Point Processes

Question 1 - A bit of statistics with Poisson processes

We are observing two independent spike trains on the same interval $[0, T]$ and we want to test if their firing rate is the same. We will assume here that we have two independent Poisson processes N^a and N^b with rate $\lambda_a(.)$ and $\lambda_b(.)$, not necessarily constant.

A – What is the distribution of N_T^a and N_T^b, respectively the number of points of N^a and N^b in $[0, T]$? Are these variables independent?

B – Compute $\mathbb{P}(N_T^a = k \cup N_T^b = l)$ for all k and l.

C – Let $N_T = N_T^a + N_T^b$. Compute $\mathbb{P}(N_t = n)$ for all n. Can you give the name and the parameters of this distribution?

D – Compute $\mathbb{P}(N_t^a = k | N_t = n)$. *You can first see that k has to be less than n and that in this case $N_T^b = n - k$.* Can you give the name and the parameters of this conditional distribution?

E – Under the null hypothesis that $\lambda_a(.) = \lambda_b(.)$, realize that the previous distribution is known and derive from it a test of $H_0 : \lambda_a(.) = \lambda_b(.)$ versus $H_1 : \lambda_a(.) \geq \lambda_b(.)$

F – Apply it on the data of the STAR package for two different neurons. NB: *If $N^1, \ldots, N^n$ are IID Poisson processes with intensity $\lambda(.)$ then $N = N^1 \cup \ldots \cup N^n$ is a Poisson process with rate $n\lambda(.)$. You can use that to glue all the trials together.*

step A

Given that the two independent Poisson processes N^a and N^b don't have necessarily constant rates $\lambda_a(.)$ and $\lambda_b(.)$, we can consider them inhomogeneous Poisson process over an interval $[0, T]$. As such, given the random variables N_T^a and N_T^b the number of points of N^a and N^b that happened in the interval $[0, T]$, we have:

$$\begin{aligned}
&N_T^a \sim \mathbb{P}\Big(\int_{[0,T]} \lambda_a(x)dx\Big) \\
&N_T^b \sim \mathbb{P}\Big(\int_{[0,T]} \lambda_b(x)dx\Big) \\
&\text{Where,} \\
&\quad \lambda_a, \text{ the intensity of the process } a \\
&\quad \lambda_b, \text{ the intensity of the process } b
\end{aligned}$$

Based on wikipedia (https://en.wikipedia.org/wiki/Poisson_point_process)'s formulas, we can state the following:

$$\begin{aligned}
\Lambda_a(0, T) &= \int_{[0,T]} \lambda_a(x)dx \\
\Lambda_b(0, T) &= \int_{[0,T]} \lambda_b(x)dx \\
P(N_T^a = k) &= \frac{\big[\Lambda_a(0, T)\big]^k}{k!} . e^{-\Lambda_a(0,T)} \\
P(N_T^a = l) &= \frac{\big[\Lambda_b(0, T)\big]^l}{l!} . e^{-\Lambda_b(0,T)}
\end{aligned}$$

Given that the two Poisson processes N^a and N^b are independent, we can consider the random variables N_T^a and N_T^b to be independent as well.

step B

Based on our independence hypothesis stated in step A, we can say that, for two random variable X and Y that $P(X \cup Y) = P(X).P(Y)$. As such:

$$
\begin{aligned}
\mathbb{P}(N_T^a = k \cup N_T^b = l) &= \mathbb{P}(N_T^a = k) * \mathbb{P}(N_T^b = l) \\
&= \frac{\left[\Lambda_a(0,T)\right]^k}{k!}.\, e^{-\Lambda_a(0,T)} * \frac{\left[\Lambda_b(0,T)\right]^l}{l!}.\, e^{-\Lambda_b(0,T)} \\
&= \frac{\left[\Lambda_a(0,T)\right]^k \left[\Lambda_b(0,T)\right]^l}{k!l!}.\, e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))} \\
&= \frac{\left[\Lambda_a(0,T)\right]^k \left[\Lambda_b(0,T)\right]^l}{k!l!}.\, e^{-(\int_{[0,T]} \lambda_a(x)+\lambda_b(x)dx)}
\end{aligned}
$$

step C

We previously found:

$$
\mathbb{P}(N_T^a = k \cup N_T^b = l) = \frac{\left[\Lambda_a(0,T)\right]^k \left[\Lambda_b(0,T)\right]^l}{k!l!}.\, e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))}
$$

As such, we can state:

$$
\begin{aligned}
\mathbb{P}(N_T^a + N_T^b = n) &= \mathbb{P}(N_T^a = k, N_T^b = n-k) \quad \forall n, k \in \mathbb{N} \text{ s.t. } n-k >= 0 \\
&= \binom{n}{k} \mathbb{P}(N_T^a = k \cup N_T^b = n-k) \\
&= \binom{n}{k} \frac{\left[\Lambda_a(0,T)\right]^k \left[\Lambda_b(0,T)\right]^{n-k}}{k!(n-k)!}.\, e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))} \\
&= e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))}.\, \binom{n}{k} \frac{\left[\Lambda_a(0,T)\right]^k \left[\Lambda_b(0,T)\right]^{n-k}}{k!(n-k)!}
\end{aligned}
$$

Relying on the binomial formula, we can then state:

$$
\mathbb{P}(N_T^a + N_T^b = n) = e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))}.\, \frac{\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n}{n!}
$$

As such, we find:

The sum of the random variables N_T^a and N_T^b also follows a Poisson distribution with parameter/rate $\Lambda_a(0,T) + \Lambda_b(0,T)$.

$$
N_T^a + N_T^b \sim \mathbb{P}\big(\int_{[0,T]} \lambda_a(x) + \lambda_b(x)dx\big)
$$

This can be verified with the Theorem 18.2 “Superposition of independent Poisson processes” (Source: MAT135B: Stochastic Processes, complete lecture notes, Spring 2011, page 200 (https://www.math.ucdavis.edu/~gravner/MAT135B/materials/)), reproduced here:

- *Assume that $N_1(t)$ and $N_2(t)$ are independent Poisson processes with rates λ_1 and λ_2. Combine them into a single process by taking the union of both sets of events or, equivalently, $N(t) = N_1(t) + N_2(t)$. This is a Poisson process with rate $\lambda_1 + \lambda_2$.*

step D

We know that:

$$
\begin{aligned}
P(X,Y) &= P(X).\, P(Y) \quad \text{when } X \text{ and } Y \text{ are independent} \\
P(X|Y) &= \frac{P(X,Y)}{P(Y)}
\end{aligned}
$$

As such:

$$
\begin{aligned}
\mathbb{P}(N_T^a = k | N_T = n) &= \frac{\mathbb{P}(N_T^a = k, N_T = n)}{\mathbb{P}(N_T = n)} \\
&= \frac{\mathbb{P}(N_T^a = k, N_T^a + N_T^b = n)}{\mathbb{P}(N_T^a + N_T^b = n)} \\
&= \frac{\mathbb{P}(N_T^a = k, N_T^b = n-k)}{e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))} . \frac{\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n}{n!}} \\
&= \frac{\mathbb{P}(N_T^a = k) . \mathbb{P}(N_T^b = n-k)}{e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))} . \frac{\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n}{n!}} \\
&= \frac{\frac{\left[\Lambda_a(0,T)\right]^k}{k!} . e^{-\Lambda_a(0,T)} . \frac{\left[\Lambda_b(0,T)\right]^{n-k}}{(n-k)!} . e^{-\Lambda_b(0,T)}}{e^{-(\Lambda_a(0,T)+\Lambda_b(0,T))} . \frac{\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n}{n!}} \\
&= \frac{\frac{\left[\Lambda_a(0,T)\right]^k}{k!} . \frac{\left[\Lambda_b(0,T)\right]^{n-k}}{(n-k)!}}{\frac{\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n}{n!}} \\
&= \frac{n!\left[\Lambda_a(0,T)\right]^k . \left[\Lambda_b(0,T)\right]^{n-k}}{k!(n-k)!\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n} \\
&= \binom{n}{k} \frac{\left[\Lambda_a(0,T)\right]^k . \left[\Lambda_b(0,T)\right]^{n-k}}{\left[\Lambda_a(0,T)+\Lambda_b(0,T)\right]^n} \\
&= \binom{n}{k} \frac{\left[\Lambda_a(0,T)\right]^k . \left[\Lambda_b(0,T)\right]^{n-k}}{\sum^n{}_{k'=0} \binom{n}{k'} \Lambda_a(0,T)^{k'} \Lambda_b(0,T)^{n-k'}}
\end{aligned}
$$

As such, we find:

step E

step F

Question 2 - Simulation

A – Simulate by thinning a Poisson process with intensity $t \to h(t) = (1-t^2)$ when $t \in [0,1]$ and 0 elsewhere.

B – How many points should produce such a process in average? Verify it on you computer.

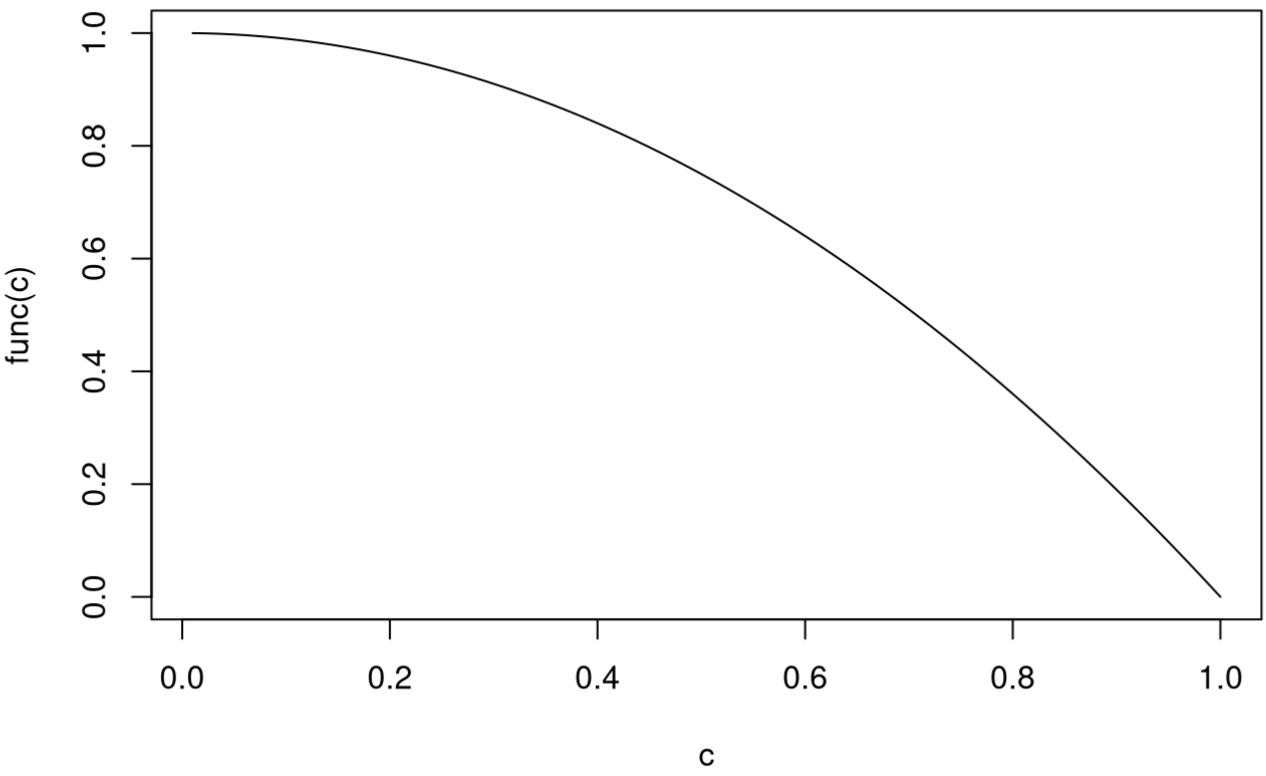
C – Simulate parents T_p according to a Poisson process of rate M. Then for each parent simulate their children that have appeared after them according to a Poisson process with intensity $h(t-T_p)$.

D – Interpret this process as a particular case of Multivariate Hawkes process with 2 processes, for which you will give spontaneous parameters and intercation functions. Find another way to simulate this by success thinning. *NB: These algorithms are complex, try first to write it down on a sheet of paper before implementing them*

step A

Overall, the intensity function decreases over the space $t \in [0,1]$ as such:

```
func <- function(t){1-t^2}
c = seq(1,100,1)/100
plot(c, func(c), type="l")
```



Intuitively, we would expect the poisson process to not always output a single point as the intensity is a decreasing function of time.

So, given the intensity $h(t) = (1 - t^2)$, we set $M = 1$ as $\forall t \in [0,1],\ h(t) <= 1$. We produce the following thinning simulator:

```
poisson_process_simulation_thinning <- function(){
  ### Simulate by thinning a Poisson process with intensity t->h(t)=(1−t2)
  ### when t∈[0,1] and 0 elsewhere.
  M = 1
  Tmax = 1
  func <- function(t){rbinom(1,1,1-t^2)}
  simulation = cumsum(rexp(100, M))
  simulation = matrix(simulation[simulation<=Tmax])
  if (length(simulation)==0) {
    return(NA)
  }
  ret = c()
  for (i in 1:length(simulation)){
    if (func(simulation[i])==1){
      ret = cbind(ret, simulation[i])
    }
  }
  if (length(ret)==0){
    return(NA)
  } else {
    ret[,!is.na(ret)]
  }
}
```

Then we compute 100 simulations of this thinning process as example:

```
for (i in 1:100){
  cat("Simulation #", i, ": ", poisson_process_simulation_thinning(), "\n")
}
```

```
## Simulation # 1 :  NA
## Simulation # 2 :  0.0004917091
## Simulation # 3 :  0.7617695
## Simulation # 4 :  0.03497047
## Simulation # 5 :  NA
## Simulation # 6 :  NA
## Simulation # 7 :  NA
## Simulation # 8 :  NA
## Simulation # 9 :  NA
## Simulation # 10 :  0.5615121
## Simulation # 11 :  NA
## Simulation # 12 :  NA
## Simulation # 13 :  NA
## Simulation # 14 :  NA
## Simulation # 15 :  0.3948427 0.5919424 0.673913
## Simulation # 16 :  NA
## Simulation # 17 :  NA
## Simulation # 18 :  0.1522257
## Simulation # 19 :  0.9250771
## Simulation # 20 :  0.4111415
## Simulation # 21 :  NA
## Simulation # 22 :  NA
## Simulation # 23 :  NA
## Simulation # 24 :  0.2781403
## Simulation # 25 :  NA
## Simulation # 26 :  NA
## Simulation # 27 :  0.8478862
## Simulation # 28 :  NA
## Simulation # 29 :  0.1730511
## Simulation # 30 :  0.4482453
## Simulation # 31 :  0.1093704 0.1904764
## Simulation # 32 :  0.3663267
## Simulation # 33 :  NA
## Simulation # 34 :  NA
## Simulation # 35 :  NA
## Simulation # 36 :  NA
## Simulation # 37 :  0.08740683 0.6983861
## Simulation # 38 :  NA
## Simulation # 39 :  0.4298249
## Simulation # 40 :  NA
## Simulation # 41 :  0.1839045 0.2811861 0.3478896 0.4200178
## Simulation # 42 :  NA
## Simulation # 43 :  0.861136
## Simulation # 44 :  NA
## Simulation # 45 :  0.28151
## Simulation # 46 :  0.4653493
## Simulation # 47 :  NA
## Simulation # 48 :  NA
## Simulation # 49 :  0.7907415
## Simulation # 50 :  0.316209
## Simulation # 51 :  0.5205582
## Simulation # 52 :  NA
## Simulation # 53 :  0.4562968
## Simulation # 54 :  0.2169355
## Simulation # 55 :  0.03221289
## Simulation # 56 :  0.04376335 0.7241839
## Simulation # 57 :  0.8019231
## Simulation # 58 :  0.634905
## Simulation # 59 :  0.1956555
## Simulation # 60 :  NA
## Simulation # 61 :  NA
## Simulation # 62 :  NA
## Simulation # 63 :  NA
## Simulation # 64 :  NA
## Simulation # 65 :  NA
## Simulation # 66 :  NA
## Simulation # 67 :  0.3876761
## Simulation # 68 :  0.5528336
```

```
## Simulation # 69 :  NA
## Simulation # 70 :  0.05754798
## Simulation # 71 :  NA
## Simulation # 72 :  0.3039784
## Simulation # 73 :  0.4895135 0.7413382
## Simulation # 74 :  NA
## Simulation # 75 :  0.4630703
## Simulation # 76 :  0.07120253
## Simulation # 77 :  0.2846515 0.3653575
## Simulation # 78 :  0.3075453 0.5297781
## Simulation # 79 :  NA
## Simulation # 80 :  0.7386869
## Simulation # 81 :  0.8327803
## Simulation # 82 :  0.6104852
## Simulation # 83 :  0.2888014
## Simulation # 84 :  NA
## Simulation # 85 :  NA
## Simulation # 86 :  0.1746774
## Simulation # 87 :  0.04439858
## Simulation # 88 :  0.488683 0.5045025 0.7589008
## Simulation # 89 :  NA
## Simulation # 90 :  NA
## Simulation # 91 :  NA
## Simulation # 92 :  NA
## Simulation # 93 :  0.8520809
## Simulation # 94 :  0.6490003
## Simulation # 95 :  NA
## Simulation # 96 :  0.3321973
## Simulation # 97 :  0.1471145
## Simulation # 98 :  NA
## Simulation # 99 :  0.8584036
## Simulation # 100 :  NA
```

step B

On average on $t \in [0,1]$, we have:

$$\begin{aligned}
N_[0,1] &\sim \mathbb{P}(\int_0^1 (1-t^2)dt) \\
\mathbb{E}[N_{[0,1]}] &= \int_0^1 (1-t^2)dt \\
&= \left[t - \frac{t^3}{3}\right]_0^1 \\
&= 1 - \frac{1}{2} \\
&= \frac{2}{3}
\end{aligned}$$

The average amount of created points should be $\frac{2}{3}$ over $t \in [0,1]$. We can confirm this by going back to our simulation function and count the average number of generated points over the total number of simulations.

```
count_nb_point_generated <- function(){
  ### Counts the number of generated points during 1000 poisson process simulation with thinning
  counter = 0
  for (i in 1:1000){
    simulation = poisson_process_simulation_thinning()
    if (length(simulation)==0 || !is.na(simulation)){counter = counter + length(simulation)}
  }
  return(list("nb"=counter, "rate"=counter/1000))
}

example = count_nb_point_generated()
cat("Example -- ", example$nb, " points were generated over 1000 simulations, i.e. a rate of ", example$r
ate, ".")
```

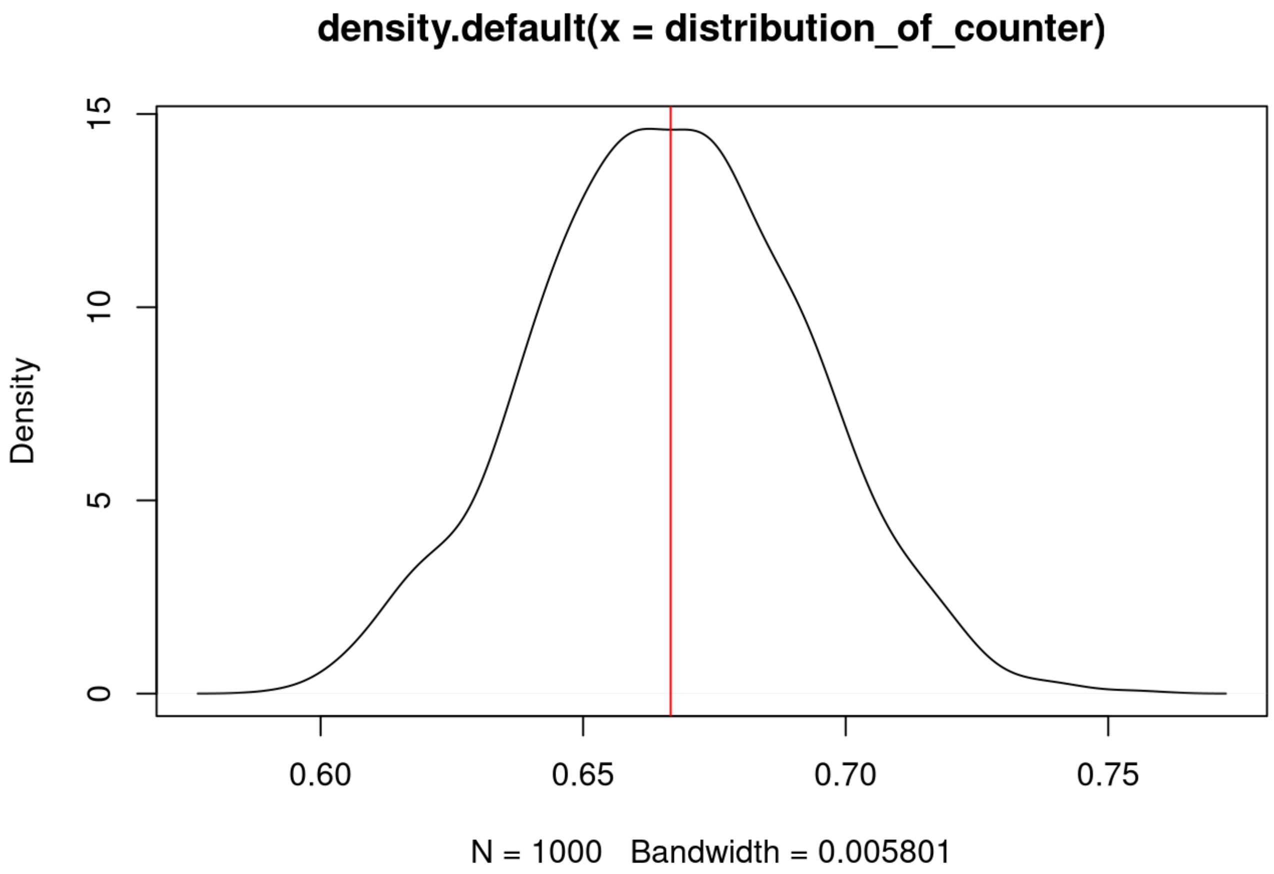
```
## Example --  673  points were generated over 1000 simulations, i.e. a rate of  0.673 .
```

We plot out the empirical distribution of the simulated rate after 1000 batches of 1000 simulations with, in red, the previously computed expected average number of generated points.

```r
distribution_of_counter = apply(matrix(c(1:1000)), 1, function(x){count_nb_point_generated()$rate})
cat("Mean rate obtained after 1000 batches of 1000 simulations", mean(distribution_of_counter))
```

```
## Mean rate obtained after 1000 batches of 1000 simulations 0.666493
```

```r
plot(density(distribution_of_counter))
abline(v=2/3, col="red")
```



Our empirical observation seems to match our mathematical computation.

step C

step D