Understanding Statistics Concepts Using Simulation In R

Leslie Chandrakantha
John Jay College of Criminal Justice of CUNY
Mathematics and Computer Science Department
524 West 59th Street, New York, NY 10019
lchandra@jjay.cuny.edu

Abstract

Learning statistics concepts are important for college students. The ability to understand, interpret, and evaluate research findings are essential skills. Many different approaches have been used for providing statistics instructions and enhancing the conceptual understanding. Some concepts such as sampling distributions, confidence intervals, and hypothesis testing are difficult to teach and to understand. In this paper, we describe how to use simulation using R programming environment. R is free, and has flexible properties which make it suitable for introductory statistics students. The paper introduces R functions relevant to each topic.

1. Introduction

Students in introductory statistics classes often struggle to understand the fundaments concepts such as sampling distributions, central limit theorem, confidence intervals, and hypothesis testing. Instead of understanding fundamental concepts and applying statistical procedures properly, many students focus on memorizing methods of performing calculations using calculators or software. This approach does not provide good foundation for their future courses, conducting research, analyzing data, and making correct conclusions. The traditional way of teaching using book and lecture based instruction does not give a good understanding of the concepts to many students. Advances in technology have enabled instructors to experiment with different teaching methods. Simulation with the help of computers can be a very effective tool in getting a good grasp of these concepts. One of the most challenging aspects to teaching and learning statistics is that many statistical concepts are based on the issue of what would happen if a random process such as random sampling from a population were repeated a large number of times. This abstract notion is very difficult for most students to grasp. Technology provides the opportunity to make this abstract idea more concrete by enabling students to repeat such random processes a very large number of times and describe their observations first hand. We can use simulation using computers to perform these types of experiments.

The American Statistical Association has published Guidelines for Assessment and Instruction in Statistics Education (GAISE) [7] in order to improve student learning. These guidelines recommend active learning of concepts approach in teaching and learning statistics. Simulations performed both manually and using computers are recommended in the GAISE report as a useful tool for enhancing student learning. In recent years, there has been much interest in the use of simulation in teaching fundamental statistical concepts. Mills [9] has given a comprehensive

review of literature of computer simulation methods used in all areas of statistics to help students understand difficult concepts. Cobb [5] noted that incorporating computer simulation techniques to illustrate the key concepts and to allow students to discover important principles themselves enhances their knowledge. For more examples of the use of simulation to teach statistics see [1], [2], [3], [4], and [8].

R is increasingly being used as a tool for statistics education. Many introductory and higher level statistics instructors are now using R to teach and perform statistical calculation, even though it is bit challenging to write statements in the command line. R can be used in simulation effectively. R can easily generate random samples from variety of probability distributions. A valuable introduction to R for introductory statistics is given in Dalgaard [6]. In this paper, we describe how to use R commands to generate different random samples from populations, compute the sampling distribution of the mean to understand its properties and the central limit theorem, and compute confidence intervals to understand the real meaning and its interpretation. In coming sections, we give an overview of R, simulation of sampling distributions, simulation of confidence intervals, and concluding remarks.

2. Brief Overview of R and Some R Commands

R is a free software environment for working with data. R can be used to create sophisticated graphs, carry out statistical analyses, and run simulations. R is also a programming language with set of built-in-functions, so with some knowledge, students can write their own codes for statistical computations. For computationally intensive tasks, one can incorporate functions written in others languages such as C, C++, and FORTRAN. R compiles and runs on wide variety of UNIX platforms, Windows and MacOS. As of writing this paper, the most recent version of R was R 3.2.3. R is available from http://www.r-project.org. To install R 3.2.3 on your operating system, download R from above site using the closest mirror site to your location and choose the appropriate link for your operating system.

R is a relatively simple syntax driven, a case sensitive language. Even though the syntax for writing instructions may be somewhat difficult initially, most students with little or no prior programming experience have become comfortable using R. R is an object oriented program that works with data structures such as vectors (one dimensional array) and data frames (two dimensional arrays). A vector contains a list of values. When R is started, we will see a window that is called R console. This is where we type our commands and see the text results. Graphics appear in a separate window. R console is similar to the window in *Figure 1*:

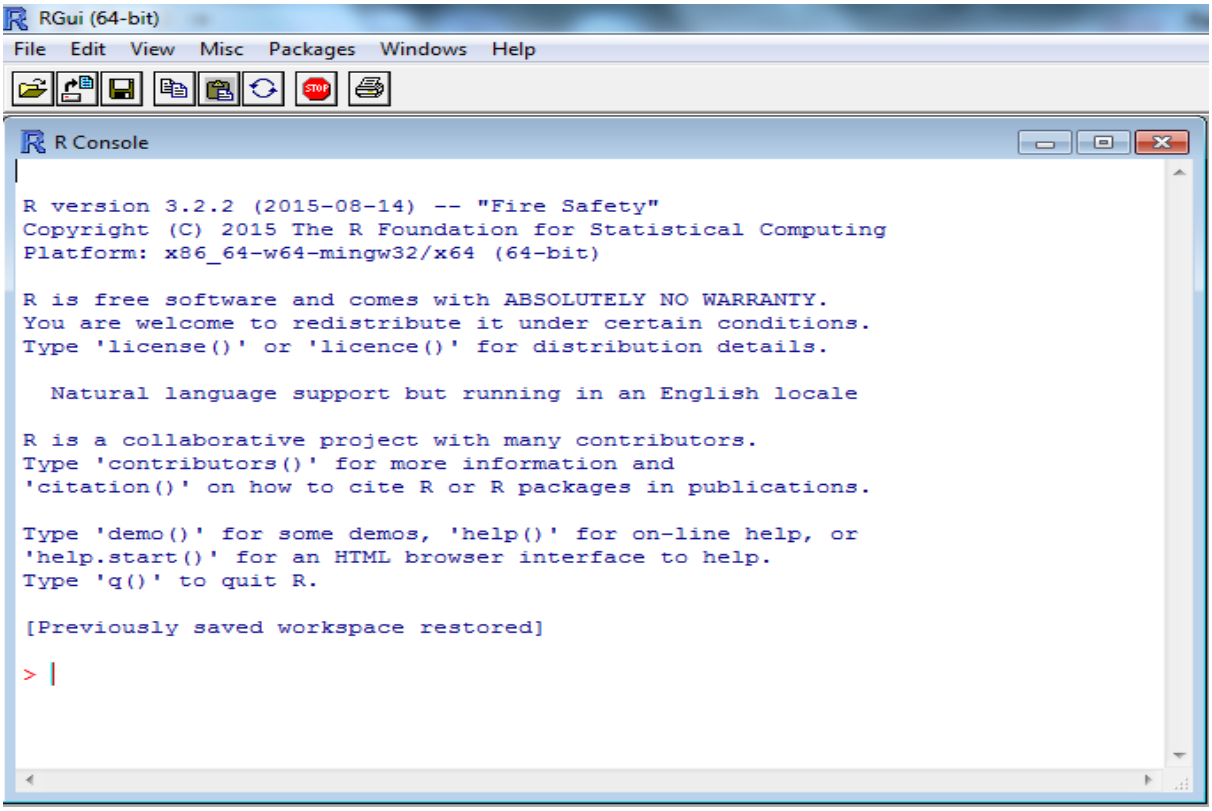

Figure 1: R Console

The > is called the prompt. We type R commands at the prompt. To quit R we type > q( ).

R as a calculator:
At the prompt we enter the mathematical expression and hit enter key and it will calculate the result and display it. Following is an example:

```
> 2 + 3
[1] 5
> 2*3 - 10
[1] -4
> 3^5
[1] 243
```

The standard arithmetic operators +, -, *, and / are used in expressions and ^ is used for exponentiation. These operations have the standard order of precedence with exponentiation highest and addition/subtraction lowest but we can control order of precedence using parentheses.

The results of a calculation can be assigned to a variable (object in R) using <- or =. In this paper, we will use <-. The following code segment demonstrates an evaluation of a mathematical expression:

```
>  x <- 2*3 + 5
> x
[1] 11
```

Vectors and functions

Even though we can work with single numbers (scalars), R is primarily designed to work with vectors and functions. In R, a vector is a sequence of data values of the same type. The function c is used to create vectors from scalars. Following statement creates a vector:

```
> 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 some useful summaries:

To obtain the sum of the values of the vector:

```
> sum(x)
[1] 30
```

To obtain the number of values of the vector:

```
> length(x)
[1] 5
```

The following functions can be used to calculate basic summary statistics:

Mean: mean(x)
Median: median(x)
Maximum: max(x)
Minimum: min(x)
Standard deviation: sd(x)
Variance: var(x)
Five number summary: summary(x)

The following functions can be used for visual data displays:

Stem-and-leaf diagram: stem(x)
Histogram: hist(x)
Box plot: boxplot(x)
Scatter Plot: plot(x, y)

The following functions can be used to generate random numbers:

To generate a random sample of size n from a vector of values x with replacement:
sample(x, n, replace = TRUE)
To generate a random sample of size n from a vector of values x without replacement:
sample(x, n, replace = FALSE)

If x is a single number in above two cases, it generates a sample from 1 to x.

The following functions generate random samples of size n from indicated probability distributions:

runif(n, a, b) – from uniform distribution from a to b.
rbinom(n, m, p) – from binomial distribution with parameters m and p.
rpois(n, λ) - from Poisson distribution with parameter λ.
rnorm(n, μ, σ) – from normal distribution with mean μ and standard deviation σ.
rt(n, df) – from t distribution with degrees of freedom df.
rchisq(n, df) – from Chi-square distribution with degrees of freedom df.

R has the standard control structures such as if–else, while, and for statements. We will introduce these statements when we discuss simulation in coming sections.

3. Simulating Sampling Distribution of Mean

The gateway to statistical inferences is the sampling distributions. It is essential to gain a good understanding about the concepts of sampling distributions for students in statistics classes. At the beginning of the class, we give the definition of the sampling distribution of the mean, introduce the properties, and explain the connection between sampling distributions and the central limit theorem. The sampling distribution of the mean is the probability distribution of sample mean based on all possible simple random samples of the same size from the same population. The sampling distribution of the mean has the following properties:

- The mean of all sample means is equal to the population mean.
- 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.
- 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. Based on this theorem, we can use the normal distribution for inferences about the mean for larger sample sizes, even if the original population is not normally distributed. Many students are using this fact without understanding the underlying concept. Simulation allows students to visualize this fact.

Now we demonstrate the simulation of the sampling distribution using R. The students have computers in the classroom so they follow our instructions and generate their own. All the computers in classrooms have R software. We consider different population distributions and different sample sizes to observe the effects of the sample size and the shape of the original distribution on the sampling distribution of the mean. The uniform, chi-square, and normal populations and sample sizes of 10, 25, and 50 are considered. These three populations have uniform, skewed, and bell shapes so each student visualizes the fact that as the sample size increases the sampling distribution approximates a normal distribution for different original shapes. This is one of the main points that we want to convince our students as a result of this lesson. Students were taught these three populations, their shapes, and parameters in previous lessons so we just briefly explain them to refresh their memory and introduce the R functions to

generate random variants form these three populations. *Table 1* gives the characteristics of the samples and the R functions. The value of n in R functions is the sample size.

Distribution	Sample Sizes	Mean	Std Deviation	R Function
Uniform (0, 100)	10, 25, 50	50	28.87	= runif(n, 0, 100)
Normal (100, 10)	10, 25, 50	100	10	= rnorm(n, 100, 10)
Chi-square (2)	10, 25, 50	2	2	= rchisq(n, 2)

Table 1: Sample characteristics

Now we show R codes for generating random samples of 10 uniform random numbers and computing the sampling distribution of the mean. The R function *dunif* is used to calculate the uniform density. First we show the probability density function (pdf) graph of above uniform distribution using following R code:

```
> x  <- seq(from = 0, to = 100, by = 0.1)
> dens <- dunif(x,0,100)
> plot(x, dens, type = "l")
```

Using this code segment, we generate a sequence of numbers from 0 to 100, incrementing by 0.1 and store them in vector x. The *dens* vector has the height of probability density function (pdf) at each x value. The plot command draws the graph of the uniform density shown in *Figure 2*:

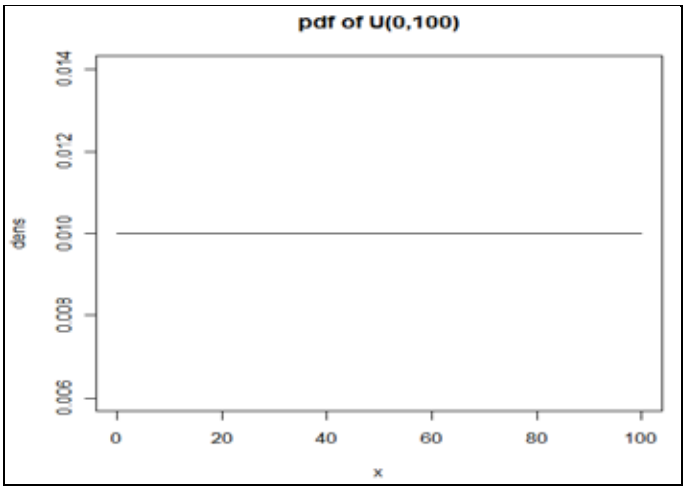

Figure 2

The following code segment generates 1000 random samples of size 10 from above uniform distribution and computes the sample means. The R function *runif* is used to generate random sample from the uniform distribution. The syntax of the function is *runif(n, a, b).* This function returns a random sample of size n from the uniform distribution from *a* to *b*.

```
> means <- c()
> for(i in 1:1000){
+ y <- runif(10, 0, 100)
+ means[i]  <- mean(y)}
> mean(means)
[1] 49.75268
```

```
> sd(means)
[1] 9.164981
> hist(means, main = "U(0,100), n = 10")
```

The *for* loop iterates 1000 times to generate 1000 random samples from uniform distribution and calculates sample means. The *means* vector holds these sample means. The *runif(10,0,100)* function generates a random sample of size 10 from uniform distribution from 0 to 100. This random sample is stored in vector y for the *i*th iteration of the *for* loop. The *means[i]* variable stores the corresponding sample mean for each sample for the *i*th iteration. The *mean* and *sd* commands compute the mean and standard deviation of 1000 sample means. The 1000 sample means are considered as the sampling distribution of the mean to verify the validity of the properties the mean and standard deviation (standard error) of the sampling distribution. To study the shape of the sampling distribution, we create a histogram of the 1000 sample means using *hist* command. Similarly, we generate the sampling distributions and histograms for all the cases we have considered in this lesson. *Figure 3* shows histograms for all the cases. In *Figure 3,* first, second and third column histograms are created from sample means from uniform, normal, and chi-square distributions respectively. The first, second and third rows represent samples sizes 10, 25 and 50 respectively. These histograms allow students to understand the meaning of the central limit theorem. 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. At the end of the lesson, students are able to comprehend the central limit theorem and understand the properties of the sampling distribution discussed in the class.

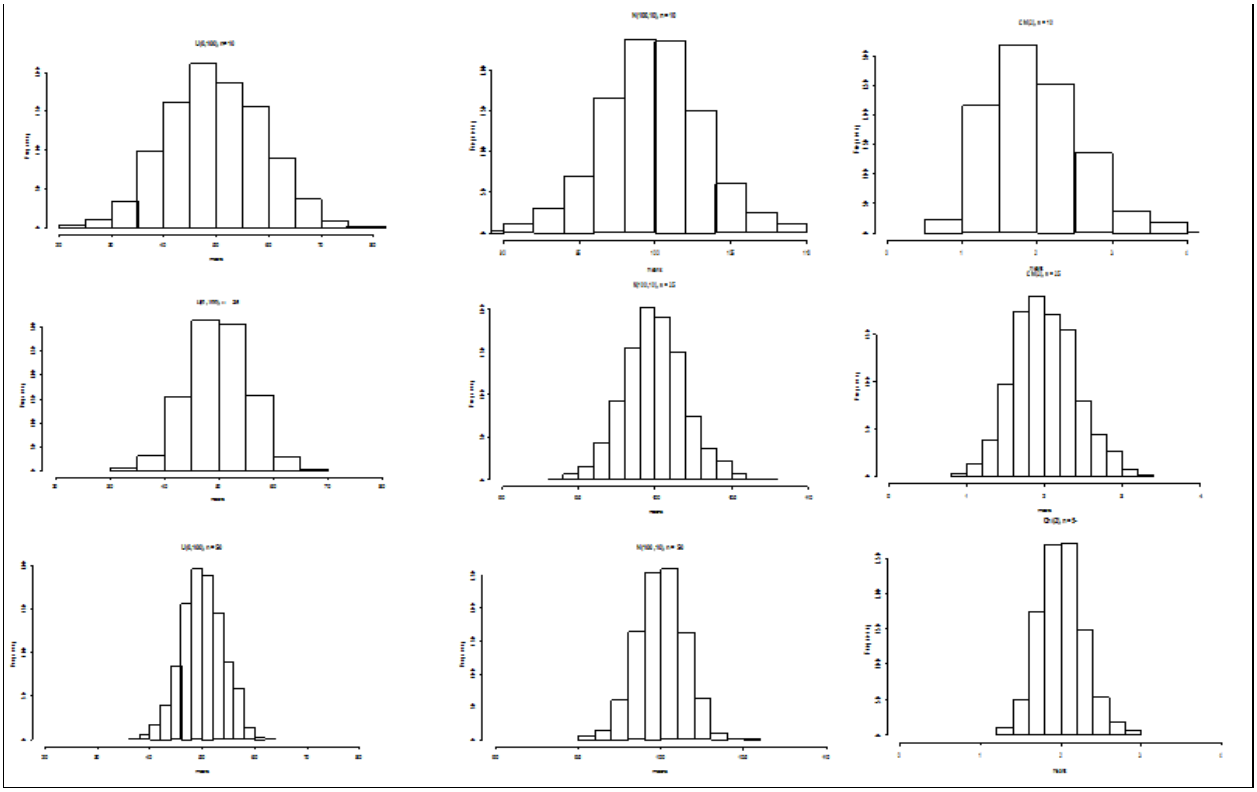

Figure 3: Histograms for Sample Means

4. Simulating Confidence Intervals

The topic of confidence intervals of parameter estimation is a difficult lesson to teach in statistics classes, particularly in introductory level. Confidence intervals give the most likely range of the unknown population parameter. In this discussion, we only consider the creating and interpreting confidence interval for mean (μ) assuming population standard deviation (σ) is known. Beginning of the class, we introduce the material that is needed and define the confidence interval for the mean as $\bar{X} \pm Z^* \sigma / \sqrt{n}$, where Z^* is the value of the standard normal curve with area C between critical points $-Z^*$ and Z^* and n is the sample size. *The confidence level C 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*, Moore [10]. Students have major difficulties in understanding this last statement. Many students misunderstand this statement as saying that the majority of individual values are in this interval. It is important in this lesson that students understand in repeated sampling from a population, C percent of intervals (say 95%) would capture the true unknown mean. In using the traditional way of teaching, we only consider one sample and calculate one interval. This leads them to believe the wrong interpretation of the interval that there is a 95% chance that this interval will have the true mean.

A computer simulation method using R will allow students to understand the true meaning of the confidence interval. After introducing the basic facts about confidence intervals to the class, we will calculate 95% confidence intervals for the population mean. The R function *rnorm* is used to generate a random sample from the normal distribution. The syntax of the function is *rnorm(n, μ, σ)*. This function returns a random sample of size n from the normal distribution with mean μ and standard deviation σ.

Now we show how to compute confidence intervals for different samples using R. For 95% confidence level, critical values –Z* and Z* are -1.96 and 1.96 respectively. The R code segment shown after this paragraph generates 100 random samples of size 30 from the normal distribution with mean 100 and standard deviation 10. The vectors *means*, *lower* and *upper* hold the sample means, lower and upper confidence interval values respectively. The parameters *n*, *mu* and *sigma* hold the sample size, mean, and standard deviation of the normal distribution. The *for* statement iterates 100 times to generate 100 random samples and calculate sample means and confidence intervals. The *rnorm(n, mu, sigma)* function with n = 30 generates a random sample of 30 values from the normal distribution with mean 100 and standard deviation 10. The *means[i]* variable stores the corresponding value of the sample mean for the *i*th iteration. Similarly, *lower[i]* and *upper[i]* variables store the lower and upper bounds of the confidence interval for the *i*th iteration. The second *for* statement count the number of confidence intervals containing the actual population mean *mu*. The variable *p* computes the proportion of intervals that contains the mean *mu*. This proportion should be close to 95% which is the assumed confidence level *C*. We noticed that in our simulation, 96% of the confidence intervals do contain the true mean. If we generate another set of samples and compute the confidence intervals, we would find the new

proportion to be 95% or close to it. Since students are doing these steps themselves, they get a clear understanding of the meaning of confidence intervals.

```
> means <- c()
> lower <- c()
> upper <- c()
> n <- 30; mu <- 100; sigma <- 10     # sample size and parameters
> for(i in 1:100) {
+ x <- rnorm(n,mu,sigma)      # generate a random sample
+ means[i] <- mean(x)          # calculate mean for each sample
+ lower[i] <- means[i] - 1.96*sigma/sqrt(n)         # calculate lower bound of interval
+ upper[i] <- means[i] + 1.96*sigma/sqrt(n)   }     # calculate upper bound of interval
> count  <- 0              # count will hold number of intervals do contain the mean mu
> for(i in 1:100) {
+ if(mu >= lower[i] && mu <= upper[i])
+ count  <- count + 1  }
> p  <- count/100     # proportion of intervals do contain the mean mu
> p
[1] 0.96
```

5. Conclusion

Many students have difficulties understanding statistics concepts such as sampling distributions and confidence intervals. Simulations can be effective learning tools for helping students to understand abstract concepts associated with repeated random processes. We have demonstrated the use of simulation using R in teaching these topics. This is a very useful way to visualize and understand the sampling distribution, central limit theorem, and confidence intervals. These simulation methods are acceptable to students with varying backgrounds of mathematics. More empirical studies need to be conducted to measure the effectiveness of using simulations as a pedagogical tool.

References

[1] Barr, Graham D. and Scott, Leanne. (2011). Teaching Statistics in a Spreadsheet Environment using Simulation. *Spreadsheetsin Education* (eJSiE), 4(3). http://epublications.bond.edu.edu.au/ejsie/vol4/iss3/2.

[2] Butler, A., Rothery, P., & Roy, D. (2003). Minitab Macros for Resampling Methods. *Teaching Statistics*, 25 (1), 22-25.

[3] Chandrakantha, Leslie. (2014), Visualizing and Understanding Confidence Intervals and Hypothesis Testing Using Excel Simulation. *The Electronic Journal of Mathematics and Technology* (EJMT), 8(3): p 212-221

[4] Christie, D. (2004). Resampling with Excel. *Teaching Statistics*, 26 (1), 9-14.

[5] Cobb, P. (1994). Where is the Mind? Constructivist and Sociocultural Perspectives on Mathematical Development. *Educational Researcher*, 23, 13-20.

[6] Dalgaard, P., *Introductory Statistics with R*, 2nd Edition, New York, NY: Springer, 2008.

[7] GAISE (2005). Guidelines for Assessment and Instruction in Statistics Education Report. American Statistical Association, Alexandria, VA. http://www.amstat.org/education/gaise/

[8] Hagtvedt, R., Jones, G. T., & Jones, K. (2008). Teaching Confidence Intervals Using Simulation. *Teaching Statistics*, 30 (2), 53-56.

[9] Mills, J. D. (2002). Using Computer Simulation Methods to Teach Statistics: A Review of the Literature. *Journal of Statistics Education (Online),* 10 (1). http://www.amstat.org/publications/jse/v10n1/mills.html

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