Stochastic Processes
Universidad Carlos III
December 2019
Rafaela Becerra & Marta Cortés

Poisson process and continuous-time Markov chains

1. *Problem 1* Use the R package retweet to get retweets times for a twitter account of your choice

   a) Use descriptive statistics and graphics to explore the retweets data set in terms of the number of retweets by time and the time between retweets. Does it fit the hypothesis of a Poisson process?

   In order to get the more accurate information about the tweets and its retweets of our selected account "Celtics" basketball team by time, we have selected 200 tweets during the month of November, 2019. Since it means they are new enough to have a good reference of its time behaviour and old enough to have almost all the retweets completed.

   After getting 200 tweets by the function "*get_timeline*", we have extracted their corresponded retweets (max. 100) and its date and time by the function "*get_retweets*", creating the data that we are going to use in this exercise.

   In order to complete the data, we have calculated the difference in time between each retweet and its corresponding tweet and the difference in time between each retweet and its previous one in order to plot and visualize the distribution of the retweets by time and the time between retweets.

   For doing this we have assumed that the obtained retweets were the first to be generated when the tweet was posted (although we are pretty sure it fulfil that).
   Also, we have treated the total retweet data as completed set of retweets dismissing the consideration that individual tweet can behave different than other according to the day of the week, the moment in the League or the hour of a day.
   Finally, we have taken the first retweet of the tweet as time 0 in order to calculate from this the inter-arrival times between the retweets.

   The next figures can show the tendency of the retweets by the posted time (we have only plot the first 24 hours in order to get better visualization.

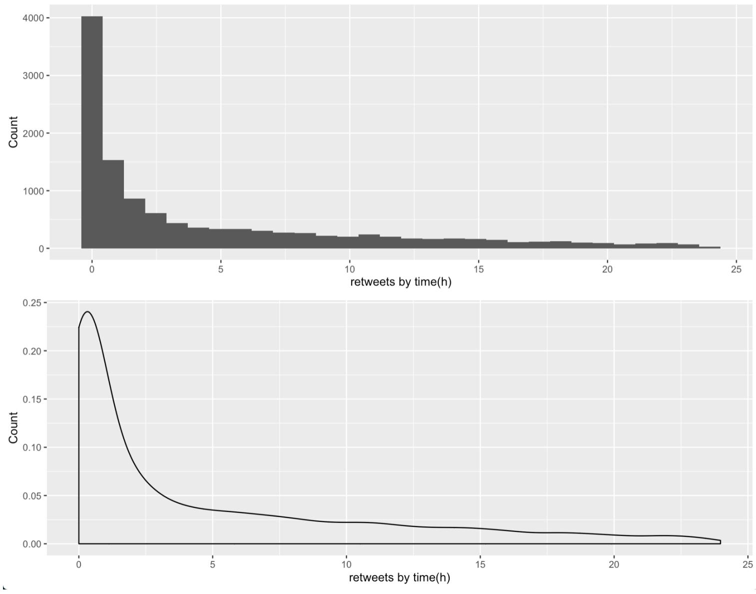

Figure 1: Distribution of the retweets by time

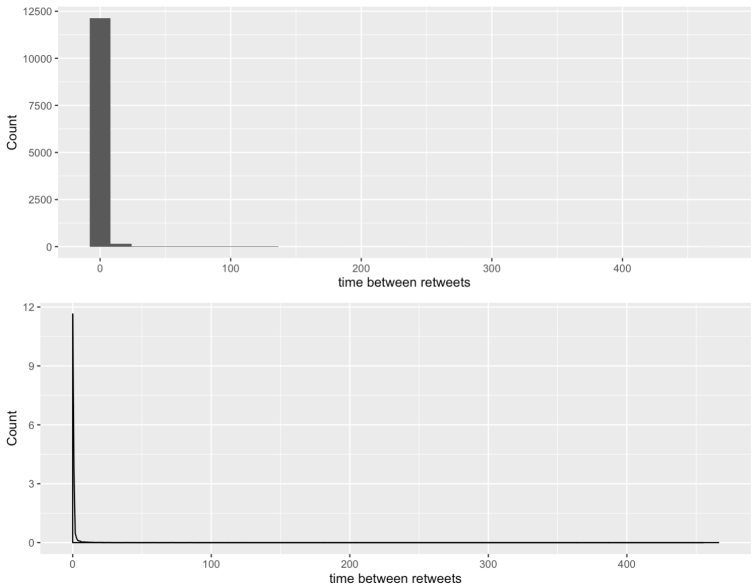

Figure 2: Distribution of the time between retweets

From the graphs we can observed that the time between the retweets (inter-arrival times) clearly follows a exponential distribution, also, we assume they are independent random variables, so we can hypothesize that our case (the numbers of retweets observed at time t) is a Poisson process.

b) Assuming you can model the number of retweets by time as a non-homogeneous Poisson process with intensity function $\lambda(t) = \theta e^{-\theta t}$, t > 0, graphically explore possible values of θ and choose the one that best fit your data. Explain all the considerations you make.

A non-homogeneous Poisson process is a counting process that fulfil the next characteristics:

- $N_0 = 0$
- N_t has a Poisson distribution with mean $E[N_t] = \int_0^t \lambda(s)ds$ for all $t > 0$
- Independent increments
- It has an intensity function

In order to find the best parameter which fit better the model, we have found first our λ for a hypothetical time-homogeneous counting process which we assume should be similar to the one for non-homogeneous Poisson process.

Then, we have used it as a reference for finding, graphically, the best θ that satisfy the intensity function by giving values to it, across a range.

This assumed λ has been obtained from our data as the arrival time when the first retweet is posted. We have calculated a mean of all ours first retweets (dismissing some outliers). It is equal to 0.266 hours ($\lambda = 0.266$).

Then,

$$\lambda = \theta e^{-\theta t}$$

$$0.266 = \theta e^{-\theta t}$$

Running a code in R giving values to θ for finding the equality to 0.266, we have found that the optimal value for the homogeneous distribution parameter is between 0.39 and 2.07.

Finally, we have plotted few graphs for different values of theta in a range contained the previous one, in order to be sure that we find our optimal non-homegeneous parameter with that range.

As we can see in the next figure, where we have plotted the initial distribution versus our intensity function, the θ parameter that better fit our non-homogeneous distribution is 0.2 ($\theta = 0.2$).

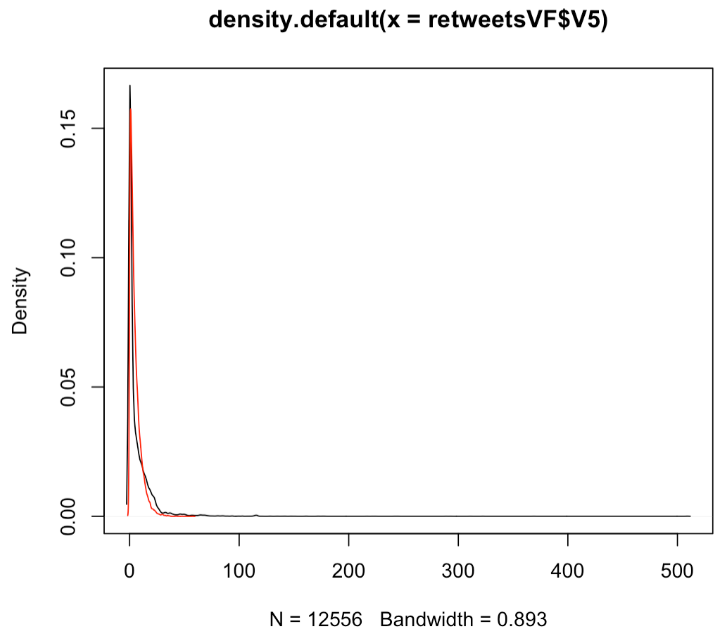

Figure 3: Fitted distribution by theta=0.2

c) Write an R function to simulate data from a non-homogeneous Poisson process during a given time interval (provide the code). Describe the inputs and outputs of your function.

The code for the simulation is shown in next lines:

```
set.seed(0)
rates<- function(l,t){
  k<- rpois(1,t*l)
  u<- runif(k)
  s<- t*sort(u)
  N<- cumsum(c(0,rep(1,k)))
  return(list(s=s,N=N))

}

arrival_times=rates(5,3)

# Lambda fucntion
Landa=function(t) (0.2*exp(-0.2*t))


NHPP=function(P) {
  x=0
  y=c(0)
```

```
  for (i in 1:length(P)){
    x=sample(c(1,0),1,prob=c((Landa(P[i])/0.2),(1-Landa(P[i])/0.2)))
    y=c(y,y[i]+x)
  }
  return (y)
}

P=arrival_times$s
I=NHPP(P)
PI=data.frame(I[-1],P)
```

The inputs of our function are the arrival times that have been generated with the initial function of the regular Poisson process called as "rates" in our code.

This function return a vector with all the realizations of each time.

On the other hand, the outputs of the function are event times between a range of time (0,S].

In the next figures, we can see the histogram and the cumulative distribution of the simulation.

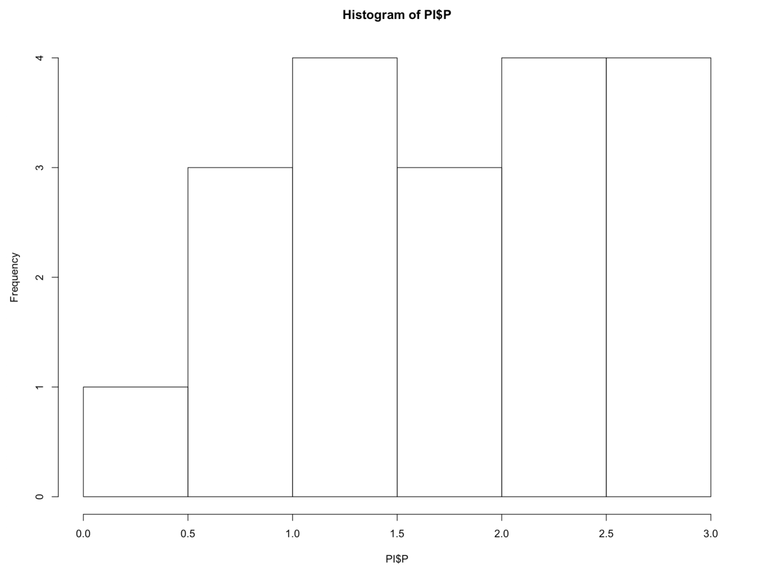

Figure 4: Histogram of the simulated NHPP distribution

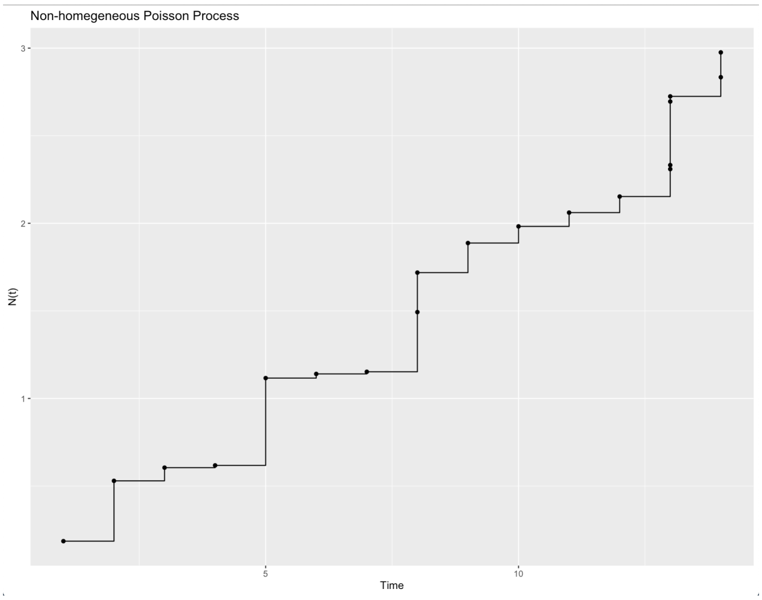

Figure 5: Cumulative function of NHPP simulated

d) Simulate data using the previous function trying to mimic the real retweets data set. Compare the distribution of arrival times for the real and simulated data set. Comment on the differences.

The code for the simulation is detailed in the next lines:

```
Dist=rates(400,24)
S_Dist<-Dist$s
N_Dist=NHPP(S_Dist)
DS<- data.frame(N_Dist[-1],S_Dist)
N_time<-vector()
H=c(0:24)
for (i in 1:length(H)) {
  for (j in 1:nrow(DS)) {
    if(DS$S_Dist[j]>=H[i]) {
      N_time=c(N_time,DS$N_Dist[j])
      break
    } else if (j==nrow(DS)) {
      N_time=c(N_time,DS$N_Dist[j])
    }
  }
}

N_arrivals <- data.frame(H,N_time)
```

In the next first figure we can see the exponential distribution of the simulation and the one for our initial distribution in the second, as we can see, they are very similar, except for the fact that the original one has the slope a bit more straight and the simulated has a more strong curve .

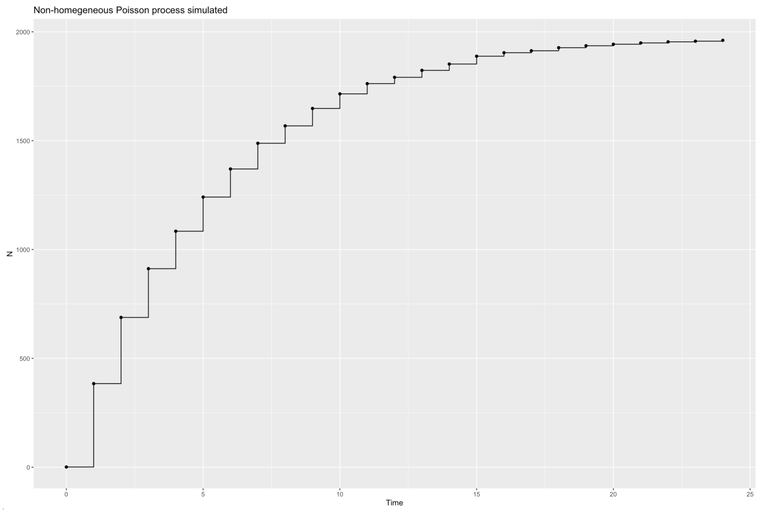

Figure 6: Cumulative function of simulation

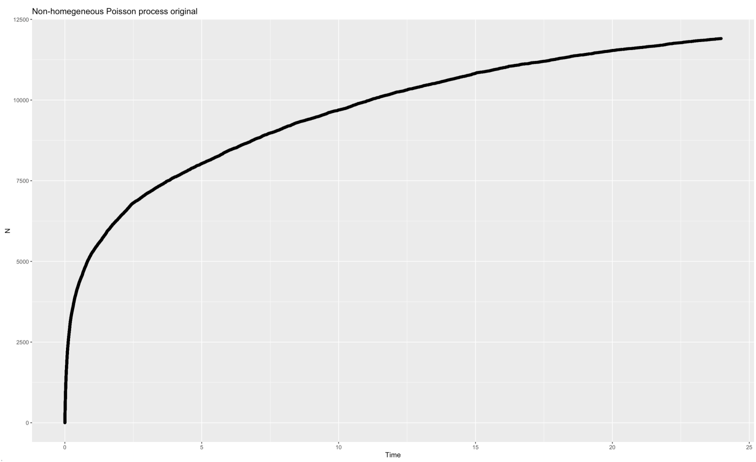

Figure 7: Cumulative function of original

e) Using the intensity function of part b), compute the probability that a tweet from this user will get more than 10 retweets in the first hour, and the expected number of retweets of a tweet after 24 hours.

$\lambda(t) = \theta e^{-\theta t}$, t > 0
$\lambda(t) = 0.2e^{-0.2t}, t > 0$

Nt, is the number of retweets of a tweet.

$$P\left(N_{3600} > 10\right) = 1 - P\left(N_{3600} <= 10\right) = \left(1 - \sum_{k=0}^{10} \frac{e^{-1}(1)^k}{k!}\right) = 1.005e - 08$$

$$E\left[N_{3600}\right] = \int_0^{3600} \lambda(t)dt = \int_0^{3600} 0.2e^{-0.2t}dt = 1$$

$$E\left[N_{86400}\right]=\int_0^{86400}\lambda(t)dt=\int_0^{86400}0.2e^{-0.2t}dt=1$$

2. *Problem 2*

Consider a supermarket with 2 cashiers. Customers arrive to the unique cashiers waiting line according to a Poisson process with rate . The times to be served (check-out times at the cashier) are independent and distributed as exponential random variables with rate . Let us assume that the queue can accomodate an unlimited number of waiting customers. This system is known as the M/M/2 queue (see section 7.6 of Dobrow, 2016). Let Xt denote the number of customers in the system (at the cashiers or in the waiting line) at time t.

a) What kind of process is X $=\{X_t, t\geq 0\}$ This is a continuous Markov time reversible chain, that can be seen as a birth-and-death process with constant birth and death rates, where X $=\{X_t, t\geq 0\}$ with countable state space $S=\{0\}\bigcup N$

Xt is the number of customers in the system i

$$X_t=i\begin{cases} i+1, & \text{if a new costumer joins the system} \\ i-1, & \text{if a new costumer is charged and leaves the system}\end{cases}$$

The arrival rate is λ, this follows an exponential distribution, and the time until a costumer is served is the minimum of the time of charging by cashier 1 and by charging cashier 2, both follow a exponential distribution, because the service time distributions of each server are independent and identical, the time until a costumer is served follows also an exponential distribution with rate 2μ.Then, Ti which is the total time the system spends on state i after a transition into i, i ϵ S, follows an exponential distribution and will be given by the min of the arrival time of a new customer or the check out of other.

Q is known as the infinitesimal generator of the Markov chain and it fully characterizes its behaviour.The Q matrix is form by qij that are called the instantaneous transition rates of the process, and give the time spent by the system in i number of customers before moving to j state.

Q=

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ \vdots \\ n-1 \\ n \end{array}
\begin{array}{c}
\begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & \dots & n-1 & n \end{array} \\
\left(\begin{array}{cccccccc}
-\lambda & \lambda & 0 & 0 & 0 & \dots & 0 & 0 \\
\mu & -(\mu+\lambda) & \lambda & 0 & 0 & \dots & 0 & 0 \\
0 & 2\mu & -(2\mu+\lambda) & \lambda & 0 & \dots & 0 & 0 \\
0 & 0 & 2\mu & -(2\mu+\lambda) & \lambda & \dots & 0 & 0 \\
0 & 0 & 0 & 2\mu & -(2\mu+\lambda) & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & \dots & -(2\mu+\lambda) & \lambda \\
0 & 0 & 0 & 0 & 0 & \dots & -\lambda & -(2\mu+\lambda)
\end{array}\right)
\end{array}$$

The rate μ_i will be given by:

$$\mu_i=\begin{cases} i\mu, & for\text{i=1} \\ 2\mu, & \text{for } i>=2\end{cases}$$

b) Compute its stationary distribution. What is the expected number of people in the system (checking-out or in the waiting line) in the long-run?

π is a stationary distribution for the chain, which means is time reversible, and the local balance equation is satisfied. Then,

$\pi_i q_{ij} = \pi_j q_{ji}, \quad$ for all $i, j \in S$

And, for matrix Q

$i = 0, j = 1$ $\pi_0 * q_{01} = \pi_1 * q_{10}$ Then, $\pi_1 = \frac{\pi_0 \lambda}{\mu}$

$i = 1, j = 2$ $\pi_1 * q_{12} = \pi_2 * q_{21}$ Then, $\pi_2 = \frac{\pi_1 \lambda}{\mu} = \frac{\pi_0 \lambda}{2\mu} * \frac{\lambda}{2\mu}$

$i = 1, j = 3$ $\pi_2 * q_{23} = \pi_3 * q_{32}$ Then, $\pi_3 = \frac{\pi_2 \lambda}{2\mu} = \frac{\pi_0 \lambda}{2\mu} * \frac{\lambda}{2\mu} * \frac{\lambda}{2\mu}$

And, so on.

Consequently, the stationary probabilities can be expressed as:

$$\pi_k = \begin{cases} \frac{\pi_0}{k!} \left(\frac{\lambda}{\mu}\right)^k, & for 0 \leq k < 2 \\ \frac{\pi_0}{2^{k-2} 2!} \left(\frac{\lambda}{\mu}\right)^k, & for k \geq 2 \end{cases}$$

In the long-run, the expected number of people in the system L can be expressed as $L = \lambda W$

And, the W can be divided as the time people spend waiting W_q and paying W_s. The time of checking out is the mean of the exponential distribution with parameter μ.

$W = W_q + W_s$

Consequently,

$W = W_q + \frac{1}{\mu}$

And, by restricting the system just to the waiting part:

$L_q = \lambda W_q$

Where, L_q is the long-term average number of people waiting in the line.There will be k costumers in the line if k if and only if there are k + 2 customers in the system.

$$L_q = \sum_{k=2}^{\infty} (k-2) \pi_k$$

$$L_q = \sum_{k=2}^{\infty} (k-2) \frac{\pi_0}{2^{k-2} 2!} \left(\frac{\lambda}{\mu}\right)^k$$

$$L_q = \frac{\pi_0}{2!} \left(\frac{\lambda}{\mu}\right)^2 \sum_{k=0}^{\infty} k \left(\frac{\lambda}{2\mu}\right)^k$$

$$L_q = \frac{\pi_0}{2!} \left(\frac{\lambda}{\mu}\right)^2 \frac{\lambda}{2\mu} \left(\frac{1}{1 - \lambda/2\mu}\right)^2$$

And p, is the probability that the cashier is busy or the proportion of time the cashiers are busy.

$p = \frac{\lambda}{2\mu}$

In terms of p:

$$L_q = \frac{\pi_0}{2} (2p)^2 p \left(\frac{1}{1-p}\right)^2$$

$$L_q = \frac{\pi_0 2\left(p\right)^3}{\left(1-p\right)^2}$$

$$L_q = \frac{(1-p)2\left(p\right)^3}{\left(1+p\right)\left(1-p\right)^2}$$

$$L_q = \frac{2p^3}{(1-p^2)}$$

$$W_q = \frac{L_q}{\lambda}$$

$$W_q = \frac{\frac{2p^3}{(1-p^2)}}{\lambda}$$

$$W_q = \frac{\frac{2p^3}{(1-p^2)}}{2\mu p}$$

$$W_q = \frac{2p^3}{(1-p^2)2\mu p}$$

$$W_q = \frac{p^2}{\mu(1-p^2)}$$

$$W = W_q + \frac{1}{\mu}$$

$$W = \frac{p^2}{\mu(1-p^2)} + \frac{1}{\mu}$$

$$W = \frac{p^2+1-p^2}{\mu(1-p^2)}$$

$$W = \frac{1}{\mu(1-p^2)}$$

Finally L, or the expected number of people in the hole charging system will be:

$$L = \lambda W$$

$$L = \lambda \frac{1}{\mu(1-p^2)}$$

$$L = \frac{2p\mu}{\mu(1-p^2)}$$

$$L = \frac{2p}{(1-p^2)}$$

And the long-term probability that no costumer will be in the system is given by:

$$\pi_0 = \left(\sum_{k=0}^{2-1} \left(\frac{\lambda}{\mu}\right)^k \frac{1}{k!} + \frac{(\lambda/\mu)^2}{2!} \left(\frac{1}{1-\lambda/2\mu}\right)\right)^{-1}$$

In terms of p can be expressed as:

$$\pi_0 = \left(1 + 2p + 4p^2 \frac{1}{2}\frac{1}{1-p}\right)^{-1}$$

$$\pi_0 = \left(1 + 2p + 2p^2 \frac{1}{1-p}\right)^{-1}$$

$$\pi_0 = \left(\frac{1-p+2p-2p^2+2p^2}{1-p}\right)^{-1}$$

$$\pi_0 = \left(\frac{1-p}{1+p}\right)$$

$$\pi_0 = \left(\frac{2\mu-\lambda}{2\mu+\lambda}\right)$$

c) We say that overtaking occurs when a customer departs the supermarket before another customer who arrived earlier to the waiting line. In steady state, find the probability that an arriving customer overtakes another customer (you may assume that the state of the system at each arrival instant is distributed according to the stationary distribution). One customer can overtake another one, if there is at least one customer in the system. Since there are two cashiers, in order to the overtaking occurs, there has to be one cashier free, and then the probability that the new costumer can overtake given that when he arrives sees 1 customer in the system will be given by:

$P\left(OV = 1 | SC = n\right) = \frac{1}{2}$

Where OV is the number of customers that the costumer can overtake, and SC the number of clients that this customer sees in th system.

This occurs because of the memory-less property of the process.

Now if we consider the probability that one customer overtakes another one. This can be expressed by:

$$P(OV = 1) = \frac{1}{2}\sum_{k=1}^{\infty} \Pi_k$$

Or equivalent to say that:

$$P(OV = 1) = \frac{1}{2}(1 - \Pi_0)$$

$$P(OV = 1) = \frac{1}{2}\left(1 - \left(\frac{2\mu-\lambda}{2\mu+\lambda}\right)\right)$$

$$P(OV=1)=\frac{1}{2}\left(\frac{2\mu+\lambda-2\mu+\lambda}{2\mu+\lambda}\right)$$

$$P(OV=1)=\left(\frac{\lambda}{2\mu+\lambda}\right)$$

When the k tends to go to infinite then $\sum_{k=1}^{\infty}\Pi_k$ will tend to be 1, and the probability will remain in 1/2.

d) Write the R code necessary to simulate the system (provide the code) and generate the times customers leave the supermarket. For the simulation we have repeated the process of this system for 1000, each time we projected 1000 of costumers. The general considerations are the ones explained before:

Xt:denote the number of customers in the system (at the cashiers or in the waiting line) at time t

Ti: total time the system spends on state i after a transition into i, i ϵ S, which is the min between:

-New arrival $\sim$ exp(λ)

-min (Time of charging 1 $\sim$ exp(μ), Time of charging 2 *sim* exp(μ)) $= \sim$ exp(2μ)

The code calculates the times for each costumer of arrival, considering that the process starts at time 0, and the distribution of the interval of arriving. Also, calculates the services time base of the distribution also, and the waiting time, given by the time it takes for the next cashier to be free. Then it calculates the probability of overtaking.

```
#Parameters needed
lambda
mu
#Simulation
pov=rep(0,1000)
system=rep(0,1000)
for (i in 1:1000){
  sim=1000 #number of clients
  interar=rexp(sim,lambda) #interarrival times
  arrivals <- cumsum(interar)  #time of arrival of the client
  service=rexp(sim,mu) #time of been charged
  departure=rep(0,2) #time of departure os the clients that occupied ea
  tout=rep(0,sim) #time of departure of each costumer
  count=0

  #For the first two clients, the cashiers are empty
  departure[1]=arrivals[1]+service[1]
  tout[1]=departure[1]

  departure[2]=arrivals[2]+service[2]
  tout[2]=departure[2]
```

```
    #From the third costumer in advanced there could be a waiting time
    for (k in 3:sim){
      #The waiting time will be the time between the arrival until one c
      waiting1=min(departure[1], departure[2])
      if(min(departure[1], departure[2]) < arrivals[k]){
      waiting=0
      }else{waiting=min(departure[1], departure[2]) - arrivals[k]}
      tout[k]=arrivals[k]+waiting+service[k]
      departure[which.min(c(departure[1], departure[2]))]=  tout[k]
    }

    #Calculation of the part of the clients that had overtaken another c
    for(j in 2:length(tout)){
      if (any(tout[j]<tout[1:j-1])){count=count+1}
    }
    pov[i]=count/sim

  }
  #Mean of leaving time
  mean(pov)
```

e) Assuming 2 customers arrive to the cashiers every 5 minutes on average and it takes an average time of 4 minutes to check-out, estimate through simulation the probability of overtaking and compare it with the result you got in part c). Also estimate the long-run average number of people in the system and compare it with the result of part b). Explain all the considerations you make and the simulation setting (provide the code).

New arrivals $\sim$ exp(λ=0.4)

Time of charging $\sim$ exp(2μ=0.5)

The probability of overtaking, according to what we got in the section c, could be define as:

$$P(OV=1) = \left(\frac{\lambda}{2\mu+\lambda}\right) = \left(\frac{0.4}{0.5+0.4}\right) = 0.44$$

The code provided in the section d estimates the time of arrival of the customer, the time of waiting and the time of paying, by estimating the part of clients which arrive early than a certain client and leave after, we can obtain the probability of overtaking. The simulation performed, give the same answer as before, the probability will be of 44

```
  lambda <- 2/5
  mu <- 1/4
  pov=rep(0,1000)
  system=rep(0,1000)

  for (i in 1:1000){
    sim=1000 #number of clients
    interar=rexp(sim,lambda) #interarrival times
```

```
arrivals <- cumsum(interar)  #time of arrival of the client
service=rexp(sim,mu) #time of been charged
departure=rep(0,2) #time of departure of the client that occupied the
tout=rep(0,sim) #time of departure of each costumer
count=0

#For the first two clients, the cashiers are empty
departure[1]=arrivals[1]+service[1]
tout[1]=departure[1]

departure[2]=arrivals[2]+service[2]
tout[2]=departure[2]

#From the third costumer in advanced there could be a waiting time
for (k in 3:sim){
  #The waiting time will be the time between the arrival until
  #one cashier gets empty for the client
  waiting1=min(departure[1],departure[2])
  if(min(departure[1],departure[2])<arrivals[k]){
  waiting=0
  }else{waiting=min(departure[1],departure[2])-arrivals[k]}
  tout[k]=arrivals[k]+waiting+service[k]
  departure[which.min(c(departure[1],departure[2]))]= tout[k]
}

#Calculation of the part of the clients that had overtaken
#another costumer
for(j in 2:length(tout)){
  if (any(tout[j]<tout[1:j-1])){count=count+1}
}
pov[i]=count/sim

}
#Mean of leaving time
mean(pov)
```

The long-term average number of people in the system is given by L, as define in the section b:

$$L = \frac{2p}{(1-p^2)}$$

And the proportion of time the client spends in the system is:

$$p = \frac{\lambda}{2\mu} = \frac{0.4}{0.5} = 0.8$$

Then,

$$L = \frac{2p}{(1-p^2)} = \frac{2 * 0.8}{(1-0.8^2)} = 4.44$$

In order to simulate the long-run of the expected amount of clients in the system, we have simulated the transition matrix Q, for 100 clients, and calculated the stationary distributions, with them we can obtain the expected amount of time that the process remains in a state understood as a certain amount of clients. The code is provided below, and the answer as expected is the same as the one obtain with the previous formula.

```
k=1000#Number of projected clients
lamb=2/5 #rate of arrivals
mu=1/4 #time of charging

#Infinitesimal generator matrix Q
Q=matrix(0,nrow=k+1,ncol=k+1)
for (i in 1:k){
  Q[i,i+1]=lamb
  if (i==1){
    Q[i,i]=-(lamb)
  }
  if (i==2){
    Q[i,i-1]=mu
    Q[i,i]=-(lamb+(Q[i,i-1]))
  }
  if (i>2){
    Q[i,i-1]=2*mu
    Q[i,i]=-(lamb+(Q[i,i-1]))
  }
}
Q[k+1,k+1]=-k
Q[k+1,k]=k

stat_CTfromQ<-function(Q){ #Computes the stationary distribution of a
#CT Markov chain
  #Q is the infinitesimal generator of a CT Markov chain
  k=nrow(Q)
  b=c(1,rep(0,k-1))
  Q[,1]=rep(1,k)
  pi=solve(t(Q),b)
  return(pi)
}

pi<-stat_CTfromQ(Q)
expected_value=sum(pi*seq(0,1000))
print(expected_value)
```

Aditionally, based on the code of the simulating times of departure, we have generated a matrix in order to check the system increments and decrements of clients over time. If we simulate this process again 100, each with a projected amount of 1000 coustumers, we get the same results, that the expected number of clients is 4.4.

The code is shown below.

```
########## number of expected clients in the system
#set.seed(1)
lambda=2/5
mu=1/4
pov=rep(0,700)
exp_v=rep(0,700)

for (i in 1:700){
  sim=1000 #number of clients
  interar=rexp(sim,lambda) #interarrival times
  arrivals <- cumsum(interar)  #time of arrival of the client
  service=rexp(sim,mu) #time of been charged
  departure=rep(0,2) #time of departure os the clients that occupied ea
  tout=rep(0,sim) #time of departure of each costumer
  count=0
  totalc=rep(0,20) #time of departure of each costumer

  #For the first two clients, the cashiers are empty
  departure[1]=arrivals[1]+service[1]
  tout[1]=departure[1]

  departure[2]=arrivals[2]+service[2]
  tout[2]=departure[2]

  #From the third costumer in advanced there could be a waiting time
  for (k in 3:sim){
    #The waiting time will be the time between the arrival until one c
    waiting1=min(departure[1],departure[2])
    if(min(departure[1],departure[2])<arrivals[k]){
    waiting=0
    }else{waiting=min(departure[1],departure[2])-arrivals[k]}
    tout[k]=arrivals[k]+waiting+service[k]
    departure[which.min(c(departure[1],departure[2]))]= tout[k]
  }

  #Calculation of the part of the clients that had overtaken another c
  for(j in 2:length(tout)){
    if (any(tout[j]<tout[1:j-1])){count=count+1}
  }

  pov[i]=count/sim

  #Extract the data from the arrivals and from the
  #time of departure from
  #the system, then we  form a sequence in order to record how many cli
  #were in the system and for how long the system remain that way
  #we do this for each simulation
```

```
    arrivals1=as.data.frame(arrivals)
    tout1=as.data.frame(tout)
    arrivals1=mutate(arrivals1,p=1)
    tout1=mutate(tout1,p=-1)
    names(tout1)[1]="arrivals"
    arrivals1=rbind(arrivals1,tout1)

    arrivals1=arrivals1[order(arrivals1$arrivals),]
    arrivals1[, 3] <- cumsum(arrivals1[, 2])
    arrivals1[,4]=c(diff(arrivals1$arrivals, lag = 1, differences = 1),0)
    arrivals1=mutate(arrivals1, p2=V3*V4)

    exp_v[i]=sum(arrivals1$p2)/max(arrivals1$arrivals)

}
```