Unit -1

1.1 Introduction to R- Language

R is a programming language and software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. The R language is widely used among statisticians and data miners for developing statistical software and data analysis. Polls, surveys of data miners, and studies of scholarly literature databases show that R's popularity has increased substantially in recent years.

R is a GNU package. The source code for the R software environment is written primarily in C, FORTRAN, and R.

R is freely available under the GNU General Public License, and pre-compiled binary versions are provided for various operating systems. While R has a command line interface, there are several graphical front-ends available.

R is an implementation of the S programming language combined with lexical scoping semantics inspired by Scheme. S was created by John Chambers while at Bell Labs. There are some important differences, but much of the code written for S runs unaltered.

R was created by **Ross Ihaka** and **Robert** Gentleman at the University of Auckland, New Zealand, and is currently developed by the *R Development Core Team*, of which Chambers is a member. R is named partly after the first names of the first two R authors and partly as a play on the name of S. The project was conceived in 1992, with an initial version released in 1994 and a stable beta version in 2000.

R and its libraries implement a wide variety of statistical and graphical techniques, including linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering, and others.

R is easily extensible through functions and extensions, and the R community is noted for its active contributions in terms of packages. Many of R's standard functions are written in R itself, which makes it easy for users to follow the algorithmic choices made.

Strength of R is static graphics, which can produce publication-quality graphs, including mathematical symbols. Dynamic and interactive graphics are available through additional packages.

The general consensus is that R compares well with other popular statistical packages, such as SAS, SPSS, and Stata. In a comparison of all basic features for statistical software R is heads up with the best of statistical software.

In January 2009, the *New York Times* ran an article about R gaining acceptance among data analysts and presenting a potential threat for the market share occupied by commercial statistical packages, such as SAS.

1.2 R as a calculator

R can be used as a calculator. The basic operations are + (addition), - (subtraction), * (multiplication), and / (division). It is also used to calculate %% (modular), ^ (power). For example

```
> 5^3
[1] 125

>5 %% 3
[1] 2
```

1.3 Measures of central tendency

We explain how to compute measures of central tendency in R. Measures of central tendency are called averages. The most frequently encountered averages are arithmetic mean, median and mode.

Example 1.1:

The age of seven people are given as 25, 35, 45, 56, 25, 89 and 65. Find mean, median and mode of the age.

Solution:

```
 age <-c(25,35,45,56,25,89,65)
 age
 mean (age)
[1] 48.57143

 median(age)
[1] 45

 x<-table(age)
 mode<-which(x==max(x))
 mode
 25
 1
```

Observe the out. The mode is 25, which is the first distinct value in the ordered series.

1.4 Quartiles, deciles and percentiles

These are some more measures of location. Median is the set of measurements in the value that divides the distribution in to two parts, each containing 50% of the observations. In the same way, quartiles Q_1, Q_2 and Q_3 are

the three values that divide the distribution in to four equal parts. The deciles are nine values that divide the distributions in to ten equal parts. The percentiles are the ninety nine values that divide the distribution in to hundred equal parts.

Example 1.2:
Obtain three quartiles, fifth decile and fiftieth percentile of the data given below:

Marks in Statistics	36	35	30	36	27	40	41	45	46	49

Solution

```
>marks <-c(36,35,30,36,27,40,41,45,46,49)
>marks
>a=sort(marks)
>Q1=quantile(a,0.25)
>Q1
25%
35.25

>Q2=quantile(a,0.5)
>Q2
50%
38

>Q3=quantile(a,0.75)
>Q3
75%
44

>D5=quantile(a,0.5)
>D5
50%
38

>P50=quantile(a,0.50)
>P50
50%
38
```

1.5 Measures of Dispersion

In statistics, dispersion (also called variability, scatter, or spread) denotes how stretched or squeezed a distribution (theoretical or that underlying a statistical sample) is. Common examples of measures of statistical dispersion are the Range, variance, standard deviation and inter quartile range etc..

Example 1.3:
Marks out of 50 in a subject of 12 students, in a class are given as follows: 12, 18, 20, 12, 16, 14, 30, 32, 28, 12, 12 and 35. Obtain Range, variance, standard deviation and inter quartile range.

Solution:

```
>marks<- c(12,18,20,12,16,14,30,32,28,12,12,35)
>marks
>range<-max(marks)- min(marks)
>range
[1] 23

>var(marks)
[1] 76.81061

>sd(marks)
[1] 8.764166

>a=sort(marks)
>Q3=quantile(a,0.75)
>Q1=quantile(a,0.25)
>IQR=Q3-Q1
>IQR
75%
16.5
```

1.6 Graphical representation of data

Statistics is a special subject that deals with large (usually) numerical data. The statistical data can be represented graphically. In fact, the graphical representation of statistical data is an essential step during statistical analysis. Statistical surveys and experiments provide valuable information about numerical scores. For better understanding and making conclusions and interpretations, the data should be managed and organized in a systematic form.

A graph is the representation of data by using graphical symbols such as bars, pie slices, dots etc. A graph does represent a numerical data in the form of a qualitative structure and provides important information.

Now we study about various types of graphical representations of the data.

1.6.1 Bar chart

A bar graph is a very frequently used graph in statistics. A bar graph is a type of graph which contains rectangles or rectangular bars. The lengths of these bars should be proportional to the numerical values represented by them. In bar

graph, the bars may be plotted either horizontally or vertically. But a vertical bar graph (also known as column bar graph) is used more than a horizontal one.

Example 1.4:

Annual sales (in lakhs of Rs.) of a pharmaceutical firm for 6 year are given below.

Year	1995	1996	1997	1998	1999	2000
Annual sales	15	25	27	28	26	26.6

Represent the data by Bar chart.

Solution:-

```
>year=(1995:2000)
>year
>sales=c(15,25,27,28,26,26.6)
>sales
>sales.year=data.frame(year,sales)
>sales.year
  year sales
1 1995  15.0
2 1996  25.0
3 1997  27.0
4 1998  28.0
5 1999  26.0
6 2000  26.6

>attach(sales.year)
>barplot(sales,xlab="year",ylab="sales",main="Bar chart",col="green")
```

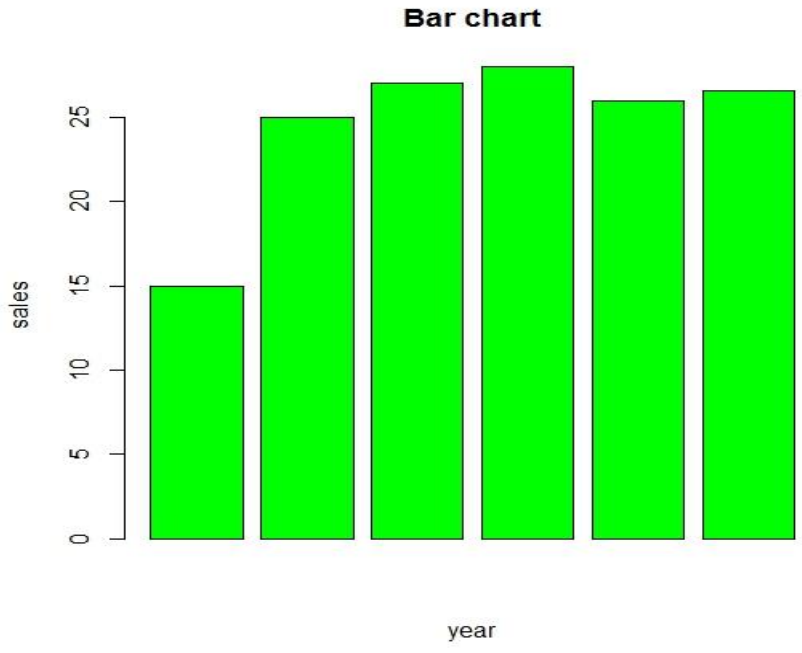


Figure 1.1 Bar Diagram

1.6.2 Pie chart

A circle is used in a pie chart to represent the whole, and "slices" are used to represent the categories, one slice for each category. The size of a slice is proportional to relative frequency of the corresponding category.

Example 1.5:

The tax revenue of Indian (in crores of Rs.), provide in 1984-85 budget, when broken into various sources are given below. Represent the data by a pie chart.

Sources	Excise	Customs	Corporation tax	Income Tax	Others
Tax Revenue	6526	7108	2568	560	763

Solution:

```
>tax=c(6526,7108,2568,560,763)
>tax
>names(tax)=c("Excise","Customs","Corporation tax","Income tax","Other")
>names(tax)
>pie(tax,main="The tax revenue of India",col=c("red","orange","green","white","pink"))
```

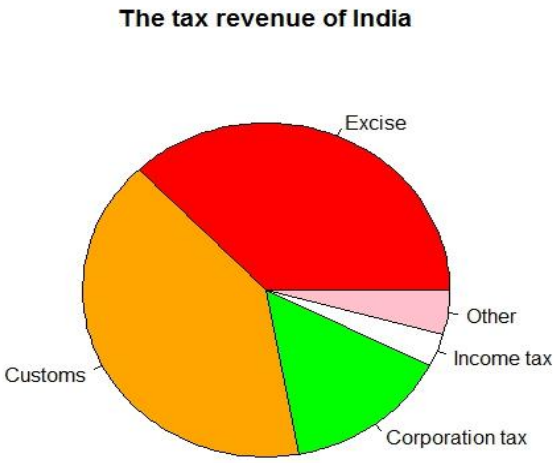


Figure 1.2: Pie Chart

1.6.3 Box plot

Example 1.6:
Titanium content in an aircraft grade is an of 20 test coupons reveals the following titanium content (in %).
8.32,8.05,8.93,8.65,8.25,8.46,8.52,8.35,8.36,8.41,8.42,8.30,8.71, 8.75, 8.60, 8.83, 8.50, 8.38, 8.29, 8.46.
Represent data by Box plot.

Solution:

```
>titanium=c(8.32,8.05,8.93,8.65,8.25,8.46,8.52,8.35,8.36,8.41,8.42,8.30,8.71,8.75,
8.60,8.83,8.50,8.38,8.29,8.46)
>titanium
>boxplot (titanium, main="Box plot", col=”green”)
```

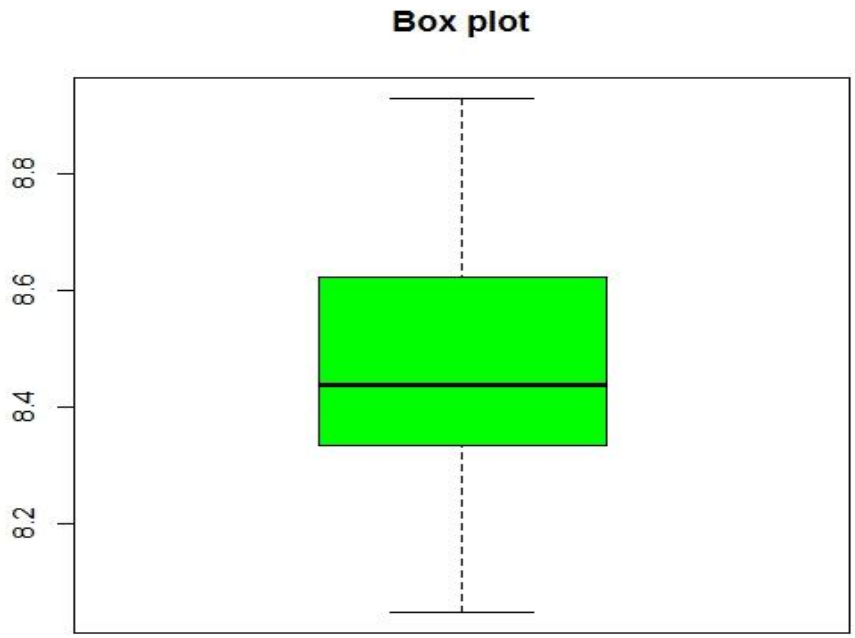


Figure 1.3: Box plot

1.6.4 Histogram

Example 1.7

Titanium content in an aircraft grade is an of 20 test coupons reveals the following titanium content (in %)

Grade:8.32,8.05,8.93,8.65,8.25,8.46,8.52,8.35,8.36,8.41,8.42,8.30,8.71,8.75,8.60,8.83,8.50,8.38,8.29,8.46. Represent data by Histogram.

Solution:

```
>titanium=c(8.32,8.05,8.93,8.65,8.25,8.46,8.52,8.35,8.36,8.41,8.42,8.30,8.71,8.75,8.60,8.83,8.50,8.38,8.29,8.46)
>titanium
>hist(titanium, main="Histogram", col="green")
```

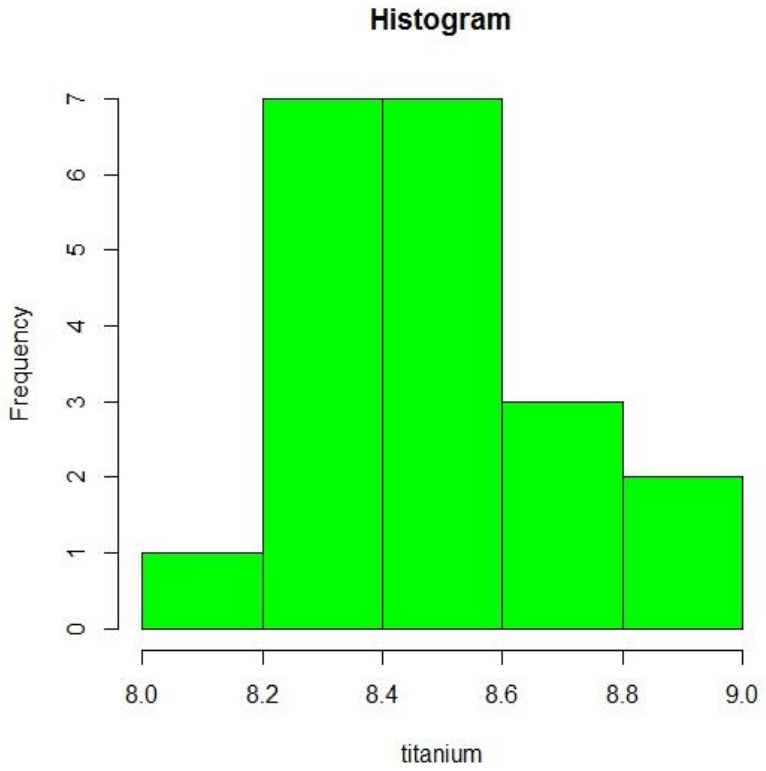


Figure 1.4: Histogram

1.6.5 Stem-and-leaf plot

Example 1.8

Following are the number of room occupied on hotel for 30 days. Draw a stem and leaf diagram.

20,14,21,29,43,17,15,26,8,14,30,23,16,46,28,11,26,35,26,28,30,22,23,7,32,19,22,18,27,9.

Solution:

```
>x=c(20,14,21,29,43,17,15,26,8,14,30,23,16,46,28,11,26,35,26,28,30,22,23,7,32,19,22,18,27,9)
>x
>stem(x)
```

The decimal point is 1 digit(s) to the right of the |

```
 0 | 789
 1 | 144
 1 | 56789
 2 | 012233
 2 | 6667889
 3 | 002
 3 | 5
 4 | 3
 4 | 6
```

1.7 Statistical tests in R

1.7.1 One sample t – test

One sample *t*-test is a statistical procedure used to examine the mean difference between the sample and the known value of the population mean. In one sample t-test, we know the population mean. We draw a random sample from the population and then compare the sample mean with the population mean and make a statistical decision as to whether or not the sample mean is different from the population mean.

We can use this analysis, for example, when we take a sample from the city and we know the mean of the country (population mean). If we want to know whether the city mean differs from the country mean, we will use the one sample *t*-test.

Assumptions:

1. The dependent variable should be measured at the interval or ratio level (i.e., continuous).
2. The data are independent (i.e. not correlated/related).
3. The dependent variable should be normally distributed.

Example 1.9

The following data refer of amount of coffee (in ounces) filled by machine in six randomly picked jars**:** 15.7, 15.9, 16.3, 16.2, 15.7 and 15.9. Is the true means amount of coffee in a jar 16 ounces?

Solution:

Hypothesis,

$$H_0 : \mu = 16 \quad \text{Vs} \quad H_1 : \mu \neq 16$$

To check the normality assumption we use the Shapiro-Wilk normality test. The hypothesis for this test is,

H_0 : Data follows normal distribution

Vs H_1: Data does not follows normal distribution.

The Test statistics,

$$t = \frac{\bar{x} - \mu}{\frac{s}{\sqrt{n}}} \qquad \text{Where } s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}$$

with (n-1) degree of freedom where n is sample size.

R-commands:

```
>x <- c(15.7,15.9,16.3,16.2,15.7,15.9)
>x
>shapiro.test(x)
```

Shapiro-Wilk normality test
data: x
W = 0.8788, p-value = 0.2636

```
> t.test(x,mu=16)
```

One Sample t-test
data: x
t = -0.48795, df = 5, p-value = 0.6462
alternative hypothesis: true mean is not equal to 16
95 percent confidence interval:
15.68659 16.21341
sample estimates:
mean of x 15.95

Interpretation of output:

The p-value of Shapiro-wilk normality test is 0.264. Which is greater than 0.05, hence we accept null hypothesis at 5% level of significance. i.e our data follows normal distribution. Thus, the assumption of normality is satisfied.

The p-value of one sample t-test is 0.6462 which is greater than 0.0.5. Hence we accept null hypothesis. Thus, we can conclude that true mean is equal to 16. i.e. true means amount of coffee in a jar 16 ounces.

If the assumption of Normality of parent population is not satisfied in one sample t-test then non-parametric alternative to one sample t-test suggested by Wilcoxon known as One sample Wilcoxon signed rank test.

1.7.2 One sample Wilcoxon signed rank test (One sample Median test)

The One-Sample Wilcoxon Signed Rank test is a non parametric alternative to a one-sample t-test. The test determines whether the median of the sample is equal to some specified value.

Example 1.10:

Titanium content in an aircraft grade is an of 20 test coupons reveals the following titanium content (in %).
8.32,8.05,8.93,8.65,8.25,8.46,8.52,8.35,8.36,8.41,8.42,8.30,8.71, 8.75, 8.60, 8.83, 8.50, 8.38, 8.29, 8.46. Apply the Wilcoxon signed rank test to test the hypothesis that median titanium content is 8.5%.

Solution:

Hypothesis,

$$H_0 : \text{M} = 8.5 \quad \text{Vs} \quad H_1 : \text{M} \neq 8.5$$

The Test statistics,

$$f(t) = \left(\frac{n_1+n_2}{t}\right) p^t \ (1-q)^{n_1+n_2-1} \quad \text{for t=0,1,2,……}(n_1 + n_2)$$

$$f(u|t) = \frac{\left(\frac{n_1}{u}\right)\left(\frac{n_2}{v}\right)}{\frac{n_1+n_2}{t}} \quad \text{for u= 0,1,2,……,n}$$

R-commands:

```
> x=c(8.32,8.05,8.93,8.65,8.25,8.46,8.52,8.35,8.36,8.41,8.42,8.30,8.71, 8.75, 8.60, 8.83, 8.50, 8.38, 8.29, 8.46)
> x
> wilcox.test(x,mu=8.5)
```

Wilcoxon signed rank test with continuity correction
data: x
V = 80.5, p-value = 0.573
alternative hypothesis: true location is not equal to 8.5

Interpretation of output:

The p-value of wilcoxon signed rank test is 0.572 which is greater than 0.05. Hence we acceptH_0 . Thus we can conclude that true Median is equal to 8.5.

1.7.3 Independent two sample t-test

The independent two-sample t-test is used to test whether two population means are significantly different from each other, using the means from randomly drawn samples.

Assumptions:

1. The dependent variable should be measured at the interval or ratio level (i.e., continuous).
2. The independent variable should consist of two categorical, independent groups.
3. The data are independent (i.e. not correlated/related).
4. The dependent variable should be normally distributed for each group of the independent variable.

Example 1.11:

By Using Independent two sample t-test, check whether the mean of two populations equal or not.

X=28, 31, 26, 27, 23, 38, 37

Y=37, 42, 34, 37, 35

Solution:

Hypothesis

$$H_0 : \mu_1 = \mu_2 \text{ Vs } H_1 : \mu_1 \neq \mu_2$$

Hypothesis for normality:

H_0 : Data follows normal distributions Vs

H_1: Data does not follow distributions

The Test statistics,

$$t = \frac{\bar{X} - \bar{Y}}{s\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}} \quad \text{If } s_1^2 = \frac{\sum_{i=1}^{n1}(X_i-\bar{X})^2}{(n_1-1)} \text{ and } s_2^2 = \frac{\sum_{i=1}^{n2}(Y_i-\bar{Y})^2}{(n_2-1)} \text{ then } s^2 = \frac{(n_1-1)s_1^2+(n_2-1)s_2^2}{(n_1+n_2-2)}$$

with $(n_1 + n_2 - 2)$ degree of freedom.

R- command:

```
>x=c(28,31,26,27,23,38,37)
>x
>y=c(37,42,34,37,35)
>y
>shapiro.test(x)
```

Shapiro-Wilk normality test
data: x
W = 0.91361, p-value = 0.4214

```
>shapiro.test(y)
```

Shapiro-Wilk normality test
data: y

W = 0.88521, p-value = 0.3336

```
>var.test(x,y)
F test to compare two variances
data:  x and y
F = 3.3684, num df = 6, denom df = 4, p-value = 0.2599
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.3662398 20.9757008
sample estimates:
ratio of variances
        3.368421

>t.test(x,y,var.equal=T)
  Two Sample t-test
data:  x and y
t = -2.4927, df = 10, p-value = 0.03184
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -13.2569512  -0.7430488
sample estimates:
mean of x mean of y
      30       37
```

Interpretation of output:

The P-value of Shapiro-wilk normality test is greater than 0.05 for both populations. Hence we acceptH_0 . i.e. Data follows normal distribution for both populations.

The p-value of two sample t-test is less than (p-value=0.03184) 0.05. Hence reject null hypothesis i.e. mean of first population is not equal to mean of second population.

In two sample t-test, we have made assumptions that parent population is normal. If this assumption is not satisfied then we have to do non-parametric test which is known as Wilcoxon Mann-Whitney U-test.

1.7.4 Wilcoxon Mann-whitney U test

Wilcoxon Mann-Whitney U test is the alternative test to the independent two sample t-test. It is a non-parametric test that is used to compare two population means that come from the same population, it is also used to test whether two population means are equal or not. It is used for equal sample sizes, and is used to test the median of two populations. Usually the Mann-Whitney U test is used when the data is ordinal.

Example 1.12:
In a study of factors thought to be responsible for adverse effects of smoking on human reproduction, cadmium level determination(nanograms per gram)were made on placenta tissue of a sample of 14 mothers who were smokers and an independent sample of 18 non-smoking mothers results were as follows.
Non –smokers:
10.0,8.4,12.8,25.0,11.8,9.8,12.5,15.4,23.5,9.4,25.1,19.5,25.5,9.8,7.5,11.8,12.2, 15
Smokers:-30.0,30.1,15.0,24.1,30.5,17.8,16.8,14.8,13.4,28.5,17.5,14.4,12.5,20.4
Does mean cadmium level is equal to smoker and non-smoker.

Solution:
Hypothesis,

$$H_0 : M_1 = M_2 \quad \text{VS} \quad H_1 : M_1 \neq M_2$$

Hypothesis for normality,
H_0 : Data follows normal distributions Vs
H_1: Data does not follow normal distributions.

Test statistics:

$$\text{U} = n_1\, n_2 + \frac{n_2(n_2+1)}{2} - \sum_{i=n_1}^{n_2} R_i$$

R- command:

```
>x=c(10.0,8.4,12.8,25.0,11.8,9.8,12.5,15.4,23.5,9.4,25.1,19.5,25.5,9.8,7.5,11.8,  12.2,15.0)
>x
>y=c(30.0,30.1,15.0,24.1,30.5,17.8,16.8,14.8,13.4,28.5,17.5,14.4,12.5,20.4)
>y
>shapiro.test(x)

      Shapiro-Wilk normality test
data:  x
W = 0.84484, p-value = 0.00703

>shapiro.test(y)
Shapiro-Wilk normality test
data:  y
W = 0.85335, p-value = 0.02467

>wilcox.test(x,y)
Wilcoxon rank sum test with continuity correction
data:  x and y
W = 58, p-value = 0.01032
```

alternative hypothesis: true location shift is not equal to 0

Interpretation of output:

For both variable x and y the p-value of Shapiro-Wilk normality test is less than 0.05. Hence we reject the null hypothesis of normality. Thus, we conclude that both variables are does not follows normal distribution. Hence we do non parametric test which is known as Wilcoxon Mann-whitney U-test.

The P-value of Mann-Whitney U-test is 0.01032. This is less than 0.05. Hence we reject the null hypothesis. Thus we conclude that the median of first population is not equal to median of second population.

1.7.5 Paired t-test

The Paired Samples *t* Test compares two means that are from the same individual, object, or related units. The two means typically represent two different times (e.g., pre-test and post-test with an intervention between the two time points) or two different but related conditions or units (e.g., left and right ears, twins). The purpose of the test is to determine whether there is statistical evidence that the mean difference between paired observations on a particular outcome is significantly different from zero.

Assumptions:

1. The dependent variable should be measured at the interval or ratio level (i.e., continuous).
2. The independent variable should consist of two categorical, "related groups" or "matched pairs".
3. The distribution of the differences in the dependent variable between the two related groups should be normally distributed.

Example 1.13:

An automotive engineer is investigating two different types of metering devices for an electronic fuel injection system to determine whether they differ in fuel mileage performance. The system is installed on 12 different cars and the test is run with each metering devise on each car. Observed fuel performance data corresponding to different devices is shown in following table. Use appropriate test to check the hypothesis that two devices do not differ in their fuel mileage performance.

Device	Mileage											
I	17.6	19.4	19.5	17.1	15.3	15.9	16.3	18.4	17.3	19.1	17.8	18.2
II	16.8	20	18.2	16.4	16	15.4	16.5	18	16.4	20.1	16.7	17.9

Solution:

$$H_0 : \bar{d} = 0 \quad \text{Vs} \quad H_1 : \bar{d} \neq 0$$

Hypothesis for normality:
H_0 : Difference (x-y) follows normal distributions. Vs
H_1: Difference (x-y) does not follow normal distribution.

Test statistics:

$$t = \frac{\bar{d}}{\frac{s}{\sqrt{n}}} \qquad \text{Where } \bar{d} = \frac{\sum_{i=1}^{n} d_i}{n} \quad \text{and} \quad s^2 = \frac{\sum_{i=1}^{n}(d_i - \bar{d})^2}{(n-1)} \text{ with (n-1)}$$

degree of freedom.

R-command:

```
>x=c(17.6,19.4,19.5,17.1,15.3,15.9,16.3,18.4,17.3,19.1,17.8,18.2)
>x
>y=c(16.8,20,18.2,16.4,16,15.4,16.5,18,16.4,20.1,16.7,17.9)
>y
>difference =(x-y)
>shapiro.test(difference)
```

Shapiro-Wilk normality test
data: difference
W = 0.93417, p-value = 0.4265

```
>t.test(x,y,paired=T)
```

Paired t-test
data: x and y
t = 1.3448, df = 11, p-value = 0.2058
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-0.1856942 0.7690275
sample estimates:
mean of the differences 0.2916667

Interpretation of output:

P-value of Shapiro-Wilk normality test is 0.4265. This is greater than 0.05. Hence we accept null hypothesis. Thus we conclude that difference follows normal distribution.

The p-value of paired t-test is 0.2058. This is greater than 0.05. Hence we accept null hypothesis. Thus, we conclude that the two types of measuring devices do not differ in their fuel mileage performance.

If the assumption of differences in the dependent variable between the two related groups should be normally distributed is not satisfied in paired t-test then

non-parametric alternative to paired t-test suggested by Wilcoxon known as Wilcoxon signed rank test (Two related test).

1.7.6 Wilcoxon signed rank test (Two related sample)

The Wilcoxon signed-rank test is the nonparametric test equivalent to the paired t-test. As the Wilcoxon signed-rank test does not assume normality in the data, it can be used when this assumption has been violated and the use of the paired t-test is inappropriate.

It is used to compare two sets of scores that come from the same participants. This can occur when we wish to investigate any change in scores from one time point to another, or when individuals are subjected to more than one condition.

Example 1.14:

Seventeen families participated in a training program in which a test was administered before and after training to one parent in each family. Following are pre and post-training scores made by the parent on test.

Pre	7	6	10	16	8	13	8	14	16	11	12	13	9	10	17	8	5
Post	11	14	16	17	9	15	9	17	20	12	14	15	14	15	18	15	9

May we conclude, on the basis of these data, that training program is effective or not?

Solution:

Hypotheses,

$$H_0: \ \bar{d} = 0 \text{ Vs } H_1: \ \bar{d} \neq 0$$

Test statistics is,

$$f(t) = \left(\frac{n_1+n_2}{t}\right) p^t \ (1-q)^{n_1+n_2-1} \quad \text{for t=0,1,2,.....}(n_1+n_2)$$

$$f(u|t) = \frac{\left(\frac{n_1}{u}\right)\left(\frac{n_2}{v}\right)}{\frac{n_1+n_2}{t}} \quad \text{for u= 0,1,2,.....,n}$$

R-command:

```
>x=c(7,6,10,16,8,13,8,14,16,11,12,13,9,10,17,8,5)
>x
>y=c(11,14,16,17,9,15,9,17,20,12,14,15,14,15,18,15,9)
>y
>wilcox.test(x,y,paired=T)
```

Wilcoxon signed rank test with continuity correction
data: x and y

V = 0, p-value = 0.0003034
alternative hypothesis: true location shift is not equal to 0

Warning message:
In wilcox.test.default(x, y, paired = T) :
cannot compute exact p-value with ties

Interpretation of output:

The p-value of Wilcoxon signed rank test is 0.0003034. This is less than 0.05. Hence we reject null hypothesis. Thus we conclude that the training program is effective.

1.7.7One-way analysis of variance (ANOVA)

The one-way analysis of variance (ANOVA) is used to determine whether there are any statistically significant differences between the means of two or more independent (unrelated) groups.

For example, you could use a one-way ANOVA to understand whether exam performance differed based on test anxiety levels amongst students, dividing students into three independent groups (e.g., low, medium and high-stressed students).

It is important to realize that the one-way ANOVA cannot tell you which specific groups were statistically significantly different from each other; it only tells you that at least one pair are different. Since you may have two, three, four, five or more pairs in your study design, determining which of these pairs differ is important. To find this pairs by using a post hoc test.

Assumptions:

1. The dependent variable should be measured at the interval or ratio level (i.e., continuous).
2. Independent variable should consist of two or more categorical, independent groups.
3. The data are independent (i.e. not correlated/related).
4. The dependent variable should be approximately normally distributed for each category of the independent variable.

Example 1.15:

Following table shows forced expiratory valume per second for patients with coronary artery disease sample at three different medical centers, denoted by A, B and C.

A	3.23	3.47	1.86	2.47	3.01	1.69	2.10	2.81	3.28	3.36	2.61	2.91
	1.98	2.57	2.08	2.47	2.47	2.74	2.88	2.63	2.53			
B	3.52	3.23	2.21	3.19	4.12	3.79	3.79	4.13	3.14	3.21	3.21	3.91

	3.37	3.11	3.89	3.67								
C	2.79	3.22	2.25	2.98	2.47	2.77	2.95	3.56	2.88	2.63	3.38	3.07
	2.81	3.17	2.23	2.19	4.06	1.98	2.81	2.85	2.43	3.2	3.53	

Test whether the average expiratory volume per second for all patients with coronary artery disease from all three centers are equal.

Solution:

Hypothesis,

$H_0: \mu_1 = \mu_2 = \mu_3$ Vs $H_1:$ at least one pair differs.

Hypothesis for normality:

H_0 : Data follows normal distributions. Vs

H_1: Data does not follow distributions.

Hypothesis for variance:

H_0 : Variances are equal Vs

H_1: Variances are not equal.

Test statistics is,

R-commands:

```
>a=c(3.23,3.47,1.86,2.47,3.01,1.69,2.10,2.81,3.28,3.36,2.61,2.91,1.98,2.57,2.08,2.47,2.47,2.74,2
.88,2.63,2.53)
>a
>b=c(3.52,3.23,2.21,3.19,4.12,3.79,3.79,4.13,3.14,3.21,3.21,3.91,3.37,3.11,3.89,3.67)
>b
>c=c(2.79,3.22,2.25,2.98,2.47,2.77,2.95,3.56,2.88,2.63,3.38,3.07,2.81,3.17,2.23,2.19,4.06,1.98,2
.81,2.85,2.43,3.20,3.53)
>c
>shapiro.test(a)
    Shapiro-Wilk normality test
data:  a
W = 0.97084, p-value = 0.7515

>shapiro.test(b)
     Shapiro-Wilk normality test
data:  b
W = 0.90868, p-value = 0.1108

>shapiro.test(c)
   Shapiro-Wilk normality test
```

data: c
W = 0.97996, p-value = 0.9056

>d=stack(list("A"=a,"B"=b,"C"=c))
>names(d)
[1] "values" "ind"

>attach(d)
>bartlett.test(values~ind)
Bartlett test of homogeneity of variances
data: values by ind
Bartlett's K-squared = 0.0062359, df = 2, p-value = 0.9969

>oneway.test(values~ind,data=d,var.equal=T)
One-way analysis of means
data: values and ind
F = 13.506, num df = 2, denom df = 57, p-value = 1.58e-05

>pairwise.t.test(values,ind,p.adj="bonferroni")
Pairwise comparisons using t tests with pooled SD
data: values and ind

	A	*B*
B	*1.1e-05*	-
C	*0.2891*	*0.0017*

P value adjustment method: bonferroni

Interpretation of output:

The p-value of Shapiro-Wilk normality test for all three populations are greater than 0.05. Hence we accept the null hypothesis. Thus, all three populations follow normal distribution.

The P-value of Bartlett test of homogeneity of variance is 0.9969. This is greater than 0.05. Hence we accept H_0. i.e. variances are equal.

The p-value of one way analysis is less than 0.05. Thus, we can conclude that H_0 is rejected. i.e. at least one pair of mean is differ significantly.

The bonferroni test suggested that there is means of pair A and B, B and C are differ significantly but means of pair A and C do not differ significantly.

In the ANOVA, we assume that distribution of each group is normally distributed. If this assumption is not satisfied in ANOVA then non-parametric alternative to ANOVA developed by Kruskal and Wallis (1952) as Kruskal-Wallis test.

1.7.8 Kruskal-Wallis test

The Kruskal-Wallis H test (sometimes also called the "one-way ANOVA on ranks") is a rank-based nonparametric test that can be used to determine if there are statistically significant differences between two or more groups of an independent variable on a continuous or ordinal dependent variable.

Example 1.16:

An instructor sets three different question papers and distributes them randomly to her students. After collecting answer books and grading them, following scores are obtained.

Test 1:63, 64, 95, 64, 60, 85.
Test 2:58, 56, 51, 84, 77.
Test 3:105, 79, 82, 80, 74, 97.

The instructor would like to know whether the three tests are equally difficult by testing equality of means of populations of scores. Carry out the appropriate test procedure.

Solution:

Hypothesis

$H_0: M_1 = M_2 = M_3$ Vs H_1: at least one pair differ.

Test statistics:

$$w = \frac{12}{N(N+1)} \sum_{i=1}^{k} \frac{R_i^2}{n_i} - 3(N+1)$$

R-Command:

```
>x=c(63,64,95,64,60,85)
>x
>y=c(58,56,51,84,77)
>y
>z=c(105,79,82,80,74,97)
>z
>d=stack(list("test1"=x,"test2"=y,"test3"=z))
>names(d)=c("values","test")
>names(d)
[1] "values" "test"

>kruskal.test(values~test,data=d)
   Kruskal-Wallis rank sum test
data:  values by test
Kruskal-Wallis chi-squared = 4.9459, df = 2, p-value = 0.08433
```

Interpretation of Output:
The p-value of kruskal-wallis rank test is 0.08433. This is greater than 0.05. Henc null hypothesis is accepted. i.e. Median of all three population are equal.

1.7.9 Correlation

(a) Pearson product-moment correlation coefficient

The Pearson product-moment correlation coefficient (Pearson's correlation) is a measure of the strength and direction of association that exists between two variables measured on at least an interval scale.

For example, you could use a Pearson's correlation to understand whether there is an association between exam performance and time spent revising. You could also use a Pearson's correlation to understand whether there is an association between depression and length of unemployment.

A Pearson's correlation attempts to draw a line of best fit through the data of two variables, and the Pearson correlation coefficient, *r*, indicates how far away all these data points are from this line of best fit.

Assumptions:

1. Two variables should be measured at the interval or ratio level (i.e. Continuous).
2. There is a linear relationship between your two variables.
3. The Two variables should be normally distributed.

Example 1.17:

Plot the scatter diagram and compute the Pearson's correlation co-efficient between amount(X) of fertilizer and the yield(Y) of potatoes for the data

X	0	4	8	12
Y	8.34	8.89	9.16	9.5

Solution:

Hypothesis,

$$H_0 : \rho = 0 \quad \text{Vs} \quad H_1 : \rho \neq 0$$

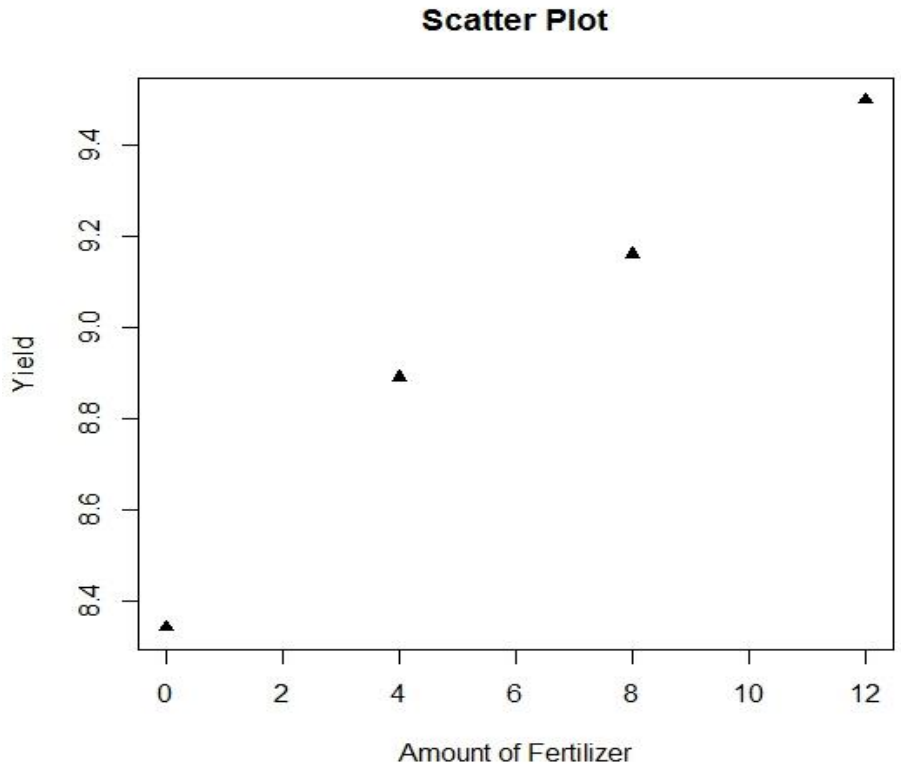
Test statistics:

$$r = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2 \sum(y-\bar{y})^2}}$$

R-command:

```
>x=c(0,4,8,12)
>x
>y=c(8.34,8.89,9.16,9.5)
```

```
>y
>plot(x,y,pch=17,xlab="Amount of Fertilizer",ylab="Yield",main="Scatter Plot")
```




```
>r=cor.test(x,y,method="pearson", alt="two.sided",conf.int=T)
>r
```

Pearson's product-moment correlation

data: x and y
t = 9.0552, df = 2, p-value = 0.01198
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
0.5341930 0.9997609
sample estimates:
cor
0.9880231

Interpretation of Output:

The Pearson product-moment correlation coefficient is 0.9880231. Hence we conclude that there is strong positive correlation between amount of fertilizer and the yield of potatoes.

The p-value is less than 0.05. Hence we rejected H_0. i.e. population correlation coefficient is not equal to zero.

(b) Spearmen's rank correlation

The Spearman rank-order correlation coefficient (Spearman's correlation) is a nonparametric measure of the strength and direction of association that exists between two variables measured on at least an ordinal scale.

Example 1.18:

Compute the Spearmen's rank co-efficient between marks in Mathematics (X) of and the marks in Statistics (Y) of 12th student for the data.

X	76	84	88	92	70
Y	83	88	91	95	77

Solution:

Hypothesis,

$$H_0 : \rho = 0 \qquad \text{Vs} \qquad H_1 : \rho \neq 0$$

Test statistics is,

$$r = 1 - \frac{6\sum d^2}{n(n^2-1)}$$

R-command:

```
>x=c(76,84,88,92,70)
>x
>y=c(83,88,91,95,77)
>y
>r=cor.test(x,y,method="spearman", alt="two.sided",conf.int=T)
>r
```

Spearman's rank correlation rho
data: x and y
S = 4.4409e-15, p-value = 0.01667
alternative hypothesis: true rho is not equal to 0
sample estimates:
rho 1

Interpretation of Output:

The Spearman rank-order correlation coefficient is 1. Hence we conclude that there is perfect positive correlation between marks in Mathematics and marks in Statistics.

The p-value is less than 0.05. Hence we reject H_0. i.e. population correlation coefficient is not equal to zero.

1.7.10 Linear Regression

Linear regression is the next step up after correlation. It is used when we want to predict the value of a variable based on the value of another variable. The variable we want to predict is called the dependent variable. The variable we are using to predict the other variable's value is called the independent variable.

For example, you could use linear regression to understand whether exam performance can be predicted based on revision time.

Example 1.19:

Following table gives the observed of gains corresponding to amount of fertilizer applied.

Amount of fertilizer(y)	Yield(x)
30	43
40	45
50	54
60	53
70	56
80	63

Fit the lines of regression of amount of fertilizer on yield.

Solution:

Hypothesis,

$$H_0 : \beta_{yx} = 0 \quad \text{Vs} \quad H_1 : \beta_{yx} \neq 0$$

Regression line of y on x is

$$y = a + b_{yx} * x$$

Where b_{yx}= Cov(x, y) / σ_x^2 and a= $\bar{y}$ - $b_{yx}\bar{x}$

R-command:

```
>x=c(43,45,54,53,56,63)
>x
>y=c(30,40,50,60,70,80)
>y
>fit=lm(y~x)
>fit
Call:
lm(formula = y ~ x)
Coefficients:
(Intercept)          x
   -72.297       2.432

>summary(fit)
Call:
```

```
lm(formula = y ~ x)

Residuals:
     1       2       3       4       5       6
-2.2973  2.8378 -9.0541  3.3784  6.0811 -0.9459

Coefficients:
            Estimate Std. Error  t value Pr(>|t|)
(Intercept) -72.297     19.259   -3.754  0.01988 *
x             2.432      0.365    6.664  0.00263 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.012 on 4 degrees of freedom
Multiple R-squared:  0.9174,    Adjusted R-squared:  0.8967
F-statistic: 44.41 on 1 and 4 DF,  p-value: 0.002634
```

Interpretation of Output:

The P-value of F-statistics is 0.002634. This is less than 0.05. Hence H_0 is rejected.

i.e. $\beta_{yx} \neq 0$.

The regression line is, y = -72.297 + 2.432 * x

1.7.11 Chi-square test

(a) Test for Association

The chi-square test for independence, also called Pearson's chi-square test or the chi-square test of association, is used to discover if there is a relationship between two categorical variables.

Assumptions:

1. The two variables should be measured at an ordinal or nominal level (i.e., categorical data).
2. The two variables should consist of two or more categorical**,** independent groups**.**

Example 1.20:

An educator would like to know whether gender (male/female) is associated with the preferred type of learning medium (online vs. books).

		Preferred learning Medium		
		Books	Online	Total
Gender	Male	16	24	40
	Female	13	27	40
	Total	29	51	80

Test the hypothesis that Gender is associated with preferred type of learning medium.

Solution:
Hypothesis,
H_0 : Gender is associated with preferred type of learning medium. Vs
H_1: Gender is not associated with preferred type of learning medium.

Test statistics is,

$$X^2 = \frac{\sum(o_i - e_i)^2}{e_i} \text{ with (k-1)(k-1) df.}$$

R-commands:

```
>o1=16
>o2=24
>o3=13
>o4=27
>e1=29*40/80
>e1
>e2=51*40/80
>e2
>e3=29*40/80
>e3
>e4=51*40/80
>e4
>chi=(o1-e1)^2/e1+(o2-e2)^2/e2+(o3-e3)^2/e3+(o4-e4)^2/e4
>chi
[1] 0.4868154

>qchisq(0.95,df=1)
[1] 3.841459
```

Interpretation of output:
The output shows that calculated value of Chi-square is 0.4868154 and tabulated value of Chi-square is 3.841459. Here $X^2_{(calculated)}$ is less than $X^2_{(tabulated)}$. Hence we accept null hypothesis and conclude that gender is associated with preferred type of learning medium.

(b) Goodness of fit test

The chi-square goodness-of-fit test is a single-sample nonparametric test, also referred to as the one-sample goodness-of-fit test or Pearson's chi-square goodness-of-fit test. It is used to determine whether the distribution of cases in a

single categorical variable follows a known or hypothesized distribution. (e.g. Binomial, Poisson, Normal etc.).

Example 1.21: (Fitting of Binomial distribution)

Fit the binomial distribution and test the goodness of fit on following data.

X	0	1	2	3	4
F	5	20	45	20	10

Solution:

H_0 : Fit of Binomial distribution is good. Vs

H_1: Fit of Binomial distribution is not good

Test statistics is,

$$X^2 = \sum_{i=1}^{k} \frac{(o_i - E_i)^2}{E_i} \quad \text{with (n-1) df.}$$

R-commands:

```
>x=0:4
>x
>f=c(5,20,45,20,10)
>f
>n=max(x)
>n
>N=sum(f)
>N
>smean=sum(f*x/sum(f))
>smean
>p=smean/n
>p
>px=dbinom(0:3,n,p)
>px
>p4=1-sum(px)
>p4
>px=c(px,p4)
>px
>px=round(px,4)
>px
>ex=px*N
>ex
>fr.dist=data.frame(x,f,px,ex)
>fr.dist
>chisq=sum((f-ex)^2/ex)
>chisq
[1] 4.665105
```

```
>qchisq(0.95,3)
[1] 7.814728
```

Interpretation of output:
The output shows that, calculated value of Chi-square is 4.665105 and tabulated value of Chi-square is 7.814728. Here $X^2_{(calculated)}$ is less than $X^2_{(tabulated)}$. Hence H_0 is accepted and we conclude that Fit of Binomial distribution is good

Example 1.22: (Fitting of Poisson distribution)
Following table shows the data on the movement of leaf hopper (Hemiptera) across a sand dune.

Leaf hopper per trap(x)	frequency (f)
0	6
1	8
2	12
3	4
4 or more	3

Fit Poisson distribution to the above data and test goodness of fit.

Solution:
Hypothesis,
H_0 : Fit of Poisson distribution is good. Vs
H_1: Fit of Poisson distribution is not good

Test statistics is,

$X^2 = \sum_{i=1}^{k} \frac{(o_i - E_i)^2}{E_i}$ with (n-2) df.

R-Commands:

```
>x=0:4
>x
>f=c(6,8,12,4,3)
>smean=sum(f*x)/sum(f)
>smean
>x=0:3
>x
>px=dpois(x,smean)
>px
>px=c(px,1-sum(px))
>px
>ex=sum(f)*px
```

```
>ex
>x=c(x,4)
>x
>fx.dist=data.frame(x,f,ex)
>fx.dist
>ex=c(ex[c(1:3)],sum(ex[c(4:5)]))
>ex
>o=c(6,8,12,7)
>o
>chiq=sum((o-ex)^2/ex)
>chiq
[1] 1.866058

>qchisq(0.95,2)
[1] 5.991465
```

Interpretation of output:
The output shows that, calculated value of Chi-square is 1.866058 and tabulated value of Chi-square is 5.991465. Here $X^2_{(calculated)}$ is less than $X^2_{(tabulated)}$. There for we accept H_0 and conclude that fit of Poisson distribution is good.

Example 1.23: (Fitting of Normal distribution)
Following table displays a frequency distribution of heights of trees in a certain Locality. Fit a normal distribution to the data and test the goodness of fit.

Heights of tress (in inches)

Class-internal	frequency
13.20-20.90	2
20.90-28.60	10
28.60-36.30	16
36.30-44.00	37
44.00-51.70	43
51.70-59.40	39
59.40-67.10	29
67.10-74.80	13
74.80-82.50	6
82.50-90.20	5

Solution:
Hypothesis,
H_0 : Fit of normal distribution is good. Vs
H_1: Fit of normal distribution is not good

Test statistics is,

$$X^2 = \sum_{i=1}^{k} \frac{(o_i - E_i)^2}{E_i} \text{ with (n-3) df.}$$

R-Commands:

```
>midy=seq(17.05,86.35,length=10)
>midy
>f=c(2,10,16,37,43,39,29,13,6,5)
>f
>mean=sum(midy*f)/sum(f)
>mean
>sd=(sum(f*(midy-mean)^2)/sum(f))^0.5
>sd
>l=seq(13.2,82.5,length=10)
>l
>l=c(l,90.2)
>l
>cdf=pnorm(l,mean,sd)
>cdf
>cdf=c(0,cdf,1)
>cdf
>pcf=diff(cdf)
>pcf
>f=c(0,f,0)
>f
>ex=round(pcf*sum(f),4)
>ex
>fr=data.frame(f,ex)
>fr
>o=c(12,16,37,43,39,29,13,11)
>o
>ex=c(sum(ex[c(1,2,3)]),ex[c(4:9)],sum(ex[c(10,11,12)]))
>ex
>chisq=sum((o-ex)^2/ex)
>chisq
[1] 2.132088

>qchisq(0.95,5)
[1] 11.0705
```

Interpretation of output:

The output shows that, calculated value of Chi-square is 2.132088 and tabulated value of Chi-square is 11.0705. Here $X^2_{(calculated)}$ is less than $X^2_{(tabulated)}$. Therefore we accept H_0 and conclude that fit of normal distribution is good.

Unit-II

2.1 Concept of Random number Generator

One of the fundamental tools required in computational statistics is the ability to simulate random variables from specified probability distributions. On the general subject of methods for generating random variates from specified probability distributions. In the simplest case, to simulate drawing an observation at random from a finite population, a method of generating random observations from the discrete uniform distribution is required. Therefore a suitable generator of uniform pseudo random numbers is essential.

Methods for generating random variates from other probability distributions all depend on the uniform random number generator. In this text we assume that a suitable uniform pseudo random number generator is available.

The uniform pseudo random number generator in R is runif. To generate a vector of *n* (pseudo) random numbers between 0 and 1 use runif(n). Throughout this text, whenever computer generated random numbers are mentioned, it is understood that these are pseudo random numbers. To generate *n* random Uniform (*a*, *b*) numbers use runif(n, a, b). To generate an *n* by *m* matrix of random numbers between 0 and 1 use matrix(runif (n*m), nrow=n, ncol=m) or matrix(runif(n*m), n, m).

The several functions are given for generating random variates from continuous and discrete probability distributions. Generators for many of these distributions are available in R (e.g. rbeta, rgeom, rchisq, etc.), but the methods presented below are general and apply to many other types of distributions.

2.2 Random Generators of Common Probability Distributions in R

In the sections that follow, various methods of generating random variates from specified probability distributions are presented. Before discussing those methods, however, it is useful to summarize some of the probability functions available in R. The probability mass function (pmf) or density (pdf), cumulative distribution function (cdf), quantile function, and random generator of many commonly used probability distributions are available. For example, four functions are documented in the help topic Binomial:

```
dbinom(x, size, prob, log = FALSE)
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
rbinom(n, size, prob)
```

Arguments

x, q Vector of quantiles.

p Vector of probabilities.

n Number of observations. If length(n) > 1, the length is taken to be the number required.

size Number of trials (zero or more).

prob Probability of success on each trial.

log, log.p Logical; if TRUE, probabilities p are given as log(p).

lower.tail Logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

The same pattern is applied to other probability distributions. In each case, the abbreviation for the name of the distribution is combined with first letter **d** for density or pmf, **p** for cdf, **q** for quantile, or **r** for random generation from the distribution.

A partial list of available probability distributions and parameters is given in Table 2.1

TABLE 2.1: Selected Univariate Probability Functions Available in R

Distribution	cdf	Generator	Parameters
beta	pbeta	rbeta	shape1, shape2
binomial	pbinom	rbinom	size, prob
chi-squared	pchisq	rchisq	df
exponential	pexp	rexp	rate
F	pf	rf	df1, df2
Gamma	pgamma	rgamma	shape, rate or scale
Geometric	pgeom	rgeom	prob
lognormal	plnorm	rlnorm	meanlog, sdlog
negative binomial	pnbinom	rnbinom	size, prob
normal	pnorm	rnorm	mean, sd
Poisson	ppois	rpois	lambda
Student's t	pt	rt	df
uniform	punif	runif	min, max

2.3 Methods for generation of pseudo random numbers

We begin our discussion of simulation with a brief exploration of the mechanics of pseudo random number generation. In particular, we will describe one of the simplest methods for simulating independent uniform random variables on the interval [0,1].

2.3.1 Multiplicative Congruential Method of generating uniform variate

A multiplicative congruential random number generator produces a sequence of pseudo random numbers, u_1, u_2, . . . which appear similar to independent uniform random variables on the interval [0,1].

Let m be a large integer, and let b be another integer which is smaller than m. The value of b is often chosen to be near the square root of m. Different values of b and m give rise to pseudorandom number generators of varying quality. There are various criteria available for choosing good values of these parameters, but it is always important to test the resulting generator to ensure that it is providing reasonable results.

To begin, an integer x_0 is chosen between 1 and m. x_0 is called the seed. We discuss strategies for choosing x_0 below.

Once the seed has been chosen, the generator proceeds as follows:

$$x_1 = b\, x_0 \ (\text{mod } m)$$
$$u_1 = x_1 / m.$$

Where u_1 is the first pseudo random number, taking some value between 0 and 1.

The second pseudo random number is then obtained in the same manner:

$$x_2 = b\, x_1 \ (\text{mod } m)$$
$$u_2 = x_2 / m.$$

Here u_2 is another pseudo random number. If m and b are chosen properly and are not disclosed to the user, it is difficult to predict the value of u_2, given the value of u_1 only. In other words, for most practical purposes u_2 is approximately independent of u_1. The method continues according to the following formulas:

$$x_n = b\, x_{n-1} \ (\text{mod } m)$$
$$u_n = x_n / m.$$

This method produces numbers which are entirely deterministic, but to an observer who doesn't know the formula above, the numbers *a*ppear to be random and unpredictable, at least in the short term.

Example 2.1: (Multiplicative Congruential Method)

The following lines produce 50 pseudorandom numbers based on the multiplicative congruential generator:

$$x_n = 171\, x_{n-1} \ (\text{mod } 30269)$$

$$u_n = x_n / 30269$$

with initial seed $x_0 = 27218$.

Solution:

```
>random.number <- numeric (50) # this will store the pseudorandom output
 >random.seed <- 27218
 >for (j in 1:50) {
+ random.seed <- (171 * random.seed) %% 30269
 + random.number[j] <- random.seed / 30269
 +}
> random.number
  [1] 0.76385080 0.61848756 0.76137302 0.19478675 0.30853348 0.75922561
  [7] 0.82757937 0.51607255 0.24840596 0.47741914 0.63867323 0.21312234
[13] 0.44391952 0.91023820 0.65073177 0.27513297 0.04773861 0.16330239
[19] 0.92470845 0.12514454 0.39971588 0.35141564 0.09207440 0.74472232
[25] 0.34751726 0.42545178 0.75225478 0.63556774 0.68208398 0.63636063
[31] 0.81766824 0.82126929 0.43704780 0.73517460 0.71485678 0.24051009
[37] 0.12722587 0.75562457 0.21180085 0.21794575 0.26872378 0.95176583
[43] 0.75195745 0.58472364 0.98774324 0.90409330 0.59995375 0.59209092
[49] 0.24754700 0.33053619
```

A similar kind of operation (though using a different formula, and with a much longer cycle) is used internally by R to produce pseudorandom numbers automatically with the function runif ().

Syntax

```
runif(n, min = a, max = b)
```

Execution of this command produces *n* pseudorandom uniform numbers on the interval [*a, b*]. The default values are $a = 0$ and $b = 1$. The seed is selected internally.

Example 2.2:

Generate five uniform pseudorandom numbers on the interval [0, 1], and 10 uniform such numbers on the interval [−3,−1].

Solution:

```
>runif(5)
>runif(10, min = -3, max = -1)
 [1] -1.666937 -2.051194 -2.493232 -2.160725 -1.532510 -2.508264 -2.071462
 [8] -2.462480 -2.301991 -1.533259
```

2.3.2 The Inverse Transform Method

The inverse transform method of generating random variables is based on the following well known result,

(a) Inverse Transform Method for Continuous distribution

Theorem 2.1: (Probability Integral Transformation)

If X is a continuous random variable with cdf $F_X(x)$, then $U = F_X(x) \sim$ Uniform(0, 1).

The inverse transform method of generating random variables applies the probability integral transformation. Define the inverse transformation,

$$F_X^{-1}(u) = \inf \{x: F_X(x) = u\}, \qquad 0 < u < 1.$$

If $U \sim$ Uniform(0, 1), then for all $x \in R$

$$\begin{aligned} P(F_X^{-1}(U) \le x) &= P(\inf\{t : F_X(t) = U\} \le x) \\ &= P(U \le F_X(x)) \\ &= F_U(F_X(x)) = F_X(x), \end{aligned}$$

and therefore $F_X^{-1}(U)$ has the same distribution as X. Thus, to generate a random observation X, first generate a Uniform (0,1) variate u and deliver the inverse value $F_X^{-1}(u)$. The method is easy to apply, provided that F_X^{-1} is easy to compute. The method can be applied for generating continuous or discrete random variables.

The method can be summarized as follows:

1. Derive the inverse function $F_X^{-1}(u)$.
2. Write a command or function to compute $F_X^{-1}(u)$.
3. For each random variate required:
   (a) Generate a random u from Uniform (0,1).
   (b) Deliver $x = F_X^{-1}(u)$.

Example 2.3*:* (continuous case)

This example uses the inverse transform method to simulate a random sample from the distribution with density $f_X(x) = 3x^2, \qquad 0 < x < 1.$

Solution:

Here $F_X(x) = x^3$ for $0 < x < 1$, and $F_X^{-1}(u) = u^{1/3}$. Generate all n required random uniform numbers as vector u. Then u^(1/3) is a vector of length n containing the sample $x_1, \ldots, x_n$.

R-commands:

```
>n <- 100
>u <- runif(n)
>x <- u^(1/3)
>x
  [1] 0.8017509 0.8092308 0.7659171 0.6885996 0.9530523 0.9322022 0.9973710
  [8] 0.7564671 0.7752620 0.7826785 0.9585013 0.8017270 0.8821938 0.5613602
 [15] 0.7773277 0.8820697 0.3013756 0.7065704 0.5502851 0.8371029 0.6610354
 [22] 0.8130874 0.5110281 0.8630272 0.4970090 0.8709242 0.6978373 0.8514141
 [29] 0.9395620 0.9280061 0.5633003 0.6425118 0.8178466 0.6948340 0.7486318
 [36] 0.8709448 0.9229255 0.5590832 0.8017308 0.9581135 0.7566680 0.7471477
 [43] 0.6538598 0.7765798 0.9336162 0.5726958 0.9556333 0.9484229 0.7977834
 [50] 0.8671924 0.9998218 0.8620407 0.7930682 0.6803484 0.9994234 0.6145732
 [57] 0.7644595 0.6422658 0.3873663 0.9850638 0.4256389 0.7753329 0.5766481
 [64] 0.9936862 0.9762227 0.8037953 0.9850256 0.8906251 0.8682515 0.6050461
 [71] 0.8532275 0.9574121 0.6661810 0.7092815 0.6293770 0.8857795 0.6183310
 [78] 0.4612910 0.5768998 0.9038709 0.6654110 0.8742584 0.6437220 0.2246501
 [85] 0.8885111 0.5929618 0.9203562 0.5340973 0.9815111 0.9348766 0.1928312
 [92] 0.8935198 0.8835455 0.5937719 0.7211663 0.7465011 0.7337043 0.9215036
 [99] 0.3692013 0.3787825
```

Example 2.4*:* (Exponential distribution)

Apply the inverse transform method to generate a random sample of size 50 from the exponential distribution with mean ($1/\lambda$). *(Take* $\lambda = 2$)

Solution:

If $X \sim \text{Exp}(\lambda)$, then for $x > 0$ the cdf of X is $F_X(x) = 1 - e^{-\lambda x}$. The inverse transformation is $F_X^{-1}(u) = -\frac{1}{\lambda}\log(1 - u)$. Note that U and $1 - U$ have the same distribution and it is simpler to set $x = -\frac{1}{\lambda}\log(u)$.

To generate a random sample of size n with parameter lambda:

-log(runif(n)) / lambda

R-command*:*

```
>n <- 50
> lambda<-2
>u <- runif(n)
>x <- -log(u) / lambda
>x
  [1] 0.3598627242 0.3347805546 0.7596332007 0.2463782606 0.2239335267
  [6] 0.2036329982 1.0090972137 0.9739047124 0.0001422623 0.6846383850
 [11] 0.7637487292 0.1994585613 0.4722244805 0.2802124042 1.1055513937
 [16] 0.1143128321 0.4937254753 0.0903049990 0.0082735617 0.2923288222
[21] 0.5257431151 0.0186001460 0.1579470824 0.2890119377 0.8753127855
[26] 0.2217230566 0.0319919366 0.2026061099 0.1926596397 1.0311402242
```

[31] 0.7202690999 0.2936336867 0.2628641877 0.4913609900 0.5257280754
[36] 0.2731594433 0.1822659334 0.3529688704 0.7823672629 0.5368235895
[41] 0.1940126521 1.0570093869 0.0291803706 0.8973065598 0.0545378281
[46] 2.0402737576 0.2391678679 0.2475638189 0.2304718313 0.4128641585

(b) Inverse Transform Method for Discrete distributions

The inverse transform method can also be applied to discrete distributions. If X is a discrete random variable and

$$...<x_{i-1}<x_i<x_{i+1}<...$$

are the points of discontinuity of $F_X(x)$, then the inverse transformation is $F_X^{-1}(u)=x_i$, where $F_X(x_{i-1}) < u \leq F_X(x_i)$. For each random variate required:

1. Generate a random u from Uniform (0,1).
2. Deliver x_i where $F(x_{i-1}) < u \leq F(x_i)$.

Note: The solution of $F(x_{i-1}) < u \leq F(x_i)$ in Step (2) may be difficult for some distributions.

Example 2.5: (Two point distribution)
Apply the inverse transform to generate a random sample of size 100 of Bernoulli (p =0.4) variates.
Solution:
This example illustrates computing the inverse cdf of a discrete random variable in the simplest case.

In this example, $F_X(0) = f_X(0) = 1-p$ and $F_X(1) = 1$. Thus, $F_X^{-1}(u)=1$ if u>0.6 and $F_X^{-1}(u) = 0$ if $u \leq 0.6$. The generator should therefore deliver the numerical value of the logical expression u>0.6.

R-command:

```
> n <- 100
> p <- 0.4
> u <- runif(n)
 > x <- as.integer(u > 0.6)     #(u > 0.6) is a logical vector
> x
  [1] 0 1 1 1 0 0 0 0 1 1 1 1 1 0 1 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1 1 0 1 0 0
 [38] 0 0 0 0 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 0 1 0 1 0 1 1 0 1 1 0 0 0 1 1 0 0
 [75] 1 0 0 0 1 1 0 1 1 0 0 0 1 1 0 1 1 0 0 0 1 0 0 0 1 0

> mean(x)
[1] 0.42
```

```
> var(x)
[1] 0.2460606
```

Compare the sample statistics with the theoretical moments. The sample mean of a generated sample should be approximately p =0 .4 and the sample variance should be approximately p(1−p)=0.24.

Example 2.6: (Geometric distribution)

Use the inverse transform method to generate a random sample of size 200 of geometric variates with parameter p = 1/4.

Solution:

The probability mass function is $f(x) = pq^x$, $x = 0, 1, 2, \ldots$, where $q = 1 - p$. At the points of discontinuity $x = 0, 1, 2, \ldots$, the cdf is $F(x) = 1 - q^{x+1}$. For each sample element we need to generate a random uniform u and solve

$$1 - q^x < u \leq 1 - q^{x+1}.$$

This inequality simplifies to $x < \log(1 - u)/\log(q) \leq x + 1$. The solution is $x + 1 = \left\lceil \frac{\log(1-u)}{\log(q)} \right\rceil$, where $\lceil t \rceil$ denotes the ceiling function (the smallest integer not less than t).

R-command:

```
> n <- 200
> p <- 0.25
> u <- runif(n)
> x <- ceiling(log(1-u) / log(1-p)) – 1
> x
  [1] 22  0  9  0  6  6  1  1  3  3  1  3  0  3  1  1  1  4  2  2  1  7  4  2 12
 [26]  0  7 10  4  1  2  2  2 10  7  3  8  4  1  0  6  0  0  0  0  3 10  4  2  0
 [51]  2  5  6 28  1  6  0  0  2  0  0  1  1 10  2  2  0  1  2  6  0 10  7  0  1
 [76]  3  0  1  3  0  2  6  2  1  3  4  1  0  1  3  4  0 10  0  0  6  2  1  6  2
[101]  0  0  7  0  1  1  4  4  9  0  5  2  7  9  0  1  0  2  4  2  0  2  1  4  8
[126]  1  2  5  2  0  1  2  0  5  0  0  1  3  3  3  0  1  0  1  7  0  4  7  4  0
[151]  5  4  4  2  0  0  7  0  0  0  1  3  3  1  3  7  4  2  5  1  0  1  0  0  0
[176]  2  4  0  1  2  3  3  7  0  1  1  6  3  1  2  4  0  2  3  0  2  6  6  5  3
```

Here again there is a simplification, because U and $1 - U$ have the same distribution. Also, the probability that $\log(1 - u)/\log(1 - p)$ equals an integer is zero. The last step can therefore be simplified to

```
>x<- floor(log(u) / log(1-p))
>x
[1]  0  6  0  6  0  0  4  4  1  1  2  1 13  1  3  4  3  1  2  2  4  0  1  2  0
```

```
 [26] 4 0 0 1 3 2 1 2 0 0 1 0 0 4 16 0 7 7 6 7 1 0 1 2 8
 [51] 2 0 0 0 2 0 5 5 2 7 10 4 3 0 2 2 9 3 1 0 5 0 0 5 3
 [76] 1 6 4 1 6 2 0 2 3 1 1 3 6 3 1 1 14 0 7 12 0 2 4 0 2
[101] 7 7 0 10 3 3 0 1 0 6 0 2 0 0 14 3 7 2 1 2 5 1 4 1 0
[126] 4 2 0 2 5 3 2 10 0 12 7 3 1 1 1 8 3 11 3 0 4 1 0 1 7
[151] 0 1 1 2 6 7 0 5 7 6 4 1 1 3 1 0 1 2 0 4 5 4 8 5 7
[176] 2 1 5 3 2 1 1 0 11 4 3 0 1 4 2 1 7 2 1 5 2 0 0 0 1
```

Note: 1. *Ceiling* takes a single numeric argument x and returns a numeric vector containing the smallest integers not less than the corresponding elements of x.
2. *floor* takes a single numeric argument x and returns a numeric vector containing the largest integers not greater than the corresponding elements of x.

2.3.3 Transformation Method

Many types of transformations other than the probability inverse transformation can be applied to simulate random variables. Some examples are,

Example 2.7:
If $Z \sim N(0,1)$, then $V = Z^2 \sim X^2(1)$.
Solution:
If $Z \sim N(0,1)$, then $V = Z^2$ has chi-square distribution with one degree of freedom. This transformation determines an algorithm for generating random chi-square variates with one degree of freedom.

1. Generate a random *z* from N(0,1).
2. Deliver $v = z^2$

R-commands:

```
> n<-50
> z<-rnorm(n,0,1)
> z
> v<-z*z
> v
[1] 0.305991600 0.936708505 0.176985090 0.019269297 4.706842164 8.082750751
 [7] 0.416388769 0.674569347 0.081927138 1.185487582 0.334547126 1.837157329
[13] 0.239230745 0.831482321 2.606537497 0.026806421 0.031947524 0.002424443
[19] 0.046569952 0.006557288 1.062038943 1.329896308 0.417741028 0.006213941
[25] 0.893791994 1.493647667 1.177504868 0.044944794 0.253231402 0.207673422
[31] 0.611983256 0.298337772 1.073820376 1.189793711 2.417628174 0.003843653
[37] 0.571621345 2.168156633 2.408532957 0.025244563 0.363919032 1.350905815
[43] 2.433898855 0.239302537 2.627804748 0.510848305 0.471539636 0.886617652
[49] 2.204411292 0.501172239
```

Example 2.8

If $U \sim X^2$(m) and $V \sim X^2$(n) are independent, then $F = \frac{U/m}{V/n}$ has the F distribution with (m, n) degrees of freedom.

Solution :

If U follows Chi-square with m degree of freedom an V follows Chi-square with n degree of freedom and they are independent then $F = \frac{U/m}{V/n}$ has F distribution with (m ,n) degrees of freedom. This transformation determines an algorithm for generating random $F_{(m,n)}$variates.

1. Generate a random u from X^2(m).
2. Generate a random v from X^2(n).
3. Deliver $f = \frac{u/m}{v/n}$

R-commands:

```
>n<-10
> df1<-25
> df2<-30
> u<-rchisq(n, df1)
>  u
> v<-rchisq(n,df2)
> v
> f<-(u/df1)/(v/df2)
> f
 [1] 1.0321807 0.9670168 2.0695526 0.7978024 1.1737535 0.8972368 0.9310880
 [8] 1.0385090 1.5052634 0.6193445 0.3652712 0.7769930 1.5944072 1.3368636
[15] 1.0971476 1.1805528 0.8509757 1.0530814 0.7395804 0.6777567 0.3872551
[22] 0.7401450 1.0015335 1.4481652 0.5785281 1.0139439 1.2096411 0.8026473
[29] 1.0641119 0.6810229 0.7686142 2.4284338 1.1694846 1.1182540 1.8254075
[36] 0.8226599 1.0921456 1.0326177 0.9808855 1.2804703 1.1069543 0.8580444
[43] 0.7096639 0.6385750 1.3748067 0.9581666 1.4010971 1.7270581 0.5388314
[50] 0.4130219
```

Example 2.9:

If Z ~ N(0,1) and V ~ X^2(n) are independent, then T = $\frac{Z}{\sqrt{V/n}}$ has the Student t distribution with n degrees of freedom.

Solution

If Z follows N (0,1) and V follows Chi-square distribution with n degree of freedom and they are independent then T = $\frac{Z}{\sqrt{V/n}}$ has the Student t distribution with n degrees of freedom. This transformation determines an algorithm for generating random t variates with n degree of freedom.

1. Generate a random z from N(0,1).
2. Generate a random v from X^2(n).
3. Deliver t = $\frac{z}{\sqrt{v/n}}$

R-commands:

```
> n <-100
> df1<-30
> z <- rnorm(n,0,1)
> z
> v <-rchisq(n, df1)
> v
> t <- z/sqrt(v/n)
>t
```

```
 [1]  0.26446379  1.39800807  3.92791932 -1.76426156  1.89240314  1.94872727
 [7] -4.36731846 -0.12477115  1.05737110  3.10173033 -2.66401373 -0.41518548
[13] -3.60220144  0.90638595  2.05071659  0.20169017  1.22424344 -1.52899330
[19]  2.34304814 -0.10279873  0.35987186  0.13248595  0.78447612  0.97159389
[25]  1.41866892  1.16106491  3.51499716  1.62117628 -0.03927726 -4.09739243
[31]  0.14825550 -2.64193688  2.01529940 -1.51096553 -1.08861161  1.60956835
[37] -1.51931718  1.42138984 -2.24604129 -2.42070540  1.39707719  0.91635615
[43]  0.16044138 -1.63285685  0.60393770  2.22841710 -0.27656098  1.26268787
[49]  1.56338839 -0.03879602  0.47203525 -0.71188275 -1.21320443 -0.48620090
[55] -3.58978950  0.45394887  2.11485752  1.06356104  2.42630003 -0.02625576
[61] -3.41974745 -0.88714154 -1.26062016  1.52293729  1.41621213  3.41113041
[67] -0.54567125  1.61609607 -0.65802077 -1.09476607  0.92254807 -1.57262214
[73]  2.13950748  0.71165649 -0.26277114 -1.72264936 -1.91914977 -5.64431301
[79]  1.25059713 -0.57808189 -1.05332032  1.69372271  3.67224005 -0.05815857
[85]  0.33717586  1.80094828  3.59441233  0.39715328 -0.10471523  0.21402481
[91]  0.34962907  0.53354374 -1.41554722 -0.45955978 -2.79763750 -0.45922718
[97] -0.93228173  1.59851480 -0.07998800 -1.92484870
```

Example 2.10:

If U, V ~ Unif(0,1) are independent, then $Z_1 = \sqrt{-2\log U}\cos(2\pi V)$,

$Z_2 = \sqrt{-2\log V}\sin(2\pi U)$ are independent standard normal variables.

Solution:

If U, V follows Uniform (0,1) distribution and they are independent, then $Z_1 = \sqrt{-2\log U}\cos(2\pi V)$ and $Z_2 = \sqrt{-2\log V}\sin(2\pi U)$ has independent standard normal distribution . This transformation determines an algorithm for generating two independent standard Normal variates.

1. Generate a random u from Uniform (0,1).
2. Generate a random v from Uniform (0,1).

3. Deliver $z_1=\sqrt{-2\ \text{logu}}\ \cos(2\pi v)$ and $z_2=\sqrt{-2\ \text{logv}}\ \sin(2\pi u)$

R-commands:

```
> n<-10
> u<-runif(n)
> u
> v<-runif(n)
> v
> z1<-sqrt(-2*log(u))*cos(2*pi*v)
> z1
 [1]  1.1250714  0.7727131 -0.3350260 -0.1239726  0.1766388 -0.2534140
 [7] -1.6997664  1.6549950 -0.1669520  1.2223246

> z2<-sqrt(-2*log(v))*sin(2*pi*u)
> z2
  [1] -0.0586010 -0.2061662 -0.4627048 -0.1096821 -0.6211043  0.8008360
 [7]  1.1698148  0.2115135 -0.8181893  0.8186927
```

Example 2.11:

If $U \sim$ Gamma(r, λ) and $V \sim$ Gamma(s, λ) are independent, then $X = \frac{U}{U+V}$ has the Beta(r, s) distribution.

Solution:

The following relation between beta and gamma distributions provides another beta generator.

If $U \sim$ Gamma(r, λ) and $V \sim$ Gamma(s, λ) are independent, then X= $\frac{U}{U+V}$ has the Beta(r, s) distribution. This transformation determines an algorithm for generating random Beta (a, b) variates.

1. Generate a random u from Gamma (a, 1).
2. Generate a random v from Gamma (b, 1).
3. Deliver $x = \frac{u}{u+v}$

This method is applied below to generate a random Beta (3, 2) sample.

R-commands:

```
> n <- 50
> a <- 3
> b <- 2
> u <- rgamma(n, shape=a, rate=1)
> v <- rgamma(n, shape=b, rate=1)
```

```
> x <- u / (u + v)
> x
 [1] 0.59648930 0.24795800 0.47529540 0.73482820 0.57455776 0.46957939
 [7] 0.16626121 0.34215751 0.90528863 0.66933936 0.91986937 0.64417831
[13] 0.60251166 0.29869794 0.66538529 0.59687444 0.60106581 0.55351348
[19] 0.61768632 0.22185070 0.70750753 0.82545663 0.64704160 0.78169584
[25] 0.50398825 0.68894688 0.91392111 0.26920499 0.80817692 0.09896981
[31] 0.63780512 0.67690692 0.64038829 0.55494909 0.88168902 0.47601543
[37] 0.70737323 0.74133759 0.79254969 0.47663673 0.85928021 0.29155043
[43] 0.66012778 0.65084000 0.51040890 0.24622534 0.44460079 0.84453544
[49] 0.66177467 0.13825054
```

Example 2.12:

If U, $V \sim$ Unif(0,1) are independent, then $X = \left\lfloor 1 + \frac{\log(V)}{\log(1-(1-\theta)^U)} \right\rfloor$ has the Logarithmic(θ) distribution, where $\lfloor x \rfloor$ denotes the integer part of x.

Solution:

This example provides another, more efficient generator for the logarithmic distribution. If U, V are independent Uniform (0,1) random variables, then

$$X = \left\lfloor 1 + \frac{\log(V)}{\log(1-(1-\theta)^U)} \right\rfloor$$

has the Logarithmic(θ) distribution . This transformation provides a simple and efficient generator for the logarithmic distribution.

1. Generate u from Uniform (0, 1).
2. Generate v from Uniform (0, 1).
3. Deliver $x = \left\lfloor 1 + \frac{\log(V)}{\log(1-(1-\theta)^U)} \right\rfloor$

R-commands:

```
> n <- 1000
> theta <- 0.5
> u <- runif(n)                    #generate logarithmic sample
> v <- runif(n)
> x <- floor(1 + log(v) / log(1 - (1 - theta)^u))
> x
 [1] 1 1 3 2 6 1 3 1 1 2 1 1 7 1 1 1 1 1 2 1 2 1 1 1 1 1 1 6 1 1 1 1 1 1 2 1 1
[38] 2 2 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 3 1 1 3 1 2 1 2 1 1 1 2 1 1 1 1 1
[75] 1 2 1 1 1 1 1 1 1 1 1 1 1 2 4 1 1 1 2 1 1 4 1 1 1 1
```

2.3.4 Sums and Mixtures

Sums and mixtures of random variables are special types of transformations. In this section we focus on sums of independent random variables (convolutions) and several examples of discrete and continuous mixtures.

Convolutions

Let $X_1, \ldots, X_n$ be independent and identically distributed with distribution $X_j \sim X$, and let $S = X_1 + \cdots + X_n$. The distribution function of the sum S is called the n-fold convolution of X and denoted $F_X^{*(n)}$. It is straightforward to simulate a convolution by directly generating $X_1, \ldots, X_n$ and computing the sum.

Several distributions are related by convolution. If $\nu > 0$ is an integer, the chi-square distribution with ν degrees of freedom is the convolution of ν i.i.d. squared standard normal variables. The negative binomial distribution NegBin(r, p) is the convolution of r i.i.d. Geom(p) random variables. The convolution of r independent Exp (λ) random variables has the Gamma(r, λ) distribution.

In R it is of course easier to use the functions rchisq, rgeom and rnbinom to generate chi-square, geometric and negative binomial random samples. The following example is presented to illustrate a general method that can be applied whenever distributions are related by convolutions.

Example 2.13: (Chi-square)

This example generates a chi-square X^2 random variable as the convolution of ν squared normal. If $Z_1, \ldots, Z_\nu$ are iid N(0,1) random variables, then
$V = Z_1^2 + Z_2^2 + \cdots + Z_\nu^2$ has the $\chi^2(\nu)$ distribution.
Steps to generate a random sample of size n from $\chi 2(\nu)$ are as follows.

1. Fill an $n \times \nu$ matrix with $n\nu$ random N(0,1) variates.
2. Square each entry in the matrix (1).
3. Compute the row sums of the squared normals. Each row sum is one random observation from the $\chi^2(\nu)$ distribution.
4. Deliver the vector of row sums.

An example with $n = 1000$ and $\nu = 2$ is shown below.

```
n <- 1000
nu <- 2
X <- matrix(rnorm(n*nu), n, nu)^2                    #matrix of sq. normals
                    #sum the squared normals across each row: method 1
y <- rowSums(X)
                                  #method 2
y <- apply(X, MARGIN=1, FUN=sum)          #a vector length n
```

mean(y)
mean(y^2)

Mixtures

A random variable *X* is a discrete mixture if the distribution of *X* is a weighted sum $F_X(x) = \sum \theta_i F_{x_i}(x)$ for some sequence of random variables X_1, X_2, . . . and $\theta i > 0$ such that $\sum \theta_i = 1$. The constants θ_i are called the mixing weights or mixing probabilities. Although the notation is similar for sums and mixtures, the distributions represented are different.

A random variable *X* is a continuous mixture if the distribution of *X* is $F_X(x) = \int_{-\infty}^{+\infty} F_{X|Y=y}(x) f_Y(y)\, dy$ for a family $X|Y = y$ indexed by the real numbers *y* and weighting function f_Y such that $\int_{-\infty}^{+\infty} F_Y(y) dy = 1$

Compare the methods for simulation of a convolution and a mixture of normal variables. Suppose $X_1 \sim N(0, 1)$ and $X_2 \sim N(3, 1)$ are independent.
The notation $S = X_1 + X_2$ denotes the *convolution* of X_1 and X_2. The distribution of *S* is normal with mean $\mu_1 + \mu_2 = 3$ and variance $\sigma_1^2 + \sigma_2^2 = 2$.

To simulate the *convolution*:
1. Generate $x_1 \sim$ N (0, 1).
2. Generate $x_2 \sim$ N (3, 1).
3. Deliver $s = x_1 + x_2$.

We can also define a 50% normal mixture X, denoted $F_X(x) = 0.5F_{X1}(x)+0.5F_{X2}(x)$. Unlike the convolution above, the distribution of the mixture X is distinctly non-normal; it is bimodal.
To simulate the mixture:
1. Generate an integer $k \in \{1, 2\}$, where P(1) = P(2) = 0.5.
2. If k = 1 deliver random x from N(0, 1);
   if k = 2 deliver random x from N(3, 1)

Unit III

3.1Methods to find solution of non-linear Equation

(A) Bisection Method

The bisection method is a root-finding method that repeatedly bisects an interval and then selects a subinterval in which a root must lie for further processing. It is a very simple and robust method, but it is also relatively slow. Because of this, it is often used to obtain a rough approximation to a solution which is then used as a starting point for more rapidly converging methods.

The method is applicable for numerically solving the equation $f(x) = 0$ for the real variable x, where f is a continuous function defined on an interval $[a, b]$ and where $f(a)$ and $f(b)$ have opposite signs. In this case a and b are said to bracket a root since, by the intermediate value theorem, the continuous function f must have at least one root in the interval (a, b).

At each step the method divides the interval in two by computing the midpoint $c = (a + b) / 2$ of the interval and the value of the function $f(c)$ at that point. Unless c is itself a root (which is very unlikely, but possible) there are now only two possibilities: either $f(a)$ and $f(c)$ have opposite signs and bracket a root, or $f(c)$ and $f(b)$ have opposite signs and bracket a root. The method selects the subinterval that is guaranteed to be a bracket as the new interval to be used in the next step. In this way an interval that contains a zero of f is reduced in width by 50% at each step. The process is continued until the interval is sufficiently small.

Explicitly, if $f(a)$ and $f(c)$ have opposite signs, then the method sets c as the new value for b, and if $f(b)$ and $f(c)$ have opposite signs then the method sets c as the new a. (If $f(c)=0$ then c may be taken as the solution and the process stops.) In both cases, the new $f(a)$ and $f(b)$ have opposite signs, so the method is applicable to this smaller interval.

Iteration Process:

Given the interval $[a, b]$, define $c = (a + b)/2$. Then

- if $f(c) = 0$ (unlikely in practice), then halt, as we have found a root,
- if $f(c)$ and $f(a)$ have opposite signs, then a root must lie on $[a, c]$, so assign $b = c$,
- else $f(c)$ and $f(b)$ must have opposite signs, and thus a root must lie on $[c, b]$, so assign $a = c$.

Halting Conditions:

There are three conditions which may cause the iteration process to halt**:**

1. As indicated, if f(c) = 0.
2. We halt if both of the following conditions are met:
   - The width of the interval (after the assignment) is sufficiently small, that is b- $a < \varepsilon_{step}$, and
   - The function evaluated at one of the end point $|f(a)|$ *or* $|f(b)| < \varepsilon_{abs}$.
3. If we have iterated some maximum number of times, say N, and have not met Condition 1, we halt and indicate that a solution was not found.

If we halt due to Condition 1, we state that c is our approximation to the root. If we halt according to Condition 2, we choose either a or b, depending on whether |f(a)| < |f(b)| or |f(a)| > |f(b)|, respectively.

If we halt due to Condition 3, then we indicate that a solution may not exist (the function may be discontinuous).

Example 3.1:

Find the root of f(x) = x^2 - 3. Let ε_{step} = 0.01, ε_{abs} = 0.01 and start with the interval [1, 2].

Solution:

Table 3.1: Bisection Method Applied to f(x) = x^2 - 3.

a	b	f(a)	f(b)	c=(a+b)/2	f(c)	Update	New(b-a)
1.0	2.0	-2.0	1.0	1.5	-0.75	a=c	0.5
1.5	2.0	-0.75	1.0	1.75	0.062	b=c	0.25
1.5	1.75	-0.75	0.0625	1.625	-0.359	a=c	0.125
1.625	1.75	-0.3539	0.0625	1.6875	-0.1523	a=c	0.0625
1.6875	1.75	-0.1523	0.0625	1.7188	-0.0457	a=c	0.0313
1.7188	1.75	-0.0457	0.0625	1.7344	0.0081	b=c	0.0156
1.71988	1.7344	-0.0457	0.0081	1.7266	-0.0189	a=c	0.0078

Thus, with the seventh iteration, we note that the final interval, [1.7266, 1.7344], has a width less than 0.01 and |f(1.7344)| < 0.01, and therefore we chose b =1.7344 to be our approximation of the root.

R programme to find the root of f(x) = x^2 - 3. Let ε_{step} = 0.01, ε_{abs} = 0.01 and start with the interval [1, 2].

R-Commands:

```
bisec<-function(a,b){
 f <- function(x) {
   x^2-3
}
```

```
  it <- 0
  eps <- 0.001
  r <- seq(a, b, length=3)
  x <- c(f(r[1]), f(r[2]), f(r[3]))
  if (x[1] * x[3] > 0)
     stop("f does not have opposite sign at endpoints")

  while(it < 1000 && abs(x[2]) > eps) {
     it <- it + 1
     if (x[1]*x[2] < 0) {
        r[3] <- r[2]
        x[3] <- x[2]
     } else {
        r[1] <- r[2]
        x[1] <- x[2]
     }
     r[2] <- (r[1] + r[3]) / 2
     x[2] <- f(r[2])
     cat(it,c(r[1],r[3], x[1],x[3],r[2],x[2]),"\n")
   }
 }
 bisec(1,2)
```

Output:

1 1.5 2 -0.75 1 1.75 0.0625
2 1.5 1.75 -0.75 0.0625 1.625 -0.359375
3 1.625 1.75 -0.359375 0.0625 1.6875 -0.1523438
4 1.6875 1.75 -0.1523438 0.0625 1.71875 -0.04589844
5 1.71875 1.75 -0.04589844 0.0625 1.734375 0.008056641
6 1.71875 1.734375 -0.04589844 0.008056641 1.726562 -0.01898193
7 1.726562 1.734375 -0.01898193 0.008056641 1.730469 -0.005477905
8 1.730469 1.734375 -0.005477905 0.008056641 1.732422 0.001285553
9 1.730469 1.732422 -0.005477905 0.001285553 1.731445 -0.00209713
10 1.731445 1.732422 -0.00209713 0.001285553 1.731934 -0.0004060268

(B) Newton-Raphson Method

In numerical analysis, Newton's method (also known as the Newton–Raphson method), named after Isaac Newton and Joseph Raphson, is a method for finding successively better approximations to the roots (or zeroes) of a real-valued function.

The Newton–Raphson method in one variable is implemented as follows:

The method starts with a function f defined over the real numbers x, the function's derivative f', and an initial guess x_0 for a root of the function f. If the function satisfies the assumptions made in the derivation of the formula and the initial guess is close, then a better approximation x_1 is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

The process is repeated as

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

until a sufficiently accurate value is reached.

Where,
x_i = value of the root at iteration i
x_{i+1} = a revised value of the root at iteration i +1
$f(x_i)$ = value of the function at iteration i
$f'(x_i)$ = derivative of f(x) evaluated at iteration i

This algorithm is first in the class of Householder's methods, succeeded by Halley's method. The method can also be extended to complex functions and to systems of equations.

Example 3.2:
Use the Newton-Raphson iteration method to estimate the root of the following function employing an initial guess of x_0 =3,

$$f(x) = x^2 - 2x - 2$$

Solution:

$$f(x) = x^2 - 2x - 2$$

Let us find the derivative of the function first,

$$f'(x) = 2x - 2$$

The initial guess is $x_0 = 3$

Table 3.2:

i	Iteration	x_i	$f(x_i) = x^2 - 2x - 2$	$f'(x_i) = 2x - 2$	$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$
0	1^{st}	3	$(3)^2 - 2(3) - 2 = 1$	$2(3) - 2 = 4$	$3 - (1/4) = 2.75$
1	2^{nd}	2.75	0.0625	3.5	2.73214
2	3^{rd}	2.73214	3.0898×10^{-4}	3.46428	2.73205
3	4^{th}	2.73205			

So the answer will be 2.732 (in three decimal places).

R- Commands:

```
NRM<-function(x){
fx<-(x^2-2*x-2)
dfx<-(2*x-2)
xf<-x-(fx/dfx)
it<-1
while(abs(fx)>0.00001&it<100){
x<-xf
fx<-(x^2-2*x-2)
dfx<-(2*x-2)
xf<-x-(fx/dfx)
it<-it+1
}
list(a=xf, iteration=it)
}
NRM(3)
```

Output:

```
$a
[1] 2.732051

$iteration
[1] 4
```

Unit IV

4.1 Iterative methods for solving linear system of equations

As a numerical technique, Gaussian elimination is rather unusual because it is direct. That is, a solution is obtained after a single application of Gaussian elimination. Once a "solution" has been obtained, Gaussian elimination offers no method of refinement. The lack of refinements can be a problem because, as the previous section shows, Gaussian elimination is sensitive to rounding error.

Numerical techniques more commonly involve an iterative method. For example, in calculus you probably studied Newton's iterative method for approximating the zeros of a differentiable function. In this section you will look at two iterative methods for approximating the solution of a system of n linear equations in n variables.

(A) Jacobi Method

The first iterative technique is called the **Jacobi method,** after Carl Gustav Jacob Jacobi (1804–1851). It is simplest iterative method for solving linear system Ax = b. This method makes two assumptions: (1) that the system given by

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ \vdots \quad\quad \vdots \quad\quad\quad\quad \vdots \quad\quad \vdots & \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned}$$

has a unique solution and (2) that the coefficient matrix *A* has no zeros on its main diagonal. If any of the diagonal entries a_{11}, $a_{22,}$ ... , a_{nn} are zero, then rows or columns must be interchanged to obtain a coefficient matrix that has nonzero entries on the main diagonal.

Main idea of Jacobi

To begin, solve the 1st equation for x_1, the 2nd equation for x_2 and so on to obtain the rewritten equations:

$$x_1 = \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3 - \cdots - a_{1n}x_n)$$

$$x_2 = \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3 - \cdots - a_{2n}x_n)$$

$$\vdots$$

$$x_n = \frac{1}{a_{nn}}(b_n - a_{n1}x_1 - a_{n2}x_2 - \cdots - a_{n,n-1}x_{n-1})$$

Then make an initial guess of the solution $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, \cdots, x_n^{(0)})$. Substitute these values into the right hand side the of the rewritten equations to obtain the first approximation, $(x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, \cdots, x_n^{(1)})$. This accomplishes one iteration.

In the same way, the second approximation $(x_1^{(2)}, x_2^{(2)}, x_3^{(2)}, \cdots, x_n^{(2)})$ is computed by substituting the first approximation's *x* -vales into the right hand side of the rewritten equations.

By repeated iterations, we form a sequence of approximations

$x^{(k)} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \cdots, x_n^{(k)})'$, k=1, 2, 3, ...

The Jacobi Method for each k $\geq$1, generates the components $x_i^{(k)}$ of $x^{(k)}$ from $x^{(K-1)}$ by,

$$x_i^{(k)} = \frac{1}{a_{ii}} \left[\sum_{\substack{j=1 \\ j \neq i}}^{n} (-a_{ij} x_j^{(k-1)}) + b_i \right] \quad \text{for i= 1, 2, ... ,n}$$

Example 4.1:

Use the Jacobi method to approximate the solution of the following system of linear equations.

$$\begin{aligned} 5x_1 - 2x_2 + 3x_3 &= -1 \\ -3x_1 + 9x_2 + x_3 &= 2 \\ 2x_1 - x_2 - 7x_3 &= 3 \end{aligned}$$

Continue the iterations until two successive approximations are identical when rounded to three significant digits.

Solution:

To begin, write the system in the form

$$\begin{aligned} x_1 &= -\frac{1}{5} + \frac{2}{5} x_2 - \frac{3}{5} x_3 \\ x_2 &= \frac{2}{9} + \frac{3}{9} x_1 - \frac{1}{9} x_3 \\ x_3 &= -\frac{3}{7} + \frac{2}{7} x_1 - \frac{1}{7} x_2 \end{aligned}$$

Because you do not know the actual solution, choose $x_1 = 0$, $x_2 = 0$, $x_3 = 0$ as a convenient initial approximation. So, the first approximation is

$$x_1 = -\frac{1}{5} + \frac{2}{5}(0) - \frac{3}{5}(0) = -0.200$$

$$x_2 = \frac{2}{9} + \frac{3}{9}(0) - \frac{1}{9}(0) \approx 0.222$$

$$x_3 = -\frac{3}{7} + \frac{2}{7}(0) - \frac{1}{7}(0) \approx -0.429$$

Continuing this procedure, you obtain the sequence of approximations shown in following Table.

Table 4.1

n	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
x_1	0.000	-0.200	0.146	0.192	0.181	0.185	0.186	0.186
x_2	0.000	0.222	0.203	0.328	0.332	0.329	0.331	0.331
x_3	0.000	-0.429	-0.517	-0.416	-0.421	-0.424	-0.423	-0.423

Because the last two columns in above Table are identical, you can conclude that to three significant digits the solution is

$$x_1 = 0.186,\ x_2 = 0.331,\ x_3 = -0.423$$

(B) The Gauss-Seidel Method

You will now look at a modification of the Jacobi method called the Gauss-Seidel method, named after Carl Friedrich Gauss (1777–1855) and Philipp L. Seidel (1821–1896). This modification is no more difficult to use than the Jacobi method, and it often requires fewer iterations to produce the same degree of accuracy.

With the Jacobi method, the values of x_i obtained in the nth approximation remain unchanged until the entire $(n+1)^{th}$ approximation has been calculated. With the Gauss- Seidel method, on the other hand, you use the new values of each x_i as soon as they are known. That is, once you have determined x_1 from the first equation, its value is then used in the second equation to obtain the new x_2. Similarly, the new x_1 and x_2 are used in the third equation to obtain the new x_3 and so on.

The Gauss-Seidel Method for each k $\geq$1, generates the components $x_i^{(k)}$ of $x^{(k)}$ from $x^{(K-1)}$ by,

$$x_i^{(k)} = \frac{1}{a_{ii}}\left[-\sum_{j=1}^{i-1}\left(a_{ij}x_j^{(k)}\right) - \sum_{j=i+1}^{n}\left(a_{ij}x_j^{(k-1)}\right) + b_i\right] \quad \text{for } i = 1, 2, \ldots, n$$

Namely,

Example 4.2:
Use the Gauss-Seidel iteration method to approximate the solution to the system of linear equations,

$$5x_1 - 2x_2 + 3x_3 = -1$$
$$-3x_1 + 9x_2 + x_3 = 2$$
$$2x_1 - x_2 - 7x_3 = 3$$

Continue the iterations until two successive approximations are identical when rounded to three significant digits.

Solution:
To begin, write the system in the form

$$x_1 = -\frac{1}{5} + \frac{2}{5}x_2 - \frac{3}{5}x_3$$
$$x_2 = \frac{2}{9} + \frac{3}{9}x_1 - \frac{1}{9}x_3$$
$$x_3 = -\frac{3}{7} + \frac{2}{7}x_1 - \frac{1}{7}x_2$$

Because you do not know the actual solution, choose $x_1 = 0$, $x_2 = 0$, $x_3 = 0$ as a convenient initial approximation. So, obtain the following new value of x_1

$$x_1 = -\frac{1}{5} + \frac{2}{5}(0) - \frac{3}{5}(0) = -0.200$$

Now that you have a new value for x_1, however, use it to compute a new value for x_2 That is,

$$x_2 = \frac{2}{9} + \frac{3}{9}(-0.200) - \frac{1}{9}(0) \approx 0.156$$

Similarly, use x_1=-0.200 and x_2=0.156 to compute a new value for x_3.That is,

$$x_3 = -\frac{3}{7} + \frac{2}{7}(-0.200) - \frac{1}{7}(0.156) \approx -0.508$$

So the first approximation is x_1=-0.200, x_2=0.156 and x_3=-0.508.
Continued this iterations produce the sequence of approximation shown in following table,

Table

n	**0**	**1**	**2**	**3**	**4**	**5**
x_1	0.000	-0.200	0.167	0.191	0.186	0.186
x_2	0.000	0.156	0.334	0.333	0.331	0.331
x_3	0.000	-0.508	-0.429	-0.422	-0.423	-0.423

Note that after only five iterations of the Gauss-Seidel method, you achieved the same accuracy as was obtained with seven iterations of the Jacobi method.