Module1: Asignment1-b

Sonia Sharapova

October 2024

1 Written Questions for Part b

Question 1

Considering the numeric types used in this calculation, what is the actual accuracy of this calculation?

Based on the pattern observed, the calculation's accuracy is dependent on the number of points in the simulation. The estimate of π gets closer to its true value with the more points, or darts, thrown. Given the nature of the calculations computed in python, the accuracy is limited by floating-point precision.

Question 2

Given your implementation, approximate how long it would take to compute π to 70,030 digits.

Since the number of points in the Monte Carlo simulation is proportional to the precision of π, and based on the trends in the benchmarking table, it would take an extremely large number of points and significant computation time to reach a precision of 70,030 digits. A different (faster) programming language like C could be used when the number of points gets exponentially larger, or the computation method could be revised. The Monte Carlo Method itself may be a limitation as it might not be the most effective/efficient method. Based on this Reddit post, methods such as Gaussian elimination may be more efficient.

	n points	t (seconds)	value of pi
0	1	0.002629	4.00000
1	10	0.000008	3.20000
2	100	0.000040	3.24000
3	1000	0.000320	3.12000
4	10000	0.003346	3.15320
5	100000	0.033543	3.14284

Figure 1: Benchmarking table from code.

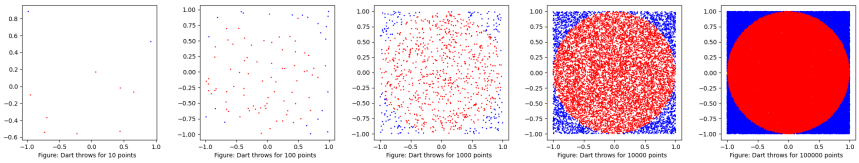

Figure 2: Hits from 'darts' with increasing number of points.

References:

Links to resources used:

Python Accuracy: Overleaf Documentation
Python Speed: Comparisons
Similar Problems (used as a reference): Monte Carlo Simulation with Python
Matplotlib (used as a reference): Documentation
Python Regular Expressions: Documentation

* Used GenAI for a few some clarifications on how to use yield in Python (for Ex.1 in Part c). It also gave pointers for making the code more efficient using regex.