Homework 9: Probability and Simulation

November 29, 2021

1

1.a

If I'm only allowed to play the game once, I wouldn't pay more than the minimum amount, which is 1 ducat. The probability of losing money when only playing the game once is too high.

1.b

First, the assumption that player B has an infinite amount of money is made, which is simply unrealistic. Second, it is assumed that player A has an infinite amount of time, the game have to be played within player A's lifetime for it to be a valuable choice.

1.c

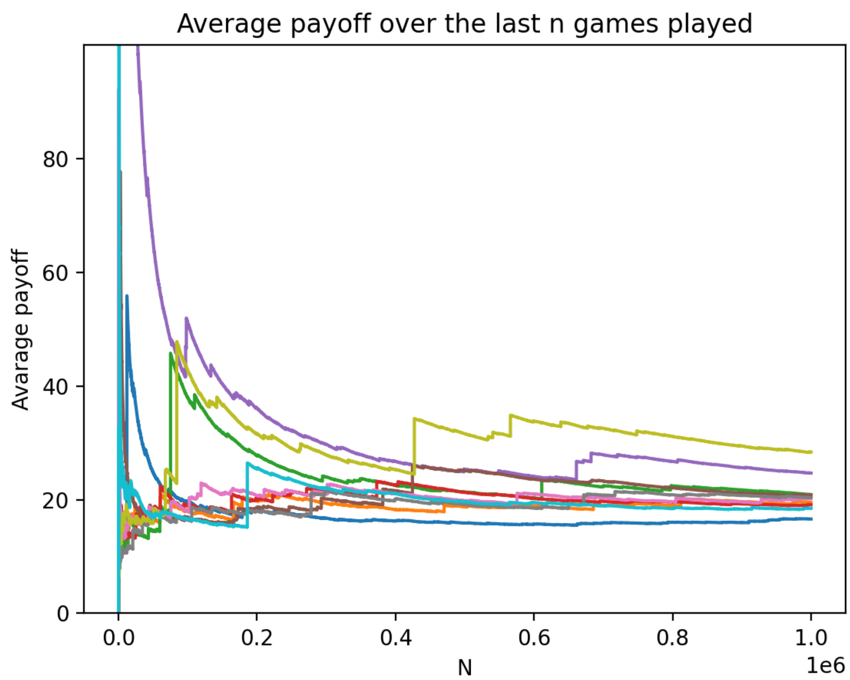

Figure 1: 10 simulations of the St Petersburg paradox with num

1.d

Considering that player B has unlimited resources, the expected value will be infinite. This is probably due to the fact that the average will continually be raised by large payouts when player A suddenly wins big. However, as the average increases, it requires larger and larger wins to raise the average. Thus big increases of average will become more seldom when N grows, which explains why the simulations converges. The expected value does however increase if N increases, even if it does so rather slowly, and this is why the expected value will approach infinity when resources is unlimited (number of games N is thus unlimited).

2

The simulation is done as a Poisson process, with an average of visitors per day being 300, and the simulation is done over 100 000 days. We can see at the right tail of Figure 2 that in 100 000 days, there is very low probability that we will have more than 360 visitors. Thus, this can't merely be explained by fluctuations, but would more reasonably be explained by some accident or other event.

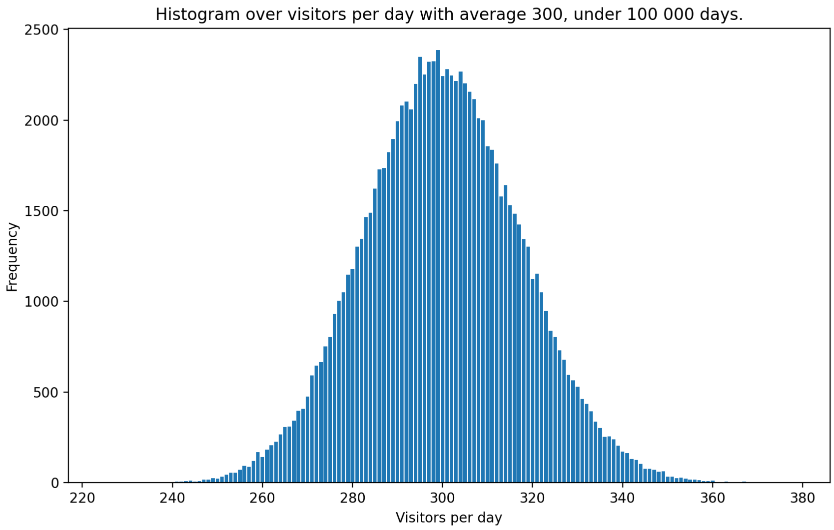

Figure 2: 10 simulations of the St Petersburg paradox with num

3 Appendix A

```
import matplotlib.pyplot as plt
from random import sample
from statistics import mean

class Simulation():

    def play_game(self, turn, win):
        flip = sample([0,1], 1)[0]
        if(flip):
            win = win + 2**turn
            return self.play_game(turn+1, win)
        else:
            return win
    # avarage payoff from last n games
    def simulate(self, number_of_games):
        average = [0]*number_of_games
        for i in range(1, number_of_games):
            res = self.play_game(1, 1)
            # use previous average to calculate running average
            average[i] = average[i-1] + (res-average[i-1])/i
        return average

if __name__ == "__main__":
    Simulation = Simulation()
    number_of_games = 1000000
    X = range(number_of_games)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.plot(X, Y)
    Y = Simulation.simulate(number_of_games)
    plt.xlabel('N')
    plt.yticks(range(0, 100, 20))
    plt.ylim(0, 100)
    plt.ylabel('Avarage payoff')
    plt.title('Average payoff over the last n games played')
    plt.plot(X, Y)
    plt.show()
```

4 Appendix B

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import poisson

if __name__ == "__main__":
    # get poisson process with average visitors 300 over 100 000 days
    X = pd.Series(poisson.rvs(300, size=100000))
    # count the values, how many times did we have 300 visitors and so on...
    data = X.value_counts().sort_index()
    dict = data.to_dict()
    fig, ax = plt.subplots(figsize=(10, 6))
    plt.xlabel('Visitors per day')
    plt.ylabel('Frequency')
    plt.title('Histogram over visitors per day with average 300, under 100 000 days.')
    ax.bar(list(dict.keys()), list(dict.values()), align='center')
    plt.show()
```