CHAPTER 10

Simulation

10.1 INTRODUCTION

How can we determine the probability of our winning a game of solitaire? (By solitaire, we mean any one of the standard solitaire games played with an ordinary deck of 52 playing cards and with some fixed playing strategy.) One possible approach is to start with the reasonable hypothesis that all (52)! possible arrangements of the deck of cards are equally likely to occur and then attempt to determine how many of these lead to a win. Unfortunately, there does not appear to be any systematic method for determining the number of arrangements that lead to a win, and as (52)! is a rather large number and the only way to determine whether a particular arrangement leads to a win seems to be by playing the game out, it can be seen that this approach will not work.

In fact, it might appear that the determination of the probability of winning at solitaire is mathematically intractable. However, all is not lost, for probability falls not only within the realm of mathematics, but also within the realm of applied science; and, as in all applied sciences, experimentation is a valuable technique. For our solitaire example, experimentation takes the form of playing a large number of such games or, better yet, programming a computer to do so. After playing, say, n games, if we let

$$X_i = \begin{cases} 1 & \text{if the } i\text{th game results in a win} \\ 0 & \text{otherwise} \end{cases}$$

then $X_i, i = 1, \ldots, n$ will be independent Bernoulli random variables for which

$$E[X_i] = P\{\text{win at solitaire}\}$$

Hence, by the strong law of large numbers, we know that

$$\sum_{i=1}^{n} \frac{X_i}{n} = \frac{\text{number of games won}}{\text{number of games played}}$$

will, with probability 1, converge to $P\{\text{win at solitaire}\}$. That is, by playing a large number of games, we can use the proportion of games won as an estimate of the probability of winning. This method of empirically determining probabilities by means of experimentation is known as *simulation*.

In order to use a computer to initiate a simulation study, we must be able to generate the value of a uniform (0, 1) random variable; such variates are called *random numbers*. To generate them, most computers have a built-in subroutine, called a *random-number generator*, whose output is a sequence of *pseudorandom numbers*—a sequence of numbers that is, for all practical purposes, indistinguishable from a sample from the uniform (0, 1) distribution. Most random-number generators start with an initial value X_0, called the *seed*, and then recursively compute values by specifying positive integers a, c, and m, and then letting

$$X_{n+1} = (aX_n + c) \text{ modulo } m \quad n \geq 0 \tag{1.1}$$

where the foregoing means that $aX_n + c$ is divided by m and the remainder is taken as the value of X_{n+1}. Thus, each X_n is either $0, 1, \ldots, m-1$, and the quantity X_n/m is taken as an approximation to a uniform (0, 1) random variable. It can be shown that, subject to suitable choices for a, c, and m, Equation (1.1) gives rise to a sequence of numbers that look as if they were generated from independent uniform (0, 1) random variables.

As our starting point in simulation, we shall suppose that we can simulate from the uniform (0, 1) distribution, and we shall use the term *random numbers* to mean independent random variables from this distribution.

In the solitaire example, we would need to program a computer to play out the game starting with a given ordering of the cards. However, since the initial ordering is supposed to be equally likely to be any of the (52)! possible permutations, it is also necessary to be able to generate a random permutation. Using only random numbers, the following algorithm shows how this can be accomplished. The algorithm begins by randomly choosing one of the elements and then putting it in position n; it then randomly chooses among the remaining elements and puts the choice in position $n-1$, and so on. The algorithm efficiently makes a random choice among the remaining elements by keeping these elements in an ordered list and then randomly choosing a position on that list.

EXAMPLE 1a Generating a random permutation

Suppose we are interested in generating a permutation of the integers $1, 2, \ldots, n$ such that all $n!$ possible orderings are equally likely. Then, starting with any initial permutation, we will accomplish this after $n-1$ steps, where we interchange the positions of two of the numbers of the permutation at each step. Throughout, we will keep track of the permutation by letting $X(i), i = 1, \ldots, n$ denote the number currently in position i. The algorithm operates as follows:

1. Consider any arbitrary permutation, and let $X(i)$ denote the element in position $i, i = 1 \ldots, n$. [For instance, we could take $X(i) = i, i = 1, \ldots, n$.]
2. Generate a random variable N_n that is equally likely to equal any of the values $1, 2, \ldots, n$.
3. Interchange the values of $X(N_n)$ and $X(n)$. The value of $X(n)$ will now remain fixed. [For instance, suppose that $n = 4$ and initially $X(i) = i, i = 1, 2, 3, 4$. If $N_4 = 3$, then the new permutation is $X(1) = 1, X(2) = 2, X(3) = 4, X(4) = 3$, and element 3 will remain in position 4 throughout.]
4. Generate a random variable N_{n-1} that is equally likely to be either $1, 2, \ldots, n-1$.
5. Interchange the values of $X(N_{n-1})$ and $X(n-1)$. [If $N_3 = 1$, then the new permutation is $X(1) = 4, X(2) = 2, X(3) = 1, X(4) = 3$.]

6. Generate N_{n-2}, which is equally likely to be either $1, 2, \ldots, n-2$.
7. Interchange the values of $X(N_{n-2})$ and $X(n-2)$. [If $N_2 = 1$, then the new permutation is $X(1) = 2, X(2) = 4, X(3) = 1, X(4) = 3$, and this is the final permutation.]
8. Generate N_{n-3}, and so on. The algorithm continues until N_2 is generated, and after the next interchange the resulting permutation is the final one.

To implement this algorithm, it is necessary to be able to generate a random variable that is equally likely to be any of the values $1, 2, \ldots, k$. To accomplish this, let U denote a random number—that is, U is uniformly distributed on $(0, 1)$—and note that kU is uniform on $(0, k)$. Hence,

$$P\{i - 1 < kU < i\} = \frac{1}{k} \quad i = 1, \ldots, k$$

so if we take $N_k = [kU] + 1$, where $[x]$ is the integer part of x (that is, the largest integer less than or equal to x), then N_k will have the desired distribution.

The algorithm can now be succinctly written as follows:

Step 1. Let $X(1), \ldots, X(n)$ be any permutation of $1, 2, \ldots, n$. [For instance, we can set $X(i) = i, i = 1, \ldots, n$.]
Step 2. Let $I = n$.
Step 3. Generate a random number U and set $N = [IU] + 1$.
Step 4. Interchange the values of $X(N)$ and $X(I)$.
Step 5. Reduce the value of I by 1, and if $I > 1$, go to step 3.
Step 6. $X(1), \ldots, X(n)$ is the desired random generated permutation.

The foregoing algorithm for generating a random permutation is extremely useful. For instance, suppose that a statistician is developing an experiment to compare the effects of m different treatments on a set of n subjects. He decides to split the subjects into m different groups of respective sizes $n_1, n_2, \ldots, n_m$, where $\sum_{i=1}^{m} n_i = n$, with the members of the ith group to receive treatment i. To eliminate any bias in the assignment of subjects to treatments (for instance, it would cloud the meaning of the experimental results if it turned out that all the "best" subjects had been put in the same group), it is imperative that the assignment of a subject to a given group be done "at random." How is this to be accomplished?[†]

A simple and efficient procedure is to arbitrarily number the subjects 1 through n and then generate a random permutation $X(1), \ldots, X(n)$ of $1, 2, \ldots, n$. Now assign subjects $X(1), X(2), \ldots, X(n_1)$ to be in group 1, $X(n_1 + 1), \ldots, X(n_1 + n_2)$ to be in group 2, and, in general, group j is to consist of subjects numbered $X(n_1 + n_2 + \cdots + n_{j-1} + k), k = 1, \ldots, n_j$. ■

10.2 GENERAL TECHNIQUES FOR SIMULATING CONTINUOUS RANDOM VARIABLES

In this section, we present two general methods for using random numbers to simulate continuous random variables.

[†]Another technique for randomly dividing the subjects when $m = 2$ was presented in Example 2g of Chapter 6. The preceding procedure is faster, but requires more space than the one of Example 2g.

10.2.1 The Inverse Transformation Method

A general method for simulating a random variable having a continuous distribution—called the *inverse transformation method*—is based on the following proposition.

Proposition 2.1. Let U be a uniform $(0, 1)$ random variable. For any continuous distribution function F, if we define the random variable Y by

$$Y = F^{-1}(U)$$

then the random variable Y has distribution function F. [$F^{-1}(x)$ is defined to equal that value y for which $F(y) = x$.]

Proof.

$$\begin{aligned} F_Y(a) &= P\{Y \leq a\} \\ &= P\{F^{-1}(U) \leq a\} \end{aligned} \tag{2.1}$$

Now, since $F(x)$ is a monotone function, it follows that $F^{-1}(U) \leq a$ if and only if $U \leq F(a)$. Hence, from Equation (2.1), we have

$$\begin{aligned} F_Y(a) &= P\{U \leq F(a)\} \\ &= F(a) \end{aligned}$$

□

It follows from Proposition 2.1 that we can simulate a random variable X having a continuous distribution function F by generating a random number U and then setting $X = F^{-1}(U)$.

EXAMPLE 2a Simulating an exponential random variable

If $F(x) = 1 - e^{-x}$, then $F^{-1}(u)$ is that value of x such that

$$1 - e^{-x} = u$$

or

$$x = -\log(1 - u)$$

Hence, if U is a uniform $(0, 1)$ variable, then

$$F^{-1}(U) = -\log(1 - U)$$

is exponentially distributed with mean 1. Since $1 - U$ is also uniformly distributed on $(0, 1)$, it follows that $-\log U$ is exponential with mean 1. Since cX is exponential with mean c when X is exponential with mean 1, it follows that $-c \log U$ is exponential with mean c. ■

The results of Example 2a can also be utilized to stimulate a gamma random variable.

EXAMPLE 2b Simulating a gamma (n, λ) random variable

To simulate from a gamma distribution with parameters (n, λ) when n is an integer, we use the fact that the sum of n independent exponential random variables, each having rate λ, has this distribution. Hence, if $U_1, \ldots, U_n$ are independent uniform $(0, 1)$ random variables, then

$$X = -\sum_{i=1}^{n} \frac{1}{\lambda} \log U_i = -\frac{1}{\lambda} \log \left(\prod_{i=1}^{n} U_i \right)$$

has the desired distribution. ■

10.2.2 The Rejection Method

Suppose that we have a method for simulating a random variable having density function $g(x)$. We can use this method as the basis for simulating from the continuous distribution having density $f(x)$ by simulating Y from g and then accepting the simulated value with a probability proportional to $f(Y)/g(Y)$.

Specifically, let c be a constant such that

$$\frac{f(y)}{g(y)} \leq c \quad \text{for all } y$$

We then have the following technique for simulating a random variable having density f.

Rejection Method

Step 1. Simulate Y having density g and simulate a random number U.
Step 2. If $U \leq f(Y)/cg(Y)$, set $X = Y$. Otherwise return to step 1.

The rejection method is expressed pictorially in Figure 10.1. We now prove that it works.

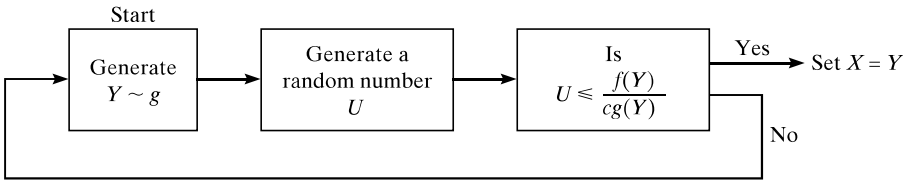

FIGURE 10.1: Rejection method for simulating a random variable X having density function f.

Proposition 2.2. The random variable X generated by the rejection method has density function f.

Proof. Let X be the value obtained and let N denote the number of necessary iterations. Then

$$\begin{aligned} P\{X \leq x\} &= P\{Y_N \leq x\} \\ &= P\left\{Y \leq x | U \leq \frac{f(Y)}{cg(Y)}\right\} \\ &= \frac{P\left\{Y \leq x, U \leq \frac{f(Y)}{cg(Y)}\right\}}{K} \end{aligned}$$

where $K = P\{U \leq f(Y)/cg(Y)\}$. Now, by independence, the joint density function of Y and U is

$$f(y,u) = g(y) \quad 0 < u < 1$$

so, using the foregoing, we have

$$\begin{aligned} P\{X \le x\} &= \frac{1}{K} \iint\limits_{\substack{y \le x \\ 0 \le u \le f(y)/cg(y)}} g(y)\, du\, dy \\ &= \frac{1}{K} \int_{-\infty}^{x} \int_{0}^{f(y)/cg(y)} du\, g(y)\, dy \\ &= \frac{1}{cK} \int_{-\infty}^{x} f(y)\, dy \end{aligned} \tag{2.2}$$

Letting X approach ∞ and using the fact that f is a density gives

$$1 = \frac{1}{cK} \int_{-\infty}^{\infty} f(y)\, dy = \frac{1}{cK}$$

Hence, from Equation (2.2), we obtain

$$P\{X \le x\} = \int_{-\infty}^{x} f(y)\, dy$$

which completes the proof. □

Remarks. (a) Note that the way in which we "accept the value Y with probability $f(Y)/cg(Y)$" is by generating a random number U and then accepting Y if $U \le f(Y)/cg(Y)$.

(b) Since each iteration will independently result in an accepted value with probability $P\{U \le f(Y)/cg(Y)\} = K = 1/c$, it follows that the number of iterations has a geometric distribution with mean c. ■

EXAMPLE 2c Simulating a normal random variable

To simulate a unit normal random variable Z (that is, one with mean 0 and variance 1), note first that the absolute value of Z has probability density function

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2} \quad 0 < x < \infty \tag{2.3}$$

We will start by simulating from the preceding density function by using the rejection method, with g being the exponential density function with mean 1—that is,

$$g(x) = e^{-x} \quad 0 < x < \infty$$

Now, note that

$$\begin{aligned} \frac{f(x)}{g(x)} &= \sqrt{\frac{2}{\pi}} \exp\left\{\frac{-(x^2 - 2x)}{2}\right\} \\ &= \sqrt{\frac{2}{\pi}} \exp\left\{\frac{-(x^2 - 2x + 1)}{2} + \frac{1}{2}\right\} \\ &= \sqrt{\frac{2e}{\pi}} \exp\left\{\frac{-(x-1)^2}{2}\right\} \\ &\le \sqrt{\frac{2e}{\pi}} \end{aligned} \tag{2.4}$$

Hence, we can take $c = \sqrt{2e/\pi}$; so, from Equation (2.4),

$$\frac{f(x)}{cg(x)} = \exp\left\{\frac{-(x-1)^2}{2}\right\}$$

Therefore, using the rejection method, we can simulate the absolute value of a unit normal random variable as follows:

(a) Generate independent random variables Y and U, Y being exponential with rate 1 and U being uniform on $(0, 1)$.

(b) If $U \leq \exp\{-(Y-1)^2/2\}$, set $X = Y$. Otherwise, return to (a).

Once we have simulated a random variable X having Equation (2.3) as its density function, we can then generate a unit normal random variable Z by letting Z be equally likely to be either X or $-X$.

In step (b), the value Y is accepted if $U \leq \exp\{-(Y-1)^2/2\}$, which is equivalent to $-\log U \geq (Y-1)^2/2$. However, in Example 2a it was shown that $-\log U$ is exponential with rate 1, so steps (a) and (b) are equivalent to

(a′) Generate independent exponentials Y_1 and Y_2, each with rate 1.

(b′) If $Y_2 \geq (Y_1 - 1)^2/2$, set $X = Y_1$. Otherwise, return to (a').

Suppose now that the foregoing results in Y_1's being accepted—so we know that Y_2 is larger than $(Y_1 - 1)^2/2$. By how much does the one exceed the other? To answer this question, recall that Y_2 is exponential with rate 1; hence, given that it exceeds some value, the amount by which Y_2 exceeds $(Y_1 - 1)^2/2$ [that is, its "additional life" beyond the time $(Y_1 - 1)^2/2$] is (by the memoryless property) also exponentially distributed with rate 1. That is, when we accept step (b′), not only do we obtain X (the absolute value of a unit normal), but, by computing $Y_2 - (Y_1 - 1)^2/2$, we also can generate an exponential random variable (that is independent of X) having rate 1.

Summing up, then, we have the following algorithm that generates an exponential with rate 1 and an independent unit normal random variable:

Step 1. Generate Y_1, an exponential random variable with rate 1.
Step 2. Generate Y_2, an exponential random variable with rate 1.
Step 3. If $Y_2 - (Y_1 - 1)^2/2 > 0$, set $Y = Y_2 - (Y_1 - 1)^2/2$ and go to step 4. Otherwise, go to step 1.
Step 4. Generate a random number U, and set

$$Z = \begin{cases} Y_1 & \text{if } U \leq \frac{1}{2} \\ -Y_1 & \text{if } U > \frac{1}{2} \end{cases}$$

The random variables Z and Y generated by the foregoing algorithm are independent, with Z being normal with mean 0 and variance 1 and Y being exponential with rate 1. (If we want the normal random variable to have mean μ and variance σ^2, we just take $\mu + \sigma Z$.)

Remarks. (a) Since $c = \sqrt{2e/\pi} \approx 1.32$, the algorithm requires a geometrically distributed number of iterations of step 2 with mean 1.32.

(b) If we want to generate a sequence of unit normal random variables, then we can use the exponential random variable Y obtained in step 3 as the initial exponential needed in step 1 for the next normal to be generated. Hence, on the average, we can simulate a unit normal by generating $1.64 (= 2 \times 1.32 - 1)$ exponentials and computing 1.32 squares. ■

EXAMPLE 2d Simulating normal random variables: the polar method

It was shown in Example 7b of Chapter 6 that if X and Y are independent unit normal random variables, then their polar coordinates $R = \sqrt{X^2 + Y^2}, \Theta = \tan^{-1}(Y/X)$ are independent, with R^2 being exponentially distributed with mean 2 and Θ being uniformly distributed on $(0, 2\pi)$. Hence, if U_1 and U_2 are random numbers, then, using the result of Example 2a, we can set

$$R = (-2 \log U_1)^{1/2}$$
$$\Theta = 2\pi U_2$$

from which it follows that

$$X = R \cos \Theta = (-2 \log U_1)^{1/2} \cos(2\pi U_2)$$
$$Y = R \sin \Theta = (-2 \log U_1)^{1/2} \sin(2\pi U_2) \tag{2.5}$$

are independent unit normals. ■

The preceding approach to generating unit normal random variables is called the *Box–Muller approach*. Its efficiency suffers somewhat from its need to compute the sine and cosine values. There is, however, a way to get around this potentially time-consuming difficulty. To begin, note that if U is uniform on $(0, 1)$ then $2U$ is uniform on $(0, 2)$, so $2U - 1$ is uniform on $(-1, 1)$. Thus, if we generate random numbers U_1 and U_2 and set

$$V_1 = 2U_1 - 1$$
$$V_2 = 2U_2 - 1$$

then (V_1, V_2) is uniformly distributed in the square of area 4 centered at $(0, 0)$. (See Figure 10.2.)

Suppose now that we continually generate such pairs (V_1, V_2) until we obtain one that is contained in the disk of radius 1 centered at $(0, 0)$—that is, until $V_1^2 + V_2^2 \leq 1$. It then follows that such a pair (V_1, V_2) is uniformly distributed in the disk. Now, let

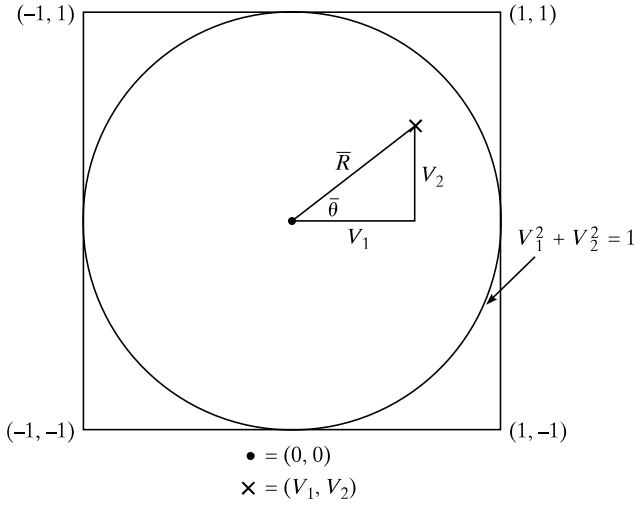

FIGURE 10.2:

$\overline{R}, \overline{\Theta}$ denote the polar coordinates of this pair. Then it is easy to verify that $\overline{R}$ and $\overline{\Theta}$ are independent, with $\overline{R}^2$ being uniformly distributed on $(0, 1)$ and $\overline{\Theta}$ being uniformly distributed on $(0, 2\pi)$. (See Problem 13.)

Since

$$\sin\overline{\Theta} = \frac{V_2}{\overline{R}} = \frac{V_2}{\sqrt{V_1^2 + V_2^2}}$$
$$\cos\overline{\Theta} = \frac{V_1}{\overline{R}} = \frac{V_1}{\sqrt{V_1^2 + V_2^2}}$$

it follows from Equation (2.5) that we can generate independent unit normals X and Y by generating another random number U and setting

$$X = (-2\log U)^{1/2} V_1/\overline{R}$$
$$Y = (-2\log U)^{1/2} V_2/\overline{R}$$

In fact, because (conditional on $V_1^2 + V_2^2 \leq 1$) $\overline{R}^2$ is uniform on $(0, 1)$ and is independent of $\overline{\theta}$, we can use it instead of generating a new random number U, thus showing that

$$X = (-2\log\overline{R}^2)^{1/2}\frac{V_1}{\overline{R}} = \sqrt{\frac{-2\log S}{S}}V_1$$
$$Y = (-2\log\overline{R}^2)^{1/2}\frac{V_2}{\overline{R}} = \sqrt{\frac{-2\log S}{S}}V_2$$

are independent unit normals, where

$$S = \overline{R}^2 = V_1^2 + V_2^2$$

Summing up, we have the following approach to generating a pair of independent unit normals:

Step 1. Generate random numbers U_1 and U_2.
Step 2. Set $V_1 = 2U_1 - 1, V_2 = 2U_2 - 1, S = V_1^2 + V_2^2$.
Step 3. If $S > 1$, return to step 1.
Step 4. Return the independent unit normals

$$X = \sqrt{\frac{-2\log S}{S}}V_1, Y = \sqrt{\frac{-2\log S}{S}}V_2$$

The preceding algorithm is called the *polar method*. Since the probability that a random point in the square will fall within the circle is equal to $\pi/4$ (the area of the circle divided by the area of the square), it follows that, on average, the polar method will require $4/\pi \approx 1.273$ iterations of step 1. Hence, it will, on average, require 2.546 random numbers, 1 logarithm, 1 square root, 1 division, and 4.546 multiplications to generate 2 independent unit normals.

EXAMPLE 2e Simulating a chi-squared random variable

The chi-squared distribution with n degrees of freedom is the distribution of $\chi_n^2 = Z_1^2 + \cdots + Z_n^2$, where $Z_i, i = 1, \ldots, n$ are independent unit normals. Now, it was

shown in Section 6.3 of Chapter 6 that $Z_1^2 + Z_2^2$ has an exponential distribution with rate $\frac{1}{2}$. Hence, when n is even (say, $n = 2k$), χ_{2k}^2 has a gamma distribution with parameters $\left(k, \frac{1}{2}\right)$. Thus, $-2\log(\prod_{i=1}^{k} U_i)$ has a chi-squared distribution with $2k$ degrees of freedom. Accordingly, can simulate a chi-squared random variable with $2k + 1$ degrees of freedom by first simulating a unit normal random variable Z and then adding Z^2 to the foregoing. That is,

$$\chi_{2k+1}^2 = Z^2 - 2\log\left(\prod_{i=1}^{k} U_i\right)$$

where $Z, U_1, \ldots, U_n$ are independent, Z is a unit normal, and $U_1, \ldots, U_n$ are uniform $(0, 1)$ random variables.

10.3 SIMULATING FROM DISCRETE DISTRIBUTIONS

All of the general methods for simulating random variables from continuous distributions have analogs in the discrete case. For instance, if we want to simulate a random variable Z having probability mass function

$$P\{X = x_j\} = P_j, \quad j = 0, 1, \ldots, \quad \sum_j P_j = 1$$

we can use the following discrete time analog of the inverse transform technique:

To simulate X for which $P\{X = x_j\} = P_j$, let U be uniformly distributed over $(0, 1)$ and set

$$X = \begin{cases} x_1 & \text{if } U \le P_1 \\ x_2 & \text{if } P_1 < U \le P_1 + P_2 \\ \cdot & \\ \cdot & \\ \cdot & \\ x_j & \text{if } \sum_1^{j-1} P_i < U \le \sum_i^{j} P_i \\ \cdot & \\ \cdot & \\ \cdot & \end{cases}$$

Since

$$P\{X = x_j\} = P\left\{\sum_1^{j-1} P_i < U \le \sum_1^{j} P_i\right\} = P_j$$

it follows that X has the desired distribution.

EXAMPLE 3a The geometric distribution

Suppose that independent trials, each of which results in a "success" with probability $p, 0 < p < 1$, are continually performed until a success occurs. Letting X denote the necessary number of trials; then

$$P\{X = i\} = (1 - p)^{i-1}p \quad i \ge 1$$

which is seen by noting that $X = i$ if the first $i - 1$ trials are all failures and the ith trial is a success. The random variable X is said to be a *geometric random variable with parameter p*. Since

$$\begin{aligned}\sum_{i=1}^{j-1} P\{X = i\} &= 1 - P\{X > j - 1\} \\ &= 1 - P\{\text{first } j - 1 \text{ are all failures}\} \\ &= 1 - (1 - p)^{j-1} \quad j \geq 1\end{aligned}$$

we can simulate such a random variable by generating a random number U and then setting X equal to that value j for which

$$1 - (1 - p)^{j-1} < U \leq 1 - (1 - p)^j$$

or, equivalently, for which

$$(1 - p)^j \leq 1 - U < (1 - p)^{j-1}$$

Since $1 - U$ has the same distribution as U, we can define X by

$$\begin{aligned}X &= \min\{j : (1 - p)^j \leq U\} \\ &= \min\{j : j \log(1 - p) \leq \log U\} \\ &= \min\left\{j : j \geq \frac{\log U}{\log(1 - p)}\right\}\end{aligned}$$

where the inequality has changed sign because $\log(1 - p)$ is negative [since $\log(1 - p) < \log 1 = 0$]. Using the notation $[x]$ for the integer part of x (that is, $[x]$ is the largest integer less than or equal to x), we can write

$$X = 1 + \left[\frac{\log U}{\log(1 - p)}\right]$$ ■

As in the continuous case, special simulating techniques have been developed for the more common discrete distributions. We now present two of these.

EXAMPLE 3b Simulating a binomial random variable

A binomial (n, p) random variable can easily be simulated by recalling that it can be expressed as the sum of n independent Bernoulli random variables. That is, if $U_1, \ldots, U_n$ are independent uniform $(0, 1)$ variables, then letting

$$X_i = \begin{cases} 1 & \text{if } U_i < p \\ 0 & \text{otherwise} \end{cases}$$

it follows that $X \equiv \sum_{i=1}^{n} X_i$ is a binomial random variable with parameters n and p.

EXAMPLE 3c Simulating a Poisson random variable

To simulate a Poisson random variable with mean λ, generate independent uniform $(0, 1)$ random variables $U_1, U_2, \ldots$ stopping at

$$N = \min\left\{n: \prod_{i=1}^{n} U_i < e^{-\lambda}\right\}$$

The random variable $X \equiv N - 1$ has the desired distribution. That is, if we continue generating random numbers until their product falls below $e^{-\lambda}$, then the number required, minus 1, is Poisson with mean λ.

That $X \equiv N - 1$ is indeed a Poisson random variable having mean λ can perhaps be most easily seen by noting that

$$X + 1 = \min\left\{n: \prod_{i=1}^{n} U_i < e^{-\lambda}\right\}$$

is equivalent to

$$X = \max\left\{n: \prod_{i=1}^{n} U_i \geq e^{-\lambda}\right\} \quad \text{where } \prod_{i=1}^{0} U_i \equiv 1$$

or, taking logarithms, to

$$X = \max\left\{n: \sum_{i=1}^{n} \log U_i \geq -\lambda\right\}$$

or

$$X = \max\left\{n: \sum_{i=1}^{n} -\log U_i \leq \lambda\right\}$$

However, $-\log U_i$ is exponential with rate 1, so X can be thought of as being the maximum number of exponentials having rate 1 that can be summed and still be less than λ. But by recalling that the times between successive events of a Poisson process having rate 1 are independent exponentials with rate 1, it follows that X is equal to the number of events by time λ of a Poisson process having rate 1; thus X has a Poisson distribution with mean λ. ■

10.4 VARIANCE REDUCTION TECHNIQUES

Let $X_1, \ldots, X_n$ have a given joint distribution, and suppose that we are interested in computing

$$\theta \equiv E[g(X_1, \ldots, X_n)]$$

where g is some specified function. It sometimes turns out that it is extremely difficult to analytically compute θ, and when such is the case, we can attempt to use simulation to estimate θ. This is done as follows: Generate $X_1^{(1)}, \ldots, X_n^{(1)}$ having the same joint distribution as $X_1, \ldots, X_n$ and set

$$Y_1 = g(X_1^{(1)}, \ldots, X_n^{(1)})$$

Now let $X_1^{(2)}, \ldots, X_n^{(2)}$ simulate a second set of random variables (independent of the first set) having the distribution of $X_1, \ldots, X_n$ and set

$$Y_2 = g(X_1^{(2)}, \ldots, X_n^{(2)})$$

Continue this until you have generated k (some predetermined number) sets and so have also computed $Y_1, Y_2, \ldots, Y_k$. Now, $Y_1, \ldots, Y_k$ are independent and identically distributed random variables, each having the same distribution as $g(X_1, \ldots, X_n)$. Thus, if we let $\overline{Y}$ denote the average of these k random variables—that is, if

$$\overline{Y} = \sum_{i=1}^{k} \frac{Y_i}{k}$$

then

$$E[\overline{Y}] = \theta$$
$$E[(\overline{Y} - \theta)^2] = \text{Var}(\overline{Y})$$

Hence, we can use $\overline{Y}$ as an estimate of θ. Since the expected square of the difference between $\overline{Y}$ and θ is equal to the variance of $\overline{Y}$, we would like this quantity to be as small as possible. [In the preceding situation, $\text{Var}(\overline{Y}) = \text{Var}(Y_i)/k$, which is usually not known in advance, but must be estimated from the generated values $Y_1, \ldots, Y_n$.] We now present three general techniques for reducing the variance of our estimator.

10.4.1 Use of Antithetic Variables

In the foregoing situation, suppose that we have generated Y_1 and Y_2, which are identically distributed random variables having mean θ. Now,

$$\text{Var}\left(\frac{Y_1 + Y_2}{2}\right) = \frac{1}{4}[\text{Var}(Y_1) + \text{Var}(Y_2) + 2\text{Cov}(Y_1, Y_2)]$$
$$= \frac{\text{Var}(Y_1)}{2} + \frac{\text{Cov}(Y_1, Y_2)}{2}$$

Hence, it would be advantageous (in the sense that the variance would be reduced) if Y_1 and Y_2 were negatively correlated rather than being independent. To see how we could arrange this, let us suppose that the random variables $X_1, \ldots, X_n$ are independent and, in addition, that each is simulated via the inverse transform technique. That is, X_i is simulated from $F_i^{-1}(U_i)$, where U_i is a random number and F_i is the distribution of X_i. Thus, Y_1 can be expressed as

$$Y_1 = g(F_1^{-1}(U_1), \ldots, F_n^{-1}(U_n))$$

Now, since $1 - U$ is also uniform over $(0, 1)$ whenever U is a random number (and is negatively correlated with U), it follows that Y_2 defined by

$$Y_2 = g(F_1^{-1}(1 - U_1), \ldots, F_n^{-1}(1 - U_n))$$

will have the same distribution as Y_1. Hence, if Y_1 and Y_2 were negatively correlated, then generating Y_2 by this means would lead to a smaller variance than if it were generated by a new set of random numbers. (In addition, there is a computational savings because, rather than having to generate n additional random numbers, we need only subtract each of the previous n numbers from 1.) Although we cannot, in

general, be certain that Y_1 and Y_2 will be negatively correlated, this often turns out to be the case, and indeed it can be proven that it will be so whenever g is a monotonic function.

10.4.2 Variance Reduction by Conditioning

Let us start by recalling the conditional variance formula (see Section 7.5.4)

$$\text{Var}(Y) = E[\text{Var}(Y|Z)] + \text{Var}(E[Y|Z])$$

Now, suppose that we are interested in estimating $E[g(X_1, \ldots, X_n)]$ by simulating $\mathbf{X} = (X_1, \ldots, X_n)$ and then computing $Y = g(\mathbf{X})$. If, for some random variable Z we can compute $E[Y|Z]$, then, since $\text{Var}(Y|Z) \geq 0$, it follows from the preceding conditional variance formula that

$$\text{Var}(E[Y|Z]) \leq \text{Var}(Y)$$

Thus, since $E[E[Y|Z]] = E[Y]$, it follows that $E[Y|Z]$ is a better estimator of $E[Y]$ than is Y.

EXAMPLE 4a Estimation of π

Let U_1 and U_2 be random numbers and set $V_i = 2U_i - 1, i = 1,2$. As noted in Example 2d,(V_1, V_2) will be uniformly distributed in the square of area 4 centered at (0, 0). The probability that this point will fall within the inscribed circle of radius 1 centered at (0, 0) (see Figure 10.2) is equal to $\pi/4$ (the ratio of the area of the circle to that of the square). Hence, upon simulating a large number n of such pairs and setting

$$I_j = \begin{cases} 1 & \text{if the } j\text{th pair falls within the circle} \\ 0 & \text{otherwise} \end{cases}$$

it follows that $I_j, j = 1, \ldots, n$, will be independent and identically distributed random variables having $E[I_j] = \pi/4$. Thus, by the strong law of large numbers,

$$\frac{I_1 + \cdots + I_n}{n} \to \frac{\pi}{4} \quad \text{as } n \to \infty$$

Therefore, by simulating a large number of pairs (V_1, V_2) and multiplying the proportion of them that fall within the circle by 4, we can accurately approximate π.

The preceding estimator can, however, be improved upon by using conditional expectation. If we let I be the indicator variable for the pair (V_1, V_2), then, rather than using the observed value of I, it is better to condition on V_1 and so utilize

$$\begin{aligned} E[I|V_1] &= P\{V_1^2 + V_2^2 \leq 1|V_1\} \\ &= P\{V_2^2 \leq 1 - V_1^2|V_1\} \end{aligned}$$

Now,

$$\begin{aligned} P\{V_2^2 \leq 1 - V_1^2|V_1 = v\} &= P\{V_2^2 \leq 1 - v^2\} \\ &= P\{-\sqrt{1 - v^2} \leq V_2 \leq \sqrt{1 - v^2}\} \\ &= \sqrt{1 - v^2} \end{aligned}$$

so

$$E[I|V_1] = \sqrt{1 - V_1^2}$$

Thus, an improvement on using the average value of I to estimate $\pi/4$ is to use the average value of $\sqrt{1 - V_1^2}$. Indeed, since

$$E[\sqrt{1 - V_1^2}] = \int_{-1}^{1} \frac{1}{2}\sqrt{1 - v^2}dv = \int_0^1 \sqrt{1 - u^2}du = E[\sqrt{1 - U^2}]$$

where U is uniform over $(0, 1)$, we can generate n random numbers U and use the average value of $\sqrt{1 - U^2}$ as our estimate of $\pi/4$. (Problem 14 shows that this estimator has the same variance as the average of the n values, $\sqrt{1 - V^2}$.)

The preceding estimator of π can be improved even further by noting that the function $g(u) = \sqrt{1 - u^2}, 0 \leq u \leq 1$, is a monotonically decreasing function of u, and so the method of antithetic variables will reduce the variance of the estimator of $E[\sqrt{1 - U^2}]$. That is, rather than generating n random numbers and using the average value of $\sqrt{1 - U^2}$ as an estimator of $\pi/4$, we would obtain an improved estimator by generating only $n/2$ random numbers U and then using one-half the average of $\sqrt{1 - U^2} + \sqrt{1 - (1 - U)^2}$ as the estimator of $\pi/4$.

The following table gives the estimates of π resulting from simulations, using $n = 10,000$, based on the three estimators.

Method	Estimate of π
Proportion of the random points that fall in the circle	3.1612
Average value of $\sqrt{1 - U^2}$	3.128448
Average value of $\sqrt{1 - U^2} + \sqrt{1 - (1 - U)^2}$	3.139578

A further simulation using the final approach and $n = 64,000$ yielded the estimate 3.143288. ■

10.4.3 Control Variates

Again, suppose that we want to use simulation to estimate $E[g(\mathbf{X})]$, where $\mathbf{X} = (X_1, \ldots, X_n)$. But suppose now that, for some function f, the expected value of $f(\mathbf{X})$ is known—say, it is $E[f(\mathbf{X})] = \mu$. Then, for any constant a, we can also use

$$W = g(\mathbf{X}) + a[f(\mathbf{X}) - \mu]$$

as an estimator of $E[g(\mathbf{X})]$. Now,

$$\text{Var}(W) = \text{Var}[g(\mathbf{X})] + a^2\text{Var}[f(\mathbf{X})] + 2a\,\text{Cov}[g(\mathbf{X}), f(\mathbf{X})] \tag{4.1}$$

Simple calculus shows that the foregoing is minimized when

$$a = \frac{-\text{Cov}[f(\mathbf{X}), g(\mathbf{X})]}{\text{Var}[f(\mathbf{X})]} \tag{4.2}$$

and for this value of a,

$$\text{Var}(W) = \text{Var}[g(\mathbf{X})] - \frac{[\text{Cov}[f(\mathbf{X}), g(\mathbf{X})]^2}{\text{Var}[f(\mathbf{X})]} \tag{4.3}$$

Unfortunately, neither $\text{Var}[f(\mathbf{X})]$ nor $\text{Cov}[f(\mathbf{X})], g(\mathbf{X})]$ is usually known, so we cannot in general obtain the foregoing reduction in variance. One approach in practice is to use the simulated data to estimate these quantities. This approach usually yields almost all of the theoretically possible reduction in variance.

SUMMARY

Let F be a continuous distribution function and U a uniform (0, 1) random variable. Then the random variable $F^{-1}(U)$ has distribution function F, where $F^{-1}(u)$ is that value x such that $F(x) = u$. Applying this result, we can use the values of uniform (0, 1) random variables, called *random numbers*, to generate the values of other random variables. This technique is called the *inverse transform* method.

Another technique for generating random variables is based on the *rejection* method. Suppose that we have an efficient procedure for generating a random variable from the density function g and that we desire to generate a random variable having density function f. The rejection method for accomplishing this starts by determining a constant c such that

$$\max \frac{f(x)}{g(x)} \leq c$$

It then proceeds as follows:

1. Generate Y having density g.
2. Generate a random number U.
3. If $U \leq f(Y)/cg(Y)$, set $X = Y$ and stop.
4. Return to step 1.

The number of passes through step 1 is a geometric random variable with mean c.

Standard normal random variables can be efficiently simulated by the rejection method (with g being exponential with mean 1) or by the technique known as the *polar algorithm*.

To estimate a quantity θ, one often generates the values of a partial sequence of random variables whose expected value is θ. The efficiency of this approach is increased when these random variables have a small variance. Three techniques that can often be used to specify random variables with mean θ and relatively small variances are

1. the use of antithetic variables,
2. the use of conditional expectations, and
3. the use of control variates.

PROBLEMS

10.1. The following algorithm will generate a random permutation of the elements $1, 2, \ldots, n$. It is somewhat faster than the one presented in Example 1a but is such that no position is fixed until the algorithm ends. In this algorithm, $P(i)$ can be interpreted as the element in position i.

Step 1. Set $k = 1$.

Step 2. Set $P(1) = 1$.

Step 3. If $k = n$, stop. Otherwise, let $k = k + 1$.

Step 4. Generate a random number U and let

$$P(k) = P([kU] + 1)$$
$$P([kU] + 1) = k$$

Go to step 3.

(a) Explain in words what the algorithm is doing.

(b) Show that at iteration k—that is, when the value of $P(k)$ is initially set—$P(1), P(2), \ldots, P(k)$ is a random permutation of $1, 2, \ldots, k$.

Hint: Use induction and argue that

$$\begin{aligned} P_k\{i_1, i_2, \ldots, i_{j-1}, k, i_j, \ldots, i_{k-2}, i\} &= P_{k-1}\{i_1, i_2, \ldots, i_{j-1}, i, i_j, \ldots, i_{k-2}\}\frac{1}{k} \\ &= \frac{1}{k!} \text{ by the induction hypothesis} \end{aligned}$$

10.2. Develop a technique for simulating a random variable having density function

$$f(x) = \begin{cases} e^{2x} & -\infty < x < 0 \\ e^{-2x} & 0 < x < \infty \end{cases}$$

10.3. Give a technique for simulating a random variable having the probability density function

$$f(x) = \begin{cases} \frac{1}{2}(x - 2) & 2 \le x \le 3 \\ \frac{1}{2}\left(2 - \frac{x}{3}\right) & 3 < x \le 6 \\ 0 & \text{otherwise} \end{cases}$$

10.4. Present a method for simulating a random variable having distribution function

$$F(x) = \begin{cases} 0 & x \le -3 \\ \frac{1}{2} + \frac{x}{6} & -3 < x < 0 \\ \frac{1}{2} + \frac{x^2}{32} & 0 < x \le 4 \\ 1 & x > 4 \end{cases}$$

10.5. Use the inverse transformation method to present an approach for generating a random variable from the Weibull distribution

$$F(t) = 1 - e^{-at^\beta} \quad t \ge 0$$

10.6. Give a method for simulating a random variable having failure rate function

(a) $\lambda(t) = c$;
(b) $\lambda(t) = ct$;
(c) $\lambda(t) = ct^2$;
(d) $\lambda(t) = ct^3$.

10.7. Let F be the distribution function

$$F(x) = x^n \quad 0 < x < 1$$

(a) Give a method for simulating a random variable having distribution F that uses only a single random number.
(b) Let $U_1, \ldots, U_n$ be independent random numbers. Show that

$$P\{\max(U_1, \ldots, U_n) \le x\} = x^n$$

(c) Use part (b) to give a second method of simulating a random variable having distribution F.

10.8. Suppose it is relatively easy to simulate from F_i for each $i = 1, \ldots, n$. How can we simulate from

(a) $F(x) = \prod_{i=1}^{n} F_i(x)$?

(b) $F(x) = 1 - \prod_{i=1}^{n}[1 - F_i(x)]$?

10.9. Suppose we have a method for simulating random variables from the distributions F_1 and F_2. Explain how to simulate from the distribution

$$F(x) = pF_1(x) + (1 - p)F_2(x) \quad 0 < p < 1$$

Give a method for simulating from

$$F(x) = \begin{cases} \frac{1}{3}(1 - e^{-3x}) + \frac{2}{3}x & 0 < x \le 1 \\ \frac{1}{3}(1 - e^{-3x}) + \frac{2}{3} & x > 1 \end{cases}$$

10.10. In Example 2c we simulated the absolute value of a unit normal by using the rejection procedure on exponential random variables with rate 1. This raises the question of whether we could obtain a more efficient algorithm by using a different exponential density—that is, we could use the density $g(x) = \lambda e^{-\lambda x}$. Show that the mean number of iterations needed in the rejection scheme is minimized when $\lambda = 1$.

10.11. Use the rejection method with $g(x) = 1, 0 < x < 1$, to determine an algorithm for simulating a random variable having density function

$$f(x) = \begin{cases} 60x^3(1 - x)^2 & 0 < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

10.12. Explain how you could use random numbers to approximate $\int_0^1 k(x)\,dx$, where $k(x)$ is an arbitrary function.
Hint: If U is uniform on $(0, 1)$, what is $E[k(U)]$?

10.13. Let (X, Y) be uniformly distributed in the circle of radius 1 centered at the origin. Its joint density is thus

$$f(x, y) = \frac{1}{\pi} \quad 0 \le x^2 + y^2 \le 1$$

Let $R = (X^2 + Y^2)^{1/2}$ and $\theta = \tan^{-1}(Y/X)$ denote the polar coordinates of (X, Y). Show that R and θ are independent, with R^2 being uniform on $(0, 1)$ and θ being uniform on $(0, 2\pi)$.

10.14. In Example 4a, we showed that

$$E[(1 - V^2)^{1/2}] = E[(1 - U^2)^{1/2}] = \frac{\pi}{4}$$

when V is uniform $(-1, 1)$ and U is uniform $(0, 1)$. Now show that

$$\text{Var}[(1 - V^2)^{1/2}] = \text{Var}[(1 - U^2)^{1/2}]$$

and find their common value.

10.15. **(a)** Verify that the minimum of (4.1) occurs when a is as given by (4.2).

(b) Verify that the minimum of (4.1) is given by (4.3).

10.16. Let X be a random variable on $(0, 1)$ whose density is $f(x)$. Show that we can estimate $\int_0^1 g(x)\,dx$ by simulating X and then taking $g(X)/f(X)$ as our estimate. This method, called *importance sampling*, tries to choose f similar in shape to g, so that $g(X)/f(X)$ has a small variance.

SELF-TEST PROBLEMS AND EXERCISES

10.1. The random variable X has probability density function

$$f(x) = Ce^x \quad 0 < x < 1$$

(a) Find the value of the constant C.

(b) Give a method for simulating such a random variable.

10.2. Give an approach for simulating a random variable having probability density function

$$f(x) = 30(x^2 - 2x^3 + x^4) \quad 0 < x < 1$$

10.3. Give an efficient algorithm to simulate the value of a random variable with probability mass function

$$p_1 = .15 \quad p_2 = .2 \quad p_3 = .35 \quad p_4 = .30$$

10.4. If X is a normal random variable with mean μ and variance σ^2, define a random variable Y that has the same distribution as X and is negatively correlated with it.

10.5. Let X and Y be independent exponential random variables with mean 1.

(a) Explain how we could use simulation to estimate $E[e^{XY}]$.

(b) Show how to improve the estimation approach in part (a) by using a control variate.

REFERENCE

[1] Ross, S. M. *Simulation*. 4th ed. San Diego: Academic Press, Inc., 2006.