Simulating Poisson Processes

Nikhil A Shah

February 3, 2023

A Poisson process is a continuous-time stochastic process that can be used in a variety of applications. The main purpose of the Poisson process is to count the number of random events that occur in some time period. What differentiates the Poisson process from say a normal counting measure is the assumptions that are enforced on the realisation of the random events, namely that the random events occur independently from one another and that the random events occur at some fixed average rate. These two demands on the structure yield the processes name; these underlying assumptions are reminiscent of the Poisson distribution. We will move through two different ways of building these Poisson processes as well as discussing their implementations.

The code that accompanies these notes can be found at this link: `https://github.com/shahnikhil12/Simulation-of-Stochastic-Processes/blob/main/Simulation%20of%20Poisson%20Process.ipynb`

In Section 1, we go over some of the foundational theory of the Poisson process such as its definition and the governing distribution. Section 2 goes over a method of simulating the Poisson process by using a Bernoulli approximation in each time window, whilst Section 3 gives the construction of the Poisson process by using interarrival times

Contents

1 Properties of the Poisson Process

We first begin with a brief overview of the theoretical structure of the Poisson process. In its simplest description, a Poisson process is a counting process that tracks the number of random events that have occurred before some (deterministic) time t ie

Definition 1.1 (Poisson Process)**.** *A (homogenous) Poisson process, $N(t)$ is a counting process satisfying the following conditions:*

1. *$N(0) = 0$ and for $s < t$ then $N(s) \leqslant N(t)$*
2. *$\mathbb{P}\left[N(t+h) = n+m | N(t) = n\right] = \begin{cases} 1-\lambda h + o(h) & m = 0 \\ \lambda h + o(h) & m = 1 \\ o(h) & m > 1 \end{cases}$*
3. *if $s < t$, then $N(t) - N(s)$ is independent of $N(s)$*

Note in particular, the second condition which states the in each time period $(t, t+h]$, the probability that the Poisson process increases by 1 can be viewed as Bernoulli trial. In conjunction with the third condition, this gives a way of teasing out the distribution of the Poisson process by making use of a Binomial approximation by making use of a Poisson Limit Theorem style argument (we will go into further detail of this in the next section as a motivation for the one of the potential constructions for the Poisson process). For now, we state and prove the form of the distribution of the Poisson process

Theorem 1.2. *$N(t)$ has a Poisson(λt) distribution, that is*

$$\mathbb{P}\left[N(t) = n\right] = \frac{(\lambda t)^n e^{-\lambda t}}{n!} \tag{1}$$

Proof. The proof will take the following course. First, we will derive a condition that relates the derivatives of the probability to itself. Second, we shall make use of generating functions to give a full distribution.
For clarity of expression, for the rest of this proof, we let $p_n(t) = \mathbb{P}[N(t) = n]$

$$\begin{aligned} p_n(t+h) = \mathbb{P}[N(t+h) = n] &= \sum_m \mathbb{P}[N(t+h) = n | N(t) = m]\mathbb{P}[N(t) = m] && (2) \\ &= (1 - \lambda h + o(h))\mathbb{P}[N(t) = n] && (3) \\ &\quad + (\lambda h + o(h))\mathbb{P}[N(t) = n-1] && (4) \\ &\quad + o(h)\mathbb{P}[N(t) \leqslant n-2] && (5) \\ &= (1-\lambda h)p_n(t) + \lambda h p_{n-1}(t) + o(h) && (6) \\ \implies \quad \frac{p_n(t+h) - p_n(t)}{h} &= \lambda p_{n-1}(t) - \lambda p_n(t) + o(1) && (7) \\ \implies p_n'(t) &= \lambda(p_{n-1}(t) - p_n(t)) && (8) \end{aligned}$$

In conjunction with an initial condition that $p_0'(t) = -\lambda p_0(t)$, we have a system that we can solve for, concluding the first part of this proof

To finish this proof, let

$$G(s,t) = \sum_{n=0}^{\infty} p_n(t)s^n = \mathbb{E}\left[s^{N(t)}\right] \tag{9}$$

We now consider the derivative of the G with respect to t:

$$\begin{aligned} \frac{\partial}{\partial t}G(s,t) = \sum_{n=0}^{\infty} p_n'(t)s^n &= \sum_{n=0}^{\infty} \lambda(p_{n-1}(t) - p_n(t))s^n && (10) \\ &= \lambda \sum_{n=0}^{\infty} p_{n-1}(t)s^n - \lambda \sum_{n=0}^{\infty} p_n(t)s^n && (11) \\ &= \lambda s G(s,t) - \lambda G(s,t) && (12) \end{aligned}$$

Hence we have to solce

$$\frac{\partial}{\partial t}G(s,t) = \lambda(s-1)G(s,t) \qquad G(s,0) = 1 \tag{13}$$

which has solution

$$G(s,t) = e^{-(1-s)\lambda t} = \sum_{n=0}^{\infty} \left(\frac{(\lambda t)^n e^{-\lambda t}}{n!} \right) s^n \tag{14}$$

hence we see that $\mathbb{P}[N(t) = n] = \frac{(\lambda t)e^{-\lambda t}}{n!}$ or rather, $N(t)$ has a Poisson(λt) distribution $\square$

2 Bernoulli Approximation

2.1 Theory

Our first method of creating a Poisson process is by creating a Bernoulli approximation. For some small timestep dt, the probability that we have a jump in our process is a Bernoulli trial with probability of success being λdt. Notice that since the probability that our process jumps by two is $o(h)$ so we have only two possible outcomes. Further, since each increment is independent, if we view our time period as a succession of smaller time windows, in which we have independent Bernoulli trials, we then can approximate the overall Poisson process by a binomial approximation. Thus, it remains to show that this Bernoulli approximation converges in distribution to the desired Poisson process. We will now show this

Theorem 2.1. *For some $T > 0$ and $k \in \mathbb{R}_{>0}$, create a time windows $\tau_n(i) = \left(\frac{iT}{n}, \frac{(i+1)T}{n}\right)$. Then*

$$\lim_{n\to\infty} \mathbb{P}[N(t) = k] = \frac{(\lambda t)^k e^{-\lambda t}}{k!} \tag{15}$$

Proof. Since we have a sequence of independent Bernoulli trials, we are working with a Binomial distribution, ie $N(t) \sim \text{Bin}\left(n, \frac{\lambda t}{n}\right)$ Hence,

$$\begin{aligned}\lim_{n\to\infty} \mathbb{P}\left[N(t) = k\right] &= \lim_{n\to\infty} \binom{n}{k} \left(\frac{\lambda t}{n}\right)^k \left(1 - \frac{\lambda t}{n}\right)^{n-k} &(16)\\ &= (\lambda t)^k \lim_{n\to\infty} \binom{n}{k} n^{-k} \left(1 - \frac{\lambda t}{n}\right)^n \left(1 - \frac{\lambda t}{n}\right)^{-k} &(17)\end{aligned}$$

We will now consider each term in the limit one by one:

$$\lim_{n\to\infty} \left(1 - \frac{\lambda t}{n}\right)^n = e^{-\lambda t} \tag{18}$$

$$\lim_{n\to\infty} \left(1 - \frac{\lambda t}{n}\right)^{-k} = 1 \tag{19}$$

$$\begin{aligned}\lim_{n\to\infty} \binom{n}{k} n^{-k} &= \frac{1}{k!} \lim_{n\to\infty} \frac{n!}{(n-k)! n^k} &(20)\\ &= \frac{1}{k!} \lim_{n\to\infty} \frac{n(n-1)\cdots(n-k+2)(n-k+1)}{n^k} &(21)\\ &= \frac{1}{k!} \lim_{n\to\infty} \frac{o(n^k)}{n^k} &(22)\\ &= \frac{1}{k!} &(23)\end{aligned}$$

Hence, putting everything together, we get that

$$\lim_{n\to\infty} \mathbb{P}\left[N(t) = k\right] = \frac{(-\lambda t)^k e^{\lambda t}}{k!} \tag{24}$$

□

This then gives us a nice way of creating our Poisson process

2.2 Application

To create our Poisson process, we start by generating pseudo-random numbers using Python's numpy library. The idea is that if we have a list of numbers that are generated from a Uniform[0,1] distribution, we can create our Bernoulli trial from above since, for a chosen intensity, λ, time period, T, and number of windows, n, we can choose to 'accept' values of our uniform random variable that are below $\frac{\lambda T}{n}$:

$$u \sim \text{Unif}[0, 1] \implies \mathbb{P}\left[u \leqslant \frac{\lambda T}{n}\right] = \frac{\lambda T}{n} \tag{25}$$

Hence, if we create a list of n unit uniform random variables, each time $u_i \leqslant \frac{\lambda T}{n}$, for $i \leqslant n$, we add one to our running count

The benefit of creating the Poisson process this way is that the user can control the final time of the process, and by varying the number of windows, can control the resolution of the process. It does come with the drawback that the user cannot control the number of jumps that occur. In Figure 1 below, we can see exactly this phenomenon: note the each path stops at our choice of T in time, but the final value of the process is different. The code that was used to create these paths is available on my github as well as in the appendix of this paper

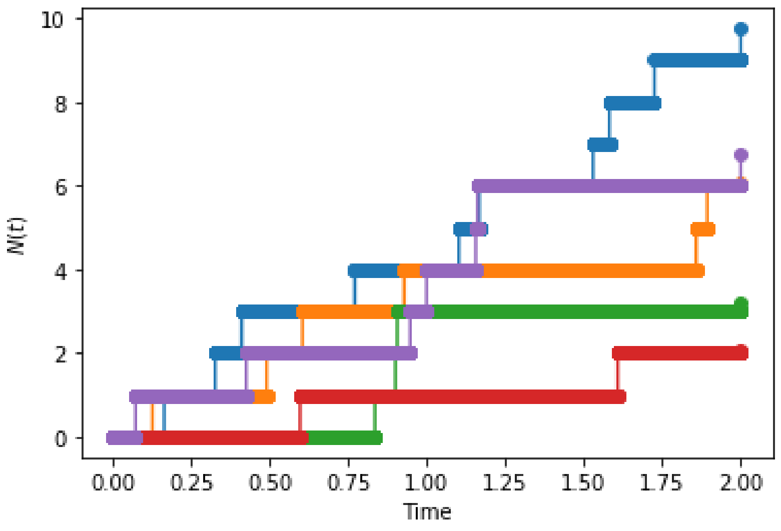

Figure 1: Here are a selection of sample paths generated using the above method, the chosen parameters are $(T, N, \lambda) = (2, 10000, 3)$. As noted above, each sample ends at $T = 2$ by the values of $N(T = 2)$ are different

2.3 Code

```
def poisson_process_v1(seed = 0, T = 1, N = 100, rate = 10, X_0 = 0, paths = 1):
#Idea: Bernoulli Approximation

        np.random.seed(seed)
        dt = T/N
        size = (paths, N+1)

        X = np.zeros(size)

        X = np.random.rand(size[0], size[1])

        for j in range(paths):
                for i in range(1, N):
                        if X[j, i] <= rate * dt:
                                X[j, i] = 1

                        else:
                                X[j, i] = 0

        X[:, 0] = X_0

        X = np.cumsum(X, axis = 1)
```

3 Interarrival Times

3.1 Theory

The second method of creating the Poisson process is by noting that the time between jumps in the process are given by the Exponential distribution. We will show this by looking at the amount of time one has to wait for the first event to occur, and then extend by a mix of the Markov Property and the Memoryless Property of the Exponential Distribution. To begin, let us prove the Memorylessness of the Exponential distribution

Theorem 3.1 (Memorylessness of the Exponential Distribution)**.** *Let X be an Exponential random variable with rate $\lambda > 0$. Then $\forall s, t \geqslant 0$:*

$$\mathbb{P}[T > s + t | T > s] = \mathbb{P}[T > t] \tag{26}$$

Proof. The proof follows by simple application of conditional probabilities. Recall that if $T \sim \text{Exp}(\lambda)$, then $F_T(t) = 1 - e^{-\lambda t}$. so

$$\begin{align}
\mathbb{P}[T > s + t | T > s] &= \frac{\mathbb{P}[\{T > s + t\} \cap \{T > s\}]}{\mathbb{P}[T > s]} \tag{27} \\
&= \frac{\mathbb{P}[T > s + t]}{1 - \mathbb{P}[T < s]} \tag{28} \\
&= \frac{1 - F_T(s + t)}{1 - F_T(s)} \tag{29} \\
&= \frac{1 - (1 - e^{-\lambda(s+t)})}{1 - (1 - e^{-\lambda s})} \tag{30} \\
&= e^{-\lambda t} = \mathbb{P}[T > t] \tag{31}
\end{align}$$

□

Theorem 3.2. *The interarrival times in a Poisson process, $N(t) \sim$ Poisson(λt), follows an exponential random variable of rate λ*

Proof. Let T_1 be the time at which the first event occurs. The probability that at least one event occurs before some time t is the complement of event that no events occur before time t, ie $\{N(t) \geqslant 1\}^C = \{N(t) = 0\}$. Computing the cumulative distribution of T_1 yields:

$$\begin{align}
F_{T_1}(t) &= \mathbb{P}[N(t) \geqslant 1] \tag{32} \\
&= 1 - \mathbb{P}[N(t) = 0] \tag{33} \\
&= 1 - \frac{(\lambda t)^0 e^{-\lambda t}}{0!} \tag{34} \\
&= 1 - e^{-\lambda t} \tag{35}
\end{align}$$

Which is exactly the cumulative distribution function of an exponential random variable with rate λt. By uniqueness of the cumulative distribution functions, we can deduce that the interarrival times follow an Exponential distribution with rate λt

This result can be extended to all interarrival times by noting the independence of increments that is a defining condition of the Poisson process □

3.2 Application

The idea is to generate a series of exponential random variables and use these to model our interarrival times. Once we have these, we can then let our Poisson process increase by 1 after each interarrival time

Given a unit uniformly distributed random variable, $u \sim$ Unif$[0, 1]$, we can construct Exponential random variables by using the inverse cdf transform by setting, for each sampled u_i, $x = -\frac{1}{\lambda}\log(1-u_i)$. We can then verify that we do get an Exponential random variable since

$$F_X(x) = \mathbb{P}[X \leqslant x] = \mathbb{P}\left[-\frac{1}{\lambda}\log(1-u) \leqslant x\right] = \mathbb{P}\left[1-u \geqslant e^{-\lambda x}\right] = \mathbb{P}[u \leqslant 1-e^{-\lambda x}] = 1-e^{-\lambda x} \tag{36}$$

The benefit of this method is that we complete freedom to choose the number of jumps that we want to occur; by choosing how many times we sample from the unit Uniform distribution, we then generate exactly the number of interarrival times. This allows us to control the final value of the Poisson process. However, the drawback of this method is that we have no control over the final time at which the process terminates. This can be seen from Figure 2 below, where each sample path stops after 10 events have occured, but the end at different times

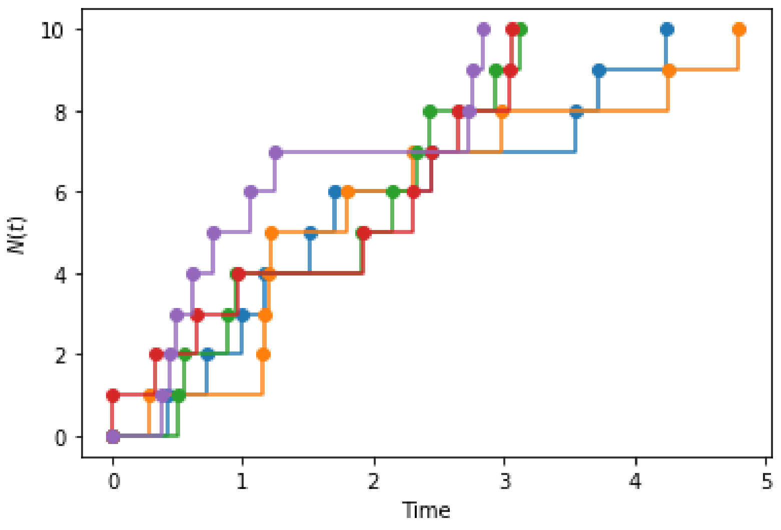

Figure 2: Here are a selection of sample paths generated using the above method, the chosen parameters are $(\lambda, N(T)) = (3, 10)$. As noted above, each sample ends at $N(T) = 10$ by the values of T are different

3.3 Code

```
def poisson_process_v2(seed = 0, X_0 = 0, lam = 3, events = 5, paths = 3):
#Idea: Model via Inter-Arrival Times

        np.random.seed(seed)

        interarrival_periods = - np.log(1-np.random.rand(paths, events+1) ) / lam

        interarrival_periods[:, 0] = X_0

        jump_time = interarrival_periods.cumsum(axis = 1)

        jump_chain = np.arange(0, events+1, 1)
```