Chapter 2
Probability Theory and Monte Carlo

In this chapter, the relationship between discrete event simulation and probability theory in general is described. Aside from creating insight-building animations, we will argue that discrete event simulation models are essentially calculators for estimating the "expected value" or "mean" of distributions. Therefore, reminding ourselves about the definition of the expected value is critical for comprehending simulation theory and the types of errors that arise in the practice of simulation.

This chapter focuses on the problem of predicting the registration and voting times in a future election. This problem is simple enough that it can be solved exactly without simulation. Therefore, it can be used to teach simulation for a case in which the true answer is known. Simulation estimates expected values with an error. Therefore, the example permits evaluation of the errors from simulation.

Further, the scope in the example here is the same as that from the previous chapter summarized by Fig. 1.1 and Table 1.2. By generating predictions for the expected times in three ways, the reader will also gain an appreciation for the "leap of faith" (LOF) and the associated concerns that are almost inevitably encountered in attempting to predict future events as well as the specific times at which the necessity for such leaps are typically encountered during the analysis process.

2.1 Random Variables and Expected Values

This section provides a review of elementary probability theory. (If the reader can confidently define expected values for continuous and discrete random variables, please skip to Sect. 2.2.)

We define a "random variable" (X) as a number whose value is not known at time of planning by the planner. While the value is unknown, generally the planner is comfortable with assuming a distribution function to summarize his or her

T. T. Allen, *Introduction to Discrete Event Simulation and Agent-based Modeling,*
DOI: 10.1007/978-0-85729-139-4_2, © Springer-Verlag London Limited 2011

beliefs about the random variable. If the random variable is discrete, i.e., it can assume only a countable number of values, then the distribution function is called a probability mass function, $\Pr\{X = x_i\}$ for $i = 1,\ldots,n$. For example, X might represent the number of fingers (not including my thumb) that I am holding up behind my back. John Doe, a student, might have beliefs corresponding to:

$$\begin{aligned} &\Pr\{X = 0\} = 0.1, \\ &\Pr\{X = 1\} = 0.2, \\ &\Pr\{X = 2\} = 0.3, \\ &\Pr\{X = 3\} = 0.3, \quad \text{and} \\ &\Pr\{X = 4\} = 0.1. \end{aligned} \tag{2.1}$$

It is perhaps true that no one can tell John Doe that he is wrong in his current beliefs, although things that John might learn later might change his beliefs and his distribution. The above distribution has no name other than "discrete distribution" in that it is not Poisson or binomial (two famous discrete distributions). It is particular to John and his current state of beliefs. Yet, if John Doe declares the first 4 probabilities and then declares a value for $\mathrm{P}\{X = 4\}$ other than 0.1, we might reasonably say that John Doe is incompetent in his ability to apply probability theory.

Next, for discrete random variables, the universally acknowledged definition for the mean or expected value is given by:

$$\text{mean} \equiv \mathrm{E}[X] \equiv \sum_{\mathrm{i}=1,\ldots,n} x_i \Pr\{X = x_i\}. \tag{2.2}$$

For example, if we apply the distribution in Eq. 2.1 above, then the expected value is:

$$\mathrm{E[X]} = (0)(0.1) + (1)(0.2) + (2)(0.3) + (3)(0.3) + (4)(0.1) = 2.1 \text{ fingers}. \tag{2.3}$$

Philosophically, the expected value is just as subjective as the distribution. It is not wrong to calculate an expected value and then to changes one's mind about his or her distribution. Yet, if one has a change of mind about the distribution in (2.1) logically (2.3) must be changed accordingly.

Similarly, if a random variable is continuous, it can (hypothetically) take on any value on at least some section of the real line. Continuous random variables are characterized by density functions also called "distribution functions" and written as $f(x)$. Distribution functions assign values proportional to the subjective likelihood of a random variable, X, achieving a value in the neighborhood of the value x. Just as appropriate or proper discrete density functions must have their probabilities sum to 1.0, appropriate continuous function distribution functions must have their values from $-\infty$ to ∞ integrate to 1.0.

As an example, consider that John Doe may state that his beliefs about the temperature in his home are characterized by the distribution function in Fig. 2.1.

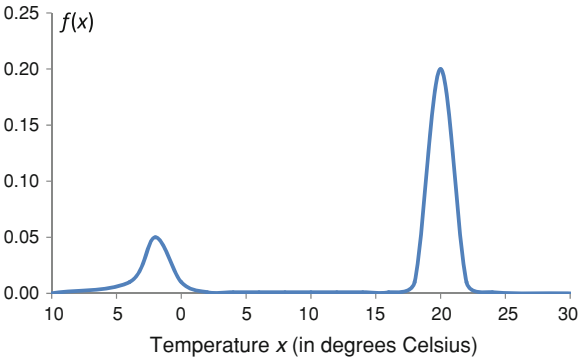


Fig. 2.1 John Doe's density function describing his beliefs on room temperature

This distribution function does not resemble any frequently encountered, i.e., "famous" probability distribution and it is particular to John Doe at the time when he makes his declaration. It reflects his concern that the heater might break down, i.e., he believes that there is at least some chance of subzero temperatures.

The definition for the expected value of a random variable X given a continuous probability density function, $f(x)$, is defined as:

$$\mathrm{E[X]} = \int_{-\infty}^{\infty} xf(x)dx \tag{2.4}$$

A custom distribution as in Fig. 2.1 provides special challenges for estimating the expected value. This follows because there are only a finite number of cases for which we have the anti-derivatives to directly calculate the value in Eq. 2.4. There is no well-known anti-derivative for the function in Fig. 2.1. Custom distributions are important in discrete event simulation because generally the random variable whose expected value one is estimating does not resemble any frequently encountered continuous distribution. It might be the waiting time of a voter which is influenced by many factors such as machine breakdowns and the arrival rates of other voters.

A visual scan might yield an expected value of approximately 15°C, which appears to lie roughly at the center of mass of the distribution. In practice, we often attempt to find the closest "famous" distribution that roughly fits our beliefs. Using these famous distributions, we gain access to pre-established formulas relating the parameters that describe our assumed distributions to the mean and other properties of these distributions.

In this book, we focus on four well-studied or "famous" continuous distributions. These are the:

- **Uniform** (equally likely to be anywhere between a and b) written U[a,b],
- **Triangular** (must be greater than a, is most likely to be m, and must be less than b) written TRIA(a,m,b),
- **Exponential** (could be anywhere in the ball-park of $1/\lambda$) written EXPO($1/\lambda$), and
- **Normal** (somewhere within 3σ of the assumed mean μ) written N[μ,σ].

Note that much of the statistical literature writes the normal distribution as N[μ,σ^2], where σ^2 is the distribution variance. We choose to follow the excel "=NORMDIST()" conventions using the standard deviation instead of the mean. Also, the exponential is typically written in terms of its parameter λ and the reciprocal, $1/\lambda$, is the exponential mean or expected value.

We will have much more to say about each of these distribution functions. We will also describe so-called "empirical distribution" functions mainly for cases in which one has a large amount of data. In these cases, it might be assumed that the future data will be like the past data and not be limited by the shape of any famous distribution function. Obviously, predictions about important future events involve some degree of uncertainty. The distortion of approximating our true beliefs by one of the famous functions generally decreases the trust in our prediction process.

2.2 Confidence Intervals

Next, we review the standard approach to derive the confidence interval for the mean of a random variable based on data. Because these details are unusually important in the context of this book, even a reader with a good knowledge of elementary statistics, may want to read this section carefully.

In our notation, the data is written $X_1, X_2, \ldots, X_n$ where n is the number of data points. This approach creates an interval having a reasonably high and regulated probability of containing the true mean under specific assumptions given by the parameter α ("alpha"). The ability to create appropriate confidence intervals is considered essential to much of the simulation theory described in this book.

2.2.1 Confidence Interval Construction Method

Step 1. Calculate the sample mean (Xbar) using:

$$\text{Xbar} = (1/n) \sum_{i=1,\ldots,n} X_i \tag{2.5}$$

Step 2. Calculate the sample standard deviation (s) using:

$$s = \left\{ \left[\sum_{i=1,\ldots,n} (X_i - \text{Xbar})^2 \right] / (n-1) \right\}^{1/2}. \tag{2.6}$$

where $\{\}^{1/2}$ means take the square root of the quantity inside $\{\}$.

Step 3. Calculate the half width of the confidence interval using:

$$\text{Half width} = t_{\alpha/2,n-1} s / \left(n^{1/2} \right) \tag{2.7}$$

Table 2.1 Critical values of the t distribution ($t_{\alpha,\mathrm{df}}$)

df	α			
	0.01	0.025	0.05	0.1
1	31.82	12.71	6.31	3.08
2	6.96	4.30	2.92	1.89
3	4.54	3.18	2.35	1.64
4	3.75	2.78	2.13	1.53
5	3.36	2.57	2.02	1.48
6	3.14	2.45	1.94	1.44
7	3	2.36	1.89	1.41
8	2.9	2.31	1.86	1.4
9	2.82	2.26	1.83	1.38
10	2.76	2.23	1.81	1.37
20	2.53	2.09	1.72	1.33

and declare that the interval equals Xbar $\pm$ half width where typically $\alpha = 0.05$ and the value of $t_{\alpha/2,n-1}$ is found by consulting Table 2.1. In applying the formula in (2.7), with an overall value of $\alpha = 0.05$ and n data, we would look for the values with 0.025 and $df = n - 1$.

Step 4. (Optional). Check that that it is reasonable to assume that the individual data derive independent, identically distributed (IID) from a (single) normal distribution. If not, then do not trust the interval. Batching described in Chap. 4 might provide a useful way to derive trustworthy intervals for some cases.

Note that the symbol "*df*" stands for degrees of freedom, which has a geometric interpretation in the context of various statistical methods such as analysis of variance. It is merely an index to help us pick the right value from the table here.

Also, Step 4 is often not included in descriptions of confidence intervals but we will argue that this test is practically important in the context of output analysis in Chap. 5. Also, the details of the IID normally distributed conditions will be discussed at length in Chap. 4. In addition, Step 4 provides some motivation for the distribution fitting based methods described later in this section. This pertains to the voting systems example considered next.

Returning to our election systems example, imagine that the input analysis phase has progressed yielding the data in Table 2.2. These would hypothetically come from $n = 9$ registered voters from time trials with a stop watch in a mock election. Such a mock election would use ballots similar in length to what is our best projection for ballots in the 2010 gubernatorial election.

In Chap. 1, an example provided the team charter associated with this prediction problem (Table 1.2). In the context of this project, we are now in a position to complete phases 2, 3, and 4 simultaneously. We can derive a defensible prediction for the expected or mean sum of registration and voting times. With such a prediction, we can skip forward to Phase 5 (Decision Support).

If we use $X_1 = 7.4, \ldots, X_9 = 5.4$ and apply the confidence interval construction method, we will have our prediction. This is based on the assumption that the

Table 2.2 Registration and direct recording equipment (DRE) times in election systems example

Person	Registration time(min)	Voting using DRE machine time (min)	Total (min)
Fred	0.2	7.2	7.4
Aysha	2.1	4.5	6.6
Juan	0.4	8.1	8.5
Mary	0.8	9.2	10
Henry	1.1	4.2	5.3
Larry	0.3	12.3	12.6
Bill	0.8	15.1	15.9
Jane	0.2	6.2	6.4
Catalina	0.6	4.8	5.4

future election will be similar to our mock election. Also, being good analysts, we will have error bars on our estimate. We will derive these from the confidence interval construction method applying the derived half widths.

Step 1. We calculate the sample mean (Xbar) using:

$$\begin{aligned} \text{Xbar} &= (1/n)\sum_{i=1,\ldots,n} X_i = (1/9)[7.4 + 6.6 + \cdots + 5.4] \\ &= 8.68\,\text{min}. \end{aligned} \tag{2.8}$$

Step 2. We calculate the sample standard deviation (s) using:

$$\begin{aligned} s &= \left\{ \left[\sum_{i=1,\ldots,n} (X_i - \text{Xbar})^2 \right] / (n-1) \right\}^{1/2} \\ &= \left\{ (7.4 - 8.68)^2 + \cdots + (5.4 - 8.68)^2 / 8 \right\}^{1/2} \\ &= 3.58\ \text{min}. \end{aligned} \tag{2.9}$$

Step 3. We calculate the half width of the confidence interval using:

$$\begin{aligned} \text{Half width} &= t_{\alpha/2,n-1} s / \left(n^{1/2} \right) = (2.31)(3.58) / \left(9^{1/2} \right) \\ &= 2.75\,\text{min}. \end{aligned} \tag{2.10}$$

The interval is 8.68 ± 2.75 min or (5.92 min to 11.43 min).

Step 4. Figure 2.2 shows a histogram of the nine summed times. More precise methods to evaluate distributions quantitatively is described in Chap. 3. Such distribution testing is generally only trustworthy with 20 or more data points. Yet, here we merely say that the normal distribution is probably not a great fit. The observation of a second hump suggests that the true distribution might not be governed by a single bell shape. Strictly speaking, we know that the famous normal distribution is rarely (if ever) a perfect fit for a real process. However, in this case the fit is "extra" unreasonable.

Next, we describe two methods for generating predictions that do not depend on the assumption that the individual data are approximately normally distributed. Yet,

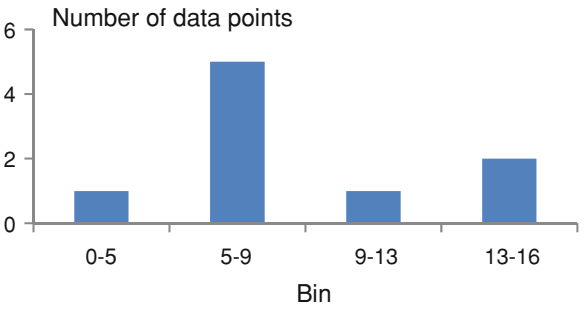


Fig. 2.2 Histogram for the registration plus voting time data showing two peaks

both of the methods that follow are associated with other strong assumptions that might give us concern. Despite the problem with normality, generating a confidence interval as shown in the above example is likely a defensible way to answer the problem stated. This explains why we say that simulation is probably not needed.

2.3 Expected Value Formula and Leaps of Faith

The method described in this section involves fitting famous distributions to the registration and voting times. The expected values are then calculated using the formulas associated with the famous distribution. Pencil and paper mathematics then permits the derivation of the forecast for the future expected registration time plus voting time.

The details of distribution fitting methods are the focus of Chap. 3. Here, let us assume that some software magically works through our data sets and fits triangular distributions to registration and voting times separately. Figure 2.3 shows the output from one such magical software package the Rockwell® Input Analyzer® which comes as a standalone in the same folder as the ARENA software. To develop this output using the Input Analyzer® one:

1. Opens the software and a new project using the File menu,
2. Puts the data in *.txt files perhaps using the NotePad built into Microsoft® operating systems, e.g., 0.2, 2.1,…,0.6 with each number in its own row and no commas or other separators,
3. Goes to File → Data File → Use Existing…, changes the "Files of type:" option to *.txt and selects the data file generated in the previous step, and
4. Selects "Triangular" from the "Fit" menu.

Ignoring the precise details temporarily, we now have a fit distribution. In our predictions of the future, let us entertain the assumption that registration times (in minutes) will come from a TRIA(0, 0.229, 2.29) distribution. Under this assumption, all times will be greater than 0 min (which makes sense), will have the most likely value of 0.229 min (we could live with that), and will be less than 2.29 min (an assumption that is somewhat limiting but may be acceptable).

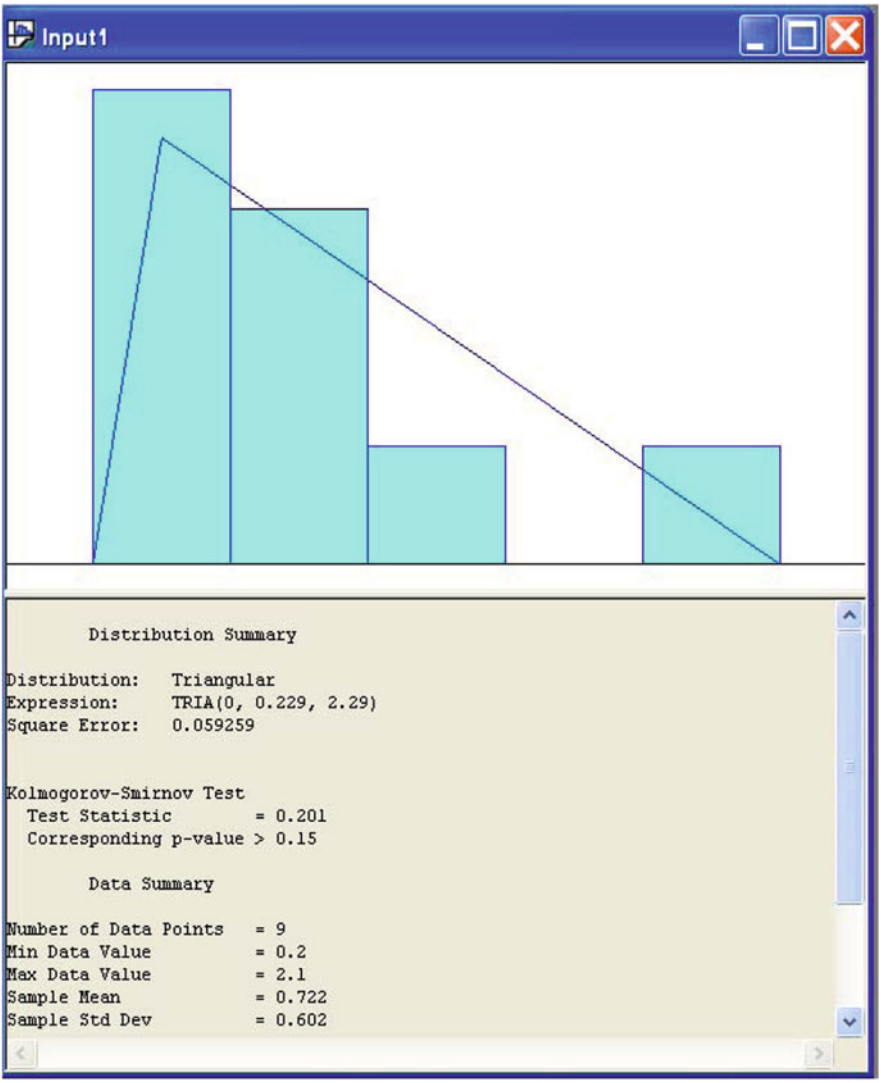


Fig. 2.3 Output from applying the Input Analyzer to the registration set

At this point let us take our leap of faith. From now on, we will play down the details of our nine real data points and simply assume TRIA(0, 0.229, 2.29) describes our beliefs. A similar process leads us to TRIA(4.0, 5.2, 16.0) min assumptions about future voting times, i.e., times voters need to use the direct recording equipment (DRE) machines once they are given access to their machines.

Having made our leap of faith, we are done with input analysis. We are ready for the calculation phase (Phase 3). In this phase, we are ready to gain one of the

benefits of applying the famous triangular distribution. This benefit is that for parameters a, m, and b, the general formula for the mean of a triangularly distributed random variable is given by:

$$\mathrm{E}[X] = (a + m + b)/3 \tag{2.11}$$

We can also access the following general rule applicable to all pairs of random variables X_1 and X_2:

$$\mathrm{E}[X_1 + X_2] = \mathrm{E}[X_1] + \mathrm{E}[X_2] \tag{2.12}$$

This rule is general because it follows directly from the definitions of expected values in Eqs. 2.2 and 2.4.

In our example, we have X_1 is TRIA(0.0, 0.229, 2.29) and X_2 is TRIA(4, 5.2, 16). Again, when we made these assumptions, we can say that we made our "leap of faith" and "entered simulation land" where our input analysis data are irrelevant. Plugging the numbers into Eqs. 2.11 and 2.12, we see that our assumptions implied a predicted expected sum of registration and voting times is:

$$\begin{aligned} \mathrm{E}[X_1 + X_2] &= (0.0 + 0.229 + 2.29)/3 + (4.0 + 5.2 + 16.0)/3 \\ &= 9.239666667 \pm 0.000000\,\mathrm{min} \end{aligned} \tag{2.13}$$

Clearly, Eq. 2.13 is misleading. We know results are not infinitely trustworthy. However, we are in "assumption land" and our choice to entertain X_1 is TRIA(0.0, 0.229, 2.29) min and X_2 is TRIA(4.0, 5.2, 16) min has effectively caused these uncertainties to be ignored or irrelevant.

At least, our answer is not dependent on the data coming approximately from a single normal distribution. Also, next we will show how simulation in the same example adds a new type of "Monte Carlo simulation error" that makes the calculation in (2.13) look good in comparison. Generally, when one can apply calculus or probability theory to directly calculate an expected value, one should do it. Discrete event simulation merely estimates expected values with an error.

2.4 Discrete Event Simulation

This section introduces two key technologies. These are: (1) linear congruential generators (LCGs) and (2) inverse cumulative distribution functions. Together, these form the nontrivial components of by-hand discrete event simulations and permit simulation using spread sheets.

In the context of our election systems example, the application of these technologies turns out to derive a less desirable prediction for the expected times than those developed previously. However, the discrete event simulation methods introduced here have advantages in more complicated situations for which the previous methods (simple confidence intervals based on data and using calculus to derive expected values) are not applicable. So, for convenience, we introduction

simulation technology in the context of an example for which we know the answer can be derived more accurately from probability theory or calculus.

A linear congruential generator (LCG) is a reasonably simple way to generate numbers that are "**pseudo random**" and associated with a uniform $a = 0$ and $b = 1$, i.e., U[0,1] distribution. We say that numbers are "pseudo random" because:

- They are not random in that we have a way to predict them accurately at time of planning and
- They closely resemble truly random numbers from the distribution in question.

There are pseudo random numbers of various levels of quality. Simulation trainers know that pseudo random numbers from LCGs are generally low in quality. This is because statistical tests can fairly easy show that they do not closely resemble actual U[0,1] random numbers. Yet, we use LCGs for instruction purposes because they illustrate the key concepts associated with pseudo random numbers.

The following method defines linear congruential generators (LCGs) in terms of three parameters: a^*, c^*, and m^*. It is just a coincidence that we typically use a, c, and m when working with triangularly distributed numbers. This explains why we use the symbol "*" to clarify the difference.

2.4.1 Linear Congruential Generators

Step 1. Start with a seed, e.g., $Z_0 = 19$, and $i = 0$.
Step 2. $i \rightarrow i + 1$;

$$\begin{aligned} Z_i &= \text{modulo}[(a*)(Z_{i-1}) + c*, m*], \\ U_i &= Z_i/(m*), \quad \text{and} \end{aligned} \tag{2.14}$$

where modulo is the standard function giving the remainder of $[(a^*)(Z_{i-1}) + c^*$ when divided by m^*.

Step 3. Got enough? Yes, stop. No, go to Step 2.

In our example, we use $a^* = 22$, $c^* = 4$, and $m^* = 63$. Yet, the above method defines an LCG for many combinations satisfying a^*, c^*, and $m^* > 3$. The quality of the random numbers depends greatly on the specific choice with generally larger numbers spawning increasingly uniform seeming sequences of pseudo random U_i. The seed can also be important.

For example, Table 2.3 shows a sequence of 10 pseudo random uniformly distributed random numbers from an LCG. For completeness sake, the standard definition of a uniformly distributed random number from $a = 0$ and $b = 1$, i.e., U[0,1], is given by $f(x) = 1.0$ for $0.0 \leq x \leq 1.0$ and $f(x) = 0$ otherwise. Given this definition, does it seem reasonable to say that the numbers U_i for $i = 1,\ldots,10$ in Table 2.3 resemble random numbers from a U[0,1] distribution?

Table 2.3 10 pseudo random U[0,1] numbers from a linear congruency generator

i	Z_i	U_i
0	19	–
1	44	0.698413
2	27	0.428571
3	31	0.492063
4	56	0.888889
5	39	0.619048
6	43	0.682540
7	5	0.079365
8	51	0.809524
9	55	0.873016
10	17	0.269841

The answer depends on how accurate we are trying to be. Yes, they seem to superficially resemble uniformly distributed random numbers. However, continuing the sequence we observe cycling every 63 numbers. Consider that, in typical real problems, one uses 1 billion or more pseudo random U[0,1] numbers. Therefore, our LCG is not up to the task. We observe one type of departure from uniformity (cycling) after only 63 numbers. This occurs long before we get to the billion numbers that we need.

Industrial strength pseudo random U[0,1] number generators like "=RAND()" function in excel are far more complicated than LCGs. Yet, like LCGs they generate pseudo random numbers. Also, they have seeds, e.g., 19 in Table 2.3.

For the same seed, their sequence or "stream" is the same. The "=RAND()" function does not permit us to access the seed and it changes every time an Excel sheet field changes. However, if the "AnalysisToolPak" is added into excel, one can access the "Tools" → "Random Number Generator" feature. In more recent versions of excel, these options are available under Data → Data Analysis → Random Number Generation. Using whichever is appropriate to your version, one can generate streams of high quality pseudo random numbers of several types with adjustable seeds.

2.4.2 Inverse Cumulative Distribution Functions

Once we have pseudo random U[0,1] random numbers, it is generally of interest to convert them to pseudo random numbers of distributions of greater interest to us. For example, we might want pseudo random TRIA(0.0, 0.229, 2.29) numbers to generate plausible registration times for our simulated election. There are generally many approaches for converting a stream of pseudo random U[0,1] into numbers of the type that we desire. However, if we merely want a sequence of uncorrelated random numbers from a distribution of interest, a common and efficient approach is based on so-called inverse cumulative distribution functions, $F^{-1}(x)$.

Consider that the inverse cumulative distribution function for triangularly distributed random numbers is:

$$F^{-1}(u|a,m,b) = a + [u(m-a)(b-a)]^{1/2} \quad \text{for } u \le (m-a)/(b-a)$$
$$\text{or } b - [(1-u)(b-m)(b-a)]^{1/2} \quad \text{otherwise} \tag{2.15}$$

where $[]^{½}$ means take the square root of the quantity in the brackets. Neglect temporarily how one derives this function. The key fact is that, once we have $F^{-1}(x)$, we simply plug in our pseudo random U[0,1] number as u and we derive a pseudo random number according to the distribution of interest.

For example, if we plug in $u = 0.698413$ and $a = 0.0$, $m = 0.229$, and $b = 2.29$ (in min), then Eq. 2.15 gives $F^{-1} = 1.07$ min. It can be checked that $u \le (m-a)/(b-a)$ so that we apply $b - [(1-u)(b-m)(b-a)]^{½}$ in this example. This is our first pseudo random number according to the TRIA(0.0, 0.229, 2.29) distribution. It is not entirely trustworthy because it derives from an LCG, but it might seem plausible as a hypothetical registration time.

To better understand how inverse cumulative distribution functions work, consider the uniform distribution function, $f(x)$, its cumulative $F(x)$, and its cumulative inverse distribution function, $F^{-1}(x)$:

$$f(x) = 1.0 \quad \text{for} \quad a \le x \le b \quad \text{or } 0.0 \quad \text{otherwise}, \tag{2.16}$$

$$F(x) = \int_{-\infty}^{\infty} f(z)dz = 0.0 \quad \text{for } x \le a, \tag{2.17}$$

$$(x-a)/(b-a) \quad \text{for} \quad a \le x \le b, \text{ and}$$
$$1.0 \quad \text{for } x \ge b, \text{ and} \tag{2.18}$$
$$F^{-1}(u) = a + (b-a)(u).$$

First, note that the inverse cumulative distribution function F^{-1}(u) in Eq. 2.18 intuitively serves our purpose. If we plug in a number between 0.0 and 1.0, u, the result obtained lies between a and b. If u is closer to 0.0, then the result will be closer to a. If it is closer to 1.0, then the result will be closer to b. That is reasonable and desirable.

Also, consider how one can derive Eq. 2.18 from Eq. 2.17. Substitute $u = F(x)$ and solve for x. The result should give the right hand side in Eq. 2.18 for the relevant cases in which u is between 0.0 and 1.0. Finally, consider the plot of the cumulative distribution function, $F(x)$, as shown in Fig. 2.4. The starting numbers are equally likely to be between 0.0 and 1.0. Figure 2.4 shows a hypothetical value of 0.3000. Reading over and reading down gives a pseudo random number 0.3 fraction of the way from a to b.

The relationship between the inverse cumulative and generating pseudo random numbers is perhaps made clearer if we consider the custom distribution that John Doe hypothesizes or believes about the room temperature. Figure 2.5 shows how

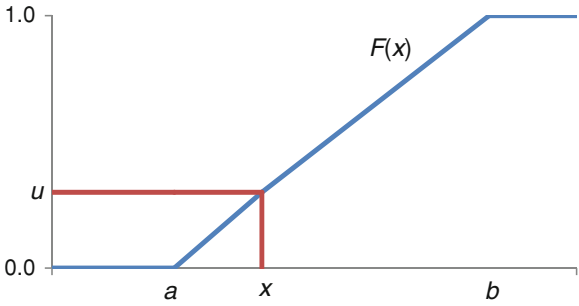


Fig. 2.4 Plugging in pseudo random (PR) U[0,1] u's generates PR U[a, b] x's

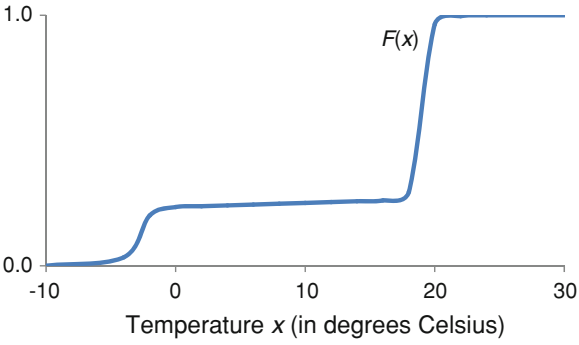


Fig. 2.5 Cumulative distribution function for custom temperature distribution

the slope of the cumulative is proportional to distribution function. John expects that the temperature will most likely be around 20°C but has a not insubstantial chance of being around −2°C. If one starts with a uniform pseudo random u on the vertical axis, one can see that it is almost certain that the result of reading over and down will either be around either −2° or 20°. Other values have very little chance (or "cross-section") to occur.

Other approaches for generating pseudo random numbers from specific distributions include the "acceptance-rejection method" which is useful even when F^{-1} is not available. In this method, variables are generated from one distribution, $g(x)$ from which it is easy to sample. Then, the numbers are conditionally eliminated based on a condition. Often, this condition relates to the ratio of probability density functions $f(x)/g(x)$.

2.4.3 Discrete Event Simulation

Now, we can generate pseudo random U[0,1] numbers using an LCG. We can also convert these numbers to pseudo random numbers from other distributions for which we have inverse cumulative distribution functions, $F^{-1}(u)$. We are ready to put the results together and generate discrete event simulations. Therefore, we will

simulate all of the random numbers needed to complete one full event or replicate (sometimes call "replication") of interest.

The outputs from our simulations will be, in general, pseudo random numbers according to custom distributions whose mean values we are trying to estimate. We call these "discrete event" simulations because the quantities being simulated generally relate to occurrences happening at specific, identifiable times, i.e., events.

For example, consider a simulation of voters registering and voting. The scope of this simulation was defined previously in the define phase (Phase 1) to include only these two activities. In this fairly trivial scope, events associated with other voters and their interactions are irrelevant. A full replicate corresponds to the experience of a single simulated voter. The response of interest is the time elapsed between the event when a voter starts registration and the event when that voter completes voting using the voting machine. Here, one is trying to predict or estimate the expected value or mean of this elapsed time.

Table 2.4 shows five replications of discrete event simulation for the voter registration and voting prediction project. On the left-hand-side is the stream of pseudo random numbers from the LCG. Next, each number is transformed to a pseudo random time using the appropriate cumulative inverse distribution function, $F^{-1}(x)$. For this purpose Eq. 2.15 is applied. If the time is a registration time, then $a = 0.0$, $m = 0.229$, and $c = 2.29$ is used. If the time needed is a voting time, then $a = 4$, $m = 5.2$, and $c = 16$ is used. The resulting pseudo random number is not from any famous distribution. It is from the sum of two triangular distributions. This sum distribution has no special name.

Yet, it can be shown that the resulting stream of numbers, 8.491, …, 7.788 are pseudo random independent identically distributed (IID) with an unknown true mean value equal to 9.239666667. This is the same number we derived previously using the exact formulas for the expected value. Since they are IID (more about this is discussed in Chap. 4) and from a distribution that is somewhat like a normal distribution, it is reasonable and appropriate to apply the confidence interval

Table 2.4 Discrete event simulation of 5 simulated voters registering and voting

I	Z_i	U_i	Simulated voter or replicate	Registration	Voting	Simulated time
0	19	–		–	–	–
1	44	0.698413	1	1.097	–	–
2	27	0.428571	1	–	7.394	8.491
3	31	0.492063	2	0.742	–	–
4	56	0.888889	2		12.205	12.947
5	39	0.619048	3	0.949		–
6	43	0.682540	3		9.586	10.535
7	5	0.079365	4	0.204		–
8	51	0.809524	4		11.032	11.236
9	55	0.873016	5	1.516	–	–
10	17	0.269841	5	–	6.272	7.788

construction method from Sect. 2.2 to generate a range describing the mean. Remember, we are pretending that we do not know the mean equals 9.239666667 min and merely estimating it using our simulation results.

Discrete event simulation is one type of a more general form of statistical simulation called "Monte Carlo" simulation. The exact relationship is not important here. The name "Monte Carlo" derives from the city located near the south of France where gambling has historically occurred. Statistical simulation theory was often motivated by applications relating to gambling including gambling in Monte Carlo.

Our general formula for Monte Carlo estimates for expected values is:

$$\text{Monte Carlo estimated expected value} \equiv \text{Xbar} \equiv \text{the sample average} \quad (2.19)$$

with Monte Carlo errors estimated using the half width from our confidence interval. In our example, the sample average equal Xbar equals 10.2 min with sample standard deviation (s) equal to 2.1 min. The half width is 1.9 min from Eq. 2.7. Therefore, the Monte Carlo simulation estimate can be quoted as 10.2 min $\pm$ 1.9 min.

Note that the error for Monte Carlo estimates in Eq. 2.19 declines according to the number of simulation replicates, n, that we choose to do. The exact proportionality is given by our confidence interval half width Eq. 2.7 as $1/n^{½}$ or the reciprocal of the square root. This means that the simulation error is directly attributable to our failure to simulate additional replicates. If we run more replicates, we reduce the error. Yet, in some cases of interest, replication times can consume more than 1 h. In those instances, we will generally need to live with large half widths. This is not true, however, for our spread sheet simulations such as the one in Table 2.4. In this case, it takes virtually no time to copy formulas down and perform 10,000 replicates. Yet, in this case, we are limited by the poor quality of our LCG.

2.5 Monte Carlo Errors

The difference between our Monte Carlo estimate, Xbar, and the true mean, which we might denote E[X], is an error. We define the Monte Carlo error as simply:

$$\text{Monte Carlo error} \equiv \text{E}[X] - \text{Xbar}. \quad (2.20)$$

From our discussion of confidence intervals, we know that we can drive this error to near zero by performing sufficient replicates. Consider that, in general, simulation is performed using pre-made software with easily adjustable numbers of replicates. Therefore, the only excuse for not driving the Monte Carlo errors to near zero is a function of the time each replicate requires from the computer processors and the patience of the human analysts who are awaiting the results.

The Monte Carlo error is in addition to any error we might imagine stemming from an imperfect input analysis phase. In our example, we know we only had nine data points. As a result, our results cannot be particularly trustworthy. Our leap of faith is making us concerned and the Monte Carlo error only adds to that concern.

In conclusion, the particular simulation example described here happens to permit us to directly estimate the Monte Carlo error. The true mean under our assumptions, E[*X*], we know from direct calculation is 9.239666667 min. The Monte Carlo estimate is Xbar = 10.2 min. Therefore, the Monte Carlo error from Eq. 2.20 equals −1.0 min. Generally, we will not know our Monte Carlo errors and need to estimate or bound them using half widths from our confidence intervals.

2.6 Monte Carlo Simulation Example

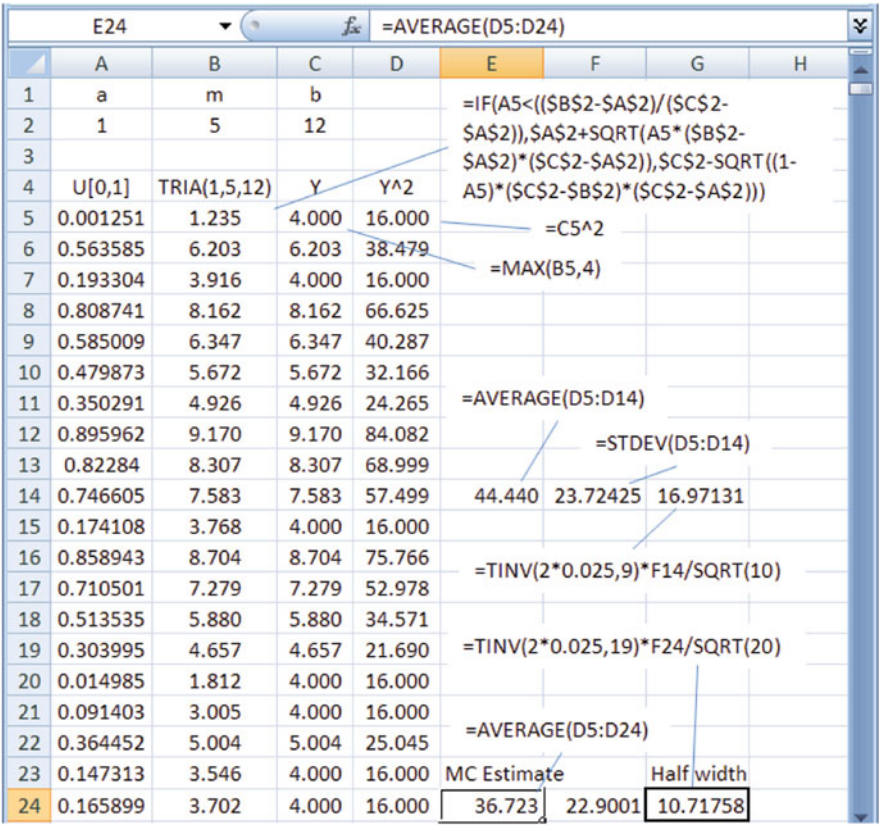
2.6.1 Problem

Consider a pseudo random number X that is assumed to be TRIA(1, 5, 12). Also, consider another random variable, *Y*, that is either *X* or 4, whichever is greater. We can write: $Y = \text{Maximum}(X,4)$. Use Monte Carlo simulation to estimate $E[Y^2]$. Perform sufficient numbers of replicates until the half width of your Monte Carlo estimate is less than or equal to 11.

2.6.2 Solution

The overall strategy is to generate pseudo random numbers Y^2 and to use the sample mean to estimate the true mean, i.e., apply Eq. 2.19. Since we are not explicitly asked to apply an LCG, we will not use them. Instead, we will apply the higher quality pseudo random numbers from Excel-based on Tools → Random Number Generation → Uniform random numbers or, in some more recent versions, Data → Data Analysis → Random Number Generation → Uniform. We set the seed equal to 1. We use the formulas shown in Fig. 2.6.

The spreadsheet shows how the process starts with pseudo random numbers in cells A5 and below. Then, the inverse cumulative is used to generate triangularly distributed pseudo random numbers, *Y* values, and Y^2 values successively. The distribution of Y^2 has no famous name but we can estimate its mean using the sample average. After 10 replicates, the Monte Carlo estimate is 44.4 but the half width is 16.97, which is too large. Therefore, we apply 20 replicates. This yields a Monte Carlo estimate for the mean equal to 36.7 and a half width of 10.7. This is good enough for our purposes. Note that the "=TINV" function has been applied. This function requires that we multiply our alpha values by 2.0 to obtain the standard critical values that we need.



	A	B	C	D	E	F	G	H
1	a	m	b					
2	1	5	12					
3								
4	U[0,1]	TRIA(1,5,12)	Y	Y^2				
5	0.001251	1.235	4.000	16.000				
6	0.563585	6.203	6.203	38.479				
7	0.193304	3.916	4.000	16.000				
8	0.808741	8.162	8.162	66.625				
9	0.585009	6.347	6.347	40.287				
10	0.479873	5.672	5.672	32.166				
11	0.350291	4.926	4.926	24.265				
12	0.895962	9.170	9.170	84.082				
13	0.82284	8.307	8.307	68.999				
14	0.746605	7.583	7.583	57.499				
15	0.174108	3.768	4.000	16.000				
16	0.858943	8.704	8.704	75.766				
17	0.710501	7.279	7.279	52.978				
18	0.513535	5.880	5.880	34.571				
19	0.303995	4.657	4.657	21.690				
20	0.014985	1.812	4.000	16.000				
21	0.091403	3.005	4.000	16.000				
22	0.364452	5.004	5.004	25.045				
23	0.147313	3.546	4.000	16.000				
24	0.165899	3.702	4.000	16.000				

Fig. 2.6 Microsoft® excel used to estimate the mean of a random variable

2.7 Voting Systems Example Summary

Collecting results from various methods, we first forecast the expected voting time to be 8.7 ± 3.6 min. This estimate involves a minimal leap of faith because we simply applied a confidence interval from the original input analysis data. Yes, the original data were not normally distributed to a good approximation (two humps). Still, the interval does reflect with some appropriateness the limitations of our nine subject data set. As a result, it is probably the best answer to the original forecasting question described here.

Our next interval derived from fitting distributions to the data. Then, standard formulas from probability theory were applied to estimate a mean of 9.239666667 ± 0.000000. We know that the precision in this estimate is misleading. The ±0.000000 does not reflect the leap of faith that we made after concluding our input analysis when we picked the two TRIA distributions.

Then, we put blinders on and took our assumed distributions seriously. Yet, at least the 9.2239666667 min estimate is associated with zero Monte Carlo estimation error.

The last estimate, from discrete event simulation was 10.2 ± 1.9 min. We know that this estimate is the worst. The predicted mean has a nonzero Monte Carlo error and the uncertainty (±1.9 min) is actually an under estimate. This is because the ±1.9 min estimate error bound is simply Monte Carlo or "replication error" and ignores the additional errors associated with our input analysis.

In our real simulation project for the county, we focused on quantities such as the expected waiting times of the worst precincts. Such quantities cannot easily be estimated in any other way besides Monte Carlo discrete event simulation or other numerical techniques. Therefore, we did make a leap of faith and ignored the errors related to our input analysis. Also, we could not calculate our true Monte Carlo errors since exact formulas for the mean were not available.

We simply estimated bounds on our errors using confidence interval half widths (based on Eq. 2.7). We tried to keep the errors to a reasonable level by applying 20 or more full replications. Usually, in research we use 10,000 replicates to get three decimal points of accuracy. But in the real election systems case, simulations were far too slow to permit that. Fortunately, the Monte Carlo errors were small enough for providing helpful decision support. In Chap. 4, we focus on other simulations of expected waiting times for additional cases in which Monte Carlo or discrete simulations are needed.

2.8 Problems

1. What is a random variable?
2. What is an expected value of a random variable?
3. What is a linear congruential generator (LCG)?
4. Why do we generally avoid using LCGs in addressing real world problems?
5. What is the "sample standard deviation"?
6. What is a "half width"?
7. What are Monte Carlo errors?
8. Consider the following data: 2.5, 9.2, 10.2, 9.8, 9.2, 10.3, 10.2, 2.8, and 10.1. Develop a 95% confidence interval for the mean assuming that the data are approximately normally distributed.
9. Consider the measured registration times 1.0, 2.4, 2.0, 3.5, and 1.4. Develop a 95% confidence interval for the mean using the given t-table.
10. Comment on how reasonable it is to assume that the data in problem 8 derive from a single, normal distribution.
11. Consider the measured voting times 5.0, 7.0, 9.0, 5.4, 3.0, and 4.4 with sample mean 5.6 and sample standard deviation 2.1. Develop a 95% confidence interval for the mean using the given t-table.

12. Considered the outputs from different replicates of a simulation given by 22.1, 18.3, 25.7, and 22.8 (waiting times in minutes). Give the Monte Carlo estimate for the mean waiting time and its half width.
13. Assume X is distributed according to $f(x)$, and 10.1, 19.4, and 23.0 are pseudo-random numbers from $f(x)$. Also, assume $\mu = \int_{-\infty}^{\infty} xf(x) = 19.0$. Estimate as accurately as possible E[$X + 3X$] and Var[X]. Estimate the errors of your estimates.
14. Assume X is distributed TRIA(4, 9, 10). Estimate E[X^2] using Monte Carlo simulation.
15. Assume X is distributed according to $f(x)$, and 9.1, 20.3, 19.4, and 23.0 are pseudo-random numbers $f(x)$. Also, assume $\int_{-\infty}^{\infty} xf(x) = 22.0$. Estimate as accurately as possible E[$2X$] and E[X^2]. Estimate the errors of your estimates.
16. Assume that X is U[10,25], what is E[X]? Estimate the answer using probability theory and also Monte Carlo simulation.
17. Assume X is triangularly distributed with parameters $a = 2$ h, $b = 10$ h, and $m = 3$ h. What does this assumption imply about E[X]? Also, describe this assumption in one or two sentences using everyday language.
18. Assume that X is U[10,25], what is E[X^2]? Estimate the answer using Monte Carlo simulation. Make sure your half width is less than 1.0.
19. Assume X is triangularly distributed with parameters $a = 2$ h, $b = 10$ h, and $m = 3$ h. Generate three pseudo random triangularly distributed random variables using the inverse cumulative and the uniform pseudorandom numbers 0.8, 0.3, and 0.5.

http://www.springer.com/978-0-85729-138-7

Introduction to Discrete Event Simulation and Agent-based Modeling
Voting Systems, Health Care, Military, and Manufacturing
Allen, T.
2011, XII, 215 p., Hardcover
ISBN: 978-0-85729-138-7