MTH511a - 2020: Lecture 2

Instructor: Dootika Vats

1 Pseudorandom Number Generation

The building block of computational simulation is the generation of uniform random numbers. If we can draw from $U(0,1)$, then we can draw from *most* other distributions. Thus the construction of sampling from $U(0,1)$ requires special attention.

Computers can generate numbers between $(0,1)$, which although are not exactly random (and in fact deterministic), but have the appearance of being $U(0,1)$ random variables. These draws from $U(0,1)$ are *pseudorandom* draws.

The goal in *pseudorandom* generation is to draw

$$x_1, \ldots, x_n \overset{\text{approx}}{\sim} U(0,1)\,.$$

so that they are as uniformly distributed as possible.

1.1 Multiplicative congruential method

A common algorithm to generate a sequence $\{x_n\}$ is the *multiplicative congruential method*:

1. Set seed x_0, and positive integers a, m.
2. $x_n = a\, x_{n-1} \mod m$
3. Return sequence x_n/m.

x_n is one of $0, 1, \ldots m-1$, and so x_n/m is between $(0,1)$.

Also note that after some finite number of steps $< m$, the algorithm will repeat itself, since when a seed x_0 is set, a deterministic sequence of numbers follows.

Example 1 *Set* $a = 123$ *and* $m = 10$*, and let* $x_0 = 7$*. Then*
$x_1 = 123 * 7 \mod 10 = 1$
$x_2 = 123 * 1 \mod 10 = 3$
$x_3 = 123 * 3 \mod 10 = 9$
$x_4 = 123 * 9 \mod 10 = 7$

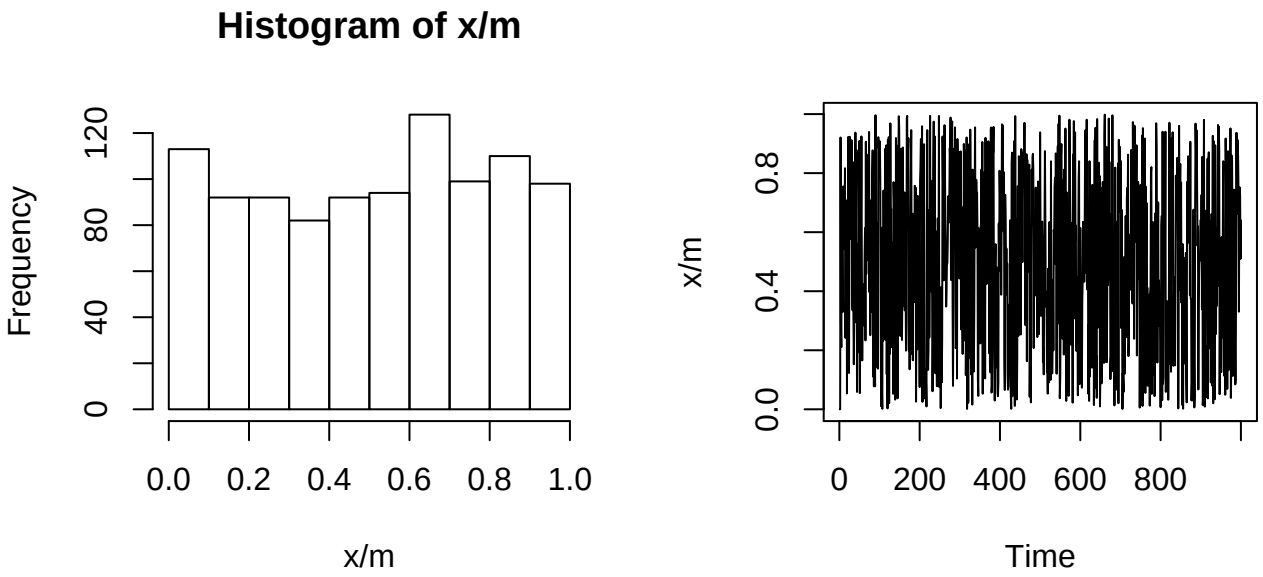
$x_5 = 123 * 7 \mod 10 = 1$
$\vdots$

Thus, we see that the above choices of a, m, x_0 repeats itself. Naturally, both a and m should be chosen to be large so as to avoid repetition.

It is recommended to set $m = 2^{31} - 1$ and $a = 7^5$. Notice that both are large.

```
m <- 2^(31) - 1
a <- 7^5
x <- numeric(length = 1e3)
x[1] <- 7

for(i in 2:1e3)
{
  x[i] <- (a * x[i-1]) %% m
}
par(mfrow = c(1,2))
hist(x/m) # looks close to uniformly distributed
plot.ts(x/m) # look like it's jumping around too
```



Any pseudorandom generation method should satisfy:

1. for any initial seed, the resultant sequence has the "appearance" of being IID from Uniform$[0, 1]$.
2. for any initial seed, the number of values generated before repetition begins is large
3. the values can be computed efficiently.

Typically m should be a large prime number

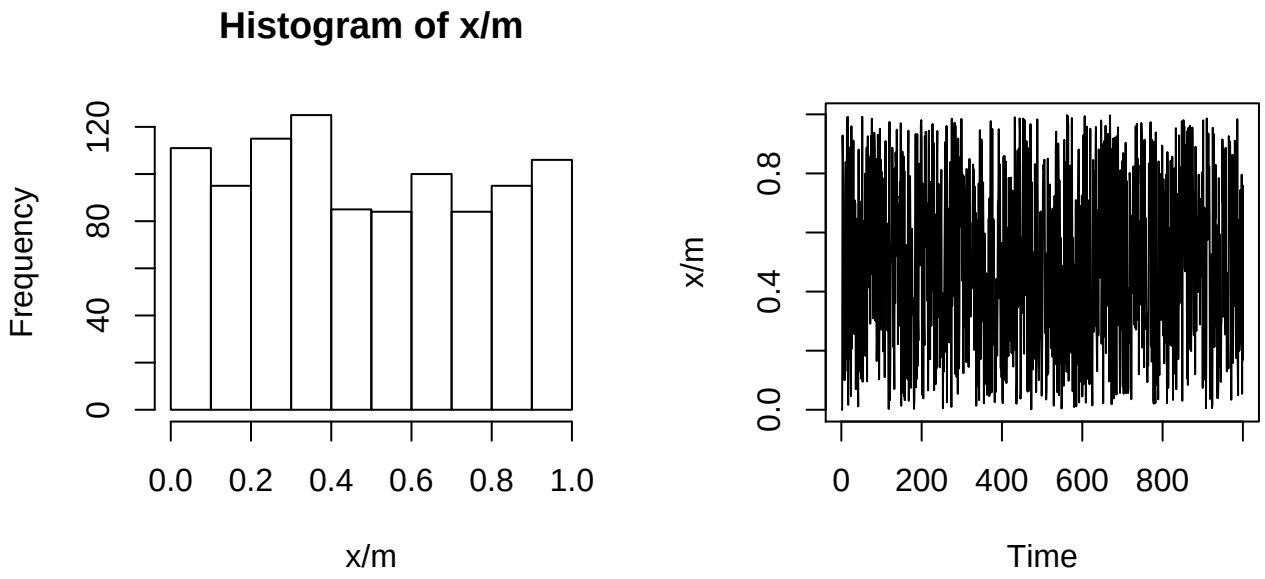
1.2 Mixed Congruential Generator

Another method is the *mixed congruential generator*:

1. Set seed x_0, and positive integers a, c, m.
2. $x_n = (a\, x_{n-1} + c) \mod m$
3. Return sequence x_n/m.

```
m <- 2^(31) - 1
a <- 7^5
c <- 2^(10) - 1
x <- numeric(length = 1e3)
x[1] <- 7

for(i in 2:1e3)
{
  x[i] <- (c + a * x[i-1]) %% m
}
par(mfrow = c(1,2))
hist(x/m) # looks close to uniformly distributed
plot.ts(x/m) # look like it's jumping around too
```

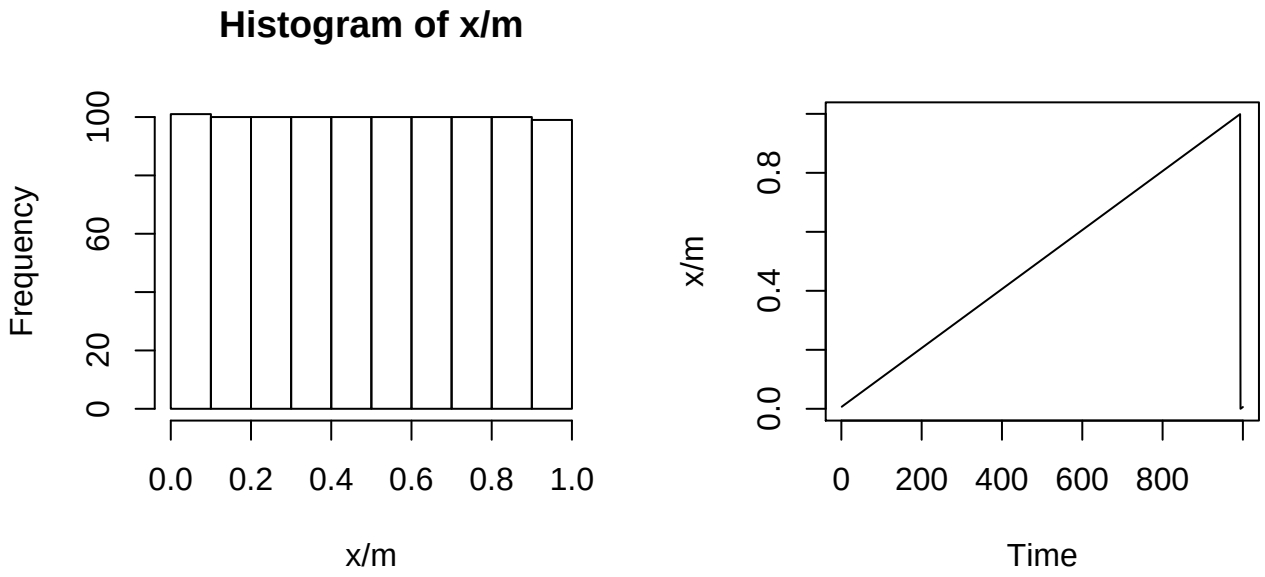


We must be cautious not to be happy with a just a histogram. A histogram shows that the empirical distribution of all samples is uniformly distributed. But we can still get a uniform looking histogram if we set $a = 1$, $m = 1e3$ and $c = 1$

```
m <- 1e3
a <- 1
c <- 1
x <- numeric(length = 1e3)
x[1] <- 7

for(i in 2:1e3)
{
  x[i] <- (c + a * x[i-1]) %% m
}
par(mfrow = c(1,2))
```

```r
hist(x/m) # looks uniformly distributed
plot.ts(x/m) # look like it's jumping around too
```



Although a histogram shows an almost perfect uniform distribution, the trace plot shows that the draws don't behave like they are independent.

We could also use

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + \cdots + a_k x_{n-k} + c) \mod m\,,$$

but this requires more flops from the computer, and so is not as computationally viable.

We claim that these methods return "good" pseudosamples, in the sense of the three points. There are statistical hypothesis tests, like the Kolmogorov-Smirnov test, one can do to test whether a sample is truly random: independent and identically distributed.

2 Integrals continued

Now that we know how to generate (pseudo) random numbers from Uniform$[0,1]$, we are equipped to *estimate* integrals. Recall our simple problem

$$\theta = \int_0^1 (3x^2 + 5x)\,dx\,.$$

We can now carry on a simple Monte Carlo procedure to *estimate* θ. What if for arbitrary a and b, interest is in

$$\int_5^{10} (3x^2 + 5x)\,dx\,?$$

If we can draw from $U(5,10)$, then we estimate the integral. But we only know how to draw from $U(0,1)$. Note that if $U \sim U(0,1)$, then for any a, b,

$$(b-a) * U + a \sim U(a,b) \quad .$$

That means, we can draw $U \sim U(0,1)$ and set $X = (b-a) * U + a$. Then $X \sim U(a,b)$.

$$\theta = \int_5^{10} (3x^2 + 5x)\, dx = 5 \int_5^{10} (3x^2 + 5x) \frac{1}{5} = \mathrm{E}_{X \sim U(10,5)}(3X^2 + 5X)$$

```
set.seed(1)
repeats <- 1e4
b <- 10
a <- 5
U <- runif(repeats, min = 0, max = 1)
X <- (b - a) * U + a #R is vectorized

5* mean(3*X^2 + 5*X)
```

```
## [1] 1063.222
```

2.1 Higher dimensional integrals

Consider estimating the integral

$$\theta = \int_{a_k}^{b_k} \dots \int_{a_1}^{b_1} g(x_1, x_2, \dots, x_k)\, dx_1, dx_2, \dots, dx_k$$

The same rules apply; We want to find a distribution that is defined on the space $(a_1, b_1) \times \cdots \times (a_k, b_k)$. Independent uniforms would do the trick!

Consider estimating

$$\theta = \int_2^3 \int_5^6 3x^2 y\, dxdy = \mathrm{E}\left[3x^2 y\right]$$

where that expectation is with respect to $U(5,6) \times U(2,3)$.

```
set.seed(1)
repeats <- 1e4
U1 <- runif(repeats, min = 0, max = 1)
X <- (6 - 5) * U + 5

U2 <- runif(repeats, min = 0, max = 1)  # have to generate different U2
Y <- (3 - 2) * U + 2

mean(3*X^2 * Y)
```

```
## [1] 230.3351
```

Recall: There are two main theoretical results that are essential in justifying the use of Monte Carlo methods:

1. Weak/Strong law of large numbers: Let $X_1, X_2, \dots$ be a sequence of iid random variables having mean $\mu < \infty$. Then for any $\epsilon > 0$,

$$\Pr\left\{\left|\frac{X_1 + \cdots + X_n}{n} - \mu\right| > \epsilon\right\} \to 0 \quad \text{as } n \to \infty\,.$$

2. Central limit theorem: Let $X_1, X_2, \ldots, X_n \overset{iid}{\sim} F(x)$ with $\mathrm{E}(X_1) = \mu$ and $\mathrm{Var}(X_1) = \sigma^2 < \infty$, then

$$\sqrt{n}(\bar{X}_n - \mu) \xrightarrow{d} N(0, \sigma^2)\,.$$

The first tells us we will get increasingly close to the truth, and the second gives us a way to measure the variability in the estimator.

2.2 Questions to think about

- Given a sample of pseudorandom draws from $U(0,1)$ and perfectly IID draws from $U(0,1)$, would you be able to tell the difference?
- What would happen in the last example if 'U1' was used to generate Y as well, instead of using a separate $U2$?