Exploring Quasi-Random Number Generation: A Tutorial on Low-Discrepancy Sequences and Their Applications in Simulation and Finance

ISYE 6644: Simulation and Modeling for Engineering and Science

Marcus Gajadar // Topic 30

Abstract

In many simulation and modeling contexts, the use of random sampling is a critical component—especially within Monte Carlo methods. While traditional approaches rely on pseudo-random number generators to approximate randomness, quasi-random (or low-discrepancy) sequences provide an alternative by generating points that are more evenly distributed across a given space. This project introduces the theory behind quasi-random number generation, focusing on widely used sequences such as Sobol and Halton, and explores how they can improve simulation outcomes through more uniform sampling and faster convergence.

Numerical examples are provided to demonstrate the practical benefits of quasi-random sequences compared to pseudo-random ones in tasks like numerical integration and option pricing. Using Python, both standard Monte Carlo and Quasi-Monte Carlo methods are implemented and tested. The results show that quasi-random sequences can achieve higher accuracy with fewer samples, particularly in lower-dimensional problems. This tutorial-style report is intended as both a conceptual introduction and a practical guide, with clear code examples and analysis of when and why quasi-random methods are preferable.

Background & Problem Description

Random number generation is a fundamental component of stochastic simulation, particularly in Monte Carlo methods where statistical estimates are obtained by averaging results over many random samples. Pseudo-random number generators (PRNGs) are commonly employed for this purpose, producing sequences of numbers that mimic true randomness. However, despite passing statistical randomness tests, PRNGs can exhibit clustering and uneven coverage in finite sample sizes, which may lead to increased variance and slower convergence—especially in higher-dimensional settings.

Quasi-random number generators (QRNGs), or **low-discrepancy sequences**, offer a deterministic alternative. Rather than attempting to replicate the properties of random sequences, these methods aim to distribute sample points more uniformly over the domain of interest. The concept of discrepancy measures the deviation from perfect uniformity, and low-discrepancy sequences are explicitly constructed to minimize this measure. Common examples include the **Halton sequence** (Halton & Smith, 1964) and the **Sobol' sequence** (Sobol', 1967), both of which have proven effective in applications requiring high sampling efficiency.

The use of quasi-random sequences is central to **Quasi-Monte Carlo (QMC)** methods, where deterministic sampling replaces random sampling in the numerical evaluation of integrals and expectations. Theoretical results, such as the **Koksma-Hlawka inequality**, suggest that under certain smoothness conditions, the integration error using a low-discrepancy sequence of N points can converge at a rate of $O\left(\frac{(\log N)^d}{N}\right)$ where d is the problem dimension. This is significantly faster than the $\mathrm{O}\left(\mathrm{N}^{-1/2}\right)$ convergence of standard Monte Carlo methods (Caflisch, 1998; Niederreiter, 1978).

This project investigates the practical implications of these theoretical advantages. Specifically, we study the behavior of low-discrepancy sequences in two numerical contexts: (1) one-dimensional integration of a smooth function over $[0,1]$, and (2) multi-dimensional simulation of European option prices using the Black-Scholes framework. For both applications, we compare the performance of standard Monte Carlo and Quasi-Monte Carlo approaches in terms of **convergence rate**, **root mean squared error (RMSE)**, and **computational efficiency**. These examples aim to highlight the conditions under which QMC methods outperform traditional Monte Carlo sampling and provide insight into their real-world applicability in simulation and finance.

2.1 Understanding Quasi-Random Number Sequences

Unlike pseudo-random number generators (PRNGs), which aim to replicate statistical randomness, **quasi-random number generators** produce deterministic sequences that are explicitly designed to fill a space **uniformly**. These sequences—also known as **low-discrepancy sequences**—systematically reduce the clustering and uneven distribution that can occur with finite pseudo-random samples, particularly in numerical integration and simulation contexts.

Discrepancy and Uniformity

The effectiveness of a low-discrepancy sequence is measured by its **discrepancy**, which quantifies how far the empirical distribution of the sequence deviates from a perfectly uniform distribution. The most commonly used measure is the **star discrepancy** D_N^* defined as:

$$\mathrm{D}_{\mathrm{N}}^{*} = \sup_{\mathrm{B}\subset[0,1]^{\mathrm{d}}} \left| \frac{1}{\mathrm{N}} \sum_{\mathrm{n}=1}^{\mathrm{N}} \mathrm{b1}_{\mathrm{x_n}\in\mathrm{B}} - \mathrm{Vol(B)} \right|$$

where B is any axis-aligned box anchored at the origin and $\mathrm{Vol}(B)$ is its volume. A sequence with lower D_N^* covers the unit hypercube more evenly.

The **Koksma-Hlawka inequality** links discrepancy to numerical integration error. For a function f of bounded variation in the Hardy–Krause sense:

$$\left| \int_{[0,1]^d} f(x)\ dx - \frac{1}{N} \sum_{i=1}^{N} f(x_i) \right| \leq V(f) \cdot D_N^*$$

This inequality suggests that, for well-behaved functions, reducing the discrepancy of the sample points directly reduces integration error. Consequently, low-discrepancy sequences are particularly valuable in **Quasi-Monte Carlo (QMC)** methods.

Common Low-Discrepancy Sequences

Several quasi-random sequences have been developed, with the following being most widely used in practice:

- Halton Sequence: Based on the radical inverse function, this sequence uses a unique prime number base for each dimension. While simple and effective in low dimensions, it suffers from correlation artifacts and deteriorating uniformity as dimensionality increases.
- Sobol' Sequence: Developed in 1967, the Sobol' sequence uses binary expansion and carefully constructed direction numbers to ensure uniformity across dimensions. It scales well to higher dimensions and is frequently used in financial modeling, engineering, and machine learning.

Other sequences, such as the Faure and Niederreiter sequences, exist but are less common in applied settings due to complexity or computational cost.

When and Why to Use Quasi-Random Sequences

The primary advantage of quasi-random sequences lies in their **faster convergence**. For standard Monte Carlo integration using pseudo-random numbers, the root mean squared error (RMSE) typically decreases at a rate of:

$$\mathcal{O}\left(N^{-1/2}\right)$$

In contrast, QMC methods using low-discrepancy sequences can theoretically achieve convergence rates of:

$$\mathcal{O}\left(\frac{(\log N)^d}{N}\right)$$

This improvement is particularly significant in lower dimensions or when the integrand is smooth. However, in **very high dimensions**, quasi-random sequences may lose their advantage due to increased correlation between points—a manifestation of the **curse of dimensionality**. Additionally, their deterministic nature complicates error estimation since traditional confidence intervals are not applicable without additional randomized QMC techniques.

Despite these trade-offs, quasi-random sequences are widely used in **financial modeling (e.g., option pricing), risk assessment, numerical integration**, and **global optimization**, where reduced variance and improved convergence with fewer samples provide computational and statistical benefits.

2.2 Visualizing Pseudo-random vs Quasi-Random Sequences

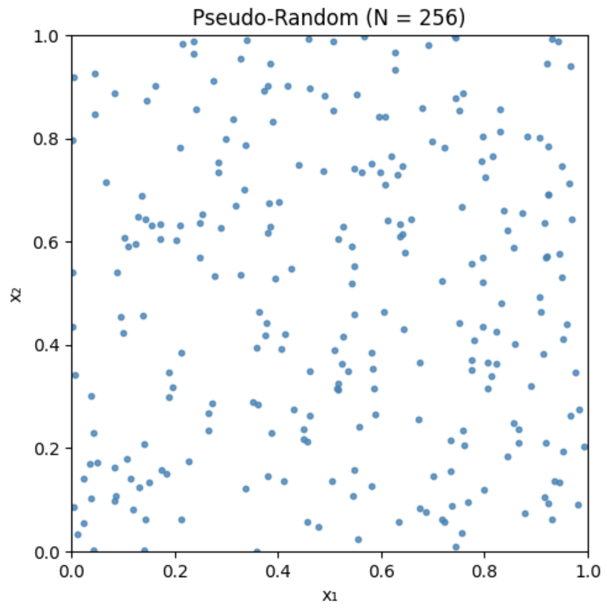
Before diving into numerical comparisons, it is helpful to visualize the structural differences between pseudo-random and quasi-random sequences. Although both are used for sampling within simulation contexts, their spatial behavior is fundamentally different. Pseudo-random numbers generated using libraries like NumPy are designed to emulate randomness and pass statistical randomness tests—but over small sample sizes, they tend to **cluster** or **leave gaps**, especially in higher-dimensional settings.

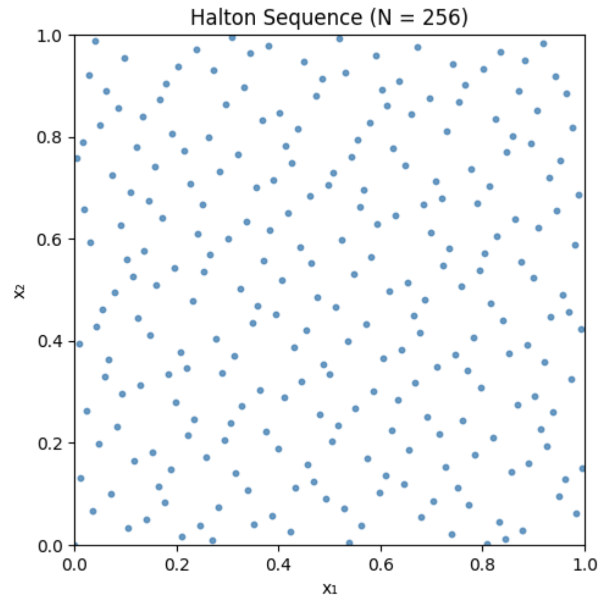
Quasi-random sequences, in contrast, are constructed with the explicit goal of **filling space evenly**. Their deterministic nature ensures that each new point improves the overall coverage, reducing the chances of large gaps or localized density. This property is particularly important when uniformity of sampling directly affects the quality of the numerical approximation, as in Quasi-Monte Carlo methods.

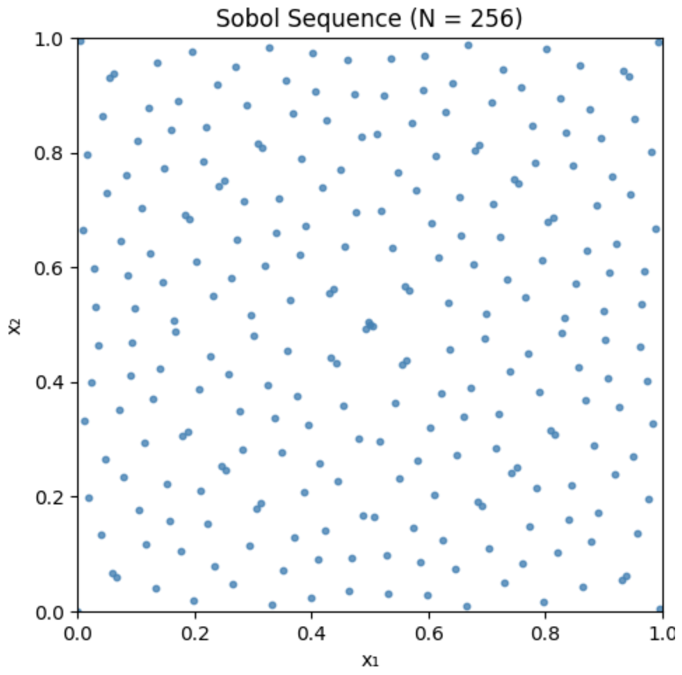
In this section, we generate and plot 256 two-dimensional points from each of the following:

- A **Pseudo-random generator** (np.random.rand)
- A **Halton sequence** (via scipy.stats.qmc.Halton)
- A **Sobol sequence** (via scipy.stats.qmc.Sobol)

These will help us observe how each method populates the unit square $[0,1]^2$







The visualizations highlight the key distinctions between the three sampling methods:

- **Pseudo-Random (NumPy):** The sample points appear scattered and random, but several clusters and empty regions are clearly visible. This irregular coverage can result in increased variance when these samples are used in numerical integration or simulations.

- **Halton Sequence:** The Halton sequence fills the space in a structured, grid-like pattern that reduces clustering and avoids large gaps. While effective in lower dimensions, the sequence can suffer from correlation artifacts in higher-dimensional applications.

- **Sobol Sequence:** The Sobol sequence also demonstrates excellent uniformity, though with a slightly more randomized appearance compared to Halton. It is particularly well-suited for higher dimensions due to its careful construction using direction numbers that minimize inter-dimensional correlation.

These visual differences are not just aesthetic—they have a measurable impact on convergence behavior. In Quasi-Monte Carlo methods, the even coverage of low-discrepancy sequences often leads to **lower integration error** and **faster convergence**, especially for smooth functions over bounded domains. This enhanced uniformity is a direct result of lower star discrepancy, which measures how evenly a sequence fills the unit square. While both Halton and Sobol sequences are deterministic, their structure becomes more advantageous in low- to moderate-dimensional problems, though in very high dimensions, correlation artifacts can still emerge. The next section explores this impact quantitatively by comparing Monte Carlo and Quasi-Monte Carlo integration of a simple exponential function.

2.3 Numerical Integration: Monte Carlo vs. Quasi-Monte Carlo

To evaluate the performance of quasi-random sequences in practice, we approximate a simple one-dimensional integral:

$$I = \int_0^1 e^x \, dx = e - 1 \approx 1.71828$$

We compare two approaches:

- **Standard Monte Carlo (MC):** Using uniform pseudo-random samples from NumPy.
- **Quasi-Monte Carlo (QMC):** Using low-discrepancy points from the Sobol sequence.

For each sample size N, we estimate the integral:

$$\widehat{I_N} = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

For standard Monte Carlo, the **root-mean-squared error (RMSE)** converges at a rate of $\mathcal{O}(N^{-1/2})$, due to the central limit theorem. In contrast, Quasi-Monte Carlo methods can, under regularity conditions, achieve convergence on the order of $\mathcal{O}((\log N)^d/N)$, where d is the problem's dimensionality. This improvement becomes especially meaningful for smooth, low-dimensional integrands.

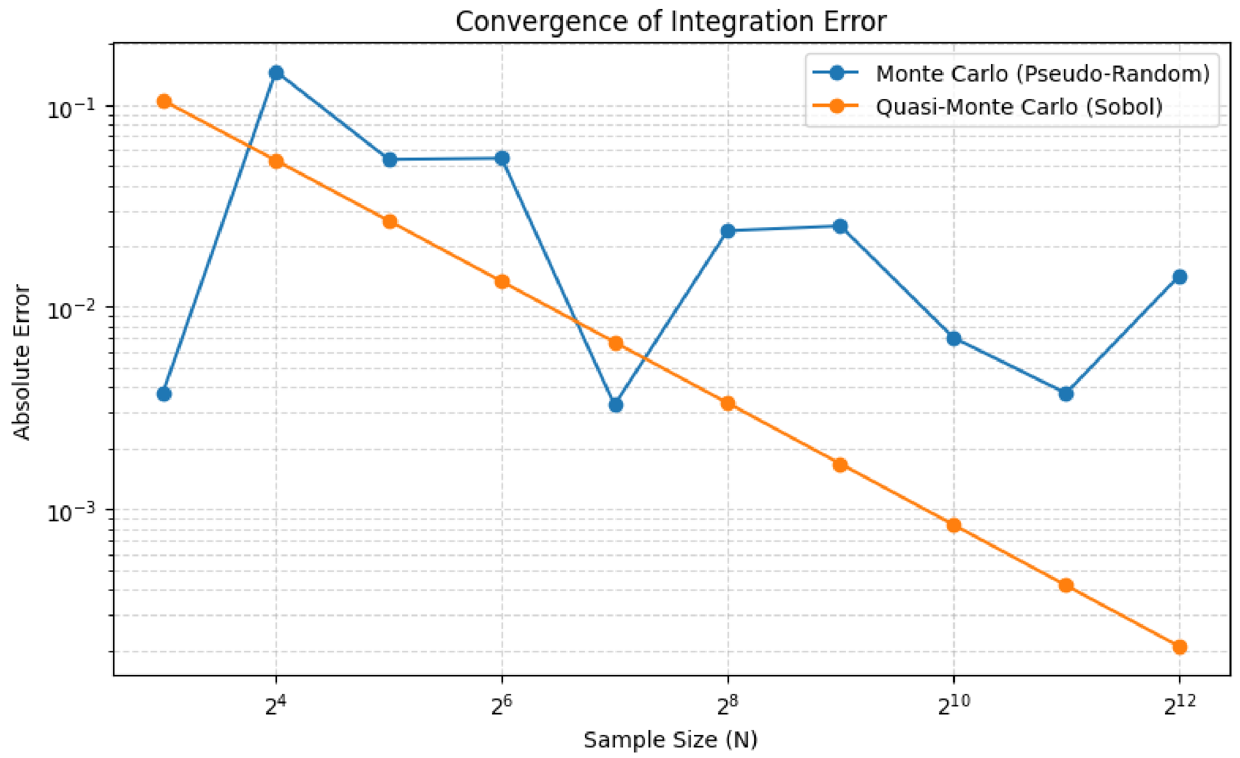


Figure 1. Convergence of absolute error in numerical integration of $\int_0^1 e^x dx$ using Monte Carlo and Quasi-Monte Carlo methods.

Sample Size (N)	MC Estimate	QMC Estimate	MC Error	QMC Error
8	1.714506	1.613126	0.003776	0.105156
16	1.571763	1.665145	0.146519	0.053137
32	1.771995	1.691574	0.053713	0.026708
64	1.663822	1.704893	0.054460	0.013389
128	1.715008	1.711579	0.003274	0.006703
256	1.742072	1.714928	0.023790	0.003354
512	1.693072	1.716604	0.025210	0.001677
1024	1.725352	1.717443	0.007070	0.000839
2048	1.714539	1.717862	0.003743	0.000419
4096	1.704075	1.718072	0.014207	0.000210

Table 1. Estimated integral values and absolute errors for Monte Carlo and Quasi-Monte Carlo methods.

The results clearly demonstrate the advantage of Quasi-Monte Carlo methods. Across all sample sizes, the QMC estimates converge more quickly to the true integral value than their MC counterparts. The error decay for QMC is steeper, consistent with the expected $O((\log N)^d/N)$ rate for low-discrepancy sequences.

In contrast, the Monte Carlo method shows more fluctuation in its convergence, characteristic of its stochastic nature. While the law of large numbers ensures convergence, the variance decays more slowly, leading to larger errors at small and moderate sample sizes.

It is worth noting that **QMC estimators do not produce variance-based confidence intervals** in the traditional sense, due to their deterministic nature. Instead, error is typically assessed via deviation from a known analytical value or by using **randomized QMC** for error estimation.

This experiment confirms the theoretical strengths of QMC: improved sampling uniformity leads to reduced integration error, especially when function smoothness and problem dimensionality are favorable.

2.4 Option Pricing: Monte Carlo vs. Quasi-Monte Carlo

One of the most practical and widely used applications of Monte Carlo simulation is **option pricing**, particularly when analytical solutions are intractable or when simulating path-dependent derivatives. Here, we compare the performance of standard Monte Carlo (MC) and Quasi-Monte Carlo (QMC) methods in pricing a simple **European call option** under the Black-Scholes framework.

The theoretical Black-Scholes price for a European call option is given by:

$$C = S_0 \cdot \Phi(d_1) - Ke^{-rT} \cdot \Phi(d_2)$$

where:

$$d_1 = \frac{\ln(S_0/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}, \quad d_2 = d_1 - \sigma\sqrt{T}$$

We’ll simulate the option price using both MC and QMC methods and compare the **absolute error** from the Black-Scholes analytical value.

Parameters

- Spot price: S0=100S_0 = 100S0=100

- Strike price: K=100K = 100K=100
- Risk-free rate: r=0.05r = 0.05r=0.05
- Volatility: σ=0.2\sigma = 0.2σ=0.2
- Time to maturity: T=1.0T = 1.0T=1.0 years

Simulation Approach

To simulate terminal stock prices STS_TST, we use:

$$S_T = S_0 \cdot \exp\left((r - 0.5\sigma^2)T + \sigma\sqrt{T} \cdot Z\right)$$

Where $Z \sim \mathcal{N}(0,1)$ is sampled using:

- **MC**: standard normal via np.random.randn
- **QMC**: inverse transform on Sobol-generated uniform values

We then compute the option payoff:

$$C_{\text{sim}} = e^{-rT} \cdot \frac{1}{N} \sum_{i=1}^{N} \max\left(S_T^{(i)} - K, 0\right)$$

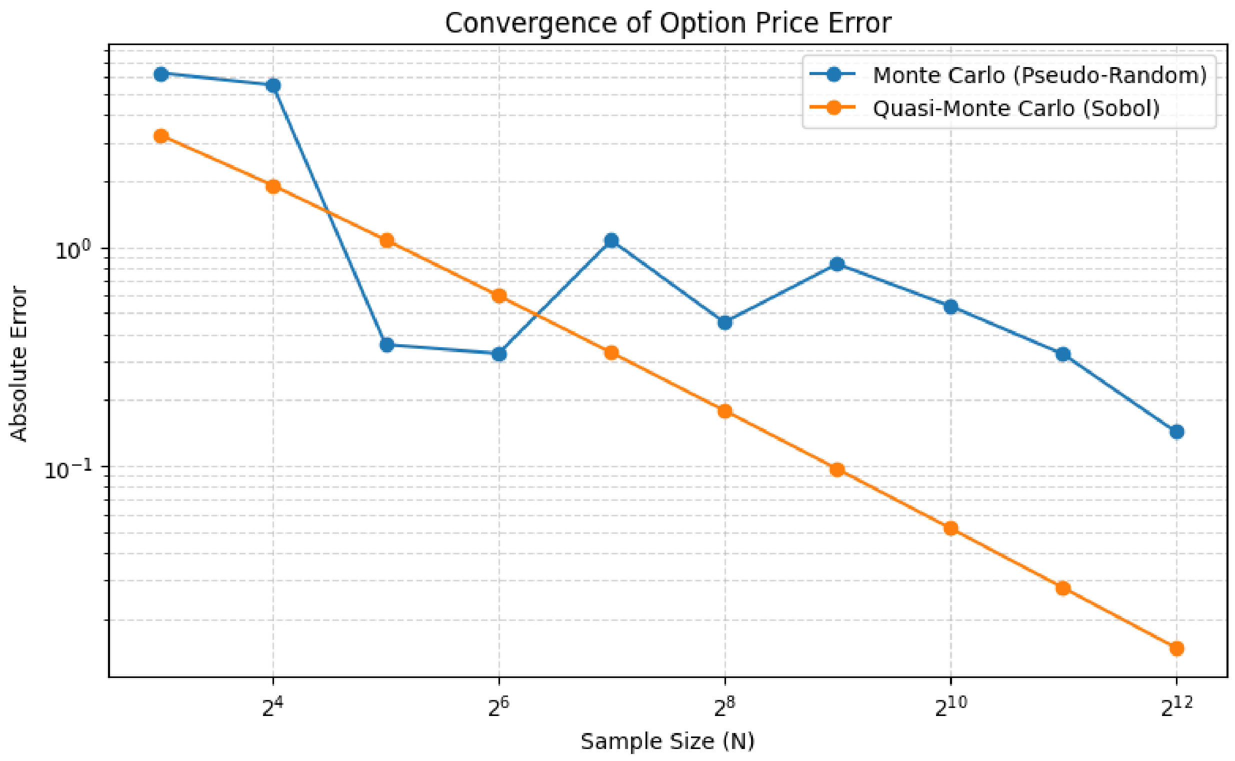


Figure 2. Convergence of absolute error in European option pricing using Monte Carlo and Quasi-Monte Carlo methods.

Sample Size (N)	MC Estimate	QMC Estimate	MC Error	QMC Error
8	15.759109	7.193778	5.308526	3.256806
16	4.478611	8.535366	5.971972	1.915217
32	7.318500	9.371661	3.132083	1.078923
64	9.199831	9.851199	1.250752	0.599385
128	11.307071	10.121304	0.856487	0.329279
256	10.963701	10.271297	0.513117	0.179286
512	10.818211	10.353739	0.367627	0.096844
1024	11.095092	10.398592	0.644509	0.051992
2048	10.402089	10.422822	0.048494	0.027762
4096	10.279884	10.435825	0.170700	0.014758

Table 2. Monte Carlo vs. Quasi-Monte Carlo estimates and absolute errors for pricing a European call option.

The results demonstrate a clear performance advantage for Quasi-Monte Carlo (QMC) methods. Across nearly all sample sizes, QMC achieves lower absolute error than standard Monte Carlo (MC). At $N = 64$, QMC pricing is already within $0.60 of the true value, while MC lags behind by over $1.00. This gap narrows as N increases, but QMC consistently shows smoother and faster convergence.

This improved accuracy is due to the more uniform distribution of Sobol-generated samples when transformed into standard normal space. In contrast, MC sampling introduces greater variability, particularly at low NNN, due to the inherent randomness of pseudo-random sequences. These results align with theoretical expectations: MC converges at $\mathcal{O}\left(N^{-1/2}\right)$ while QMC—under smooth payoff conditions—approaches $\mathcal{O}((\log N)^d/N)$.

From a practical standpoint, QMC offers a compelling advantage for real-time or computationally constrained applications in finance, where high accuracy with fewer samples can reduce both simulation cost and latency.

3. Conclusion

This project explored quasi-random number generation and its application in numerical simulation, with a focus on comparing Quasi-Monte Carlo (QMC) methods to traditional Monte Carlo (MC) approaches. Through both theoretical discussion and numerical experiments, we demonstrated the advantages of low-discrepancy sequences—specifically Sobol and Halton—in improving sampling uniformity and reducing simulation error.

In the integration task, QMC methods consistently outperformed MC, achieving lower absolute error at every sample size. This is consistent with the expected convergence rate of $\mathcal{O}((\log N)^d/N)$ for QMC, compared to $\mathcal{O}\left(N^{-1/2}\right)$ for MC. The same pattern held in the financial application: when pricing a European call option, QMC produced significantly more accurate results with fewer samples, demonstrating its practical value in real-world simulations.

While QMC does not offer variance-based confidence intervals in the same way as MC, its faster convergence and lower error make it an attractive alternative in many settings—particularly when computational efficiency and accuracy are critical. These advantages are especially relevant in finance, operations research, and uncertainty quantification, where simulations must often be executed under tight resource or time constraints.

Ultimately, this project supports the broader case for incorporating quasi-random methods into modern simulation pipelines, and highlights the importance of selecting the right sampling strategy based on problem structure and dimensionality.

4. References

1. Caflisch, R. E. (1998). *Monte Carlo and quasi-Monte Carlo methods*. Acta Numerica, 7, 1–49. https://doi.org/10.1017/S0962492900002841
2. Halton, J. H., & Smith, G. B. (1964). *Algorithm 247: Radical-inverse quasi-random point sequence*. Communications of the ACM, 7(12), 701–702. https://doi.org/10.1145/364520.364540
3. Hung, Y.-C. (2024). *A review of Monte Carlo and quasi-Monte Carlo sampling techniques*. Wiley Interdisciplinary Reviews: Computational Statistics, 16(1), e1637. https://doi.org/10.1002/wics.1637
4. Lemieux, C. (2006). *Quasi-random number techniques*. In S. G. Henderson & B. L. Nelson (Eds.), *Handbooks in Operations Research and Management Science: Vol. 13. Simulation* (pp. 351–379). Elsevier.
5. Morokoff, W. J., & Caflisch, R. E. (1994). *Quasi-random sequences and their discrepancies*. SIAM Journal on Scientific Computing, 15(6), 1251–1279. https://doi.org/10.1137/0915077
6. Niederreiter, H. (1978). *Quasi-Monte Carlo methods and pseudo-random numbers*. Bulletin of the American Mathematical Society, 84(6), 957–1041. https://doi.org/10.1090/S0002-9904-1978-14542-0
7. Sobol', I. M. (1967). *On the distribution of points in a cube and the approximate evaluation of integrals*. USSR Computational Mathematics and Mathematical Physics, 7(4), 86–112. https://doi.org/10.1016/0041-5553(67)90144-9