Lecture A5. Simulation 2

Sim, Min Kyu, Ph.D., mksim@seoultech.ac.kr

서울과학기술대학교 데이터사이언스학과

I. Random uniform number

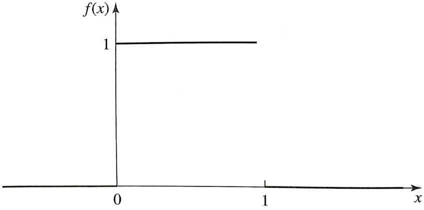
Recap

- In the previous simulation lecture, random numbers that follows $U(-1, 1)$ were the initial components of the simulation process for estimating π.
- Since a random variable that follows $U(-1, 1)$ is merely a linear transformation of $U(0, 1)$, we will discuss the generation process for $U(0, 1)$.

$U(0,1)$

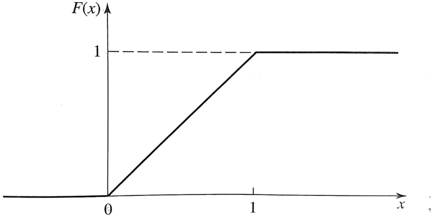
- pdf

$$\text{pdf } f(x) = \begin{cases} 1 & \text{if } \ 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$



- cdf

$$\text{cdf } F(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x < 1 \\ 1 & \text{if } x > 1 \end{cases}$$



Generating $U(0,1)$ - bisection method

- Following step will generate a random number u that follows $U(0,1)$.
  1. Let $X = (0,1]$, and we know u falls into some point within X.
  2. Divide X into half. Call its lower half interval as A and its upper half interval as B.
  3. Flip a coin. If head, let $X = A$. If tail, let $X = B$.
  4. Goto step 2, unless the length of X is less than some precision tolerance, say, ϵ.
  5. Let u be the mid-point of the interval X.
- Since u must fall into the bounded interval of $(0,1]$ anywhere equally likely, one can device such as coin or dice.
- There are serious mathematicians who are devoted to generate uniform random numbers efficiently.
- Then, what about the random number that follows non-uniform distribution?

II. Inverse transform method

Motivation for random number generation from a general cdf.

- For a continuous random variable X, its cdf has following properties.
  1. Its lower limit is always 0.
  2. Its upper limit is always 1.
  3. The function is always monotonically non-decreasing.

- Discussion
  1. From the property 3 above, a cdf is one-to-one function.
  2. Since one-to-one, the cdf has an inverse function. It means that finding the cdf's y-value automatically gives the function's x-value.
  3. The function's y value is in the bounded interval $[0, 1]$

- Motivated by the above points of 2 & 3, one can simply 1) find u from $U(0, 1)$, and then 2) take its inverse value with respect to the cdf.

Inverse transform method

Theorem 1 (inverse transform method)

If X is a continuous random variable with cdf $F(x)$, then the random variable's CDF $F(X) \sim U(0,1)$.

Remark 1

The above theorem suggests a way to generate realizations of the random variable X. Namely,

1. Pick u from $U(0,1)$
2. Solve $u = F(x)$ for x, or $x = F^{-1}(u)$.
3. Then, x is a random number from the random variable with cdf $F(x)$

Exponential random numbers

Remark 2

For example, we want to find a x from $X \sim exp(5)$ and we picked $u = 0.3$ from $U(0, 1)$, then what is the random number x that follows $exp(5)$?

1. $u = 0.3$
2. $u = 1 - e^{-5x} \;\Rightarrow\; 1 - u = e^{-5x} \;\Rightarrow\; log(0.7) = -5x \;\Rightarrow\; x = \frac{-log(0.7)}{5}$
3. $x = \frac{-log(0.7)}{5}$

Exercise 1

Using **`ruinf()`** *function in R, complete the following code block that generates 1,000 random numbers that follow* $exp(3)$.

```
N <- 1000
u <- runif(N)
x <- (complete here)
head(x)
```

- Uniform random number is indeed the building block for all random numbers!
- What about a random number from a discrete distribution? It's easy.

Random number for discrete distribution

- Suppose a discrete r.v. X has the distribution of the following.

x	1	2	3	4
$\mathbb{P}(X = x)$	.1	0	.4	.5
$\mathbb{P}(X \leq x)$	.1	.1	.5	1.0

- The process is the same. First to pick a u from $U(0, 1)$. Next,
  1. if $u \leq .1$, then let $x = 1$.
  2. if $.1 < u \leq .5$, then let $x = 3$.
  3. if $.5 < u$, then let $x = 4$.
- x is a random number for X.

III. Various random numbers

Using built-in function

- Most programming languages provide built-in random number generator.
- R does so as well with functions whose prefix `r-`, such as `runif()`, `rnorm()`, `rexp()`, `rpois()`, and so on.
- Code in `help(runif)` in console opens helper as below.

R: The Uniform Distribution

Uniform {stats} R Documentation

The Uniform Distribution

Description

These functions provide information about the uniform distribution on the interval from `min` to `max`. `dunif` gives the density, `punif` gives the distribution function `qunif` gives the quantile function and `runif` generates random deviates.

Usage

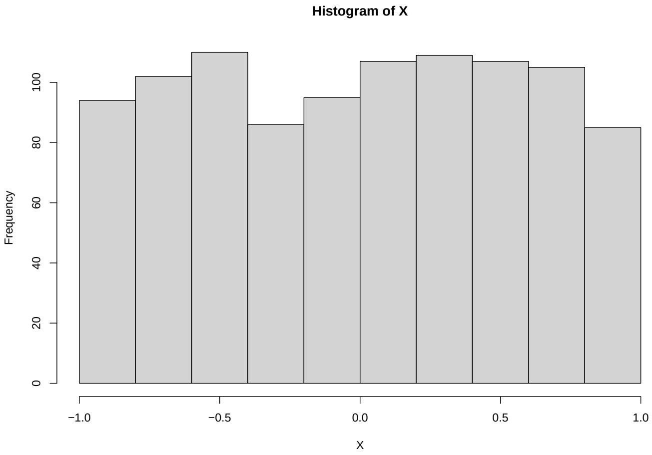
```
dunif(x, min = 0, max = 1, log = FALSE)
punif(q, min = 0, max = 1, lower.tail = TRUE, log.p = FALSE)
qunif(p, min = 0, max = 1, lower.tail = TRUE, log.p = FALSE)
runif(n, min = 0, max = 1)
```

Arguments

`x, q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`min, max`	lower and upper limits of the distribution. Must be finite.
`log, log.p`	logical; if TRUE, probabilities p are given as log(p).
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \le x]$, otherwise, $P[X > x]$.

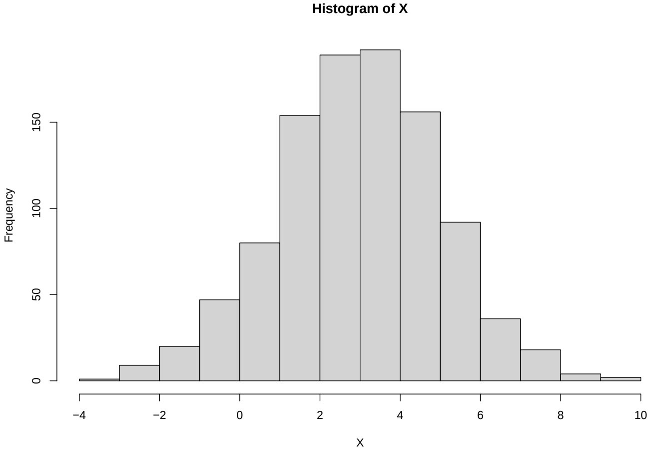
Uniform random numbers

```
X <- runif(n=1000, min=-1, max=1)
hist(X)
```



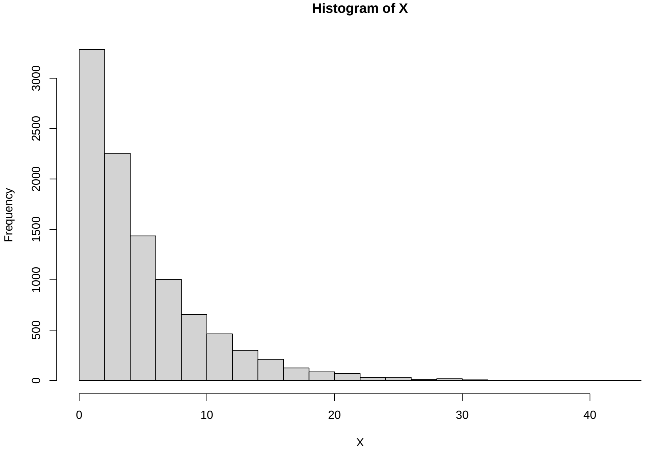
Normal random numbers

```r
X <- rnorm(n=1000, mean=3, sd=2)
hist(X)
```



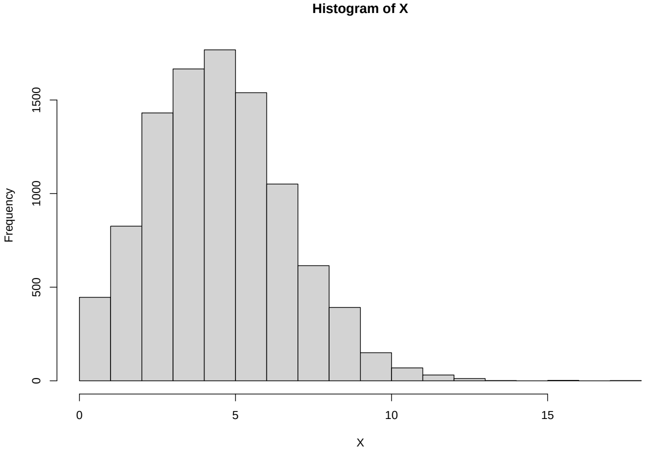
Exponential random numbers

```
X <- rexp(n=10000, rate = 1/5) # meaning lambda=5
hist(X, breaks = 20)
```



Poisson random numbers

```r
X <- rpois(n=10000, lambda = 5) # meaning lambda=5
hist(X, breaks = 20)
```



Exercise 2

Write a concise python code including histograms in this section.

```
"If I only had an hour to chop down a tree, I would spend the first 45 minutes sharpening my axe.
- A. Lincoln"
```