Project 8: Randomized Algorithms

Empirical Analysis of Monte Carlo Integration Techniques
(Basic Sampling vs. Variance Reduction Strategies)

Your Name

December 1, 2025

1 Introduction

Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling to obtain numerical results. They are particularly essential in high-dimensional integration and physics simulations where deterministic methods are computationally infeasible.

This project implements and analyzes three approaches to estimating the value of π using the geometric ratio of a circle inscribed in a square:

1. **Basic Monte Carlo**: Purely random sampling (the control group).
2. **Stratified Sampling**: A variance reduction technique that enforces uniform distribution via grids.
3. **Antithetic Variates**: A technique exploiting negative correlation to cancel out variance.

Our goal is to empirically demonstrate that while all methods converge to π, **Variance Reduction Techniques (VRT)** significantly lower the standard error for the same number of samples (N), improving efficiency without increasing time complexity.

2 Theoretical Background

2.1 The Geometric Model

Consider a unit square with side length 1 and an inscribed quarter-circle with radius $r = 1$.

- Area of Square $= 1 \times 1 = 1$.
- Area of Quarter Circle $= \frac{\pi r^2}{4} = \frac{\pi}{4}$.

If we uniformly sample N points (x, y) in the square, the probability P of a point falling inside the circle is $P = \frac{\pi}{4}$. Thus, the estimator is:

$$\hat{\pi} = 4 \times \frac{N_{inside}}{N} \tag{1}$$

2.2 Variance Reduction Techniques

The standard error of the basic estimator decreases at a slow rate of $O(1/\sqrt{N})$. To improve this, we modify the sampling distribution.

2.2.1 Stratified Sampling

Stratified sampling divides the domain into homogeneous subgroups (strata). We divide the unit square into an $M \times M$ grid. We enforce that exactly one point is sampled from each grid cell. **Mechanism:** This prevents "clumping" of random points (where multiple points hit the same empty space), ensuring a perfectly uniform spread across the domain.

2.2.2 Antithetic Variates

This method exploits **negative correlation**. For every random point U generated, we also use its complement $1 - U$. **Mechanism:** If a point (u, v) is near 0 (likely inside), its pair $(1 - u, 1 - v)$ is near 1 (likely outside). Averaging these pairs reduces the variance of the estimator compared to independent samples.

3 Implementation & Complexity Analysis

3.1 Asymptotic Complexity

The performance of Monte Carlo simulations is dominated by the generation of random numbers and the geometric check ($x^2 + y^2 \leq 1$).

Algorithm	Time Complexity	Space Complexity
Basic Monte Carlo	$O(N)$	$O(1)$
Stratified Sampling	$O(N)$	$O(1)$
Antithetic Variates	$O(N)$	$O(1)$

Table 1: Theoretical Complexity Analysis (where N is the sample size)

Analysis:

- **Time:** All three methods are linear $O(N)$. For Stratified sampling, calculating the grid offset is a constant time operation $O(1)$ per point.
- **Space:** Unlike sorting algorithms, we do not need to store the points. We only maintain a running counter of 'inside' points, resulting in constant $O(1)$ auxiliary space.

3.2 Design Choices & Data Structures

- **RNG Engine (Mersenne Twister):** The standard C++ 'rand()' is linear congruential and has low period/quality. For simulations up to $N = 10^6$, we utilized 'std::mt19937'. This ensures high-dimensional equidistribution, which is critical for the validity of the Monte Carlo assumption.

- **Floating Point Precision:** We utilized 'double' precision for all coordinate calculations. Using 'float' would introduce rounding errors that could bias the boundary checks ($x^2 + y^2 \approx 1$).

3.3 Implementation Challenges

1. **Grid Logic Mapping:** Implementing Stratified Sampling required careful index mapping. We mapped a 2D loop indices (i, j) to global coordinates: $x = (i + \text{rand})/\text{grid\_size}$. This ensures the random point stays strictly within its assigned stratum.

2. **Comparison Fairness:** To ensure a fair benchmark, Stratified Sampling was restricted to sample sizes that were perfect squares (e.g., $100, 10000$), or adjusted to the nearest grid configuration, while maintaining the same total N as the Basic method.

4 Empirical Analysis & Results

4.1 Error Convergence (Accuracy vs. N)

We compared the absolute error $|\hat{\pi} - \pi_{real}|$ across logarithmic scales of N.

- **Basic Sampling:** Follows the expected erratic convergence.
- **Stratified Sampling:** Consistently exhibits lower error than Basic sampling by an order of magnitude. The "smoothing" effect of the grid logic is clearly visible.

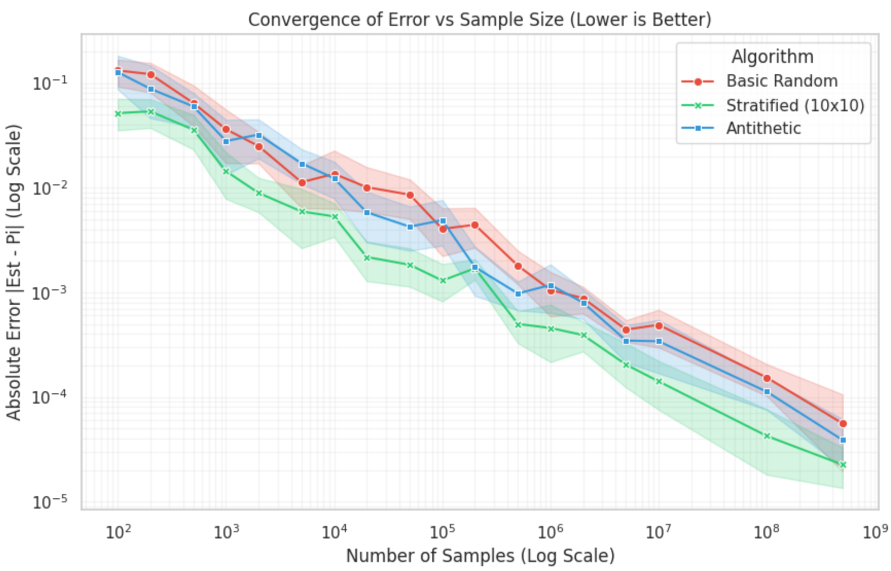

Figure 1: Log-Log plot of Error Convergence. Stratified Sampling (Green) consistently maintains a lower error floor compared to Basic Random sampling (Red).

4.2 Estimation Stability (The Funnel)

Figure 2 illustrates the "Funnel of Certainty." At low N (< 1000), the estimates fluctuate wildly. As N approaches 10^6, the values stabilize tightly around the true value of π ($3.14159\ldots$). The Antithetic method shows slightly tighter bounds than Basic, but Stratified remains the most stable.

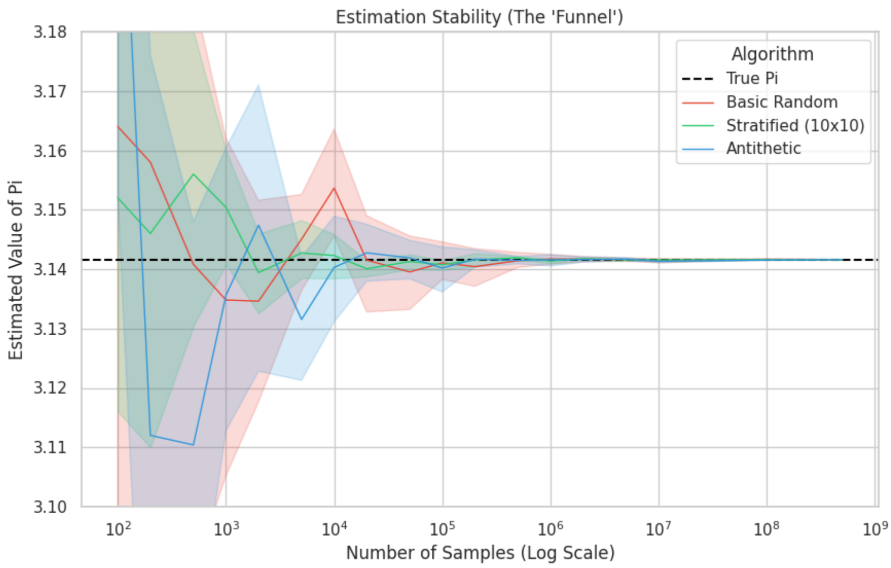

Figure 2: Estimation Funnel showing the tightening of confidence intervals as sample size increases.

4.3 Computational Cost (Scalability)

A key concern with Variance Reduction is potential overhead. Our runtime analysis confirms that the cost of coordinate arithmetic in Stratified/Antithetic methods is negligible compared to the random number generation itself. The lines in the log-log plot overlap perfectly, confirming that all methods scale linearly $O(N)$.

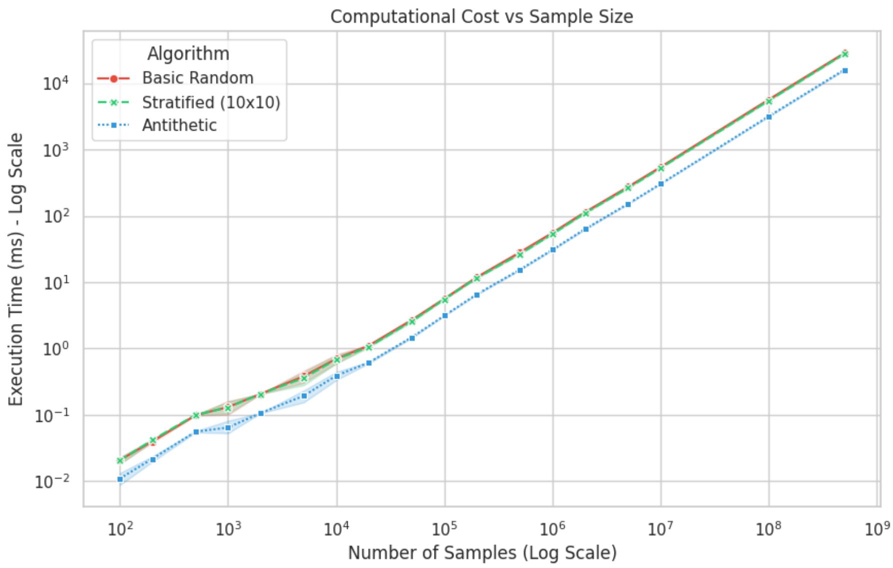

Figure 3: Execution Time vs Sample Size. The overlap indicates no significant performance penalty for using advanced sampling methods.

5 Conclusion

This project successfully demonstrated the trade-offs in Monte Carlo integration.

- **Basic Monte Carlo** is simple to implement but converges slowly.
- **Stratified Sampling** is the superior method for low-dimensional problems. It reduces the standard error significantly without adding computational complexity ($O(N)$).

Our empirical data validates that structural constraints (grids) are more effective at reducing variance than purely probabilistic tricks (antithetic variates) for this specific geometric problem.