R Introdcution

Week1

Part I : Introduction of R Basics

R is statistical programming language. We will apply R functions and packages to our data sets in order to analyze the data and then we will interpret the output from these functions. There are lots of quick tutorials online. The An Introduction to R from https://cran.r-project.org/ is very useful.

Here, a few key components will be briefly introducted. We will illustrate how to use R for the data analysis of examples or homework problems in the following week. You will learn how to use R for analyzing the DOE data through hand-on examples and writing your own programs to solve problems.

1. R can be used a calculator: we assign variable with "=" or "<-".

```
X=3
Y=5
X+Y
```

```
## [1] 8
```

2. We use help() to get the help document for a function or build-in data set in R

3. We need to use install.packages() to install new packages not from the standard distribution.

4. The commonly types of variables that we will use include integer/numeric, character, factor, logic.

```
X <- 1:5
X
```

```
## [1] 1 2 3 4 5
```

```
class(X)
```

```
## [1] "integer"
```

```
X <- 1:5+0.1
X
```

```
## [1] 1.1 2.1 3.1 4.1 5.1
```

```
class(X)
```

```
## [1] "numeric"
```

```r
(X <- c("male", "female", "female", "male", "female"))
```

```
## [1] "male"   "female" "female" "male"   "female"
```

```r
class(X)
```

```
## [1] "character"
```

```r
(X <- factor(c("male", "female", "female", "male", "female")))
```

```
## [1] male   female female male   female
## Levels: female male
```

```r
class(X)
```

```
## [1] "factor"
```

```r
X<- (5>3)
X
```

```
## [1] TRUE
```

```r
class(X)
```

```
## [1] "logical"
```

5. The commonly used data types in this course are vector, matrix and data frames.

5.1 Vectors

We used the colon operator, :, for creating sequences from one number to another, and the **c** function for concatenating values and vectors to create longer vectors.

```r
8.5:4.5 #sequence of numbers from 8.5 down to 4.5
```

```
## [1] 8.5 7.5 6.5 5.5 4.5
```

```r
c(1, 1:3, c(5, 8), 13) #values concatenated into single vector
```

```
## [1]  1  1  2  3  5  8 13
```

5.2 Matrix

The vector variables that we have looked at so far are one-dimensional objects, since they have length but no other dimensions. Matrices are two-dimensional arrays of data of the same type.

```r
X <- matrix (1:12, nrow = 4)
X
```

```
##      [,1] [,2] [,3]
## [1,]    1    5    9
## [2,]    2    6   10
## [3,]    3    7   11
## [4,]    4    8   12
```

```r
class(X)
```

```
## [1] "matrix"
```

```r
dim(X)
```

```
## [1] 4 3
```

5.3 Data frames

Data frames are used to store spreadsheet-like data. They can either be thought of as matrices where each column can store a different type of data.

```r
mydata <- data.frame( x = letters[1:5],
                      y = rnorm(5))
mydata
```

```
##   x         y
## 1 a 0.8259921
## 2 b 0.1585643
## 3 c 0.4132164
## 4 d 0.1519460
## 5 e 1.2077406
```

```r
dim(mydata)
```

```
## [1] 5 2
```

```r
class(mydata)
```

```
## [1] "data.frame"
```

6. One dimension data : quick summary

```r
X <- rnorm(100)
length(X)
```

```
## [1] 100
```

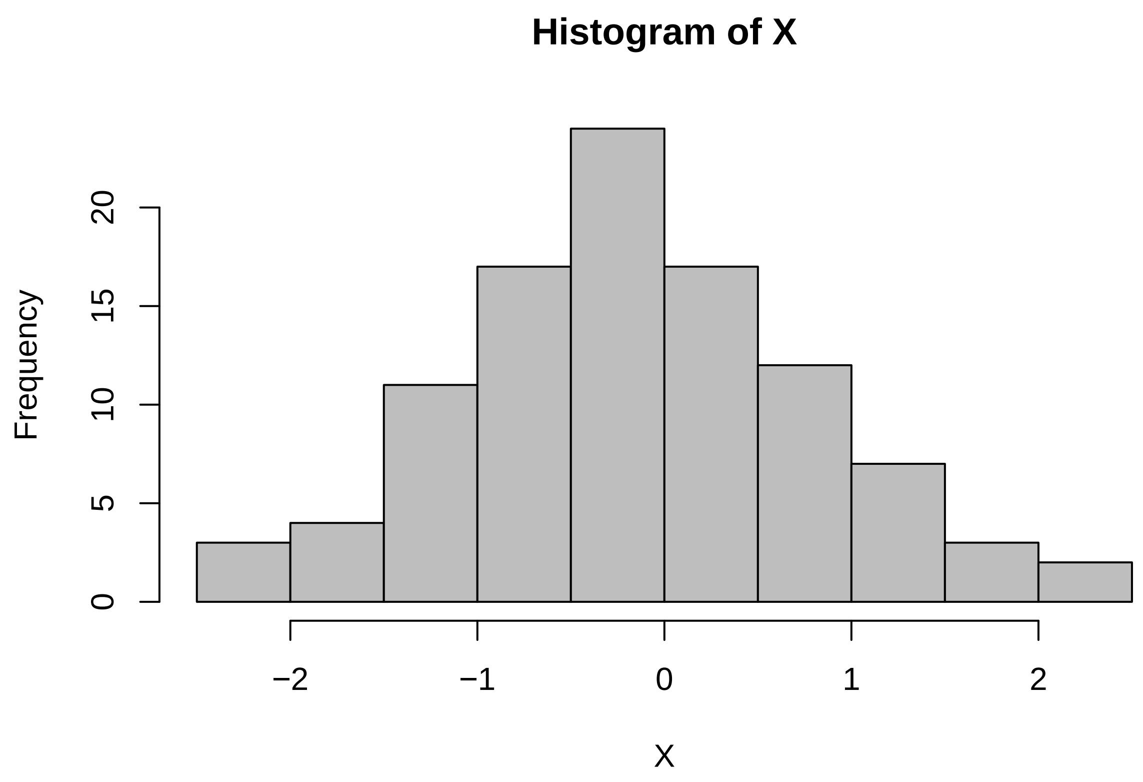
```r
summary(X)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.4366 -0.7716 -0.2186 -0.1512  0.4549  2.3780
```

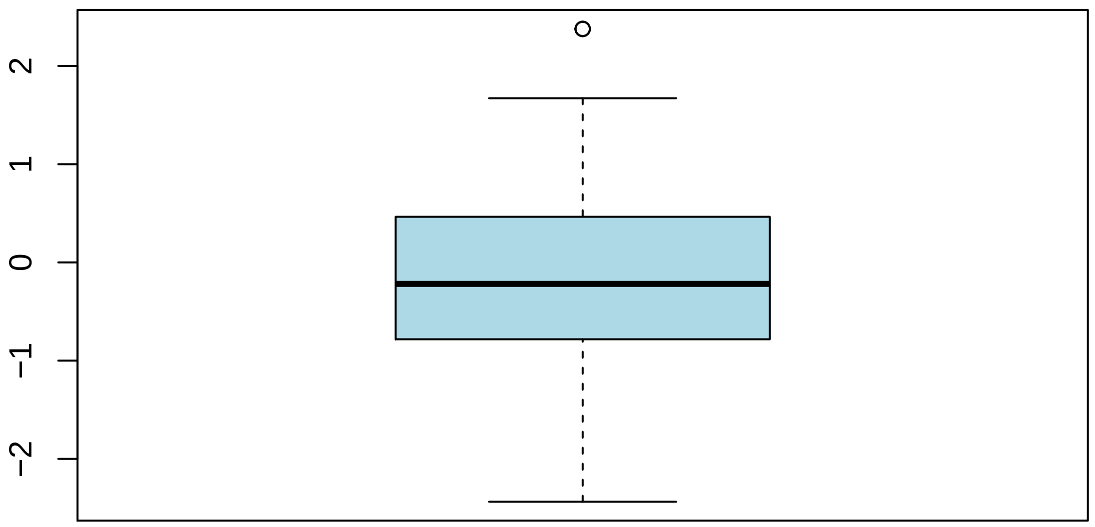
```r
sd(X)
```

```
## [1] 0.9525934
```

```r
hist(X, col='gray')
```



```r
boxplot(X, col='light blue')
```



7. Two dimension data, matrix operation for matrix x

- **t(x)** transpose
- **diag(x)** diagonal
- %*% matrix multiplication
- **solve(a)** matrix inverse of a
- **rowSum(x)** sum of rows for a matrix-like object;
- **colSums(x)** sum for columns
- **rowMeans(x)** fast version of row means
- **colMeans(x)** id. for columns

8. Data frame: data summary and simple linear model

Example: cars is a R-buid-in data set. The data give the speed of cars and the distances taken to stop Try > help(cars).

```
str(cars)
```

```
## 'data.frame':    50 obs. of  2 variables:
##  $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
##  $ dist : num  2 10 4 22 16 10 18 26 34 17 ...
```

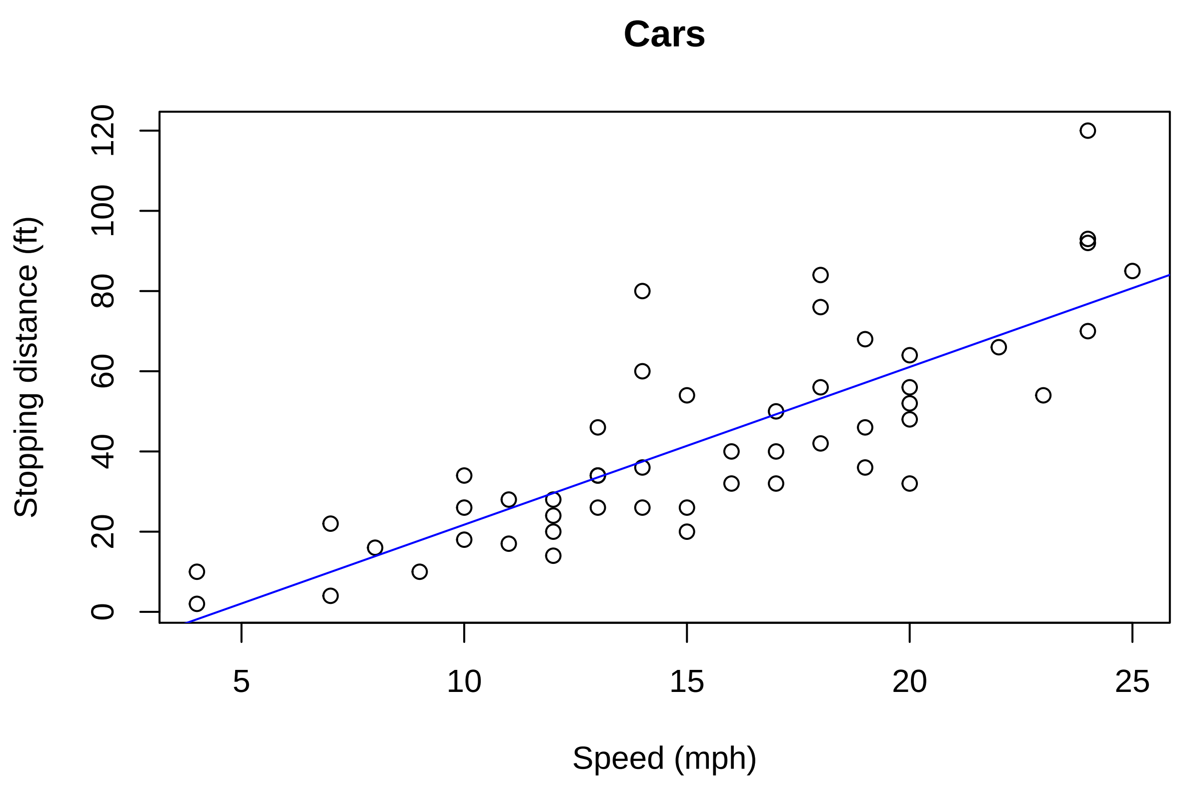
```r
head(cars)
```

```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

```r
summary(cars)
```

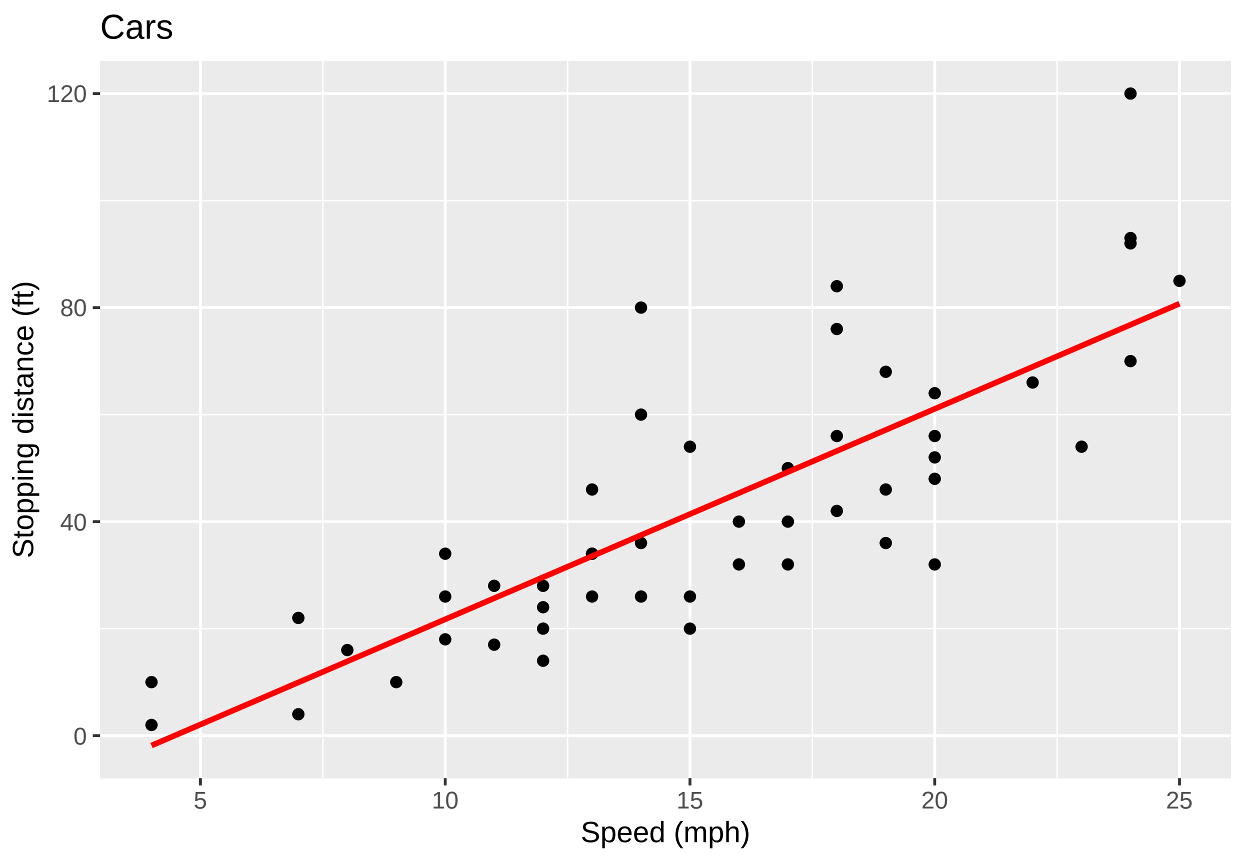
```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

```r
# plot scatterplot and add least square fit
plot( dist~speed, data=cars, main='Cars', xlab="Speed (mph)", ylab="Stopping distance (ft)")
abline(lsfit(cars$speed, cars$dist)$coef, col=4)
```



If you want to use more advanced graphic function in R, you can install the ggplot2 package.

```
if (!"ggplot2"  %in% installed.packages()) install.packages("ggplot2")
library(ggplot2)
ggplot(cars, aes(x=speed, y=dist)) +
  geom_point()+
  stat_smooth(method="lm", col='red', se=FALSE)+
  labs(title='Cars', x="Speed (mph)", y="Stopping distance (ft)")
```



Part II : Illustrate of Example in Chapter 15.5

1. Step 1. Read the data. You can go to stat3119/Week1 folder to save the data, then run with **read.table(file.choose(), header=F)**, read the file directly from URL.

```
#Ex15 <- read.table(file.choose(), header=F) # find the file location from file browser
Ex15 <- read.table(url("https://raw.githubusercontent.com/npmldabook/Stat3119/master/Week1/CH15TA01.txt
dim(Ex15)
```

```
## [1] 20  4
```

```
head(Ex15, 2) # show 2 lines of data
```

```
##   V1   V2   V3    V4
## 1  1 0.59 0.43 -0.16
## 2  2 0.69 0.53 -0.16
```

2. Name the variables and check data.

```r
names(Ex15) <- c("subject", "Control","Exp", "Dif")
head(Ex15, 2)
```

```
##   subject Control  Exp   Dif
## 1       1    0.59 0.43 -0.16
## 2       2    0.69 0.53 -0.16
```

3. Run the statistical tests for paried comparison. Here, we run paired comparison for control and experimental groups with a paired t-test. The results match the textbook (p. 671).

```r
t.test( Ex15$Control ,Ex15$Exp, paired=T )
```

```
## 
##  Paired t-test
## 
## data:  Ex15$Control and Ex15$Exp
## t = 17.1, df = 19, p-value = 5.377e-13
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1680611 0.2149389
## sample estimates:
## mean of the differences
##                  0.1915
```

4. This is a randomized complete block design with block size of 2 and 20 subjects as the main factor (20 levels). We will revisit this example later. Basically, we need to transform this data to specify the factor X and outcome Y, then we run the ANOVA to get the test reesults.

```r
# Data transformation
Ex15v2 <- data.frame( Treatment=c(rep(0,20), rep(1,20)), Subject=factor(c(1:20, 1:20)),
                     Y= c( Ex15$Control ,Ex15$Exp))
head(Ex15v2)
```

```
##   Treatment Subject    Y
## 1         0       1 0.59
## 2         0       2 0.69
## 3         0       3 0.82
## 4         0       4 0.80
## 5         0       5 0.66
## 6         0       6 0.67
```

```r
summary(aov(Y~  Treatment+Subject, data=Ex15v2 ))
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)    
## Treatment    1 0.3667  0.3667 292.424 5.38e-13 ***
## Subject     19 0.2120  0.0112   8.898 7.33e-06 ***
## Residuals   19 0.0238  0.0013                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```