# Lecture 2: Random number generation

Ciaran Evans

### Last time

```
mu_x <- 0
n <- 20
x <- rnorm(n, mean=mu_x, sd=1)</pre>
```

- The rnorm function provides a random sample from a univariate normal distribution with specified mean and standard deviation
- ► What other functions exist in R for sampling from probability distributions?

## Our goal for this unit

Goal: Learn how to simulate random variables

Two main steps:

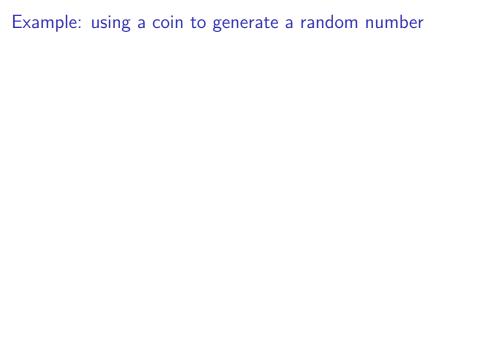
- 1. Generating "random" (really, pseudo-random) numbers
- 2. Using random numbers to simulate from a specified distribution

### Warm-up question

Suppose that someone asked you to generate a random number (e.g. between 0 and 1). Without resorting to existing software, what would you do? (Your answer does not have to involve a computer!)

Example: using a coin to generate a random number

First, note that we can represent integers in binary (base 2):



### "Random" numbers

- ► The typical way to generate "random" numbers is with a computer
- By themselves, computers can't generate truly random numbers
- ► Instead, computers use a deterministic algorithm to generate pseudo-random numbers

Example: what does it mean to "behave" like a random number?

Consider the following strings of 0s and 1s

#### **Questions:**

- 1. If P(0) = P(1) = 0.5, what is the probability of each string?
- 2. Which string do you think was actually randomly generated?

### Linear congruential generator

One of the oldest (and historically, widely used) generators is the **linear congruential generator**:

## Examples:

$$ightharpoonup$$
  $a = 1$ ,  $c = 1$ ,  $m = 8$ 

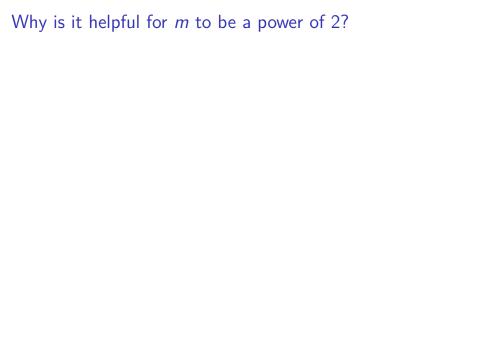
$$ightharpoonup$$
  $a = 3$ ,  $c = 0$ ,  $m = 16$ 

$$\rightarrow$$
  $a = 5$ ,  $c = 3$ ,  $m = 16$ 

## Choosing the parameters

A sufficient condition for a period of length m (for any initial seed) is:

- ightharpoonup c and m are coprime (i.e., greatest common divisor is 1)
- ightharpoonup a-1 is divisible by all prime factors of m
- ightharpoonup a-1 is divisible by 4 if m is divisible by 4



### Your turn

Write code to implement a LCG in R, and experiment with different values of  $\it{m}$ ,  $\it{a}$ , and  $\it{c}$