Stochastic Processes: Assignment 1

Group 1: Javier Esteban Aragoneses, Mauricio Marcos Fajgenbaun, Danyu Zhang, Daniel Alonso

November 27th, 2020

Importing libraries

```
#> Package:  markovchain
#> Version:  0.8.5-2
#> Date:     2020-09-07
#> BugReport: https://github.com/spedygiorgio/markovchain/issues
#>
#> Attaching package: 'dplyr'
#> The following objects are masked from 'package:stats':
#>
#>     filter, lag
#> The following objects are masked from 'package:base':
#>
#>     intersect, setdiff, setequal, union
```

Problem 1

```
#> [1] "n bnq ndrsw awornoc aslnxs rzs dowfr ksfrswc ljicrwq rj nirzjwoms n ljwjcngowif gnllocs aworofz
```

Problem 2

(a)

Let $N(t)$ be the number of cars arriving at a parking lot by time t, according to the proposed scenario, we can model $N(t)$ as a non-homogenous Poisson process. Such process has almost the same process as any other Poisson process, however, its rate is a function of time.

$N(t), t \in [0, \infty)$ is the non-homogenous Poisson process with rate $\lambda(t)$ where:

- $N(0) = 0$
- $N(t)$ has independent increments

We define 8:00 as $t = 0$ with the following integrable function and each unit of t equals to 1 hour:

$$\lambda(t) = \begin{cases} 100 & 0 \le t \le \frac{1}{2} \\ 600t - 200 & \frac{1}{2} < t \le \frac{3}{4} \\ 400t - 50 & \frac{3}{4} < t \le 1 \\ -500t + 850 & 1 < t \le 1.5 \end{cases}$$

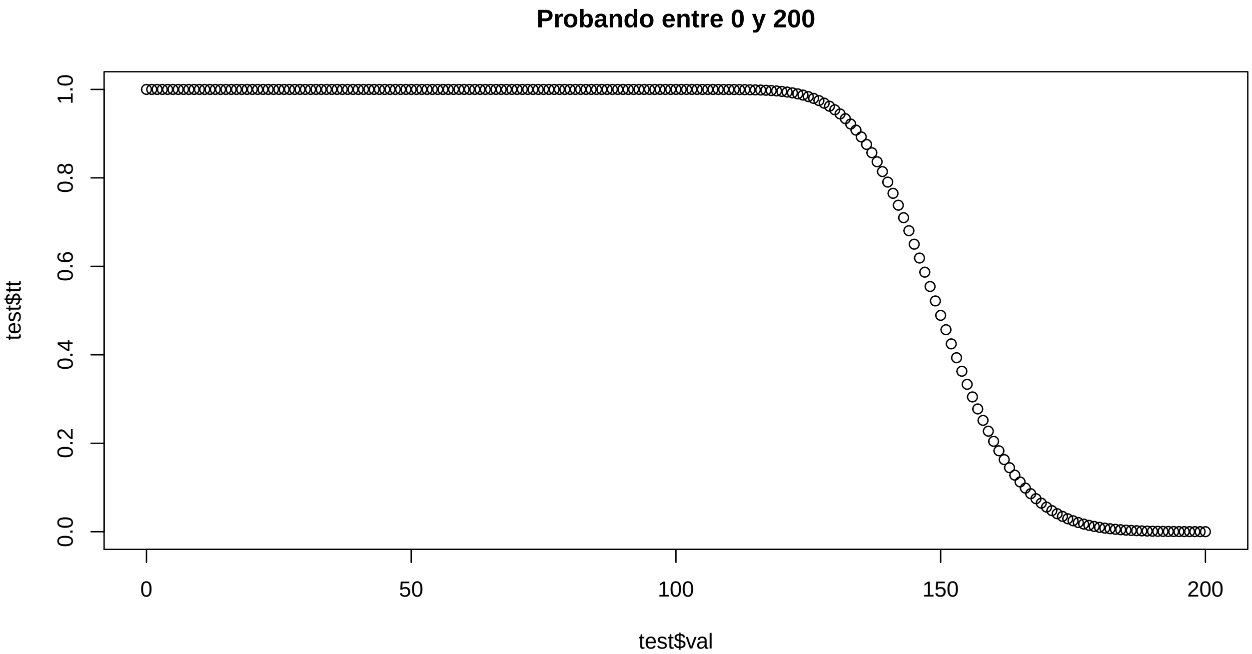
So,

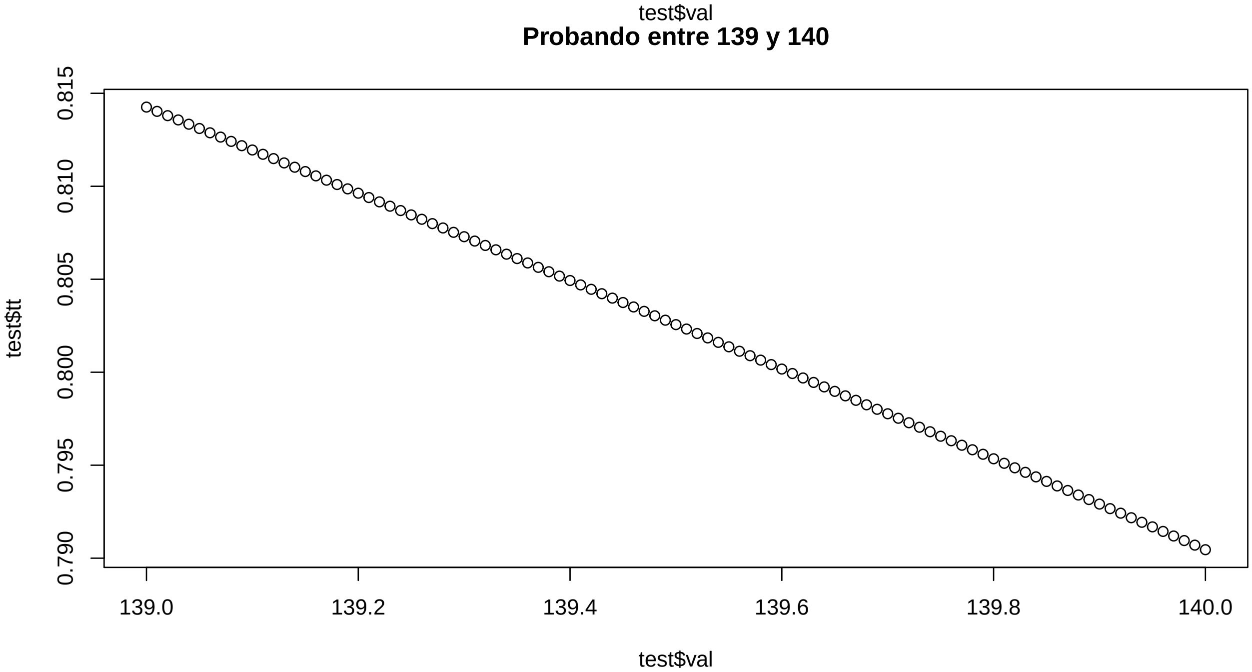
$$E[N(t)] = \begin{cases} \int_0^t 100\,dt = 100t & 0 \le t \le \frac{1}{2} \\ \int_{\frac{1}{2}}^t 600t - 200\,dt + 50 = 300(t^2 - \frac{1}{4}) - 200(t - \frac{1}{2}) + 50 & \frac{1}{2} < t \le \frac{3}{4} \\ \int_{\frac{3}{4}}^t 400t - 50\,dt + 93.75 = 25(8t^2 - 2t - 3) + 93.75 & \frac{3}{4} < t \le 1 \\ \int_1^t -500t + 850\,dt + 168.75 = -50(5t^2 - 17t + 12) + 168.75 & 1 < t \le 1.5 \end{cases}$$

Given that there is a limit of 150 vehicles:

$$E[N(t)] = \begin{cases} 100t & 0 \le t \le \frac{1}{2} \\ 300(t^2 - \frac{1}{4}) - 200(t - \frac{1}{2}) + 50 & \frac{1}{2} < t \le \frac{3}{4} \\ 25(8t^2 - 2t - 3) + 93.75 & \frac{3}{4} < t < 0.94468 \\ 150 & t \ge 0.94468 \end{cases}$$

(b)





Luego de hacer las pruebas para $\lambda(t)$ obtenemos lo siguiente:

```
lambda = 139.6
t = 0.91232
# 8:44 AM
```

Por lo que $t = 0.91232$ horas.

(c)

The following function simulates a non-homogenous poisson process from a homogenous poisson process:

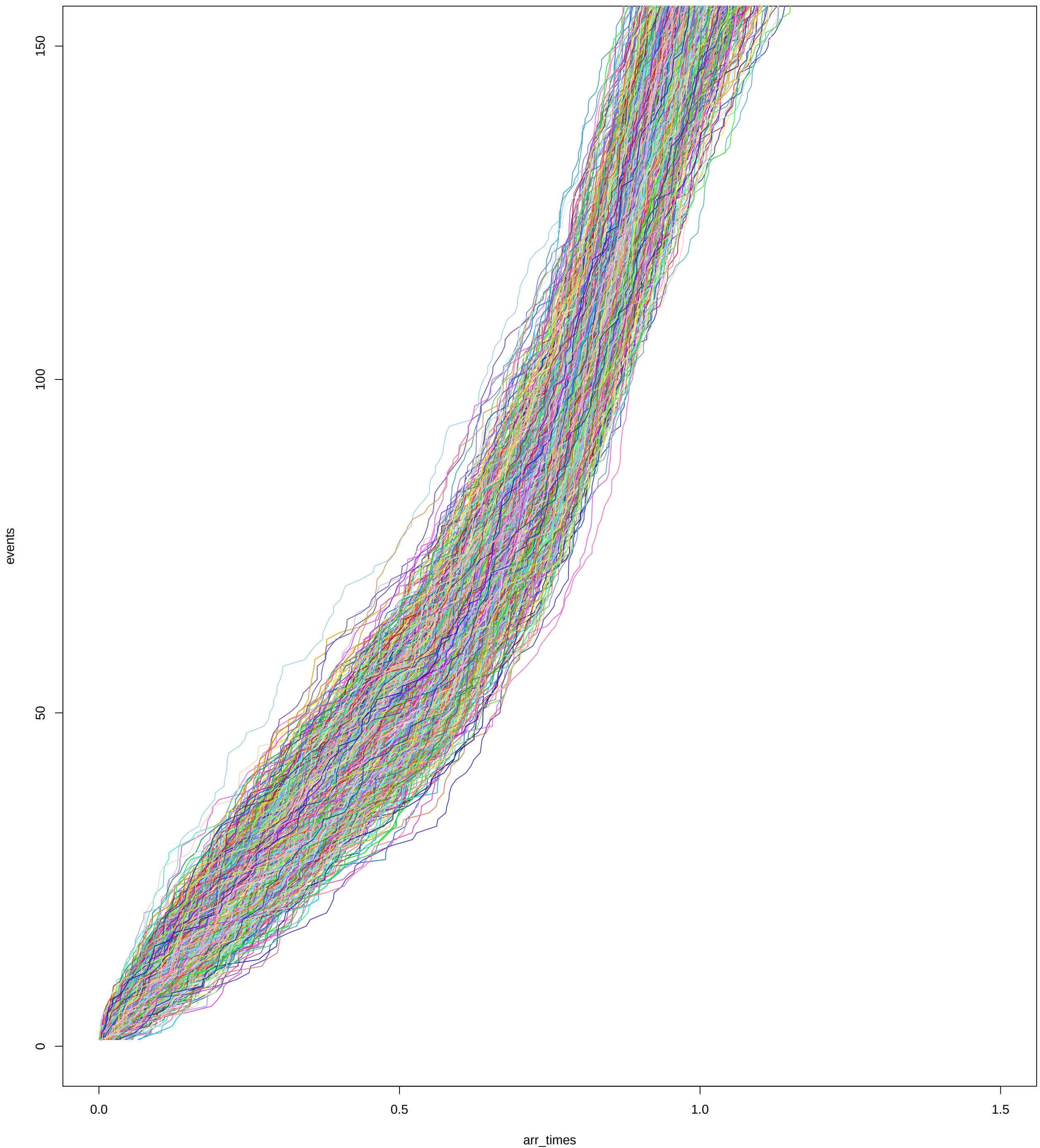
```r
non_hom_poisson <- function(fun,l,a,b,start=0) {
    # This function generates a non-homogenous poisson
    # process from a homogenous poisson process
    # PARAMS:
    # fun:   if the non-homogenous poisson process has
    #        multiple functions per time subinterval
    #        this parameters represents such function
    # l:     lambda for the homogenous poisson process
    # a:     lower bound for the time subinterval
    # b:     upper bound for the time subinterval
    # start: this parameter is used to keep track of
    #        the process count.

    # We generate the homogenous poisson process
    # arrival times
    val <- rpois(1,l*(b-a))
    intervals <- (b-a) * sort(runif(val)) + a

    # Non-homogenous poisson process
    evs <- length(intervals) # lenght of arrival times
    nh_val <- 0 + start # start of the event count
    nh_intervals <- c() # arrival times for the NHPP
    for (i in 1:evs) {
        if (runif(1) < fun(intervals[i])/l) {
            # only including intervals from the HPP which
            # match with fun(intervals[i])/l probability
            nh_intervals <- c(nh_intervals, intervals[i])
            nh_val <- nh_val+1 # adding one to the event count
        }
    }
    nh_events <- seq(1+start,nh_val,1) # events since the previous group
    return(list(arrival_times=nh_intervals, events=nh_events))
}
```

```r
simulation <- function(iters, functions, lambdas, ints) {
    # This function simulates from the NHPP
    # iters:     number of iterations to plot and add to the list of
    #            dataframes
    # functions: list of functions corresponding to the lambda function
    # lambdas:   list of lambdas for each subinterval
    # ints:      lists of vectors of 2 elements each containing the intervals
    #            that correspond to each element of lambdas and functions lists
    p <- list()
    for (i in 1:iters) {
        maximum <- 0 # start for the next NHPP simulation to continue count
        arr_times <- c() # arrival times
        events <- c() # event counts
        for (k in 1:4) {
            int <- non_hom_poisson(lambda_funs[[k]],lambdas[[k]],
                                   ints[[k]][1],ints[[k]][2],
                                   start=maximum)
            maximum <- max(int$events) # remembering last event count
            arr_times <- c(arr_times, int$arrival_times)
            events <- c(events, int$events)
        }
        p[[i]] <- data.frame(arrival_times=arr_times, events=events)
        # plots
        if (i == 1) {plot(arr_times, events, cex=0.5, pch='.',
                          col=randomColor(), xlim=c(0,1.5),
                          ylim=c(0,150))}
        lines(arr_times, events, col=randomColor())
    }
    return(p)
}
```

```
data <- simulation(1000, lambda_funs, lambdas, ints)
```




```
ratio <- 0
for (i in 1:length(data)) {
    df <- data.frame(data[[i]])
    cnt <- df %>% filter(arrival_times < 0.91232 & events >= 150) %>% dplyr::count()
    if (cnt[1] >= 1) {
        ratio <- ratio + 1
    }
}
```

```r
ratio/1000
#> [1] 0.097
```

Problem 3

(a)

Our infinitesimal generator is the following:

$$Q = \begin{pmatrix} -\lambda & \lambda & 0 & 0 & \dots \\ \mu & -(\lambda+\mu) & \lambda & 0 & \dots \\ 0 & 2\mu & -(2\mu+\lambda) & \lambda & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

(b)

We solve the following system:

$$\begin{cases} \sum_{i=1}^{\infty} \pi_i = 1 \\ \lambda\pi_0 = \mu\pi_1 \\ \lambda\pi_1 = 2\mu\pi_2 \\ \vdots \\ \lambda\pi_{n-1} = 2\mu\pi_n \\ \vdots \end{cases}$$

First we have:

$\pi_1 = \frac{\lambda\pi_0}{\mu}$
$\pi_2 = \frac{\lambda^2\pi_0}{2\mu^2}$
$\pi_3 = \frac{\lambda^3\pi_0}{2^2\mu^3}$
$\vdots$
$\pi_n = \frac{\lambda^n\pi_0}{2^{n-1}\mu^n}$
$\vdots$

Then:

$\sum_{i=0}^{\infty} \pi_i = \pi_0 + \frac{\lambda\pi_0}{\mu} + \frac{\lambda^2\pi_0}{2\mu^2} + \dots + \frac{\lambda^n\pi_0}{2^{n-1}\mu^n} + \dots = 1$

And so factoring π_0 we get:

$\pi_0(1 + \frac{\lambda}{\mu} + \frac{\lambda^2}{2\mu^2} + \dots) = \pi_0(1 + \sum_{i=1}^{\infty} \frac{1}{2^{i-1}}(\frac{\lambda}{\mu})^i)$

Then multiplying $\frac{2}{2}$ to the summation:

$\pi_0(1 + \frac{2}{2}\sum_{i=1}^{\infty} \frac{1}{2^{i-1}}(\frac{\lambda}{\mu})^i)$

$= \pi_0(2\sum_{i=0}^{\infty}(\frac{\lambda}{2\mu})^i - 1)$

$\pi_0(2(\frac{1}{1-\frac{\lambda}{2\mu}}) - 1) = 1$

$\pi_0 = \frac{1}{2(\frac{1}{1-\frac{\lambda}{2\mu}})-1}$

$\vdots$

$\pi_n = \frac{\lambda^n}{2^{n-1}\mu_n \pi_0} \frac{1}{2(\frac{1}{1-\frac{\lambda}{2\mu}})-1}$

finally:

$\pi_n = \frac{1}{2^{n-1}}(\frac{\lambda}{\mu})^n \pi_0$