Simulating Poisson processes

University of Toronto, ACT 350

Prof. S. Pesenti

19/10/2019

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 $X_1, X_2, \dots$ that are Exponentially distributed with rate $\lambda > 0$, a Poisson process with rate $\lambda > 0$ can be written as

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

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  0.7710511  1.9766163  2.5490375  3.8402585  4.3894914
##  [7]  4.5991250  5.2985292  5.5634107  6.0447705  6.6841338  6.7330930
## [13]  7.9062607  8.5320512  9.5138575  9.9292654 10.3527286 10.5161066
## [19] 11.2484897 11.6012433 11.7904597 12.0419340 12.8653281 13.0935543
## [25] 14.0772328 14.3700252 14.3704266 14.5123538 14.7670595 15.0328448
## [31] 15.2067167 16.8173278 17.1601292 19.1450169 20.2356333 20.6471794
## [37] 21.6085865 22.2734971 22.3758865 22.4861826 23.3388395 23.3526366
## [43] 23.4580754 24.1411786 24.4912990 24.7683712 25.1674035 26.1955755
## [49] 26.2422427 26.8822922
```

A more efficient implementation, avoiding the for loop, is

```
jump_times_2 = 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]  0.09787964  0.67355308  1.79832867  2.50241717  2.64045638  2.88000501
##  [7]  3.70988024  3.80590905  4.35522367  4.42168202  5.05323435  5.28817854
## [13]  7.32387666  7.53815983  7.75520884  8.12326497  9.35796371 10.13898046
## [19] 10.26638269 10.47626295 10.56056457 10.67376468 11.12322739 11.80503020
## [25] 12.55199219 12.55428373 12.92831138 13.82868334 13.84061270 14.03072853
## [31] 14.08215935 14.42053169 14.45806303 14.90810019 15.18942510 15.97733679
## [37] 16.31668801 16.90883674 16.93651483 16.96316577 17.10216948 17.78880329
## [43] 19.17752464 19.60783778 21.34105342 21.86050181 22.00320659 23.19247609
## [49] 23.22252249
```

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.

```r
# 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
```

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

```
## [1] 3
```

```r
poisson_process(10)
```

```
## [1] 15
```

```r
poisson_process(15)
```

```
## [1] 28
```

The first jump time is 0.7710511. Let us check whether the Poisson process jumps at that time.

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

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

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

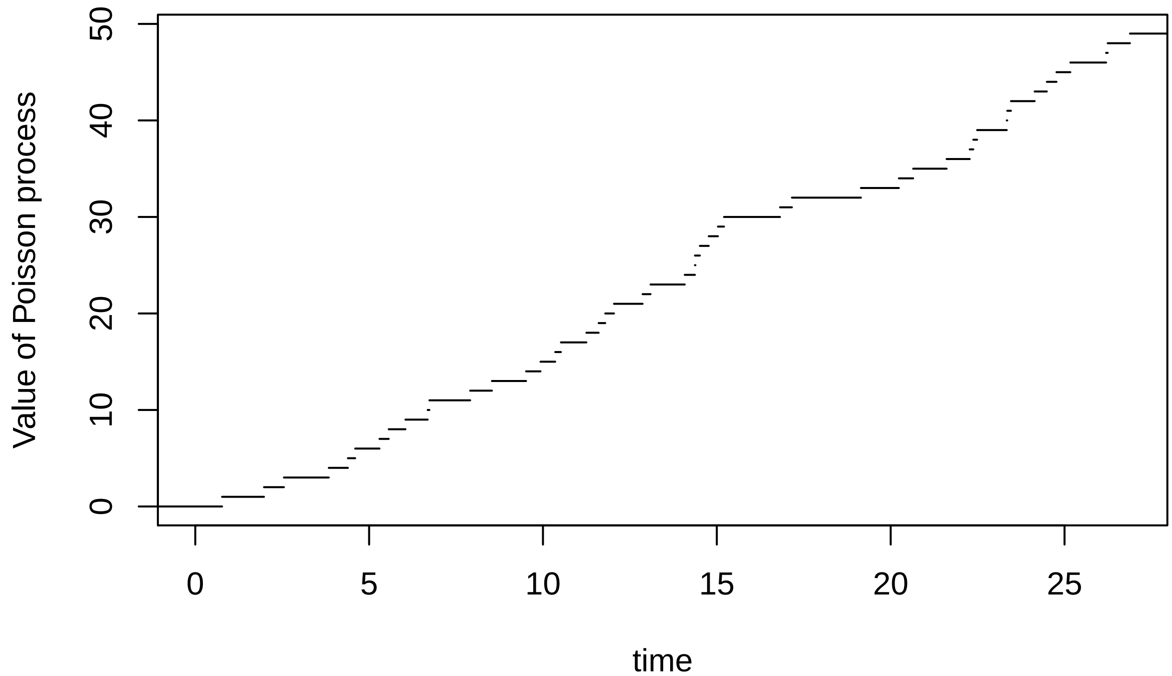
```
## [1] 1
```

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`.

```r
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

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

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.