APPM 4560 Lab Two

Zoe Farmer

February 26, 2024

Note, all code/algorithms are in Appendix A.

1 Simulating a Homogeneous Poisson Process (HPP)

Consider the following algorithm to simulate the arrival times of a HPP with certain given intensity $\lambda > 0$ on the interval $[0, t]$.

Step 1	Set $i := 0$ and $T(0) := 0$.
Step 2	Generate $U \sim Unif(0, 1)$.
Step 3	Set $i := i + 1$ and $T(i) := T(i-1) - \ln(U)/\lambda$.
Step 4	If $T(i) > t$, set $N := (i-1)$ and stop. Otherwise, GOTO 2.

Table 1: Algorithm 1

1.1 Questions

1. What do the random variables $T(1), \ldots, T(N)$ generated by Algorithm 1 represent? Explain.

   These are each the points from the Poisson Process. If we imagine our algorithm as drawing points on a line of $[0, t]$ then each T value is the next point on the line.

Figure 1: Homogeneous Poisson Process

2. What's the distribution of the random number N? Explain.

   We can think of N being the number of points on the real line from $(0, t]$, therefore the distribution should be the Poisson distribution with parameter $\lambda \cdot t$. This follows as on the unit interval the amount of points will have Poisson distribution with parameter λ, so here we're simply scaling by our length, t.

3. What does the random quantity $T(N+1)$ represent? Explain.

   This quantity is the last generated value of our Homogeneous Poisson Process which (by definition) falls outside the interval $(0, t]$. To an extent, it represents the end of our Homogeneous Poisson Process.

4. What's the distribution of the random quantity $T(N+1) - t$? Explain.

   This should be the exponential distribution. Our Homogeneous Poisson Process has the property of being "memoryless", which means that after every point the probability of a new point follows the exponential distribution. Since the quantity $T(N+1)$ represents the final, non-included point, the location of this point should follow the same process that all other points follow.

5. Do the random variables $T(N+1) - T(N)$ and $T(N+1) - t$ have the same distribution? Explain.

   No, these should not follow the same distribution as they are inherently different quantities. The former is the difference between the final and the pre-final points, and the latter is how far from t the final point falls.

6. Determine the p.d.f. of $T(N+1)$. Include this calculation.

   Since we've already determined that, based on the memoryless property of the Homogeneous Poisson Process, the quantity $T(N+1) - t$ should have an exponential distribution with parameter λ, then the quantity $T(N+1)$ should simply be the shifted exponential distribution with parameter λ, which is

$$T(N+1) \sim \lambda e^{-\lambda(x-t)}$$

7. Implement Algorithm 1 with $\lambda = 3$ and $t = 4$ and obtain 10K simulations of the random vector $(N, T(N), T(N+1))$. Use the 10K draws to obtain the histograms associated with the quantities N, $T(N+1) - T(N)$, $T(N+1) - t$, and $T(N+1)$, respectively.

   See the next three questions for histograms.

8. Do the generated values of N support your answer to question 2? Comment on any expected/unexpected behavior.

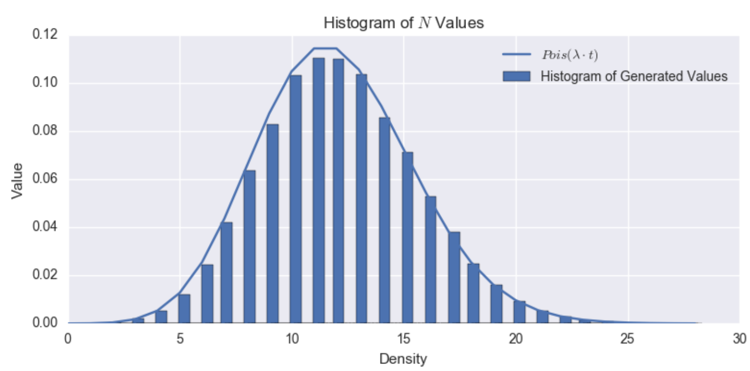

Figure 2: Histogram of N

   Yes they do, however we see some fluctuation, which is most likely a symptom of the simulation.

9. Do the generated values of $T(N+1) - T(N)$ and $T(N+1) - t$ support your answer to question 5? Comment.

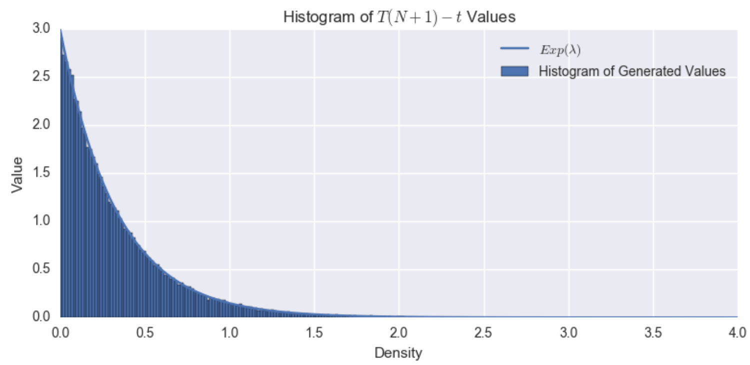

Figure 3: Histogram of $T(N+1)-t$

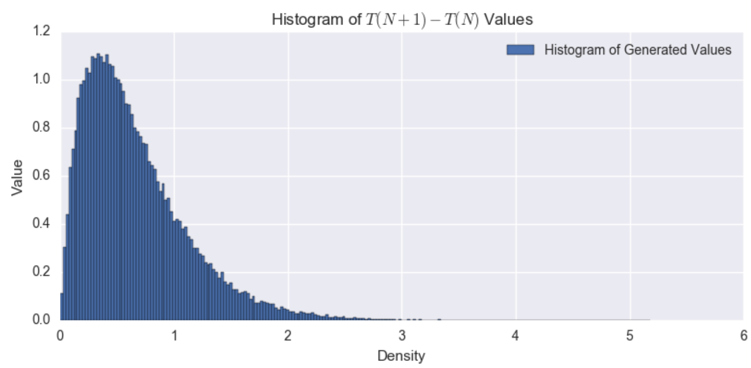

Figure 4: Histogram of $T(N+1)-T(N)$

As we can see from these above plots, these two quantities do not have the same distribution, which supports our previous prediction.

10. Do the generated values of $T(N+1)$ support question 6? Comment.

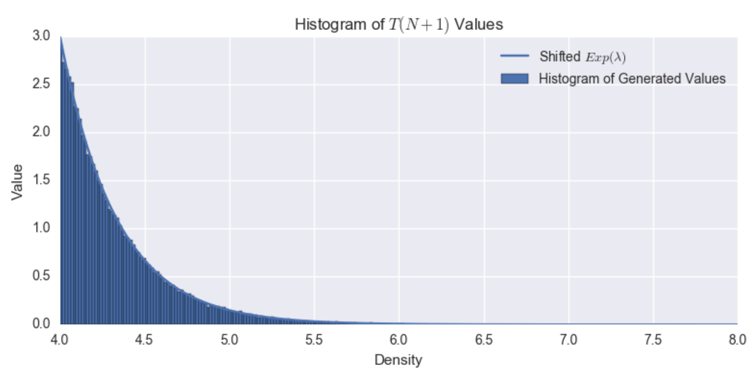

Figure 5: Histogram of $T(N+1)$

Like we surmised previously, this is simply the shifted exponential distribution!

2 Simulating a Non-Homogeneous Poisson Process (NHPP)

Let $T > 0$ be a given real number. We need to simulate a NHPP with intensity function $\lambda(t)$, for $0 \leq t \leq T$. This will require

- Precompute a constant $C > 0$ such that $0 \leq \lambda(t) \leq C$, $\forall t \in [0, T]$.
- Simulate the arrival times $T_1, T_2, \ldots$ of HPP with intensity C.
- Simulate a sequence $U_1, U_2, \ldots$ of i.i.d. Uniform$[0, 1]$ random variables that are independent of the arrival times $T_1, T_2, \ldots$.

The claim is that the process

$$N(t) := \# \left\{ i : T_i \leq t \text{ and } U_i \leq \frac{\lambda(T_i)}{C} \right\}, \text{ with } N(0) := 0$$

is a Poisson Process with intensity function $\lambda(t)$ over the interval $[0, T]$. In words, $N(t)$ is the number of pairs (T_i, U_i) which satisfy that $T_i \leq t$ and $U_i \leq \lambda(T_i)/C$. To simulate a Poisson process with intensity $\lambda(t)$ for $0 \leq t \leq T$, it therefore suffices to simulate $N(t)$. This is easy since we can simulate the arrival times T_i of a HPP with intensity C (using Algorithm 1) as well as a sequence of Uniform$[0, 1]$ random variables U_i.

Let $T = 9$ and $\lambda(t) := t^2 - 10t + 26$.

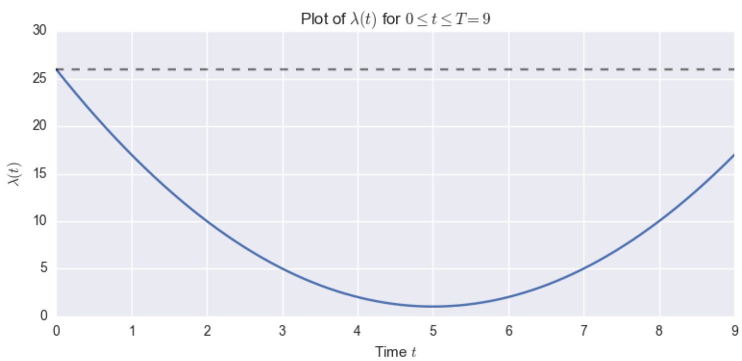

Figure 6: $\lambda(t) = t^2 - 10t + 26$

2.1 Questions

1. Let W be the random number of arrivals in the time interval $[0, T]$ of a NHPP with intensity function $\lambda(t)$. Use the above discussion to design a simple algorithm to simulate W.

   We can simulate the positions, and then simply obtain the number of values.

Step 1	Calculate $C = \max\{\lambda(t) : t \in [0, T]\}$
Step 2	Set $N_0 = 0$, $t = 0$, $i = 0$
Step 3	Set $u \sim Unif(0, 1)$
Step 4	Set $t = t - \ln(u)/C$, and $v \sim Unif(0, 1)$
Step 5	If $v \leq \lambda(t)/C$
	Set $N_i = t$, $i = i + 1$
Step 6	If $N_i > T$
	Set $W = i - 1$, and STOP
	Else
	GOTO Step 3

Table 2: Algorithm 2 - Simulation of W

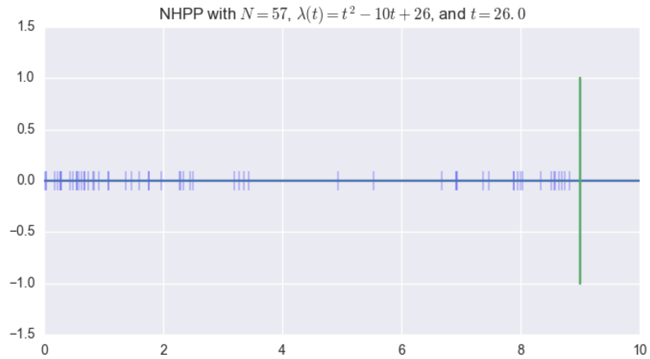

Figure 7: Non-Homogeneous Poisson Process

This looks like the following.

2. What is the theoretical distribution of W? What is $\mathbb{E}(W)$? Explain.

   Similar to the Homogeneous case, the distribution of W is Poisson-distributed with parameter (and expected value)

$$\Lambda(W) = \int_W \lambda(t)\, dt$$

   In this case this will be defined as

$$\Lambda(W) = 72$$

   We can think about this $\lambda(t)$ as defining the probability over the line from 0 to T, therefore the mean intensity can be found using the integral.

3. Implement the algorithm to simulate 10K independent draws of W. Does the histogram support your answer? Is the sample average of the simulated values comparable to the theoretical expected value of W? Comment.

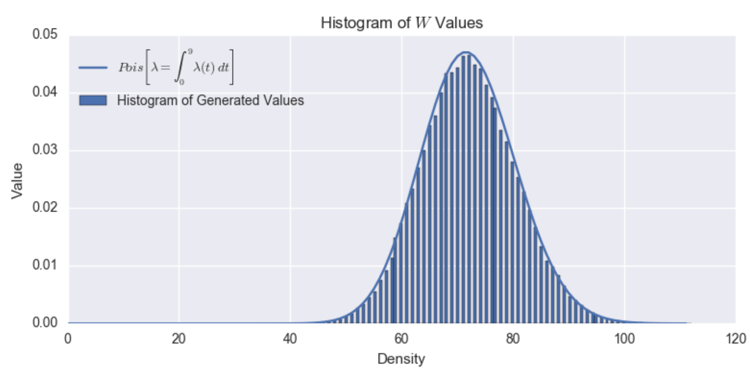

Figure 8: Histogram of W

This looks exactly like we imagined it would.

A Code

```
#!/usr/bin/env python3.5

import sys
import os
import math
import random
import numpy as np
import matplotlib.pyplot as plt
import seaborn
import sympy as sp
import scipy
import scipy.stats as sc_st

sp.init_printing()

FIGSIZE = (8, 4)

def main():
    algo1(3, 4)
    algo2(9, lambda t: t**2 - 10 * t + 26)

    l, t = 3, 4
    T, N = HPP(l, t)
    plt.figure(figsize=FIGSIZE)
    plt.plot(np.arange(t + 2), np.zeros(t + 2))
    plt.plot(t * np.ones(3), np.arange(-1, 2))
    plt.plot(T, np.zeros(len(T)), 'k*', markersize=20)
    plt.title(r'HPP with $N={}$, $\lambda={}$, and $t={}$'.format(N, l, t))
    plt.savefig('HPP')


def HPP(l, t):
    T = [0]
    i = 0
    while True:
        u = random.random()
        i += 1
        T.append(T[i - 1] - math.log(u) / l)
        if T[i] > t:
            N = i - 1
            break
    return T, N


def algo1(l, t):
    num_generate = 100000

    vals = np.zeros(shape=(num_generate, 3))

    for j in range(num_generate):
        TN = 0
        i  = 0
```

```
        while True:
            TN1 = TN - (math.log(random.random()) / l)
            i += 1
            if TN1 > t:
                N = i - 1
                break
            else:
                TN = TN1
        vals[j] = [N, TN, TN1]


    plt.figure(figsize=FIGSIZE)
    hist, edges = np.histogram(vals[:, 0], bins='auto', density=True)
    plt.bar(edges[:-1], hist / 6, width=0.5, label='Histogram of Generated Values')
    x = np.arange(int(edges[-1] + 1))
    rv = sc_st.poisson(l * t)
    plt.plot(x, rv.pmf(x), label=r'$Pois(\lambda \cdot t)$')
    plt.xlabel('Density')
    plt.ylabel('Value')
    plt.title(r'Histogram of $N$ Values')
    plt.legend(loc=0)
    plt.tight_layout()
    plt.savefig('part1_N_hist.png')

    plt.figure(figsize=FIGSIZE)
    hist, edges = np.histogram(vals[:, 2] - vals[:, 1], bins='auto', density=True)
    plt.bar(edges[:-1], hist, width=edges[1] - edges[0], label='Histogram of Generated Values')
    plt.xlabel('Density')
    plt.ylabel('Value')
    plt.title(r'Histogram of $T(N+1) - T(N)$ Values')
    plt.tight_layout()
    plt.legend(loc=0)
    plt.savefig('part1_TN1_TN_hist.png')

    plt.figure(figsize=FIGSIZE)
    hist, edges = np.histogram(vals[:, 2] - t, bins='auto', density=True)
    plt.bar(edges[:-1], hist, width=edges[1] - edges[0], label='Histogram of Generated Values')
    x = np.linspace(0, int(edges[-1] + 1), 1000)
    rv = lambda x: l * np.exp(-l * x)
    plt.plot(x, rv(x), label=r'$Exp(\lambda)$')
    plt.xlabel('Density')
    plt.ylabel('Value')
    plt.title(r'Histogram of $T(N+1) - t$ Values')
    plt.tight_layout()
    plt.legend(loc=0)
    plt.savefig('part1_TN1_t_hist.png')

    plt.figure(figsize=FIGSIZE)
    hist, edges = np.histogram(vals[:, 2], bins='auto', density=True)
    plt.bar(edges[:-1], hist, width=edges[1] - edges[0], label='Histogram of Generated Values')
    x = np.linspace(t, int(edges[-1] + 1), 1000)
    rv = lambda x: l * np.exp(-l * (x - t))
    plt.plot(x, rv(x), label=r'Shifted $Exp(\lambda)$')
    plt.xlabel('Density')
```

```
    plt.ylabel('Value')
    plt.title(r'Histogram of $T(N+1)$ Values')
    plt.tight_layout()
    plt.legend(loc=0)
    plt.savefig('part1_TN1_hist.png')


def algo2(T, l_func):
    x = np.linspace(0, T, 1000)
    plt.figure(figsize=FIGSIZE)
    plt.plot(x, l_func(x))
    plt.plot(x, max(l_func(x)) * np.ones(len(x)), 'k--', alpha=0.5)
    plt.xlabel(r'Time $t$')
    plt.ylabel(r'$\lambda(t)$')
    plt.title(r'Plot of $\lambda(t)$ for $0 \leq t \leq T=9$')
    plt.tight_layout()
    plt.savefig('part2_lambda.png')

    C = max(l_func(x))
    Tvals, N = HPP(C, T)
    Tvals = np.array(Tvals)
    plt.figure(figsize=FIGSIZE)
    plt.plot(np.arange(T + 2), np.zeros(T + 2))
    plt.plot(T * np.ones(3), np.arange(-1, 2))
    plt.scatter(Tvals, np.zeros(len(Tvals)), marker='|', s=200)
    plt.title(r'HPP with $N={}$, $\lambda={}$, and $t={}$'.format(N, C, T))
    plt.xlim(0, T+1)
    plt.savefig('HPPC')


    N, count = NHPP(l_func, T, C)
    plt.figure(figsize=FIGSIZE)
    plt.plot(np.arange(T + 2), np.zeros(T + 2))
    plt.plot(T * np.ones(3), np.arange(-1, 2))
    plt.scatter(N, np.zeros(len(N)), marker='|', s=200)
    plt.title(r'NHPP with $N={}$, $\lambda(t)=t^2-10t+26$, and $t={}$'.format(count, C, T))
    plt.xlim(0, T+1)
    plt.savefig('NHPP')


    num = int(1e5)
    W = np.zeros(num)
    for i in range(num):
        _, w = NHPP(l_func, T, C)
        W[i] = w

    plt.figure(figsize=FIGSIZE)
    hist, edges = np.histogram(W, bins='auto', density=True)
    plt.bar(edges[:-1], hist / 2, width=edges[1] - edges[0], label='Histogram of Generated Values')
    rv = sc_st.poisson(72)
    x = np.arange(int(edges[-1]))
    plt.plot(x, rv.pmf(x), label=r'$Pois\left[ \lambda=\int_0^9 \lambda(t) \, dt \right]$')
    plt.xlabel('Density')
    plt.ylabel('Value')
```

```
    plt.title(r'Histogram of $W$ Values')
    plt.tight_layout()
    plt.legend(loc=0)
    plt.savefig('part2_W.png')


def NHPP(l_func, T, C):
    N = [0]
    i = 0
    t = 0
    while True:
        t = t - math.log(random.random()) / C
        if random.random() <= l_func(t) / C:
            N.append(t)
            i += 1
            if N[i] > T:
                count = i - 1
                break
    return N, count



if __name__ == '__main__':
    sys.exit(main())
```