Problem 1: Simulating Data from a Poisson Process

You are to first implement an algorithm that can simulate $K$ arrivals from a Poisson process with rate $\lambda$. The easiest way to do this is to simulate each successive inter-arrival time from an exponential distribution, since the inter-arrival times for a Poisson process are exponential.

Thus, you will need to simulate $K - 1$ values from the exponential distribution $\lambda e^{-\lambda x}, x \geq 0$ (you can assign the time of the first arrival to time 0 without loss of generality). To generate these $K - 1$ random values you can use the fact that if $U$ has a uniform distribution between 0 and 1 (as generated by `rand.m`) then the quantity $-\frac{\log U}{c}$ is exponentially distributed with mean $c$. This means you can simulate values from $U$ easily using a standard uniform random number generator and then transform them into a set of exponentially-distributed random numbers—see page 645 in the text for more details.

Your algorithm should take as inputs $K$, $\lambda$, and a state value (or seed) for the random number generator, and return a list of numbers $x_1, \ldots, x_K$ that represent the times of occurrence for a sample of $K$ events from a Poisson process (and as we said above, $x_1$ is set to 0). The “state value” is an integer that you can provide to the uniform random number generator so that when you reset the random number generator with the same state value it produces the same sequence of “random” numbers (this can be useful for debugging and for running experiments).

```matlab
function times = poisson_simulate(K, lambda, state)

% function to simulate K arrival times from a Poisson process

% INPUTS:
```
% K: number of arrival times to generate
% lambda: parameter of the exponential model
% state: positive integer = initial state (seed) for the random number generator
%
% OUTPUTS:
% times: K x 1 vector of arrival times with inter-arrival
% times having an exponential density
%
% ICS 178 Homework 6, 2006

% code follows below......

You should test that your algorithm is working correctly before moving on in the assignment. How can you test? one way is to plot a histogram of the inter-arrival times for a fairly large value of N, e.g., N = 10,000, and verify that it looks like an exponential density function. You can also calculate the mean of the sample points and verify that this mean is reasonably close to the theoretical values for the value of λ you are using. And you could compute the fraction of sample points that lie above a specific time t and see if that agrees with theory, etc.

Problem 2: Simulating an M/M/1 Queue

The second part of the assignment is to implement an algorithm to simulate an M/M/1 queue with infinite queue size. The inputs to this algorithm will be as follows:

1. K, the number of customers to be simulated.
2. The arrival rate λ.
3. The service rate μ.
4. An initial state value for the random number generator.

It will produce as outputs the following:

1. A list of K arrival times of the K customers (again the first arrival time can be 0 without loss of generality).
2. A list of K times at which service was begun for each input event (customer)—so for each customer, these times will be between the arrival time for the customer and the departure time.
3. A list of the K departure times for each of the K customers.
function [times] = MM1_simulation(K, lambda, mu, state)

% function to simulate K customers in an M/M/1 queueing model

% INPUTS:
% K: number of customers
% lambda: parameter of the exponential model for arrival times
% mu: parameter of the exponential model for service times
% state: positive integer = initial state (seed) for the random number generator

% OUTPUTS:
% times: K x 3 vector of times for each customer, where
% times(i,1) = arrival time of customer i
% times(i,2) = start of service time for customer i
% times(i,3) = departure time of customer i

% code follows below......

You can begin your simulation code by generating $K$ arrival times and $K$ service times using the code from the first part of the homework. Note that to generate a list of service times (how long each service lasts) you may want to create a separate function from `poisson_simulate` that returns a list of times drawn from an exponential density (since `poisson_simulate` returns a list of absolute times of events where the time gaps between events are from an exponential density).

To figure out when the service for each customer starts, you will need to sequentially go through the arrival times and keep track of the state of the system. When a new event happens (when a new customer arrives or when the server finishes serving a customer) your simulation code will decide, based on the state of the system, what the next state of the system is (e.g., if a customer’s service finishes, and there are customers in the queue, then the next service starts). Note that to track the state you need to keep track of both how many people are in the queue and whether or not anyone is being served. The only time the state can change is either when the next person arrives or when the server finishes serving someone—in between these events the state cannot change so you only need to look at state changes when these events occur. Implementing this will require some careful thought, but is relatively straightforward once you realize that everything proceeds sequentially based on state changes.
Problem 3: Simulating an M/M/1 Queue with Finite Queue Length

Repeat the last part, but now for a model with a finite queue of size \( N - 1 \) (so no more than \( N \) customers can be in the system at any time). Note that with a finite queue that if the system is full when a new customer arrives, then that customer does not join the system at all and does not ever get service.

function \([s, \text{times}] = \text{MM1\_finite\_simulation}(K, N, \lambda, \mu, \text{state})\)

\%
function to simulate \( K \) customers in an M/M/1 queueing model
\%
with finite queue size \( N - 1 \)
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
\%
What to Submit

Submit 3 MATLAB files and 1 text document containing the answers to the questions below. Submit all files to the EEE online homework folder for Homework 7.

1. Documented code for each of the 3 MATLAB functions above. Please use comments liberally in your code!

2. For the infinite queue, histograms of both the waiting times for each customer and the amount of time spent in the queue for each customer, for $K = 10,000$ (or larger $K$ if you wish) with 50 bins, for each of the following cases (a) $\lambda = 1, \mu = 10$, (b) $\lambda = 2, \mu = 10$, (c) $\lambda = 9.9, \mu = 10$, (d) $\lambda = 0.99, \mu = 10$.

3. For each of the 4 cases in part 2, calculate the mean value of the samples in the histogram and briefly discuss why they are different across the 4 cases.

4. Now compare the mean values in part 3 with the appropriate theoretical values of $W$ and $W_Q$ as discussed in class, and briefly discuss (i.e., they should be close enough that they validate both the theory and the correctness of your simulator).

5. Now for a finite queue, generate histograms of both the waiting times for each customer who entered the system, and the amount of time spent in the queue for each customer who entered the system, for $K = 10,000$ (or larger $K$ if you wish) with (for example) 50 or 100 bins, with $\lambda = 0.9, \mu = 1$ for both (a) $N = 3$ and (b) $N = 15$.

6. For each of the 2 finite queue lengths, calculate the number of customers who did not enter the system (for each simulation), compare to the theoretically-predicted values, and briefly discuss the difference between theory and empirical numbers and between the 2 queue lengths.