6.5 Random Numbers

The final topic we will cover in connection with modular arithmetic is that of random numbers, and how computers generate them. It turns out that there is an amazing, though quite mysterious, connection between modular arithmetic with large exponents and the generation of "random numbers". We put the words random numbers in quotations because in a sense that will become clear in a moment, the random numbers we generate will not really be random at all, even if a person looking at them doesn't really know that.

Random numbers appear regularly in our everyday lives in different forms. If you've ever watched an NFL ref flip a coin to determine how a game would start, or if you've ever rolled a die while playing a board game with friends, or if you've ever chosen a random card from a deck of cards, then you've witnessed the role that randomness can play in determining some course of events.

Computers too use randomness on a regular basis. If you've ever asked iTunes to shuffle your playlist, you have asked your computer to do something that is random. If you've ever played a video game, you've witnessed a computer make choices about many random details. If you've ever used a computer to create realistic-looking pictures, for a movie or a video game, the computer has likely needed to generate random numbers, to help mimic the randomness that appears so ubiquitous in nature. If you are trying to use a computer to simulate some physical, biological, chemical, economic process, chances are that you will need some randomness to make sure that your simulation is realistic. And finally, if you have ever communicated secure information through the internet, chances are that your computer has needed to generate some kind of random number.

Discussing the generation of random numbers by computers requires us to first consider the more general question of what we mean by random numbers. We begin with a very brief discussion of some basic ideas of probability.

What are random numbers?

We begin with several simple exercises, to highlight three simple lessons of probability. As an exercise, think of a random number between 1 and 10. Imagine that you chose seven. Does that mean that seven is a random number? Is it more random than two or three or nine? Of course these are silly questions, and it doesn't make much sense to discuss whether individual numbers are random or not. The more interesting, and fruitful, question is whether some sequence of numbers is random. **Lesson 1**: There is no such thing as a random number. Instead we consider sets or sequences of numbers that are random.

Next, consider the following two sequences of numbers. Perhaps both sets of numbers correspond to a sequence of coin-flips, with heads indicated by 1's and tails indicated by 0's:

a) 0 1 0 1 0 1 0 1 0 1 0 1 0 1

b) 0 1 1 0 0 1 1 1 0 0 1 0 1 1

Sequence **a)** does not appear "random" – it's merely a repeating pattern of 0 and 1. What about the sequence **b)**? This one appears to be random, or at least much more so than the first. Now consider the following two sequences of integers.

c) 2 6 5 3 5 8 9 7 9 3 2 3

d) 7 5 2 3 10 4 6 9 8 1

Is the first set of numbers random? What about the second set? Looking back at **a)** and **b)**, these two appear significantly "more" random than those. Indeed, even if we thought that **b)** appeared random, we would admit that it appears random only when considering sequences of 0's and 1's, but not when considering more arbitrary sequences. If we we allow integers all the way up to 10, then neither of **a)** nor **b)** seem at all random. **Lesson 2**: Randomness is "relative". Whether or not a sequence of numbers is "random" depends on what numbers we are choosing from.

Finally, consider the following set of numbers, chosen from 5 to 30.

e) 20 19 14 12 19 18 17 19 20 14 18 12 20

Are these numbers random? Even though you might believe that these numbers are all from between 5 and 30, they certainly don't seem very random, as all of them are greater than 10 and no larger than 20. Could these numbers have indeed been chosen from between 5 and 30? Indeed these numbers were chosen from between 5 and 30, yet the manner in which they were chosen was not *uniform*. That is, there was not an equal chance that 5 and 15 and 25 were chosen. In fact, these numbers were obtained as follows. To obtain each number, I rolled a die five times and added the sum of their values. Since there were five dice, of course the minimum value I could get was 5 (if I rolled only 1's) and the maximum was 30 (if I rolled only 6's). But it's much easier to obtain numbers in the middle than numbers at the extremes. **Lesson 3**: Randomness does not need to be *uniform*; the probability of choosing one object can be different from the probability of choosing another one.

Along these lines, consider the following "random" SAT scores:

f) 1590 1470 2100 830 1930 2040 840 2050 1950 840 640.

Of course these numbers must be between 600 and 2400, and must be divisible by 10. But despite being among certain values, they don't of course, represent a "random" sample, certainly not among Penn students, but not even among the general population. Therefore, when choosing numbers randomly, we must always specify the probability of each outcome. Of course, many more students score a 1700 than score an 800 or 2300.

Lessons 2 and 3 highlight the need for describing a probability distribution before considering whether a particular set of numbers is random or not – randomness cannot be sensibly discussed without this kind of frame of reference. A probability distribution describes that frame of reference by giving us a list of possible values that can be chosen, and the probabilities of choosing each of those values. In all of our examples, we have considered distributions with a finite number of possible choices, though in theory we can consider more complicated ones.

In discussing how computers generate random numbers, we will only consider a **uniform distribution** on a finite set of numbers. In particular, suppose we want a computer to choose a number between 1 and 100 with a one percent chance of choosing any particular number. More generally, we might want to ask a computer to choose a number between 1 and N, with an equal chance of choosing any one of the N choices. How can a computer do this?

How Computers Generate Random Numbers

Computers are very, very, very good at completing certain tasks. For example, computers are very good at adding numbers. Or multiplying them. Or dividing them and taking square roots and exponents. And they can do this all really, really, really well, and much faster and more accurately than any of us. But they can only do what you tell them to do, with specific instructions. This raises the question of how can a computer generate a random number? How can it generate something random by following a fixed set of rules? The honest truth is that it can't. A deterministic machine that just follows orders, and follows fixed instructions, cannot generate true random numbers. However, we will see that, using modular arithmetic, a computer *can* generate numbers that appear random for most intents and purposes.

Irrational numbers. One source of random numbers can be found in the decimal expansion of irrational numbers. Consider, for example, the decimal expansions of π, e, and $\sqrt{2}$:

g) (3.) 1 4 1 5 9 2 6 5 3 5 8 9 7 9 3 2 3 8 4 6

h) (2.) 7 1 8 2 8 1 8 2 8 4 5 9 0 4 5 2 3 5 3 6

i) (1.) 4 1 4 2 1 3 5 6 2 3 7 3 0 9 5 0 4 8 8 0

It is commonly believed that these numbers are very random, in a way that can be made precise. In practice, people have looked at the first billion digits of π and the digits seem to appear randomly distributed. For example, roughly one tenth of all digits are 0's, one tenth are 1's, and so forth. No one, though, knows whether this is true indefinitely. As far as we know, after the first trillion digits, there tend to be considerably fewer 3's than any other digit; it is also possible that there are no 7's appearing after the first fifty trillion digits. No one knows how to prove anything about this.

In practice, irrational numbers are not commonly used to generate random-looking numbers for several reasons. First, generating digits in this manner is expense, both in terms of computational time and memory. Furthermore the resulting numbers are very predictable. If I can figure out that you are using π or e or $\sqrt{2}$ to generate your random numbers, then in theory I can exactly predict every number that you will ever create. We will see later that this doesn't have to be a fatal flaw, but in practice, it often is.

Linear congruent generators

The most widely-used random number generators are called linear congruent generators, and in this section we will learn about what they are and how they are used. Remember that computers can follow orders, so we are trying to find directions that create numbers that appear random.

The simplest type of random congruent generator is a sequence of numbers of the form:

$$s, sa^1, sa^2, sa^3, sa^4, \dots \pmod m, \tag{75}$$

where a is called the multiplier, m is the modulus, and the first element s is called the "seed". Each term in the sequence can be obtained by multiplying the term before it by a, and then taking it mod m (i.e., the remainder after dividing it by m). We can rewrite the above in a slightly more condensed form. In particular, if we use x_i ot indicate the i^{th} number in the sequence, we can write:

$$x_i = a \cdot x_{i-1} \pmod m, \tag{76}$$

where we let $x_0 = s$. This definition defines each term as the product of a and the preceding term in the sequence.

Let us consider a simple example. We choose a seed $s = 1$, a multiplier $a = 7$, and a modulus $m = 11$. These choices of s, a, and m give us a sequence:

$$1, 7, 5, 2, 3, 10, 4, 6, 9, 8, 1, 7, 5, \dots \tag{77}$$

These numbers look fairly random, but the problem is that they repeat too quickly. For reasons we have already discussed, the length of this sequence can be no longer than $m - 1$. Therefore, in practice, random number generators try to use a large modulus m.

Next we consider a much larger modulus $m = 2^{31}$ and multiplier $a = 65539$; this was the basis for an historically-important random number generator developed and used by IBM in the 1960's. If we choose a seed $s = 123456789$, then our first several numbers are:

$$123456789, 1663592255, 280507837, 1743102263, 1491592101, \dots \tag{78}$$

At first glance these numbers might appear fairly "random", and indeed they are evenly distributed between 1 and 2^{31}. However, you might notice that every term is odd, which occurs when the seed s is chosen odd; if s is chosen even, then every subsequent term will be even. Although we can get around this problem by always dropping the last digit, this problem highlights some of the challenges involved in designing random number generators.

In practice, however, linear congruent generators are the most widely-used pseudo-random number generators in common use, and much work has been put into determining good choices of multiplier a, modulus m, and seed s.

A slight generalization of the example described here involves not only multiplying preceding terms by a constant a, but also adding a number to it. More concretely, we choose a constant integer c which we call the *increment* and add that after multiplying the previous term by a. In equation form,

$$x_i = a \cdot x_{i-1} + c \pmod{m}. \tag{79}$$

To see how this works, let's consider the simple example we considered before, where we chose a multiplier $a = 7$, a modulus $m = 11$, and a seed $s = 1$. Let us now also choose an increment $c = 5$. Instead of sequence in (77), we now get:

$$1, 10, 7, 8, 4, 9, 0, 3, 2, 6, 1, 10, 7, \ldots \tag{80}$$

The pattern is again periodic, but the order of the numbers have changed. In some situations, including the extra incremental term c can improve certain properties of the random numbers generated, but sometimes it can make things much worse. Consider, for example, what happens in the previous example if we instead use an increment $c = 5$. Notice that $(7 \cdot 1 + 5) = 12 \equiv 1 \pmod{11}$. In words, if we begin with 1, multiply it by 7 and add 5, and then consider the remainder after dividing by 11, then the remainder is 1. Therefore the "random" sequence generated will end up being an endless sequence of 1's – not a very random sequence at all!

Randomness testing

We have already seen some potential pitfalls in the development of random number generators. Sometimes the period of the repeating pattern is very short. Other times the pattern might be quite long but the generated numbers are all odd, or all even. Even if we are content with using deterministic algorithms to generate number sequences that only look random, we still want to make sure that they indeed appear random. Over the last fifty years, many tests have been developed to determine whether a particular set of pseudo-randomly-generated numbers indeed appear random. One of the best-known and most powerful such tests is called the spectral test.

To help understand the spectral test, consider the following sequence of numbers generated by linear congruence generators with seed $s = 7$, multiplier $a = 5$, and modulus $m = 97$:

$$7, 35, 78, 2, 10, 50, 56, 86, 42, 16, 80, 12, 60, 9, 45, 31, \ldots \tag{81}$$

Next consider the sequence of numbers when we keep the modulus and seed, but change the multiplier a to 10:

$$7, 70, 21, 16, 63, 48, 92, 47, 82, 44, 52, 35, 59, 8, 80, \ldots \tag{82}$$

The two sequences appear at first to be equally "random". However, consider what happens if we plot all pairs of consecutive terms in each sequence. For the first sequence, for example, we would plot points $(7, 35)$, $(35, 78)$, $(78, 2)$, etc.

For the second sequence, we plot (7, 70), (70, 21), (21, 16), etc. Something mysterious occurs the quickly highlights the different strengths of the two sequences. When we used the multiplier $a = 5$, all of the 96 points are "bunched up" on

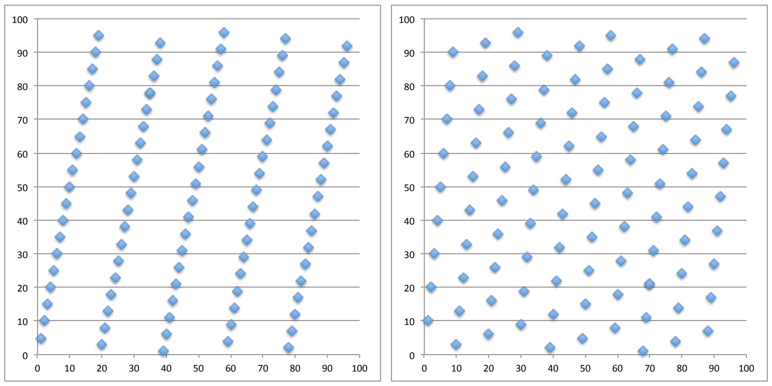

Figure 37: Linear congruence generators using modulus $m = 97$ and initial seed $s = 7$ The points on the left are taken from a sequence generated using a multiplier $a = 5$; those on the right are taken from a sequence generated using a multiplier $a = 10$.

five lines, where when we use the multiplier $a = 10$, the 96 generated points are much more spread out. A branch of mathematics called Fourier analysis can be used to make these ideas precise, and in practice this kind of test is used to determine whether particular choices of multiplier and modulus are random "enough" for particular applications.