Quasi-Monte Carlo π estimation algorithm

Jan Chylarecki

December 6, 2025

Overview

The **Quasi-Mone Carlo π estimation algorithm** is a set of tools based on the Monte Carlo method. The program runs Quasi (QMC) and "normal" Monte Carlo (MC) simulations, plots their convergence rates and graphs the scatter plot of points used in the QMC.

This project is designed to highlight the differences between random and deterministic sampling approaches. At the same time, it helps further develop my Python programming skills and lays the foundation for more complex, statistic-heavy topics.

Features

- Sobol QMC - a deterministic version of MC, which chooses evenly spaced points using the Sobol's sequence
- classic MC - used for comparison in convergence plots
- QMC and MC convergence plots - used to highlight the (generally) slower convergence rates of non-Quasi methods
- scatter plot - visualization of samples made by Sobol's QMC

Methodology

The value of π can be expressed in many ways. One geometric interpretation is particularly convenient for Monte Carlo–based methods.

Let O_1 be a circle with radius r inscribed in the square $ABCD$. Then, $ABCD$ sides are of length $2r$. Thus, the area of $ABCD$ is equal to $4r^2$ and area of O_1 is πr^2. Taking the ratio of these areas we notice that:

$$\frac{area\ O_1}{area\ ABCD} = \frac{\pi}{4} \implies \pi = 4 \cdot \frac{area\ O_1}{area\ ABCD}$$

This identity forms the basis of the Monte Carlo approximation of π. However, instead of computing these areas directly, we estimate this ratio by counting how many (uniformly) sampled points fall inside the circle versus inside the square.

As points are sampled uniformly, the probability that a point falls inside the circle is exactly the ratio of the two mentioned earlier areas:

$$P_{point \in O_1} = \frac{area\ O_1}{area\ ABCD}$$

Now, for n - number of points generated and for k - number of points inside the circle, by the *Law of Large Numbers*:

$$\frac{k}{n} \approx P_{point \in O_1} = \frac{area\ O_1}{area\ ABCD}$$

Therefore:

$$\pi \approx 4 \cdot \frac{k}{n}$$

Where the points are sampled using the Sobol sequence:

$$lim_{n\to\infty}\ \frac{1}{n}\sum_{i=1}^{n} f(x_i) = \int_{I^S} f$$

where $I^S = [0,1]^S$ is a S dimensional unit hypercube and f is a real integrable function over I^S.
The points x_i are vectors with S elements. In this case, $S = 2$, and the vectors are essentially coordinates of points on a plane. Point sampling is deterministic, unless scrambling is used - which is not the case in this project.

Sobol QMC, thanks to Sobol sequence being a low-discrepancy sequence, has a much faster convergence rate compared to MC. The rate of convergence for MC is around $O(\frac{1}{\sqrt{n}})$ and for QMC it is close to $O(\frac{1}{n})$. The QMC was used in this project as it gives smoother and less deviation-prone results. This can be seen in the comparison between MC and QMC convergence graphs.

Usage

Run the program from a terminal:

```
pi_estimate_main.py
```

Input the number of points you want to run the simulation on. A file with a scatter plot and a convergence plot will appear, and after that a file with two convergence plots (QMC vs MC).

Requirements

- Python 3.x
- matplotlib
- scikitlearn

Install dependencies:

```
pip install matplotlib
pip install scikitlearn
```

Future Work

- focus on the differences between convergence rates of QMC and MC
- explore how Sobol QMC performs vs MC in higher dimensions
- work on error bounds of QMC (*Koksma–Hlawka inequality*)

License