Stochastics – Homework 5

Moira Lampe

NOTE: I was not able to finish in time. Please send me an email to moira.lampe@outlook.com, so I can send you the completed version. Thank you!

Point Processes

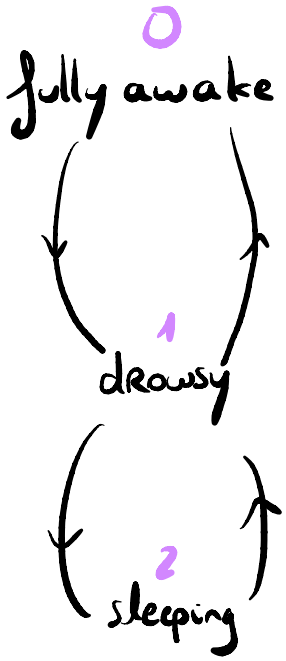
① Maximum likelihood for Markov Chain

Let's imagine a cognitive context where we are interested in sleeping patterns. Every 5 minutes, a check is performed on the participants & we note their state.

We want to model this by a (discrete) Markov chain with 3 states: "sleeping", "(fully) awake", "drowsy".

a) Plot the diagrams & parametrize the two following (informal) models with the least possible amount of parameters

- First case

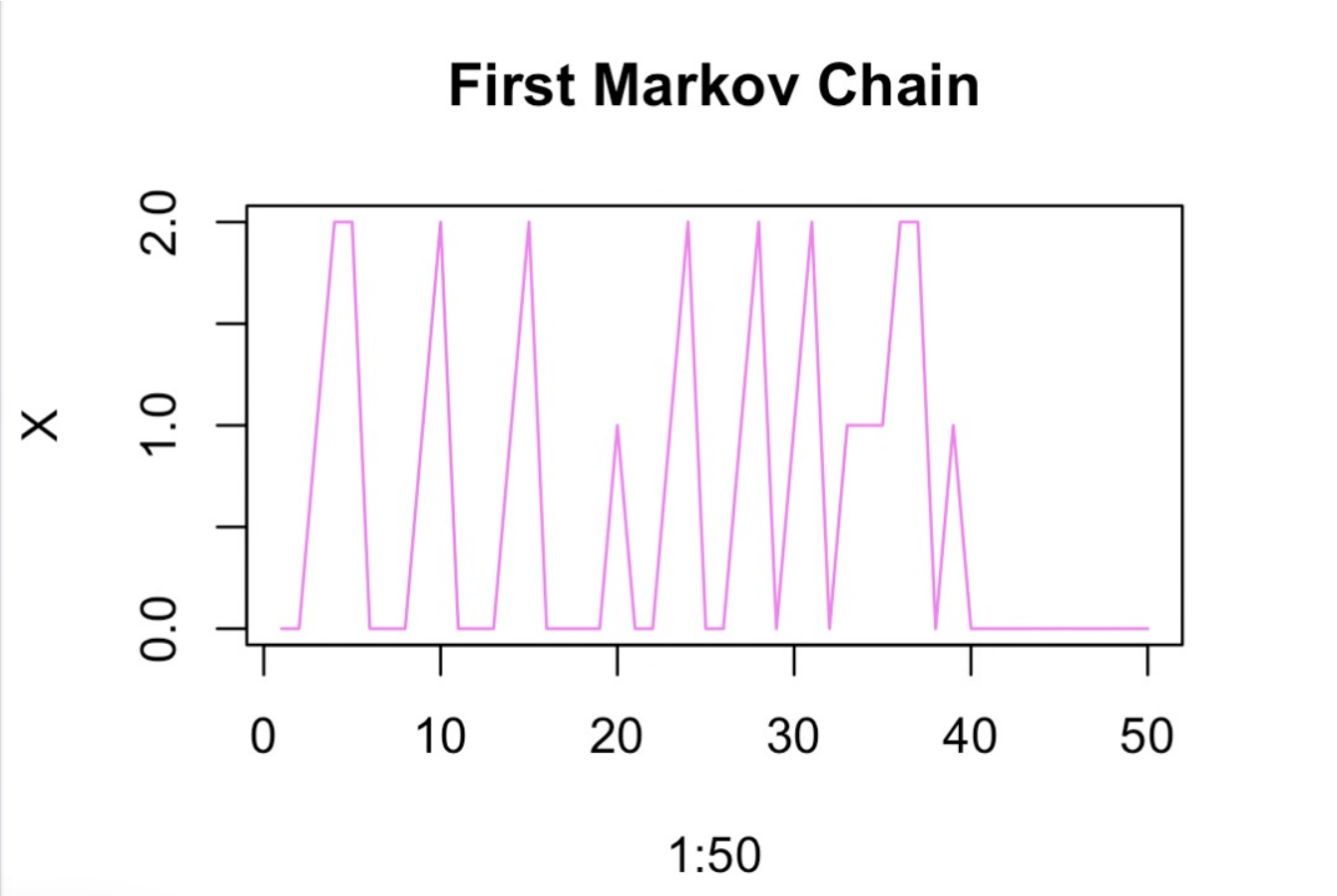


1. If $X=0$, we have three possibilities: stay at 0, go to 1
2. If $X=1$, we have three possibilities: stay at 1, go to 0, go to 2
3. If $X=2$, we have three possibilities: stay at 2, go to 1

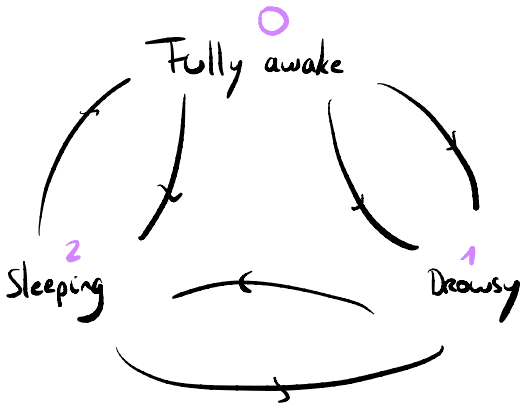
```r
#First case

Y = rbinom(50,1, 0.3)
print(Y)
Z = rbinom(50,1, 0.7)
print(Z)
W = rbinom(50,1, 0.4)
print(W)
U = rbinom(50,1, 0.5)
print(U)
X <- c()
for (i in 2:50){
  X[1] = 0 #initializing state
  if (X[i-1] == 0){
    X[i] <- X[i-1]*(1 - Z[i])+(1 - X[i-1])*Y[i]
  }
  else if (X[i-1] == 1){ #we are in state drowsy
    #if we are in state drowsy, we can go back to sleep or awake, so we can go to 0 or to 1, and we can stay in 1

    X[i] <-X[i-1]*(1 + W[i]) - X[i - 1]*Y[i]
  }
  else if (X[i-1] == 2){
    X[i] <- X[i-1]*(1 - U[i]) #if U is 0, we stay in 2, if U is 1, we go back to 1
  }
}
plot(1:50, X, type = 'l', main = 'First Markov Chain',col = 'violet')
```



· Second case

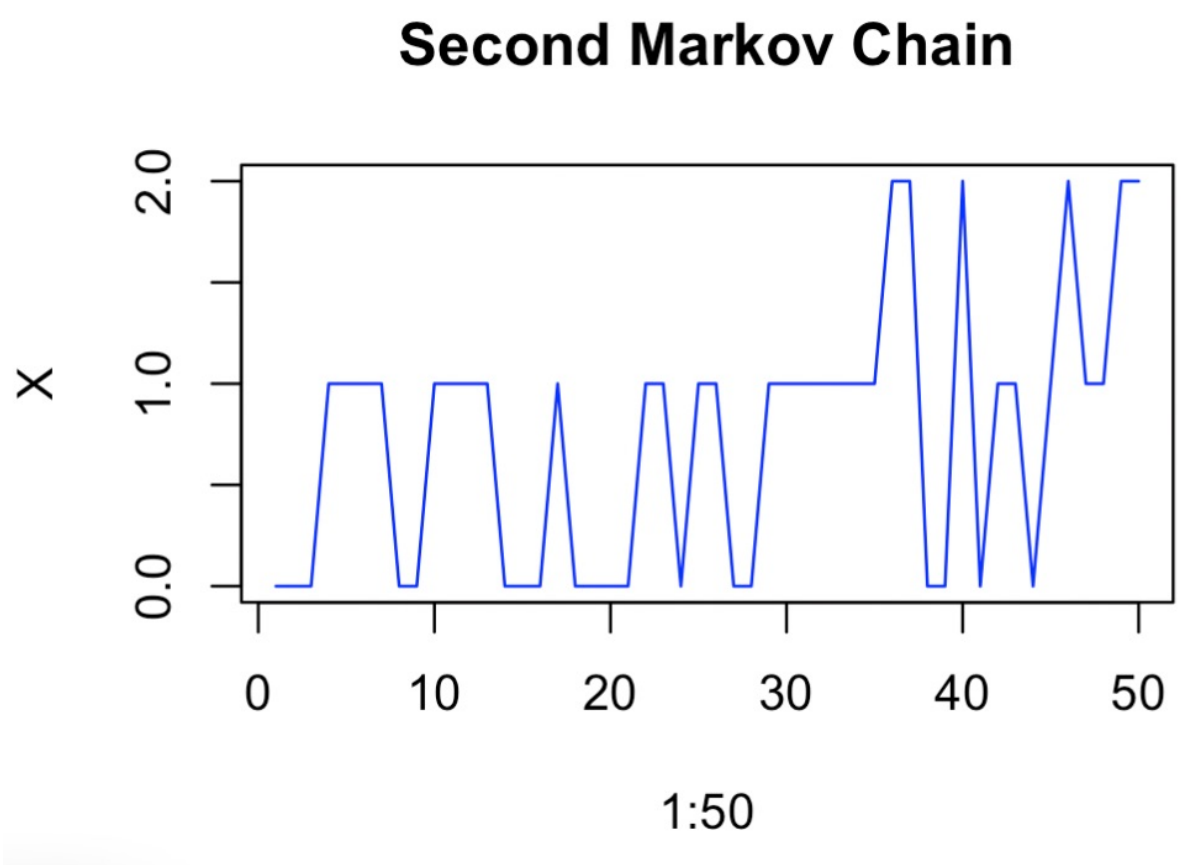


1. If X=0, we have three possibilities : stay at 0
   go to 1
   go to 2
2. If X=1, we have three possibilities : stay at 1
   go to 0
   go to 2
3. If X=2, we have three possibilities : stay at 2
   go to 0
   go to 1

```
#Second case

Y = rbinom(50,1, 0.3)
print(Y)
Z = rbinom(50,1, 0.6)
print(Z)
W = rbinom(50,1, 0.2)
print(W)
U = rbinom(50,1, 0.8)
print(U)
P = rbinom(50,1, 0.2)
print(P)
G = rbinom(50,1, 0.5)
print(G)
X <- c()
for (i in 2:50){
  X[1] = 0 #initializing state
  if (X[i-1] == 0){
    X[i] <- Y[i] + P[i]
  }
  else if (X[i-1] == 1){ #we are in state drowsy
    #if we are in state drowsy, we can go back to sleep or awake, so we can go to 0 or to 1, and we can stay in 1
    X[i] <- W[i] + Z[i]
  }
  else if (X[i-1] == 2){
    X[i] <- U[i] + G[i] #if U is 0, we stay in 2, if U is 1, we go back to 1
  }
}

plot(1:50, X, type = 'l', main = 'Second Markov Chain',col = 'blue')
plot(1:50, X, main = 'First Markov Chain',col = 'red')
```



b) Make a function in R which simulates the first model if you give in the entries the state x_0, the parameters of the model (only the ones parameterizing the transitions) and the length of the chain to simulate n.

```r
#First case

MarkovChain_1 = function(p_01, p_10, p_12, p_21, n, X_0){
  Y = rbinom(n,1, p_01)
  print(Y)
  Z = rbinom(n,1, p_10)
  print(Z)
  W = rbinom(n,1, p_12)
  print(W)
  U = rbinom(n,1, p_21)
  print(U)
  X <- c()
  for (i in 2:n){
  X[1] = X_0 #initializing state
      if (X[i-1] == 0){
        X[i] <- Y[i]
      }
      else if (X[i-1] == 1){ #we are in state drowsy
        #if we are in state drowsy, we can go back to sleep or awake, so we can go to 0 or to 1, and we can stay in 1
        X[i] <- Z[i] + W[i]
      }
      else if (X[i-1] == 2){
        X[i] <- U[i] + 1 #if U is 0, we stay in 2, if U is 1, we go back to 1
      }
    }
  return(X)
  }

MarkovChain_1(0.5, 0.5, 0.5, 0.5, 10, 0)
print(X)
```

```r
#Second Case

MarkovChain_2 = function(p_01, p_10, p_12, p_21, p_02, p_20, n, X_0){
  Y = rbinom(n,1, p_01)
  print(Y)
  Z = rbinom(n,1, p_10)
  print(Z)
  W = rbinom(n,1, p_12)
  print(W)
  U = rbinom(n,1, p_21)
  print(U)
  P = rbinom(n,1, p_02)
  print(P)
  G = rbinom(n,1, p_20)
  print(G)
  X <- c()
  for (i in 2:n){
    X[1] = X_0 #initializing state
    if (X[i-1] == 0){
      X[i] <- Y[i] + P[i]
    }
    else if (X[i-1] == 1){ #we are in state drowsy
      #if we are in state drowsy, we can go back to sleep or awake, so we can go to 0 or to 1, and we can stay in 1
      X[i] <- W[i] + Z[i]
    }
    else if (X[i-1] == 2){
      X[i] <- U[i] + G[i] #if U is 0, we stay in 2, if U is 1, we go back to 1
    }
  }
  return(X)
}

MarkovChain_2(0.2, 0.9, 0.2, 0.1, 0.1, 0.8, 30, 0)
```

c) Write the (log) likelihood of this problem and give the maximum likelihood estimator of $p_{0\to s}$, the probability to fall asleep when you are drowsy. Would it be different if we were in the second model?

For a given model, $X \sim \mathbb{P}_\theta$, $g_\theta(x) = \mathbb{P}_\theta(X=x)$, the likelihood will be computed as follows:

$$\theta \longmapsto \mathcal{L}_\theta(x) = \mathbb{P}_\theta(X=x)_{|x=X}$$

The MLE is $\hat{\theta} = \underset{\theta\in\Theta}{\operatorname{argmax}}\ \mathcal{L}_\theta(x) = \underset{\theta\in\Theta}{\operatorname{argmax}}\ \ell_\theta(x)$ (ℓ = log likelihood)

$\Rightarrow$ We need θ to be the minimal possible set of parameters

↳ as here we are only interested in $p_{0\to s}$, so going from state 1 to 0: $p_{1\to 0}$, we can say that we have full knowledge of the setup if we know $\pi = \mathbb{P}(X_0=0)$ and $p_{0\to 1}$ & $p_{1\to 0}$

Our parameter space here is: $\Theta = (\pi, p_{0\to 1}, p_{1\to 0})$

For a given realisation $x_0, \dots, x_n$:

$$\begin{aligned}
\mathbb{P}_\theta((X_0,\dots,X_n)) = ((x_0,\dots,x_n)) &= \mathbb{P}(X_0=x_0,\dots,X_n=x_n) \\
&= \mathbb{P}_\theta(X_0=x_0,\dots,X_{n-1}=x_{n-1}) \cdot \mathbb{P}_\theta(X_n=x_n \mid \text{all the points from 0 to } n-1) \\
&= \mathbb{P}_\theta(X_0=x_0) * \dots * \mathbb{P}_\theta(X_n=x_n \mid X_{n-1}=x_{n-1}) \\
&= \pi^{x_0}(1-\pi)^{1-x_0} \cdot p_{0\to 1}^{n_{0\to 1}} \cdot p_{1\to 0}^{n_{1\to 0}} \cdot p_{0\to 0}^{n_{0\to 0}} \cdot p_{1\to 1}^{n_{1\to 1}}
\end{aligned}$$

With $n_{i\to j}$ the number of transition from i to j seen in the sequence $x_0, \dots, x_n$.

Likelihoods:

So the likelihood sequence $X_0, \dots, X_n$ with counts $N_{i \to j}$ is the number of times a transition $i \to j$ occurs:

$$L_\theta(X) = p^{X_0}(1-p)^{1-X_0} \cdot p_{0\to 1}^{N_{0\to 1}} \cdot (1-p_{0\to 1})^{N_{1\to 0}} \cdot p_{1\to 0}^{N_{0\to 0}} (1-p_{1\to 0})^{N_{1\to 1}}$$

So the log likelihood is:

$$\begin{aligned} l_\theta(x) = {} & X_0 \log(p) + (1-X_0)\log(1-p) + N_{1\to 0}\log(p_{1\to 0}) \\ & + N_{1\to 1}\log(1-p_{1\to 0}) + N_{0\to 0}\log(1-p_{0\to 1}) \\ & + N_{0\to 1}\log(p_{0\to 1}) \end{aligned}$$

To find the MLE of the transitions:

$$\frac{d\, l_\theta(x)}{d\, p_{1\to 0}} = \frac{N_{1\to 0}}{p_{1\to 0}} - \frac{N_{1\to 1}}{1-p_{1\to 0}}$$

The derivative vanishes when $\frac{N_{1\to 0}}{p_{1\to 0}} = \frac{N_{1\to 1}}{1-p_{1\to 0}}$

that is $N_{1\to 0} = (N_{1\to 1} + N_{1\to 0})\, p_{1\to 0}$ that is $p_{1\to 0} = \frac{N_{1\to 0}}{N_{1\to 1} + N_{1\to 0}}$

Hence: $\hat{p}_{1\to 0} = \frac{N_{1\to 0}}{N_{1\to 1} + N_{1\to 0}}$ } The number of times "1 to 0" occurred divided by the number of times 1 was seen.

$$\frac{N_{i\to j}}{\#\{X_k = i\}}$$

$\Rightarrow$ This estimator is consistent if the chain is recurrent. Recurrence means that the chain will visit the transition an infinite number of times when $n \to +\infty$

It would be different if one of the models were not recurrent.

State i is recurrent if "starting from i, and from wherever you can go, there is a way of returning to i."

Transient: at least one state where you can go & then there will be no path back to it.

→ as there are no transient states in both models, it will not be different.

d) Show on simulation that this estimator converges towards the true probability when n tends to infinity.

Let's say $p\_10 = 0.5$

By simulating with large number n, we can see that $\hat{p}_{-10} \to p_{-10}$

```
amount_drowsy = function(X, n){
  ones <- c()
  for (k in 1:n){
    if (X[k] == 1){
      ones = c(ones, +1)
      print(ones)
    }
  }
return(n_ones = length(ones))
  }


drowsy_to_sleep = function(X, n){
  one_to_zero <- c()
  for (j in 1:n){
    if (X[j-1] == 0 && X[j] == 1){
      one_to_zero = c(one_to_zero, +1)
      print(one_to_zero)
    }
  }
  return(n_10 = length(one_to_zero))
}

p_10_hat = function(n_10, n_ones){
  return(n_10/n_ones)
}
```

c) How would you decide if we are in the first or second mode when you have the observations?

→ See figures

② Maximum likelihood for Poisson Processes

We are observing an inhomogeneous Poisson Process with intensity $\lambda_0(\cdot)$ on $[0, T_{max}]$.

We want to estimate $\lambda_0(\cdot)$ by a piecewise constant function on a regular partition which means that the candidate intensities $\lambda(\cdot)$ are, for a certain integer d giving the number of bins, of the form:

$$\lambda(t) = \sum_{i=1}^{d} a_i^d \mathbb{1}_{b_{i-1}^d < t \leq b_i^d} \quad \text{, where } b_i^d = i\frac{T_{max}}{d}$$

a) When $d=1$, write the likelihood of the process & provide & MLE of a_1^1. Implement it & show on simulation that when T_{max} grows, the estimation is converging when the process is homogeneous.

Likelihood of N is:

$$\mathcal{L}_\lambda(N) = \prod_{T \in N} \lambda(t) \cdot e^{-\int_0^{T_{max}} \lambda(t)dt}$$

The log likelihood of N is:

$$l_\lambda(N) = \sum_{T \in N} \log(\lambda(t)) - \int_0^{T_{max}} \lambda(t)dt$$

$$= \int_0^{T_{max}} \log(\lambda(t)) dN_t - \int_0^{T_{max}} \lambda(t)dt$$

$$l_{[a,b]}(N) = \log(n \cdot a) N_{[0, \frac{T_{max}}{2}]} + \log(n \cdot b) N_{]\frac{T_{max}}{2}, T_{max}]} - na\frac{T_{max}}{2} - nb\frac{T_{max}}{2}$$

$\frac{d}{da}$

$$\hat{a} = \frac{N_{[0, \frac{T_{max}}{2}]}}{n \frac{T_{max}}{2}}$$

$$\hat{b} = \frac{N_{]\frac{T_{max}}{2}, T_{max}]}}{n \frac{T_{max}}{2}}$$

Because $d=1$, we have that $\lambda(t) = \sum_{i=1}^{d} a_i^d \mathbb{1}_{b_{j-1}^d < t < b_i^d}$

$$= \sum_{i=1}^{\hat{1}} a_i^{\hat{1}} \mathbb{1}_{b_{j-1}^{\hat{1}} < t < b_i^{\hat{1}}}$$

$$= a_1^{\hat{1}} \mathbb{1}_{0 \leq t < T_{max}}$$

If we fill in from above $= \int_0^{T_{max}} \log(\lambda(t)) dN_t - \int_0^{T_{max}} \lambda(t)\, dt$

log likelihood of a point process

↳ We have that

$$\ell = \log a_1^{\hat{1}} N_{[0,T_{max}]} - a_1^{\hat{1}} T_{max}$$

Doing our derivation : $\frac{d\ell}{da_1^{\hat{1}}} = \frac{1}{a_1^{\hat{1}}} N_{[0,T_{max}]} - T_{max}$

Hence, we find that our MLE is:

$$\hat{a}_1^{\hat{1}} = \frac{N_{[0,T_{max}]}}{T_{max}}$$

	$\frac{N_{[0,T_{max}]}}{T_{max}}$		
$\frac{d\ell}{da_1^{\hat{1}}}$	+	0	−
ℓ	↗		↘

I was not able to prove it...

```
#function for homogeneous poisson process

Homo_Poisson = function(Tmax, lambda)
{
  t = 0 #t starts out with zero
  N = numeric()
  res = c()
  while(t < Tmax){ #we want the process to go on until the cumulative sum of t reaches T, which is 1 here
    u = runif(1) #as long as the above is true, we create u, which is a uniform random variable
    t = t - log(u)/lambda #then our new t will be this #it is the same thing as doing it separately and then taking the cumulative sum of it.
    N = c(N, t)
  }
  res = c(N,"The a_hat is:",sum(N/Tmax), "the sum of N is:", sum(N))
  return(res)
}

Homo_Poisson(10, lambda = 3)

#this is not working, it is definitely not converging when Tmax grows...
```

```
#Let's try a different way of simulating a poisson process:
#Let's try with exponentially distributed interarrival times.
#We simulate the arrival times until the maximum time horizon is achieved.

My_Poisson = function(Tmax, rate){
  Y = c()
  arrivals <- rexp(50, rate)
  while (arrivals[length(arrivals)] < Tmax){
  Y = c(Y, arrivals)
  }
  return(Y)
}

My_Poisson(1, 0.4)
print(Y)

#I thought I could simulate it differently but somehow this one also doesn't work
```

b) For a more general d write the corresponding likelihood & give the MLE of the different a_i^d.

$$\lambda(t) = \sum_{i=1}^{d} a_i^d \mathbb{1}_{b_{i-1}^d < t \leq b_i^d} \quad \text{, where } b_i^d = i\frac{T_{max}}{d}$$

As seen above, we know that the formula for the log likelihood is the following:

$$L_\lambda(N) = \sum_{T \in N} \log(\lambda(t)) - \int_0^{T_{max}} \lambda(t)\,dt$$

$$= \int_0^{T_{max}} \log(\lambda(t))\,dN_t - \int_0^{T_{max}} \lambda(t)\,dt$$

↳ we know that this is equal to a_i^d as long as t is in between b_{i-1}^d & b_i^d

$\Rightarrow$ We can hence rewrite the formula like this:

$$\sum_{i=1}^{d} \left((\log a_i^d) N_{[b_{i-1}^d, b_i^d)} - a_i^d \left(b_i^d - b_{i-1}^d \right) \right)$$

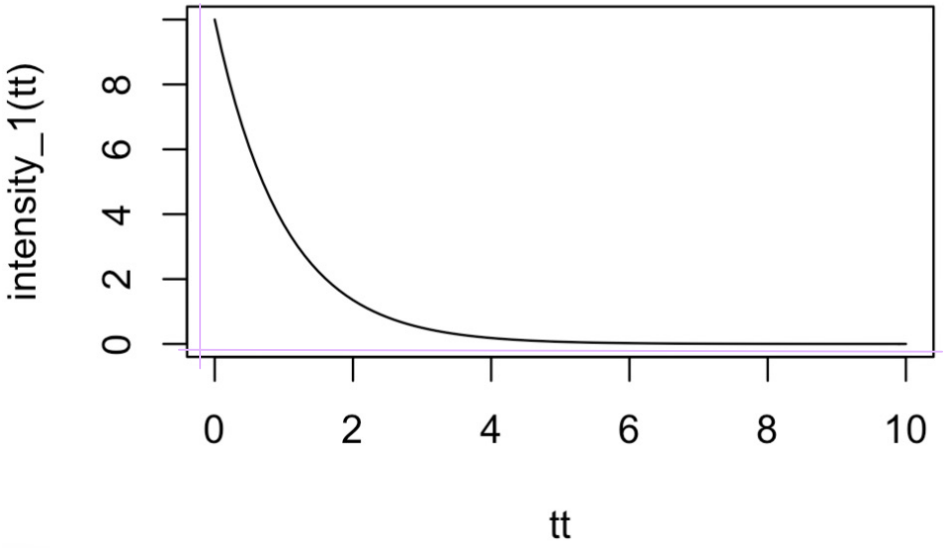
↳ from the ex. we know that $b_i^d = i\frac{T_{max}}{d}$, so we know that 'this' is $\frac{T_{max}}{d}$

$\Rightarrow$ We do derivation again:

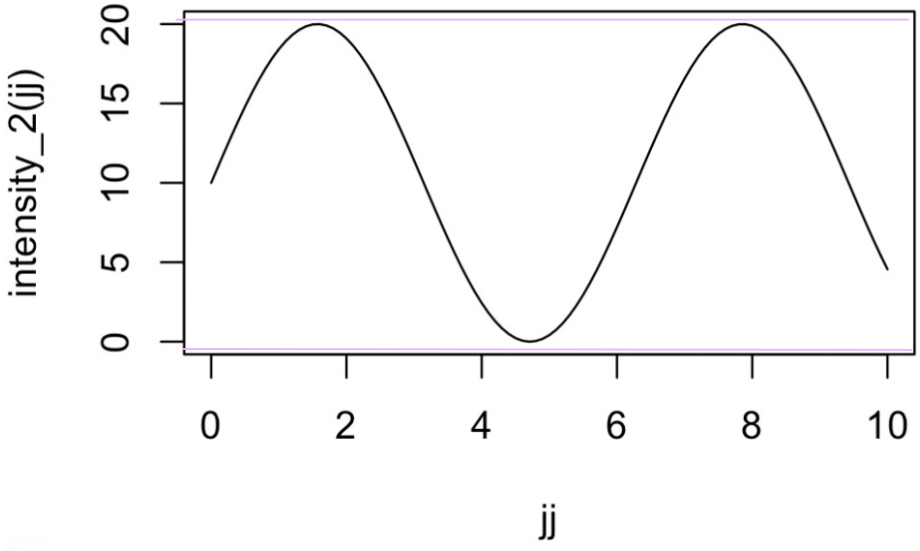
$$\frac{dl}{da_i^d} \quad \& \text{ find that } \hat{a}_i^d = \frac{N_{[b_{i-1}^d, b_i^d]}}{T_{max}/d}$$

c) Simulate by thinning a Poisson Process of intensity ne^{-x} and $n(1+\sin(x))$ on $(0, T_{max})$ for various n.

Superpose different MLE of the intensities for various choices of d & n.

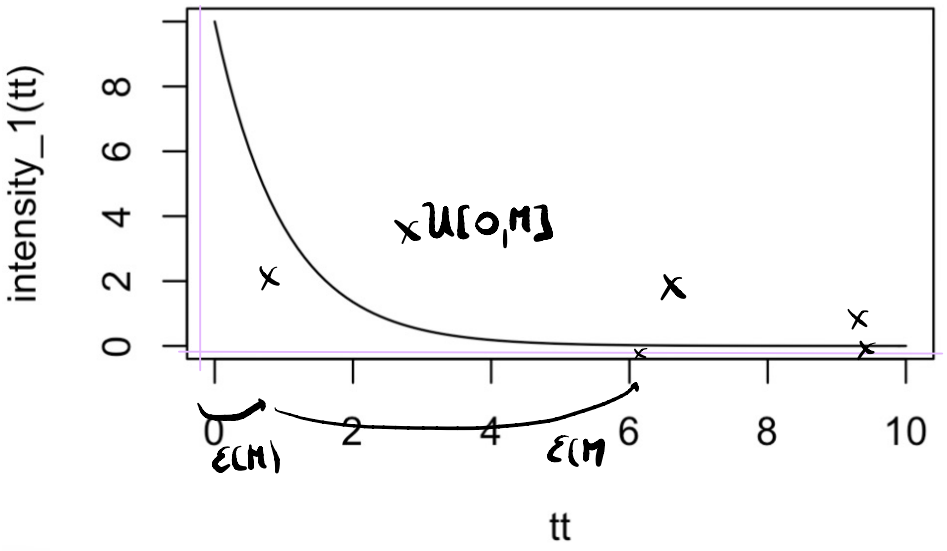


=> We can see that the first intensity function is bounded by 0.

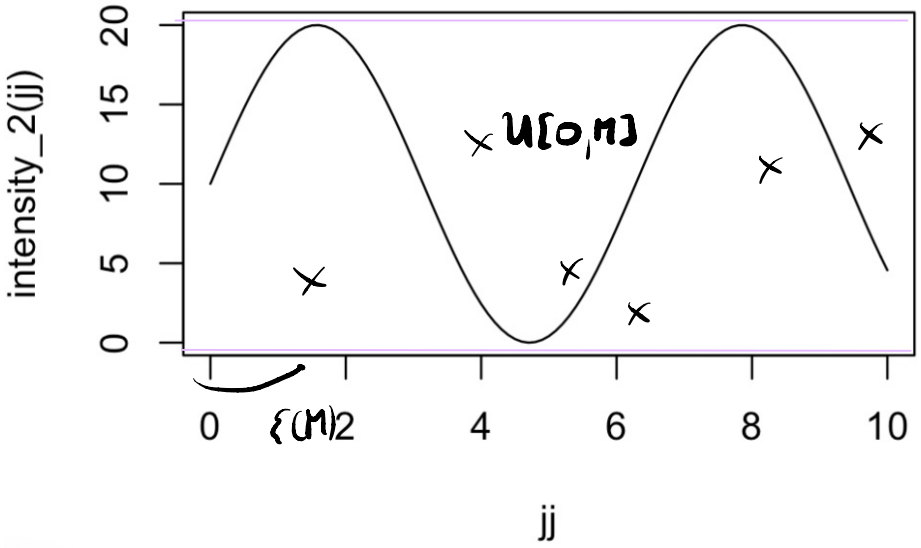


=> We can see that the second intensity function is bounded by 20.

Hence, the thinning Process :



where M=1



where M=1

```r
#first intensity function

intensity_1 = function(x, n){
  return(n*exp(-x))
}

#second intensity function

intensity_2 = function(x, n){
  return(n*(1 + sin(x)))
}

intesity_1_plus = 0

intensity_2_plus = 20

Non_Homo_Poisson = function()
{
  t = 0 #t starts out with zero
  T = 1 #t ends with 1
  X = numeric()
  while(t < T){ #we want the process to go on until the cumulative sum of t reaches T, which is 1 here
    u = runif(1) #as long as the above is true, we create u, which is a uniform random variable
    t = t - log(u)/intensity_1_plus #then our new t will be this #it is the same thing as doing it separately and then taking the cumulative sum of it.
    d = runif(1)
    if (d < h(t)/intensity_1_plus){ # compare our uniform variable to the probability
      X = c(X, t) #if the above is the case, we add this to our X
    }
  }
  return(X)
}
```

d) Apply AIC criterion to select for a given observed Poisson Process, a good d.

MISSING

③ How do we check that our simulation is correct?

MISSING