MONTE CARLO SIMULATION OF RANDOM VARIABLES

Being able to generate (or simulate) random values from a Uniform (0, 1) distribution is fundamental is to the generation of random variables from other distributions.

Every programming language has a *random number generator*, an intrinsic function such as "rand ()", that simulates a random value from Uniform (0, 1) distribution. Subsequent calls to this function will give "independent" random values from this distribution. When we say "independent", we mean independent for all practical purposes. These random numbers are also known as "pseudo-random numbers" as they are not truly independent.

We will assume that we can generate random values from Uniform (0, 1) distribution, without discussing how this is done.

HW: Find the function that generates Uniform(0, 1) random values in your favorite programming language.

Simulating from Bernoulli (p) distribution:

Recall: If X ~ Bernoulli (p) then $P(X = 1) = p$ and $P(X = 0) = 1 – p$.

Algorithm:
1. Generate U, a rv from Uniform (0, 1).
2. If $U \le p$; set $X = 1$,
   else set $X = 0$.

Verification:
$P(X = 1) =$
$P(X = 0) =$

Simulating from Binomial (n, p) distribution:

Recall: Binomial (n, p) random variable = sum of n independent Bernoulli (p) variables.

Simulating from Geometric (p) distribution:

Recall: If X ~ Geometric(p), then X = # independent Bernoulli (p) trials needed to get the first success.

Algorithm:
Initialize $X = 0$ and repeat the following steps until $U < p$.
1. Generate $U \sim$ Uniform (0, 1).
2. Increment $X = X + 1$. If $U \le p$, stop; else go to step 1.

This method is not very efficient. Alternative methods are available.

Simulating from continuous distributions:

Result: If X is a continuous random variable with CDF $F(x)$, then $U = F(X)$ follows Uniform (0, 1) distribution.

So, to simulate a value X,

1. Generate U ~ Uniform(0, 1)
2. Set U = F(X)
3. Solve for X (i.e., invert the CDF).

Note: Many time the equation U=F(X) cannot be solved explicitly. Alternatives are available.

Simulating from Exponential (λ) distribution:

Recall: If X ~ Exponential (λ), then F(x) = $1 - \exp(-\lambda x)$.

Setting $U = 1 - \exp(-\lambda X)$ leads to $X = -(1/\lambda)\log(1-U)$.

Algorithm:

1. Generate U ~ Uniform(0, 1)
2. $X = -(1/\lambda)\log(1-U)$.

Simulating from Gamma (r, λ): (Difficult to invert its CDF)

Recall: Gamma (r, λ) random variable = sum of r independent Exponential (λ) random variables.

Simulating from Normal (μ, σ^2): (Difficult to invert its CDF)

Result (Box-Muller method): If U_1 and U_2 are two independent Uniform (0, 1) variables, then

$$Z_1 = \sqrt{-2\log(U_1)}\sin(2\pi U_2),\ Z_2 = \sqrt{-2\log(U_1)}\cos(2\pi)U_2$$

are two independent Normal (0, 1) variables.

So, to generate X ~ Normal (μ, σ^2), take $X = \sigma Z_1 + \mu$.

Using simulation to estimate a probability:

Recall the relative frequency interpretation of probability:

P(A) ≈ proportion of times the event A happens in a large # of independent repetitions of the experiment.

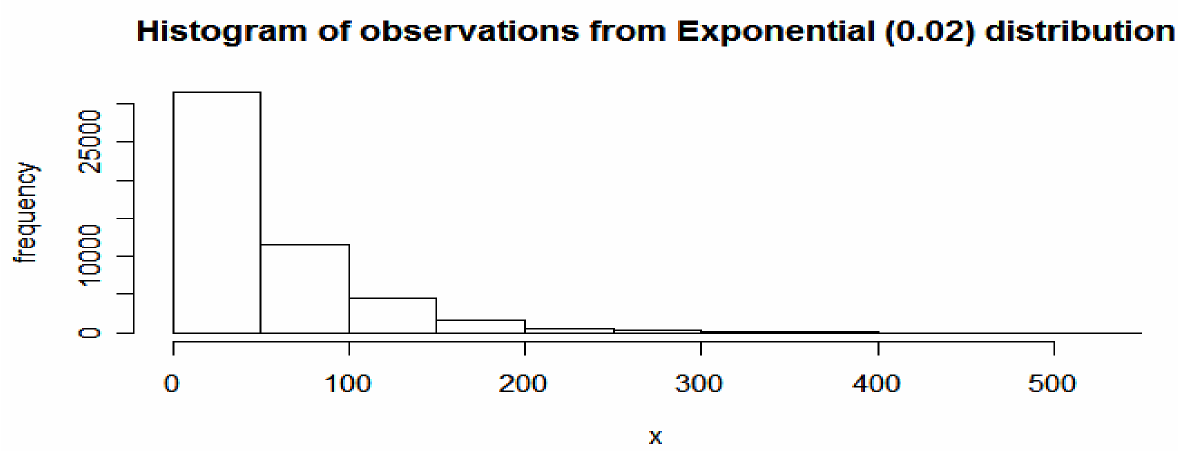
Use Monte Carlo simulation to repeat the experiment.

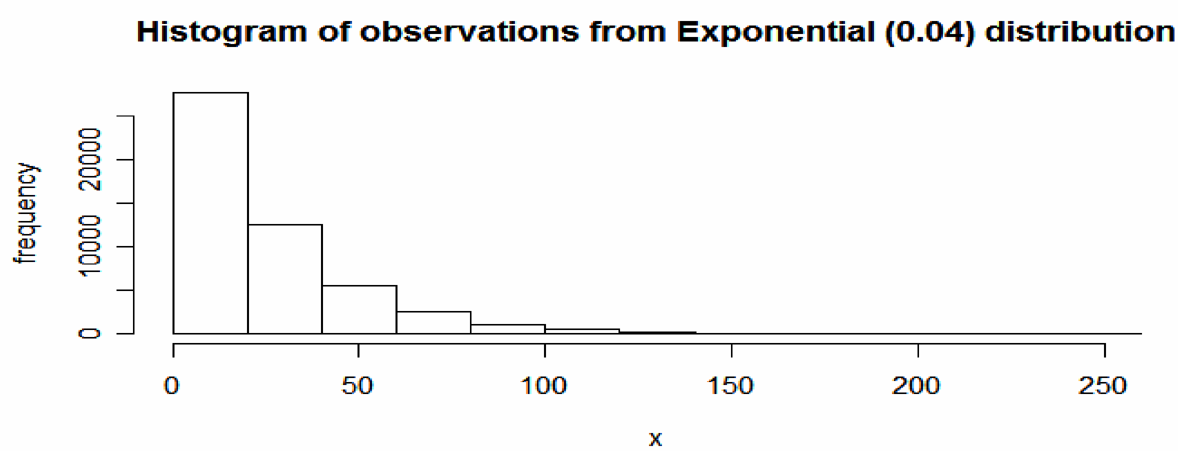
Ex: Suppose an engineer wants to choose between two types of cooling fans to install in a computer. The lifetime of type 1 fan, say X_1, follows exponential distribution with mean 50 months and the lifetime of type 2 fan, say, X_2, follows exponential distribution with mean 25 months. Since fans of type 1 are more expensive, the engineer decides that she will choose type 1 fans if $P(X_1 > X_2) > 0.5$. Find $P(X_1 > X_2)$.

Initialize N = 0.

1. Simulate $X_1 \sim$ Exponential ($\lambda = 1/50 = 0.02$) and $X_2 \sim$ Exponential ($\lambda = 1/25 = 0.04$).
2. If $X_1 > X_2$, N = N + 1
3. Repeat 1 – 2, a large number of times, say 50,000.

Then N/50,000 = observed proportion of times the event $\{X_1 > X_2\}$ happens = estimate of $P(X_1 > X_2)$.





Estimated probability = 33480/50000 = 0.6696
Exact probability = 0.04/(0.02 + 0.04) = 0.667