Statistical Analysis Of A Class: Monte Carlo And Multiple Imputation Spreadsheet Methods For Estimation And Extrapolation

Laurel J. Fish, California State University, Northridge, USA
Dennis Halcoussis, California State University, Northridge, USA
G. Michael Phillips, California State University, Northridge, USA

ABSTRACT

The Monte Carlo method and related multiple imputation methods are traditionally used in math, physics and science to estimate and analyze data and are now becoming standard tools in analyzing business and financial problems. However, few sources explain the application of the Monte Carlo method for individuals and business professionals who are not immersed in the realm of mathematics or science. This paper introduces these Monte Carlo methods for the non-mathematician and business student, providing examples where the Monte Carlo method is applied when only small samples are available. Statistical analysis and statistically sound extrapolation of sample characteristics to the larger class population can be facilitated by applying Monte Carlo methods and the related concept of multiple imputation, which is also explained. Appendices provide step-by-step instructions for using two popular spreadsheet add-ins to run Monte Carlo based analysis.

Keywords: Monte Carlo Method; Multiple Imputation; Small Samples; Analysis of a Class; Spreadsheets

INTRODUCTION

When studying some classes of people or entities, it may be impossible to study the whole population and it might even be infeasible or impossible to obtain a large enough sample to be able to use traditional statistical methods. As new data science and computational statistics methods become popular, they are offering alternative "computationally intensive" and "distribution free" methods for analysis that may permit the study of even small samples. Foremost of these methods are the "Monte Carlo" family of analytics. While well known in some areas of mathematical physics, computer science, and certain areas of management science, the Monte Carlo techniques have now become mainstream and are easily applied by business students and professionals across the business disciplines to analyze real data without invoking traditional normality assumptions.

Designed to introduce Monte Carlo methods, this paper also presents a synopsis of multiple imputation methods for estimating and extrapolating characteristics of small samples to larger classes. This paper is written for use as a supplement for classroom instruction, an introduction for students and faculty interested in pursuing these methods in their research, and as an overview for those generally interested in modern data analytics.

Our paper begins with a general overview, a review of the colorful history of Monte Carlo analysis and an overview of key principles. We next provide several useful business examples and show how those examples can be implemented by students and faculty using a popular "off the shelf" spreadsheet add-ins for performing such analysis. Detailed instructions for using the spreadsheet add-ins are attached in the appendix.

When you hear the words 'Monte Carlo' you may automatically think of the city in Monaco that sits on the Mediterranean Sea, the famous casino in that same city, gambling in general, or even the somewhat recent film released in 2011. Whatever you thought of before, forget it, because when mathematicians, scientists, statisticians, or economic consultants say the words 'Monte Carlo', they are referring to a broad range of mathematical and

statistical calculations referred to as Monte Carlo methods and the related multiple-imputation techniques. This paper introduces multiple-imputation and Monte Carlo methods for business students.

Traditional business statistics courses focus on "asymptotic distribution analysis", using tests based on the normal probability distribution and other large-sample based tests. These methods require large sample sizes to meet basic assumptions underlying the statistical procedures. In real life business situations, there might not be large populations or large samples available for analysis. In some cases, the entire size of a statistical population may be smaller than the sample size needed to be able to properly use normality-based testing for data analysis. The methods discussed in this paper facilitate the effective statistical analysis of smaller samples and populations.

Since the advent of modern computers, computationally intensive techniques, including Monte Carlo methods, have developed that permit using the results of a sample from a class in powerful new ways. This helps alleviate difficulties in common data analysis, including summarizing characteristics of a class and extrapolating from a sample to describe characteristics of the class. In this paper, we define a class as a group of people or any other similarly situated and distinctly identified individuals or items with something in common. A class could be comprised of students enrolled in a professor's course, the students enrolled in all the sections offered at a college, a "legal class" in litigation, or an "asset class" or group of similarly situated investments. The possibilities are nearly endless.

Monte Carlo style techniques are growing in popularity in industries and academia and while there are increasingly specialized textbooks exploring these methods, there are few simple introductions for the non-statistician. Since these methods are increasingly being applied, even non-statisticians need to have an understanding of the basics and know how to interpret results in order to be reasonable consumers of research and to create efficient solutions to personal or work related data issues.

Historical Overview

As early as the 1700s a French scientist by the name of Georges Louis LeClerc, Comte de Buffon used randomization in his successful efforts to simulate and mathematically estimate the numerical value of π (Stigler, 1999, p. 141). This Greek letter "pi" is used to indicate the ratio of the circumference to the diameter of a circle and is extremely important in many mathematical calculations. Simulations were also used by mathematicians, astronomers and scientists including Francis Galton, George H. Darwin and Erastus L. DeForest (Stigler, 1999, p. 151-152, 144). Later, in 1908, William Sealy Gosset also used random sampling techniques to empirically verify his work when estimating the t-distribution (Gosset, 1908, p. 14). Until the early 1930s there is no documentation to suggest that such random sampling techniques were used in any documented scientific or statistical analysis. In the early 1930s there is some evidence that Enrico Fermi, an Italian physicist, developed and frequently utilized something very familiar to the Monte Carlo method that we know of today, but Fermi did not record or publish this process (Metropolis, 1987, p. 128).

During the 1940s the ENIAC, the "first electronic computer" was created and became operable (Metropolis, 1987, p. 125-126). It was during this era of increased computer-based computation that Stanislaw Ulam and John von Neumann developed the Monte Carlo resampling process (Metropolis, 1987, p. 126-127). Ulam began developing the process in 1946 when he was contemplating ways to calculate the probability that a "Canfield solitaire" would be solved (Eckhardt, 1987, p. 131). Neumann saw that statistical random sampling could also be applied to work regarding "neutron diffusion in fissionable material," (Metropolis, 1987, p. 127). The process was termed as Monte Carlo when Ulam's and von Neumann's coworker Nicholas Metropolis, a physicist, suggested the name (Metropolis, 1987, p. 127). Metropolis suggested this name because Ulam's uncle often traveled to Monte Carlo and gambled with money borrowed from relatives (Metropolis, 1987, p. 127).

In 1947 von Neumann and R.D. Richtmyer, another physicist, wrote a report divulging the premise of the first application of Monte Carlo simulation which Ulam and von Neumann worked on so tirelessly (Richtmyer and von Neumann, 1947, p. 751). Additionally, the report includes letters; in the first, von Neumann provided a rough yet thorough outline of how the, then unnamed, Monte Carlo process could be applied to neutron diffusion issues and the second highlights some of Richtmyer's analysis of and suggested modifications to the process (Richtmyer and

von Neumann, 1947, p. 751-764). Attached to John von Neumann's letter was a "Tentative Computing Sheet" which outlines the coding that could be used to program the computer, namely ENIAC, to run such simulations (Richtmyer and von Neumann, 1947, p. 758-762). The title page of this report and its attachments remained as classified material until the U.S. Atomic Energy Commission declassified them in 1959 (Ulam, Bednarek and Ulam, 1990, p. 16).

Later on, this group of exceptional mathematicians and physicists applied the Monte Carlo method to multiple aspects of their work on the hydrogen bomb (Newman and Barkema, 1999, p. 26). Although Metropolis and Ulam did publish a paper entitled "The Monte Carlo Method" in 1949, the brief paper managed to provide a loose description of the process while simultaneously remaining vague enough to conceal the actual mechanics and applications of the method (Metropolis and Ulam, 1949, p. 335-341). Several decades later the use of the Monte Carlo method has finally become prevalent and has spread into applications for many other fields.

BASIC FRAMEWORK AND DISCUSSION

Initially, early versions of the Monte Carlo method were often applied only to specific applications, such as estimating the value of π or in estimating the t-distribution. As it is now however, the Monte Carlo method can be applied widely and will consistently provide excellent statistical assessment of and predictions for a class, whatever that class may be. Unfortunately, it is often only the mathematicians, scientists, statisticians, or economic consultants who understand what Monte Carlo methods are and how they work. The intent of the rest of this paper is to explain Monte Carlo methods in a clear and straightforward manner. Let us start with a basic framework.

Suppose that somehow information is obtained from a subset of a class. Perhaps several students in a finance course were randomly selected to complete a questionnaire regarding learning styles; or, perhaps a professor chose a sample of final exams to grade to provide guidelines to teaching assistants; or, perhaps a group of stocks was chosen from the class of "Large Capitalization Growth Stocks" and subjected to accounting analysis; or, to use a business law example, perhaps a group of litigants in a class action were deposed and the results of those depositions were used to estimate characteristics of the larger legal class.

Suppose that a sample of 10 was taken from a class of 50. Say the data collected from the sample were 1,3,3,5,5,9,2,4,5,1. Using basic statistics, the mean of this series is 3.8 and the sample standard error (the sample standard deviation divided by the square root of the sample size) is 0.757 (= 2.394/(10^0.5)). Using the assumptions of normality, one might expect that the class average would be within +/- 2.262*.757 =1.712 of 3.8, resulting in a confidence interval of (2.088,5.512). If instead a Monte Carlo bootstrap was set up, the results would still be a mean value of 3.8 but a 95% confidence interval of (3.18,4.36). This difference is because the small underlying empirical distribution of the sample did not follow a normal distribution well.

Monte Carlo

Notice that the 95% confidence interval estimated by the bootstrap Monte Carlo is tighter than the confidence interval calculated when it is assumed that the sample fits a normal distribution. Why does using a bootstrap Monte Carlo result in a tighter, stronger estimate?

First of all, a Monte Carlo bootstrap does not assume that the sample class is normally distributed. In other words, it does not assume that the sample fits into a smooth symmetric pattern where the average and mode lie practically in the center of the class where the median is. Typically, a class does not perfectly fit into a normal distribution and instead fits into one of a multitude of distributions or does not fit any distribution at all. Because a Monte Carlo bootstrap does not assume normality, it allows you to use the information that you have about the sample to estimate characteristics about the class regardless of the type of distribution it fits, or even if the sample does not fit any distribution at all.

Monte Carlo for Samples with no Distribution

When a computer runs a Monte Carlo simulation, the program takes the original values from the sample and records them as the initial pool of inputs. This is called a "bootstrap." In our initial example that means that the computer records the values 1,3,3,5,5,9,2,4,5,1 as the inputs that will be used later. Then the program generates another sample of the same size, in our case a sample of size ten. In order to generate this new sample, the program pulls the values for each entry in the sample from the original pool of values. While it would be easy to believe that this new sample is practically identical to the original sample, that is not the case here for one important reason. When conducting a Monte Carlo bootstrap, the program draws a value from the initial pool of values, then records that value as a data point in the new sample, and finally returns that value to the initial pool of values so that it may be used again. This is repeated for every data point until the new sample is complete, and then continued until the program has generated as many samples as you desire. Resampling is the term used to describe this entire process.

Fortunately, the Monte Carlo bootstrap does not stop here, or else the result would simply be tens, or hundreds, or thousands of samples with no statistic that describes the characteristics of the data or the overall class. After each sample is generated, the program can calculate the mean, and the upper and lower percentiles for a confidence interval of your choosing. Note that these statistics are the basic statistical results from a Monte Carlo bootstrap and with certain programs you will have the option of obtaining additional statistical values. With these values in hand you can continue to describe your sample.

Monte Carlo for Samples with a Distribution

Now that you can see the fundamentals underlying a basic Monte Carlo bootstrap, the next step is to consider what happens when your sample fits a distribution. In that case the Monte Carlo bootstrap once again generates new values for subsequent samples, in other words resamples. Instead of using the values of the original sample as the initial pool however, this time the program draws values from a distribution where the mean and standard deviation are the mean and standard deviation of the original sample. From here on, the process is identical to the process described above.

Multiple Imputation and Monte Carlo

Surely at this point, Monte Carlo bootstrap sounds wonderful, that is until you ask the question of how you are supposed to use a Monte Carlo bootstrap when you only possess incomplete data. Obtaining or possessing incomplete data plagues many statistical analyses on a regular basis. Fortunately, there has been much research and discussion on this matter and how to solve or ameliorate the issue.

The answer lies in a concept known as imputation. Imputation is the practice and procedure by which the gaps or missing pieces of information in the data are filled with appropriate estimates or even removed. Regardless of which type of imputation you choose to apply to your incomplete data it is always important to remember to clearly state what changes you have made to your data set before analyzing it. Although the realm of imputation is quite vast, a discussion of the basics and some pitfalls is necessary to make you a bit more informed about the matter.

Knowing why data are missing from your data set is often key in being able to distinguish and select an appropriate method of imputation. As outlined by Donald B. Rubin (1976, p. 584) and more thoroughly by Stef van Buuren (2012, p. 6-7), there are three main types of missing data. Missing data can be predominately classified as missing at random, missing completely at random or not missing at random (van Buuren, 2012, p. 7). Data are missing at random when the probability that a value is missing is the same as for any other value in the same group (note that this does not mean that the probability that values are missing is the same *across* the groups) (van Buuren, 2012, p. 7). As for data that is missing completely at random, the chance that a value is missing is the same across the whole data set (van Buuren, 2012, p. 7). The presence of data that are missing completely at random is rare and unlikely (van Buuren, 2012, p. 7). When data are classified as not missing at random, the data are missing in such a fashion that you do not know why they are missing (van Buuren, 2012, p. 7). Below we discuss the appropriate types of imputation to be used in each case.

Unless your data are missing completely at random there are a few types of imputation that you want to avoid. When you delete the observations containing missing data values, you are using listwise deletion (van Buuren, 2012, p. 8-9). Similarly, pairwise deletion works by taking the correlation between two variables into account and is also inappropriate to use unless your data are missing completely at random (van Buuren, 2012, p. 9-10). Why? If you already have a smaller data set it can be very costly in terms of information loss to simply delete entire entries. Also, if the data are not missing completely at random, it is quite possible that the missing data are inherently different than the available data. As such, any estimated statistics based on the existing data may be biased and skewed (van Buuren, 2012, p. 8-10).

Mean imputation, a type of single imputation, is a process by which you calculate the mean for the existing data and use the mean to fill in the 'gaps.' While this might sound like a good idea, if the data are not missing completely at random, then the mean will be initially biased, skewing the imputed data and throwing off all other statistics and calculations done on the imputed data (van Buuren, 2012, p. 10-11). If the data are missing completely at random however, the mean (of the data that is present) will not be biased (van Buuren, 2012, p. 10-11). Regardless that the mean will not be biased, mean imputation will produce biased estimates of any statistical value and "underestimate the [aggregate] variance," (van Buuren, 2012, p. 10-11). van Buuren even recommends that mean imputation should only be used when the amount of missing data is minute (van Buuren, 2012, p. 10-11).

While it would be advisable to avoid the types of imputation that are mentioned above, the type of imputation that we discuss next is a much stronger option. Unlike single imputation methods, multiple imputation creates a pool of plausible values (2 or more) for each missing value (Rubin, 1988, p. 80). It then fills in each missing entry with a value from each entry's corresponding pool of replacement values (Rubin, 1988, p. 80). Continuing until each value from each replacement pool has been used (Rubin, 1988, p. 80), the process generates two or more complete data sets across which the observed data are the same and only the imputed data values vary (van Buuren, 2012, p. 16). At this point, the process of imputation is complete.

Calculating statistical estimates that describe each of the completed data sets is now possible. Furthermore, it is possible to obtain summary statistical estimates for the class. A unique feature of multiple imputation is that because it has multiple replacement values for each missing value, it allows you to calculate a confidence interval for each observation and for the class as a whole. The confidence interval for the class takes into account not only the variation within each complete data set but also includes the variation between the completed data sets (van Buuren, 2012, p. 44, 49). The reason why this is helpful is summed up by de Waal, Pannekoek, and Scholtus quite nicely: "multiple imputation was meant from the outset as a method to provide not only a solution to the missing data problem by imputation but also to reflect the uncertainty inherent in the imputation," (de Waal, et al. 2011, p. 266).

Note that, "imputation is not prediction," (van Buuren, 2012, p. 46) and should not be used as such. Another important distinction that van Buuren makes is that, "the goal of multiple imputation is to obtain statistically valid inferences from incomplete data," and it should not be used in an attempt to "re-create the lost data," (2012, p. 45).

EXAMPLES

In this section, we will provide five examples of how to use imputation and Monte Carlo. As you will see, Monte Carlo simulations do not just apply to math, physics and science, but have many applications to business related topics.

The Dinner Party

Suppose you are planning the food budget for a college awards dinner. 300 people were invited and have accepted, however, 100 of the individuals have still not returned their "vegan, gluten-free, chicken, or fish" card indicating what they would want to eat. What should be ordered? This is an example where "multiple imputation" may be applied with Monte Carlo. Or, you could order proportionally. What would the margin of error tell you? What is the range that you would expect? (Suppose you prepare the upper 95% confidence interval for each option.)

Of your 200 responses, 7 people requested a vegan meal, 4 requested a gluten-free meal, 113 have requested chicken and 76 have requested fish. Now, how do you use Monte Carlo to estimate how many vegan, gluten-free, chicken, or fish dinners to order?

Your first step is to take all of the meal responses that you do have and enter the values into one column of an Excel spreadsheet (we suggest column B, you will see why in a moment). In the column to the left of this one (column A if you are following our suggestion) insert a count of the responses. Since in this example you have 200 responses, Column A should be filled with the numbers 1 through 200 in ascending order, as if you are making a list and numbering the items on the list.

Next, assign every meal type a different numerical value and create a key or a legend with this information in it so that you do not forget what each numerical value represents. In our example we have assigned a value to each meal type so that vegan = 1, gluten-free = 2, chicken = 3, and fish = 4. Now that you have your legend, appropriately fill in the third column (column C) with these numerical values. Remember to label each column appropriately.

In a new tab of the Excel spreadsheet, go down to the fourth row and click into a cell in the first column, you will want to leave some room at the top for calculations done later. Starting with that cell, enter the numbers 1 through 100 in ascending order. Label that column something like 'Guest_Number.' Now in the next few steps you will be implementing multiple imputation. In the cell to the right of number 1 use a RANDBETWEEN function to generate a random number between one and two hundred. Copy this cell down for 100 cells. Label this column as something like 'Guest_Chosen.' In the cell two cells to the right of number 1 create a VLOOKUP function that looks up the chosen guest number in the cell to the left on the previous sheet (Sheet 1) and returns the meal value that corresponds to that guest number. Label this column something like, 'Dinner_Type.' Note that in this case we are not having Excel randomly choose values from any distribution, rather we are pointing Excel to randomly choose a value from the distribution of meals actually requested by guests.

Now we could simply run a Monte Carlo on this and determine the average and confidence interval for each individual this way. However, in almost all instances the mean will turn out to be three (winner, winner, chicken dinner!) but intuitively we know that there must be some people who would prefer or even need one of the other meal choices, so ordering 100 chicken dinners does not make sense. Therefore, instead of running a Monte Carlo at this stage we press on a little further to gain a bit more insight.

Part of the problem here is that the values for each dinner (vegan=1, gluten-free=2, etc.) are arbitrary. Thus a mean of 3 indicating chicken doesn't have the meaning most people expect from a mean. Someone might mistakenly believe that the mean of 3 means we are somehow "below" fish and "above" gluten-free but this makes no sense. We could have just as easily set chicken = 1 in which case the mean would be one, which would be "below" gluten-free, chicken, and fish. The order of the numbers are arbitrary; this can cause confusion to others who did not design the experiment. Below, we show how to address this issue.

In the third cell to the right of number 1, create an "if" function that returns a 1 if the value in the Dinner_Type cell directly to its left is a 1 and a 0 otherwise. Copy this formula down for one hundred cells and then repeat this in the next three columns so that there is an "if" function that returns a 1 for each of the meal types if that number turns up. Finally, create sum formulas above these columns so that you know how many of each meal type have appeared.

In the last portion write formulas that will obtain the mean and percentiles for the number of meals for each meal type. There should be 4 functions for the mean, and four for each percentile since there are only 4 meal types.

After running a simulation with 1,000 iterations, we had an average of 3.472 vegan meals, 1.977 gluten-free meals, 56.547 chicken meals, and 38.014 fish meals. Of course, you can't order partial meals, so with rounding you would order 3 vegan meals, 2 gluten-free meals, 57 chicken meals, and 38 fish meals for a total of 100 extra meals.

Quality Control

Let us say that you are teaching a class in a large lecture auditorium. After you give an exam you randomly place each exam into one of four batches and give the batches to your four teaching assistants for them to grade. After your teaching assistants have graded the exams and entered the grades with their identifier (their initials appear in the cell next to each of the exams grades), you want to test whether the four batches of exams are reasonably similar to the grades that a fair grader would give, or if one of your teaching assistants is a bit too lenient or tough on grading. This is a perfect opportunity to use Monte Carlo.

Let us say that you have a list of grades that were given on this exam from this same course in a previous semester by a fair grader. And since you now have grades given on this exam by four new graders, you would like to see whether these new exam grades fall into a reasonable range when compared to the old fair grades. How would you determine this?

Admittedly, there are perhaps several ways that you could accomplish this. One way would be to determine the distribution of the old fair grades, run a Monte Carlo simulation that randomly draws values from that distribution and then estimates the overall average, and confidence interval. After accomplishing that, for each new set of scores, you would find the average of the scores and then determine whether that average falls within the confidence interval. If it falls within the confidence interval, then you might conclude that the new grader is within the range of "fairness," however, if the new grader's scoring average falls below or above the range of the confidence interval, then you might conclude that the new grader is too lenient or too tough, respectively.

While this would be a perfectly acceptable process, you may not be able to easily determine what distribution the old fair grades belong to. Thankfully, you do have a list of values for the old fair grades, and with this list you can run a Monte Carlo bootstrap which will help you estimate the mean, standard deviation, confidence intervals and other statistics. Here is how to accomplish this. To set this up, let us momentarily assume that the fair grader graded 50 exams. As such there are 50 exam scores that were graded fairly and can be used as a basis for the simulation.

In a new Excel spreadsheet, enter an index column with the values 1 through 50, let us say that you put this in column A. Enter or paste in the 50 fair exam scores in the next column, i.e. in column B. Subsequently, create a formula in the cells of column C that returns a randomly chosen value between 1 and 50. Then in the cells of column D write a VLOOKUP formula that will return whatever value lies in the cell directly to the right of the randomly chosen index number. In other terms, =VLOOKUP(C,$A$1:$B$50,2,FALSE).

Once this is complete, you can run a Monte Carlo simulation and receive estimates of the mean and standard deviation. This estimate of the mean will provide you with an idea of what the average score should approximately be for any batch of fairly graded exams. Granted, it will be rare that the average score for any batch of exams exactly matches this estimate, but this is why you also have an estimated confidence interval. A 95% confidence interval tells you that you are 95% confident that the true value of the average of fairly graded exams lies somewhere within that range. For future purposes then, it is highly unlikely that any batch of fairly graded exams should have a mean score that is below or above the range of the 95% confidence interval.

Now that you have these estimates, you can compare the average score of each TA's batch of graded exams to the average score of the fairly graded exams and determine whether they fall inside or outside of the confidence interval. Keep in mind that the results are estimates, and although Monte Carlo simulations provide highly accurate results, you still need to interpret the results in a logical and reasonable manner. What we mean by this, is that just because the average score from one TA's exam batch lies just above the upper bound of the confidence interval does not mean that the TA is too lenient of a grader. Remember that the average/mean is skewed by outliers. Thus, you would need to assess the exam scores given by that TA to see whether any high outliers occurred, and then review that student's exam to see if it was indeed graded appropriately. This situation is the same if the average score from one TA's exam batch lies just below the lower bound of the confidence interval.

Furthermore, this does not mean that the average score of one TA's exam batch is acceptable merely because it lies just below the upper bound of the confidence interval, or just above the lower bound of the confidence interval, i.e. barely within the bounds of the confidence interval. If this occurs, it would be reasonable to assess the distribution of grades within those batches of exams.

While this example may seem to only concern college professors, it can certainly apply to people in multiple fields. What if you are a scientist or a science student working in a lab and you are running multiple identical tests of an experiment? Furthermore, what if the results are measured and calculated by several different individuals to test a hypothesis. In this case, you would want to use a Monte Carlo to estimate whether the results obtained by the individuals are within a reasonable range of each other and the results from the original run of the experiment. Of course, in such a case the first run of the experiment may have been skewed, biased, or simply an outlier and the results from the secondary experiments will make that clear. In such a situation, each version of the experiment will be used to test all the other versions of the experiment.

Additionally, you might work for a city, county, state, province, or any other jurisdiction and your department is soliciting bids for a project. Whether the project is concerning repairing or building a road, park or building, Monte Carlo can assist you in assessing and comparing each bid to the others or to an older bid that the department deemed acceptable. Or you might even work for a manufacturing company and want to test the quality of production from multiple factories.

As you can see there are a plethora of available applications for a situation similar to our TA grading example, and it is highly likely that this can be applied in your field of work. Regardless, for these types of comparison situations be cautious of falling into the trap of using confidence intervals as a strict law of accept or reject. Use the statistical estimates from your Monte Carlo to help you dig deeper into borderline graders, experiments, bids, or quality production factories.

Another Take on Quality Control

Suppose you are performing a test of the 'class' of new computers delivered for a large campus lab (your firm's IT, HR, or sales department, whatever best applies to you), but you don't want to check every computer. How many have the proper software already installed?

Let us say that you had 200 new computers delivered and need to check whether the computers have the proper software, that you purchased, installed. You do not have the time to check all 200 but have the time to sample thirty of the computers. You randomly select 30 computers and have your IT crew assess whether each has the proper software installed. The IT crew finds that ten of the thirty computers do not have the proper software already installed. While you might desire to immediately assume that a third, or 33%, of the 200 new computers do not have software properly installed, it is possible that you randomly selected the only 10 computers, or randomly selected a majority of computers that do not have the proper software. As such you want to statistically assess approximately how many of the computers need software installed. This is a good time to use Monte Carlo, so here is how you would do it.

In the top left-hand corner of a blank spreadsheet, create a legend that assigns two separate values to the computers with or without the correct software. We have chosen to label computers that do not have the proper software installed with a one, and the computers with the correct software with a zero. A few columns over, create an index that numbers each computer so that you have a list of 1 through 200. To the right of the index in the first 30 slots, enter the appropriate 1's and 0's to represent the computers that you discovered that did or did not have software.

Subsequently, for the computers that your IT crew did not personally check (the computers indexed as 31-200) write the formula =RANDBETWEEN(1,30) into the column to the left of the index column. Then in the column with the 1's and 0's, write a VLOOKUP formula that will lookup up the value from the RANDBETWEEN function in the first thirty computers and return a 1 or a 0 indicating whether the computer has the software or not. Once you have copied the VLOOKUP formula down, have Excel calculate the sum of the 1's. This will give you an estimate of how many computers do not have the proper software installed.

It is important to note that the RANDBETWEEN and VLOOKUP functions do not affect the values from the original 30 computers sampled. Those values stay the same because they are what they are. Altering these initial set values in any way would undermine the simulation and bias the estimated results.

At this point, you may want to run a simulation or write additional formulas that will return the mean, confidence interval, standard deviation, etc. of the sum of computers without software from each iteration of a simulation. If you chose or needed to write additional formulas, then run the simulation afterwards with the add-in or program of your choice.

When we ran our own simulation for this example, the program estimated that on average, the number of computers without software out of the 200 was 66.584, or 67 computers. This told us that on average 0.33292 or 33.29% of the computers did not have the proper software installed. Notice that this estimate is extremely close to the proportion of computers without the proper software installed in your sample. While it was possible for you to make the same assessment based on your IT crew's initial search, this example shows you just how precise Monte Carlo simulations are. The same preciseness generated from the Monte Carlo simulation here, is generated in all Monte Carlo simulations that are properly set up, regardless of the complexity or simplicity of the scenario involving the class.

Cheating or Not Cheating?

Suppose you are teaching a large section of a 100-level class and just gave the first exam with 50 multiple choice questions. While grading, you find that there are two exams that have many matching incorrect answers. You want to test whether these two students are copying answers on an exam. But how do you accomplish this? Well, you can use the results of the exam to calculate the probability that a particular question is answered incorrectly and then use Monte Carlo simulation to estimate the probability that these two students would make the same errors on their exams.

First and foremost, you need to have some automated way to ascertain which pairs of the student's matching answers are incorrect answers. One way this can be accomplished is by setting up a spreadsheet in Excel that successfully tells us how many pairs of matching incorrect answers exist in the two students' exams. First, make sure that the actual answers are entered into column A. Note that regardless of whether you use letters or numbers to indicate the answers, this process will work. In columns B and C enter the two student's complete set of answers, one student's answers in the first column and the other student's answers in the second column. Then, in column D, write an IF formula that will return the answer value if the students answers match each other's and a zero otherwise. This formula should look something like =IF(B2=C2,B2,0) and means that if the students responded with the same answer for this particular question, return the answer; if not, return a zero. Finally, in column E write a similar formula that will compare the answer in column D to the actual answer listed in column A. If you used numerical values to represent the answers, then this formula should look like, =IF(D2=A2,0,IF(D2>0,1,0)). However, if you used letters, then it should look like =IF(D2=A2,0,IF(D2<>A2,1,0)). The formula means that if both students answered the question correctly, return a zero; if however the students answers did not match the correct answer, but they matched each other's answers, then return a one, otherwise zero. The last step would be to insert a row above all of these columns and, in one cell, write a formula that tells you exactly how many of the student's incorrect answers are matching.

Now you might be wondering why we chose to assign a zero to the correct answers and a one to incorrect answers. You are attempting to answer a problem regarding the incorrect answers, not the correct answers, and by assigning a one to the incorrect answers it becomes possible to obtain estimates regarding the frequency and probability of matching incorrect answers occurring. This would not be possible if you assigned a zero to the incorrect answers. Why? Recall that in many of these problems you have obtained the mean and some other statistical estimates that rely on the value of the mean. If all of the data points that you are trying to look at are zeros then the sum, average, and everything else will be zeros too.

Alright, so you have completed setting up a series of columns and formulas that provides you with the number of times in which the students have matching incorrect answers. But what comes next? Next, you should run a Monte

Carlo simulation that estimates the average number of times students have matching incorrect answers. Once you have done this, write a short formula that divides the average number of times students have matching incorrect answers on a test by the number of questions on the exam. This is the proportion of the time that this event occurs for these two exams.

Of course, this proportion is, by itself, inconsequential. However, note that roughly 25% of the time two individuals will randomly pick the same answer regardless of whether that matching answer is incorrect or not. Whatever proportion you receive from you Monte Carlo simulation then should be compared this this value.

Think about it for a moment, if any two students have studied then it is much more likely for them to select the correct answer. And since there is only one correct answer for each problem it is highly likely that any two well studied students will have matching correct answers. But let us say that there are four options for every multiple choice question, then three of the options are wrong. While there is a high probability that a student will pick an incorrect answer, the probability that two students will pick the same incorrect answer is much lower.

Keeping Track of Time

Suppose that you have two factories that produce the same product with the same process, and both have several thousand employees. One (Factory B) keeps perfect time records of the hours their employees work. The other, your factory, Factory A, has a time clock system that stopped working properly a year ago and thus had imperfect records for the last year. In order to avoid a law suit, you must estimate the hours worked by the employees in Factory A over the last year. Using the good time records from Factory B as a basis you can then run Monte Carlo simulation to estimate the appropriate hours employees worked at Factory A.

Since you do not have any accurate records from your factory (Factory A), you will have to rely on the time records from the other factory (Factory B). Of course you could go through and log or record all the time logs from the Factory B, but that will require more time and money than you have at your disposal. To further complicate matters, some employees are regular factory workers, some are inspectors, and some are managers. At times the hours worked by employees from each of the three groups vary widely; furthermore, the wage rates are at different levels for each of the three groups. So if you do not have the time to record all the time logs from Factory B, and the workers in your factory are from different groups and are paid at different rates, then how can you appropriately estimate the hours and pay your employees the right amount?

You could take a per pay period average of the hours worked by members of each of the three groups from Factory B, however if you assigned these three averages to each member of each respective group, you would end up overpaying some workers and underpaying others. The workers you would be overpaying probably wouldn't mind, but the workers you would be underpaying would not be happy and likely seek work somewhere else, maybe even at Factory B. Of course, you don't want that to happen since the people who worked more than the average are probably your harder working employees and you do not want to lose them!

Using a Monte Carlo bootstrap is an excellent alternative. First of all, it will solve your first problem of not having enough time or money to enter all the hourly data provided by Factory B. With Monte Carlo you will only need the hourly data from a sample of randomly chosen individuals, not from the entire class of employees. Secondly, the Monte Carlo simulation will provide you with a point estimate and a range of possible hours for each individual, not just one estimate for the whole class. Of course, you can still obtain aggregated estimates, but this solves your original concern of not being able to provide each person with an individualized value.

First, you will need to collect three randomly sampled groups of employee records, one random sample for each type of employee, general employees, inspectors, and managers. To make things straightforward, say you decide to randomly sample the hourly records of fifty employees from each of the three types of employees at Factory B. Next, you would create a spreadsheet that listed the hours worked per employee from Factory B per pay period. Create one sheet like this for each of the three employee types. For future purposes, make sure that you index each employee from Factor B with a number, such as one to fifty.

Subsequently, on a fourth and separate sheet you would create an index in one column (like we did in the previous examples) so that you have one row identified for each individual from Factory A. Unlike our previous examples however, perhaps you could use an employee ID number in place of a standard indexed value to index the employees. When you do this, it would be a good idea to create another column that indicated which employee type each individual was. For example, if the particular employee was a general employee perhaps you would key them in with a 1, an inspector as a 2, and a manager as a 3. If you create this column indicating whether an employee belongs to the general, inspector, or manager employee type, then you can use sorting or filtering to view the result for just one employee type if ever the need arises.

In the next (third) column, you will want to write a RANDBETWEEN function that returns a randomly chosen value between one and fifty (recall that you have sampled fifty Factory B employees from each employee type). In the fourth column, write a formula that takes into account the type of employee each individual is and the value produced by the RANDBETWEEN function in the third column and then uses these two pieces of information to look at the appropriate sheet and return the total number of hours worked by an employee in Factory B. Then in the fifth column write another formula that returns a particular wage rate value according to each individuals' employee type. Lastly, in the sixth column write a simple formula that is the value in column four multiplied by the value in column five, i.e. the values in column six will equal the amount of money that each employee will receive for their work in the last year.

Now that you have finished setting up the spreadsheet, it is time to run a Monte Carlo simulation. After running a Monte Carlo simulation, regardless of which program you use, you should end up with at least a mean, standard deviation, and confidence interval for the earnings of each employee. You can always sum these earnings to understand how much money Factory A will need to doll out, however the point here is that you can and should obtain earnings estimates for each employee.

Two appendices continue the discussion of how you can use Monte Carlo simulation to estimate statistics regarding a class of anything. The appendices provide detailed steps for downloading, installing, and using two different simulation software programs to run Monte Carlo simulations. One of the programs we discuss, called Poptools, is a free add-in that is used in Excel, and the other program, called RiskAMP, is a reasonably priced commercial program that smoothly interfaces with Excel.

CONCLUSION

Since the early 1700s, Monte Carlo style techniques have been used to estimate the values of unknown parameters. Through the diligent efforts of Stanislaw Ulam, John von Neumann and several others, what we know of today as Monte Carlo was transformed from an obscure method of resampling that was limited by the constrains of time and human based computations, into a revolutionary technique that steam-lined and freed the method from its earlier constraints by recruiting the help of early programmable computers.

A remarkable aspect of the current Monte Carlo is that it is “distribution free”. This simulation process does not require the data from your class (group) to fit any particular type of distribution or, in fact, any distribution as all. The beauty of this is that you often do not know whether the data from your class fits a particular distribution, and even if you do, it might fit a peculiar and often unused distribution. Some computer programs designed to run Monte Carlo simulations contain features to account for the presence of a distribution or to completely ignore one altogether. Additionally, Monte Carlo simulation is applicable to more analysis situations by incorporating multiple imputation to correct for missing answers or missing values; this method is preferred to filling missing values with means (mean imputation) or merely deleting the records with missing fields (listwise deletion or pairwise deletion). Whether you are planning an enormous dinner party, evaluating the work of your TA's or students, working in the IT department, or running analysis as an HR specialist, you can employ the use of Monte Carlo to generate statistically sound estimates even when your data are missing valuable pieces of information.

AUTHOR BIOGRAPHIES

Laurel Fish is a graduating senior at California State University, Northridge majoring in Economics with Honors in Business and minoring in Mathematics with an emphasis in statistics. She will be pursuing an M.S. in Economics at Cal Poly Pomona.

Dennis Halcoussis, Ph.D., is Professor of Economics at California State University, Northridge, where he teaches courses in econometrics and data analysis. He is the author of the textbook "*Understanding Econometrics*" and is the author of numerous papers applying econometric methods to diverse economic topics. He received his Ph.D. from the University of Pennsylvania in 1992.

G. Michael Phillips, Professor of Finance, Financial Planning, and Insurance, and Director of the Center for Financial Planning and Investment, California State University, Northridge. He also currently serves as President and Chief Scientist of the Center for Computationally Advanced Statistical Techniques, in Pasadena, where he specializes in statistics, risk measurement, risk management and investments. He received his M.A. and Ph.D. in Economics from University of California, San Diego.

REFERENCES

de Waal, T., Pannekoek, J., Scholtus, S. (2011). *Handbook of statistical data editing and imputation.* Hoboken, NJ: John Wiley & Sons, Inc.

Eckhardt, R. (1987). Stan Ulam, John von Neumann and the monte carlo method. *Los Alamos Science*, Special Issue.

Gosset, W. S. (1908). The probable error of a mean. Biometrika, 6, 1-25.

Metropolis, N. (1987). The beginning of the monte carlo method. *Los Alamos Science*, Special Issue.

Metropolis, N. and Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association*, 44(247), 335-341.

Newman, M. E. J. and Barkema, G. T. (1999). *Monte carlo methods in statistical physics.* New York, NY: Oxford University Press.

Richtmyer, R. D., & von Neumann, J. (1947). Statistical methods in neutron diffusion. *LAMS-551.*

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3), 581-592.

Rubin, D. B. (1988). An overview of multiple imputation. *Proceedings of the Survey Research Methods Section, American Statistical Association*, 79-84.

Stigler, S. M. (1999). *Statistics on the table: The history of statistical concepts and methods.* Cambridge, MA: Harvard University Press.

Ulam, S. M., Bednarek, A. R., Ulam, F. (1990). *Analogies between analogies: The mathematical reports of S.M. Ulam and his Los Alamos collaborators.* Berkley and Los Angeles, CA: University of California Press.

van Buuren, S. (2012). *Flexible imputation of missing data*. Boca Raton, FL: CRC Press.

APPENDIX 1

PopTools

This appendix provides an outline of steps that describe how to download, install and use the spreadsheet add-in called PopTools.

In order to download and install PopTools:

1. Go to www.poptools.org
2. Use the Navigation panel on the left of the site to navigate to the 'Download' page.
3. Read the directions listed there and choose the appropriate link. If you have a 64 bit computer with 32 bit Excel go ahead and click on the link in the second paragraph that just says 'here'.
4. Follow the computer prompted installation prompts and the rest of the download process should be straightforward.
5. Once installed, Excel should automatically open and a file entitled, 'Readme' will appear. In this Excel sheet, navigate over to the Add-ins tab on the Excel ribbon. On the left side of the ribbon, you should see 'Poptools' with a tiny little arrow to the right.

Using PopTools:

1. If you already have data that you want to run a Monte Carlo simulation on, go ahead and open that file.
2. Go to the Add-ins tab on the Excel ribbon and click on 'PopTools'.
3. In the subsequent drop-down menu scroll over 'Simulation tools' and select the first option, 'Monte Carlo analysis'. The Monte Carlo analysis option has a little symbol next to it that looks like a coffee cup with steam rising out of it. A menu window should open. Note that if you have two or more monitors, sometimes this menu window appears on a different monitor than the one your Excel spreadsheet is on.
4. Click on the button to the right of the box for 'Dependent range'.
5. Then select the portion of the spreadsheet that has your random variables or variables randomly chosen from a distribution, i.e. select the portion of the spreadsheet that contains the values that you want to simulate.
6. Next, click on the button to the right of the box labeled, 'Output (choose 1 cell)'.
7. Accordingly choose one cell that will serve as the base for the output. Choose this cell carefully, the Monte Carlo simulation output uses six columns and as many rows as you have variables and will overwrite whatever data that may be in its way. The cell that you choose will be the top left cell in the output.
8. Next, in the 'Number of replicates' box, verify how many replicates you wish the simulation to run. If you want the simulation to simulate values 1,000 times, then the 'Number of replicates' box should contain 1,000.
9. Finally set your Lower and Upper percentiles in their appropriately labeled boxes. PopTools has these preset to create a 95% confidence interval which is frequently used and typically appropriate. If you desire to use a 90% confidence interval instead, then simply enter 0.05 and 0.95 for the Lower and Upper percentiles respectively.
10. At this point, if you wish to save the raw results from each simulation for each data point, make sure that you check the little box next to 'Keep results'. The program will automatically save these results to a separate tab.
11. Finally click 'Go' and your simulation will run. Note that the time it takes to run a Monte Carlo will depend on the size and complexity of your model. If you have any formulas in your sheet that are not absolutely necessary for the Monte Carlo simulation to work, it may be a good idea to save a copy of your file and then copy and 'paste values' for all the cells with the unnecessary formulas before running the Monte Carlo.
12. The results will appear where you directed the output to go. From here you can then perform your intended analysis.

APPENDIX 2

RiskAMP

This appendix provides an outline of steps that describe how to download, install and use the spreadsheet add-in called RiskAMP. RiskAMP is software that works smoothly with Excel. Once installed it actually appears as a separate tab on your Excel ribbon and is labeled, 'Monte Carlo.' With a little practice the RiskAMP software will make a lot of sense to current Excel users and its plethora of functions and random distributions increase its functionality over competing software.

In order to purchase download and install RiskAMP:

1. Visit www.riskamp.com
2. Use the navigation bar at the top of the site to navigate to the 'Downloads' tab. This takes you to a screen where you can download a free 30-day trial. To purchase click on the 'Purchase' tab and explore your options.
3. After downloading RiskAMP, follow the computer prompts to install the software.
4. Once in Excel the RiskAMP software will appear in its on tab on the Excel ribbon and will be labeled as 'Monte Carlo'.

If the values you wish to simulate come from a distribution, then follow the directions from part 1 through part 3. However, if the values you wish to simulate do not come from a distribution, then skip the directions in part 1 and start with part 2.

Using RiskAMP Part 1: First, let us assume that you know that your population comes from a certain distribution with certain parameters and that you want Excel to randomly choose values for your sample from that distribution with the specified parameters.

1. In order to do this, you can use one of the numerous formulas that RiskAMP provides. The formula will always be of the form, ='DistributionName'Value() and the appropriate parameters will appear inside of the parentheses.
   a. Ex. If you want to pull values from a Normal distribution, then use the formula, =NormalValue(). For the Cauchy distribution, use the =CauchyValue() formula. Etc…
   b. To browse all of your options go to the Formulas tab on the Excel ribbon, choose Insert Function, select the RiskAMP Random Distributions category and scroll through the available options.
2. Copy your formula down for as many rows as desired.

Using RiskAMP Part 2: Running a Simulation

1. In order to run a simulation simply click on the large green play button entitled 'Run Simulation' that appears on the Monte Carlo Tab.
2. A separate window will appear. Enter the number of simulations/iteration you want the program to run and then click 'Start'. *There is no need to select the portion of the sheet you wish to be simulated, the program senses what portions can be simulated and which are not.
3. If, part way through a larger simulation you realize that you made an error somewhere, you can stop the simulation by pressing the 'Esc' button on your keyboard (which essentially pauses the simulation) and then closing the Monte Carlo Simulation window.

Using RiskAMP Part 3: Obtaining results from a simulation.

Option 1:

1. To obtain the mean, standard deviation, percentile etc… for any one variable, choose an appropriate empty cell and use the formula =SimulationMean(), = SimulationStandardDeviation(), =SimulationPercentile() etc…
2. Within parentheses for the simulation mean and standard deviation, simply refer to the cell that contains the previously simulated value. For the simulation percentile, after entering the cell reference, enter a comma followed by the specific percentile you wish the formula to return (i.e. 0.95 for the 95th percentile or 0.05 for the 5th percentile).
3. Copy that formula down for as many rows as you have simulated values.
   WARNING: The RiskAMP =SimulationStandardDeviation() function returns the population standard deviation, not the sample standard deviation.

Option 2: If you wish to incorporate regular Excel formulas, read through the following steps.

1. To obtain the mean, standard deviation, percentile etc… for a variable, use the regular, =AVERAGE(), =STDEV.S(), or =Percentile.INC() etc… functions.
2. Within the average and standard deviation functions, nest a SimulationValuesArray() formula that will reference the simulated value. For the percentile functions, still nest the SimulationValuesArray() formula, however continue to indicate which percentile you wish the formula to return.
3. Once again copy the functions down for as many rows as you have simulated values.

Option 3: To obtain the complete raw values from each simulation using RiskAMP, use the following instructions.

WARNING: This process becomes slow and tedious when large numbers of iterations are used.

1. After running a simulation, make sure that you have plenty of blank space in the Excel sheet. You need as many rows as you had iterations in the simulation, and as many columns as you have random values. The result will be that every column you are about to generate contains the raw simulation values for a different random value.
2. In your chosen cell, enter the function =SimulationValuesArray() with the appropriate cell reference in parentheses.
3. Edit the cell reference so that the column reference is locked but the row reference remains fluid. I.e. if you are referencing cell C3, the cell reference should be $C3.
4. Drag this function down so that there is a cell with a formula for each random value.
5. Next, copy the cells and past transpose so that the column of functions is now one row of functions. Now, each random value has its own column for raw simulation values.
6. Let us say that you had 1,000 iterations. Then in the first column select the cell that has the function inside of it then click and drag to highlight a total of 1,000 cells in that column.
7. Then click into the formula bar for that first cell, and press CTRL + SHIFT + ENTER. Brackets will appear around the function in the cell and the rest of the cells in your highlighted column will populate with the same function. While the functions will all look the same, each cell is retrieving a different raw simulation value and now displays them.
8. Finally, apply the standard Excel formulas (this time without the SimulationValuesArray() function nested inside) to obtain the mean, standard deviation, percentiles etc… of each column. This will give you the mean, standard deviation, percentiles etc… of each random value.

NOTES