APMA 3100
Fall 2020

APMA 3100 Project 1

A representative of a high-speed Internet provider calls customers to assess their satisfaction with the service. It takes the representative 6 seconds to turn on a phone and dial a number; then 3 additional seconds to detect a busy signal, or 25 additional seconds to wait for 5 rings and conclude that no one will answer; and 1 second to end a call. After an unsuccessful call, she re-dials (in the course of several days) until the customer answers or she has dialed four times. The outcome of each dialing is determined in an identical way: the customer being called is using the line with probability 0.2; or is unavailable to answer the call with probability 0.3; or is available with probability 0.5 and can answer the call within $\boldsymbol{X}$ seconds, which is a continuous random variable with the mean of 12 seconds and the exponential distribution.

Let $\boldsymbol{W}$ denote the total time spent by the representative on calling one customer. We do not know what type of random variable W is. We need to discover the distribution of W using a Monte Carlo simulation. This will allow us to answer questions such as: What is the probability the representative spends more than 50 seconds calling a single customer? or What are the possible amounts of time the representative could spend calling a customer?
Toward this end, perform the following:

1. Draw a detailed tree diagram/flowchart for the calling process. It should show the different possible outcomes and the probabilities of those outcomes. Because some customers are called multiple times, you may want to organize your diagram as a loop.
due Monday, Oct 12th at 11:59pm

2. **Design and use a Monte-Carlo simulation algorithm** (see appendix below for a description of the Monte-Carlo simulation) You will simulate the calling process 500 times. You will develop an algorithm that generates a realization for W for each of 500 customers. Each realization will be independent from the others, as they will be created using different random numbers. A realization of the calling process should end when the customer answers the call or when four unsuccessful calls have been completed. The way of implementing the algorithm on a computer is up to you.
due Monday, Oct 19th at 11:59pm

2a. Design a random number generator using the description shown in the appendix below. Report the values of u_{51}, u_{52}, u_{53}.

2b. Design an algorithm that uses a random variable to determine whether a customer is busy, unavailable, or available.

2c. Design an algorithm that uses a random variable to determine how much time it takes an available customer to pick up the phone. As part of this, you will need to find the inverse of the cdf of X. (See Continuous Random Variable Generator description in the appendix)

2d. Run your program to produce 500 realizations of W. That is – suppose there are 500 different customers. Use your algorithms to determine – for each of those customers, how long does the representative spend calling that customer? Report those 500 values of W.

3. Discover the distribution of W.

due Friday, Oct 23rd at 11:59pm

3a. **Estimate** from the generated sample of W: (i) the mean; (ii) the first quartile, the median, the third quartile; (iii) the probabilities of events

$$W \leq 15,\ W \leq 20,\ W \leq 30,\ W > 40,\ W > w_5,\ W > w_6,\ W > w_7,$$

where $w_5,\ w_6,\ w_7$ are the values you choose in order to depict well the right tail of the cumulative distribution function of W.

3b. **Analyze** the results and draw conclusions. In particular:

*Compare the mean with the median. What does this comparison suggest about the shape of the probability density function of W?
*Determine the sample space of W.
*Graph the cumulative distribution function of W using the probabilities estimated in step 3a, and interpolating between them whenever appropriate.
*Could W be an exponential random variable? Justify your answer.
* Discuss briefly how the cdf you have created might be useful and to whom.

4. Write your report.

due Friday, Oct 23th at 11:59pm

You should write a report that includes all of the requested information in an order that "flows." Assume that your audience has not read this assignment. Explain your work, summarize the results, and answer all the questions. You should write to an audience that is familiar with programming and probability, but not with your particular programming language. You should explain how your algorithms work in detail, but be concise and DO NOT include your code. Archive your code just in case the grader needs to see it, but the idea is to explain your algorithms well enough in your report so that we won't need to see your code.

Draw figures professionally: to scale, with labels on axes and captions. Number your pages.

You should print the honor pledge on a cover page and each group member must sign the honor pledge. You may discuss the project with the instructor, but all work must be your own.

APPENDIX

MONTE-CARLO SIMULATION SYSTEM

This project will introduce you to the fundamentals of *Monte-Carlo simulation* — a technique of artificially creating samples of random variables which can be large, and which can be used to simulate the behavior of random processes, as well as to solve problems intractable analytically (e.g., those involving multivariate integration or differential equation systems).

In a nutshell, it is a technique for *numerical experimentation* on a computer. While many software packages offer various simulation programs, we shall not use them here. For our objective is to train not as users, but as engineers — the designers and developers of new or specialized systems. And to succeed in such creative tasks, we must understand the fundamentals of system simulation.

A Monte-Carlo simulation system has three main components (Yakowitz, 1977).

1. ***A random number generator*** — a numerical algorithm which outputs a sequence of real numbers from the interval $(0, 1)$, termed *pseudo-random numbers,* which the observer unfamiliar with the algorithm, as well as the standard statistical tests, cannot distinguish from a sample of independent realizations (observations) of a uniform random variable.

2. ***A random variable generator*** — an algorithm that maps any realization of the uniform random variable into a realization of the random variable having a specified cumulative distribution function. (The specification comes from the third component.)

3. ***A mathematical model*** of the process, system, or problem whose behavior or solution is to be simulated.

Yakowitz, S.J. (1977). *Computational Probability and Simulation*, Addison-Wesley, Reading, Massachusetts.

1. Random Number Generator

A popular algorithm, known as the *linear congruential random number generator*, specifies the recursive rule:

$$x_i = (a\, x_{i-1} + c)(modulo\ K), \tag{1a}$$

$$u_i = decimal\ representation\ of\ x_i/K, \tag{1b}$$

for $= 1, 2, 3, \ldots$. The meaning of rule (1a) is: multiply x_{i-1} by a and add c; then divide the result by K, and set x_i to the remainder. Every term is an integer: a and K are positive; c and

x_{i-1} are non-negative. Rule (1b) states that the i^{th} random number u_i is the quotient x_i/K in the decimal representation, which is a real number between 0 and 1.

Every sequence of pseudo-random numbers eventually cycles. To achieve maximum cycle length, the parameters must satisfy certain proven rules. For this project, we chose

starting value (seed)	$x_0 = 1000$,
multiplier	$a = 9429$,
increment	$c = 3967$,
modulus	$K = 2^{14}$.

They yield the cycle of length $K = 2^{14}$. The first three random numbers are 0.7426, 0.3566, 0.3160. Show numbers u_{51}, u_{52}, u_{53} in your report.

2. **Random Variable Generator**

2a. Discrete Random Variable

Let X be a discrete random variable with the sample space $\{x : x = 1, \ldots, k\}$ and a probability mass function $p(x) = P(X = x)$ for $x = 1, \ldots, k$. The corresponding cumulative distribution function F is specified by

$$F(x) = P(X \leq x) = \sum_{y \leq x} p(y), \quad x = 1, \ldots, k. \tag{2}$$

A given random number u_i generates realization x_i of the random variable X via the rule:

$$x_i = \min\{x : F(x) \geq u_i\}. \tag{3}$$

The rule may be executed by searching sequentially over $x = 1, \ldots, k$ until $F(x) \geq u_i$ for the first time.

2b. Continuous Random Variable

Let X be a continuous random variable having a continuous cumulative distribution function F whose inverse F^{-1} exists in a closed-form. That is, $F(x) = P(X \leq x)$, and $x = F^{-1}(u)$ for any $u \in (0, 1)$.

A given random number u_i generates realization x_i of the random variable X through evaluation:

$$x_i = F^{-1}(u_i). \tag{4}$$