Homework 5

Due Thursday, March 5, 2020

Name: ______________________________

Name: ______________________________

Homework submission guidelines:

- Make sure both partner's names are written at the top of the page.
- Submit your solutions as a pdf or Word document to onCourse before 11:59pm on Tuesday. If you would prefer to submit a hard copy, this must be handed in to my office before 4pm on Tuesday.
- All R code used, and any images generated, must be included in your submission.
- All plots must have a title and axis labels.
- **For this assignment:** You will submit R code separately from your write up. I will need to be able to download your code and run it on my computer. You should comment your code *thoughtfully*, so that I can understand what you are doing.

  Your write up should include images and plots along with your written responses. I should be able to read your write up linearly, with the images placed so that I don't have to flip pages or jump around.
- **After** completing the homework, each student must submit the *individual reflection* on OnCourse.

There is only ONE problem this week! But it is a long one. The challenge problems are complete optional.

1. Another inventory model The problem is based on the Inventory Model in Ross Section 7.5.

A certain convenience store is open 24/7 but only stocks one product, which is sold for a price r per unit. Suppose the time between customer arrivals is exponentially distributed with parameter $\lambda_a = 6$ (this translates to 6 customers per hour on average). The quantity demanded by each customer is a Poisson random variable with parameter $\lambda_d = 1$. Note that the outcome of a Poisson random variable can be 0, meaning that customers may enter the shop without buying anything. We assume that sales occur *instantaneously*, meaning that as soon as a customer enters the shop (and has positive demand) then the appropriate number of items are immediately sold.

The shopkeeper uses an (s, S) ordering policy, meaning that whenever the on-hand inventory is less than or equal to s and there is no outstanding order, then an amount is ordered to bring it up to S. The cost of ordering y units of the product is determined by a cost function $c(y)$, and it takes L hours until the order is delivered, with payment being made upon delivery.

Suppose further that whenever a customer demands more of the product than is presently available, then the amount on hand is sold and the remainder of the order is lost to the shop.

(a) First define the model *parameters* by entering the following.

```
> lambda_a <- 6
> lambda_d <- 1
> r <- 1.3
> cost <- function(y){20 + y}
> s <- 150
> S <- 350
> L <- 24
```

(b) In this problem we will practice *Discrete Event Simulation*, meaning that we keep track of several variables over time. Whenever an "event" occurs, the values of the variables are updated and we record any relevant data of interest. In order to do this, we must keep track of an "event list", which is a list of the nearest future events and when they are scheduled to occur. This allows us to follow a system as it evolves over time.

There are two types of events in this simulation: (1) customer arrivals (with instantaneous sales) and (2) delivery of new stock. We will need to keep track of the time of the next customer arrival, `t_arrival`, as well as the time of the next order delivery, `t_delivery`. We will move along the time axis until we encounter the next event and then update the system accordingly.

The variables we are interested in are

- `t`, the total amount of time that has elapsed.
- `profit`, which we can think of as the store's current bank account balance.
- `stock`, the number of products currently on hand.
- `order`, the amount of product currently on order.

Initialize the system with

```
> t <- 0
> profit <- 0
> stock <- S
> order <- 0
```

We will also need to keep track of the "next" events. Assuming we start our simulation at time 0, the first customer arrival is an exponential random variable. We set the time of the next arrival to be

```
> t_arrival <- rexp(1,6)
```

There are currently no inventory orders out, so we set the time of the next delivery to be ∞.

```
> t_delivery <- Inf
```

We will record all of our data in a matrix with six columns, which you can initialize as

```
> mat <- c(t,t_arrival,t_delivery,profit,stock,order)
```

(c) Suppose the next event is a customer arrival. How will each of the variables and event times change after this event?

(d) Suppose the next event is an order delivery. How will each of the variables and event times change after this event?

(e) **Simulate the update step:** write R code that uses the "current" values of the variables and event times to determine the "next event". Update the variables and event times accordingly.

At the end of the update step, add the current variables and event times to the bottom of the matrix:

```
> mat <- rbind( mat, c(t,t_arrival,t_delivery,profit,stock,order) )
```

(f) Use the code from your update step to simulate the system for 720 hours (30 days).

Hint: you may wish to use a `while` loop. Alternately, you can use a `for` loop along with the `break` command.

Hint#2: After you generate data, you may wish to write your data to a .txt or .csv file so that you can access it later. The command is

```
> write( mat , [filename] )
```

or

```
> write.csv( mat , [filename] )
```

(g) Respond to each of the following questions in a few sentences. Back up your discussion with plots and statistical analyses of your simulation data.

i. What does the store's inventory look like over time?

ii. What does the store's profit look like over time? What is the total monthly (30-day) profit in this simulation?

iii. How many orders are lost over the course of 30 days? Can you determine this from the data we have? If so, then find the number of lost orders. If not, make an educated guess about how many orders are lost, based on the data we have?

(h) Now suppose that the shop pays an inventory holding cost of h per unit item per hour. Set the parameter

```
> h <- 0.01
```

and adapt your code to account for this holding cost.

(i) Run a new 720 hour simulation taking the holding cost into account. Repeat part (g) with your new simulation results.

(j) The only parameters that the shopkeeper can control are s and S. (We are assuming that the revenue r per item is fixed. This may be true if the store is part of a franchise, for instance.)

Repeat the 720 hour simulation with several different values of (s, S) and repeat your analyses. After trying several different combinations, identify values of s and S so that:

i. The store is unlikely to ever dip into the negative.

ii. The store is unlikely to ever lose customers.

iii. The expected monthly profit is optimized.

(k) Respond to each of the following questions in a few sentences.

i. Is this a realistic model? How would you adapt the model to make it more realistic? How might the model need to be adapted for different types of stores?

ii. What other questions do you have that we can answer with this model and simulation? What other questions do you have that we could answer after ADAPTING our model and/or simulation?

Challenge #1 Now suppose the shopkeeper can control revenue r per item as well as s and S. How might we change our model if we are allowed to increase or reduce prices? Think about supply and demand.

Challenge #2 How would you adapt the model and simulation if the store sold multiple products?