Title

Probability and Statistics – 2023 Summer Training Tasks Report

By

Abhash Rai

Contents

Part 1: Coin Flipping Simulation and Analysis

Introduction

This coin flipping simulation and probability analysis aims to explore the fundamental probability concepts through practical experimentation. The primary goal is to design a Python program that simulates multiple fair coin flips. Subsequently, experimental probabilities for events such as obtaining heads (P(Heads)), tails (P(Tails)), and specific coin flip sequences (e.g., HTH) will be calculated. Additionally, theoretical probabilities for the same events will be derived based on the foundational principle of a fair coin. Through a comparative analysis of experimental and theoretical probabilities, valuable insights into the practical application of probability theory in real-world scenarios will be obtained.

The code and classes used in this task have been entirely developed independently, constructing a comprehensive coin flipping simulation. In this experiment, coin flips were conducted across different trials, with varying numbers of flips per trial, including 10, 50, 100, 500, 1,000, 10,000, 100,000, and 1,000,000 flips or trials. The experiment consisted of two main parts:

1. Probability of Heads and Tails Visualization: In the first part, focus is on visualizing the probability of getting heads and tails for each trial with a specific number of coin flips.
2. Probability of a Specific Sequence (Head-Tail-Head) Visualization: In the second part, the probability of observing a specific sequence of coin flips were explored, namely "head-tail-head," for each trial with 3 flips.

Theoretical Probabilities

In the case of a fair coin, there are two possible outcomes for each flip: Heads and Tails. Assuming a fair coin, the probability of getting a "Head" (H) or "Tail" (T) on a single flip is 0.5 (50%) each, because there are two equally likely outcomes

- P(Heads) (Probability of getting Heads) = ½ = 0.5
- P(Tails) (Probability of getting Tails) = ½ = 0.5

To find the probability of sequence "HTH" (Head-Tail-Head), we can use the concept of probability. The probability of a sequence of events is calculated by multiplying the individual probabilities of each event in the sequence. So, for "HTH" to occur in a sequence of three flips:

- The probability of getting a "Head" on the first flip (H) is 0.5 (50%).
- The probability of getting a "Tail" on the second flip (T) is also 0.5 (50%).
- The probability of getting a "Head" on the third flip (H) is again 0.5 (50%).

Probability of the entire sequence "HTH" occurring: 0.5 (H) * 0.5 (T) * 0.5 (H) = 0.125 or 12.5%

Experimental Probabilities

The probability of obtaining heads and tails in a series of coin flips was investigated for various numbers of flips. To explore this, a coin flipping for various number of flips was simulated.

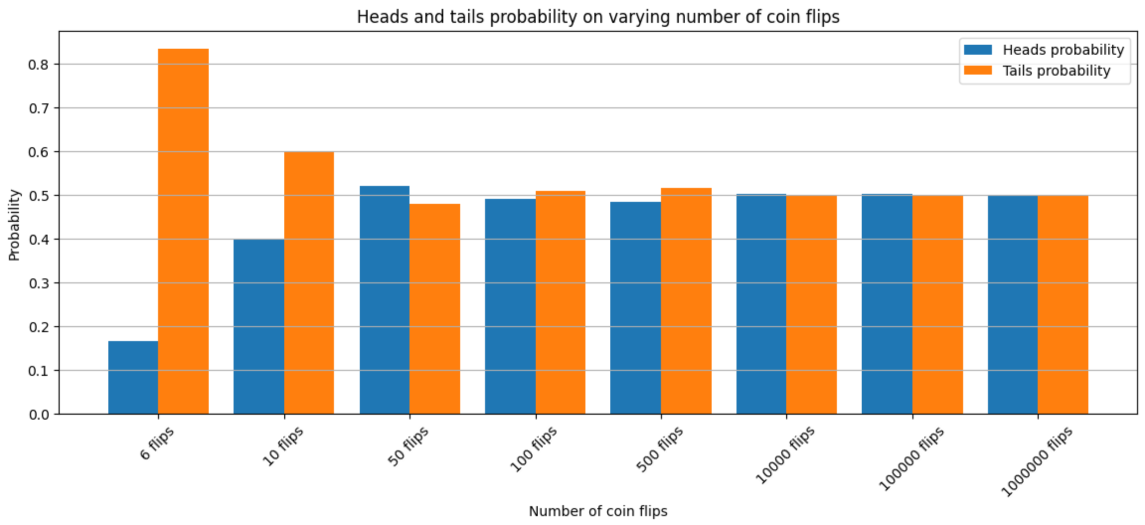

Figure: Bar plot of probabilities of getting head or tail on different number of coin flips

To find the probability of the sequence "HTH" (Head-Tail-Head) occurring in a set of three coin flips, the coin was flipped three times in each iteration. The outcome of each set of three flips was recorded, and the occurrence of "HTH" within these sets was tallied. After numerous trials were conducted, the number of instances where "HTH" appeared was divided by the total number of trials to determine the experimental probability of observing this specific sequence in a sequence of three coin flips.

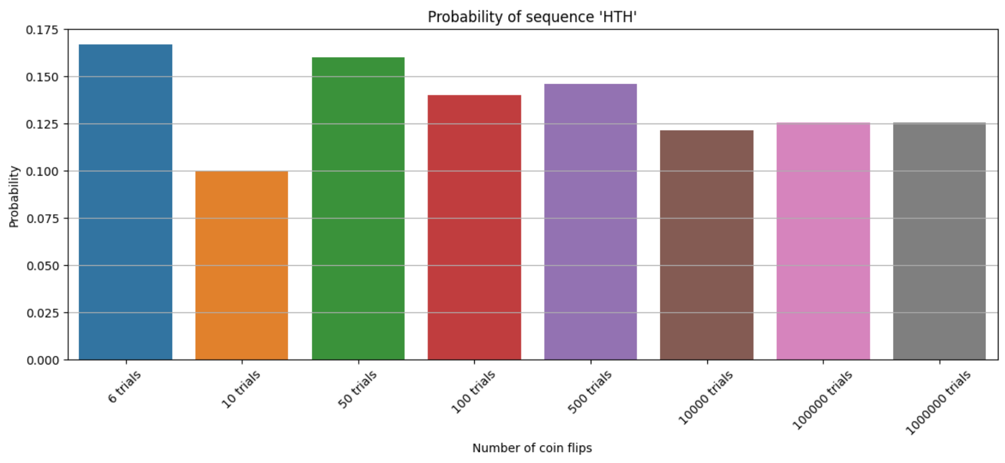

Figure: Bar plot of probabilities of getting sequence ‘HTH’ on multiple numbers of trials

Comparison and Interpretation

Probability of getting Head or Tail:

In theory, when we flip a fair coin, the chance of getting Heads or Tails should be exactly the same - both 0.5 or 50%. Experimentally, when performed a small number of coin flips, a slight imbalance in probability can be noticed like getting more Heads than Tails or vice versa which happens because in a small sample size, chance plays a bigger role. But as the number of coin flips is increased, something interesting occurs. The probabilities start to even out. So, with more flips, things become more predictable and balanced, just as we would expect in theory.

Probability of getting sequence 'HTH':

In theory, the probability of getting the HTH sequence is 0.125 or 12.5%. This means that if we were to flip a fair coin three times in a row, we'd expect to get HTH about 12.5% of the time.

When conducting a small number of trials (say, flipping the coin three times a few times), we might notice that the actual probability varies quite a bit. This variability occurs because small sample sizes are more susceptible to randomness. As the number of trials or iterations is increased (flipping the coin three times a larger number of times), we start to see the experimental probability of getting HTH settles down and gets very close to the theoretical probability of 0.125.

Conclusion

In summary, when flipping a fair coin, the probability of getting Heads or Tails is expected to be 50% each. In a limited number of trials, slight imbalances may occur due to chance. However, with an increase in the number of trials, the probabilities of getting Heads or Tails approach the expected 50%, aligning with theoretical predictions.

Similarly, when examining the specific sequence Head, Tail, Head (HTH), the theoretical probability stands at 12.5%. While some variability was observed in a small number of attempts, conducting a more extensive number of trials brought the experimental results much closer to the expected 12.5%.

These convergence is due to the law of large numbers, which states that with more trials, experimental results tend to converge to the expected theoretical probabilities which illustrates that increased trial numbers enhance predictability, reinforcing the reliability of probability theory. So, in practical terms, this shows us that probability theory isn't just theoretical; it works in real life. It helps us make better decisions and predictions in situations where there's uncertainty, like in finance or scientific experiments. It's like having a tool that helps us understand and deal with uncertainty in the real world.

Part 2: Monte Carlo Estimation of Pi

Introduction

The primary objective of this project is to develop a Python program implementing the quarter-circle technique using Monte Carlo simulation. In this technique, random (x, y) coordinates are generated within a unit square with a side length of 1 unit, considering the bottom-left corner of the square as the origin (0, 0). Concurrently, a circle with a radius of 1 unit, centered at the origin (0, 0) is employed where the upper right quadrant of the circle is enclosed within the unit square. The main objective is to identify the points that fall within this quarter-circle among all the generated points. By comparing the count of points inside the quarter-circle to the total number of generated points, we can derive an approximate estimation of the mathematical constant π (pi).

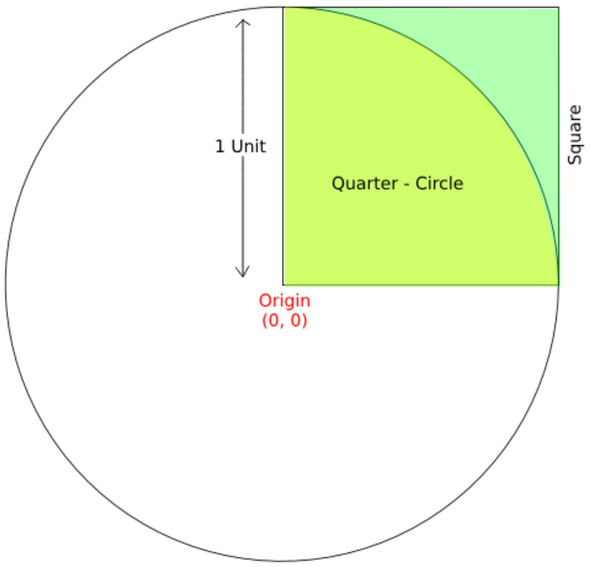

Fig: Quarter-Circle Monte Carlo Simulation Framework Visualization

Monte Carlo simulation is a technique that uses random sampling and repeated experiments to estimate complex outcomes or solve problems by simulating a large number of possible scenarios. It relies on the principles of randomness and probability to generate a large number of random samples, often referred to as "Monte Carlo trials". These trials imitate how a system or process might work by picking random numbers based on the chances of different outcomes. By repeatedly running these simulations, often in the thousands or millions, and aggregating the results, Monte Carlo simulations provide valuable insights into the range of possible outcomes, their probabilities, and the associated risks.

Simulation design and parameters

The simulation design encompassed two distinct approaches: non-sequential and sequential simulations. In the non-sequential variant, simulations were executed utilizing N values of 100 and 500, accompanied by the presentation of simulation diagrams illustrating the corresponding estimated values of pi.

In the sequential simulation method, a broader range of N values was employed, including 10, 50, 100, 500, 1000, 5000, 50,000, 100,000, 500,000, 750,000, and 1,000,000. For each of these N values, the simulation was performed, and the resulting estimated pi values were visually represented using bar plots.

The code and classes used in this task have been entirely developed independently. The implemented Quarter-Circle Monte Carlo pi estimator is designed to accept a list of tuple coordinates (x, y), with the constraint that all provided coordinates must fall within the range of 0 to 1 along both the x and y axes. To ensure compliance with this constraint, the list of tuple coordinates is generated using NumPy beforehand.

The generated data is fed to the pi estimator function which keeps track of points which fall within the upper right quadrant of the quarter circle and total number of points passed. These specific calculations serve as the foundational data for estimating the value of pi (π) using the following formula:

π (pi) ≈ 4 * (Number of Points Inside the Quarter-Circle / Total Number of Points)

Experimental Simulation Results

In the initial phase of non-sequential experimentation, a Monte Carlo simulation was conducted to estimate the value of π, employing a total of 100 randomly generated points. The outcome of this simulation revealed that 78 points fell within the quarter-circle, while 22 points resided outside the quarter-circle but within the encompassing square. Consequently, the estimated value of π was found to be approximately 3.12.

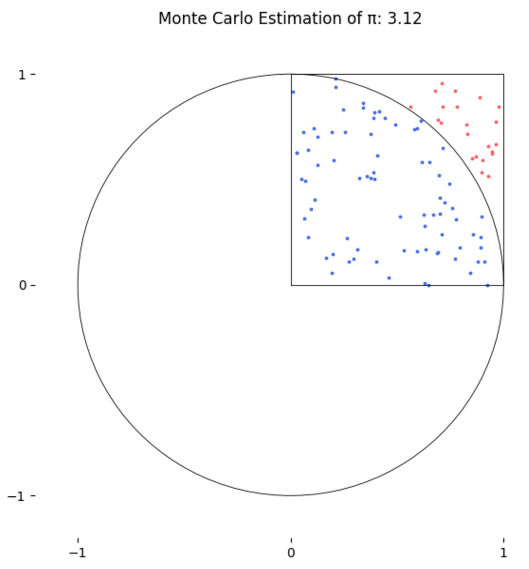

Fig: Quarter-Circle Monte Carlo Simulation Visualization with 100 (N) points

In the second phase of non-sequential experimentation, a Monte Carlo simulation was conducted to estimate the value of π, employing 5000 randomly generated points. Among these points, 3937 were found to fall within the quarter circle, while 1063 points resided outside the quarter circle but within the confines of the square. The resultant estimation of π based on this dataset of 5000 total points yielded an approximate value of 3.1496

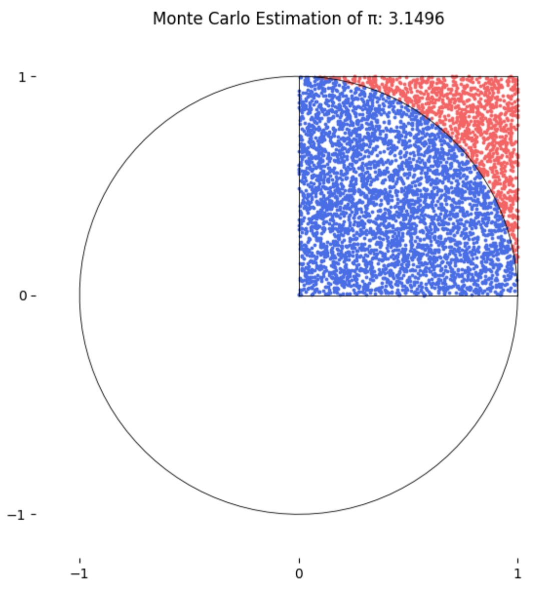

Fig: Quarter-Circle Monte Carlo Simulation Visualization with 5000 (N) points

For the sequential experimentation, Monte Carlo simulation was conducted to estimate the value of π (pi) using randomly generated (x, y) coordinates within a unit square. The number of total points varied across iterations, ranging from 10 to 1,000,000.

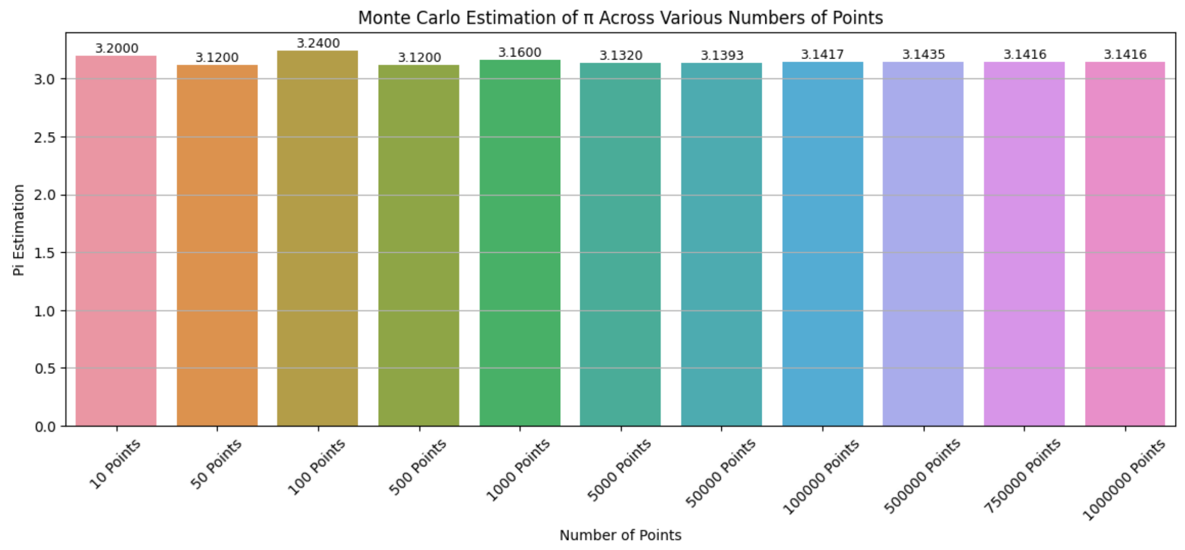

Fig: Visualization of Monte Carlo simulation results for π estimation using varying number of points

Conclusion

The analysis of the Monte Carlo simulation results revealed a notable trend: as the total number of points increased, a larger proportion fell within the quarter circle characterized by a radius of 1 unit, centered at the origin. Consequently, the estimation of π, derived from the ratio of points inside the quarter circle to the total points generated, exhibited an increasingly accurate approximation of the mathematical constant as the sample size expanded. The calculated π values for the respective sample sizes demonstrated a discernible convergence towards the actual value of π. This observation is in line with the probabilistic nature of the simulation, where a larger sample size provides a more comprehensive representation of the underlying probability distribution. his phenomenon underscores the efficacy of the Monte Carlo simulation technique in progressively approaching the precise value of π through random sampling, illustrating its practical utility in approximating complex mathematical constants.

Part 3: Poisson Process Simulation and Analysis

Introduction

The Poisson process, a fundamental concept in probability theory and statistics, provides a robust framework for modeling the occurrence of discrete events in diverse fields such as telecommunications, epidemiology, finance, and manufacturing. It is characterized by two key features: randomness and independence. Events unfold unpredictably in time or space, with a known constant average rate of occurrence. The arrival of an event is independent of the event before (waiting time between events is memoryless).

The objective in simulating a Poisson process is to explain the occurrence of events, as well as their precise timing and distribution within a specified time interval or spatial domain. A Poisson process is like a mathematical tool that helps us make sense of events happening randomly in time or space. When we simulate this process, it helps us figure out how likely events are to occur, how far apart they are, and how they tend to happen over time. This is super useful in different fields like statistics, physics, and engineering. The main aim here is to use computers to create a kind of imitation of a Poisson process so we can make smart choices, manage resources effectively, and study real-world stuff with a lot of precision. It's basically a way to understand randomness and make it work for us.

Simulation design and parameters

The simulation design encompasses a crucial function characterized by parameters such as the average rate (lambda) and either a singular duration or an array of time durations as inputs. This function is instrumental in generating essential information, including the total count of events, the mean inter-arrival time, and the standard deviation of inter-arrival times. These vital metrics are coupled with insightful visualizations, contributing to a comprehensive analysis of the simulated Poisson process.

In the context of non-sequential experimentation, the parameters are set with an average rate of 5 events per unit time and a fixed duration of 10 seconds. Conversely, in the case of sequential experimentation, the duration remains consistent at 10 seconds, but the average rate parameter varies. Specifically, a range of average rates, namely [2, 4, 6, 10], is applied.

Single Rate Simulation Result

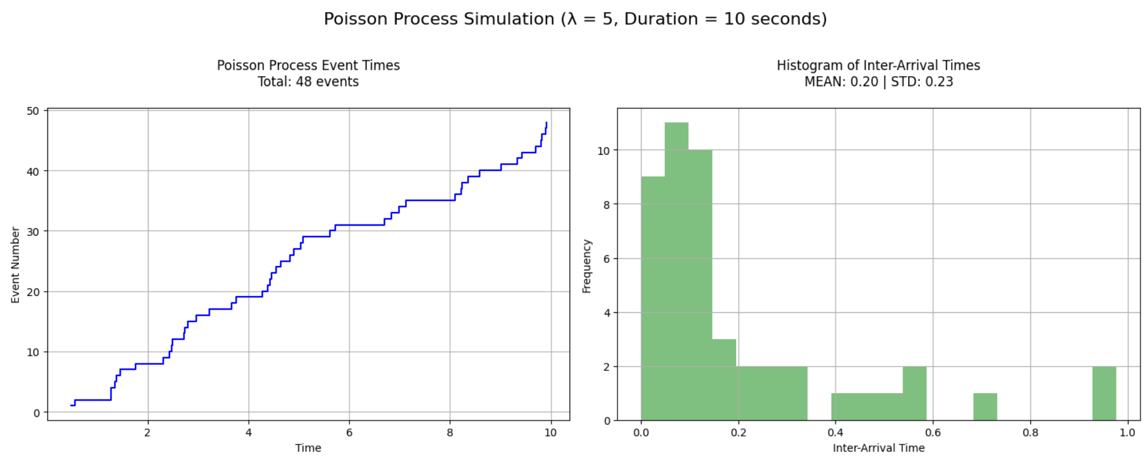

Fig: Poisson Process Simulation with average rate of 5 events for 10 seconds

Rate Variation Simulation Result

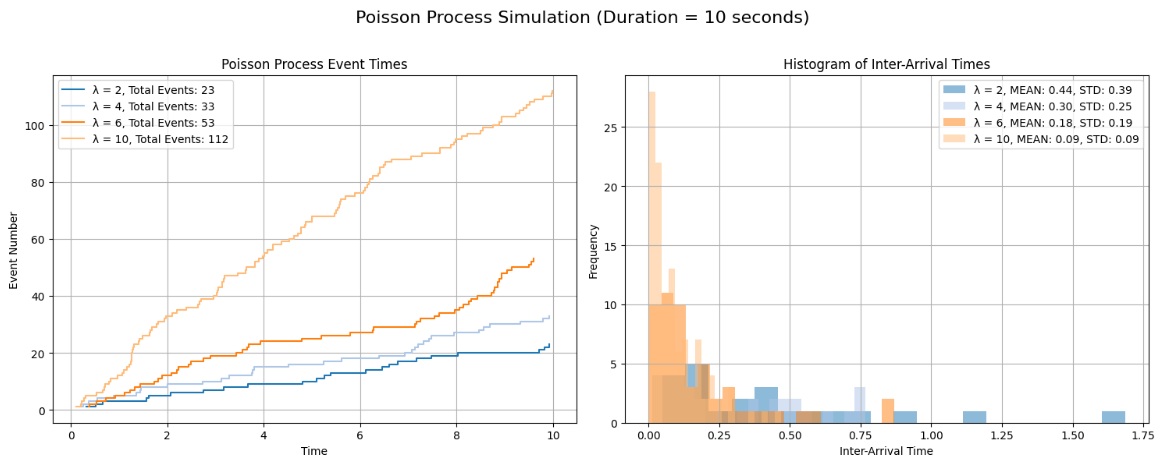

Fig: Poisson Process Simulation with varying average rate for different time intervals`

Findings and Conclusion

The experimental results reveal distinct patterns as the average rates of event occurrence vary within the sequential experiment, all conducted over a fixed duration of 10 seconds. As the average rate increases from 2 to 10 events per unit time, the total count of events exhibits a clear upward trend, demonstrating a direct relationship between the rate parameter and event frequency. Simultaneously, the mean inter-arrival times decrease substantially, indicating that events tend to occur more closely together at higher average rates. Moreover, the standard deviation of inter-arrival times exhibits a similar trend, reflecting reduced variability in event spacing as the rate increases. These findings underscore the fundamental principle of Poisson processes, where higher average rates result in more frequent and less variable event occurrences within the specified time frame