Simulating Poisson processes

University of Toronto, ACT 350

Prof. S. Pesenti

08/09/2020

How to simulate a Poisson process

Recall that a Poisson process with rate $\lambda > 0$ is a counting process $\{N(t),\ t \geq 0\}$ that has the following properties

1. *starts at 0*: $N(0) = 0$ with probability 1
2. *is increasing*: $N(s) \leq N(t)$ for all $s \leq t$
3. *has Poisson distributed increments*: $P(N(s+t) - N(s) = n) \sim \text{Pois}(\lambda t)$ for all $s, t \geq 0$.

However, this definition does not help if we want to simulate a Poisson process. Thus, to simulate a Poisson process we use its alternative definition. That is, for independent random variables $X_1, X_2, \ldots$ that are Exponentially distributed with rate $\lambda > 0$, a Poisson process with rate $\lambda > 0$ can be written as

$$N(t) = \max\Big\{ n \in \mathbb{N} \ \Big| \ \sum_{i=1}^{n} X_i \leq t \Big\}, \quad \text{for all } t \geq 0.$$

Thus, for simulating a Poisson process we first need to simulate exponential random variables, add them up and check whether they are smaller or equal to t. However, we will run into problems when we want to store the value of $N(t)$ for every $0 \leq t < \infty$. The way out is to implement $N(t)$ as a function of t in the following way. First generate the jump points of the Poisson process. Second, we know that a Poisson process can only jump one unit at a time. Thus, we can define the Poisson process as a step function that jumps exactly one unit at every jump point.

1. Generating jump times

This is an algorithm for generating the jump times of a Poisson process

1. Set $\tau_0 = 0$.
2. For $k \geq 1$ do
   a) generate $X \sim$Exp(λ)
   b) set $\tau_k = \tau_{k-1} + X$

The $\tau_k,\ k = 0, 1, \ldots$ are the time points, where the Poisson process jumps.

Numerical implementation: R code

```
# set the seed for generating random variables (for reproducability)
# set.seed(2019)

# set the rate of the Poisson process/Exponential distribution
lambda <- 2
# number of jumps
n <- 50
# initialise the jump times
jump_times <- rep(0, lenthought = n)

# note that the first value of the jump_times is aleady 0
for(i in 2:n){
  jump_times[i] <- jump_times[i - 1] + rexp(1, lambda)
}
jump_times
```

```
##  [1]  0.0000000  0.3657863  0.6985693  0.9037437  1.3874964  1.5824862
##  [7]  2.7414484  2.9258883  3.5194333  4.3018563  4.8163236  5.0802495
## [13]  5.2966542  6.2712359  6.5429966  8.7170691  8.8094938  9.3510023
## [19]  9.4226721 10.1646970 10.2452846 11.0375235 11.5509258 11.6724420
## [25] 11.8054088 12.0322339 12.2532001 12.5912044 12.6569167 13.4786725
## [31] 14.6044068 14.7282155 14.9488410 16.5839158 17.0343030 18.1863211
## [37] 18.9673347 20.1408289 20.6653919 20.8535683 21.0323070 21.9087191
## [43] 22.5147239 22.7129120 24.6344521 24.8173668 25.5318620 25.6795619
## [49] 28.3016039 28.3480622
```

A more efficient implementation, avoiding the for loop, is

```
jump_times_2 = c(0, cumsum(rexp(n-1, lambda)))
# note that the two generated sample_paths are different,
# since the Exponential random variables are different.
jump_times_2
```

```
##  [1]  0.0000000  0.4009741  0.5440014  1.4830016  1.7637160  1.9839844
##  [7]  2.3296803  2.7128009  4.2811846  5.2633345  6.1325501  6.1955119
## [13]  6.8141297  7.2022298  7.8788800  8.1415832  8.1672448  8.8373971
## [19]  9.2143069 10.2757270 10.4982963 11.5811705 11.5858376 12.5710177
## [25] 12.8327120 13.0030185 15.9318224 16.2933001 16.3322137 18.9462440
## [31] 19.1759323 19.2258522 19.3492272 20.5470677 20.9160906 21.7731081
## [37] 21.8319746 23.2369857 23.7165745 24.7505991 25.0722585 25.8092987
## [43] 26.1613257 26.2784365 26.5230741 26.7588178 26.9302098 28.1475509
## [49] 29.2782265 29.4516041
```

2. Defining the Poisson process as a function

Next, we can define the Poisson process as a step function that jumps exactly one unit at every value in `jump_times`. Note that the implemented Poisson process is now a *function* of the time t.

```
# define the Poisson process as a function
poisson_process <- stepfun(jump_times[2 : 50], seq(0, n - 1, by = 1))
# the first argument of `stepfun` is where the step function jumps
# the second argument is the hight of the step function

# Let us check some values.
# The starting value is 0.
poisson_process(0)
```

```
## [1] 0
```

```
# What is the value of N(3)? (Note that we use jump_times and not jump_times_2)
poisson_process(3)
```

```
## [1] 7
```

```
poisson_process(10)
```

```
## [1] 18
```

```
poisson_process(15)
```

```
## [1] 32
```

The first jump time is 0.3657863. Let us check whether the Poisson process jumps at that time.

```
# The first jump of the Poisson process is
first_jump <- jump_times[2]
poisson_process(first_jump - 0.001)
```

```
## [1] 0
```

```
poisson_process(first_jump + 0.001)
```

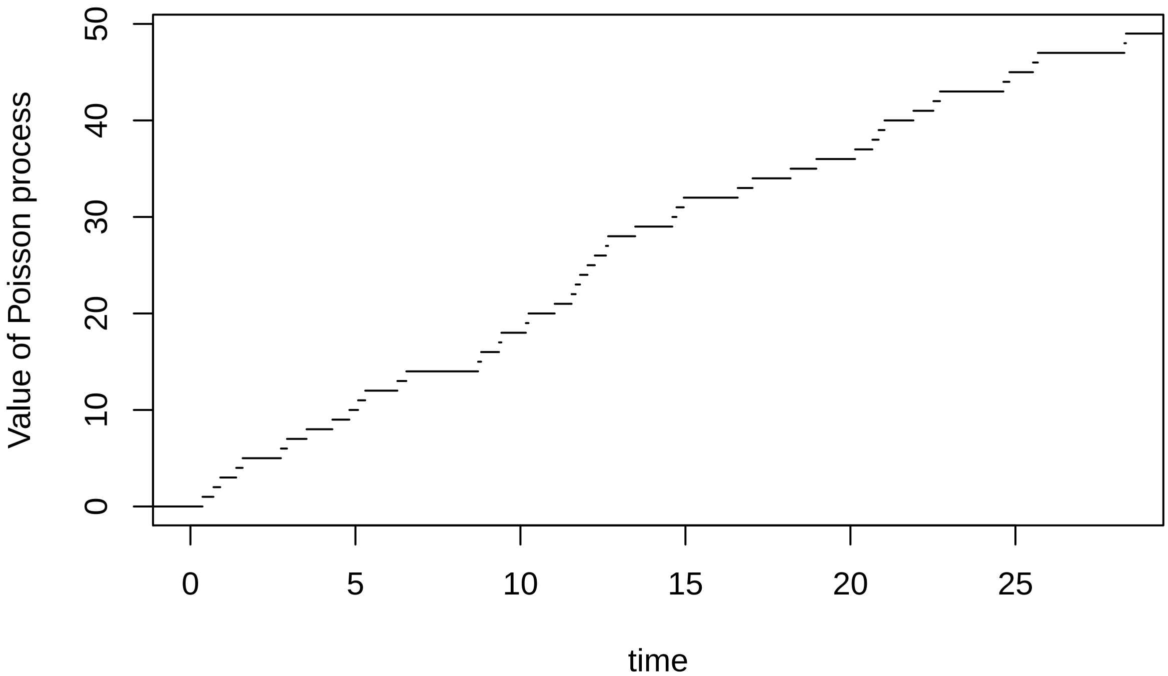
```
## [1] 1
```

Note you can evaluate the Poisson process for any time point t, however, keep in mind that we only simulated 50 jumps of the Poisson process. Thus, evaluating `N(t)` for t large does not make sense.

3. Plotting a sample path

To plot a sample path we have to plot the implemented step function `poisson_process`.

```
plot(poisson_process, xlab = "time", ylab = "Value of Poisson process", main = NULL,
     verticals = FALSE, do.points = FALSE, xlim = c(0,jump_times[n]))
```



Question: Rerun the code, what do you observe?
Question: Rerun the code with a different λ or change the number of jumps. How does the sample path change?

Renewal Theory

A renewal process is defined as follows: Consider a sequence $T_1, T_2, \ldots$ of independent identically distributed (i.i.d.) non-negative random variables. Define the stochastic process

$$X_0 = 0\,, \quad X_n = T_1 + \cdots + T_n\,, \quad n = 1, 2, \ldots\,,$$

and set

$$N(t) = \max\left\{n \in \mathbb{N} \;\Big|\; \sum_{i=1}^{n} X_i \leq t\right\}, \quad \text{for all } t \geq 0.$$

Then the process $\{N(t)\,|\,t \geq 0\}$ is called a renewal process.

If the $T_i\,, i = 1, 2, \ldots$ are i.i.d. Exponential with parameter $\lambda > 0$, then the renewal process $\{N(t)\,|\,t \geq 0\}$ is a Poisson process.

Task:

a) Generate a code to simulate from a renewal process, where the interarrival times $T_i\,, i = 1, 2, \ldots$ are i.i.d. LogNormal distributed with mean 2 and standard deviation 0.5. *Hint: use the function* `rlnorm`.

b) Plot a sample path of the renewal process define in **a)**.

c) What do you observe compared to the Poisson process?

d) Repeat tasks **a)** to **b)** with interarrival times being Gamma distributed with the same mean and standard deviation as in **a)**. *Hint: use the function `rgamma`.*

e) Compare the Poisson process with the renewal process from **a)** and the renewal process from **d)** and give interpretations.