IEJME
OPEN ACCESS

INTERNATIONAL ELECTRONIC JOURNAL OF MATHEMATICS EDUCATION
e-ISSN: 1306-3030. 2019, Vol. 14, No. 3, 599-610
https://doi.org/10.29333/iejme/5773

Using R as a Simulation Tool in Teaching Introductory Statistics

Xuemao Zhang [1*], Zoe Maas [1]

[1] *East Stroudsburg University, USA*

* CORRESPONDENCE: xzhang2@esu.edu

ABSTRACT

The use of computer simulations in the teaching of introductory statistics can help undergraduate students understand difficult or abstract statistics concepts. The free software environment R is a good candidate for computer simulations since it allows users to add additional functionality by defining new functions. In this paper, we illustrate how computer simulations with R are used in statistics classrooms and student homework assignments by examples. These examples include sampling distributions and the central limit theorem, the *t*-distributions, confidence intervals, hypothesis testing, regression analysis and nonparametric tests.

Keywords: computer simulations, introductory statistics, R software, teaching statistics

INTRODUCTION

Statistical analysis of data is increasingly common in the fields of education, economics, biology and genetics, geology, healthcare, social psychology, sports and more due to our increasing data collection abilities (Best & Kahn, 2006; Haining, 2003; Koh & Tan, 2005; Ray, 2012; Quackenbush, 2001). Therefore, there is a growing movement to include statistics in all levels of education. To better prepare students for careers, especially in higher education, almost all majors require students to take at least one statistics course.

Statistics is not simply a collection of methods and techniques. The primary goal of statistical training at all levels should be to help students develop statistical thinking. According to Aoyama (2007), most statistics students, ranging from middle school to graduate school, have a functional ability to read graphs of data, but fewer engage in critical statistical thinking about the data. Brown and Kass (2009) states that statistical thinking uses probabilistic descriptions of variability in inductive reasoning and analysis of procedures for data collection, prediction, and scientific inference. In statistical inferences, uncertainty is involved when people draw a conclusion about a population from a smaller random sample selected from the population (Innabi, 2007). Computer simulation is a good tool for students to understand all statistical concepts related to uncertainty. For example, Arnholt (1999) describes a procedure to use simulation and graphical output as education tools to help students gain a deeper understanding of sampling distributions.

The tool of computer simulations is an essential part of understanding statistics. The nature of many statistical concepts is difficult for students to grasp without visual representations or a hands-on exercise (Alias, 2009; Ridgway, Nicholson, & McCusker, 2007). Many basic aspects of introductory statistics involve hypothetical ideas, and thus they can only be properly observed through the use of computer simulations (Sigal & Chalmers, 2016). When proper visualization and data manipulation is available, students as young as 12 could develop at least a functional understanding of multivariate data (Ridgway, Nicholson, & McCusker, 2007). For example, a study conducted by Raffle and Brooks (2005) involved the use of the software called MC4G—Monte Carlo Analysis for up to 4 Groups—to help students understand various types of data tests, including t and ANOVA F tests. Not only did the use of the MC4G improve students' exam scores, the students

Article History: Received 25 February 2019 ◆ Revised 25 February 2019 ◆ Accepted 30 April 2019

also responded positively to a survey asking if they felt that the software improved their understanding of statistical concepts like the ANOVA test and type I error (Raffle and Brooks, 2005). Similarly positive findings were reported in a more recent study by Mendoza and Mendoza (2018) on computer software for geometry and algebra.

Mills (2002) literature review revealed abundant evidence that computer simulation of statistical concepts is a useful teaching method. It recounted articles reporting that computer simulations can be used to help teach students about the central limit theorem, *t*-distribution, confidence intervals, binomial distributions, regression analysis, sampling distributions, and hypothesis testing. While the literature review was using studies of many different types of computer simulation software, the statistical software R is capable of being a learning tool that can help students understand all the statistical concepts that the review covered.

Smith (2003) outlines some of the benefits and drawbacks of three types of simulation software: web-based programs, standalone statistical software, and simulations within pre-existing software. First, web-based programs are often free and can be used on any type of computer with internet access. They do not require strong computer skills, but are often just simple graphics depicting statistical phenomenon, and may not fully illustrate statistical theory without supplemental explanation from the instructor (Smith, 2003). Statistical software R is more powerful than these simple programs, and can be used to run simulations, create different types of graphs, and calculate various statistics. Second, standalone software can be downloaded and often require the students to have a more hands-on learning experience with simulations. This type of software, like SPSS and SAS, often has expensive license which makes them hard to access for students that cannot pay for them (Smith, 2003). R is similar to these packages, however, it is free to use. Moreover, R is also available for all three popular operating systems - Windows, Mac, and Linux. Last, simulations within pre-existing software can be found in some software packages such as Microsoft Excel. Microsoft Excel can illustrate some statistical concepts and is easy to use for students with weak computer skills, but lacks some essentials that full statistical software possesses (Chandrakantha, 2014). For example, Excel cannot properly handle missing data or create some statistical simulation illustrations, like boxplots (Chandrakantha, 2014). R, on the other hand, is fully equipped to simulate or utilize any statistical concept.

R can be learned easily by students that have little programming experience. At first glance, it is obvious that Software R is not as simple as Microsoft Excel, and does involve coding that can appear daunting to students lacking in strong computer skills. However, R also has abundant online resources that can aid student learning. R allows students to become familiar with computer programming as well as statistics, and may benefit students who also want to develop their computer skills for their career.

In this paper, a brief introduction to R is covered in Section 2. In Section 3, we use several typical examples to illustrate how easy to implement computer simulations using R. These examples include sampling distributions and the Central Limit Theorem, *t*-distributions, confidence intervals, power of a hypothesis test, simple linear regression models, and comparison of the sign test and the *t*-test. And a discussion follows in Section 4.

INTRODUCTION TO R

R is a free software that is used to compute and graph statistics. It was developed by Robert Gentleman and Ross Ihaka. R allows users to practice many different types of statistical techniques, like linear regression, ANOVA, classic statistical tests, and much more. It also has thousands of downloadable packages that are meant for specific methods and types of statistics. It can be downloaded at http://www.r-project.org/ or one of its numerous mirror websites (R Development Core Team, 2017).

R language is object-orientated. Objects may have attributes, such as name, dimension, and class. This means that there are pre-set data types, data structures, and functions that students can use to manipulate their own data (R Development Core Team, 2017). The tool of computer simulations allows students to focus more on the theory and concepts behind statistics rather than just focusing on memorizing formulas. Students then can use R's pre-set functions to create their own Monte Carlo simulations in order to grasp statistical concepts. Sigal and Chalmers (2016) overviewed the essential aspects for writing Monte Carlo simulation code in R in detail. It also advertises the use of one of R's many useful packages that make creating Monte Carlo simulations easier.

R's basic data types are character, numeric, integer, complex, and logical. Variables are assigned with R objects and the main R objects or data structures include vector, list, matrix, array, data frame, and factor (R Development Core Team, 2017). The data structure vector is the only R object to be used in this paper.

The vector object is one of the most common and basic data structures in R. A vector often contains one or more of four types of data: characters, logical, integers and numeric. Numeric vectors, the most common form of vector, use numbers. Vectors are incredibly useful for analyzing univariate datasets. Once a univariate data set is put into a vector, it can be analyzed in many ways—from simply finding the five number summary, plots to conducting hypothesis tests and constructing confidence intervals.

A list is an R object which can contain multiple data structures, functions and even other lists. Lists are useful for creating functions since they allow us to gather a variety of objects under one name. Specifically, they are used so that all the information a function returns can come back in the simple form of a single list, rather than a meaningless jumble of numbers.

A matrix is a two-dimensional rectangular data set. All elements in a matrix must have the same data type. Matrices can be created using a vector input to the matrix function. This data structure in linear algebra is used for representing linear transformations.

Factors are used to store labels or nominal data as a vector. The labels are always character irrespective of whether it is numeric or character or Boolean etc. in the input vector. For instance, a factor may look like, x=factor(c("green", "yellow", "red", "red"))". By using character vectors instead of numbers or labelled integers, factors are already self-described, so it's easy to understand what the code means when looking at it.

Data frames are similar to matrices, but different columns can have different data types. A data frame is created using the data.frame() function or a data set is a data frame by default after it is imported to R. Arrays are similar to matrices but can have more than two dimensions. For more information about data types and data structures in R, see Software Carpentry (2017).

Loops and conditional statements in R allows us to create our own simulations. This is how Monte Carlo simulations, which use known parameters to generate random samples of data that can then be used to analyze the behavior of certain statistics, are created (Sigal & Chalmers, 2016). R loops include For loop, While loop and Repeat loop. Moreover, a conditional statement allows users to perform different computations or actions depending on whether a predefined boolean condition is TRUE or FALSE. R conditional statements include if(), the combination if() and esle(), ifelse(), and switch().

R is a software environment in which many classical and modern statistical techniques have been implemented. A few of these are built into the base R environment, but many are supplied as packages. Currently, the CRAN package repository features 13767 available packages as of a day in February 2019 (https://cran.r-project.org/web/packages/). The R packages increase the power of R by improving existing base R functionalities, or by adding new ones.

EXAMPLES OF SIMULATION STUDIES USING R

In this section, we use several typical examples to illustrate how to conduct computer simulations using R. These examples include sampling distributions and the Central Limit Theorem, *t*-distributions, confidence intervals, power of a hypothesis test, simple linear regression models, and comparison of the sign test and the *t*-test.

Sampling Distributions And The Central Limit Theorem

The point estimator of a population parameter is a random variable in the context of sampling and thus has a distribution. The concept of sampling distribution is critical for statistical thinking. For example, the sampling distribution of the sample mean in the general statistics textbook is defined as "the distribution of all possible sample means, with all samples having the same sample size taken from the same population". Students in daily life generally observe one sample data set only from an actual or conceptual population. Simulations can help students understand sampling distributions much better.

R can be used to demonstrate sampling distributions using simulations and visual representations, like histograms. For example, to study the sampling distribution of the sample mean, instructors and students can conduct a simulation study in which the means of thousands of samples with the same size are calculated. Furthermore, the effect of the sample size on the sampling distribution can be visualized by different choices

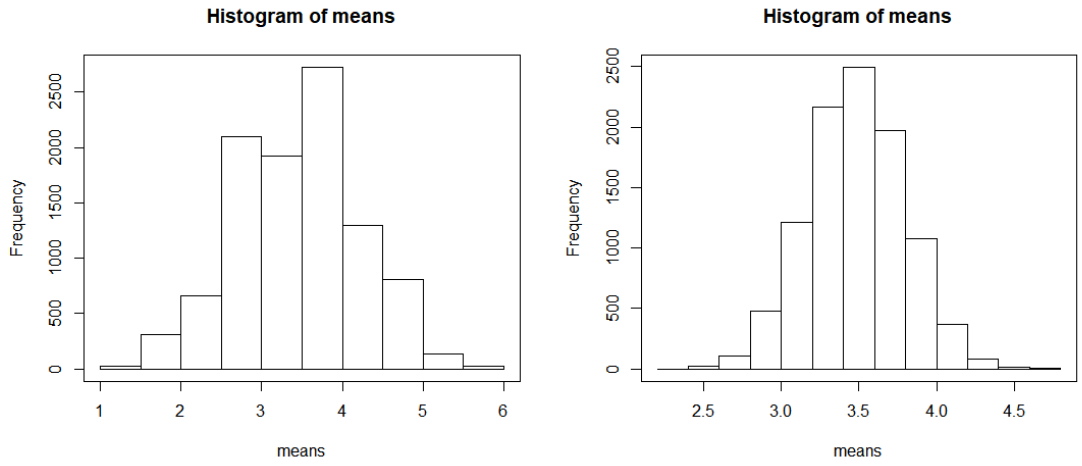

Figure 1. Histograms of 10,000 sample means where each sample is resulted from rolling a balanced six-sided die n times. The sample size n=5 for the histogram to the left and n=30 for the histogram to the right

of sample sizes. These types of simulations tend to be used to help students understand sampling distributions and the Central Limit Theorem (Mills, 2002).

The following R code is the study of the sampling distribution of the sample mean when a balanced six-sided die being rolled five times. That is, the observations can be regarded as a sample of size 5 taken from an infinite population resulted from rolling the die an infinite amount of times. The simulation study uses 10,000 samples.

```
iter = 10000;
n=5; #sample size
means=numeric(); #vector of sample means
variances=numeric(); #vector of sample variances
for (i in 1:iter)
{
x=sample(1:6, n, replace=T);
means[i]= mean(x);;
variances[i]= var(x);
}
mean(means); mean(variances);
hist(means); box();
```

In this example, the data values in each sample can be any numbers from 1-6. For each simulated sample, the sample mean is calculated and then stored in the vector of means of length 10,000. And finally, the average of the 10,000 sample means can be calculated and the histogram can be produced as well. It can be seen from **Figure 1** that the histogram is unimodal but not that symmetric if n=5. When the sample size becomes larger, the resulted histogram becomes almost symmetric. For example, students can change the value of n in the R code to 30. We can draw the conclusion that the larger the sample size is, the more normal the distribution of the sample mean will become. Furthermore, the average of the 10,000 of the sample variances stored in the vector variances in the R code can be calculated. By comparison, it can be easily seen that the variance of the sample mean becomes smaller as the sample size increases.

The Central Limit Theorem, the most important statistical theory in introductory statistics, states that if the sample size n is large (n >30 in most statistics textbooks) and the variance of the population in which samples are selected from is finite, the average of all sample means of the same size n from the same population will be equal to the mean of the population. Furthermore, all the sample means will follow an approximate normal distribution, with variance being equal to the variance of the population divided by the sample size. By the above simulation study, students can understand the Central Limit Theorem intuitively. The R

simulations let students investigate and find out how the sample size affects the distribution pattern of the sample means.

The T-distributions

The *t*-distributions are generally used in small sample statistical inferences given that the samples are selected from normal distributions with unknown standard deviations. For example, in 1-sample confidence interval estimation or hypothesis testing problems,

$$t = \frac{\bar{X} - \mu}{S/\sqrt{n}},$$

where $\bar{X}$ is the sample mean, S is the sample standard deviation and μ is the population mean, is said to have a *t*-distribution with degrees of freedom *n*-1. The properties of the *t*-distribution are generally compared with the standard normal distribution. The *t*-distribution looks very similar to the standard normal distribution, but it is slightly more spread out than the normal distribution. It is closer to the standard normal distribution as the sample size increases. Although the *t*-distribution often is taught through traditional methods, supplementary lessons in R can help students better grasp some concepts (Mills, 2002).

Running R simulations can better help students understand how the elements in the student *t*'s formula - particularly sample mean, standard error, and sample size - interact. For example, one might consider 10,000 samples of same size *n* from a normal population with mean of 1 and standard deviation of 1.5 and a density curve of the 10,000 *t* statistics can be obtained using the density function in R. Furthermore, the mean and standard deviation of these 10,000 statistics can be compared with 0 and 1, respectively. In the following simulation study, *n*=10 and *n*=30 are used to generate two sets of samples.

```
n = 10; #n=30;
iter=10000; t=numeric();
for (i in 1:iter)
{
y = rnorm(n, mean=1, sd=1.5); #Generation of random samples
ybar = mean(y); #Sample mean
s = sd(y); #Sample standard deviation
t[i] = (ybar-1)/(s/sqrt(n)); #t formula
}
x=seq(from=-6,to=6,by=0.1);
plot(x, exp(-x^2/2)/sqrt(2*pi), type= 'l', col='red', lwd=2, main='Z-density curve and the density curve of t, n=10',ylab='density curve');
lines(density(t), col='blue', lwd=2);
mean(t); sd(t);
```

In each of the following two graphs in **Figure 2**, the red curve is the standard normal density curve and the blue curve is the density curve of the 10,000 *t* statistics. It can be seen that both *t*-density curves are very close to the standard normal density curve. As the sample size *n* is increased from 10 to 30, the *t*-density curve is closer to standard normal. The conclusion can be verified by the means and standard deviations of the two sets of the data. Both mean values are very close to zero. But the standard deviations for sample size *n*=10 and *n*=30 were 1.136 and 1.037, respectively.

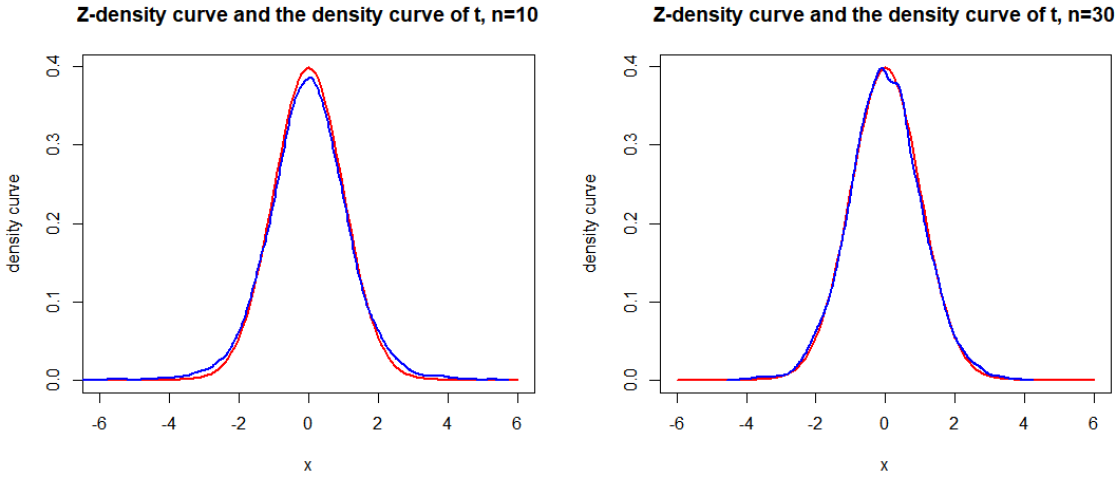

Figure 2. Comparison of the standard normal density curve and simulated density curves of the *t*-statistic. The red curves are the standard normal density curve and the blue curves are the simulated density curves with degrees of freedom 9 and 29, respectively

Confidence Intervals

Students know that confidence interval estimation is better than point estimation since they do not have any expectation that the point estimate will be identical to the parameter of interest. However, they have difficulty understanding what confidence level means, though some claimed that they can interpret a confidence interval intuitively. For example, some students believe that a given parameter is contained in a confidence interval with a known probability. The fact that a confidence interval may or may not capture the parameter of interest is often confusing for students. The study conducted by Hagtvedt, Jones and Jones (2008) found that not a single student in his graduate course had a proper understanding of what a confidence interval is.

Deriving the formula of a confidence intervals using probability statements can be easily understood by students. For example, the derivation of a 100(1-α)% *t*-interval of a population mean μ with normality assumption starts with

$$P\left(-t_{\alpha/2} < \frac{\bar{X}-\mu}{S/\sqrt{n}} < t_{\alpha/2}\right) = 1-\alpha,$$

where $\bar{X}$ is the sample mean, S is the sample standard deviation, and the *t*-distribution has degrees of freedom n-1. Manipulating the inequalities inside the parentheses results in the form of

$$P\left(\bar{X} - t_{\alpha/2}\cdot\frac{S}{\sqrt{n}} < \mu < \bar{X} + t_{\alpha/2}\cdot\frac{S}{\sqrt{n}}\right) = 1-\alpha.$$

The 100(1-α)% confidence interval $\left(\bar{X} - t_{\alpha/2}\cdot\frac{S}{\sqrt{n}}, \bar{X} + t_{\alpha/2}\cdot\frac{S}{\sqrt{n}}\right)$ of μ is a random interval since the sample mean $\bar{X}$ is a random variable in repeated sampling. However, the randomness disappears if a confidence interval is constructed from a specified sample. Traditionally, statistics textbooks use a diagram where several confidence intervals are constructed based on different samples with most of them containing the parameter while some do not contain the parameter to illustrate the concept of confidence interval estimation. It is helpful. But this type of static image cannot illuminate the dynamic relationship between repeated samples and confidence intervals. A computer simulation study makes students observe a resampling process where the relationship between these confidence intervals and the parameter is explicit.

In the following R simulations, 100 samples of same size n=15 are generated from the standard normal population. The t.test()function in R is used to produce a 95% confidence interval for each sample. The 100 confidence intervals are indicated by vertical lines with the true population mean μ=0 shown as a horizontal line. Vertical lines that intersect the horizontal line would indicate a confidence interval that contains the true population mean. If a confidence interval does not contain the parameter, the confidence interval will be colored red. Otherwise, the color will be green. After a confidence interval is plotted, a 0.5 second time is

suspended so that students can visualize the relationship between each confidence interval and the parameter dynamically.

```
n = 15; #sample size
mu=0; #mu=1; #population mean
conlevel=0.95; #confidence level
iter=100;
count=0; #number of intervals containing mu
L=numeric(); U=numeric();
CI=1:iter; par=rep(mu,iter);
plot(CI,par,type="l",xlim=c(0,iter), ylim=c(-2,2),lwd=2, col="green");
for (i in 1:iter)
{
Sys.sleep(0.5); #suspend 0.5 second
x=rnorm(n, mean=0, sd=1); #sample from standard normal
#x=runif(n, min = 0, max = 2); #sample from uniform[0,2]
interval = t.test(x, conf.level=conlevel)$conf.int; #C.I.
L[i]= interval[1]; U[i]=interval[2];
if (L[i]<=mu && U[i]>=mu) count=count+1;
time=c(i,i); bds=c(L[i], U[i]);
if (L[i]>mu | U[i]<mu)
{lines(time,bds,type="l", col="red", lwd=2);
} else {
lines(time,bds,type="l", col="blue", lwd=2);
}
}
count;
count/iter;
```

Figure 3 is produced by one of these simulations where 96% of the confidence intervals capture the true population parameter. To understand the confidence level, students can increase the number of samples, for example, to 10,000, to calculate the empirical coverage which is the number of confidence intervals containing the true parameter out of 10,000. Just make sure that the time suspend is removed, otherwise it is too slow. For example, one simulation shows that the empirical coverage is 95.052% which is very close to the confidence level 95%. The illustration is a solid way to strengthen students' understanding of the definition of confidence intervals.

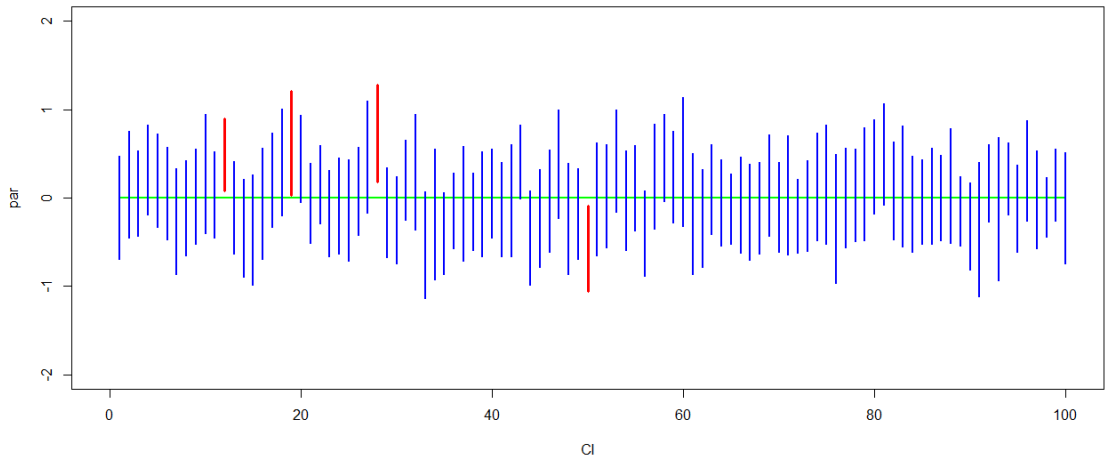

Figure 3. One hundred simulated *t*-confidence intervals of the standard normal population mean. The blue lines are intervals containing the population mean while the red lines are intervals not containing the population mean

Hypothesis Testing

Hypothesis testing in general statistics is used to make inference about population parameters. By assuming that the null hypothesis (H_0) is true, sample information is used to check if there is sufficient evidence to reject the null hypothesis in favor of the alternative hypothesis (H_A). Computer simulations then can be used to explore concepts such as type I error, type II error, and power in hypothesis testing.

R is able to run hypothesis tests for many different types of population parameters. It has special built-in functions that can run a hypothesis test. For example, prop.test() is used to test population proportions, t.test() is used to test population means, and var.test() is used to compare two normal population variances.

The type I error occurs when the null hypothesis is rejected based on information in a random sample while the null hypothesis is true. The type II error occurs when the null hypothesis is failed to be rejected while the alternative hypothesis is true. The probability of type I error, also called the significance level, is denoted by α, and the probability of type II error is denoted by ß. α and ß are related. Decreasing one will increase another if the sample information is fixed. To decrease both α and ß, the sample size must be increased. The power of a hypothesis test is defined as 1- ß which is the probability that the test correctly rejects the null hypothesis when a specific alternative hypothesis is true. Again, simulations using R can help students to better grasp the relationship between α and ß.

In the following simulation of hypothesis tests of a population mean, 10,000 samples of size n=15 are used. The hypotheses to be tested are H_0: $\mu = \mu_0 = 1.0$ versus H_A: $\mu > 1.0$. The hypothesis tests are conducted at significance level α=0.05. Furthermore, it is assumed that the samples are generated from a normal population under the alternative hypothesis with population mean μ=1.5 and variance $\sigma^2 = 2$. The empirical power of the *t*-test then is calculated as the number of random samples resulting in rejection of the null hypothesis out of 10,000. One simulation produces an empirical power of 0.3634. When the sample size in increased to n=50, an empirical power is 0.7867. Sample size of 100 produces an empirical power of 0.9725. Again, students can choose different significance levels or different mean μ values under the alternative hypothesis to investigate the relationship between α and ß.

```
n = 15; df = n-1; alpha = 0.05; counter = 0;
iter = 10000;            #Looping 10000 times
for (i in 1:iter)
{
x = rnorm(n, mean=1.5, sd=sqrt(2));          #Data generation
test = t.test(x, alternative = "greater", mu=1.0); # t-test
```

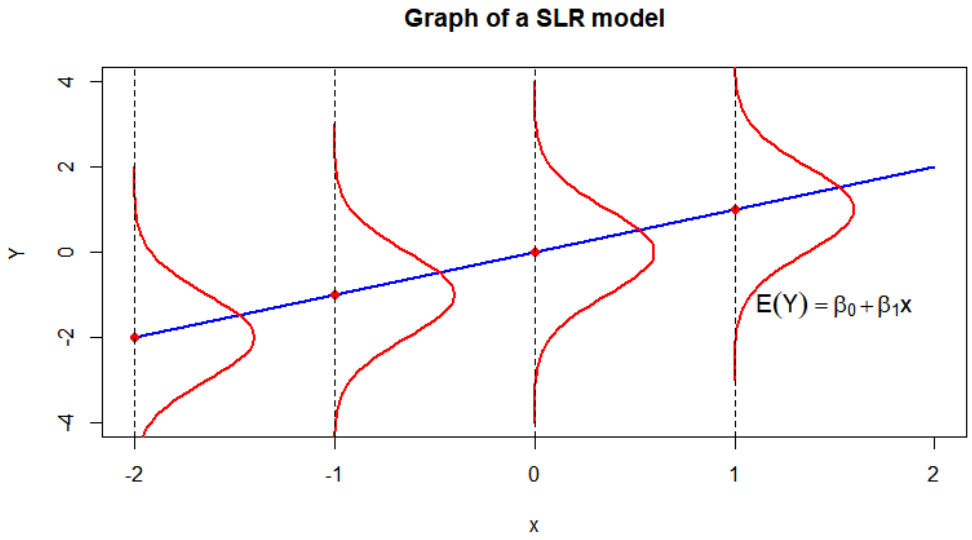

Figure 4. Probabilistic model of a simple linear regression. The blue straight line is the true regression line; the red dots are the mean of the response variable given a specified x value and the red curves are the Normal density curve of each conditional distribution

```
if(test$p.value <= alpha) counter = counter + 1;
 #Raising counter if H0 is rejected
}
power = counter/iter; power; #Calculating empirical power
```

Linear Regression Models

Simple linear regression is a statistical method to summarize and study relationships between two quantitative variables: the response variable, denoted by Y, and the predictor variable, denoted by X. The simple linear regression model is not a bi-variate distribution of X and Y. Instead, it is about the conditional distribution of Y given X=x: $Y|_{X=x_i} = \beta_0 + \beta_1 x_i + \varepsilon_i, i = 1, \ldots, n,$ where β_0 and β_1 are two parameters. And in general the random errors ε_i are assumed to be independent normal with mean 0 and common standard deviation, $i = 1, \ldots, n$. From this probabilistic model, the mean of the random variable Y given $X = x_i$ is $\beta_0 + \beta_1 x_i$, $i = 1, \ldots, n$. Therefore, the regression line $Y = \beta_0 + \beta_1 x$ is actually the line of means of $Y|_{X=x_i}$, where β_0 is the common intercept and β_1 is the common slope. Introductory statistics textbooks generally uses a diagram similar as **Figure 4** to illustrate the conditional distribution and the regression line is shown by the blue straight line, where the red dots are the mean of the response variable for each specified x value and the red curves are the Normal density curve for each conditional distribution. Again, like the diagram of a confidence interval, this type of static image cannot illuminate the relationship between the response variable and the predictor variable in the conditional distribution. R simulations can help students understand the two statistical concepts better.

In the following simulation study, the two population parameters β_0 and β_1 are chosen to be 0 and 1, respectively. The values of the predictor variable x are set to be -2, -1, 0, 1, and 2. Given each x-value, x_i, 10 realized values of the response variable y are generated from the normal distribution with mean $\beta_0 + \beta_1 x_i$ and common standard deviation 1. Thus the sample size is 50, the product of the number conditional distributions, 5 and the number of the values of the response variable for each population, 10. It can be easily seen from the scatter plot that there is a linear relationship between x and y. Thus the regression line is added to the scatter plot. Furthermore, the estimate of the regression line, colored by red, using the least squares method can be obtained by the lm() function in R. **Figure 5** is produced by one of these simulations. Students can change the values of the regression coefficients, β_0 and β_1 and the population variance to run lots of simulations.

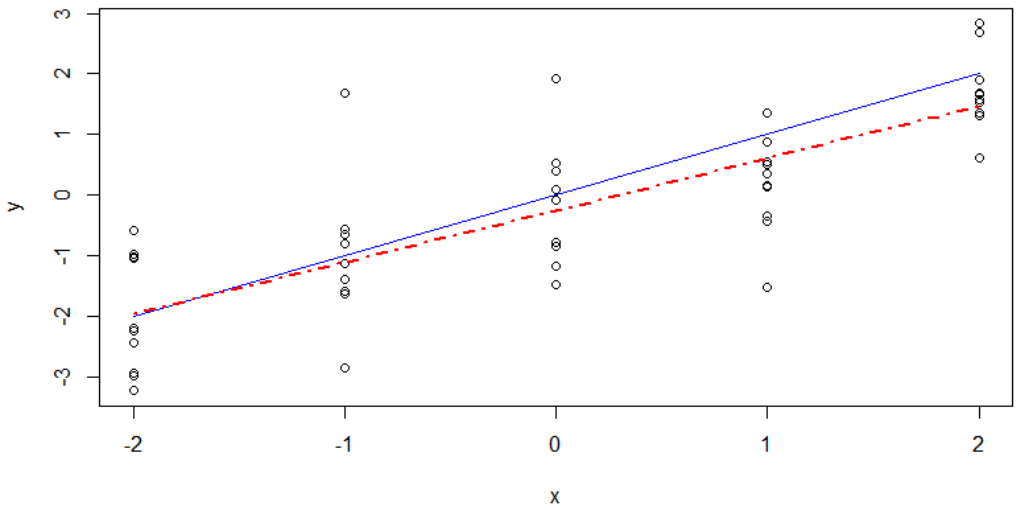

Figure 5. Comparison of the true regression line and the least squares line for a simulated data set. The blue line is the true regression line and the red dashed line is the least squares regression line fitted based on a data set of size 50 with 5 distinct *x* values

```
beta0=0; beta1=1;
xi=c(-2, -1, 0, 1, 2);
x=numeric(); y=numeric(); k=10;
for (i in 1:5)
{
x = rbind(x, matrix(rep(xi[i],k), k,1));
yi = rnorm(k, mean=beta0+beta1*xi[i], sd=1);
y = rbind(y, matrix(yi,k,1)); #data generation
}
plot(x, y); #scatterplot
lines(x, beta0+beta1*x, col='blue'); #plot the regression line
b0=lm(y~x)$coefficients[1];
b1=lm(y~x)$coefficients[2]; #model fit
lines(x, b0+b1*x, lty=4, lwd=2, col='red'); #plot of fitted regression line
```

Comparison of Nonparametric Tests and Parametric Tests

Simulation methods can also be used to compare nonparametric tests with parametric tests. Nonparametric tests do not require that samples come from populations with specified distribution assumptions, for example normal distributions, or homogeneous variances. Therefore, nonparametric methods can be applied to a wide variety of situations. However, nonparametric methods tend to waste information because exact numerical data are often reduced to a qualitative form such as ranks and thus they are not as powerful as parametric tests if assumptions for parametric tests are satisfied. R simulations can help students investigate the power of these two types of tests. The following simulation study compares the power of the sign test and *t*-test when the sample data of size n=15 are from a normal population with mean μ=0.5>0 and standard deviation σ=1. Let θ be the parameter describing the center of a population: $\theta=\mu$ for the *t*-test and θ=median for the sign test. To test H_0: $\theta = 0$ versus H_A: $\theta > 0$ at the significance level α=0.05, 10,000 samples are generated. One simulation produces an empirical power of 0.5825 for the *t*-test and an empirical power of

0.2647 for the sign test. Students can choose different population mean values under the alternative hypothesis to conduct more investigations.

```
n=15; alpha=0.05;
counter1=0; counter2=0;
iter=10000;
for (i in 1:iter)
{
x=rnorm(n, mean=0.5, sd=1); #data from Normal distribution
#x=rgamma(n, shape=0.2, scale = 1); #data from Gamma distribution
test1=t.test(x, alternative= "greater", mu=0);
if (test1$p.value <=alpha) counter1=counter1+1;
x1=sum(ifelse(x>0,1,0)); #number of positive signs
y1= sum(ifelse(x<0,1,0)); #number of negative signs
m=x1+y1; #sample size for the sign test
if (m<n) print("sample size is decreased");
test2=binom.test(x=x1, n=m, p=0.5, alternative="greater");
if (test2$p.value <=alpha) counter2=counter2+1;
}
counter1; counter1/iter;
counter2; counter2/iter;
```

It is expected that the sign test will be more powerful if the conditions for the *t*-test are not satisfied. Students can investigate this aspect by generating data from a highly skewed distribution under the alternative hypothesis, for example, Gamma distribution with shape and scale parameter 0.2 and 1, respectively. One simulation produces an empirical power of 0.7851 for the *t*-test and an empirical power of 1.0000 for the sign test. Students can choose different distributions under the alternative hypothesis to conduct more investigations. That is, the sign test is more powerful.

SUMMARY

Statistical inferences including confidence interval estimation and hypothesis testing involve uncertainty arising from the processes of sampling. It is not appropriate to discuss large amounts of probability theories in introductory statistics courses. Therefore, computer simulations are indispensable for students to understand most statistical concepts. Nowadays, statistical software is becoming an essential teaching and learning tool because it is able to provide students with a hands-on learning experience. Hagtvedt, Jones and Jones (2008), Mills (2002), and Raffle and Brooks (2005) have all reported positive outcomes from the use of statistical software. Sigal and Chalmers (2016) elaborated the essential aspects for writing Monte Carlo simulation code in R. Alias (2009) had positive outcomes for even basic forms of statistical visualizations using technology, both in terms of an improvement in student grades and student satisfaction with the class.

R can be used freely and is particularly strong software to use in a statistics course. It has built-in functions that students can use to solve statistical inference problems. Furthermore, students can improve their programming skills by conducting more simulation studies following the examples given by their instructor. From the experience of the authors, one is an instructor and one is a student, learning R and statistics together is feasible in an introductory statistics course. This paper just lists several typical simulation studies in introductory statistics, and vector is the only data structure used. Instructors can use this data structure or other data structures in R to investigate more simulation studies.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Xuemao Zhang – East Stroudsburg University, USA.

Zoe Maas – East Stroudsburg University, USA.

REFERENCES

Alias, M. (2009). Integrating Technology into Classroom Instructions for Reduced Misconceptions in Statistics. *International Electronic Journal of Mathematics Education, 4*(2), 77-91.

Aoyama, K. (2007). Investigating a Hierarchy of Students' Interpretations of Graphs. *International Electronic Journal of Mathematics Education, 2*(3), 298-318.

Arnholt, A. T. (1999). Simulating Sampling Distributions. *Teaching Statistics, 21*(1), 14–16. https://doi.org/10.1111/j.1467-9639.1999.tb00791.x

Best, J. W., & Kahn, J. V. (2006). *Research in education* (10th edition). New York: Pearson Higher Ed.

Brown, E. N., & Kass, R. E. (2009). What Is Statistics. *The American Statistician, 63*(2), 105-110. https://doi.org/10.1198/tast.2009.0019

Chandrakantha, L. (2014). Visualizing and Understanding Confidence Intervals and Hypothesis Testing Using Excel Simulation. *Electronic Journal Of Mathematics & Technology, 8*(3), 211-221.

Hagtvedt, R., Jones, G. T., & Jones, K. (2008). Teaching Confidence Intervals Using Simulation. *Teaching Statistics, 30*(2), 53–56. https://doi.org/10.1111/j.1467-9639.2008.00308.x

Haining, R. (2003). *Spatial Data Analysis: Theory and Practice* (1st edition). Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511754944

Innabi, H. A. (2007). Factors Considered by Secondary Students When Judging the Validity of a Given Statistical Generalization. *International Electronic Journal of Mathematics Education, 2*(3), 168-186.

Koh, H. C., & Tan G, (2005). Data Mining Applications in Healthcare. *Journal of Healthcare Information Management, 19*(2), 64-72.

Mendoza, D. J., & Mendoza, D. I. (2018). Information and Communication Technologies as a Didactic Tool for the Construction of Meaningful Learning in the Area of Mathematics. *International Electronic Journal of Mathematics Education, 13*(3), 261-271. https://doi.org/10.12973/iejme/3907

Mills, J. D. (2002). Using Computer Simulation Methods to Teach Statistics: A Review of the Literature. *Journal of Statistics Education, 10*(1), 1-21. https://doi.org/10.1080/10691898.2002.11910548

Quackenbush, J. (2001). Computational analysis of microarray data. *Nature Reviews Genetics, 2*(6), 418-427. https://doi.org/10.1038/35076576

R Development Core Team (2017). The R Project for Statistical Computing. Retrieved from http://www.R-project.org

Raffle, H., & Brooks, G. P. (2005). Using Monte Carlo Software to Teach Abstract Statistical Concepts: A Case Study. *Teaching of Psychology, 32*(3), 193-196. https://doi.org/10.1207/s15328023top3203_12

Ray, S. C. (2012). *Data Envelopment Analysis: Theory and Techniques for Economics and Operations Research*. Cambridge: Cambridge University Press.

Ridgway, J., Nicholson, J., & McCusker, S. (2007). Reasoning with Multivariate Evidence. *International Electronic Journal of Mathematics Education, 2*(3), 245-269.

Sigal, M. J., & Chalmers, R. P. (2016), Play It Again: Teaching Statistics with Monte Carlo Simulation. *Journal of Statistics Education, 24*(3), 136-156. https://doi.org/10.1080/10691898.2016.1246953

Smith, B. (2003). Using and Evaluating Resampling Simulations in SPSS and Excel. *Teaching Sociology, 31*(3), 276-87. https://doi.org/10.2307/3211325

Software Carpentry (2017). Understanding basic data types in R. Retrieved from http://resbaz.github.io/2014-r-materials/lessons/01-intro_r/data-structures.html