LECTURE 6: STARTING SIMULATION

§1. THE ABC'S OF RANDOM NUMBER GENERATION

(1.1) Computing Background. I will start the lectures on simulation by first assuming that you have access to (i) a language (such as C or better still C_{++}); or (ii) an environment (such as `Matlab`.) If you do not know how to use any programming, you need to get a crash-course, and your T.A.'s (in particular, Sarah and Robert) will help you along if you seek their help. At this point, you should make sure that you (i) have a computer account; and (ii) know how to log in, check mail, and run a program that you know how to run.

(1.2) Generating a Uniformly Distributed Random variable. All of simulation starts with the question, "*How do I choose a random number uniformly between* 0 *and* 1*?*" This is an intricate question, and you will have a detailed lecture on this topic from Dr. Nelson Beebe later this week or the next. These days, any self-respecting programming language or environment has a routine for this task (typically something like `rand`, `rnd`, or some other variant therefrom). Today, we will use such random number generators to generate a few other random variables of interest; we will also apply these methods to simulate random walks.

(1.3) Generating a ± 1 Random Variable. Our first task is to generate a random variable that takes the values ± 1 with probability $\frac{1}{2}$ each. Obviously, we need to do this in order to simulate the one-dimensional simple walk.

The key observation here is that if U is uniformly distributed on $[0, 1]$, then it follows that $P\{U \leq \frac{1}{2}\} = \frac{1}{2}$. So, if we defined

$$X := \begin{cases} +1, & \text{if } U \leq \frac{1}{2}, \\ -1, & \text{if } U > \frac{1}{2}, \end{cases} \tag{1.4}$$

then $P\{X = +1\} = P\{U \leq \frac{1}{2}\} = \frac{1}{2}$ and $P\{X = -1\} = P\{U \geq \frac{1}{2}\} = \frac{1}{2}$. That is, we have found a way to generate a random variable X that is ± 1 with probability $\frac{1}{2}$ each. This leads to the following.

(1.5) Algorithm for Generating ± 1-Random Variables

1. Generate U uniformly on $[0, 1]$
2. If $U \leq \frac{1}{2}$, let $X := +1$, else let $X := -1$

(1.6) Exercises. Try the following:

- **(a)** *Write a program that generates* 100 *independent random variables, each of which is* ± 1 *with probability* $\frac{1}{2}$ *each.*
- **(b)** *Count how many of your generated variables are* ± 1*, and justify the statement that, "with high probability, about half of the generated variables should be* ± 1*."*
- **(c)** *Come up with another way to construct* ± 1 *random variables based on uniforms; a variant of (1.5) is acceptable.*

(1.7) The Inverse Transform Method. We now want to generate other kinds of "discrete random variables," and we will do so by elaborating on the method of (1.5). Here is the algorithm for generating a random variable X such that $P\{X = x_j\} = p_j$ $j = 0, 1, \ldots$ for any predescribed set of numbers $x_0, x_1, \ldots$, and probabilities $p_0, p_1, \ldots$. Of course, the latter means that $p_0, p_1, \ldots$are numbers with values in between 0 and 1, such that $p_0 + p_1 + \cdots = 1$.

(1.8) Algorithm for Generating Discrete Random Variables.

```
1. Generate U uniformly on [0,1]
2. Define
```

$$X := \begin{cases} x_0, & \texttt{if } U < p_0\,, \\ x_1, & \texttt{if } p_0 \le U < p_0 + p_1\,, \\ x_2, & \texttt{if } p_0 + p_1 \le U < p_0 + p_1 + p_2\,, \\ \vdots & \vdots \end{cases}$$

(1.9) Exercise. Prove that the probability that the outcome of the above simulation is x_j is indeed p_j. By specifying $x_0, x_1, \ldots$ and $p_0, p_1, \ldots$ carefully, show that this "inverse transform method" generalizes Algorithm (1.5).

(1.10) Exercise. In this exercise, we perform numerical integration using what is sometimes called *Monte Carlo simulations.*

(a) (Generating random vectors) *Suppose that $U_1, \ldots, U_d$ are independent random variables, all uniformly distributed on $[0,1]$, and consider the random vector $\mathbf{U} = (U_1, \ldots, U_d)$. Prove that for any d-dimesional hypercube $A \subseteq [0,1]^d$, $P\{\mathbf{U} \in A\}$ = the volume of A. In other words, show that $\mathbf{U}$ is uniformly distributed on the d-dimensional hypercube $[0,1]^d$.*

(b) *Let $\mathbf{U}_1, \ldots, \mathbf{U}_n$ be n independent random vectors, all distributed uniformly on the d-dimensional hypercube $[0,1]^d$. Show that for any integrable function f with d variables, the following holds with probability one:*

$$\text{(1.11)} \qquad \lim_{n\to\infty} \frac{1}{n} \sum_{\ell=1}^{n} f(\mathbf{U}_\ell) = \int_0^1 \cdots \int_0^1 f(x_1, \ldots, x_d)\, dx_1 \cdots dx_d.$$

(c) *Use this to find a numerical approximation to the following integrals:*

i. $\int_0^1 e^{-x^2}\, dx$.

ii. $\int_0^1 \int_0^1 y^x\, dx\, dy$.

§2. SHORT-CUTS: GENERATING BINOMIALS

(2.1) The Binomial Distribution. A random variable is said to have the *binomial distribution* with parameters n and p if

$$(\mathbf{2.2}) \qquad P\{X = j\} = \binom{n}{j} p^j (1-p)^{n-j}, \qquad j = 0, 1, \ldots, n.$$

Here n is a positive integer, and p is a real number between 0 and 1.

(2.3) Example. For example, suppose n independent success/failure trials are performed; in each trial, $P\{\text{success}\} = p$. Then, if we let X denote the total number of successes, this is a random variable whose distribution is binomial with parameters n and p. ♣

(2.4) Example. Suppose $\xi_1, \ldots, \xi_n$ are independent with $P\{\xi = 1\} = p$ and $P\{\xi = 0\} = 1 - p$. Then, $X := \xi_1 + \cdots + \xi_n$ is binomial.

Proof: Let $\xi_i = 1$ if the ith trial succeeds and $\xi_i = 0$ otherwise. Then X is the total number of successes in n independent success/failure trials where in each trial, $P\{\text{success}\} = p$. ♣

(2.5) Example. If S_n denotes the simple walk on the integers, then $S_n = X_1 + \cdots + X_n$, where the X's are independent and every one of the, equals ± 1 with probability $\frac{1}{2}$ each. On the other hand, $Y_i := \frac{1}{2}(X_i + 1)$ is also an independent sequence and equals ± 1 with probability $\frac{1}{2}$ each (why?) Since $X_i = 2Y_i - 1$,

$$(\mathbf{2.6}) \qquad S_n = 2 \sum_{i=1}^{n} Y_i - n.$$

Therefore, the distribution of the simple walk at a fixed time n is the same as that of $2 \times \text{binomial}(n, p) - n$.

(2.7) A Short-Cut. Suppose we were to generate a binomial(n, p) random variable. A natural way to do this is the inverse transform method of (1.7) and (1.8). Here, $x_0 = 0, x_1 = 1, \ldots, x_n = n$, and p_j is the expression in (2.2). The key here is the following short cut formula that allows us to find p_{j+1} from p_j without too much difficulty:

$$(\mathbf{2.8}) \qquad \begin{aligned} p_{j+1} &= \binom{n}{j+1} p^{j+1} (1-p)^{n-j-1} \\ &= \frac{p}{p-1} \times \frac{n!}{(j+1)! \times (n-j-1)!} \times p^j (1-p)^{n-j} \\ &= \frac{p}{p-1} \times \frac{n-j}{j+1} \times \binom{n}{j} p^j (1-p)^{n-j} \\ &= \frac{p}{p-1} \times \frac{n-j}{j+1} \times p_j. \end{aligned}$$

So we can use this to get an algorithm for quickly generating binomials.

(2.9) Algorithm for Generating Binomials.

```
1. Generate U uniformly on [0,1].
2. Let Prob := (1 - p)^n and Sum := Prob.
3. For j = 0,...,n, do:
    i. If U < Sum, then let X = j and stop.
   ii. Else, define
```

$$\texttt{Prob} := \frac{\texttt{Prob}}{1-\texttt{Prob}} \times \frac{n-j}{j+1} \times \texttt{Prob}, \qquad \texttt{and} \qquad \texttt{Sum} := \texttt{Prob} + \texttt{Sum}.$$

You should check that this really generates a binomial. ♣

(2.10) Algorithm for Generating the One-Dimensional Simple Walk. Check that the following generates and plots a $1-d$ simple walk.

```
1. (Initialization) Set W := 0 and plot (0,0).
2. For j = 0,...,n, do:
    i. Generate X = ±1 with prob. 1/2 each.
   (See (1.5) for this subroutine.)
   ii. Let W := W + X and plot (j,W).
```

If you are using a nice plotting routine like the one in `Matlab`, try filling in between the points to see the path of the walk.

(2.11) Exercise. Generate 2-dimensional simple walks that run for (a) $n = 100$ time units; (b) $n = 1000$ time units.