Table of Contents

ECE537: Lab 3 Report

*It is recommended to access this report by opening the `html` file on the browser or by clicking **here**.*

In the first part of the lab, we will be simulating a discrete random walk process and will analyze the stationarity of the process and the effects of bias in step directions. In the second part, we will simulate a Poisson point process and will compare the results (estimates) for the distribution parameters with the theoretical values.

Throughout this lab, the **Distributions.jl** package in Julia has been utilized to be able to use the probability constructs in code.

```
using Distributions , StatsBase , StatsPlots , LinearAlgebra , LaTeXStrings
    , PlutoUI
```

1. Simulating A Random Walk Process

A discrete random walk process, $Z(n)$ is given by,

$$Z(n) = \sum_{i=1}^{n} X_i \ ,$$

where X_i are i.i.d. random variables with pmf $p_X(1) = p$ and $p_X(-1) = 1 - p$. The pmf for X_i can either be expressed as the transformation $2B - 1$, where $B \sim \text{Bernoulli}(p)$, or as a location-shifted Categorical distribution with a prescribed pmf array, $\mathbf{p}$, such that $\mathbf{p}[k] = p_X(k - k_0)$. In code, we have used the latter.

Note that the mean of a discrete random walk process is given by,

$$\mu_Z(n) = n(2p - 1).$$

And the variance of a discrete random walk process is given by,

$$\sigma_Z^2(n) = 4np(1 - p).$$

1.1 Numerical Simulation

We will now create a simple interactive demo of a random walk process over n steps with the probability of progressing forward p.

n = 50

p = 0.5

We first begin by defining the pmf for the i.i.d. random process, and then creating a realization of the process by taking one instance (or sample) of each random step.

```
pmf = [1-p, 0, p]; # for k = 1, 2, 3. thus, need to shift location for k = -1, 0, 1
```

```
X (generic function with 1 method)
```

```
X(pmf, m) = [LocationScale(-2, 1, Categorical(pmf)) for k ∈ 1:m] # pₓ(1*k-2) = pmf[k]
```

```
Xᵢ = rand.(X(pmf, n));
```

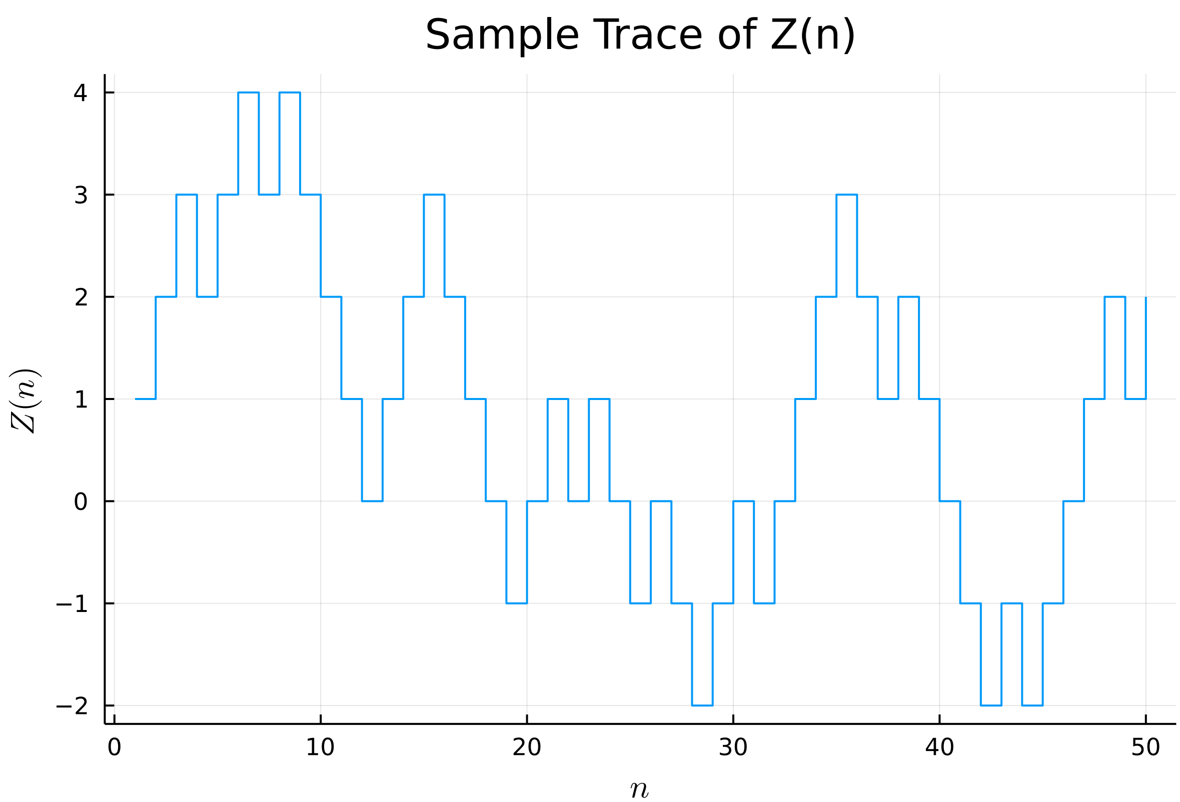
We then define the process $Z(n)$, evaluate it for the given realization, and then plot the sample path or trace.

```
Z (generic function with 1 method)
```

```
Z(Xᵢ, m) = sum(Xᵢ[1:m])
```

```
Zₙ =
 [1, 2, 3, 2, 3, 4, 3, 4, 3, 2, 1, 0, 1, 2, 3, 2, 1, 0, -1, 0,   more ,-1, -2, -1, -2, -1,
```

```
Zₙ = [Z(Xᵢ, k) for k ∈ 1:n]
```



We can observe, especially for a higher number of steps, that the traces resemble uniform discrete Brownian (or red) noise!

1.2 Summary of Results

Here we will compile the results over a fixed number of steps, $n = 500$, and estimate the mean and variance of the random process over $N = 50$ traces. We will then also visualize the effect of varying p .

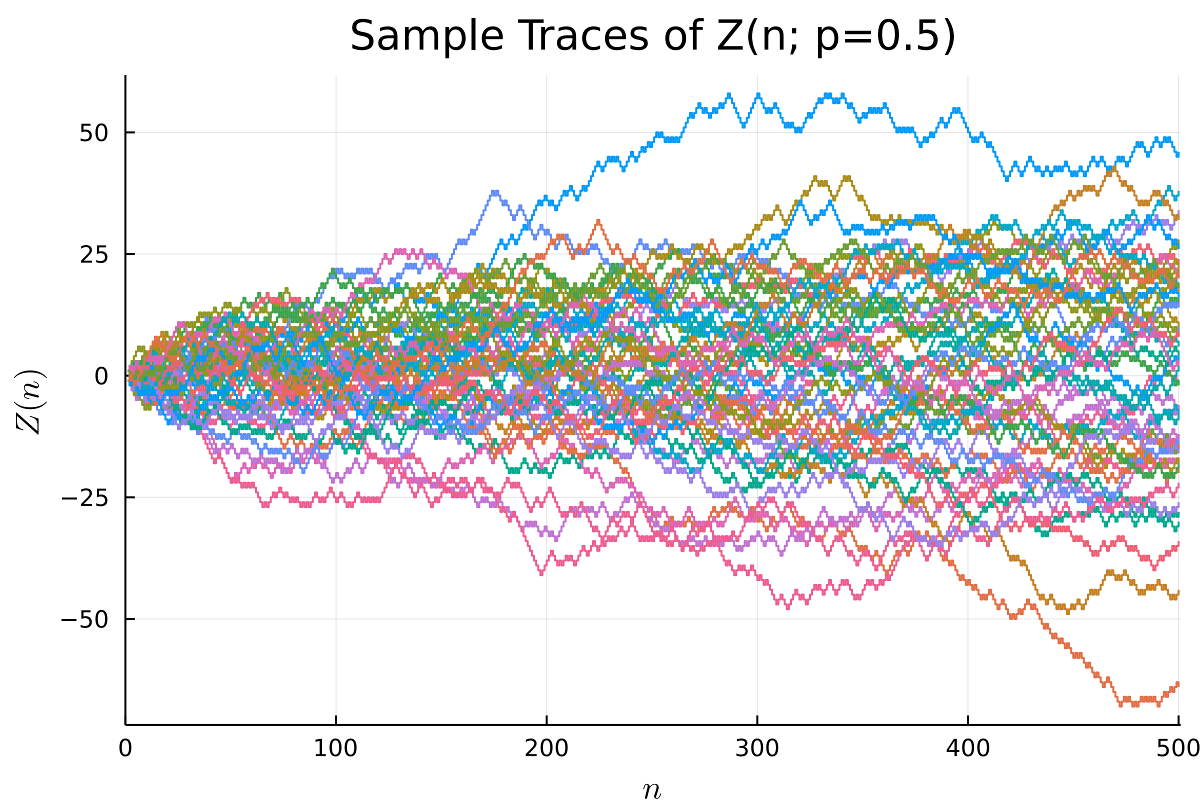
```
nfixed = 500;
```

```
N = 50;
```

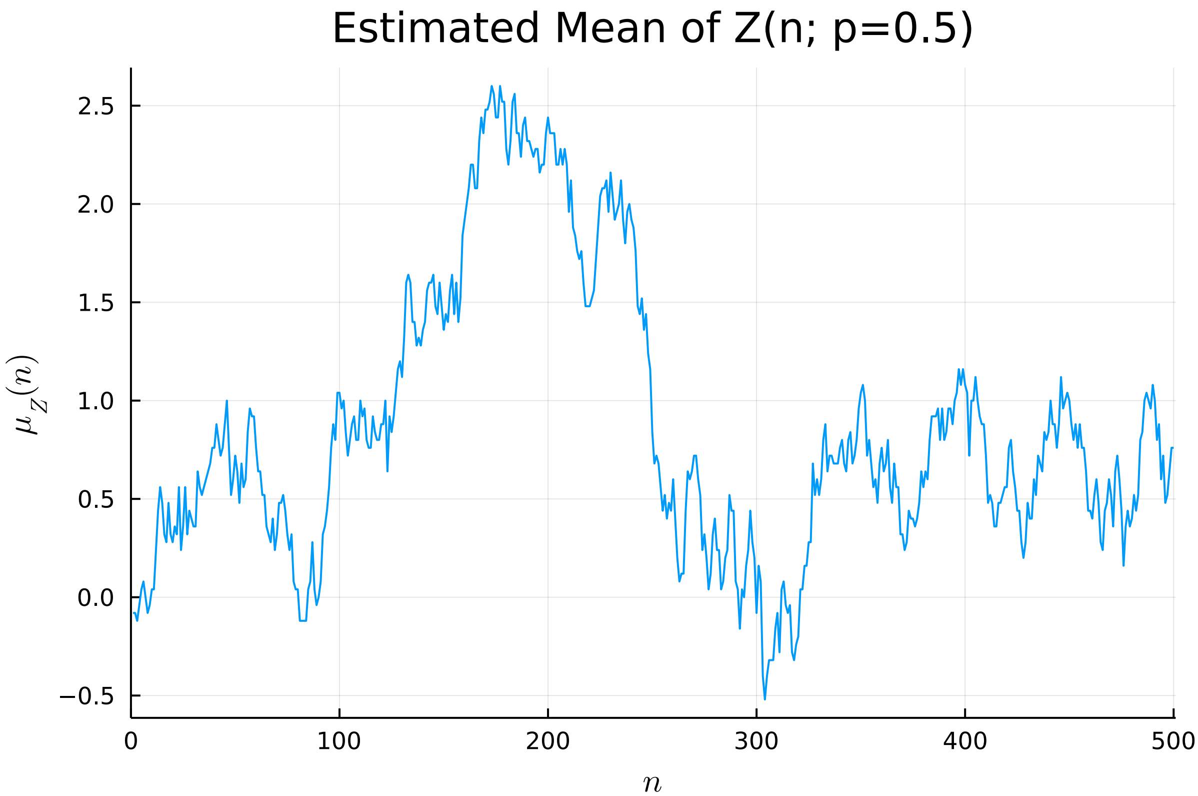
Below is a matrix where each column represents independent realizations (or traces) of the discrete random walk process, $Z(n)$, over the fixed number of steps. Notice in the plot below that the "spread", or the variance, of the process is getting larger as the number of steps increases.

```
500×50 Matrix{Int64}:
 -1   -1  -1   1  -1  -1   1  -1    1   1  …  1  -1  -1   -1   1  1  1   1  -1  -1
  0    0   0   2  -2  -2   0  -2    2   2     2  -2  -2   -2   0  2  2   0  -2   0
 -1    1   1   3  -1  -1  -1  -3    3   3     1  -3  -1   -3  -1  1  1  -1  -1  -1
 -2    0   2   2  -2   0   0  -2    2   4     2  -4   0   -2   0  2  2   0  -2  -2
 -1   -1   1   1  -1  -1   1  -3    3   5     1  -3  -1   -1   1  1  3   1  -3  -1
  0   -2   0   0   0   0   0  -2    2   6  …  0  -4   0   -2   0  2  2   0  -4   0
  1   -3   1  -1  -1   1   1  -3    3   5     1  -5  -1   -3   1  1  1  -1  -3   1
  ⋮                    ⋮                   ⋱  ⋮                   ⋮
 49  -65  15  -5  13   5   1  21  -29  25     3  11  19  -15  15  7  3  21  17  19
 48  -64  16  -4  12   6   0  20  -30  26  …  4  10  18  -16  14  6  2  20  18  18
 47  -65  15  -5  11   7  -1  19  -31  27     5   9  19  -15  15  7  1  19  17  19
 46  -64  16  -6  12   8   0  18  -32  26     6  10  20  -14  14  8  2  18  18  20
 45  -63  15  -7  11   7  -1  17  -31  27     5   9  19  -15  15  9  3  19  17  21
 46  -64  16  -6  10   6  -2  18  -30  28     4   8  18  -16  14  8  2  18  16  20
```

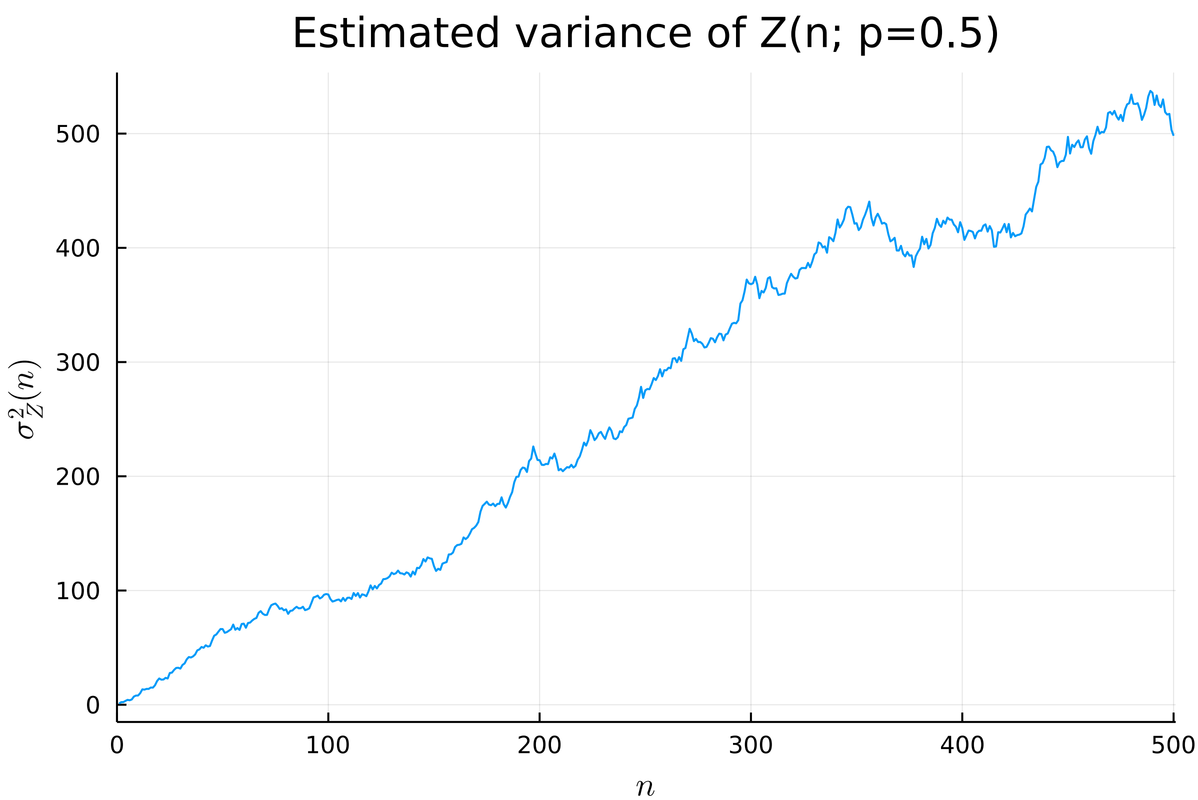
```julia
begin
    p₁ = 0.5;
    pmf₁ = [1-p₁, 0, p₁];
    𝑋ᵢ₁ = [rand.(𝑋(pmf₁, nfixed)) for k ∈ 1:N];
    𝑍ₙ₁ = [[𝑍(𝑋ᵢ₁[k], j) for j ∈ 1:nfixed] for k ∈ 1:N]; 𝑍ₙ₁ = hcat(𝑍ₙ₁...);
end
```



We can estaimte the expected value of each index $Z(k)$, which is a random variable, by averaging over all the traces. Similarly, we can also estimate the variance. Both have been plotted below.



Estimated Mean of Z(n; p=0.5)
$\mu_Z(n)$
n



Estimated variance of Z(n; p=0.5)
$\sigma_Z^2(n)$
n

Note from the plots above that, as anticipated for $p = 0.5$, $\mu_Z(n)$ remains close to 0. Especially for a higher number of traces, we would expect the trend to be further flattened to 0 as a better approximation. The variance, $\sigma_Z^2(n)$, also as anticipated for $p = 0.5$, almost follows a linear trend with $\sigma_Z^2(500) \approx 500$. This matches the observation from the traces above that the variance increases with the number of steps even when the process has zero mean, and it also matches our intuition; if we take an increasing number of such unbiased steps, we would expect ourselves to be within a ballpark of an increasing bound even though the mean is logically zero.

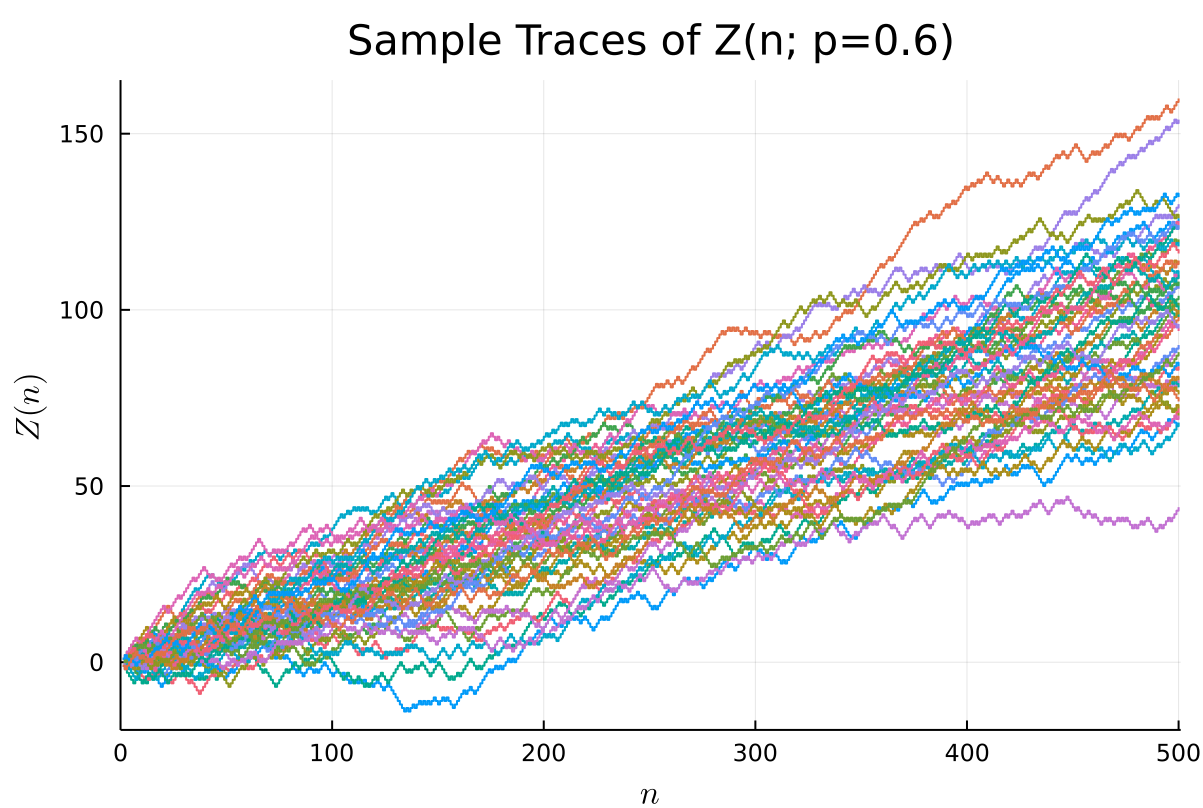
Furthermore, note that the discrete random process above is not strictly stationary since $\mathrm{VAR}[Z(n)] = \mathrm{VAR}[X_1 + \cdots + X_n] = n \neq \mathrm{VAR}[Z(n-k)]$ for $k \neq 0$, even though $E[Z(n)] = E[Z(n-k)] = 0$. So any two points can not have the same distribution in this process, meaning non-stationarity. Also, intuitively, the distribution changes over time because the variance increases, hence the random walk process should (at least in the strong sense) not be stationary.

We will now observe the effects of varying p to add bias to the general direction of movement of the random process.

We first, let $p = 0.6$ and plot the results of 50 traces below.

```
[0.4, 0.0, 0.6]
```

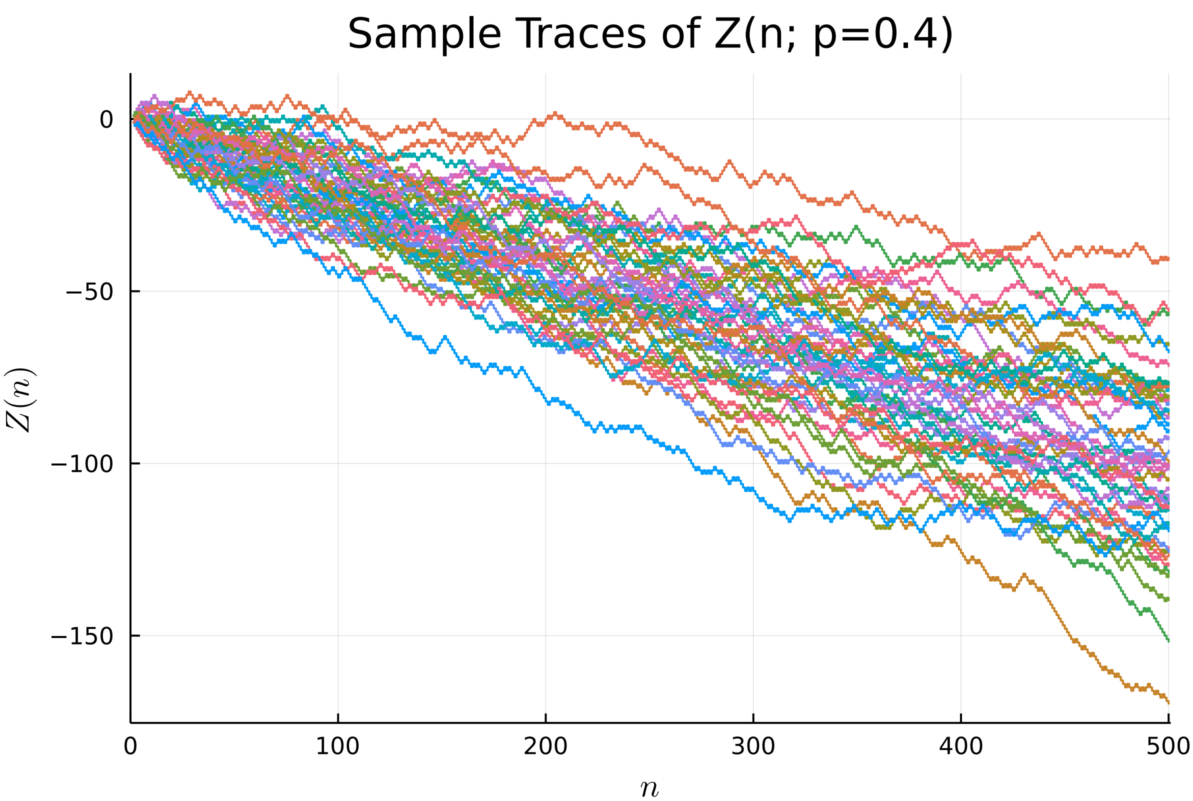
```
begin
    p₃ = 0.6;
    pmf₃ = [1-p₃, 0, p₃];
end
```



As expected, we see a linearly upwards trend with an increasing spread. However, we would expect this spread to decrease as steps become more deterministic in nature with increasing values of p, thereby resulting in almost straight line traces or paths with little deviation. The same goes for $p = 0.4$, where, instead, we see a linearly downwards trend because of being more likely to take a step backward instead of forward. The sample traces for this case are shown below.

```
[0.6, 0.0, 0.4]
```

```
begin
    p₂ = 0.4;
    pmf₂ = [1-p₂, 0, p₂];
end
```



Note that both cases above appear to follow the theoretical linear trend of $\mu_Z(n)$.

2. Simulating a Poisson Random Process

The Poisson random process is a continuous-time discrete-state process that can be interpreted as counting the number of events that have happened by time t, where successive events have exponential i.i.d. inter-occurrence times. Let this counting process be called $N(t)$, which can be succinctly expressed as,

$$N(t) = \sum_{i=1}^{\infty} I(X_1 + \cdots + X_i < t) \ ,$$

where, $I(P)$ is an identifier function defined to be,

$$I(P) = \begin{cases} 1 & \text{if predicate } P \text{ is true} \\ 0 & \text{if predicate } P \text{ is false} \end{cases}$$

Essentially, this sum calculates the number of events that have happened before time t by comparing it to the accumulation of inter-occurrence times.

Note that the resulting process will be Poisson distributed with parameter λt, i.e. $N(t) \sim \text{Pois}(\lambda t)$, where the i.i.d. inter-occurrence times $X_i \sim \text{Exp}(\lambda)$. We will also verify this numerically from the simulations.

2.1 Numerical Simulation

The infinite sum is impractical to carry out. Therefore, to simulate such a process in code, we can consider a "large enough" sequence of i.i.d. exponential random variables that we should expect to cumulatively amount to more than the maximum value of t in consideration. Here we have chosen the number of exponentials to be $N_{\text{exp}} = 100$, and for all trials run so far, this has been sufficient.

λ = 2.0

t = 5.0

Similar to section 1.1, we now sample a realization of the random process and plot the trace. Note that the vertical lines are the point of occurrence of one event which are the cause of the jump in the value of $N(t)$.

```
Nₑₓₚ = 100; # large enough to almost definitely span the range of t
```

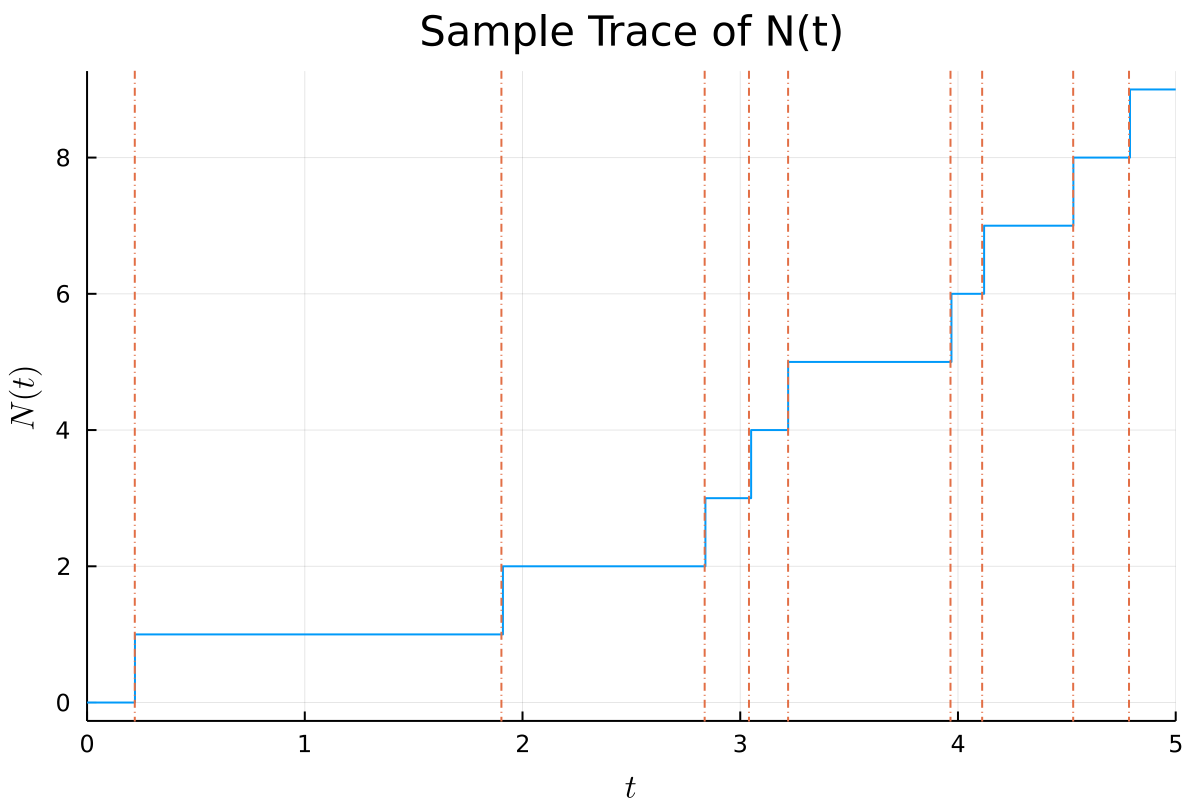
```
E(λ, n) = [Exponential(1/λ) for k ∈ 1:n];
```

```
𝑋ₑₓₚ =
 [0.219228, 1.68382, 0.933526, 0.203526, 0.179673, 0.745832, 0.145447, 0.418024, 0.2564
```

```
𝑋ₑₓₚ = rand.(𝐸(λ, Nₑₓₚ))
```

```
𝑁 (generic function with 1 method)
```

```
𝑁(𝑋ₑₓₚ, t) = (cumsum(𝑋ₑₓₚ) .< t) |> sum
```



2.2 Summary of Results

We begin by utilizing the code above to plot $N = 5$ independent traces of $N(t)$. We fix $\lambda = 2$ seconds and consider $t \in [0, 5]$ seconds.

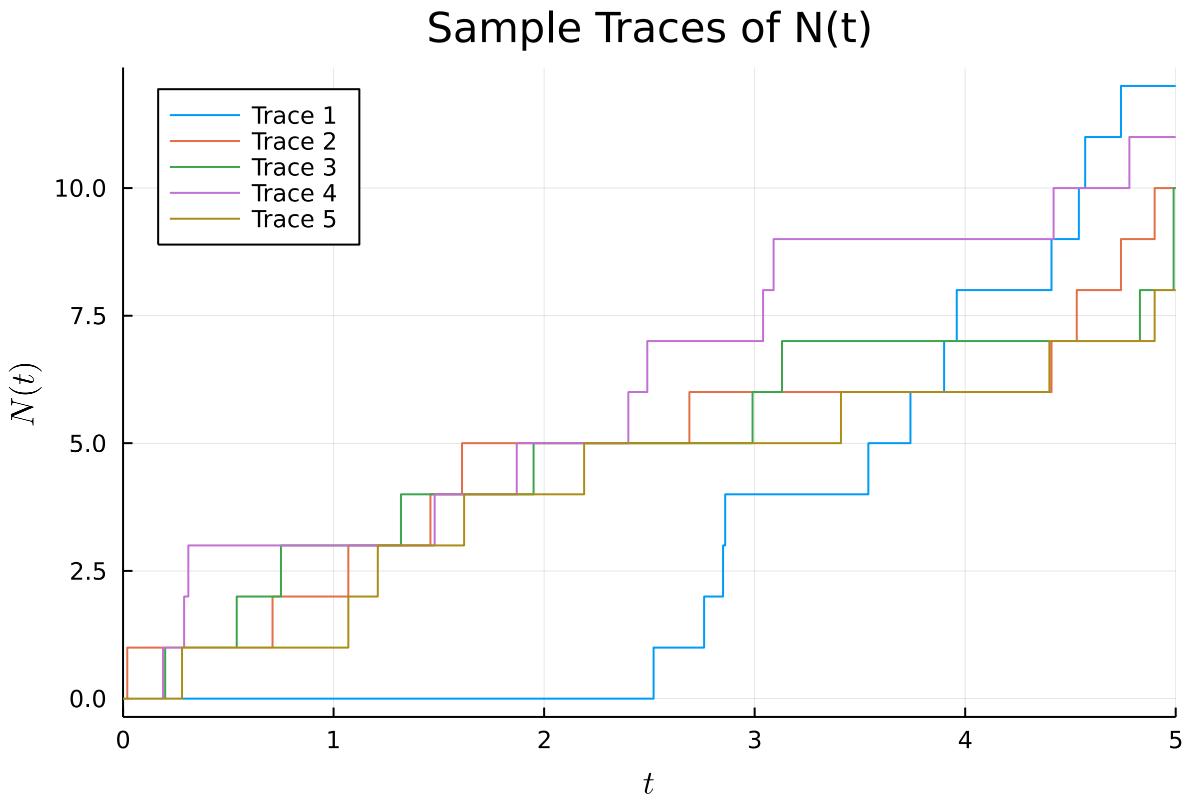
```
λfixed = 2;
```

```
tfixed = 5;
```

```
Ntraces = 5;
```

```
𝑁ₑₓₚᵢ (generic function with 1 method)
```

```
begin
    ts = 0:0.01:tfixed;
    𝑋ₑₓₚᵢ        = [rand.(𝐸(λfixed, Nₑₓₚ)) for k ∈ 1:Ntraces];
    𝑁ₑₓₚᵢ(ts, k) = [𝑁(𝑋ₑₓₚᵢ[k], t) for t ∈ ts];
end
```



We now verify if the simulations are accurate by comparing the numerical estimate of the statistics of the distribution to their theoretical values. We know that $N(t) \sim \text{Pois}(\lambda t)$, for which both $\mu_N(t) = \sigma_N^2(t) = \lambda t$. We continue with our choice of $t = 5$ and $\lambda = 2$ and sample the random variable, $N(5)$, 1000 times to compare with the theoretical mean and variance, either of which is a sufficient statistic for characterizing the underlying Possion distribution. Note that we should expect $N(t)$ to have the pmf,

$$p_N(k; \lambda, t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}$$

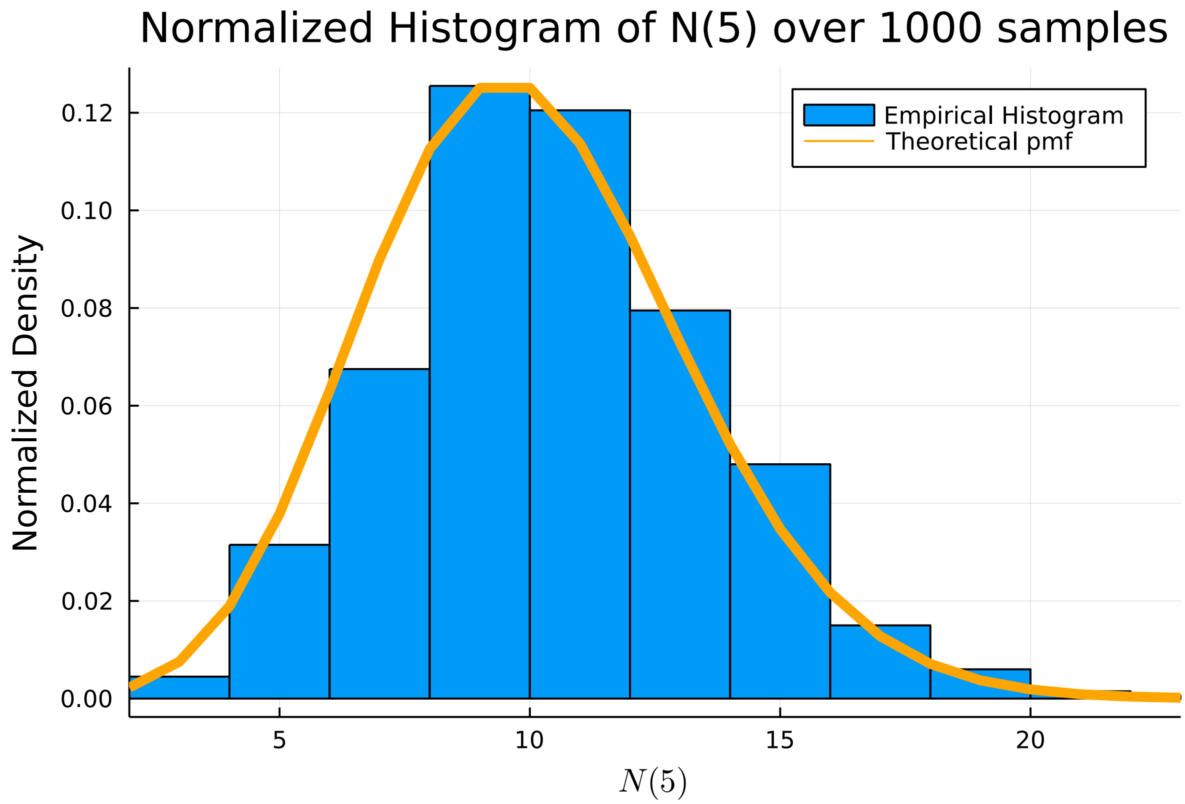
1000

```
Nsamples = 1000; # number of samples of the random variable N(t=tfixed; λ=λfixed)
```

𝑁samples =
 [15, 9, 10, 13, 16, 14, 13, 6, 6, 16, 6, 16, 6, 6, 10, 10, 15, 6, 9, 10,   more ,15, 9, 9

```
𝑁samples = [𝑁(rand.(𝐸(λfixed, Nₑₓₚ)), tfixed) for n ∈ 1:Nsamples]
```

```
𝑁hist = fit(Histogram, 𝑁samples, nbins=10);
```



10.036

```
mean(Nsamples)
```

10.138704000000011

```
var(Nsamples; corrected=false)
```

Note that both values are close to $\lambda t = 10$ which verifies that the simulation is accurate.