Ethan Glaser

Modeling HW2

1a Expected obtained = 20 − (expected not obtained)

$20 - E[X] = 20\left(\left(\frac{19}{20}\right)^{60} \cdot 20\right) = 20 - 0.9214 = \boxed{19.0786}$

b expected number of times to get 1 unique →

expected number to get 2 unique 2 draws: $\frac{19}{20}$ 3 draws: $\frac{1}{20} \cdot \frac{19}{20}$ n draws: $\frac{19}{20}\left(\frac{1}{20}\right)^{n-2}$

expected number of additional to get in draws: immediate: $\frac{20-n+1}{20}$

$E[X] = \sum_{K=1}^{\infty} \frac{20-X}{20} \frac{X}{20}^{K-1}$ → $\sum_{X=0}^{19} \sum_{K=1}^{\infty} \frac{20-x}{20}\left(\frac{x}{20}\right)^{K-1}$ → $\sum_{x=0}^{19} \frac{20}{20-x}$

(↳ go from x to x+1)

$p = \frac{20-x}{20}$ → $E[X] = \sum_{x=0}^{19} \frac{20}{20-x}$

$Var(x) = \sum_{x=0}^{19} \frac{\frac{x}{20}}{\left(\frac{20-x}{20}\right)^2} = \sum_{x=0}^{19} \frac{20x}{(20-x)^2}$

3. $f_{x+y}(t) = \int_0^t f_{x,y}(x, t-x)\,dx = \int_0^t f_x(x) f_y(t-x)\,dx$

$f_{y|x}(y|x) = \frac{f_{x,y}(x,y)}{f_x(x)} \rightarrow \frac{P(X=x, Y=t-x)}{P(X+Y=t)}$ → $\frac{f_x(x) f_y(t-x)}{\int_0^t f_x(x) f_y(t-x)\,dx}$

$f_{x,y}(x,y) = f_{x,y}(x, t-x) = f_x(x) f_y(t-x)$

← using λ for x and y

$\frac{\lambda e^{-\lambda t}}{\lambda \int_0^t e^{-\lambda t}\,dx}$ ← $\frac{\lambda e^{-\lambda x} \lambda e^{-\lambda(t-x)}}{\int_0^t \lambda e^{-\lambda x} \lambda e^{-\lambda(t-x)}\,dx}$

$\frac{e^{-\lambda t}}{\left[e^{-\lambda t} x\right]_0^t} = \frac{e^{-\lambda t}}{e^{-\lambda t} t - e^{-\lambda t} 0} = \frac{e^{-\lambda t}}{e^{-\lambda t} t} = \frac{1}{t} = \int_0^t \frac{1}{t}\,dx$

uniform distribution from 0 to t

4.a $X = X_1 + X_2 \quad \lambda = \lambda_1 + \lambda_2$

$P_X(X) = P(X = X) = \sum_{j=0}^{X} P(X_1 = j, X_2 = X - j) \rightarrow X_1 + X_2 = X \rightarrow$

$\frac{e^{-\lambda}\lambda^X}{X!} \leftarrow \frac{e^{-\lambda}}{z!}(\lambda_1+\lambda_2)^z \leftarrow \frac{e^{-\lambda}}{z!}\sum_{j=0}^{z}\binom{z}{j}\lambda_1^j\lambda_2^{z-j} \leftarrow \sum_{j=0}^{z}\binom{z}{j}\frac{e^{-\lambda_1}\lambda_1^j e^{-\lambda_2}\lambda_2^{z-j}}{z!} \leftarrow \sum_{j=0}^{z}\frac{e^{-\lambda_1}\lambda_1^j}{j!}\frac{e^{-\lambda_2}\lambda_2^{z-j}}{(z-j)!}$

↳ poisson distribution, Mean and Variance are $\lambda = \lambda_1 + \lambda_2$

b. $P[X_1 = i \mid Y = n] \rightarrow P[X = x \mid Y = y] = \frac{P(X = x, Y = y)}{P(Y = y)}$ | X, Y - total customers

$\frac{\lambda_1}{\lambda+\lambda_2} \cdot p \rightarrow \binom{n}{i}\left(\frac{\lambda_1}{\lambda_1+\lambda_2}\right)^i\left(\frac{\lambda_2}{\lambda_1+\lambda_2}\right)^{n-i}$

$P(X=x) = \sum P(X=x, Y=y_i) = P(X=x \mid Y=y_i) \cdot P(Y=y_i) \rightarrow P(X=x) = \sum P(X=x \mid Y=y_i) \cdot P(Y=y_i)$

$P(X=i) = \sum P[X=i \mid Y=n] \cdot P(Y=n) \rightarrow \sum_n \frac{e^{-(\lambda_1+\lambda_2)}(\lambda_1+\lambda_2)^n}{n!} \cdot \frac{n!}{(n-i)!\,i!} \cdot \left(\frac{\lambda_1}{\lambda_1+\lambda_2}\right)^i \cdot \left(\frac{\lambda_2}{\lambda_1+\lambda_2}\right)^{n-i}$

Stage 1: poisson(λ_1)
Stage 2: poisson(λ_2)

$\leftarrow \frac{e^{-\lambda_1}\lambda_1^i}{i!}\sum_n \frac{e^{-\lambda_2}\lambda_2^{n-i}}{(n-i)!} \leftarrow \frac{e^{-(\lambda_1+\lambda_2)}\lambda_1^i}{i!}\sum_n \frac{\lambda_2^{n-i}}{(n-i)!}$

Sum of Poisson → 1

c. $0.8 = \frac{Cov(X_1, X_2)}{\sqrt{Var(X_1)Var(X_2)}} \rightarrow Cov(X_1, X_2) = 0.8\sqrt{Var(X_1)Var(X_2)}$

$Var(X_1+X_2) = Var(X_1) + Var(X_2) + 2\left[0.8\sqrt{Var(X_1)Var(X_2)}\right]$

↳ $\lambda_1 + \lambda_2 + 1.6\lambda_1\lambda_2$ → equal to mean since poisson

5.b income → poisson($\log(1+t)$) · ρ − Ct → Mean trips is $\log(1+t)$

$\frac{d}{dt}(\rho\log(1+t) - Ct) = 0 \leftarrow \rho\log(1+t) - Ct$

$\rho \cdot \frac{1}{1+t} - C = 0 \rightarrow \rho \cdot \frac{1}{1+t} = C$

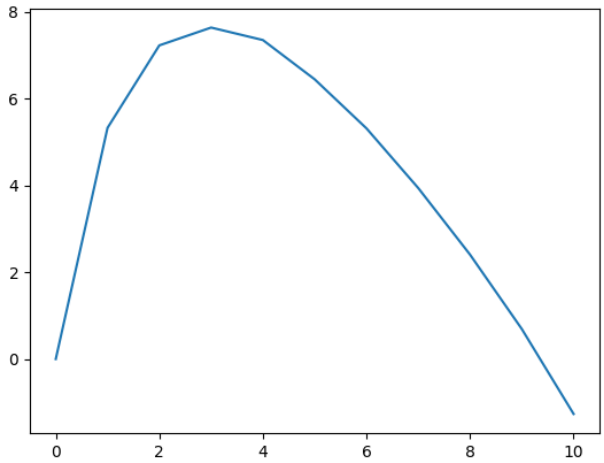
$\leftarrow \frac{1}{1+t} = 0.25 \leftarrow 12 \cdot \frac{1}{1+t} = 3$

$1 = 0.25 + 0.25t \rightarrow 0.75 = 0.25t \rightarrow t = \frac{0.75}{0.25} = 3$

```python
import numpy as np
from matplotlib import pyplot as plt

def experiment(target):
    total = 0
    tries = 0
    while total < target:
        total += np.random.uniform()
        tries += 1
    return tries

print("2a.")
for target in [1, 2]:
    for experiments in [100, 1000, 10000]:
        print("Estimated E[N" + str(target) + "] from generating " + str(experiments) + " values of N"
        + str(target) + ": " + str(sum([experiment(target) for i in range(experiments)]) / experiments))



print("5.")
def maximize(p, c, t, num_tests):
    income = p * np.random.poisson(np.log(1 + t), num_tests) - t * c
    return np.average(income)

x, y = [], []
for t in range(11):
    x.append(t)
    y.append(maximize(12, 3, t, 100000))

plt.plot(x, y)
plt.savefig('hw2_5')

print("Optimal number of hours worked is " + str(y.index(max(y))))
```

```
2a.
Estimated E[N1] from generating 100 values of N1: 2.69
Estimated E[N1] from generating 1000 values of N1: 2.744
Estimated E[N1] from generating 10000 values of N1: 2.7173
Estimated E[N2] from generating 100 values of N2: 4.63
Estimated E[N2] from generating 1000 values of N2: 4.665
Estimated E[N2] from generating 10000 values of N2: 4.6673
5.
Optimal number of hours worked is 3
```

Income based on hours worked



Hours worked