Introduction to R

Installation

- http://www.r-project.org
- R-studio is recommended.

Vectors and assignment

- R operates on named data structures.

```
x <- c(1,2,3,4)

x = c(1,2,3,4)

x = 1:4

x = seq(1, 4, by = 1)
```

Vectors and assignment

- Example: "seq"

```
> x1 = seq(1, 10, by = 2)
> x1
[1] 1 3 5 7 9
> x2 = seq(2, 10, by = 2)
> x2
[1]  2  4  6  8 10
> x3 = c(x1, x2)
> x4 = seq(0, 1, by = 0.2)
> x4
[1] 0.0 0.2 0.4 0.6 0.8 1.0
> x5 = seq(0, 1, length = 11)
> x5
[1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

Vectors and assignment

- Example: "rep"

```
> rep(1, 5)
[1] 1 1 1 1 1
> rep( c(2,3), 3)
[1] 2 3 2 3 2 3
> ABC = c("A", "B", "C")
> rep(ABC, 2)
[1] "A" "B" "C" "A" "B" "C"
> rep(ABC, times = 2)
[1] "A" "B" "C" "A" "B" "C"
> rep(ABC, times = c(4,2,1))
[1] "A" "A" "A" "A" "B" "B" "C"
> rep(ABC, each = 2)
[1] "A" "A" "B" "B" "C" "C"
> rep(ABC, each = 2, times = 2)
[1] "A" "A" "B" "B" "C" "C" "A" "A" "B" "B" "C" "C"
```

Vectors and assignment

- Example: Elements of a vector.

```
> x = c(1,2,3,4)
> x[1]
[1] 1
> x[c(1,3)]
[1] 1 3
> x[1:3]
[1] 1 2 3
> x[-1]
[1] 2 3 4
> x[c(T,F,T,F)]
[1] 1 3
```

Factor

```
> x = c("high", "medium", "low", "medium", "high")
> class(x)
[1] "character"
> xf = factor(x)
> class(xf)
[1] "factor"
> xf
[1] high   medium low    medium high
Levels: high low medium
> as.numeric(xf)
[1] 1 3 2 3 1
```

Factor

- Re-arrange the factor elements.

```
> xf2 = factor(x, levels = c("low", "medium", "high"))
> xf2
[1] high   medium low    medium high
Levels: low medium high
> as.numeric(xf2)
[1] 3 2 1 2 3
```

Matrix

- Matrix Generation

```
> A = matrix(1:9, nrow = 3)
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> A = matrix(1:9, ncol = 3)
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Matrix

Matrix Calculation

```
J = matrix(1, nrow = 3, ncol = 3)
> J
[,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
> I = diag(3)
> I
[,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> 0.3*J
[,1] [,2] [,3]
[1,]  0.3  0.3  0.3
[2,]  0.3  0.3  0.3
[3,]  0.3  0.3  0.3
> 0.7*I
[,1] [,2] [,3]
[1,]  0.7  0.0  0.0
[2,]  0.0  0.7  0.0
[3,]  0.0  0.0  0.7
```

Matrix

Matrix Inversion

```
A = 0.3*J + 0.7*I
> A
[,1] [,2] [,3]
[1,]  1.0  0.3  0.3
[2,]  0.3  1.0  0.3
[3,]  0.3  0.3  1.0
> Ainv = solve(A)
> Ainv
[,1]       [,2]       [,3]
[1,]  1.1607143 -0.2678571 -0.2678571
[2,] -0.2678571  1.1607143 -0.2678571
[3,] -0.2678571 -0.2678571  1.1607143
> A %*% Ainv
[,1]         [,2] [,3]
[1,]  1.000000e+00 1.387779e-17    0
[2,] -5.551115e-17 1.000000e+00    0
[3,] -5.551115e-17 0.000000e+00    1
```

Matrix

- Matrix Multiplication and Determinant

```
b = c(1,0.5,1.5)
> A %*% b
     [,1]
[1,] 1.60
[2,] 1.25
[3,] 1.95
> det(A)
[1] 0.784
```

Matrix

Matrix Transpose

```
> B = matrix(1:9, 3, 3)
> B
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> C = A+B
> C
     [,1] [,2] [,3]
[1,]  2.0  4.3  7.3
[2,]  2.3  6.0  8.3
[3,]  3.3  6.3 10.0
> t(C)
     [,1] [,2] [,3]
[1,]  2.0  2.3  3.3
[2,]  4.3  6.0  6.3
[3,]  7.3  8.3 10.0
```

Matrix

Matrix Elements

```
> C
     [,1] [,2] [,3]
[1,]  2.0  4.3  7.3
[2,]  2.3  6.0  8.3
[3,]  3.3  6.3 10.0
> C[1,1]
[1] 2
> C[-1,]
     [,1] [,2] [,3]
[1,]  2.3  6.0  8.3
[2,]  3.3  6.3 10.0
> C[,-2]
     [,1] [,2]
[1,]  2.0  7.3
[2,]  2.3  8.3
[3,]  3.3 10.0
```

List

```
> MyList = list(Eng = eng, Math = math, Name = "Score")
> MyList
$Eng
[1]  60  70  80  90 100

$Math
[1] 65 73 81 89 97

$Name
[1] "Score"

> MyList$Eng
[1]  60  70  80  90 100
> MyList[[1]]
[1]  60  70  80  90 100
> MyList$Eng[2]
[1] 70
> MyList[[1]][2]
[1] 70
```

Data Frame

```
> MyData = data.frame(Eng = eng, Math = math)
> MyData
Eng Math
1  60   65
2  70   73
3  80   81
4  90   89
5 100   97
> dim(MyData)
[1] 5 2
> MyData$Math
[1] 65 73 81 89 97
> MyData[,2]
[1] 65 73 81 89 97
```

Data Frame

- Iris Example

```
data(iris)
> class(iris)
[1] "data.frame"
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Import and Export Data

- Import Data

Import and Export Data

- Export Data

```
> write.table(iris, "iris.txt")
```

Import and Export Data

- Import Data

```
> data = read.table("iris.txt")
> head(data)
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Data Exploration

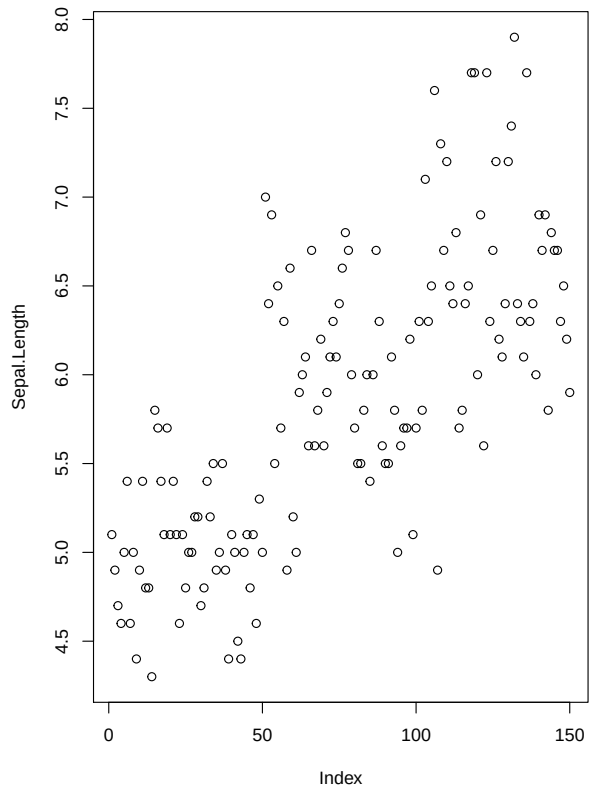
- Exploratory Data Analysis (EDA)

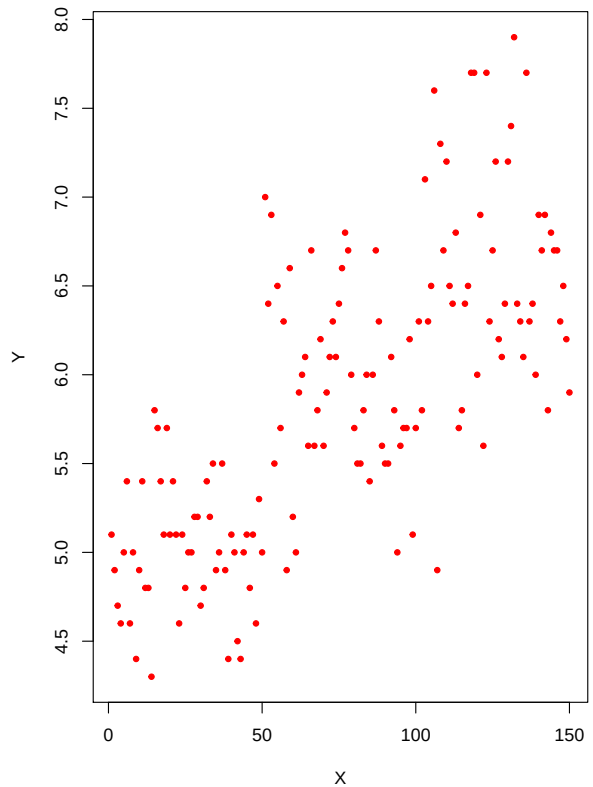
```
median.data= median(data$Sepal.Length)
> median.data
[1] 5.8
> mean.data = mean(data$Sepal.Length)
> mean.data
[1] 5.843333
> var.data= var(data$Sepal.Length)
> var.data
[1] 0.6856935
> median.data= median(data$Sepal.Length)
> median.data
[1] 5.8
> summary(data$Sepal.Length)
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
4.300   5.100   5.800   5.843   6.400   7.900
```

Graphics

- Scatter Plot

```
> plot(Sepal.Length)
> plot(Sepal.Length, pch = 20, col = "red", xlab="X", ylab ="Y")
```

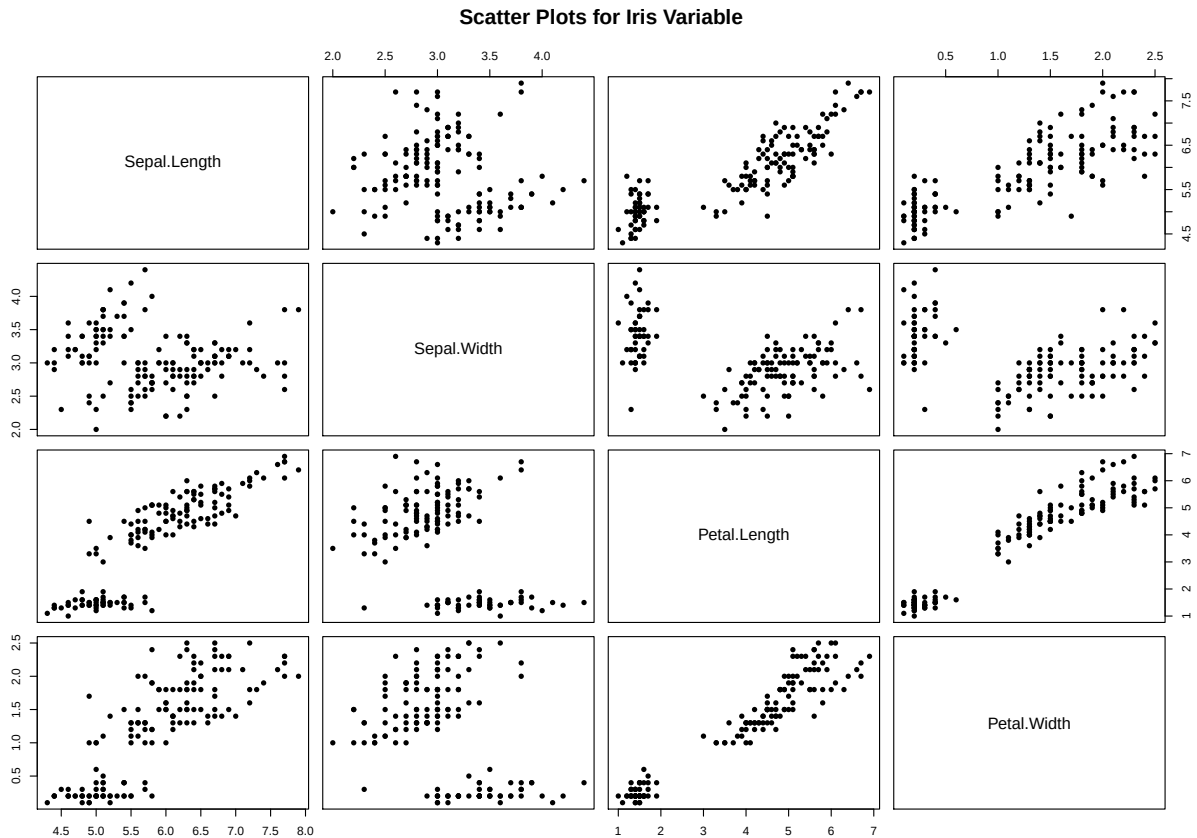




Graphics

- Scatter Plot

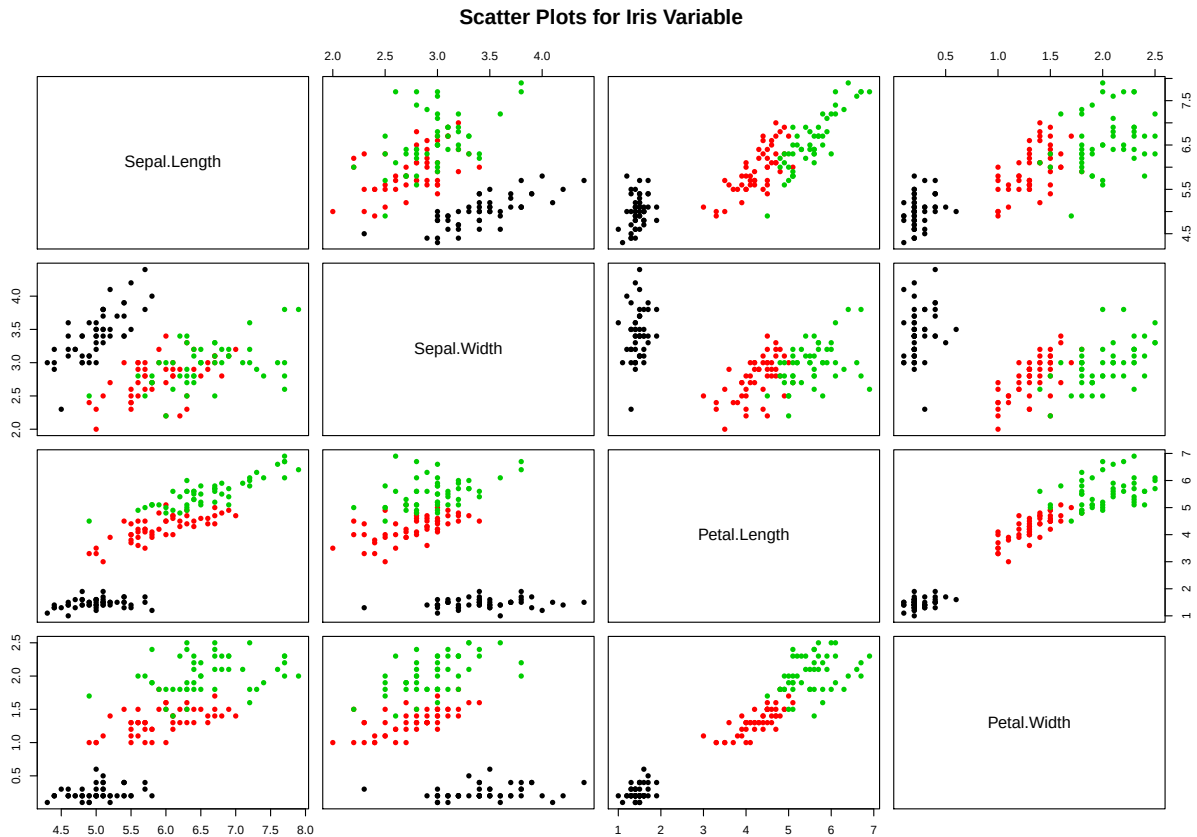
```
plot(iris[,1:4], main="Scatter Plots for Iris Variable", pch=16)
```



Graphics

- Scatter Plot

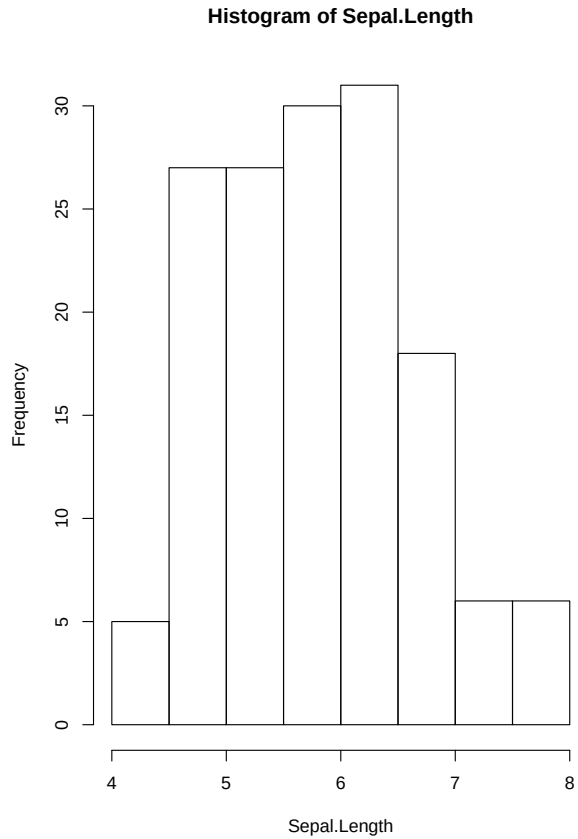
```
plot(iris[,-5], main="Scatter Plots for Iris Variable", pch=16,
col = as.numeric(iris[,5]) )
```

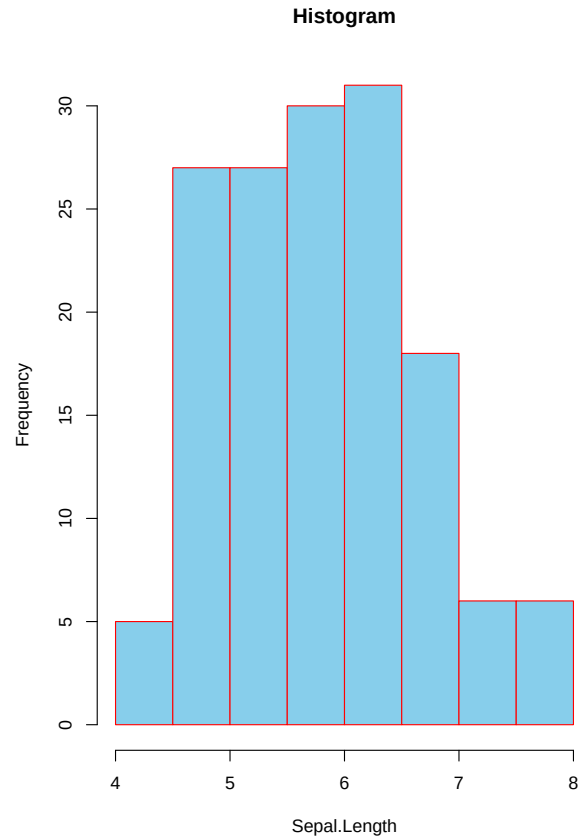


Graphics

- Histogram

```
> hist(Sepal.Length)
> hist(Sepal.Length, col = "skyblue", border = "red", main = "Histogram")
```

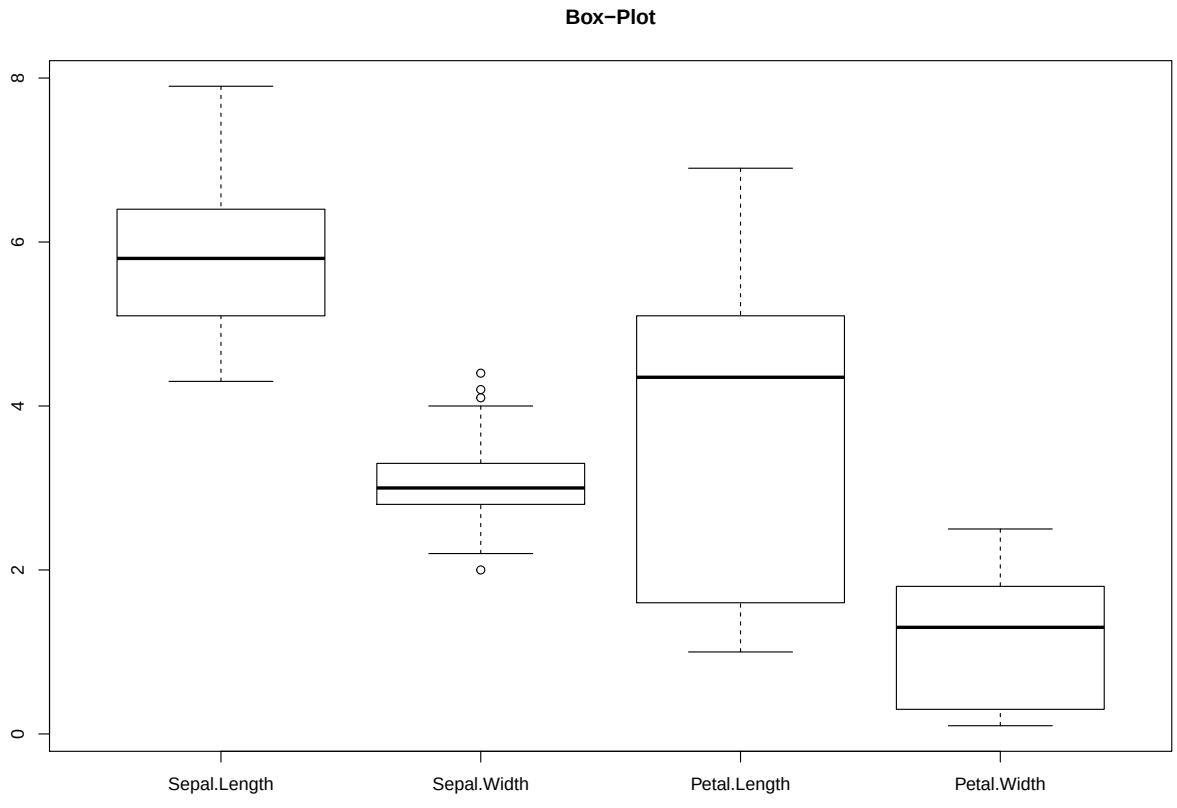




Graphics

- Box-Plot

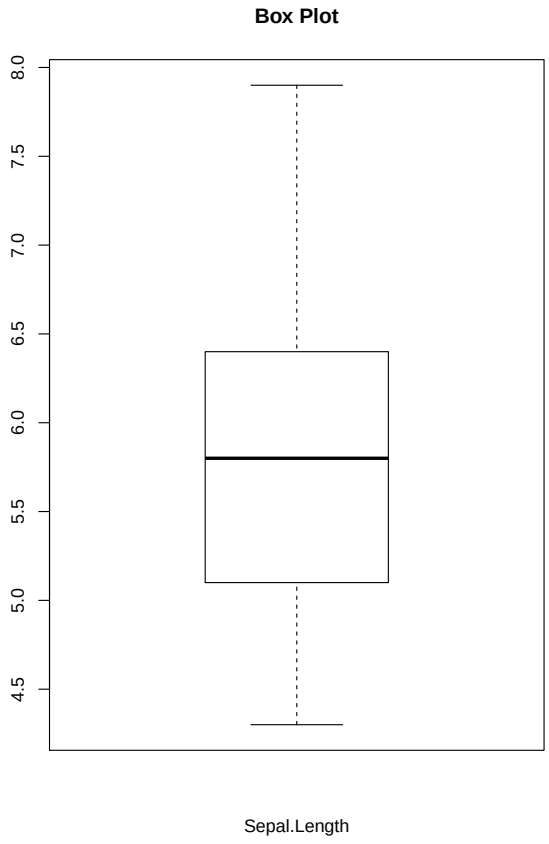
```
> boxplot(iris[,-5], main = "Box-Plot")
```

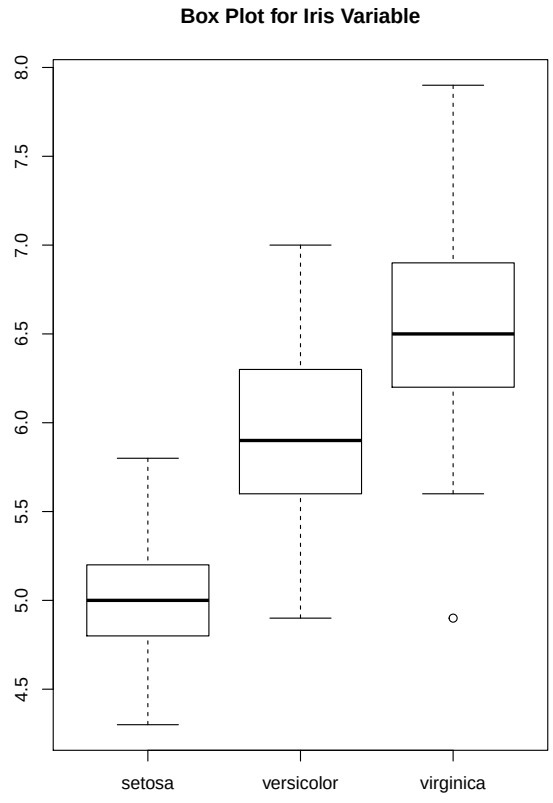


Graphics

- Box-Plot

```
> boxplot(iris[,1], xlab = "Sepal.Length", main = "Box Plot")
> boxplot(iris[,1] ~ iris[,5], main = "Box Plot for Iris Variable")
```

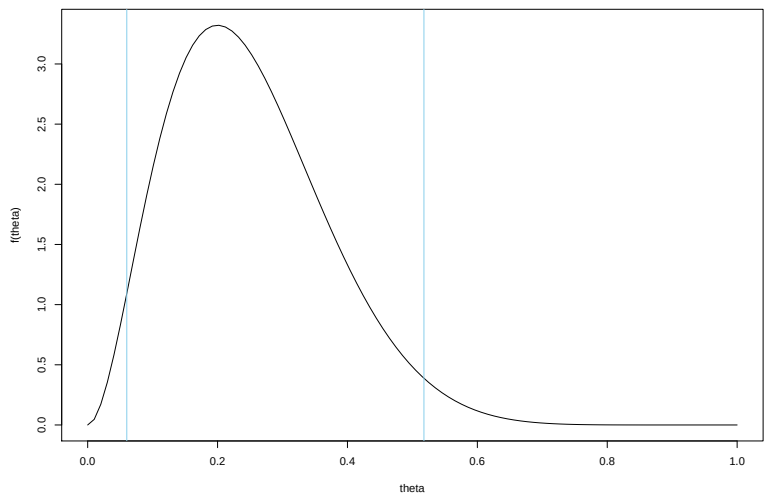




Graphics

- Beta(3,9) density function

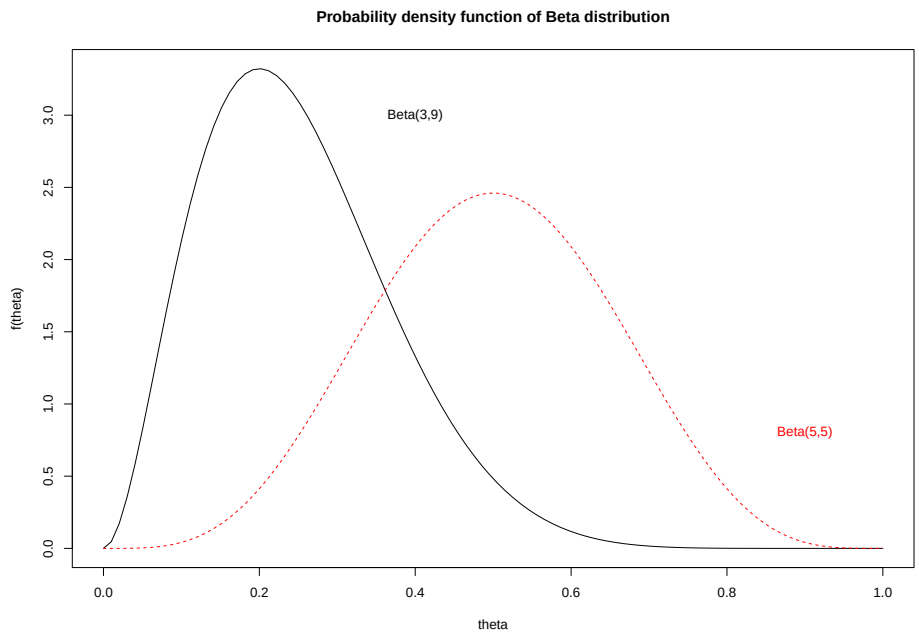
```
> par(mfrow = c(1,1))
> theta = seq(0,1,length = 100)
> ftheta = dbeta(theta,3,9)
> plot(theta, ftheta, type = "l", xlab ="theta", ylab="f(theta)")
> abline(v=qbeta(c(0.025,0.975),3,9), col="skyblue")
> qbeta(c(0.025,0.975),3,9)
[1] 0.06021773 0.51775585
```



Graphics

- Beta(3,9) and Beta(5,5) density function

```
> plot(theta, ftheta, type = "l", xlab ="theta", ylab="f(theta)")
> ftheta2 = dbeta(theta,5,5)
> lines(theta,ftheta2,lty=2,col=2)
> text(0.9,0.8,"Beta(5,5)",col=2)
> text(0.4,3.0,"Beta(3,9)")
> title("Probability density function of Beta distribution")
```



Functions

- Addition

```
> Addition = function(x, y){
+   x + y
+ }
> Addition(1,2)
[1] 3
> Addition(1,3)
[1] 4
```

Package

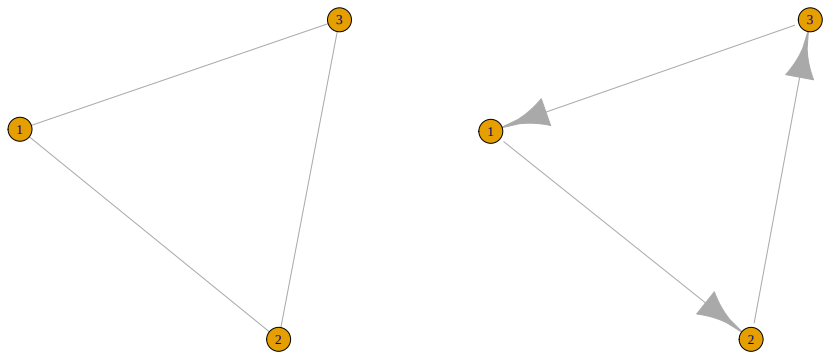
- Install Packages

```
> install.packages("MCMCpack")
> library(MCMCpack)
Loading required package: coda
Loading required package: MASS
##
## Markov Chain Monte Carlo Package (MCMCpack)
## Copyright (C) 2003-2018 Andrew D. Martin, Kevin M. Quinn, and Jong Hee
##
## Support provided by the U.S. National Science Foundation
## (Grants SES-0350646 and SES-0350613)
```

Package: igraph

- Graph

```
> library(igraph)
> par(mfrow = c(1,2))
> g1 <- graph( edges=c(1,2, 2,3, 3, 1), n=3, directed=F )
> plot(g1) # A simple plot of the network
> g2 <- graph( edges=c(1,2, 2,3, 3, 1), n=3, directed=T )
> plot(g2) # A simple plot of the directed graph
```



Package: igraph

- Network

```
> par(mfrow = c(1,1), mar = c(2,2,2,2))
> plot(zach, vertex.size=15)
```