Connecting with Data: Exploring Statistical Concepts using R

Leslie Chandrakantha
lchandra@jjay.cuny.edu
Department of Mathematics & Computer Science
John Jay College of Criminal Justice of CUNY, USA

Abstract*: R is a language and environment for statistical computing and graphics. R has proven to be an excellent tool for understanding and visualizing statistical concepts. In this paper, we present how to use R for data analysis, elementary simulation, and understanding concepts in an introductory level statistics course. We consider examples from descriptive statistics, normal distributions, sampling distributions, confidence intervals, and hypothesis testing to demonstrate the use of R.*

1. Introduction

Difficulty in understanding statistics concepts has been a major problem for instructors for many years. This has driven them to explore new techniques to deliver instructions that would improve the academic performances of students. Using technology as a tool to teach statistics was one of those techniques. Nowadays, it is hard to imagine teaching statistics at a college level without using some form of technology. Research has shown that the use of technology enhances a student's ability to grasp the abstract concepts of statistics [1]. Many instructors use technology in their classrooms to explain difficult statistical concepts such as sampling distributions, confidence intervals and hypothesis testing at an introductory level. Due to the abstract nature of these topics, traditional way of teaching using books and lecture-based instructions does not give a good understanding of the concepts to many students [2]. While software has been available for doing statistical analysis, the use of technology in teaching and learning statistics is continuously evolving. There are several types of technology being used in statistics instructions, including but not limited to: statistical packages and spreadsheets, Web or computer-based tools, graphing calculators, and programming languages. Calculators and computers reduce the computational burden and allow more extensive exploration of statistical concepts. Statistical packages such as SAS, SPSS and MINITAB have been widely used in statistics classes at a college level. Presently, R is increasingly being used in statistics instruction including in an introductory level. In this paper, we describe how to use R to teach introductory statistics concepts.

There are many good reasons to use R for teaching statistics [8]. R is a free and multi-platform (Windows, Unix/Linux, Mac OS X) and programming language. It has excellent graphical capabilities that are very useful in learning statistical concepts. R offers much more opportunities to modify and optimize graphs and charts. R charts are easily made interactive which allow users to play with data. In SPSS and Excel, graphs and charts are not that interactive as in R where you can create only basic and simple charts. R can read data files from many different formats. It has an extensive collection of add on packages as well. These are some of the pedagogical benefits offered by R over other software packages. Many introductory and higher level statistics instructors are now using R to teach and perform statistical data analysis. Although it is an initial challenge for students to write statements in the command line, R can be used to conduct data analysis effectively. R can easily generate random samples from many data sets and a variety of probability distributions. A valuable introduction to R for introductory courses is given in [5]. We use examples to demonstrate the use of R in teaching statistics in several important topics.

2. Examples

2.1 Descriptive Statistics

To describe how to use R, our first example is calculating descriptive statistics and creating related graphical tools. Variables with small data sets can be directly entered at the keyboard, but this approach is limited. R has many other ways to input data. Data can be read from text files, csv (comma-separated values) files, and attached packages.

R works with data structures such as vectors (one-dimensional array) and data frames (two-dimensional arrays). When R is started, we will see a window that is called the R console. This is where we type our commands and see the text results. Graphics appear in a separate window. The > is called the prompt, where R commands are written. The results of an R command can be assigned to a variable using <- or =. In this paper, we will use <-. In R, a vector is a sequence of data values of the same type. The function, *c*, is used to create vectors from scalars. The following statements create a vector and display it.

```
> x <- c(2, 4, 6, 8, 10)
> x
[1]  2  4  6  8 10
```

Once we have a vector of numbers, we can apply built-in functions to get useful statistical summaries and visual displays.

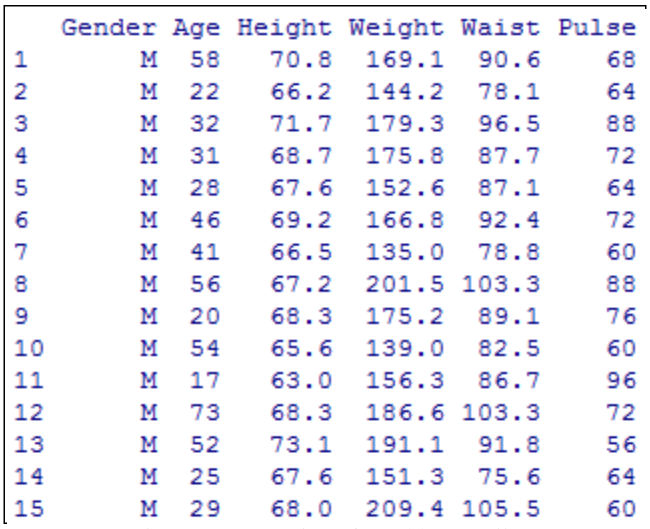
In this paper we read data input from csv (comma-separated values) files. *read.csv* function imports data from a csv file. To read data into a data frame named *mydata* from a csv file, we type

```
> mydata  <-  read.csv(file.choose(), header = TRUE)
```

After typing the above command, you can manually select the directory and the file where your dataset is located. The first line of the file should have a name for each variable. However, if the first row does not contain the names of the variables then the header argument should be set to FALSE. The *attach* function can be used to make objects contained in data frames accessible. The following command allows the user to access data in *mydata* data frame:

```
> attach(mydata)
```

Now we read the contents of a csv file named HealthData into a data frame named *mydata* using the above commands. This file has the health data information (gender, age, height, weight, waist and pulse rate) of 80 individuals. *Figure 2.1* shows an R snapshot of the data.



```
   Gender Age Height Weight Waist Pulse
1       M  58   70.8  169.1  90.6    68
2       M  22   66.2  144.2  78.1    64
3       M  32   71.7  179.3  96.5    88
4       M  31   68.7  175.8  87.7    72
5       M  28   67.6  152.6  87.1    64
6       M  46   69.2  166.8  92.4    72
7       M  41   66.5  135.0  78.8    60
8       M  56   67.2  201.5 103.3    88
9       M  20   68.3  175.2  89.1    76
10      M  54   65.6  139.0  82.5    60
11      M  17   63.0  156.3  86.7    96
12      M  73   68.3  186.6 103.3    72
13      M  52   73.1  191.1  91.8    56
14      M  25   67.6  151.3  75.6    64
15      M  29   68.0  209.4 105.5    60
```

Figure 2.1: Snapshot of *HealthData* File

To find the descriptive statistics of a variable (say Age), we can use the command *summary* in following way. It computes the mean, median, quartiles, minimum and maximum.

```
> summary(Age)

  Min.    1st Qu.    Median    Mean    3rd Qu.    Max.
 12.00     23.75     32.00     34.35    42.50     73.00
```

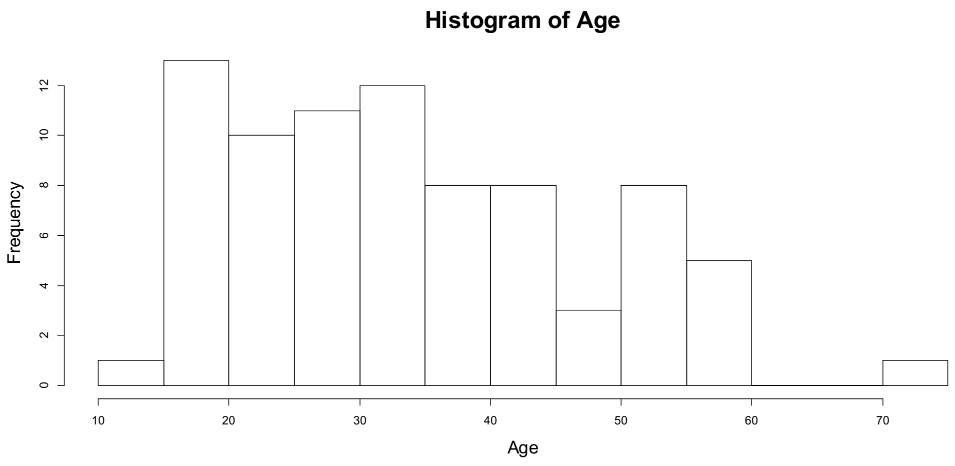
We can find the mean, median, minimum and maximum individually using the *mean*, *median*, *min* and *max* commands respectively. The *quantile* command returns the quartiles corresponding to given probabilities. The *sum* command returns the sum of the values in the vector. The *table* command returns the frequency table

To find the standard deviation and variance, the following commands can be used:

```
> sd(Age)

[1] 13.17564

> var(Age)

[1] 173.5975
```

R can be used to create graphical displays. The *hist* command creates a histogram while *stem* command creates a stem and leaf plot. The number of bars in a histogram can be specified using the breaks. The main option displays a title.

```
> hist(Age, breaks = 10, main = "Histogram of Age")
```

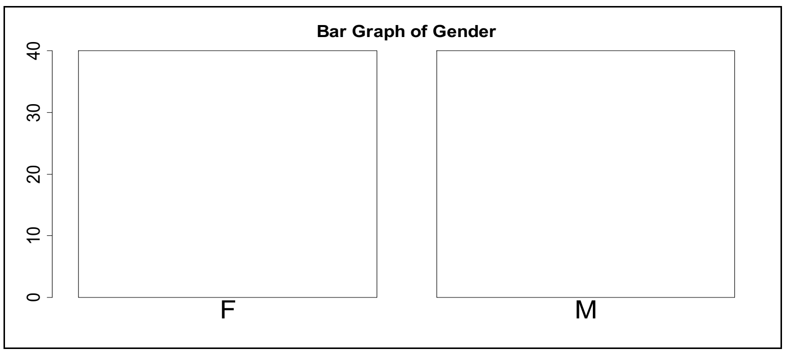


> stem(Age)

```
 1 | 2677788899
 2 | 0000223333455566778889999
 3 | 11122222334466777
 4 | 00011112455678
 5 | 2223345566789
 6 |
 7 | 3
```

To create a bar graph of a categorical variable, first count the frequencies using *table* command and then use the *barplot* command.

> y <- table(Gender)
> barplot(y, main = " Bar Graph of Gender")

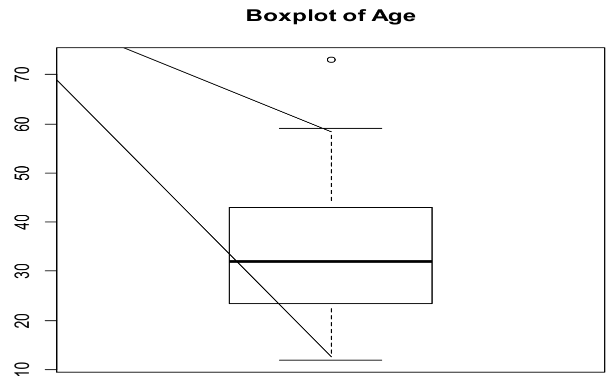


To obtain the relative frequencies for each category (M and F in this example), we could use

> y/length(Gender)
```
Gender
  F   M
0.5 0.5
```
The boxplot is used to summarize a dataset using the five number summary. The five number summary of a dataset include the minimum value, first quartile, median, third quartile, and the maximum value. The boxplots allow us to check the symmetry or the skewness of a dataset

and identify possible outliers. The following commands compute five number summary and create the boxplot.

```
> fivenum(Age)
[1] 12.0    23.5    32.0    43.0    73.0
> boxplot(Age, main = "Boxplot of Age")
```



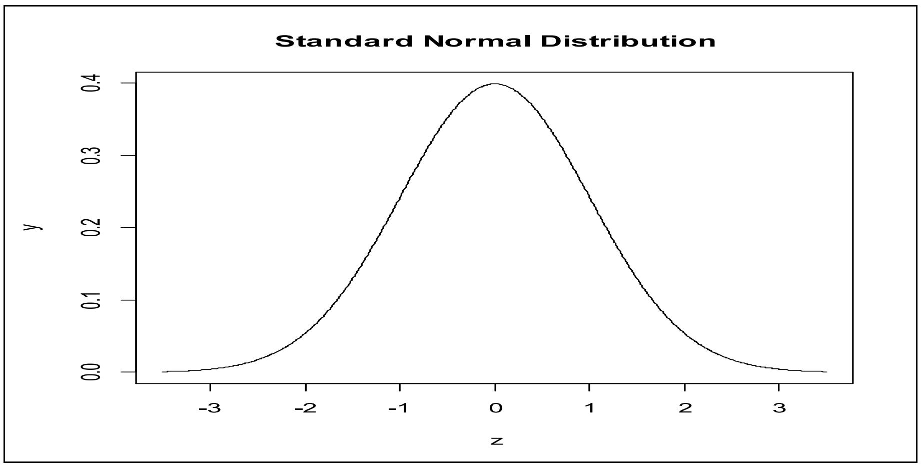
The above boxplot indicates that the Age data is slightly skewed to right and has one outlier.

2.2 Normal Distribution

Normal distribution is a very important topic in introductory statistics. Many instructors still use normal probability tables to find the normal probabilities. In this example, we show how to use R commands to find normal probabilities, quantiles and generating normal random variates. In a similar manner with appropriate commands, one can work with other probability distributions.

The following R code segment plots the standard normal distribution ($\mu = 0, \sigma = 1$) between -3.5 and 3.5.

```
> z <- seq(from = -3.5, to = 3.5, by = 0.01)
> y <-  dnorm(z,0,1)
> plot(z,y, type = "l", main = "Standard Normal Distribution")
```



The *seq* command creates a vector values from -3.5 to 3.5 with an increment of 0.01. The *dnorm(x, μ, σ)* function gives the height of the density function (pdf) at a value of x of the normal distribution with mean μ and standard deviation σ. The *plot* function plots the points and type = "l" option connects the points.

pnorm(x, μ, σ) function gives the area under the normal curve (cdf) with mean μ and standard deviation σ to the left of x. This is the probability that $P(X \leq x)$.

To calculate the probability $P(X \leq 15)$ of the normal distribution with $\mu = 20$ and $\sigma = 4$, we use the following command. The area representing $P(X \leq 15)$ is shown in *Figure 2.2*

```
> pnorm(15, 20, 4)
[1] 0.1056498
```

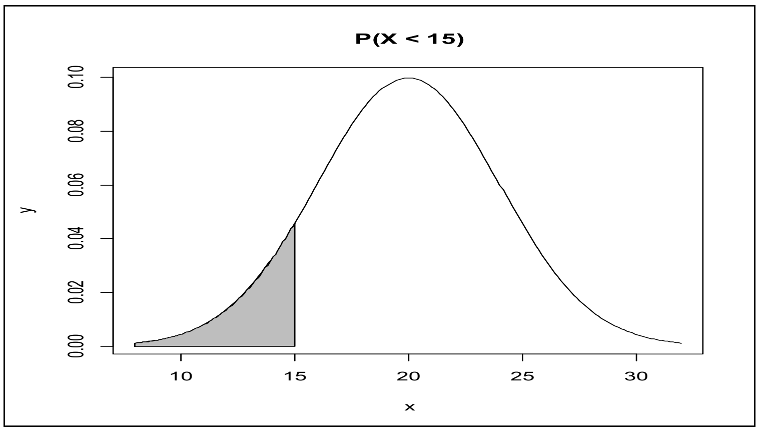


Figure 2.2: Area representing $P(X \leq 15)$

To compute the area above 15, ($P(X > 15)$), simply subtract the probability above from 1. In other words, it is 1 – *pnorm(15, 20, 4)*.

The *qnorm(p, μ, σ)* function gives the value at which the cdf { $P(X \leq x)$} of the normal distribution with mean μ and standard deviation σ is *p*. In other words, it computes the *p*th quantile of the normal distribution.

To find x such that $P(X \leq x) = 0.90$ in the normal distribution with $\mu = 20$ and $\sigma = 4$:

```
> qnorm(0.90, 20, 4)
[1] 25.12621
```

rnorm(n, μ, σ) function generates *n* random numbers from the normal distribution with mean μ and standard deviation σ. To generate 10 random numbers from the normal distribution with $\mu = 20$ and $\sigma = 4$:

```
> S <- rnorm(10, 20, 4)
> S
 [1] 27.42365  19.44874  21.67754  21.58621  14.44868  27.54056  24.74141  29.02238
 [9] 17.90459  14.52994
```

2.3 Sampling Distributions

Sampling distributions is a difficult concept for many students at an introductory level. Sampling distributions are important because inferential statistics are based on them. The sampling distribution of the mean is the probability distribution of the sample mean based on all possible simple random samples of the same size from the same population. We can use simulations to understand and visualize the following properties of the sampling distributions:

- The mean of all sample means is equal to the population mean (μ).
- The standard deviation of the sample means (known as the standard error) is equal to the population standard deviation divided by square root of the sample size$(\sigma/\sqrt{n})$.
- Sample means are more normal than individual observations.

The central limit theorem explains the shape of the sampling distribution. This theorem tells that for a population of any distribution, the distribution of the sample mean approaches a normal distribution as the sample size increases. The larger the sample size, the better the approximation.

As an example, we generate random samples from exponential distribution (with parameter λ = 2) and show that the sampling distribution of sample mean approaches a normal distribution as the sample size increases. We consider sample sizes of 10, 25, and 50. In each case we generate 10,000 random samples, compute the sample mean, and observe the distributional shape using histograms and normal probability plots. We use a *for* loop to generate 10,000 samples and compute sample means. The following R code does the simulation of the process described above.

```
> means <- c()
> for(i in 1: 10000)
{
+   y <-  rexp(10,2)
+   means[i] <- mean(y)
}
```

In the above code, *rexp(10,2)* command generates sample of 10 exponential random numbers with parameter λ = 2. The *means* vector holds 10,000 sample means. This gives the approximate sampling distribution of the sample mean. The approximate mean and standard deviation of the sampling distribution can be obtained by computing the mean and standard deviation of the sample means. They should be equal (or approximately equal) to μ (1/λ = 1/2 = 0.5) and $(\sigma/\sqrt{n})$ (= (1/2)/√10 = 0.15811) respectively as indicated in the properties of sampling distribution. R produces the mean and the standard deviation of the sample means using following commands:

```
> mean(means)
[1] 0.4963717
> sd(means)
[1] 0.1558105
```

To observe the shape of the approximate sampling distribution, we can create a histogram and a normal probability plot of sample means. This can be done using the command sequence given below. The *qqline* command adds the theoretical normal QQ-line.

```
> hist(means)
> qqnorm(means)
> qqline(means)
```

Figure 2.3 shows the histograms and normal probability plots for n = 10, 25, and 50. Students will be able to visualize that as the sample size increases, the shape of the distribution is becoming more normal and the sample means are less variable. Furthermore, they notice that for skewed data such as exponential data, the sampling distribution does not approach a normal distribution for sample sizes as large as 50 values.

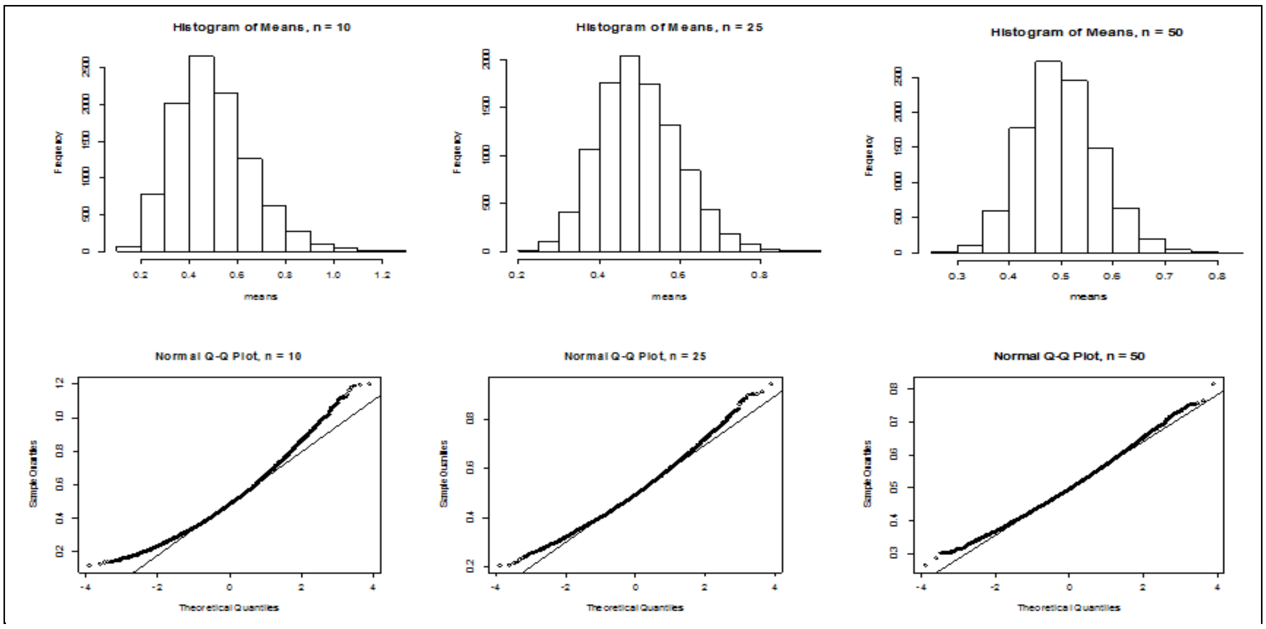


Figure 2.3: Histograms and Normal Probability Plots for n = 10, 25 and 50.

2.4 Statistical Inferences

A statistical inference is the process of making conclusions about population parameters based on sample information. Statistical inference topics such as confidence intervals and hypothesis testing play a major role in introductory statistics classes. Many students at an introductory level struggle to understand the concepts of statistical inferences [3]. R can enhance their understanding of concepts by allowing them to experiment and obtain the required results and visualizing them. In this section, we show how to use R in confidence intervals and hypothesis testing.

2.4.1 Confidence Intervals

Confidence intervals are commonly used to estimate the values of unknown population parameters. In this discussion, we consider estimating the population mean. Garfield, delMas & Chance [4] have noted that students should understand that the confidence interval estimates the unknown mean based on a random sample from the population. The level of confidence tells the probability that the method produces an interval that includes the true population parameter. We assume the population standard deviation (σ) is known. Confidence interval for mean μ is given as $\bar{X} \pm Z_{\alpha/2}\sigma/\sqrt{n}$, where $Z_{\alpha/2}$ is the value of the standard normal curve with the area (1-α) between critical points $-Z_{\alpha/2}$ and $Z_{\alpha/2}$, *n* is the sample size. The confidence level (1-α) is the probability that the confidence interval actually does contain the population mean μ, assuming the estimation process is repeated a large number of times.

Performing a simulation using R will allow students to understand the true meaning of confidence intervals. We use the same dataset (*HealthData* file) used in section 2.1 and consider the Age variable. This variable represents age of respondents. This will be considered as our population or sampling frame. We generate many samples of size 50 from this population and compute 95% confidence intervals for the mean age. The sample size 50 is large enough to assume that the sampling distribution of mean is approximately normal. The normality of the sampling distribution of mean is necessary for the validity of the confidence intervals. Normality of sample means can be verified using the following R code. It generates 10,000 random samples and computes the mean. The *sample* command generates a random sample from the population.

```
> means <- c()
> for(i  in  1:10000)
{
+   x <- sample(Age, 50, replace = FALSE)
+   means[i] <- mean(x)
}
```

The histogram and the normal probability plot in *Figure 2.4* confirm the approximate normality of the sampling distribution of means.

```
> hist(means)
> qqnorm(means)
```

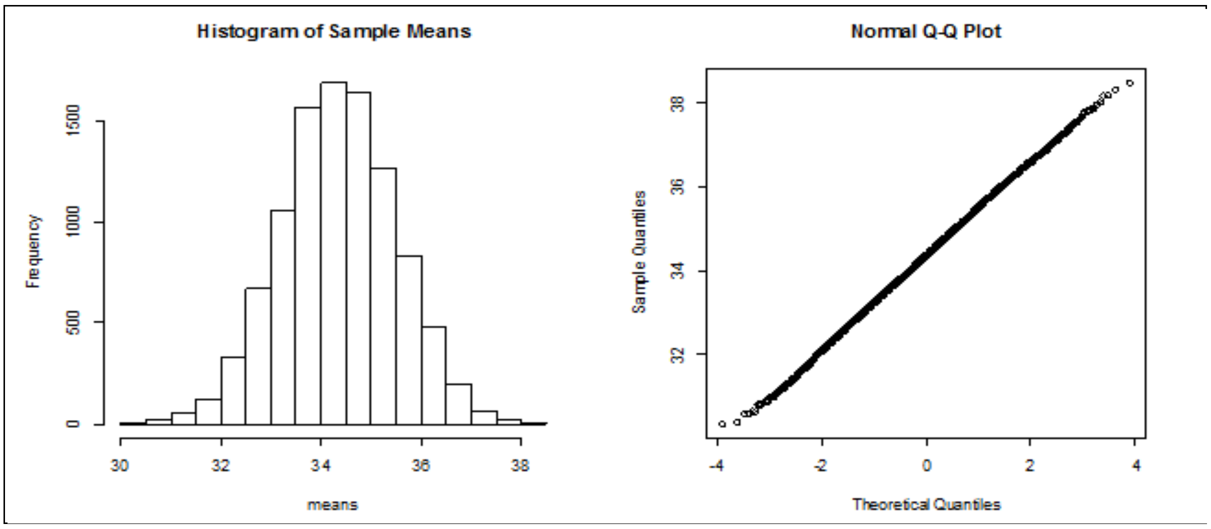


Figure 2.4: Histogram and Normal Probability plot of 10,000 Sample Means

The end points of the 95% confidence interval are estimates of the 2.5th and 97.5th percentiles of the distribution of $\bar{X}$. The 95% confidence interval limits can be obtained using the *quantile* function in the following R code segment. The *quantile* function computes the sample percentiles for the given probabilities. For a 95% confidence level, we identify the values at the 2.5th and 97.5th percentiles of the *means* vector (these values could be adjusted to obtain confidence interval limits for different confidence levels).

```
> limits <- c(quantile(means,0.025), quantile(means, 0.975))
> limits
 2.5%   97.5%
32.12   36.56
```

To get a better understanding of the meaning of confidence intervals, we can compute a confidence interval for each sample and find the proportion of intervals containing the true mean. This proportion should be the confidence level (95% in this case) if the process is repeated a large number of times. For a 95% confidence level, the confidence interval formula is $\bar{X} \pm 1.96\,\sigma/\sqrt{n}$. The following R code simulates this process.

```
> lLim <-  c()
> uLim <- c()
> for(i in 1:10000)
{
+   x <- sample(Age, 50, replace = FALSE)
+   lLim[i] <- mean(x) - 1.96*sd(Age)/sqrt(50)
+   uLim[i] <- mean(x) + 1.96*sd(Age)/sqrt(50)
}
> count <- 0
> for(i in 1:10000)
{
+    if(lLim[i] < mean(Age) & uLim[i] > mean(Age))
+        count <- count + 1
}
> count/10000
[1] 0.9602
```

In the first *for* loop of the above code segment, 10,000 samples are drawn and confidence intervals are computed. The second *for* loop counts the number of intervals containing the actual mean. In this simulation run, this proportion is 0.9602 (96%). This indicates that in long run, approximately 95% (assumed confidence level) of the intervals will contain the population mean. This process will give students a better understanding of the confidence intervals and the confidence level.

2.4.2 Hypothesis Testing

Hypothesis testing is another key topic of statistical inferences although it is one of the more difficult concepts to understand. A sound knowledge of terms such as null and alternative hypotheses, significance level, and *p*-value is essential in performing a hypothesis test and making the correct decision. Since all statistical software calculate *p*-values, now more and more instructors are using the *p*-value approach to make decisions. Many students do not have a good understanding about the *p*-value and they blindly use the rejection criterion that "if the *p*-value is less than the significance level, rejects the null hypothesis". The definition of the *p*-value is the probability, assuming the null hypothesis is true, that the test statistics would take a value as extreme or more extreme than that actually observed [7]. We demonstrate the use of R in hypothesis testing using following example.

One sample t test. Data for this example are reading times [7].
Does using fancy fonts slow down the speed of reading text from a computer screen? Adults can read four paragraphs of text in an average time of 22 seconds in the common Times New Roman font. 25 adults were asked to read this text in the ornate font named Gigi. Here are their times:
23.2, 21.2, 28.9, 27.7, 23.4, 27.3, 16.1, 22.6, 25.6, 32.6, 23.9, 26.8, 18.9, 27.8, 21.4, 30.7, 21.5, 30.6, 31.5, 24.6, 23.0, 28.6, 24.4, 28.1, 18.4.
Suppose that reading times are normally distributed. Is there good evidence that the mean reading time for Gigi fonts is greater than 22 seconds? In other words, is μ greater than 22 seconds for Gigi fonts?

The null and alternative hypotheses are: H_0: $\mu = 22$ seconds and H_1: $\mu > 22$ seconds. The test statistics used for this test is $(\bar{X} - \mu)/(s/\sqrt{n})$ and that follows the t-distribution with 24 degrees of freedom. *s* is the sample standard deviation. The value of the test statistics based on the above sample assuming the null hypothesis is 3.6742.

In R, *t.test* command performs the t-test and produces the test statistic and the *p*-value. The reading times are stored in a csv file under the column heading named 'Times'. Following are the results:

```
> t.test(Times, mu = 22, alternative = 'greater')
        One Sample t-test
```

```
data:  Times
t = 3.6724, df = 24, p-value = 0.0006001
alternative hypothesis: true mean is greater than 22
```

To perform a left tailed or two tailed test, 'greater' should be replaced with 'less' or 'two.sided'. Since the *p*-value (0.0006) is less than the significance level α (say 0.05), we have sufficient evidence to reject the null hypothesis and conclude that the mean reading time is greater than 22 seconds.

To get a better understanding of the *p*-value and to assess the evidence of this sample, we generate new samples that are consistent with the null hypothesis that the population mean is 22 [6]. The sample mean of the original sample is 25.152. To ensure that the null hypothesis (μ = 22) is satisfied, we subtract 3.152 from each reading time to produce a new set of times with a mean exactly equal to 22. To generate a randomization distribution of sample means while assuming that the null hypothesis is true, we select samples of size 25 at a time (with replacement) from the modified data and compute the mean of each sample. A set of sample means generated by this process will be a randomization distribution of values produced at random under the null hypothesis that μ = 22. The R code below simulates this process.

```
> NewTimes <- Times - 3.152
> means <- c()
> for(i in 1:10000)
{
+    x <- sample(NewTimes,25, replace = TRUE)
+    means[i] <- mean(x)
}
```

Figure 2.5 shows a dot plot of the sample means for 10,000 samples generated by this randomization process. As expected, the distribution is centered at the null value 22.

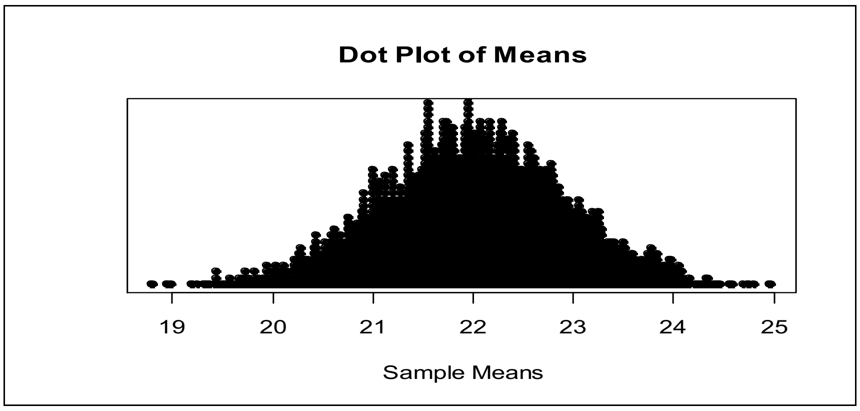


Figure 2.5: Dot Plot of 10,000 Sample Means

We compute the proportion of sample means that are as large (or larger) than the original sample mean 25.152. This proportion is equivalent to the *p*-value of the hypothesis test. The following code segment computes this proportion.

```
> count <-  0
> for(i in 1:10000)
 {
+   if(means[i] >= 25.152)
+      count <- count + 1
}
> count/10000
[1] 0
```

This *p*-value (0) gives strong evidence against the null hypothesis (as noted using *t.test* command) and supports the alternative hypothesis that the average reading time is greater than 22 seconds. While the *t.test* command allows students to perform the hypothesis test, the above described randomization process provides a better understanding of the *p*-value.

3. Final Remark

In this paper, we have presented ways of using R in several important topics in introductory statistics. Many students have difficulties in understanding the concepts, in particular statistical inferences. We believe students get a better feel about the concepts using examples in their lessons. Using R to perform simulation allows students to visualize the concepts in sampling distributions, confidence intervals, and hypothesis testing. These simulation methods can be adopted in a variety of introductory level classes.

References

[1] Chance, B., Ben-Zvi, D., Garfield, J., & Medina, E. (2007). The Role of Technology in Improving Student Learning of Statistics, *Technology Innovations in Statistics Education*, 1(1), https://escholarship.org/content/qt8sd2t4rr/qt8sd2t4rr.pdf.

[2] Chandrakantha, L. (2018). Simulating Confidence Intervals for Mean and Variance using Real Data in R programming Environment, *Proceedings of the 23rd Asian Technology Conference in Mathematics,* Yogyakarta, Indonesia, 186-199.

[3] Garfield, J. & Ahlgren, A. (1998). Difficulties in Learning Basic Concepts in Probability and Statistics: Implications for Research*, Journal for Research in Mathematics Education*, 19(1), 44-63.

[4] Garfield, J., delMas, R., & Chance, B. (1999). Tools for Teaching and Assessing Statistical Inferences. Retrieved from www.tc.umn.edu/~delma001/stat_tools/

[5] James, G, Witten, D, Hastie, T, & Tishirani, R. (2013). *An Introduction to Statistical Learning with Applications in R*, Springer, New York.

[6] Lock, R.H., Lock, P.F., Morgan, K.L., Lock, E.F. & Lock, D.F. (20173). *STATISTICS, Unlocking the Power of Data*, 2nd Edition, New York, USA: John Wiley & Sons.

[7] Moore, D. S. (1996). *Essential Statistics.* New York, USA: W. H. Freeman & Company.

[8] Verzani, J. (2008). Using R in Introductory Statistics Courses With the pmg Graphical User Interface, *Journal of Statistics Education*, 16(1), DOI:10.1080/10691898.2008.11889558