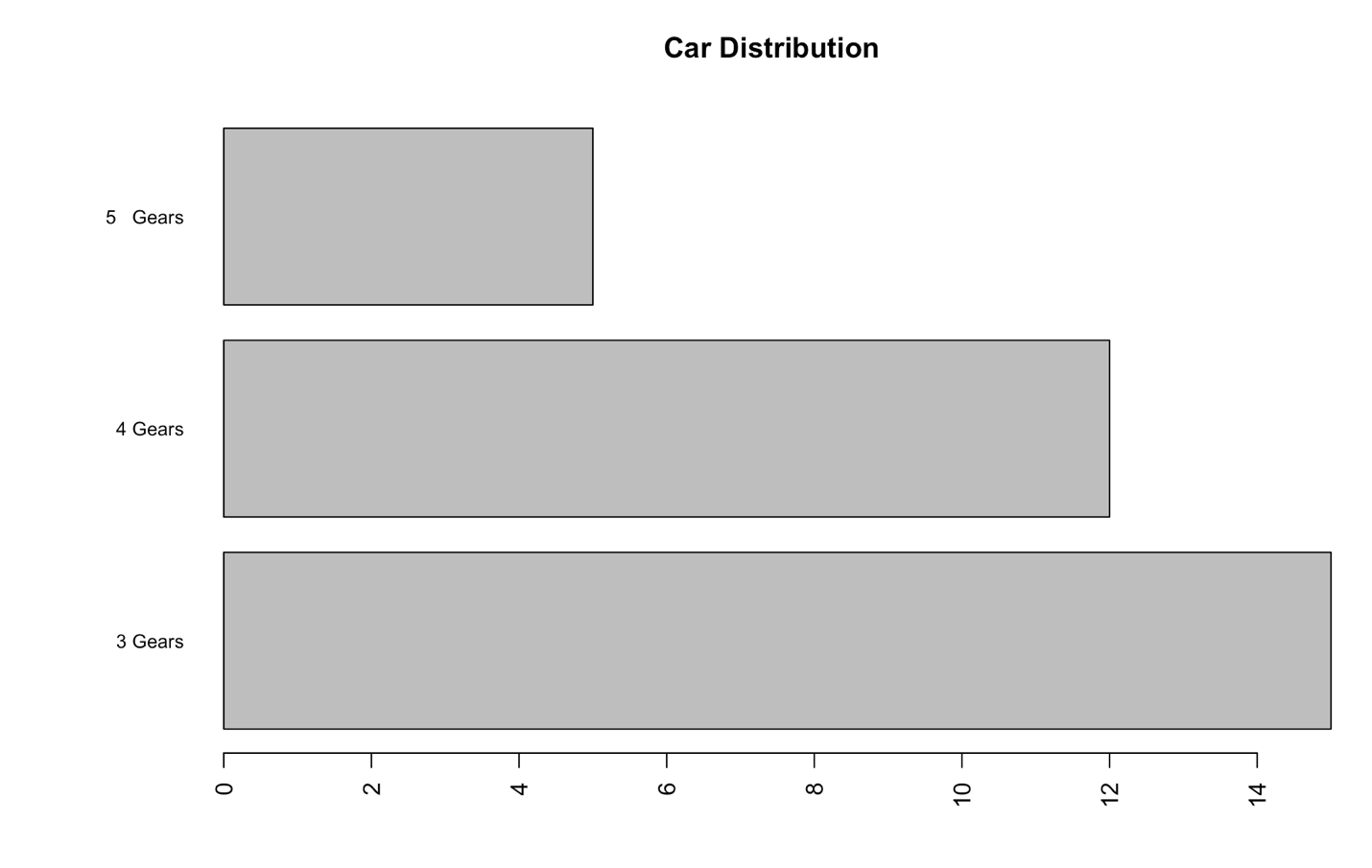
# Stacked Bar Plot with Colors and Legend  
counts <- table(mtcars$vs, mtcars$gear)  
barplot(counts, main="Car Distribution by Gears and VS",  
  xlab="Number of Gears", col=c("yellow","green"),  
  legend = rownames(counts))



# Grouped Bar Plot  
counts <- table(mtcars$vs, mtcars$gear)  
barplot(counts, main="Car Distribution by Gears and VS",  
  xlab="Number of Gears", col=c("blue","pink"),  
  legend = rownames(counts), beside=TRUE)



# Fitting Labels   
par(las=2) # make label text perpendicular to axis  
par(mar=c(5,8,4,2)) # increase y-axis margin.  
  
counts <- table(mtcars$gear)  
barplot(counts, main="Car Distribution", horiz=TRUE, names.arg=c("3 Gears", "4 Gears", "5   Gears"), cex.names=0.8)



5. Line Chart

Line charts are created with the function lines(*x*,*y*,type**=**) where *x* and *y* are numeric vectors of (x,y) points to connect. type= can take the following values:

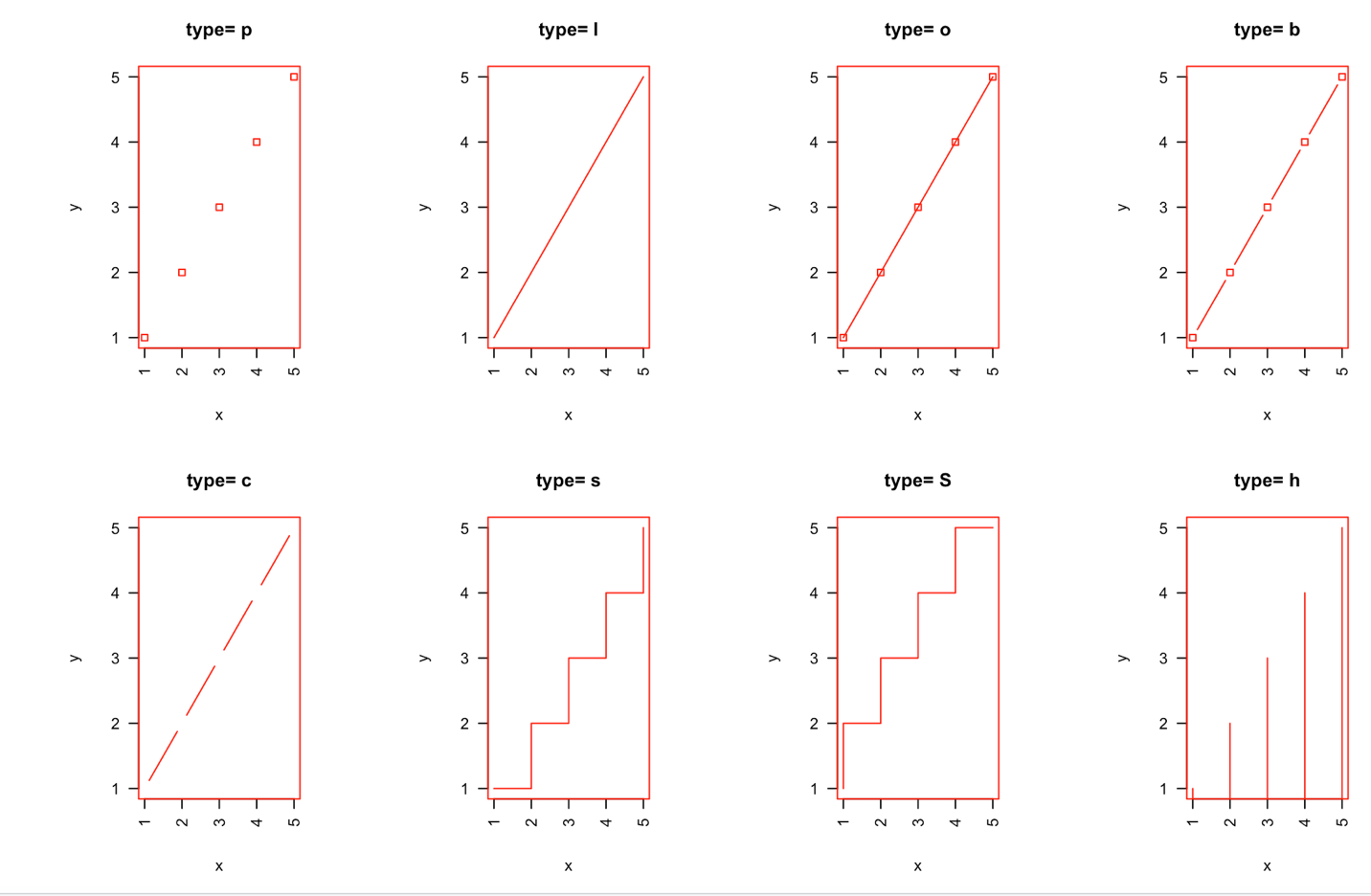
|  |  |
| --- | --- |
| **type** | **description** |
| **p** | points |
| **l** | lines |
| **o** | overplotted points and lines |
| **b, c** | points (empty if "c") joined by lines |
| **s, S** | stair steps |
| **h** | histogram-like vertical lines |
| **n** | does not produce any points or lines |

The lines() function *adds* information to a graph. It cannot produce a graph on its own. Usually it follows a plot(*x*,*y*) command that produces a graph.

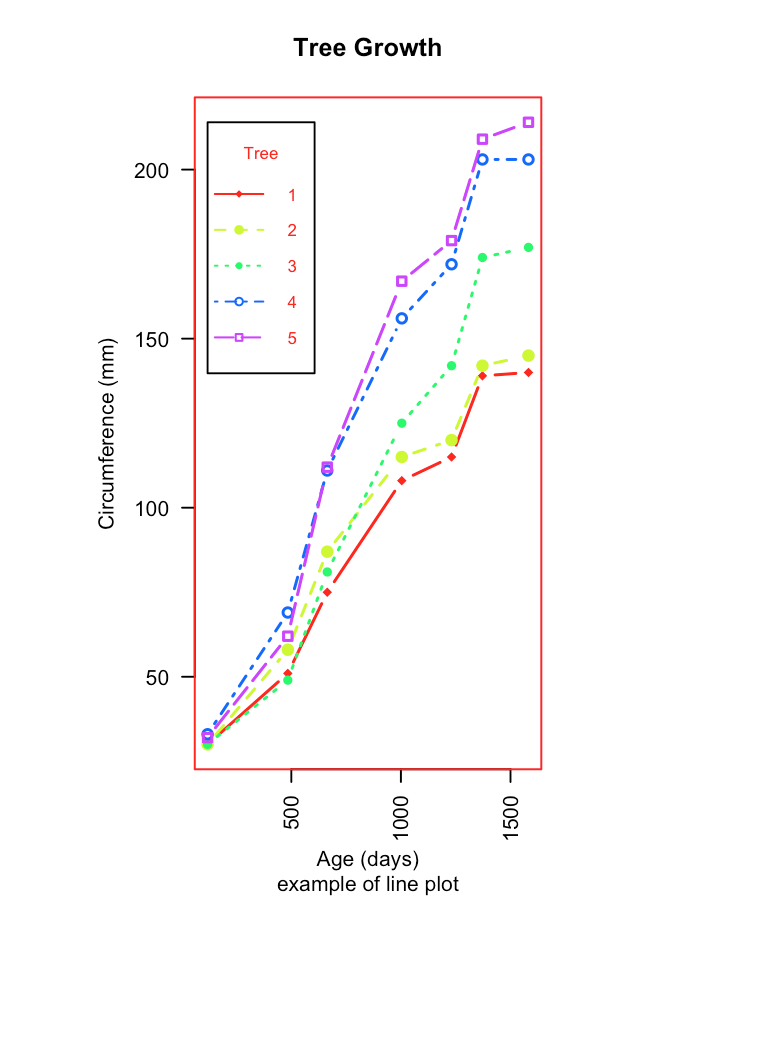
By default, plot()plots the (x,y) points. Use the type="n" option in the plot() command, to create the graph with axes, titles, etc., but *without* plotting the points.

**Example**

x <- c(1:5); y <- x # create some data   
par(pch=22, col="red") # plotting symbol and color   
par(mfrow=c(2,4)) # all plots on one page   
opts = c("p","l","o","b","c","s","S","h")   
for(i in 1:length(opts)){   
  heading = paste("type=",opts[i])   
  plot(x, y, type="n", main=heading)   
  lines(x, y, type=opts[i])   
}

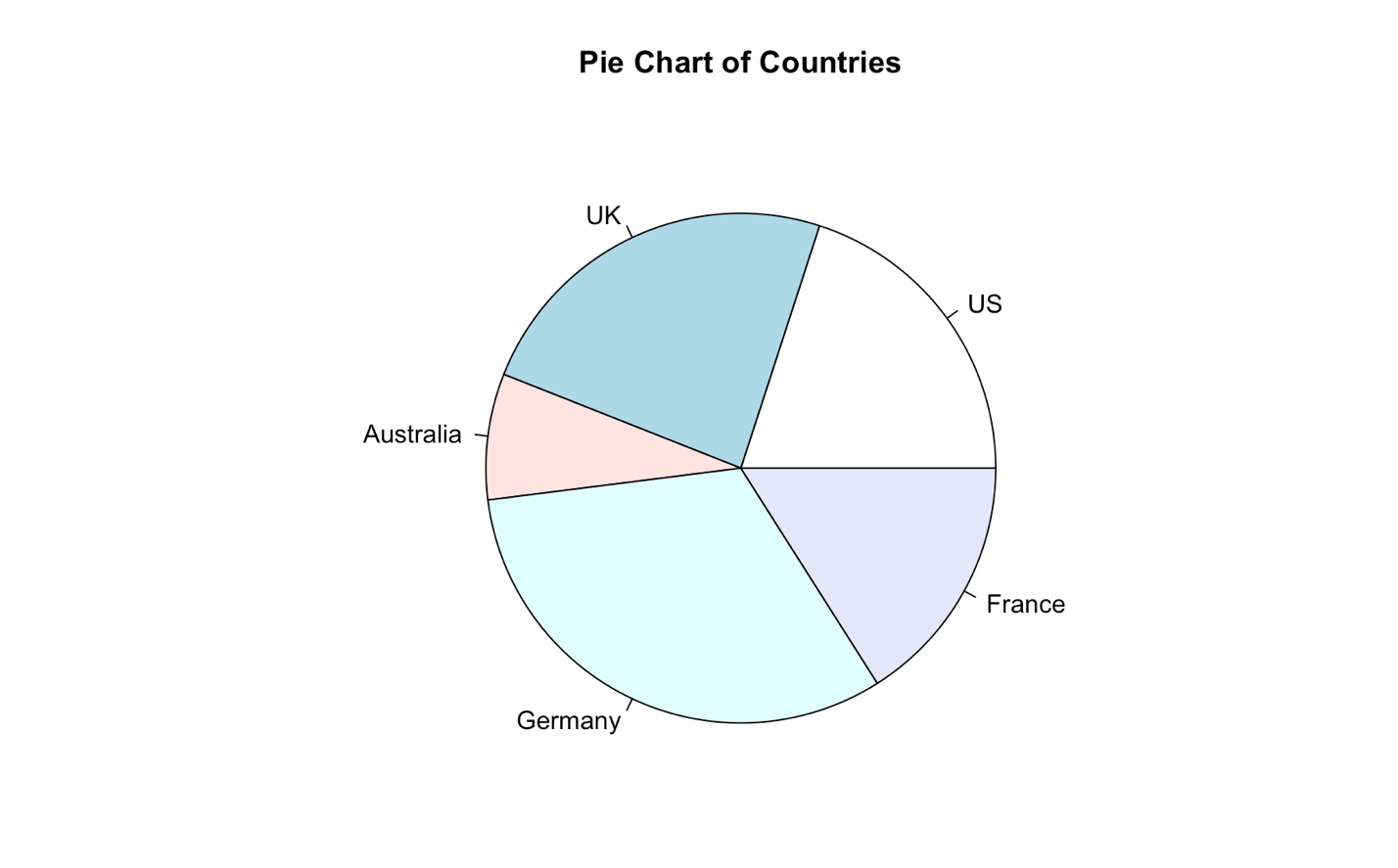


# Create Line Chart  
  
# convert factor to numeric for convenience   
Orange$Tree <- as.numeric(Orange$Tree)   
ntrees <- max(Orange$Tree)  
  
# get the range for the x and y axis   
xrange <- range(Orange$age)   
yrange <- range(Orange$circumference)   
  
# set up the plot   
plot(xrange, yrange, type="n", xlab="Age (days)",  
   ylab="Circumference (mm)" )   
colors <- rainbow(ntrees)   
linetype <- c(1:ntrees)   
plotchar <- seq(18,18+ntrees,1)  
  
# add lines   
for (i in 1:ntrees) {   
  tree <- subset(Orange, Tree==i)   
  lines(tree$age, tree$circumference, type="b", lwd=1.5,  
    lty=linetype[i], col=colors[i], pch=plotchar[i])   
}   
  
# add a title and subtitle   
title("Tree Growth", "example of line plot")  
  
# add a legend   
legend(xrange[1], yrange[2], 1:ntrees, cex=0.8, col=colors,  
   pch=plotchar, lty=linetype, title="Tree")

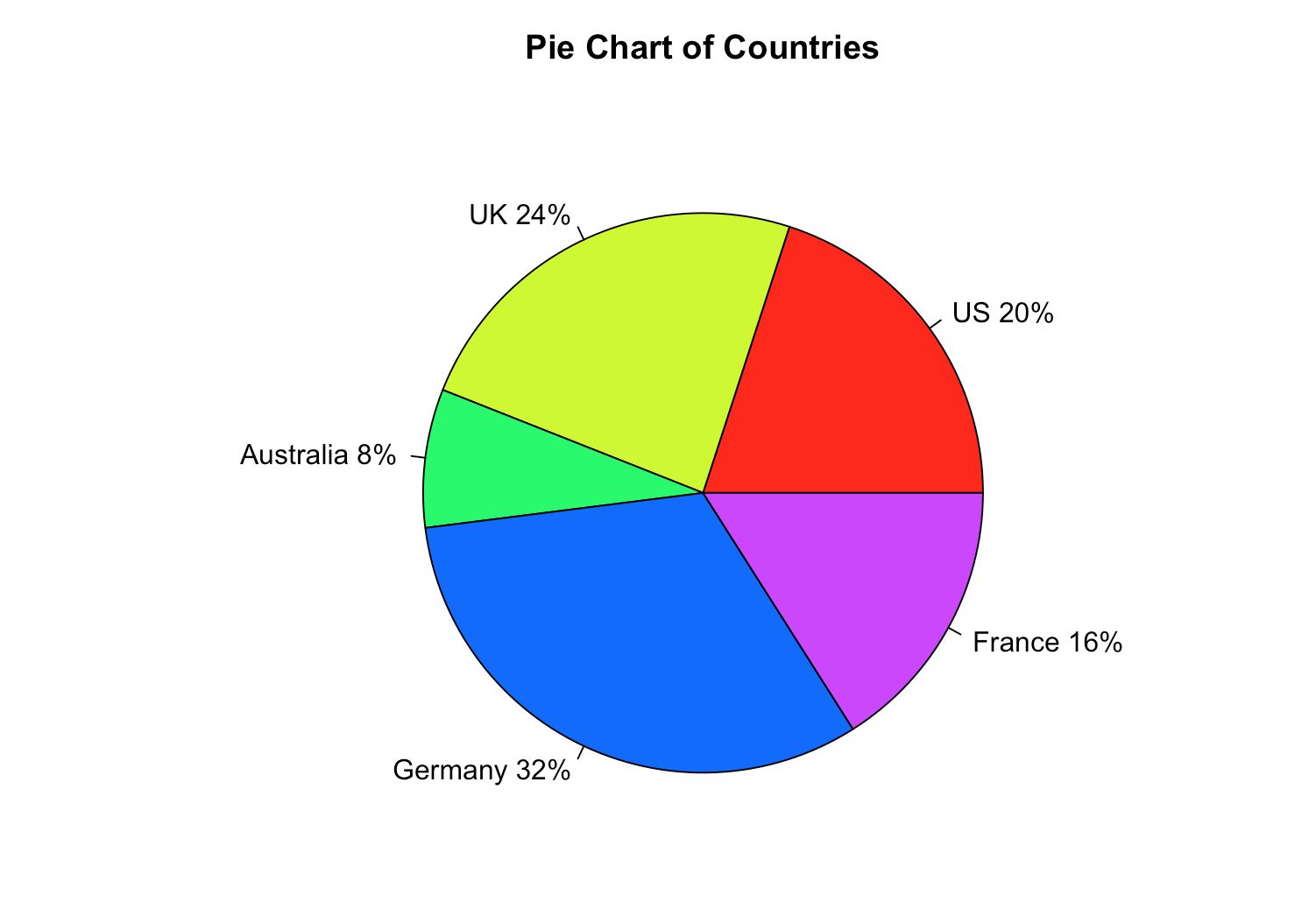


1. **Pie Chart**

# Simple Pie Chart  
slices <- c(10, 12,4, 16, 8)  
lbls <- c("US", "UK", "Australia", "Germany", "France")  
pie(slices, labels = lbls, main="Pie Chart of Countries")



# Pie Chart with Percentages  
slices <- c(10, 12, 4, 16, 8)   
lbls <- c("US", "UK", "Australia", "Germany", "France")  
pct <- round(slices/sum(slices)\*100)  
lbls <- paste(lbls, pct) # add percents to labels   
lbls <- paste(lbls,"%",sep="") # ad % to labels   
pie(slices,labels = lbls, col=rainbow(length(lbls)),  
   main="Pie Chart of Countries")



# 3D Exploded Pie Chart

>install.packages("plotrix")

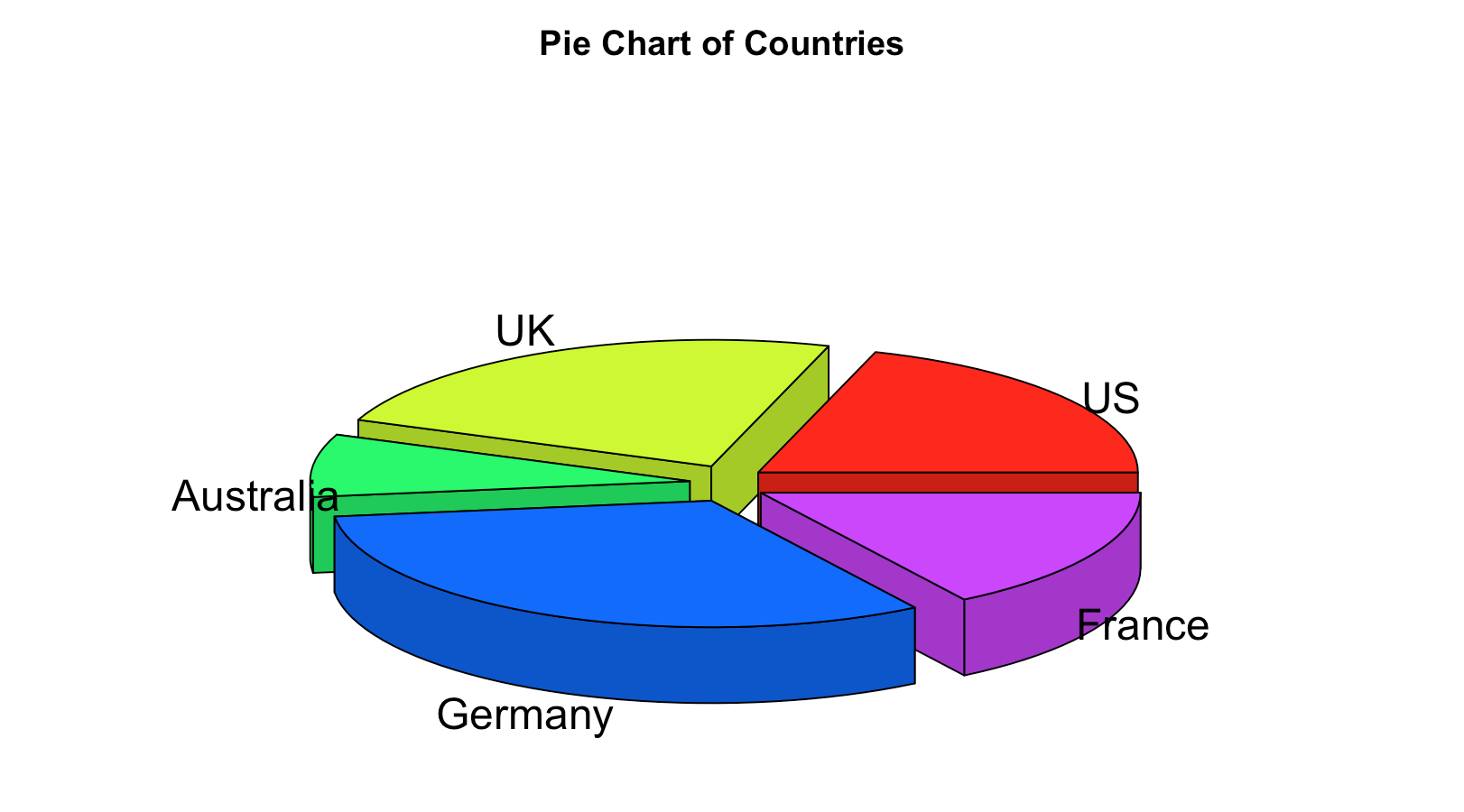
> library(plotrix)

> slices <- c(10, 12, 4, 16, 8)

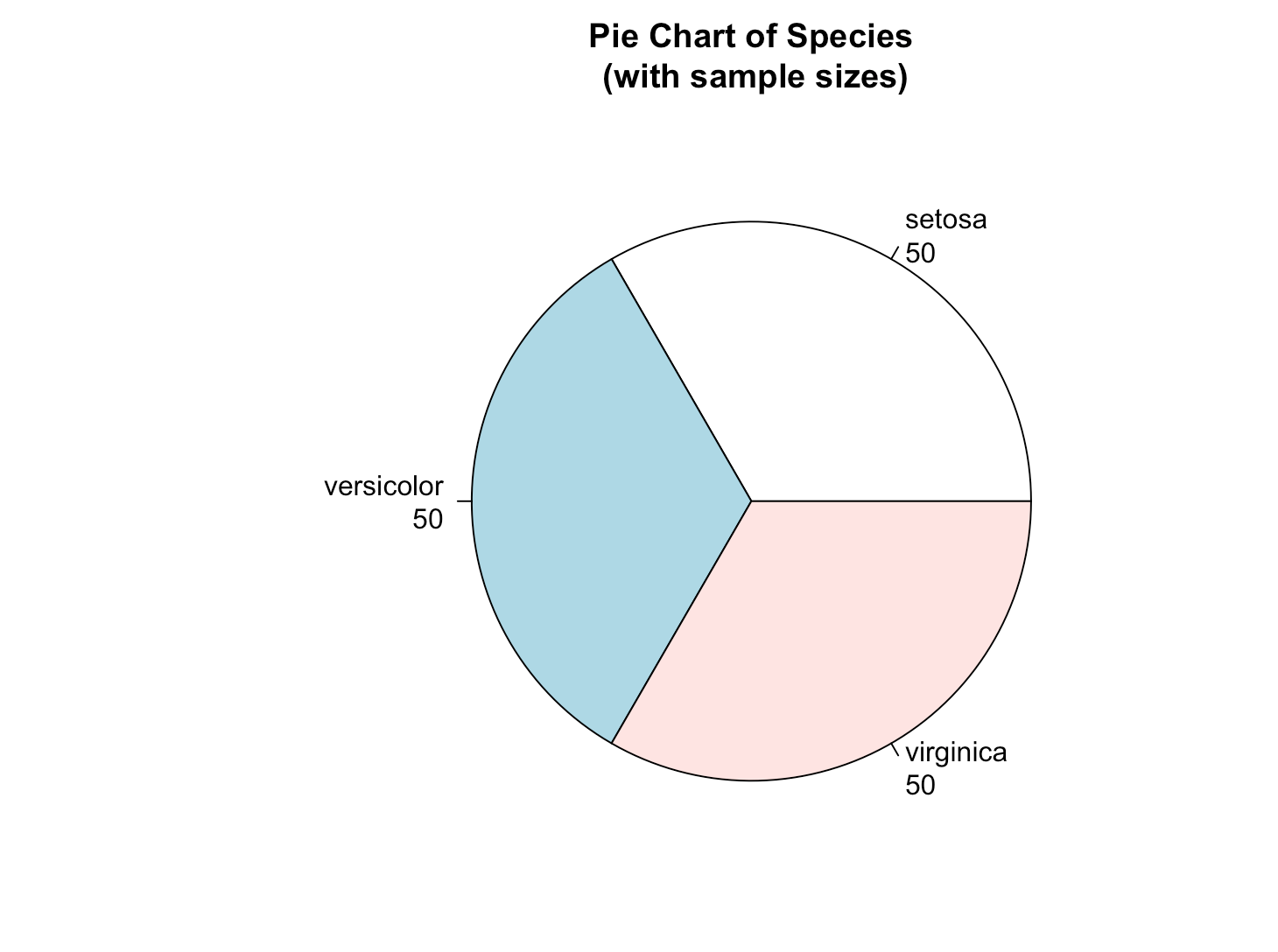
> lbls <- c("US", "UK", "Australia", "Germany", "France")

> pie3D(slices,labels=lbls,explode=0.1,

+ main="Pie Chart of Countries ")

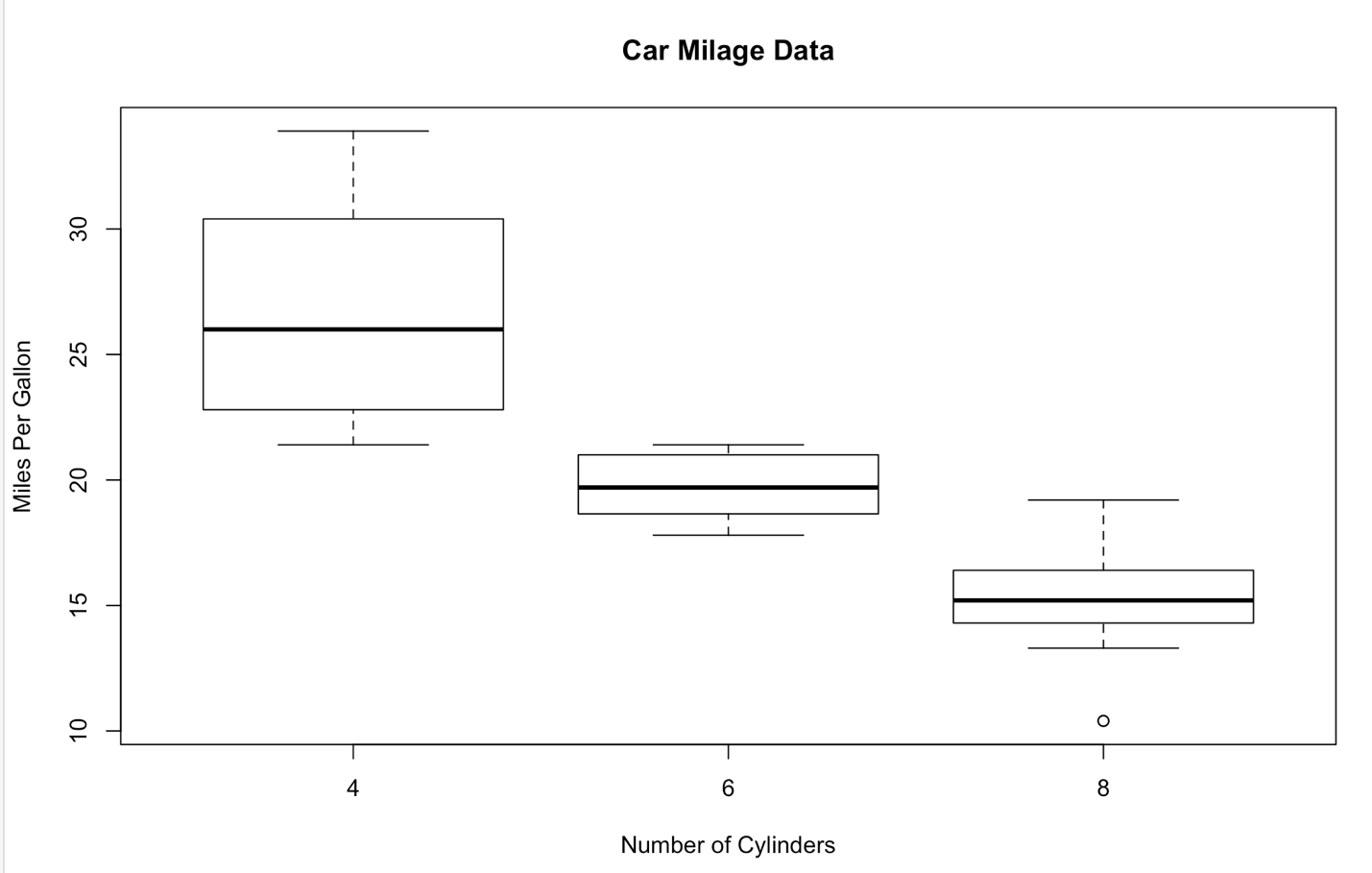


# Pie Chart from data frame with Appended Sample Sizes  
mytable <- table(iris$Species)  
lbls <- paste(names(mytable), "\n", mytable, sep="")  
pie(mytable, labels = lbls,   
   main="Pie Chart of Species\n (with sample sizes)")

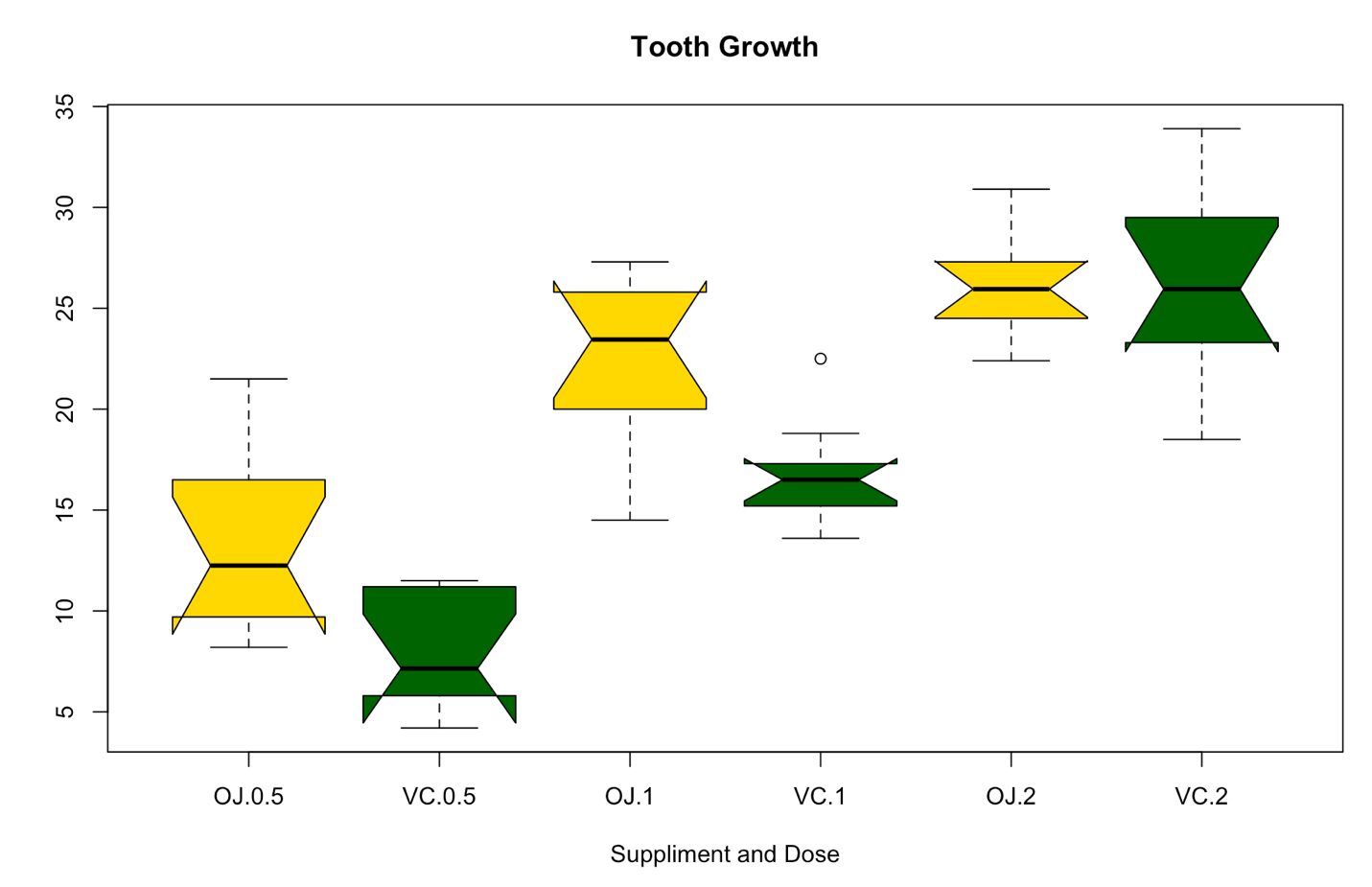


1. **BoxPlot**

# Boxplot of MPG by Car Cylinders   
boxplot(mpg~cyl,data=mtcars, main="Car Milage Data",   
   xlab="Number of Cylinders", ylab="Miles Per Gallon")

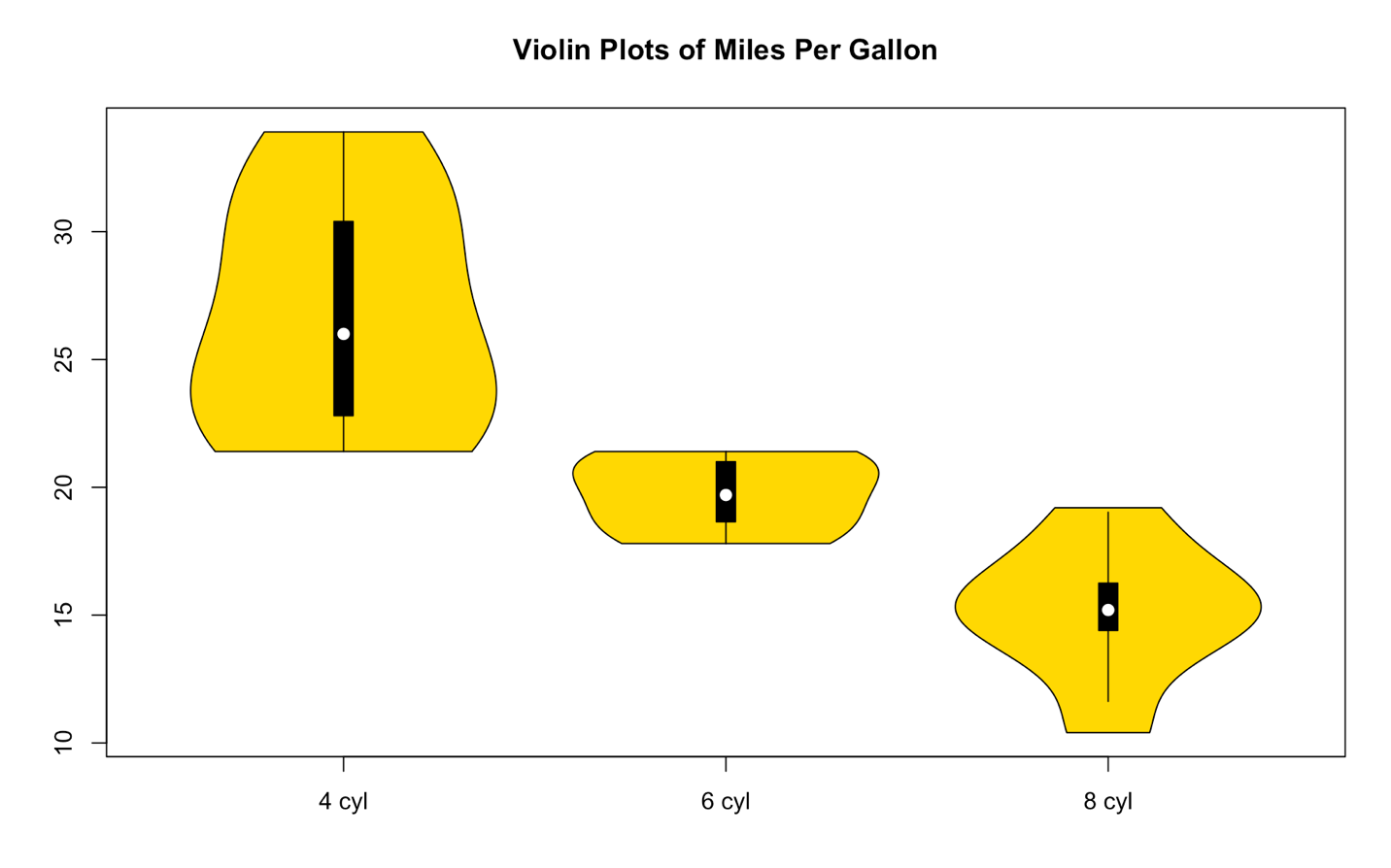


# Notched Boxplot of Tooth Growth Against 2 Crossed Factors  
# boxes colored for ease of interpretation   
boxplot(len~supp\*dose, data=ToothGrowth, notch=TRUE,   
  col=(c("gold","darkgreen")),  
  main="Tooth Growth", xlab="Suppliment and Dose")



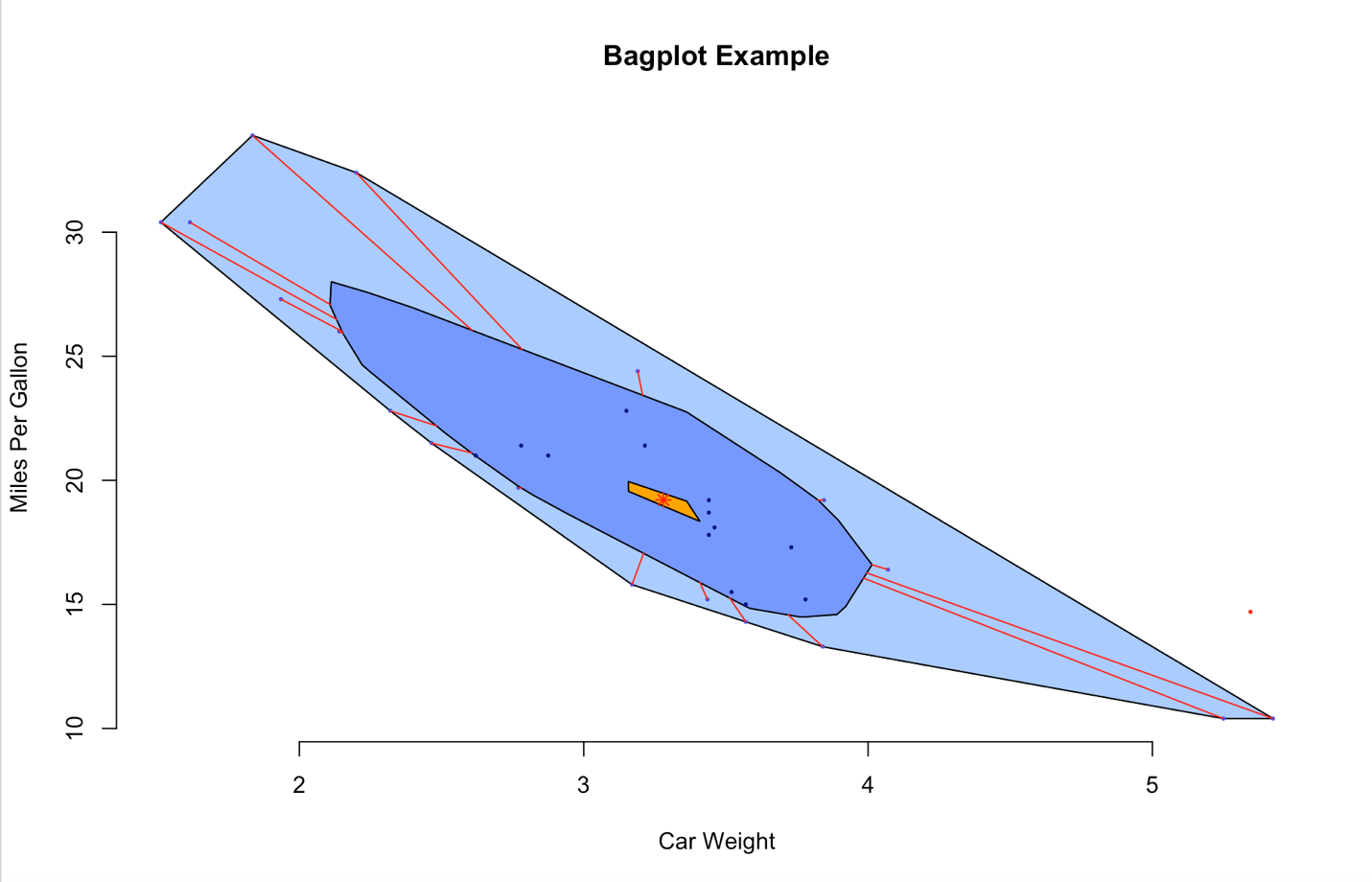
# Violin Plots

install.packages("vioplot")  
library(vioplot)  
x1 <- mtcars$mpg[mtcars$cyl==4]  
x2 <- mtcars$mpg[mtcars$cyl==6]  
x3 <- mtcars$mpg[mtcars$cyl==8]  
vioplot(x1, x2, x3, names=c("4 cyl", "6 cyl", "8 cyl"),   
   col="gold")  
title("Violin Plots of Miles Per Gallon")



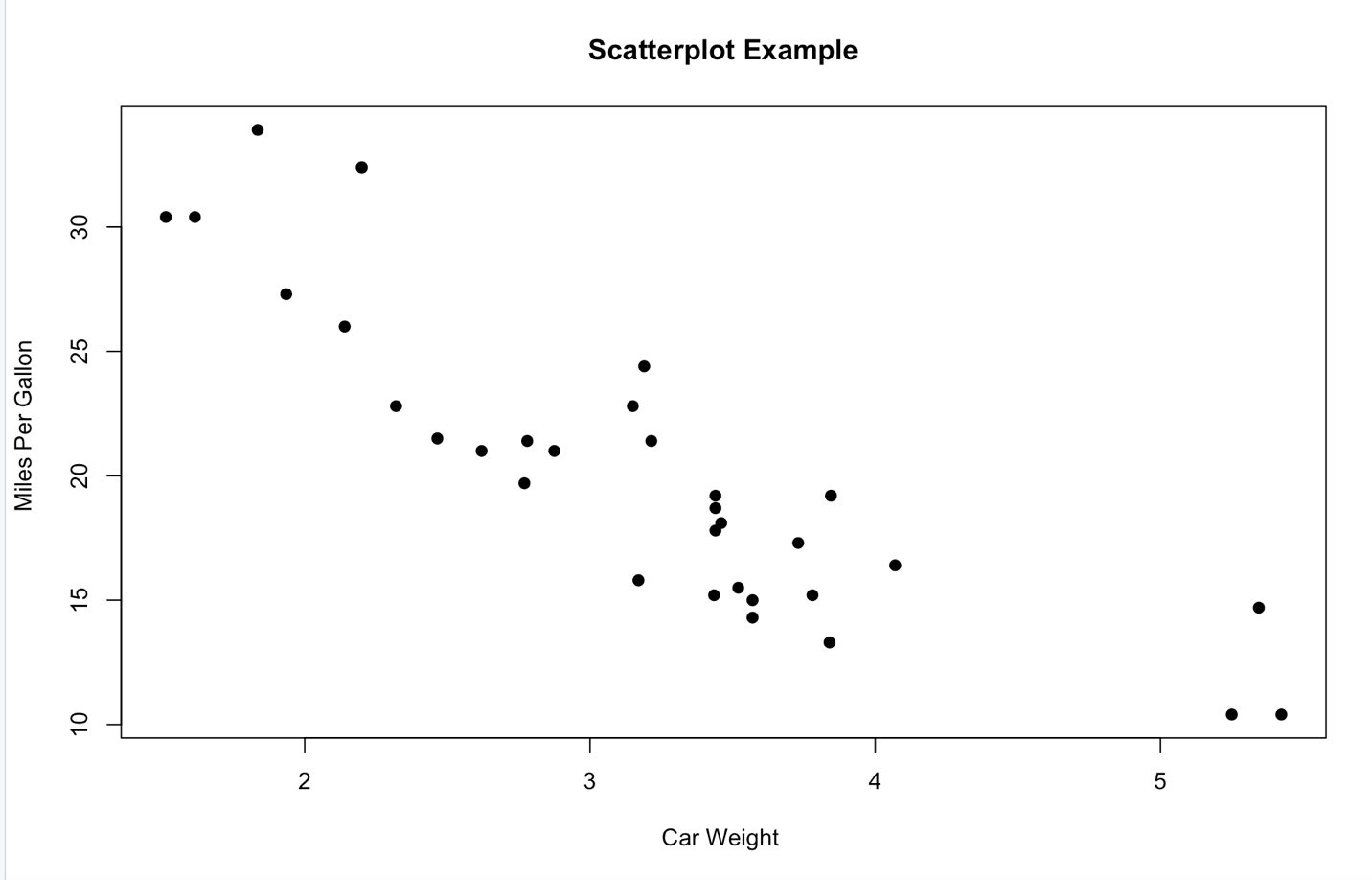
# Example of a Bagplot

install.packages("aplpack")  
library(aplpack)  
attach(mtcars)  
bagplot(wt,mpg, xlab="Car Weight", ylab="Miles Per Gallon",  
  main="Bagplot Example")

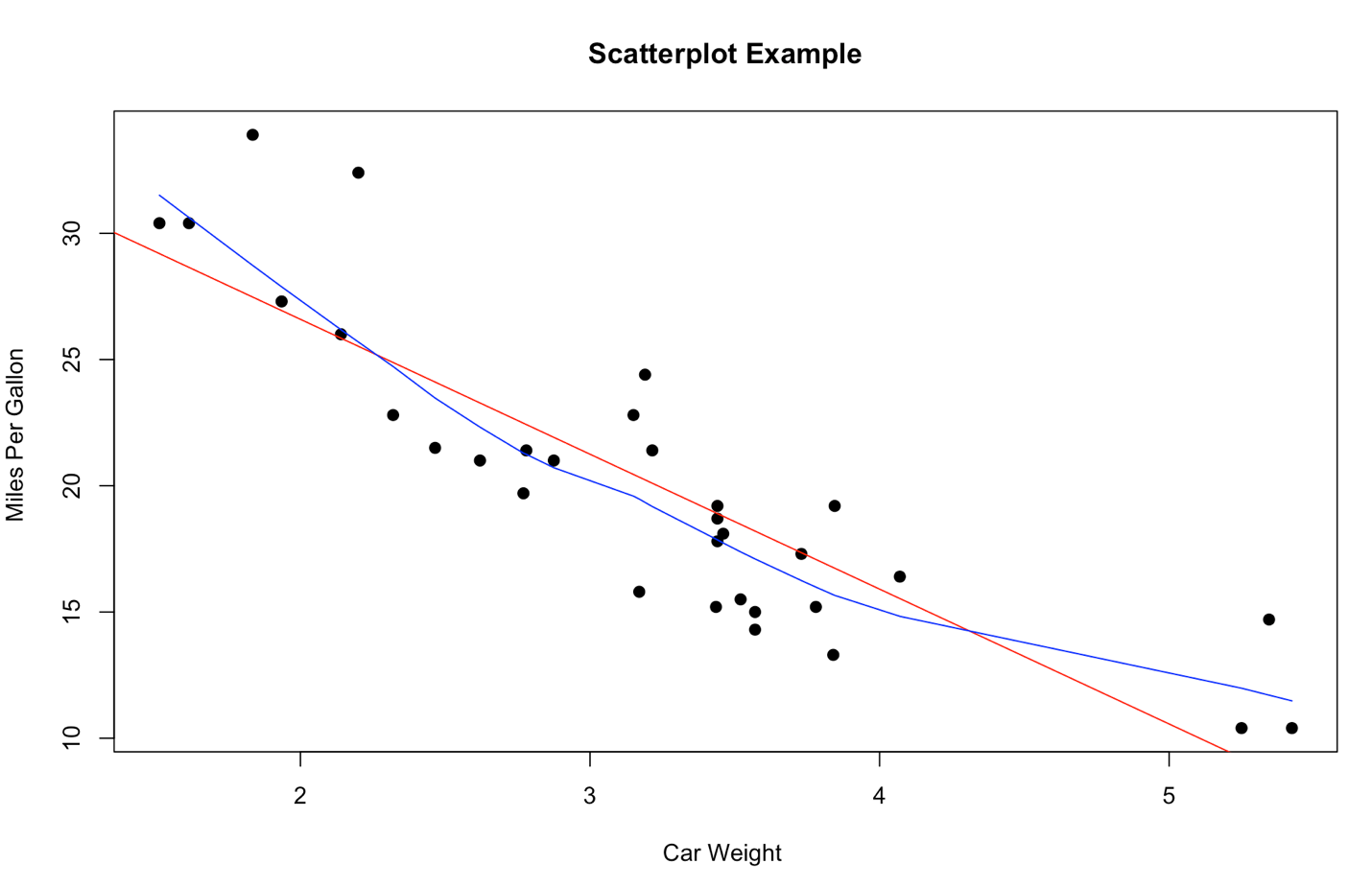


**9. Scatterplot**

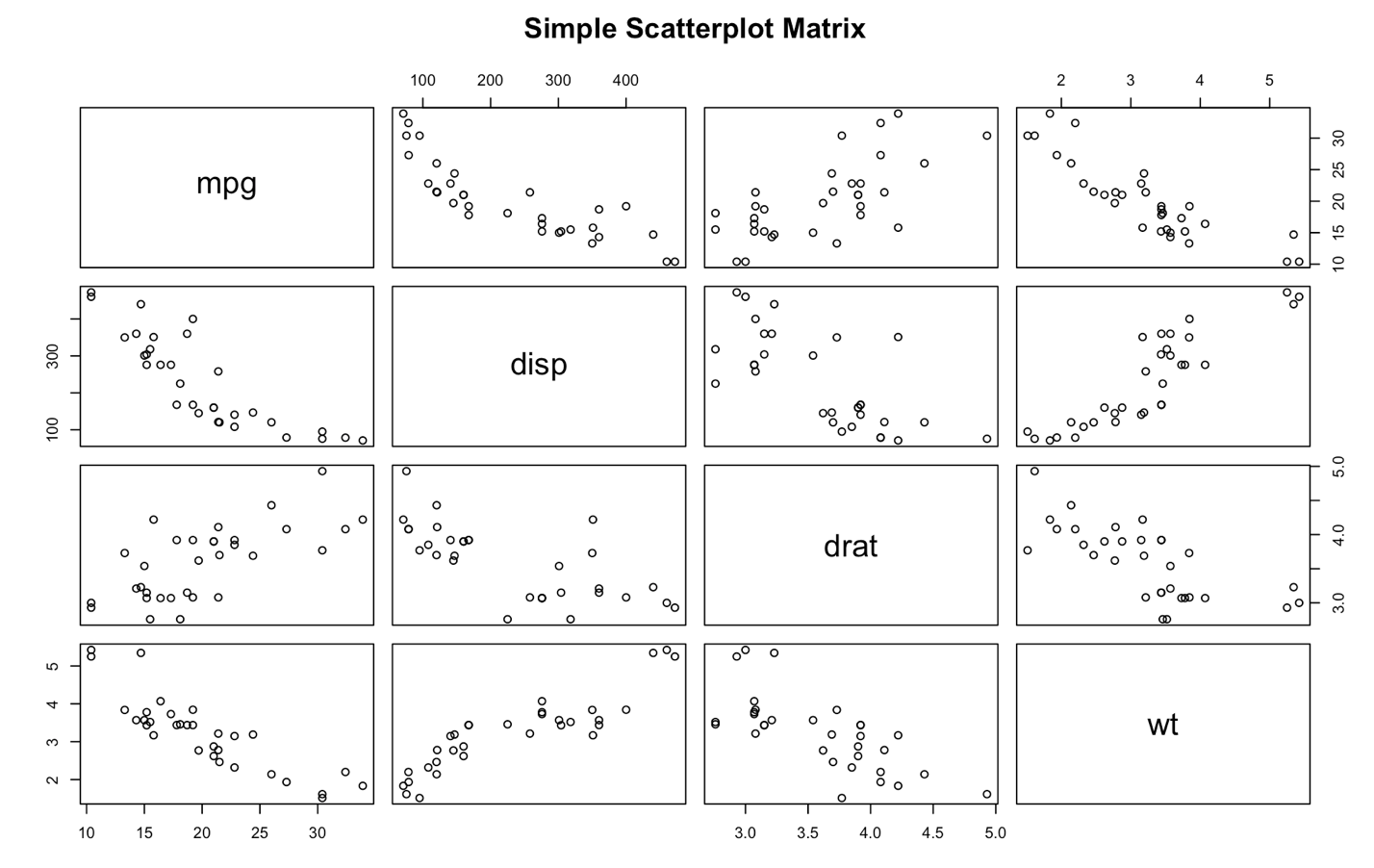
# Simple Scatterplot  
attach(mtcars)  
plot(wt, mpg, main="Scatterplot Example",   
   xlab="Car Weight ", ylab="Miles Per Gallon ", pch=19)

****

# Add fit lines  
abline(lm(mpg~wt), col="red") # regression line (y~x)   
lines(lowess(wt,mpg), col="blue") # lowess line (x,y)

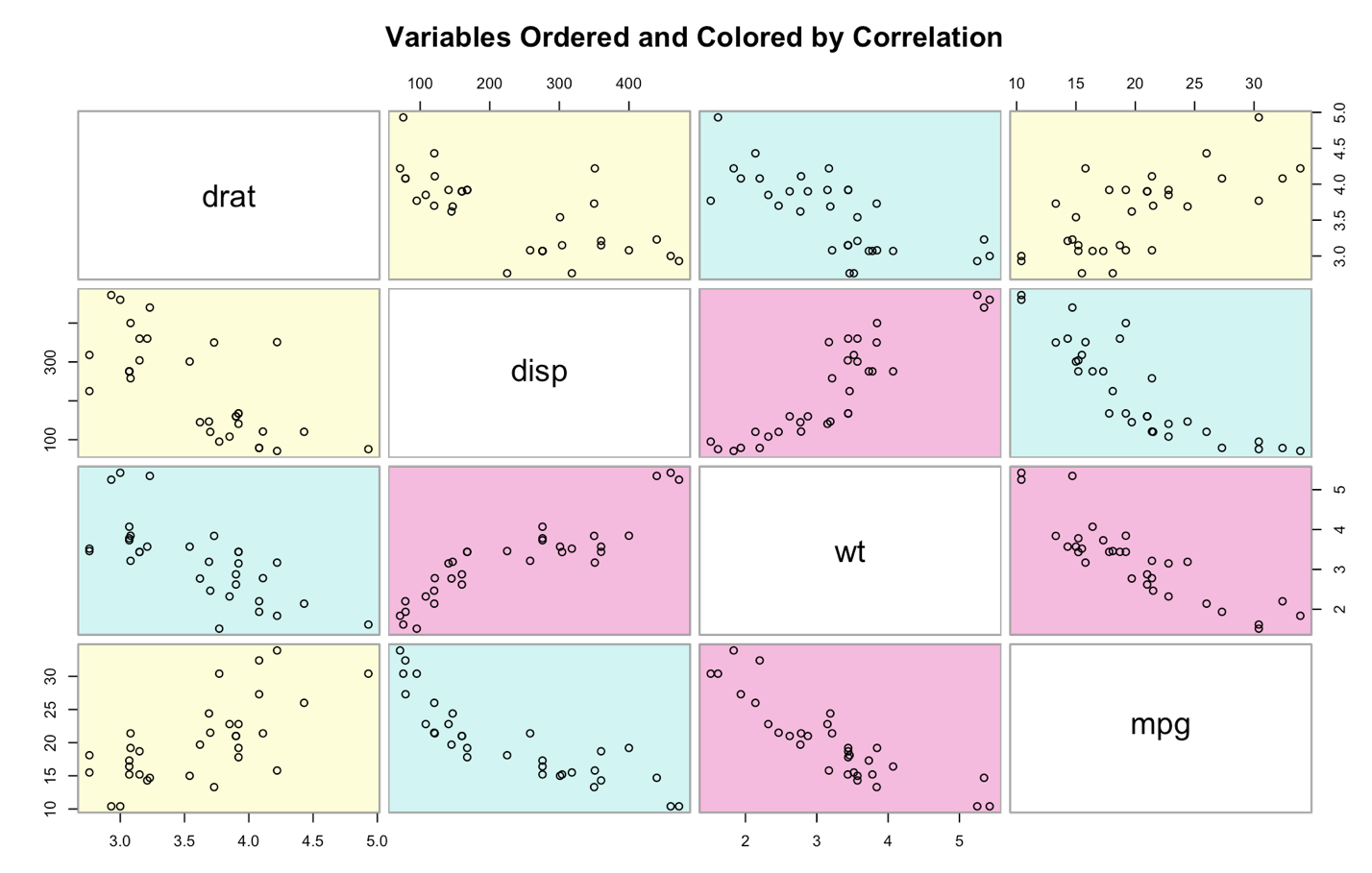
****

# Basic Scatterplot Matrix  
pairs(~mpg+disp+drat+wt,data=mtcars,   
   main="Simple Scatterplot Matrix")

****

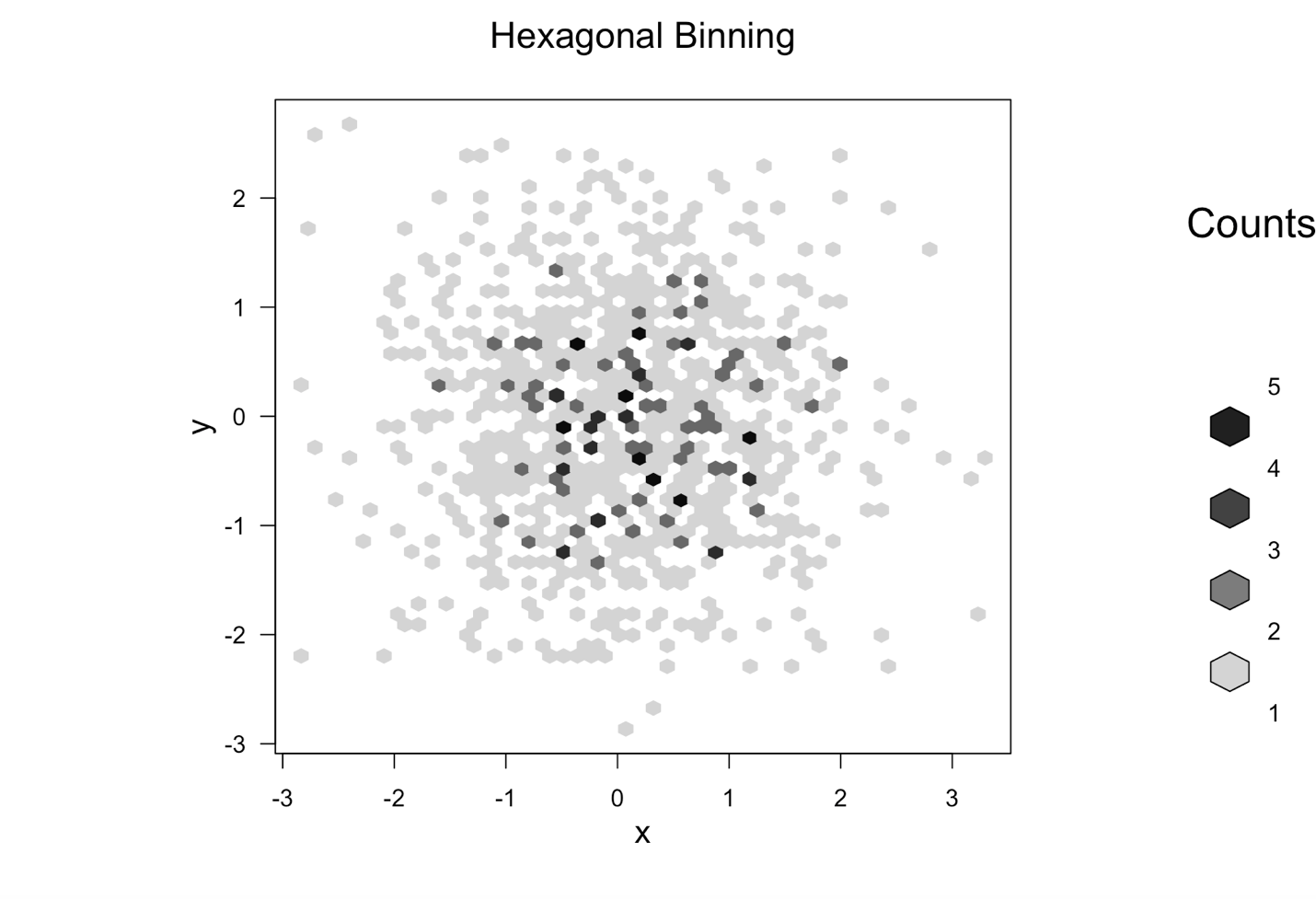
# Scatterplot Matrices from the glus Package

install.packages("gclus")  
library(gclus)  
dta <- mtcars[c(1,3,5,6)] # get data   
dta.r <- abs(cor(dta)) # get correlations  
dta.col <- dmat.color(dta.r) # get colors  
# reorder variables so those with highest correlation  
# are closest to the diagonal  
dta.o <- order.single(dta.r)   
cpairs(dta, dta.o, panel.colors=dta.col, gap=.5,  
main="Variables Ordered and Colored by Correlation" )

****

# High Density Scatterplot with Binning

install.packages("hexbin")  
library(hexbin)  
x <- rnorm(1000)  
y <- rnorm(1000)  
bin<-hexbin(x, y, xbins=50)   
plot(bin, main="Hexagonal Binning")

****

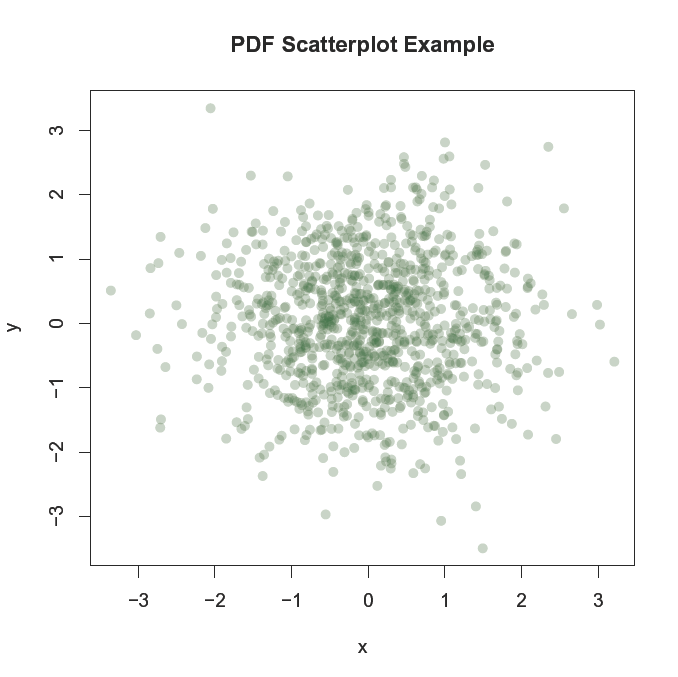
# High Density Scatterplot with Color Transparency

pdf("c:/scatterplot.pdf") //file location

x <- rnorm(1000)

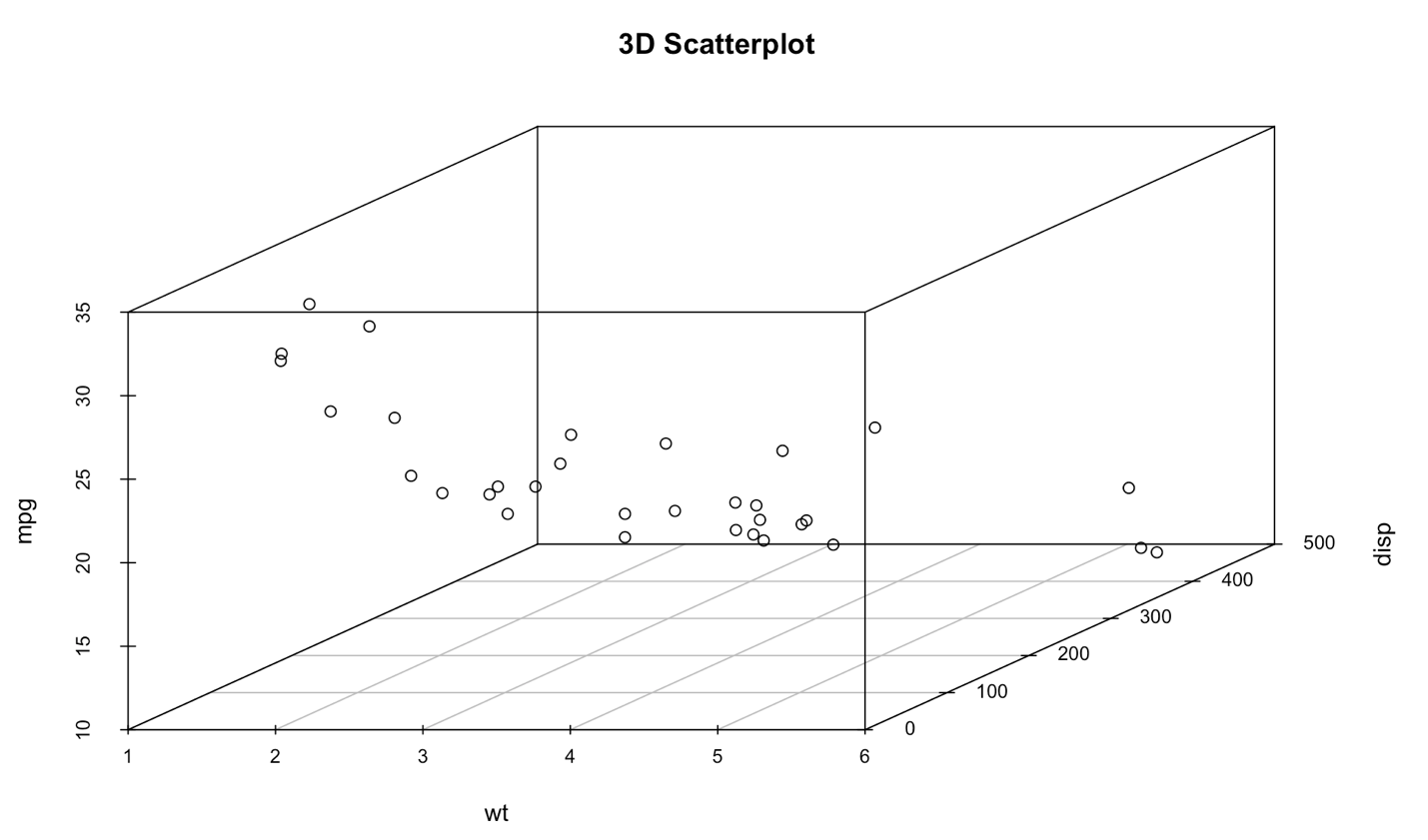
y <- rnorm(1000)

plot(x,y, main="PDF Scatterplot Example", col=rgb(0,100,0,50,maxColorValue=255), pch=16)

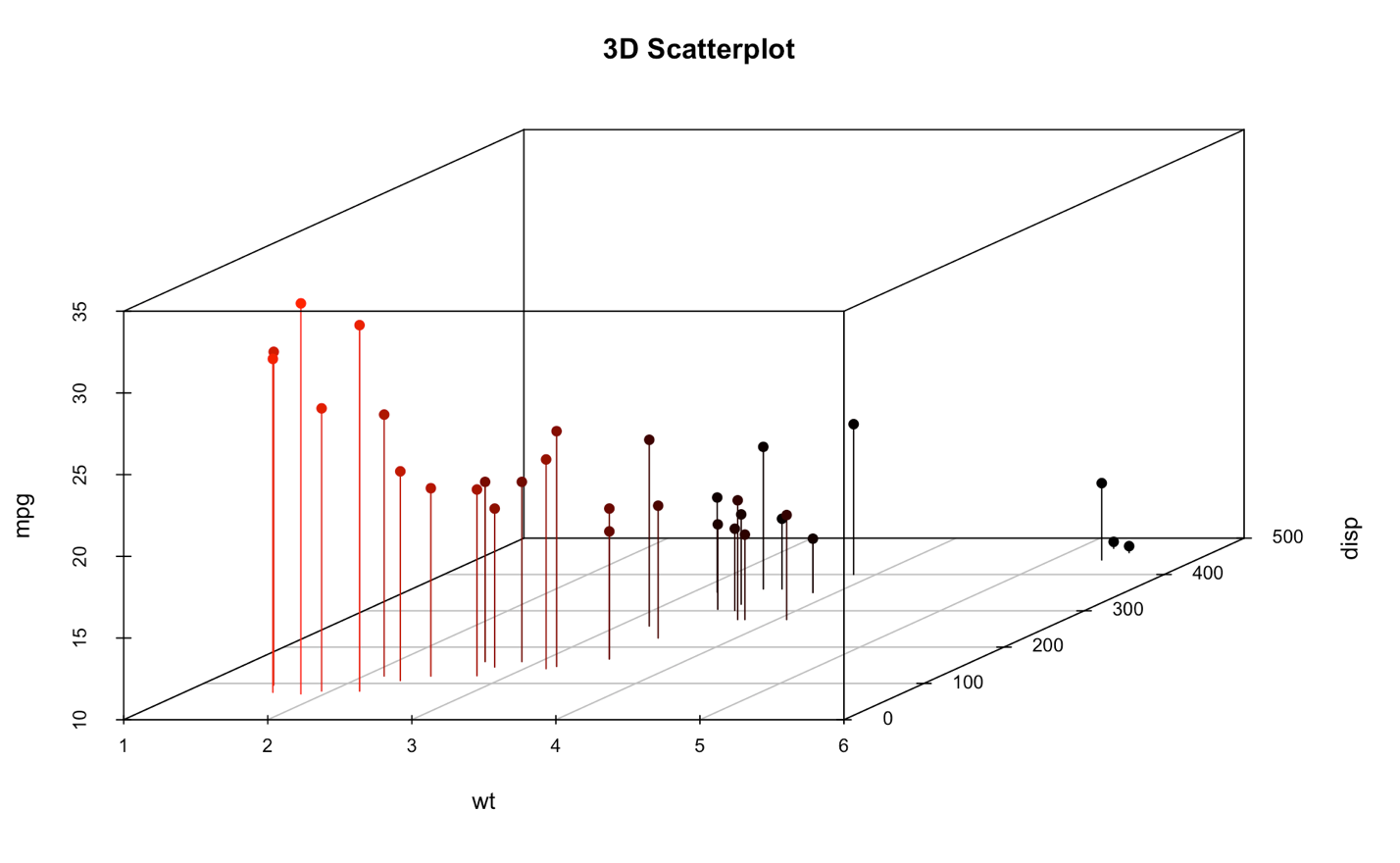


# 3D Scatterplot

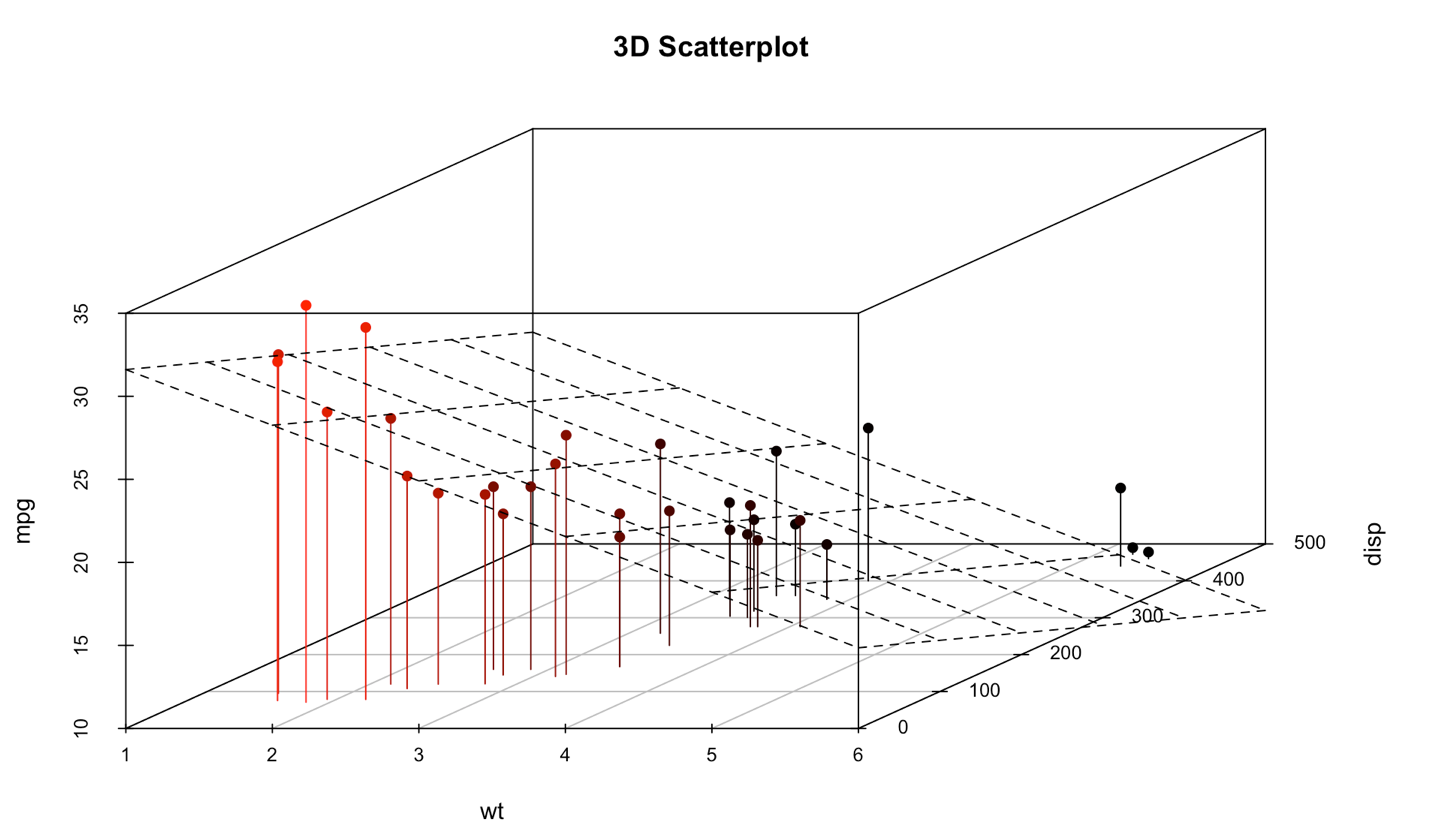
install.packages("scatterplot3d")  
library(scatterplot3d)  
attach(mtcars)  
scatterplot3d(wt,disp,mpg, main="3D Scatterplot")



# 3D Scatterplot with Coloring and Vertical Drop Lines  
library(scatterplot3d)   
attach(mtcars)   
scatterplot3d(wt,disp,mpg, pch=16, highlight.3d=TRUE,  
  type="h", main="3D Scatterplot")

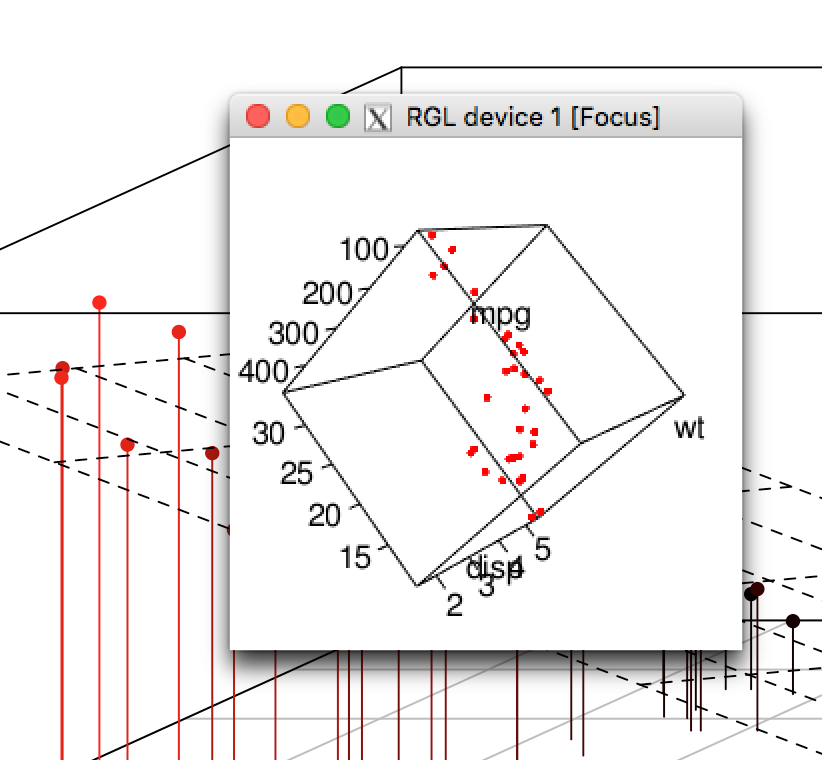


# 3D Scatterplot with Coloring and Vertical Lines  
# and Regression Plane   
library(scatterplot3d)   
attach(mtcars)   
s3d <-scatterplot3d(wt,disp,mpg, pch=16, highlight.3d=TRUE,  
  type="h", main="3D Scatterplot")  
fit <- lm(mpg ~ wt+disp)   
s3d$plane3d(fit)



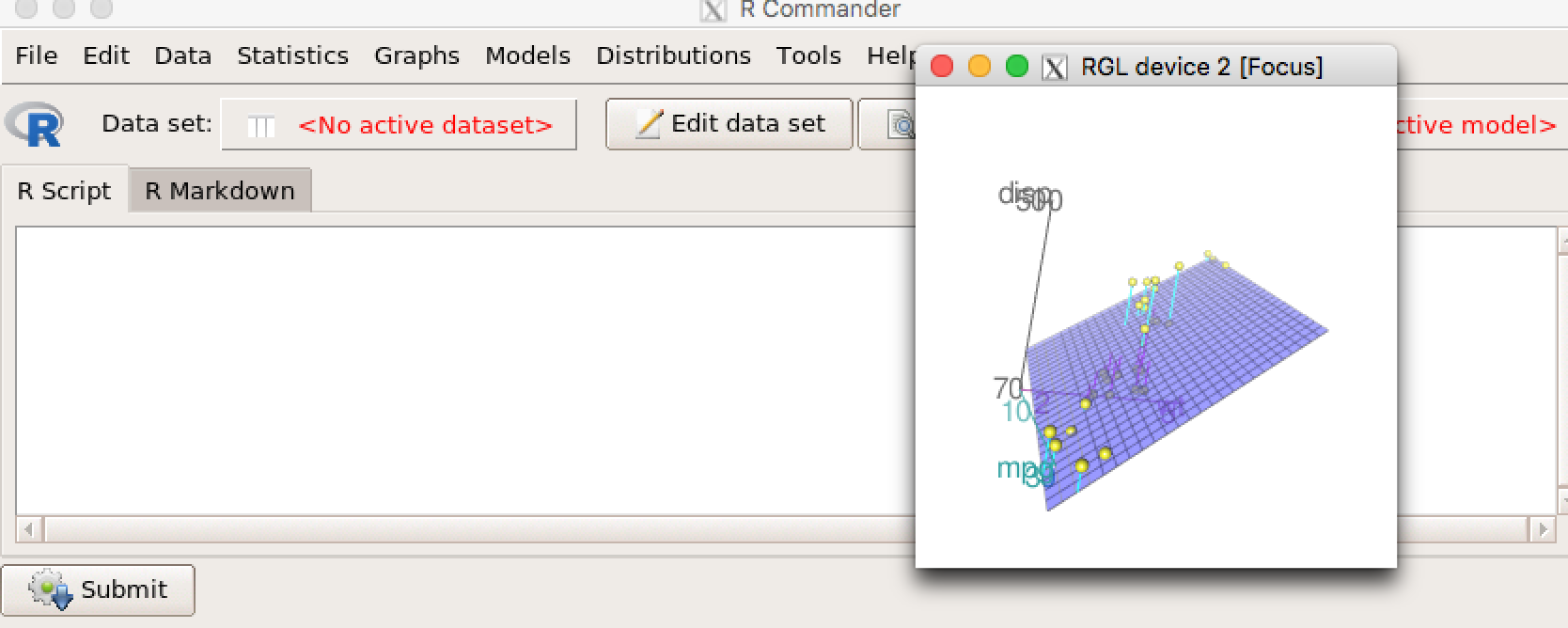
# Spinning 3d Scatterplot

Install.packages(“rgl”)  
library(rgl)  
plot3d(wt, disp, mpg, col="red", size=3)



# Another Spinning 3d Scatterplot

install.packages(“Rcmdr”)  
library(Rcmdr)  
attach(mtcars)  
scatter3d(wt, disp, mpg)



**Reading files in R**

Importing data into R is fairly simple. For Stata and Systat, use the foreign package. For SPSS and SAS I would recommend the [Hmisc](http://cran.r-project.org/web/packages/Hmisc/index.html) package for ease and functionality. See the Quick-R section on packages, for information on obtaining and installing the these packages. Example of importing data are provided below.

## **CSV**

# first row contains variable names, comma is separator   
# assign the variable id to row names  
# note the / instead of \ on mswindows systems   
  
mydata <- read.table("c:/mydata.csv", header=TRUE,   
   sep=",", row.names="id")

## **Excel**

One of the best ways to read an Excel file is to export it to a comma delimited file and import it using the method above. Alternatively you can use the **xlsx** package to access Excel files. The first row should contain variable/column names.

# read in the first worksheet from the workbook myexcel.xlsx  
# first row contains variable names  
library(xlsx)  
mydata <- read.xlsx("c:/myexcel.xlsx", 1)  
  
# read in the worksheet named mysheet  
mydata <- read.xlsx("c:/myexcel.xlsx", sheetName = "mysheet")

**SPSS**

# save SPSS dataset in trasport format  
get file='c:\mydata.sav'.  
export outfile='c:\mydata.por'.   
  
# in R   
library(Hmisc)  
mydata <- spss.get("c:/mydata.por", use.value.labels=TRUE)  
# last option converts value labels to R factors

## **SAS**

# save SAS dataset in trasport format  
libname out xport 'c:/mydata.xpt';  
data out.mydata;  
set sasuser.mydata;  
run;  
  
# in R   
library(Hmisc)  
mydata <- sasxport.get("c:/mydata.xpt")  
# character variables are converted to R factors

## **Stata**

# input Stata file  
library(foreign)  
mydata <- read.dta("c:/mydata.dta")

## **systat**

# input Systat file  
library(foreign)  
mydata <- read.systat("c:/mydata.dta")

**END OF CLASS**

Next Class:

1. Descriptive Statistics
2. Frequencies and Crosstabs
3. Correlations
4. t-tests
5. Nonparametric Tests of Group Differences
6. Multiple (Linear) Regression
7. Regression Diagnostics
8. ANOVA/MANOVA
9. (M)ANOVA Assumption
10. Resampling Statistics
11. Power analysis
12. Using with() and by()