SIMULATING THE POISSON PROCESS

PATRICK MCQUIGHAN

Abstract. This paper describes a way to simulate a Poisson process. It begins by explaining what the Poisson process is, and how it can model natural phenomena. Then it provides an easy way to simulate this process based on the waiting times between the occurrence of events. Lastly, it discusses the generation of pseudo-random numbers. It concludes with an algorithm for a simulation, and results from a simulation generated using Python.

Contents

1. Introduction 1
2. Defining the Poisson Process 2
3. Simulating random variables 5
4. Data 7
Acknowledgments 7
References 7

1. Introduction

Poisson processes occur in nature; some examples include people arriving at a store at a fixed rate, or the number of radioactive particles discharged in a fixed amount of time [1]. The Poisson process is a random process which counts the number of random events that have occurred up to some point t in time. The random events must be independent of each other and occur with a fixed average rate. In Section 2, these properties are formalized, and we show that the number of events that occur before time t follows a Poisson distribution, motivating the name for Poisson process. We prove that the waiting time from one event to the next is an exponentially distributed random variable. So to simulate the process, we only need a sequence of exponentially distributed random variables.

Computers do not have a source for generating a sequence of independent identically distributed (i.i.d.) random variables of a given distribution and instead must create *pseudo-random numbers*. These numbers are created deterministically by a function. In contrast, a sequence of truly random numbers would be generated as a result of a physical experiment, such as rolling a die. A good pseudo-random number generator must generate a sequence that satisfies the same properties that a sequence of random numbers has. The most common way to generate a sequence of uniform random variables is by creating a sequence of natural numbers with maximal value M, which can be turned into a pseudo-random sequence in $[0, 1]$ by

Date: July 23, 2010.

dividing all numbers by M. In Section 3, we show how this sequence can be used to generate a sequence of i.i.d. random variables with exponential distribution.

In Section 4, we combine these results to create an algorithm for simulating the Poisson process, and display the results of running the simulation. The most important graph is a histogram which shows that the Poisson process does indeed appear to be related to the Poisson distribution.

In brief, in Section 2 we introduce Poisson processes and study some properties. In Section 3 we explore ways to produce i.i.d. random variables of a given distribution. In Section 4, we simulate Poisson processes.

2. Defining the Poisson Process

Let λ be a positive real number.

Definition 2.1. A *Poisson random variable* X with parameter λ is a discrete random variable such that

$$\mathbf{P}\{X = k\} = \frac{e^{-\lambda}\lambda^k}{k!}, \qquad k = 0, 1, 2, \cdots \tag{2.2}$$

Definition 2.3. The *Poisson distribution*, $Poi(\lambda)$ is a discrete probability distribution with a single parameter λ that has probability mass at k defined as in Equation 2.2.

Definition 2.4. A *counting process* $N(t)$ is a stochastic process defined by the number of occurrences of a random event before time t.

Definition 2.5. A *Poisson process* is a counting process satisfying the following conditions:

(1) $N(0) = 0$.
(2) For all $t_1 \leq t_2 \leq s_1 \leq s_2$ the random variables $N(t_2) - N(t_1)$ and $N(s_2) - N(s_1)$ are independent.
(3) There exists a $\lambda > 0$ such that given any $0 \leq t_1 < t_2$, $\mathrm{E}[N(t_2) - N(t_1)] = \lambda(t_2 - t_1)$
(4) If $P(s) = \mathbf{P}\{N(t+s) - N(t) > 2\}$, then $\lim_{s \to 0} \frac{P(s)}{s} = 0$.

Condition 2 can be restated as "the number of occurrences in one time interval is independent of the number of occurrences in any other time interval," known as *independent increments*. Condition 3 assures that the expected number of occurrences between intervals of the same size is constant, known as *stationary increments*. Condition 4 formalizes the idea that occurrences only happen one at a time. The limit represents this notion by saying that the probability of two or more events happening in an interval of length s is much smaller than s.

Theorem 2.6. *$N(t) \sim Poi(\lambda t)$.*

Proof. Since $N(t)$ is the number of events that have occurred before time t, and the number of events in disjoint intervals is independent, we can split up the interval $[0, t]$ into intervals of size $\frac{1}{n}$ for some n in $\mathbb{N}$ and write

$$N(t) = \sum_{i=1}^{n} \left[N\left(\frac{it}{n}\right) - N\left(\frac{(i-1)t}{n}\right) \right]$$

If the probability that any one of those summands is greater than 2 is small, then $N(t)$ can be approximated by the binomial distribution. This is justified because $N(t)$ is the sum of the n random variables specified above, and under that assumption these would almost always only take on the values 0 or 1. To show that this approximation is valid, let $q = \mathbf{P}\{N\left(\frac{it}{n}\right) - N\left(\frac{(i-1)t}{n}\right) \geq 2 \text{ for some } i\}$. Then q represents the error between the actual probability distribution of $N(t)$ and the binomial distribution. First note that q is less than or equal to the sum of the probabilities that the ith interval is greater than 2

$$q \leq \sum_{i=1}^{n} \mathbf{P}\left\{N\left(\frac{it}{n}\right) - N\left(\frac{(i-1)t}{n}\right) \geq 2\right\}$$

But since all of these intervals are the same size, we have by stationary increments that

$$q \leq n\mathbf{P}\left\{N\left(\frac{t}{n}\right) - N(0) \geq 2\right\} = nP\left(\frac{t}{n}\right) \tag{2.7}$$

By the fourth condition on the Poisson process we know that $P\left(\frac{t}{n}\right)$ is much smaller than $\frac{t}{n}$, formally

$$\lim_{\frac{t}{n}\to 0}\left(\frac{P\left(\frac{t}{n}\right)}{\frac{t}{n}}\right) = 0$$

But $\frac{t}{n}$ going to 0 is the same as saying n goes to infinity since t is fixed, so

$$\lim_{n\to\infty}\left(\frac{P\left(\frac{t}{n}\right)}{\frac{t}{n}}\right) = 0,$$

which implies

$$\frac{1}{t}\lim_{n\to\infty}\left(nP\left(\frac{t}{n}\right)\right) = 0$$

So combining this with Equation 2.7 and the fact that $q \geq 0$ since it is a probability, we have

$$\lim_{n\to\infty} q = 0 \tag{2.8}$$

Which means that it is reasonable to approximate $N(t)$ by the binomial distribution since the error goes to 0 as n becomes large.

Now we need to determine the probability of success for the binomial distribution that will approximate the probability distribution of $N(t)$. The probability of success for a given trial is given by the stationary increments property, that is we have

$$\mathbf{P}\left\{N\left(t\frac{i}{n}\right) - N\left(\frac{i-1}{n}\right) = 1\right\} = \lambda\frac{t}{n}$$

so $\frac{\lambda t}{n}$ should be the probability of success. Then by the definition of the binomial distribution we have

$$\mathbf{P}\left\{N(t) = k\right\} \approx \binom{n}{k}\left(\frac{\lambda t}{n}\right)^k\left(1 - \frac{\lambda t}{n}\right)^{n-k}$$

But the error in this approximation is due to the probability that two or more events show up in a particular interval, and Equation 2.8 shows that the error goes to 0 as n goes to infinity, so these are in fact equal if we take that limit. Recalling that λ and t are constants we get

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

Evaluating the three parts of the limit gives

$$\begin{aligned}\lim_{n\to\infty}\binom{n}{k}n^{-k} &= \frac{1}{k!}\\ \lim_{n\to\infty}\left(1-\frac{\lambda t}{n}\right)^{n} &= e^{\lambda t}\\ \lim_{n\to\infty}\left(1-\frac{\lambda t}{n}\right)^{-k} &= 1\end{aligned}$$

So we get overall

$$\mathbf{P}\{N(t)=k\}=\frac{\left(\lambda t\right)^k e^{\lambda t}}{k!},$$

which is precisely Definition 2.3 substituting λt for λ. [2] □

To simulate a Poisson process, we use the following fact.

Theorem 2.9. *The waiting time between two events occurring in a Poisson process is an exponentially distributed random variable.*

Proof. Let T_i be the time at which the ith event occurs. Since the Poisson process has independent increments, these are all independent random variables. Now we will compute the cumulative distribution function (CDF) of T_1

$$F(t)=\mathbf{P}\{T_1\le t\}$$

which is the probability that at least one event occurs before time t, meaning that we want the probability that the number of occurrences before time t is more than 1. So

$$F(t)=\mathbf{P}\{N(t)\ge 1\}$$

But the complement of the set $\{N(t)\ge 1\ \}$ is the set $\{N(t)=0\}$, so

$$F(t)=1-\mathbf{P}\{N(t)=0\}$$

By Theorem 2.6 and Definition 2.1, the probability that $N(t)=0$ is given by

$$\mathbf{P}\{N(t)=0\}=\frac{e^{-\lambda t}(\lambda t)^0}{0!}=e^{-\lambda t}$$

And thus,

$$F(t)=1-e^{-\lambda t},$$

which is precisely the CDF of the exponential distribution. Since the waiting time T_k is independent of the waiting time T_{k-1} the Poisson process essentially restarts at time T_{k-1} and then the same argument gives that T_k is an exponentially distributed random variable. □

Using the above fact, we can easily simulate a Poisson process by generating a random variable $Y(i) \sim \text{Exp}(\lambda)$, $i = 1, 2, 3, ...$, and then saying that our Poisson process has occurrences at time $t = Y(0), Y(0) + Y(1), Y(0) + Y(1) + Y(2)$, etc. In other words, the value $Y(i)$ is the time between the occurence of events $i - 1$ and i.

3. Simulating random variables

Note that in the same way that many random events are discrete (such as the outcomes of a die) when creating pseudo-random numbers we generate them from a discrete set of numbers $\{0,1,...,M-1\}$. The sequence of pseudo-random numbers must satisfy some of the properties of truly random numbers, meaning that they must "appear" to be random. Some desirable characteristics for a sequence of pseudo-random numbers are:

- Since the sequence is deterministic, as soon as one number is repeated the entire sequence is repeated. Hence the sequence should repeat with maximal period, meaning that all the numbers from the set $\{0,1,...,M-1\}$ are used.
- If we plot the points $(x_i,\, x_{i+1})$ where x_i is the ith number in the sequence, these points should not lie on a few lines. This *good lattice property* should also hold for n-tuples $(x_i,...,x_{i+n})$ and hyperplanes in $[0,M-1]^n$.
- Though the sequence is deterministic, we do not wish it to appear so, implying there should be low predictability between numbers.

Examples 3.1. Bad sequences include:

- $\{1,2,3,4,5,6,...\}$ has a simple relation between the numbers.
- If $M = 10$ then $\{1,5,0,1,5,0...\}$ has a period of 3 instead of 10.
- Figure 1 shows a congruential generator with bad lattice properties. [3]

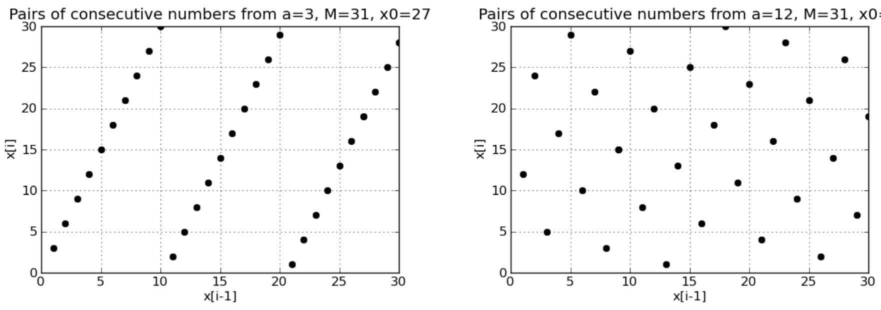

Figure 1. These plots show pairs of consecutive values generated with a multiplicative generator. Note that both choices are bad generators since in the left figure all consecutive pairs are on one of 3 lines; in the figure on the right all are on 6 lines.[3]

If we have such a sequence of numbers where each is in $\{0, 1, ..., M-1\}$, then we can create a pseudo-random sequence of numbers in [0,1] by taking $\{y_i = \frac{x_i}{M}\}$. Ideally this sequence looks like a Uniform(0,1) random variable. This can be tested with a χ^2-goodness-of-fit test by making k equally-sized bins in the interval [0,1] and determining the difference between the number of elements in each bin compared to the expected number in each bin. Note that if we take $k > M$ bins, then we have a bin size $\frac{1}{k} < \frac{1}{M}$, so the test will almost surely fail since there will be bins without any elements in them. For example the bin $[\frac{k-1}{k}$,1] will be empty since the largest value generated is $\frac{M-1}{M}$ which is less than $\frac{k-1}{k}$. Typically, however, M is substantially large (one common value is $2^{31} - 1$ and others are this order of magnitude or larger)[3], so this is not a concern.

One common method for generating the $\{x_i\}$ above is to use a *congruential generator*. Specifically, it is easiest to use a *multiplicative congruential generator* which has a seed x_0, a modulus M, and a multiplier a. Then we compute

$$x_i \equiv ax_{i-1} \text{ mod} M$$

For this type of generator, we can only obtain values in $\{1, ..., M-1\}$, since if x_j = 0, then for all $k > j$, we have $x_k = 0$ as well.
It is clear that not every choice of M, x_0, and a will satisfy all of the above properties, for instance choosing $M = 10, x_0 = 2, a = 2$ will not have maximal period as it will only generate the numbers {2,4,6,8}. Additionally the third sequence in Example 3.1 with bad lattice properties was generated with $M = 31, a = 3, x_0 = 27$. There are some values that do work such as $M = 2^{31} - 1$ and $a = 7^5$ [3].

Exponential Random Variables. At this point, we will assume we have a means to generate a pseudo-random variable U~Uniform(0,1) and will use this to generate an exponentially distributed random variable X. It is unimportant how U was obtained as long as it satisfies the properties discussed above, and the discussion of congruential generators was just to explain the possibility of creating such pseudo-random variables.
There are several common methods for generating random variables with different distributions from a Uniform(0,1) random variable. For exponential random variables, we only need one, the *method of inversion*. This method starts with a CDF F_X and computes the inverse of that function. Then, it takes a uniform random variable U and sets $X = F_X^{-1}(U)$. Note that a CDF for a random variable only needs to be a non-decreasing, right-continuous function that satisfies $\lim_{x\to-\infty} F_X(x) = 0$ and $\lim_{x\to-\infty} F_X(x) = 0$. In particular, it is not necessarily injective, so does not necessarily have an inverse. If F_X is constant on an interval $[a, b]$, then $\mathbf{P}\{X = y$ for $y \in (a, b]\} = 0$, so for the inverse we can define $F_X^{-1}(u) = a$ and leave the points in $(a, b]$ without an inverse. However, for simplicity the proof below we will assume that F_X is strictly increasing and thus has an inverse.

Theorem 3.2. *Let F_X be a strictly increasing CDF. If U~Uniform(0,1) and $X = F_X^{-1}(U)$ then X is a random variable with CDF F_X.*

Proof. By definition X has CDF

$$f(x) = \mathbf{P}\{X \leq x\}$$

But then we can plug in to obtain

$$\begin{aligned} f(x) &= \mathbf{P}\{F_X^{-1}(U) \leq x\} \\ &= \mathbf{P}\{U \leq F_X(x)\} \\ &= F_X(x) \end{aligned}$$

The third step is justified because U is a Uniform(0,1) random variable and hence has $\mathbf{P}\{U \leq y\} = y$ for all y in [0,1]. Therefore the CDF of X is F_X as claimed. □

If $X \sim \text{Exp}(\lambda)$, then the CDF for X is

$$F_X(x) \quad = \quad 1 - e^{-\lambda x}$$

This is strictly increasing on the interval $[0, \infty)$ so it has an inverse on that domain. By Theorem 3.2 if we can compute F_X^{-1} then we can generate an exponential random variable X from a uniform random variable U. So we have the following computation of F_X^{-1}.

$$\begin{aligned} u &= F_X^{-1}(x) & \iff \\ u &= 1 - e^{-\lambda x} & \iff \\ e^{-\lambda x} &= 1 - u & \iff \\ -\lambda x &= \log(1-u) & \iff \\ x &= \frac{-\log(1-u)}{\lambda} \end{aligned}$$

4. Data

The results from the previous sections yield the following algorithm for simulating a Poisson process with rate λ.

(1) Generate $u_i \sim$Uniform(0,1), $i = 1, 2, ...,$ using a congruential generator or through other means.
(2) Set $y_i = \frac{-\log(1-u_i)}{\lambda}$
(3) Set $x_i = \sum_{j=1}^{i} y_j$

Then x_i denotes the time at which $N(t)$ increases by 1 for the ith time.

Figure 2 and Figure 3 were generated using a simulation of a Poisson process written in Python. The simulation used precisely the algorithm stated above. The rate used for both figures is $\lambda = 0.5$. The first figure shows a sample run of a Poisson process with $N(t)$ on the vertical axis and time t on the horizontal axis. The simulation is from the range $0 \leq t \leq 25$. The second figure shows a histogram of the value of $N(25)$ for 2000 simulations of a Poisson process. Note that by Theorem 2.6, $N(25) \sim Poi(25\lambda) = Poi(12.5)$, so the histogram should approximate the mass function $Poi(12.5)$. The graph for that function is shown with a dotted line on the graph.

Acknowledgments. It is a pleasure to thank my mentors, Marcelo Alvisio and Shawn Drenning, for providing me with reading materials and ideas for my paper, discussing pseudo-random numbers and LaTeX, and spending time helping me develop and edit this paper.

References

[1] Ross, S. (2009). A First Course in Probability, Eigth Edition. Prentice Hall.
[2] Lawler, G. F. (2006). Introduction to Stochastic Processes, Second Edition. Chapman & Hall.
[3] Gramercy, R. B. (2008). Monte Carlo Inference. Retrieved May 6, 2008, from http://www.statslab.cam.ac.uk/ bobby/teaching.html
[4] Jordan, Johnathan (2010). Applications of Probability and Statistics. Retrieved July 22, 2010, from http://jonathanjordan.staff.shef.ac.uk/mas174/Epoissnotes.pdf

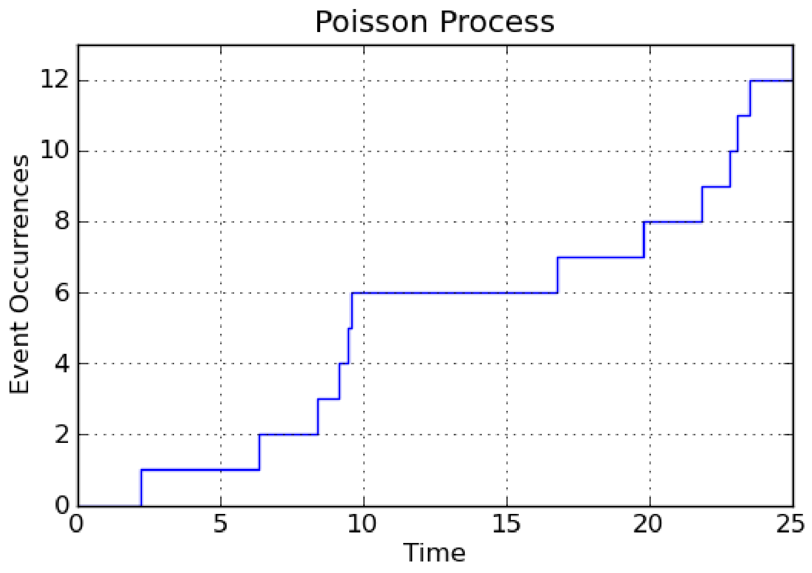

FIGURE 2. A Poisson process with $\lambda = .5$ for time $0 \leq t \leq 25$.

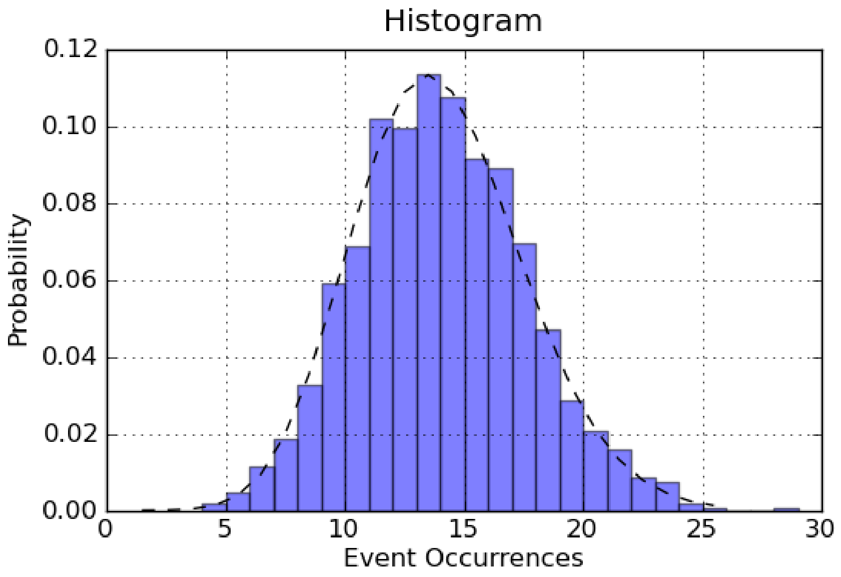

FIGURE 3. A histogram of the value of $N(25)$ for 2000 Poisson processes with $\lambda = .5$.