MAT2001-STATISTICS FOR ENGINEERS-EMBEDDED LAB

Table and Graphical presentations -LAB-3

To consult the statistician after an experiment is finished is often merely to ask him to conduct a post mortem examination. He can perhaps say what the experiment died of.

(Ronald Fisher)

izquotes.com

- **Importing CSV and Tabular Data Files**

We can change the current working directory as follows:

- **setwd("<location of the dataset>")**

- **Example**

```
>setwd("C:\\Users\\admin\\Desktop\\")
>data=read.csv("stud.csv")
```

- Comma-separated values (CSV) files
- Data files have many formats and accordingly we have options for loading them.

```
>data=read.csv("C:\\Users\\admin\\Desktop\\Mokesh
\\stud.csv")
```

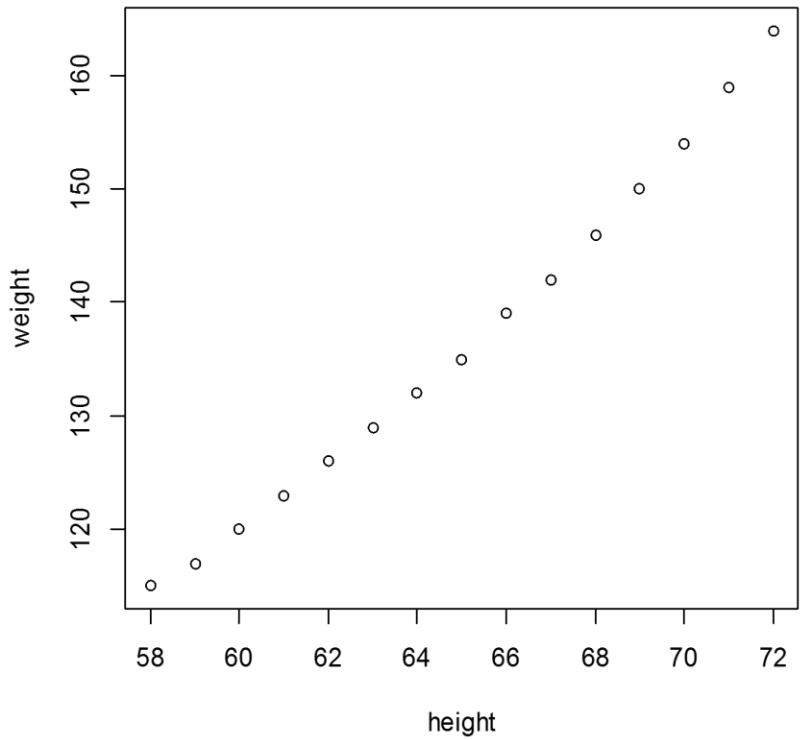
Or

```
>data=read.csv("C:/Users/admin/Desktop/Mokesh/stud
.csv")
```

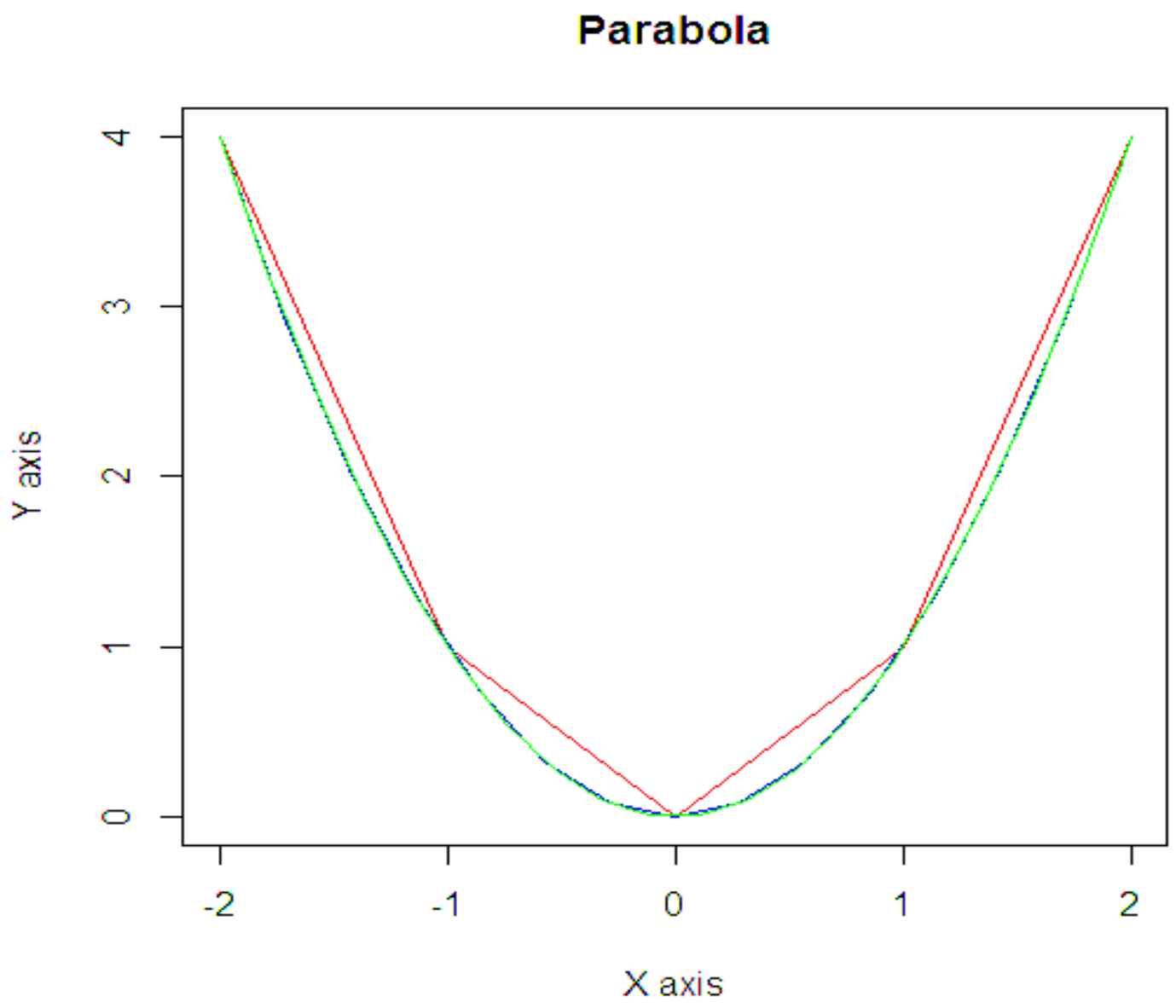
Graphics on R

- A simple plot plot(X) has each element of a discrete variable X ploted on the y-axis and the element's index on the x-axis

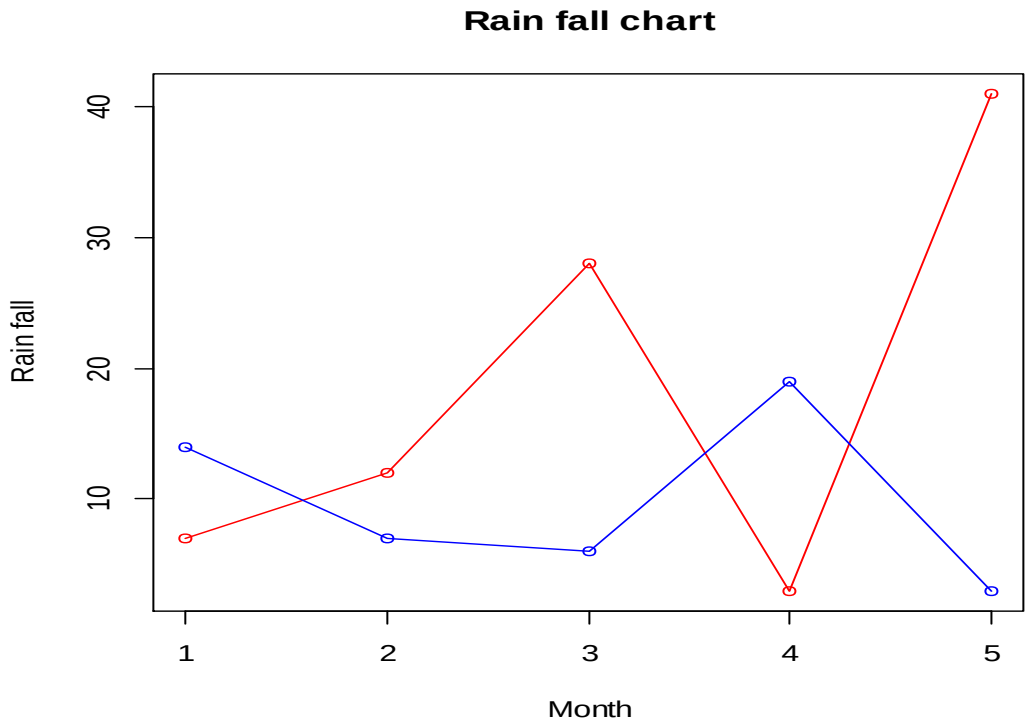
```
> # simple plot
> women   #inbuilt data set
   height weight
1      58    115
2      59    117
3      60    120
4      61    123
5      62    126
6      63    129
7      64    132
8      65    135
9      66    139
10     67    142
11     68    146
12     69    150
13     70    154
14     71    159
15     72    164
> plot(women)
```



```
> # function plot
> x = seq(-2,2)
> y = x^2
> # edgy graph!
> plot(x,y,type="l",xlab="X axis",ylab="Y axis",main="Parabola", col = "red")
> # better
> sp <- spline(x, y) # spline interpolation of data points
> lines(sp, col = "blue")
> # much better
> sp <- spline(x, y,n=20) # interpolation at n points spanning [xmin, xmax]
> lines(sp, col = "green")
>
```

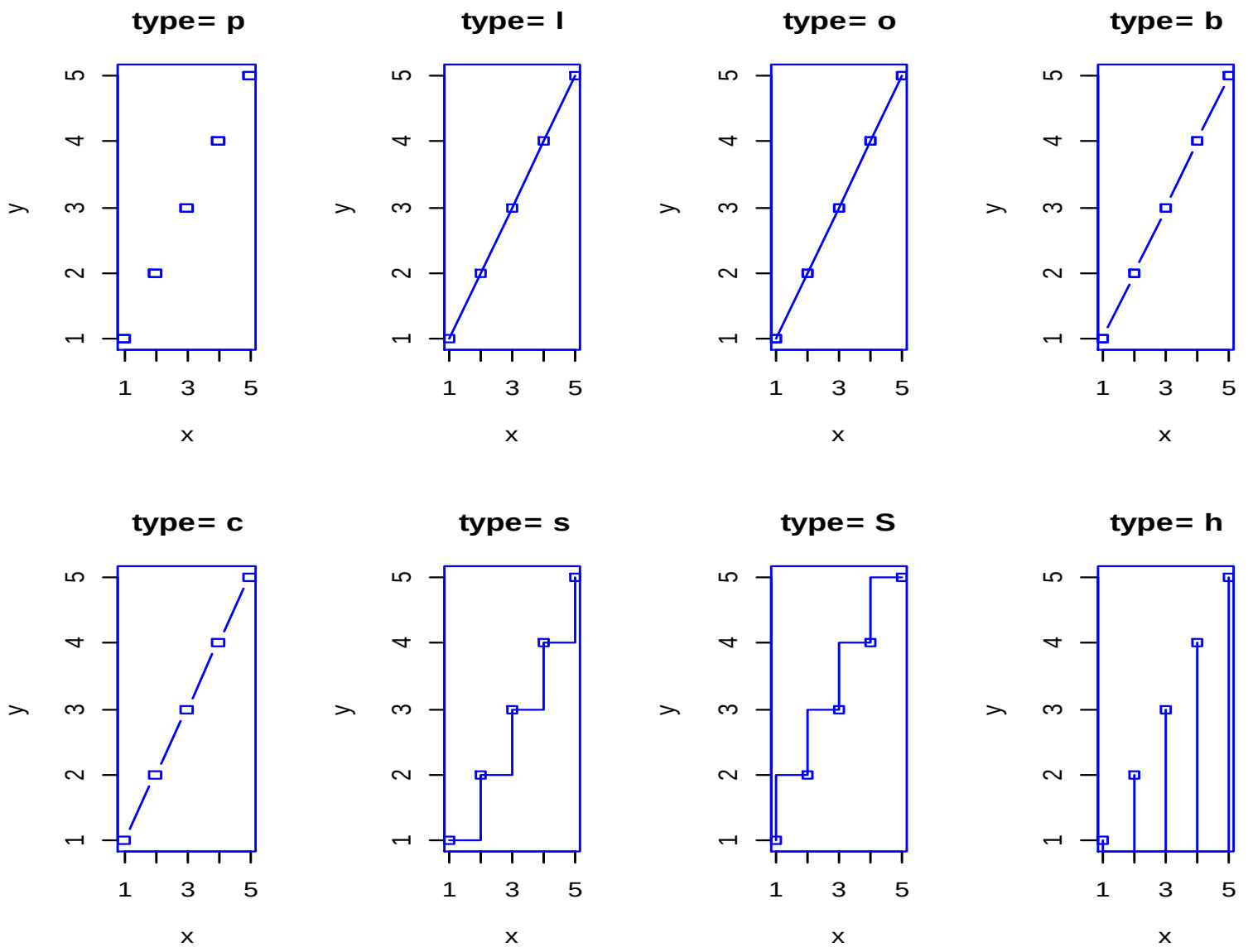

Parabola
Y axis
X axis
0
1
2
3
4
-2
-1
0
1
2

```
>v <- c(7,12,28,3,41)
>t <- c(14,7,6,19,3)
> plot(v,type = "o", col = "red", xlab = "Month", ylab = "Rain  fall",main = "Rain fall chart")
>lines(t, type = "o", col = "blue")
```



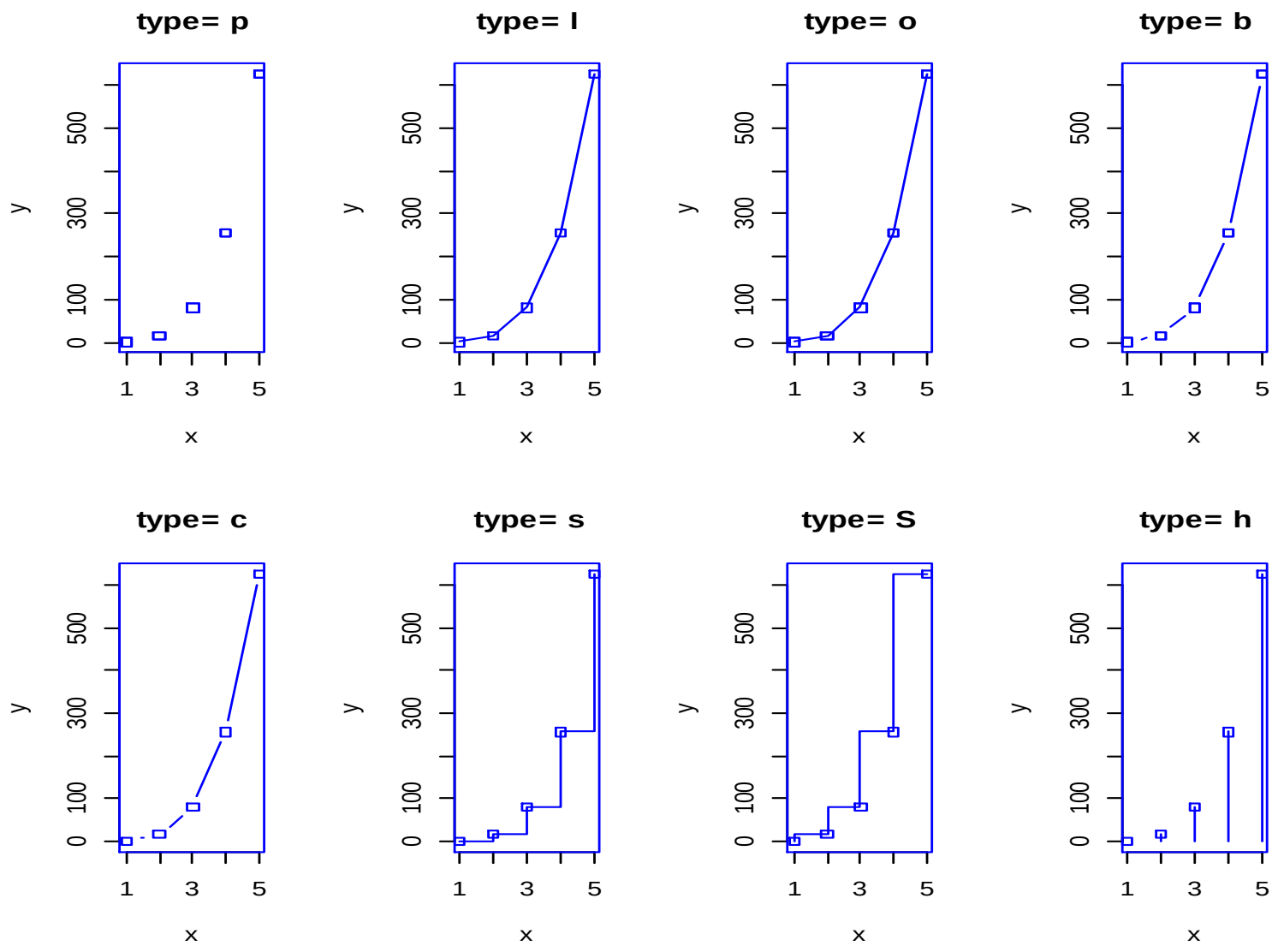
- **Line chart**
- A line chart is a simple plot with consecutive plots connected by lines

```
> x <- c(1:5)
> y <- x # create some data
> par(pch=22, col="blue") # 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, main=heading)
+ lines(x, y, type=opts[i])
+ }
```


type= p
type= l
type= o
type= b
type= c
type= s
type= S
type= h
y
x
1
2
3
4
5
1
3
5

R Console

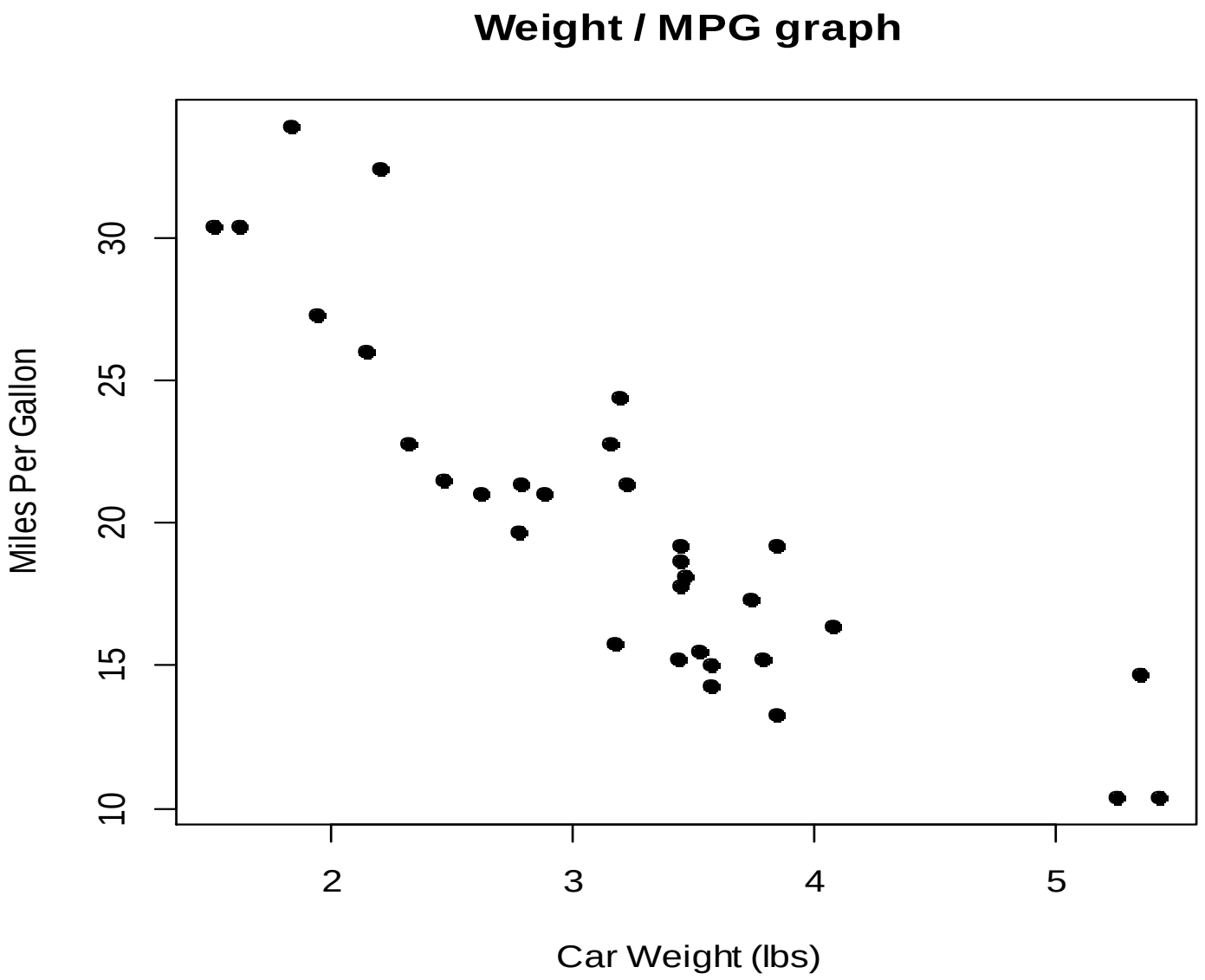
```
> x <- c(1:5)
> y <- x^4 # create some data
> par(pch=22, col="blue") # plotting symbol and color
> par(mfrow=c(2,4)) # all plots on one page
> opts = c("p","l","o","b","c","s","S","h")
> for(i in 1:length(opts))
+ {
+ heading = paste("type=",opts[i])
+ plot(x, y, main=heading)
+ lines(x, y, type=opts[i])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


type= p
type= l
type= o
type= b
type= c
type= s
type= S
type= h
y
x
0
100
300
500
1
3
5

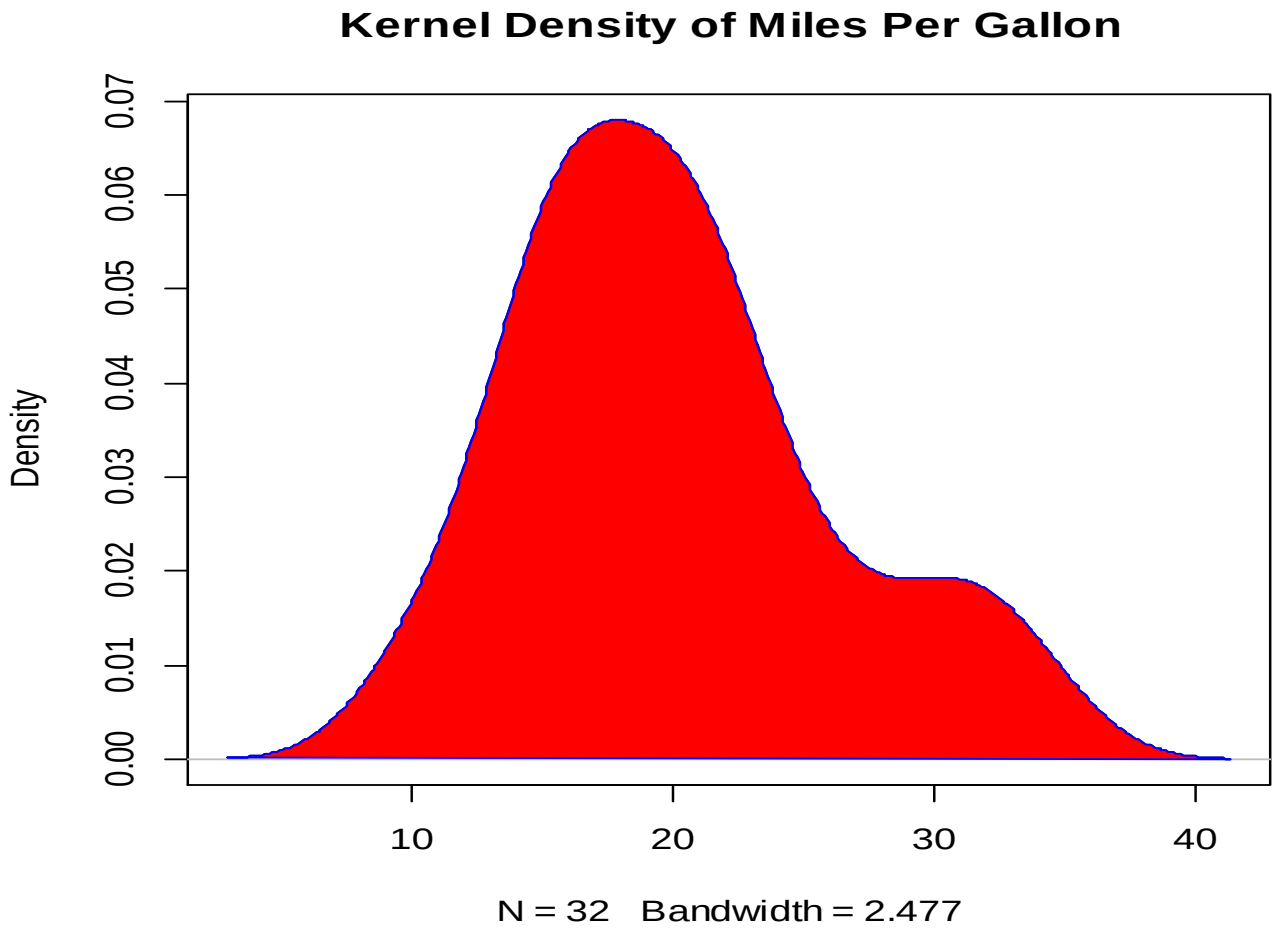
- **Scatterplot**

A scatterplot plot(X,Y) has each element of a variable Y ploted on the y-axis and the corresponding element for variable X on the x-axis

```
# scatterplot
>attach(mtcars)
>plot(wt, mpg, main="Weight / MPG graph",
xlab="Car Weight (lbs)", ylab="Miles Per Gallon",
pch=19)
```


Weight / MPG graph
Miles Per Gallon
10
15
20
25
30
2
3
4
5
Car Weight (lbs)

- **Kernel density plots**
- Kernel density plots nicely visualize the shape of a distribution. They can be better than histograms, even with normal curves because histograms are strongly affected by the number of bins used and by outliers.
- **# Kernel density plot**
- **>d <- density(mtcars$mpg) # kernel density estimates**
- **>plot(d)**
- **# Filled density plot**
- **>d <- density(mtcars$mpg)**
- **>plot(d, main="Kernel Density of Miles Per Gallon")**
- **>polygon(d, col="red", border="blue")**


Kernel Density of Miles Per Gallon
Density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
10
20
30
40
N = 32 Bandwidth = 2.477

- **boxplot(X) is a plot that, if X is a vector, the vector elements are the heights of the bars in the plot, if X is a matrix, the matrix columns are the heights of the bars in the plot, stacked after the first bar (column)**

- **If the argument beside=TRUE, then the values in each column are juxtaposed, not stacked.**

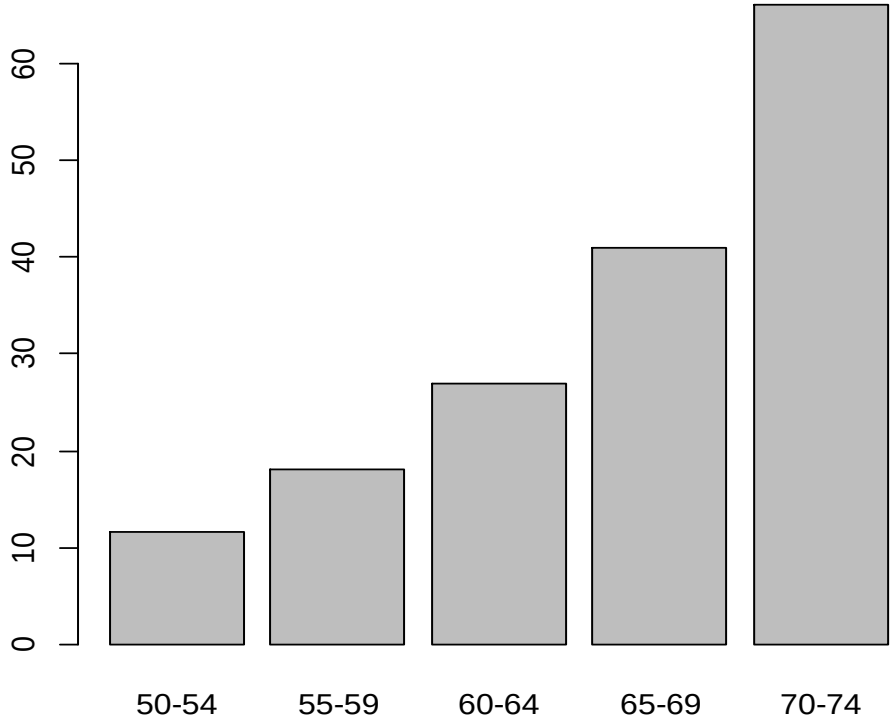
- **The argument horiz=TRUE creates an horizontal barplot.**

R Console

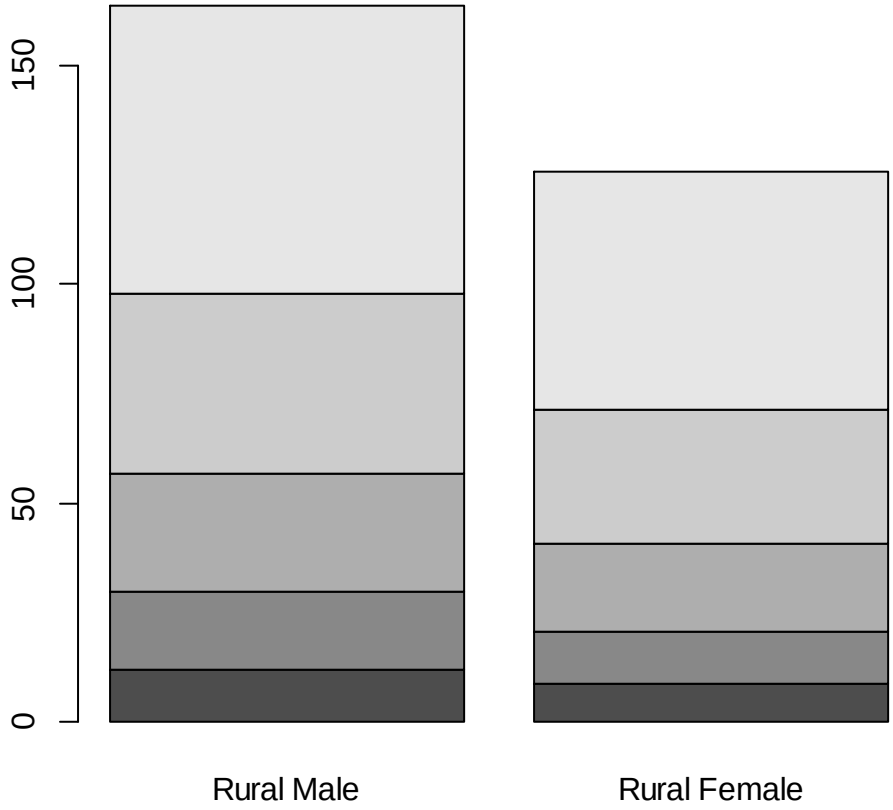
```
> VADeaths
      Rural Male Rural Female Urban Male Urban Female
50-54       11.7          8.7       15.4          8.4
55-59       18.1         11.7       24.3         13.6
60-64       26.9         20.3       37.0         19.3
65-69       41.0         30.9       54.6         35.1
70-74       66.0         54.3       71.1         50.0
> class(VADeaths)
[1] "matrix"
> dimnames(VADeaths)
[[1]]
[1] "50-54" "55-59" "60-64" "65-69" "70-74"

[[2]]
[1] "Rural Male"   "Rural Female" "Urban Male"   "Urban Female"
```

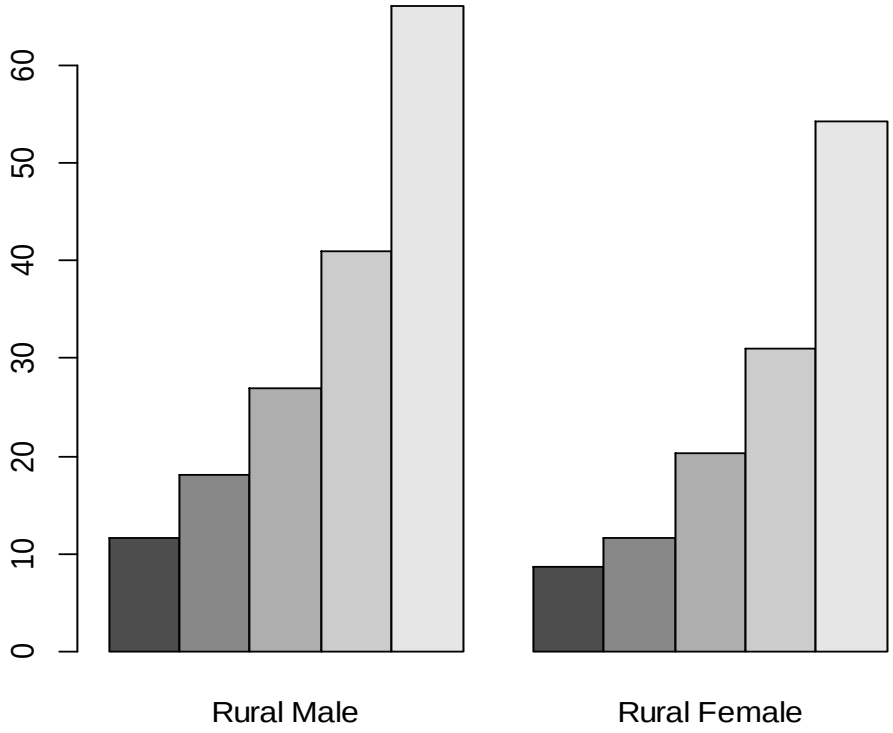
- > simple barplot
- > barplot (VADeaths[,"Rural Male"])



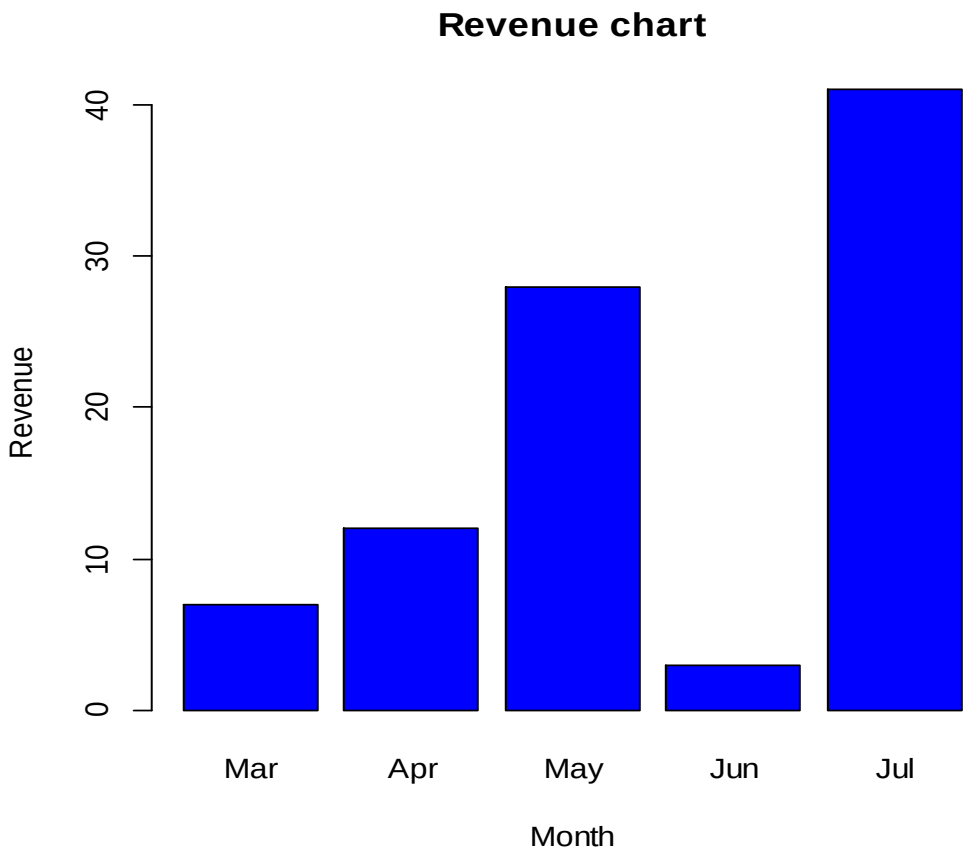
- # stacked barplots
  - barplot(VADeaths[,c("Rural Male", "Rural Female")])



- > # juxtaposed barplots
- > barplot(VADeaths [,c("Rural Male", "Rural Female")],beside=T)

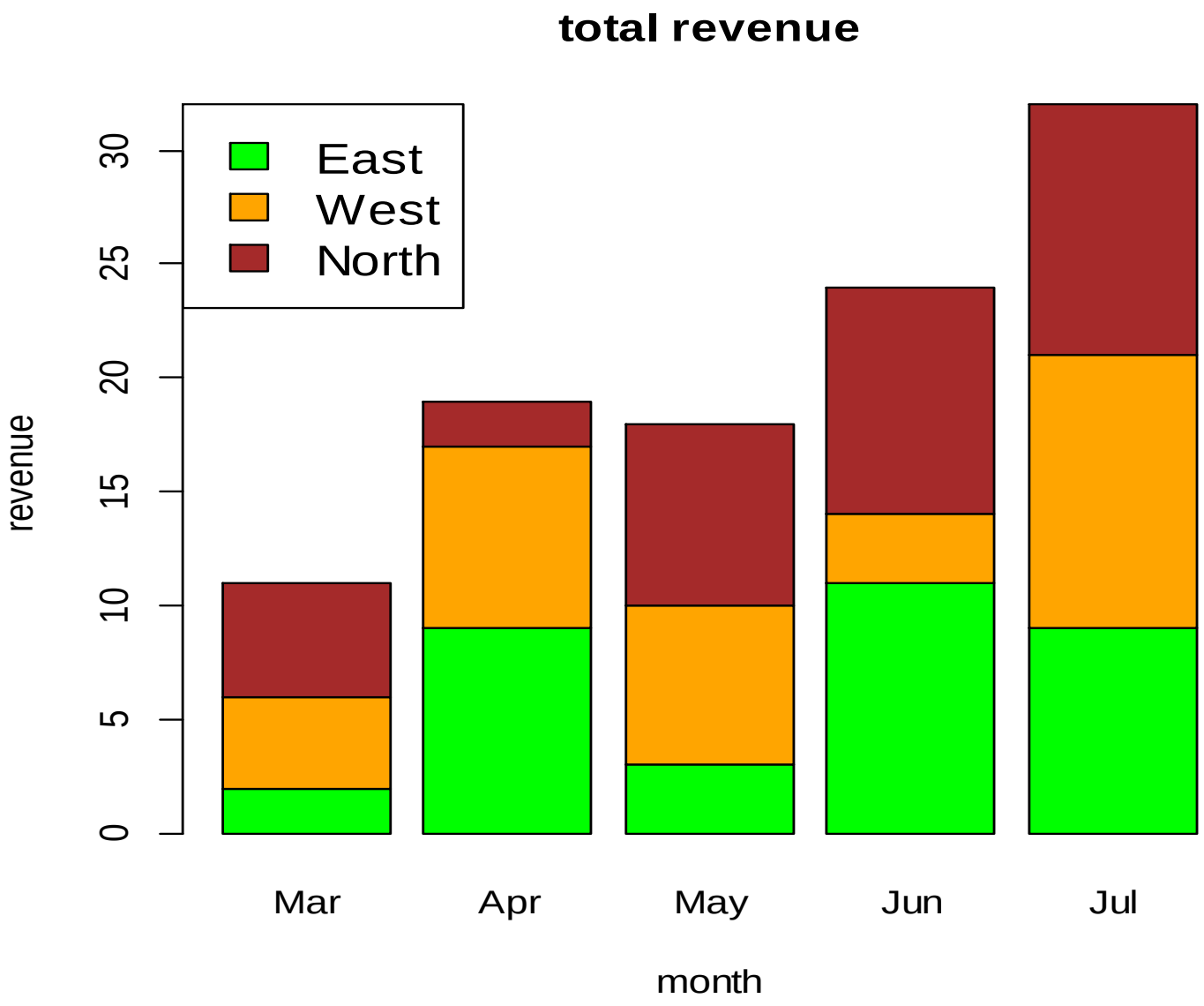


```
>H <- c(7,12,28,3,41)
>M <- c("Mar","Apr","May","Jun","Jul")
>barplot(H,names.arg = M,xlab = "Month",ylab = "Revenue",col="blue",main = "Revenue chart")
```



Example :-

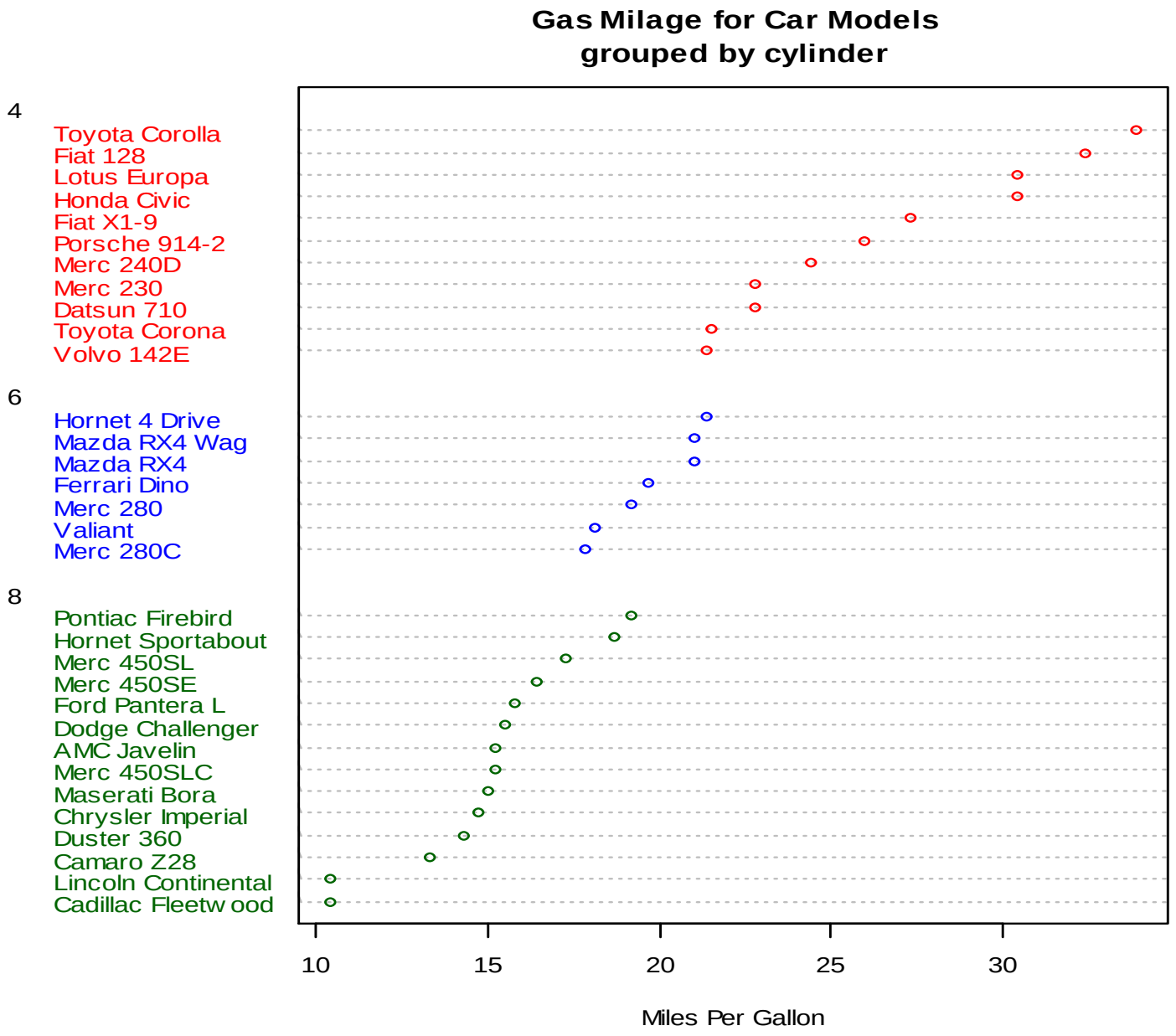
```
>colors <- c("green","orange","brown")
>months <- c("Mar","Apr","May","Jun","Jul")
>regions <- c("East","West","North")
>Values <-
matrix(c(2,9,3,11,9,4,8,7,3,12,5,2,8,10,11),nrow = 3,ncol
= 5,byrow =TRUE)
>barplot(Values,main = "total revenue",names.arg =
months,xlab = "month",ylab = "revenue",col=colors)
>legend("topleft", regions, cex = 1.3, fill = colors)
```


total revenue
East
West
North
revenue
0
5
10
15
20
25
30
Mar
Apr
May
Jun
Jul
month

- # Simple Dotplot

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

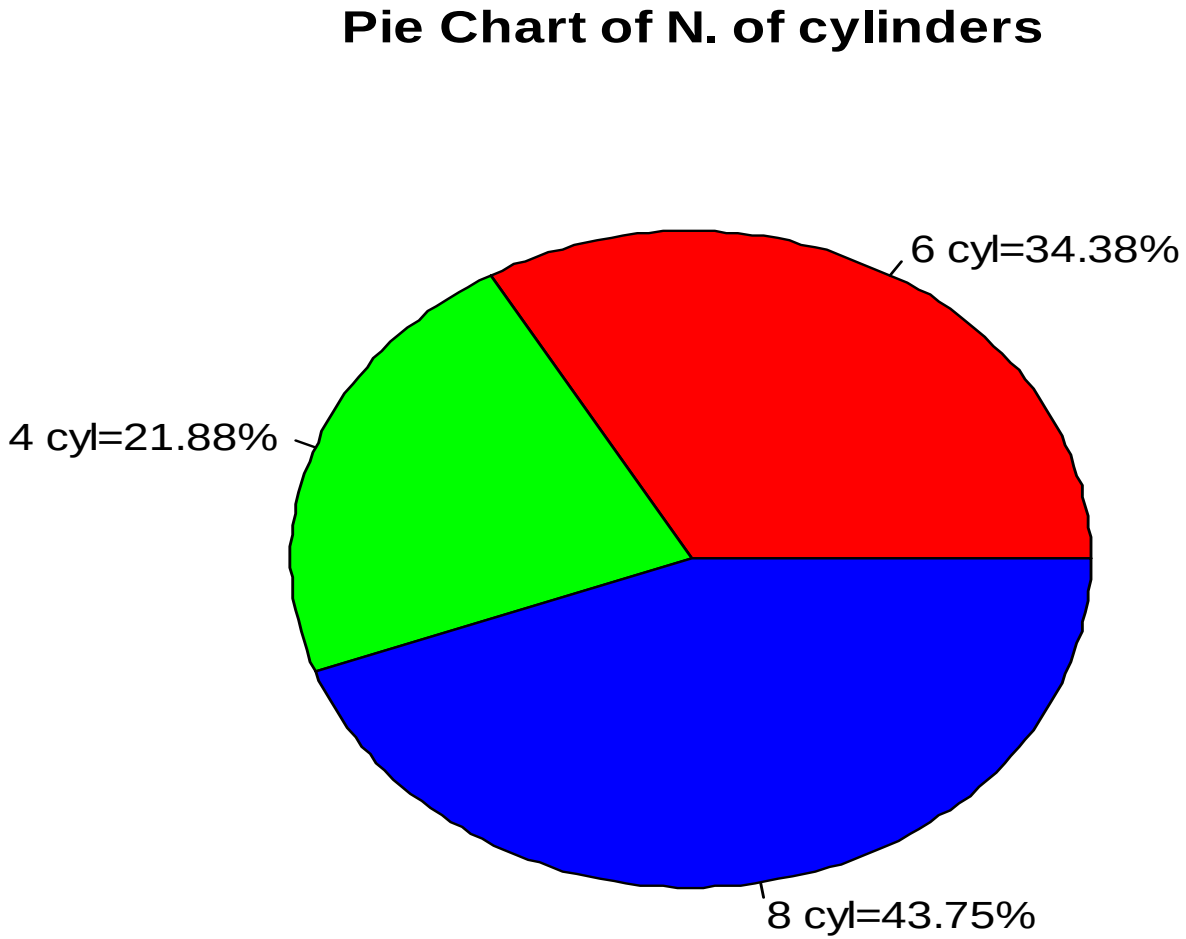
- # Dotplot: Grouped Sorted and Colored
- # Sort by mpg, group and color by cylinder
- >x <- mtcars[order(mtcars$mpg),] # sort by mpg
- >x$cyl <- factor(x$cyl) # it must be a factor
- >x$color[x$cyl==4] <- "red"
- >x$color[x$cyl==6] <- "blue"
- >x$color[x$cyl==8] <- "darkgreen"
- >dotchart(x$mpg,labels=row.names(x),cex=.7,groups=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


Gas Milage for Car Models
grouped by cylinder
4
Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1-9
Porsche 914-2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
6
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Merc 280
Valiant
Merc 280C
8
Pontiac Firebird
Hornet Sportabout
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
10
15
20
25
30
Miles Per Gallon

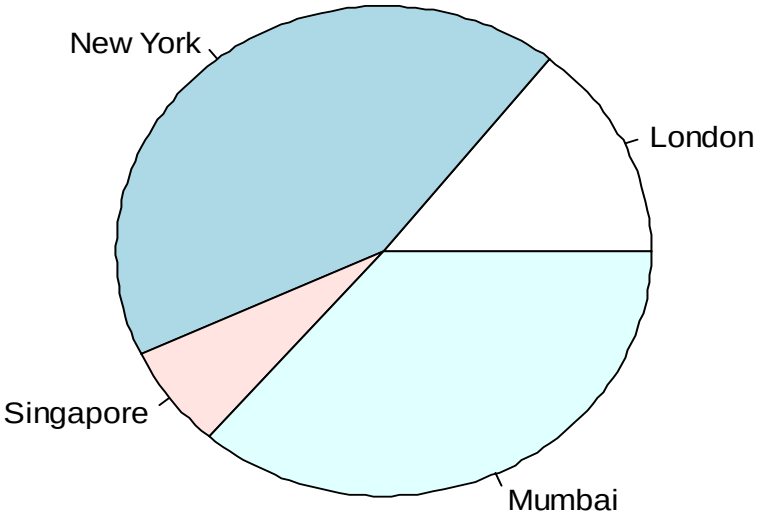
- **Pie**
- pie(x) draws a circle (pie) cut into segments (slices), each slice represents a unique value from the elements of x and the sixe of the slice and the relative frequency of each unique value is represented by the size of t

 # simple pie

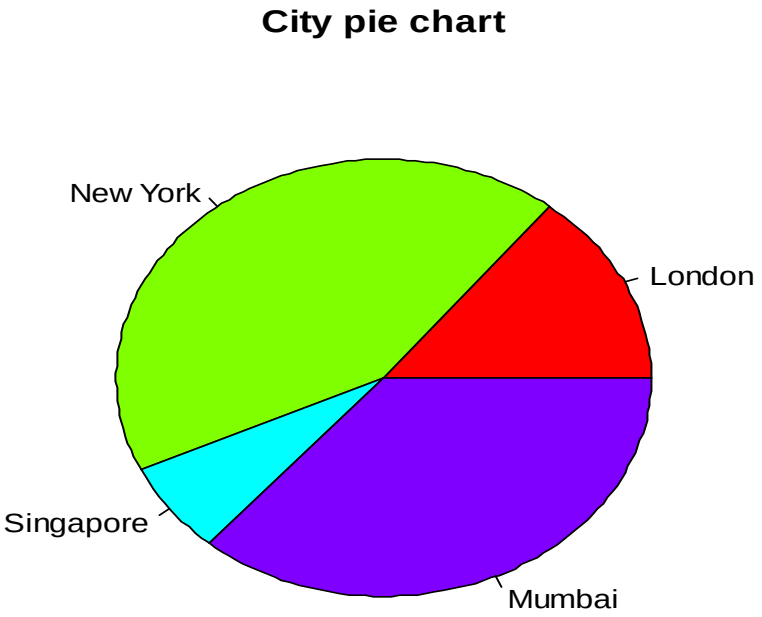
```
>pie(unique(mtcars$cyl), labels = unique(mtcars$cyl), main="Pie Chart of N. of cylinders")                    # pie with percentages and colors
>with(mtcars, {
>n.cyl <- unique(cyl)
>percent.cyl <-round(table(cyl)/dim(mtcars)[1]*100,2)
>lbls <- paste(n.cyl," cyl=",percent.cyl,"%", sep="")
>pie(n.cyl, labels = lbls , main="Pie Chart of N. of cylinders", col=rainbow(length(lbls)))})
```


Pie Chart of N. of cylinders
6 cyl=34.38%
4 cyl=21.88%
8 cyl=43.75%

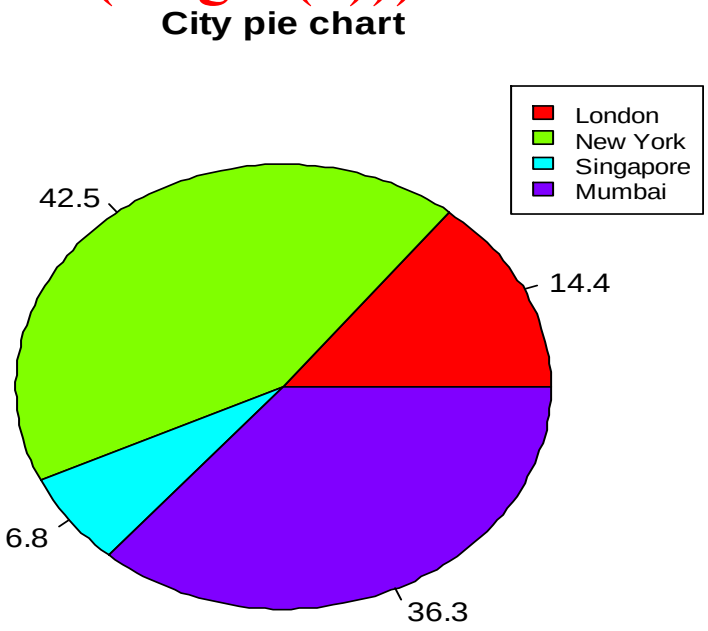
```
>x <- c(21, 62, 10, 53)
>labels <- c("London", "New York", "Singapore",
"Mumbai")
>pie(x,labels)
```



```
>x = c(21, 62, 10, 53)
>labels = c("London", "New York", "Singapore", "Mumbai")
>pie(x, labels, main = "City pie chart", col = rainbow(length(x)))
```

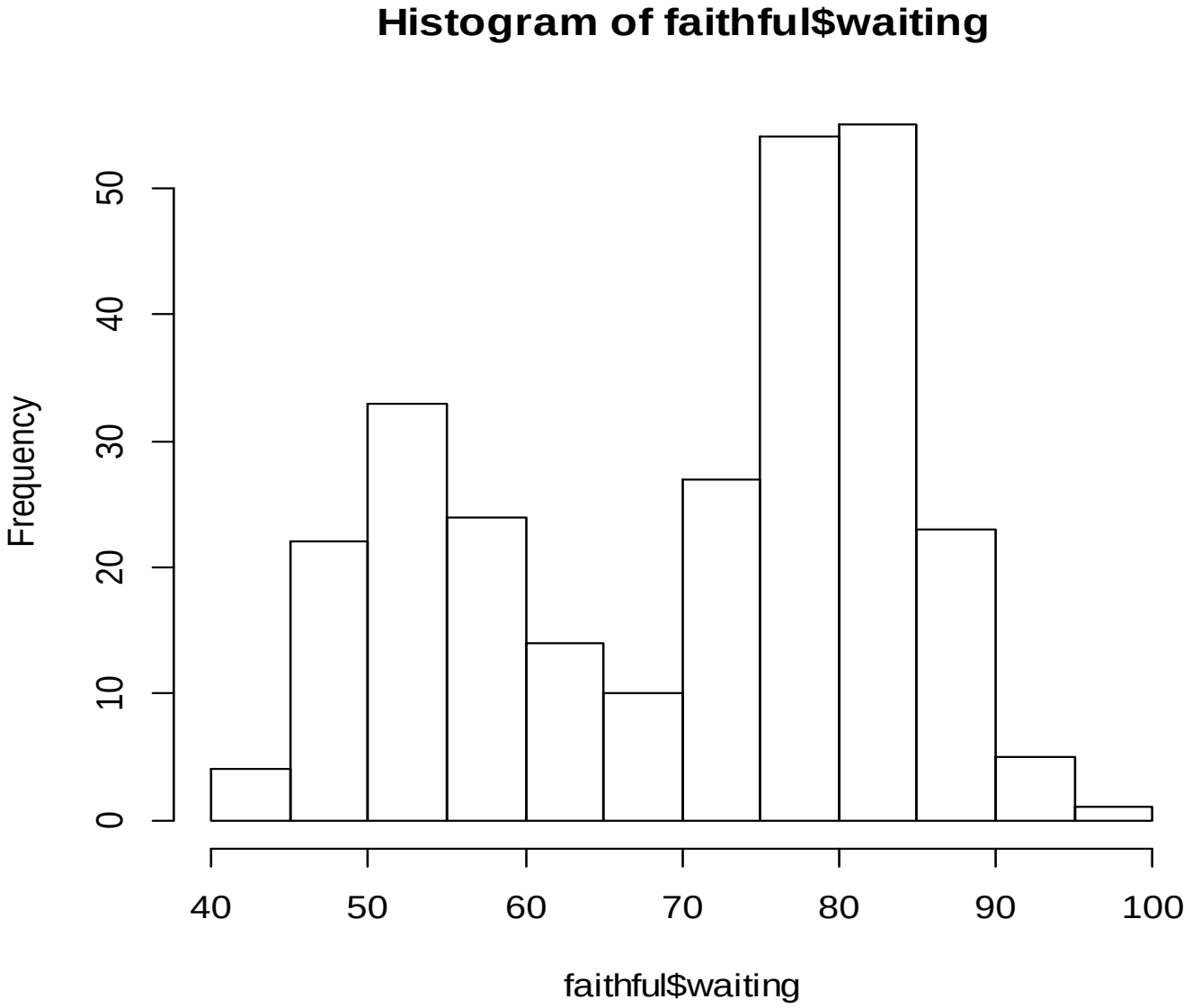


```
>x <-  c(21, 62, 10,53)
>labels <-  c("London","New York“ ,"Singapore“ ,"Mumbai" )
>piepercent<- round(100*x/sum(x), 1)
>pie(x, labels = piepercent, main = "City pie chart",col = rainbow(length(x)))
>legend("topright", c("London","New York","Singapore", "Mumbai"), cex = 0.8,fill = rainbow(length(x)))
```



- **histogram**
- hist(X) is an histogram, a bar plot with the frequencies of the values in X on the y-axis and the ranges of values on the x-axis
- A cumulative distribution curve is the proportion of X on the y-axis, up to the current position on the x-axis

- **> # simple histogram**
- **> hist(faithful$waiting)**

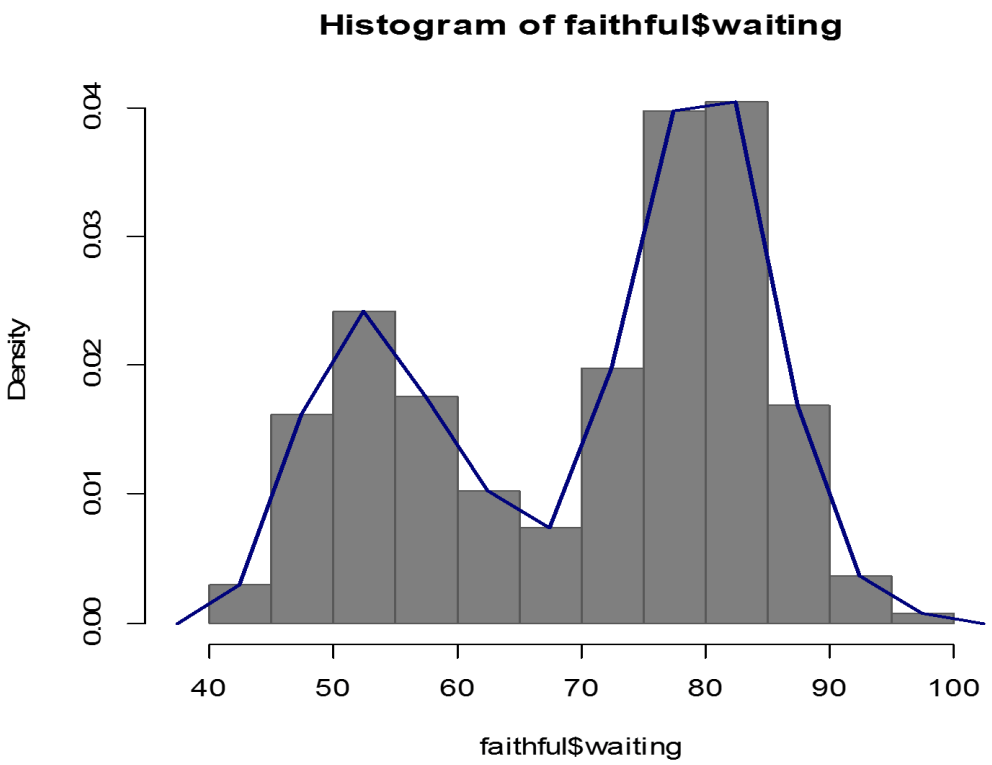

Histogram of faithful$waiting
Frequency
0
10
20
30
40
50
40
50
60
70
80
90
100
faithful$waiting

draw the histogram

>hist(faithful$waiting, prob =TRUE, xlim=range(xx) , border = "gray" , col="gray90")

adds the frequency polygon

>lines(xx, yy, lwd=2, col = "royalblue")

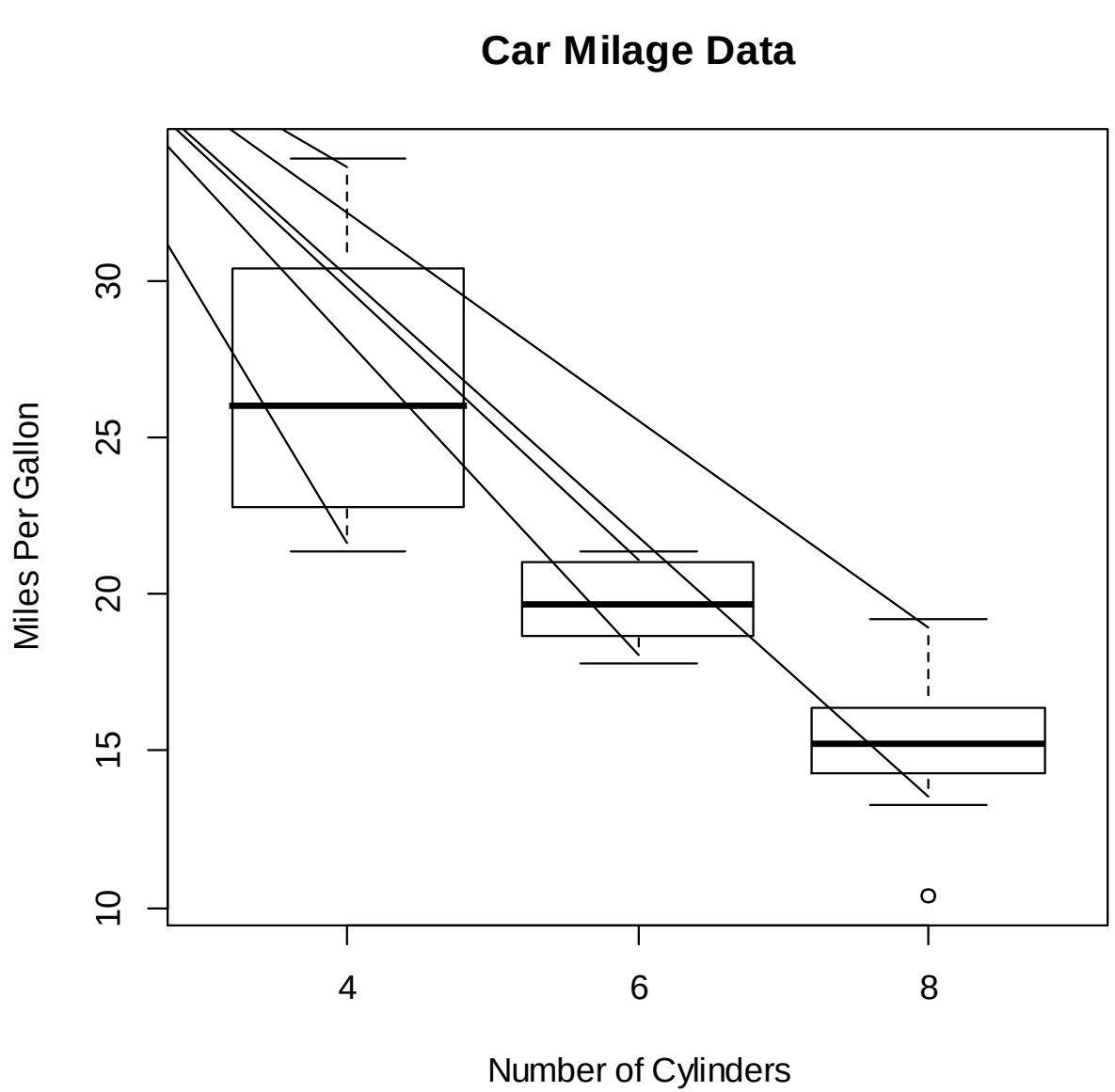


- **boxplot**

boxplot(X) is a box-and-whisker plot with the values of variable X, this is an effective way to summarize larger datasets.

 # Boxplot of MPG by Car Cylinders

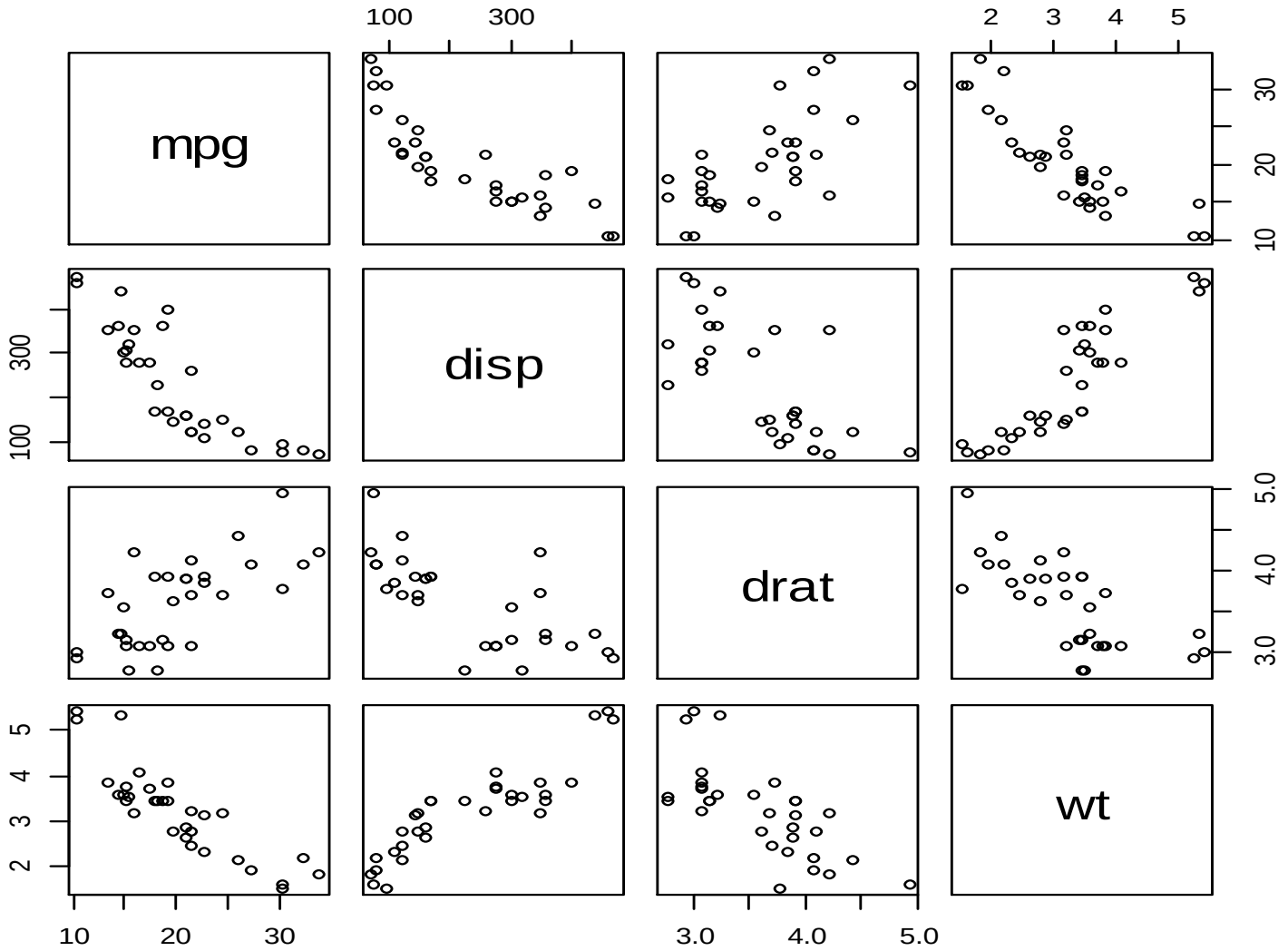
> boxplot(mpg~cyl,data=mtcars, main="Car Milage Data",xlab="Number of Cylinders", ylab="Miles Per Gallon")


Car Milage Data
Miles Per Gallon
10
15
20
25
30
4
6
8
Number of Cylinders

- **Pairs**
- pairs() shows a matrix with all the scatterplots for the columns of variable X

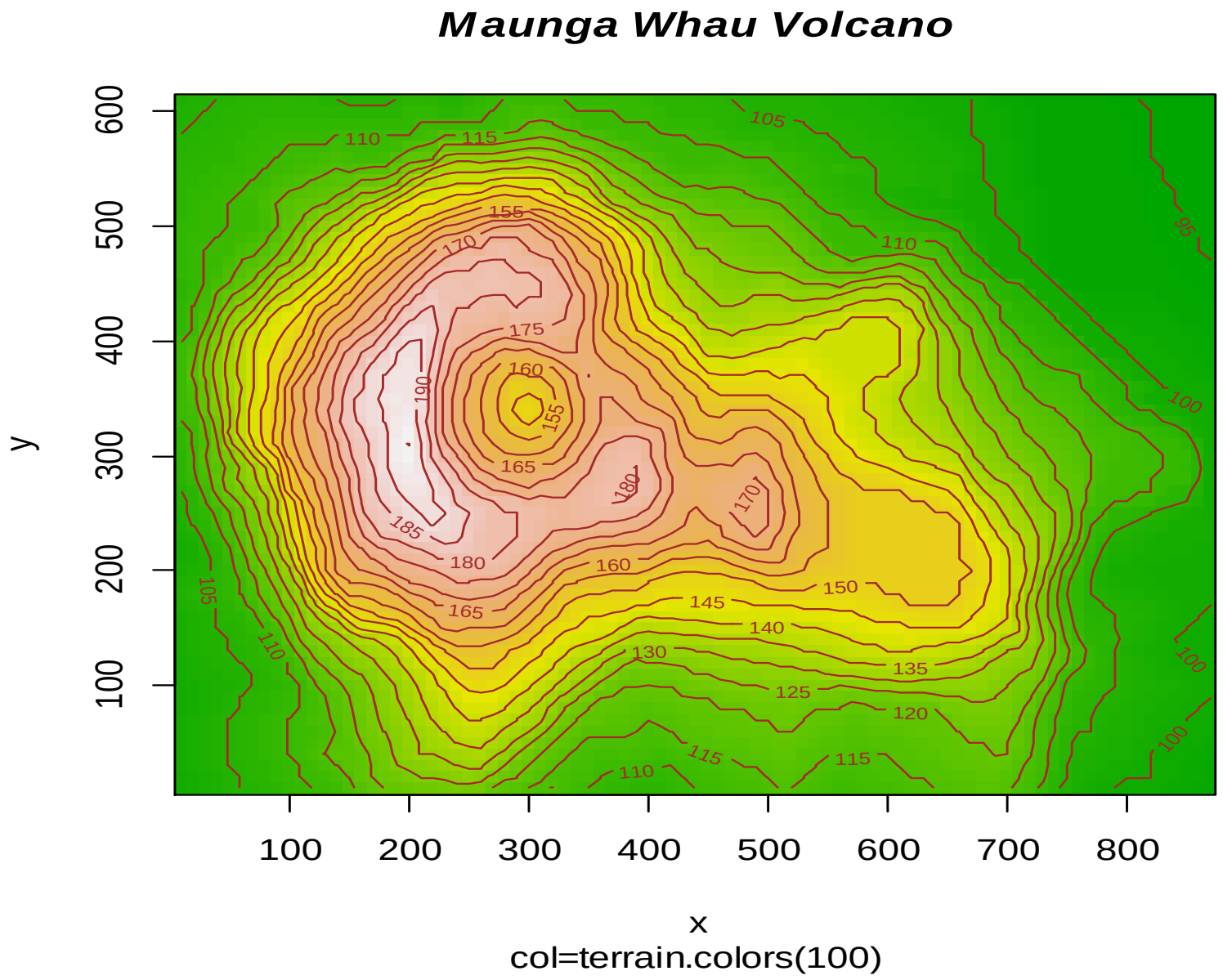
- **pairs(~mpg+disp+drat+wt,data=mtcars, main="Scatterplot Matrix MPG, Displacement,Rear axle ratio,Weight")**

Scatterplot Matrix MPG, Displacement,Rear axle ratio, Weight



- **Contour**
- contour(X,Y,Z) draws a contour plot, with vector X for the rows, vector Y for the columns and matrix X for the data

```
>x <- 10*(1:nrow(volcano)); x.at <- seq(100, 800, by=100)
>y <- 10*(1:ncol(volcano)); y.at <- seq(100, 600, by=100)
# Using Terrain Colors
>image(x, y, volcano, col=terrain.colors(100),axes=FALSE)
>contour(x, y, volcano, levels=seq(90, 200, by=5), add=TRUE, col="brown")
>axis(1, at=x.at)
>axis(2, at=y.at)
>box()
>title(main="Maunga Whau Volcano", sub = "col=terrain.colors(100)", font.main=4)
```


Maunga Whau Volcano
y
x
col=terrain.colors(100)
100
200
300
400
500
600
700
800

- **Persp**

persp(X,Y,Z) draws a 3d graph, with vector X for the rows, vector Y for the columns and matrix X for the data

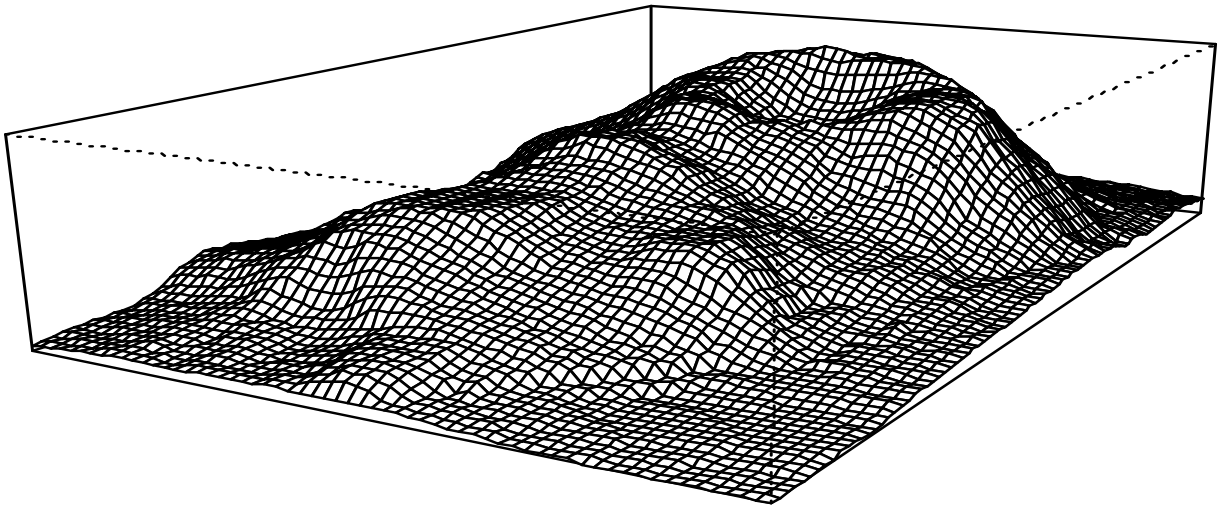
(2) Visualizing a simple DEM model

>z <- 2 * volcano # Exaggerate the relief

>x <- 10 * (1:nrow(z)) # 10 meter spacing (S to N)

>y <- 10 * (1:ncol(z)) # 10 meter spacing (E to W)

>persp(x, y, z, theta = 120, phi = 15, scale = FALSE, axes = FALSE)



- **Tables**

Example:

```
> library(MASS)
>ships
> table(ships$type)
```

```
A B C D E
8 8 8 8 8
```

```
>table(ships$type,ships$year)
```

	60	65	70	75
A	2	2	2	2
B	2	2	2	2
C	2	2	2	2
D	2	2	2	2
E	2	2	2	2

Example :-

```
>library(MASS)
>USArrests
>table(USArrests[,3])
```

```
32 39 44 45 48 50 51 52 53 54 56 57 58 59 60 62 63 65 66 67 68 70 72 73 74 75
 1  1  2  2  2  1  1  1  1  1  1  1  1  1  2  1  1  1  4  2  1  2  2  1  1  1
77 78 80 81 83 85 86 87 89 91
 1  1  4  1  2  1  1  1  1  1
```

>table(cut(USArrests[,3],pretty(USArrests[,3])))

```
(30,40] (40,50] (50,60] (60,70] (70,80] (80,90] (90,100]
      2       7      10      12      11       7        1
```

Example :-

> airquality

> table(airquality[,4],airquality[,5])

>table(cut(airquality[,4],pretty(airquality[,4])),
airquality[,5])

```
             5  6  7  8  9
(50,60]      8  0  0  0  0
(60,70]     16  2  0  0  7
(70,80]      6 18  3 11 14
(80,90]      1  8 25 15  5
(90,100]     0  2  3  5  4
```

Example :-

```
> library(MASS)
> cars
```

```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

```
> tail(cars)
   speed dist
45    23   54
46    24   70
47    24   92
48    24   93
49    24  120
50    25   85
```

```
> head(cars,3)
  speed dist
1     4    2
2     4   10
3     7    4
```

```
> tail(cars,3)
   speed dist
48    24   93
49    24  120
50    25   85
```