Chapter 12

Introduction to Simulation Using MATLAB

12.1 Analysis versus Computer Simulation

A computer simulation is a computer program which attempts to represent the real world based on a model. The accuracy of the simulation depends on the precision of the model. Suppose that the probability of heads in a coin toss experiment is unknown. We can perform the experiment of tossing the coin n times repetitively to approximate the probability of heads.

$$P(H) = \frac{\text{Number of times heads observed}}{\text{Number of times the experiment executed}}$$

However, for many practical problems it is not possible to determine the probabilities by executing experiments a large number of times. With today's computers processing capabilities, we only need a high-level language, such as MATLAB, which can generate random numbers, to deal with these problems.

In this chapter, we present basic methods of generating random variables and simulate probabilistic systems. The provided algorithms are general and can be implemented in any computer language. However, to have concrete examples, we provide the actual codes in MATLAB. If you are unfamiliar with MATLAB, you should still be able to understand the algorithms.

12.2 Introduction: What is MATLAB?

MATLAB is a high-level language that helps engineers and scientists find solutions for given problems with fewer lines of codes than traditional programming languages, such as C/C++ or Java, by utilizing built-in math functions. You can use MATLAB for many applications including signal processing and communications, finance, and biology. Arrays are the basic data structure in MATLAB. Therefore, a basic knowledge of linear algebra is useful to use MATLAB in an effective way. Here we assume you are familiar with basic commands of MATLAB. We can use the built-in commands to generate probability distributions in MATLAB, but in this chapter we will also learn how to generate these distributions from the uniform distribution.

12.3 Discrete and Continuous Random Number Generators

Most of the programming languages can deliver samples from the uniform distribution to us (In reality, the given values are pseudo-random instead of being completely random.) The rest of this section shows how to convert uniform random variables to any other desired random variable. The MATLAB code for generating uniform random variables is:

$$U = rand;$$

which returns a pseudorandom value drawn from the standard uniform distribution on the open interval (0,1). Also,

$$U = rand(m, n);$$

returns an m-by-n matrix containing independent pseudorandom values drawn from the standard uniform distribution on the open interval (0,1).

12.3.1 Generating Discrete Probability Distributions from Uniform Distribution

Let's see a few examples of generating certain simple distributions:

Example 1. (Bernoulli) Simulate tossing a coin with probability of heads p.

Solution: Let U be a Uniform(0,1) random variable. We can write Bernoulli random variable X as:

$$X = \begin{cases} 1 & U < p \\ 0 & U \geq p \end{cases}$$

Thus,

$$\begin{aligned} P(H) &= P(X = 1) \\ &= P(U < p) \\ &= p \end{aligned}$$

Therefore, X has $Bernoulli(p)$ distribution. The MATLAB code for $Bernoulli(0.5)$ is:

$$\begin{aligned} &p = 0.5; \\ &U = rand; \\ &X = (U < p); \end{aligned}$$

Since the "rand" command returns a number between 0 and 1, we divided the interval $[0, 1]$ into two parts, p and $1 - p$ in length. Then, the value of X is determined based on where the number generated from uniform distribution fell.

Example 2. (Coin Toss Simulation) Write codes to simulate tossing a fair coin to see how the law of large numbers works.

Solution: You can write:

```
$n = 1000;$
$U = rand(1, n);$
$toss = (U < 0.5);$
$a = zeros(n + 1);$
$avg = zeros(n);$
$for \quad i = 2 : n + 1$
$a(i) = a(i - 1) + toss(i - 1);$
$avg(i - 1) = a(i)/(i - 1);$
$end$
$plot(avg)$
```

If you run the above codes to compute the proportion of ones in the variable "toss," the result will look like Figure 12.5. You can also assume the coin is unbiased with probability of heads equal to 0.6 by replacing the third line of the previous code with:

```
$toss = (U < 0.6);$
```

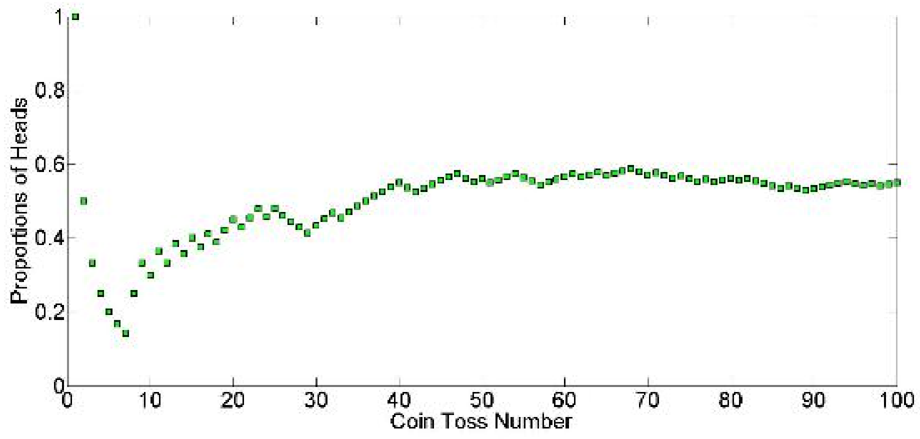

Figure 12.1: MATLAB coin toss simualtion

Example 3. (Binomial) Generate a $Binomial(50, 0.2)$ random variable.

Solution: To solve this problem, we can use the following lemma: