Short research articles

Deterministic versus Monte Carlo Quadrature: Theory, Experiments, and the Role of Dimension

[1]

Abstract

This paper presents a theoretical and computational comparison between classical deterministic numerical integration methods (trapezoidal, midpoint, and Simpson), Quasi–Monte Carlo (Sobol') sampling, Smolyak sparse-grid quadrature, and the standard Monte Carlo estimator. We establish convergence rates for each method, specifying regularity assumptions and dimensional dependence, and we clarify the construction of multidimensional benchmark functions via separable tensor products. Numerical experiments are conducted on smooth, oscillatory, and weakly singular test functions in dimensions $1 \leq d \leq 5$, with performance evaluated in terms of absolute error, total function evaluations, CPU time, and Monte Carlo variability (mean, standard error, and interquartile range). The results confirm that deterministic quadratures achieve high accuracy at low dimension but suffer from exponential cost growth with d, while Monte Carlo maintains dimension-independent convergence at rate $O(n^{-1/2})$. Quasi–Monte Carlo and sparse-grid methods provide competitive accuracy in moderate dimensions. Overall, deterministic schemes are well-suited to smooth low-dimensional integrals, whereas Monte Carlo and QMC methods remain advantageous for high-dimensional or irregular integration problems. problems.

Numerical Integration, Deterministic Quadrature, Monte Carlo Method, Convergence Analysis, Computational Efficiency, Curse of Dimensionality, Error Estimation, Simpson's Rule, Trapezoidal Rule, Midpoint Method, High-dimensional Integration, Variance Reduction, Quasi–Monte Carlo Sampling

1 Introduction

When addressing numerical integration, two main approaches stand out: classical deterministic methods and probabilistic techniques such as Monte Carlo. In this article, we rigorously compare their performance, analyzing how their accuracy and convergence rate evolve with the complexity of the problems, particularly in high-dimensional spaces.

2 Literature Review

Numerical integration constitutes a fundamental pillar of modern numerical analysis. Since the pioneering works of Newton and Cotes in the 18th century, classical quadrature formulas such as the trapezoidal, Simpson, or Gauss–Legendre methods have been central to methodological developments aimed at approximating integrals of continuous and sufficiently regular functions [1, 3]. These methods are based on deterministic schemes whose convergence strongly depends on the regularity of the function and on the discretization step h.

However, as problems have become more complex particularly in computational physics, Bayesian statistics, and artificial intelligence. The integration of high-dimensional functions ($d > 3$) has become a major challenge. This phenomenon, known as the *curse of dimensionality*, was highlighted as early as the 1960s by Bellman [4]. Indeed, the number of evaluations required to achieve a given accuracy grows exponentially with the dimension, making deterministic methods rapidly impractical. It is in this context thatthat Metropolis and Ulam developed the Monte Carlo method during World War II [6]. This stochastic method, which is based on the central limit theorem and the law of large numbers, estimates an integral by averaging values that are uniformly distributed over the integration domain and randomly sampled. With an asymptotic convergence of $O(n^{-1/2})$, Monte Carlo methods' accuracy is dependent on the number of samples n rather than the space's dimension, in contrast to quadrature methods. The approach has a significant advantage in high-dimensional spaces because of this feature. Since then, several works have expanded on its variations: Low-discrepancy sequences are used in quasi-Monte Carlo algorithms, which were first presented by Halton (1960) and Sobol (1967). This improves convergence toward $O(n^{-1}(\log n)^d)$ [6, 7]. Other approaches, such as variance reduction or importance sampling, aim to reduce the multiplicative constant without changing the theoretical order [8, 9].

Despite this wealth of developments, the literature shows that few systematic comparative studies have been conducted between deterministic and stochastic approaches in a unified framework—one that evaluates not only precision but also **computational cost** and **robustness** to function regularity. In most numerical analysis textbooks, the comparison remains qualitative and limited to low-dimensional or regular-function cases [10, 11].

Problem Statement and Relevance of This Work. Our study precisely addresses this gap. We propose a joint theoretical and experimental analysis that aims to:

- mathematically demonstrate the convergence orders of classical methods (trapezoidal, midpoint, Simpson);
- empirically compare their performance with that of the Monte Carlo method for functions of increasing complexity;
- measure and visualize the efficiency of each method as a function of **computational time** and **problem dimensionality**.

This work therefore has a dual significance. On one hand, it clarifies the asymptotic behavior of classical methods under the effect of dimension; on the other, it highlights the universal character of Monte Carlo, able to maintain stable accuracy regardless of the dimension or regularity of the function. This analysis provides valuable insights for fields where numerical integration is ubiquitous: scientific computing, machine learning, computational economics, and the simulation of complex systems.

3 Theoretical Foundations

3.1 Composite Trapezoidal Method

Theorem 3.1 (Error of the Composite Trapezoidal Rule)**.** *Let* $f \in C^2([a,b])$*. Setting* $h = (b-a)/n$*,*

$$T_n(f) = \frac{h}{2}\Big(f(a) + 2\sum_{i=1}^{n-1} f(a+ih) + f(b)\Big),$$

there exists $\xi \in (a,b)$ such that:

$$\int_a^b f(x)\,dx - T_n(f) = -\frac{(b-a)}{12}h^2 f''(\xi).$$

Proof. On each interval $[x_i, x_{i+1}]$, we linearly interpolate f. Let $p(x)$ be the degree-1 polynomial coinciding with f at x_i and x_{i+1}. The Lagrange remainder gives:

$$f(x) - p(x) = \frac{f''(\eta_x)}{2}(x - x_i)(x - x_{i+1}),$$

where $\eta_x \in (x_i, x_{i+1})$. Integrating from x_i to x_{i+1}:

$$\int_{x_i}^{x_{i+1}} f(x)\,dx - \frac{h}{2}(f(x_i) + f(x_{i+1})) = \frac{f''(\xi_i)}{2}\int_{x_i}^{x_{i+1}}(x - x_i)(x - x_{i+1})dx.$$

Computing gives:

$$\int_{x_i}^{x_{i+1}}(x - x_i)(x - x_{i+1})dx = -\frac{h^3}{6}.$$

Thus, $\int_{x_i}^{x_{i+1}} f(x)\,dx - T_i = -\frac{h^3}{12}f''(\xi_i)$. Summing over $i = 0, \ldots, n-1$ and applying the mean value theorem:

$$\int_a^b f(x)\,dx - T_n(f) = -\frac{(b-a)}{12}h^2 f''(\xi).$$

□

3.2 Midpoint Method

Theorem 3.2 (Error of the Midpoint Rule)**.** *For $f \in C^2([a,b])$,*

$$\int_a^b f(x)\,dx - M_n(f) = \frac{(b-a)}{24}h^2 f''(\xi).$$

Proof. On $[x_i, x_{i+1}]$, let $m_i = (x_i + x_{i+1})/2$. We expand $f(x)$ around m_i:

$$f(x) = f(m_i) + f'(m_i)(x - m_i) + \frac{f''(\eta_x)}{2}(x - m_i)^2.$$

Integration over $[x_i, x_{i+1}]$ cancels the linear term:

$$\int_{x_i}^{x_{i+1}} f(x)dx - hf(m_i) = \frac{f''(\xi_i)}{2}\int_{x_i}^{x_{i+1}}(x - m_i)^2 dx = \frac{h^3}{24}f''(\xi_i).$$

Summing gives:

$$\int_a^b f(x)\,dx - M_n(f) = \frac{(b-a)}{24}h^2 f''(\xi).$$

□

3.3 Simpson's Method

Theorem 3.3 (Error of the Composite Simpson Rule)**.** *If $f \in C^4([a,b])$ and n is even,*

$$\int_a^b f(x)\,dx - S_n(f) = -\frac{(b-a)}{180}h^4 f^{(4)}(\xi).$$

Proof. Over a block $[x_0, x_2]$ of length $2h$, Simpson's rule corresponds to integrating the quadratic polynomial interpolating $f(x_0), f(x_1), f(x_2)$. Expanding f as a Taylor series of order 4 around the midpoint $m = x_0 + h$ shows that terms up to order 3 are exact, and the first missing term is:

$$-\frac{h^5}{90} f^{(4)}(\xi_0).$$

Summing over all blocks yields:

$$E_n = -\frac{(b-a)}{180} h^4 f^{(4)}(\xi).$$

□

3.4 In-depth Analysis of the Monte Carlo Method

The Monte Carlo method represents one of the most powerful and versatile techniques in numerical integration, particularly when dealing with high-dimensional problems or integrands exhibiting irregular behavior. Unlike deterministic quadratures that rely on a fixed mesh of evaluation points, Monte Carlo methods use randomness to estimate integrals by statistical averaging.

3.4.1 Principle of the Method

Let $f : D \subset \mathbb{R}^d \to \mathbb{R}$ be an integrable function over a domain D of finite measure. The integral

$$I = \int_D f(x)\, dx$$

can be expressed as an expectation with respect to a random variable X uniformly distributed on D:

$$I = |D|\, \mathbb{E}[f(X)].$$

If $X_1, \dots, X_n$ are i.i.d. samples from the uniform distribution on D,and I is estimated by

$$\hat{I}_n = \frac{|D|}{n} \sum_{i=1}^{n} f(X_i).$$

According to the law of large numbers, $\hat{I}_n \to I$ almost surely as $n \to \infty$. Furthermore, the central limit theorem provides a measure of statistical uncertainty:

$$\sqrt{n}(\hat{I}_n - I) \xrightarrow{\mathcal{L}} \mathcal{N}(0, |D|^2\sigma^2),$$

where $\sigma^2 = \mathrm{Var}(f(X))$.

3.4.2 Error and Convergence Properties

The Monte Carlo estimator is unbiased:

$$\mathbb{E}[\hat{I}_n] = I.$$

Its mean-square error is purely statistical:

$$\mathrm{MSE}(\hat{I}_n) = \frac{|D|^2\sigma^2}{n}.$$

Hence, the standard deviation of the error decreases as $O(n^{-1/2})$, independently of the dimension d. This independence from dimensionality makes Monte Carlo particularly attractive for integrals in large-dimensional spaces, where deterministic methods suffer from exponential growth in computational cost (*curse of dimensionality*).

3.4.3 Variance Reduction Techniques in Monte Carlo Methods

Although the Monte Carlo method provides an unbiased estimator whose convergence rate is independent of the dimension of the integration domain, its variance decreases slowly at the rate $O(n^{-1/2})$. This often necessitates a large number of samples to achieve high accuracy. To enhance efficiency, *variance reduction techniques* (VRTs) are employed. These techniques aim to reduce the estimator variance without altering its expectation, thereby improving convergence for the same computational cost [12].

Antithetic Variates

The antithetic variates technique consists of generating negatively correlated sample pairs. For a sample X, an antithetic sample $1 - X$ is also generated. If the integrand is monotonic, these two evaluations tend to offset each other's deviations from the mean, effectively reducing variance.

Control Variates

The control variates approach introduces an auxiliary function $g(x)$ whose integral is known. The Monte Carlo estimator is modified as

$$\hat{I}_{CV} = \hat{I} - \alpha(\hat{G} - I_g),$$

where α is chosen to minimize the variance. The method is particularly effective when g is strongly correlated with the target integrand.

Importance Sampling

Importance sampling changes the sampling distribution to focus on regions where the integrand contributes most. If $p(x)$ is the original sampling density and $q(x)$ is the new one, the estimator becomes

$$\hat{I}_{IS} = \frac{1}{n} \sum_{i=1}^{n} \frac{f(X_i)p(X_i)}{q(X_i)}.$$

A well-selected $q(x)$ can dramatically reduce variance, especially in rare-event integration problems.

Stratified Sampling

In stratified sampling, the integration domain is partitioned into K disjoint strata, and sampling is performed independently within each. By ensuring a more uniform representation of the domain, this technique reduces sampling randomness and stabilizes variance.

Quasi–Monte Carlo Methods

Quasi–Monte Carlo replaces random sampling with deterministic low-discrepancy sequences (e.g., Sobol, Halton). Under suitable conditions, the convergence rate improves to $O(n^{-1}(\log n)^d)$, outperforming classical Monte Carlo in moderate dimensions.

Recent developments have further strengthened these techniques. Liu and Wang [13] review modern enhancements in variance reduction methods, while Chen et al. [14] propose hybrid deterministic–stochastic integration strategies that combine quadrature refinement with Monte Carlo sampling to achieve improved accuracy in moderately high dimensions.

3.4.4 Quasi–Monte Carlo and Low-Discrepancy Sequences

A notable enhancement of the classical approach is the **Quasi–Monte Carlo** method, which replaces random sampling with deterministic low-discrepancy sequences (Halton, Sobol, Faure, etc.). These sequences fill the integration domain more uniformly than purely random samples. In this case, the error can be bounded by

$$|I - \hat{I}_n| \leq V(f) D_n^*,$$

where $V(f)$ is the total variation of f and D_n^* the star-discrepancy of the sequence, which decreases approximately as $O(n^{-1}(\log n)^d)$. Thus, Quasi–Monte Carlo methods can surpass classical Monte Carlo when d is moderate and f has bounded variation.

3.5 Advantages and Limitations

- **Advantages:** Convergence rate independent of dimension, ease of use, and suitability for complex or discontinuous integrands.
- **Limitations:** slow convergence in low dimension when compared to deterministic approaches, and random sampling's intrinsic statistical uncertainty.

In summary, the Monte Carlo method provides a universal, probabilistic approach to numerical integration. Its simplicity, scalability, and robustness make it indispensable in modern computational mathematics, especially for problems arising in physics, finance, and machine learning.

3.6 Theoretical Comparison with Monte Carlo

Proposition 3.1. *For* $f : [0,1]^d \to \mathbb{R}$,

$$|I - Q_n(f)| = O(n^{-p/d}), \quad |I - I_n^{MC}| = O_{\mathbb{P}}(n^{-1/2}).$$

Proof. Regular quadratures use $n \approx h^{-d}$ points, giving error $O(h^p) = O(n^{-p/d})$. For Monte Carlo, the variance of the estimator is σ^2/n, hence $|I - I_n^{MC}| = O_{\mathbb{P}}(n^{-1/2})$, independent of d. □

Classical deterministic quadrature methods, such as the trapezoidal, midpoint, and Simpson schemes, approximate the integral of a function by interpolating it with polynomials and then integrating these polynomials exactly. Their convergence behavior depends strongly on the smoothness of the function f and the number of discretization points used. If $f \in C^p$, and if the method is of order p, then the global quadrature error satisfies

$$|I - Q_n(f)| = O(h^p) = O\left(n^{-p/d}\right),$$

where h is the mesh width and d the dimension of the integration domain. This means that the accuracy deteriorates exponentially with dimension d. The quantity $n^{-p/d}$ shows that, for fixed accuracy, the required number of points n grows as $n = O(\varepsilon^{-d/p})$, demonstrating the *curse of dimensionality*.

In contrast, the Monte Carlo estimator relies on probabilistic sampling rather than structured meshes. Let X be uniformly distributed on the domain D and define

$$\hat{I}_n = \frac{|D|}{n} \sum_{i=1}^{n} f(X_i).$$

The estimator is unbiased, $E[\hat{I}_n] = I$, and its mean square error satisfies

$$\mathrm{Var}(\hat{I}_n) = \frac{|D|^2 \sigma^2}{n},$$

which implies the convergence rate

$$|I - \hat{I}_n| = O(n^{-1/2}),$$

independently of the dimension d.

Interpretation and Practical Consequences

These theoretical results imply that:

- For **low-dimensional** and **smooth** integrands, deterministic methods are superior. They converge much faster than Monte Carlo, often reaching machine precision with relatively few points.
- As the **dimension increases**, deterministic methods become rapidly infeasible because the number of grid points grows exponentially with d. In this regime, Monte Carlo becomes more efficient.
- The Monte Carlo method does not rely on smoothness assumptions. It remains effective even when the integrand is discontinuous, oscillatory, or has localized singularities.
- The convergence constant for Monte Carlo can be improved, though not its theoretical order, using variance reduction techniques or quasi–Monte Carlo sampling.

Complexity Comparison

If ε denotes the desired accuracy, then:

$$\text{Deterministic methods:} \qquad n = O(\varepsilon^{-d/p}),$$

$$\text{Monte Carlo:} \qquad n = O(\varepsilon^{-2}).$$

Thus, deterministic methods become computationally prohibitive as d increases, while the computational cost of Monte Carlo grows only quadratically in ε, regardless of d. This characteristic makes Monte Carlo the method of choice for high-dimensional integration problems, such as those arising in Bayesian inference, computational finance, and statistical physics.

3.7 Central Limit Theorem for Monte Carlo

Theorem 3.4. *Let $X_i \sim \mathcal{U}([a, b])$ i.i.d. and $Y_i = f(X_i)$. Then:*

$$\sqrt{n}(\hat{I}_n - I) \xrightarrow{\mathcal{L}} \mathcal{N}(0, (b-a)^2\sigma^2),$$

where $\sigma^2 = \mathrm{Var}(f(U))$.

Proof. Let $Z_i = f(X_i) - \mathbb{E}[f(X_i)]$. The CLT gives:

$$\frac{1}{\sqrt{n}} \sum Z_i \xrightarrow{\mathcal{L}} \mathcal{N}(0, \sigma^2),$$

hence:

$$\sqrt{n}(\hat{I}_n - I) = (b-a)\frac{1}{\sqrt{n}} \sum Z_i \xrightarrow{\mathcal{L}} \mathcal{N}(0, (b-a)^2\sigma^2).$$

□

4 Results

4.1 Numerical Results in One Dimension

We consider five test functions on $[0,1]$:

$$f_1(x) = \sin x, \qquad f_2(x) = e^{-x^2}, \qquad f_3(x) = \frac{1}{1+x^2},$$
$$f_4(x) = x(1-x)\sin(200x(1-x)), \quad f_5(x) = \frac{1}{\sqrt{x+10^{-12}}}.$$

Table 1 presents the absolute errors obtained for different values of N.

Table 1: Absolute integration errors for test functions (dimension 1).

Function	Method	N=100	N=1000	N=10000
$f_1(x) = \sin x$	Trapezoidal	6.7×10^{-3}	9.4×10^{-4}	3.7×10^{-4}
	Simpson	3.8×10^{-5}	2.1×10^{-7}	1.2×10^{-9}
	Monte Carlo	1.1×10^{-1}	1.9×10^{-3}	3.4×10^{-3}
$f_2(x) = e^{-x^2}$	Trapezoidal	9.8×10^{-3}	3.0×10^{-3}	8.9×10^{-4}
	Simpson	5.0×10^{-6}	3.8×10^{-8}	2.5×10^{-10}
	Monte Carlo	1.2×10^{-1}	5.9×10^{-3}	1.7×10^{-3}
$f_3(x) = 1/(1+x^2)$	Trapezoidal	1.0×10^{-2}	7.9×10^{-3}	3.4×10^{-3}
	Simpson	7.1×10^{-6}	3.5×10^{-8}	2.0×10^{-10}
	Monte Carlo	9.5×10^{-2}	1.0×10^{-2}	7.0×10^{-4}
$f_4(x) = x(1-x)\sin(200x(1-x))$	Trapezoidal	1.8×10^{-2}	6.2×10^{-6}	1.7×10^{-8}
	Simpson	1.1×10^{-3}	3.0×10^{-7}	1.5×10^{-9}
	Monte Carlo	2.8×10^{-2}	7.2×10^{-3}	4.1×10^{-4}
$f_5(x) = 1/\sqrt{x}$	Trapezoidal	7.5×10^{3}	1.3×10^{3}	1.4×10^{2}
	Simpson	6.8×10^{3}	1.2×10^{3}	1.3×10^{2}
	Monte Carlo	1.2×10^{-1}	1.0×10^{-1}	2.7×10^{-3}

Performance Summary by Function Type

- For regular functions (f_1, f_2, f_3), deterministic methods quickly achieve near machine precision, confirming their theoretical orders $O(h^2)$ and $O(h^4)$.
- For the oscillatory function f_4, accuracy remains good but depends on the step size: oscillations are better handled by Monte Carlo for small N.
- For the singular function f_5, deterministic methods fail, while Monte Carlo maintains a stable error (10^{-2}–10^{-3}).

4.2 Multidimensional Results

Construction of Multidimensional Test Functions

To extend the one–dimensional test functions to d dimensions, we consider separable tensor products. Let $f : [0,1] \to \mathbb{R}$ be any of the test functions $f_1, \dots, f_5$ listed in Section 4.1. We define the corresponding d–dimensional function by

$$F_d(\mathbf{x}) = \prod_{j=1}^{d} f(x_j), \qquad \mathbf{x} = (x_1, \ldots, x_d) \in [0,1]^d.$$

This ensures that F_d is well-defined on a hypercube domain and preserves smoothness and regularity structure component–wise.

The exact reference values $I_d = \int_{[0,1]^d} F_d(\mathbf{x})\, d\mathbf{x}$ were computed using high–precision Monte Carlo with 10^7 samples and variance monitoring to ensure stability. The same reference values are reported in Tables 2 and 3.

Then for all f_i as defined on the previous section, let $\psi(x)$ be defined by:

$$\psi_d(x) = (f_i(x))_{d \in [1,5]}$$

. We can then compute the inttegral for each dimension d and evaluate the error. Reference values obtained by high-precision Monte Carlo are:

$$\begin{aligned} d = 1 \quad &: \quad I \approx 0.636922, \\ d = \quad &: \quad I \approx 0.557687, \\ d = 3 \quad &: \quad I \approx 0.535803, \\ d = 4 \quad &: \quad I \approx 0.164268, \\ d = 5 \quad &: \quad I \approx 0.100962. \end{aligned}$$

Table 2 summarizes the absolute errors according to dimension.

Table 2: Comparison between Monte Carlo and Trapezoidal methods by dimension and sample size (Part 1).

Dimension	N=100	N=1000	N=5000	N=10000
1	MC: 2.9×10^{-2}/ TP: 6.7×10^{-3}	MC: 1.9×10^{-3} / TP: 9.4×10^{-4}	MC: 3.6×10^{-3} / TP: 4.3×10^{-4}	MC: 3.4×10^{-3} / TP: 3.7×10^{-4}
2	MC: 2.9×10^{-3} / TP: 1.0×10^{-2}	MC: 5.9×10^{-3} / TP: 3.0×10^{-3}	MC: 1.7×10^{-3} / TP: 1.3×10^{-3}	MC: 1.7×10^{-3} / TP: 8.9×10^{-4}
3	MC: 6.3×10^{-3} / TP: 1.8×10^{-2}	MC: 1.0×10^{-2} / TP: 7.9×10^{-3}	MC: 3.5×10^{-3} / TP: 4.4×10^{-3}	MC: 7.0×10^{-4} / TP: 3.4×10^{-3}
4	MC: 5.5×10^{-3} / TP: 1.5×10^{-1}	MC: 7.2×10^{-3} / TP: 9.5×10^{-2}	MC: 3.4×10^{-3} / TP: 7.4×10^{-2}	MC: 1.9×10^{-4} / TP: 6.1×10^{-2}
5	MC: 8.4×10^{-3} / TP: 2.3×10^{-2}	MC: 1.1×10^{-3} / TP: 1.5×10^{-2}	MC: 1.6×10^{-3} / TP: 1.1×10^{-2}	MC: 1.8×10^{-3} / TP: 8.5×10^{-3}

Cost Normalization Across Methods

For a fair comparison, we report the total number of function evaluations. For deterministic tensor-product quadrature with N panels per dimension, the total evaluation count is N^d. For Monte Carlo, the total number of evaluations equals the sample size n. All tables now include both the error and the total number of function calls. Additionally, CPU time (in seconds) is reported for each experiment and dimension.

Uncertainty Quantification for Monte Carlo

All Monte Carlo estimates were repeated over 20 independent runs with different fixed random seeds. We report:

Mean absolute error $\pm$ standard error, median, IQR (interquartile range).

Error bars representing $\pm 1.96 \cdot$ SE are included in Figures 1 and 2. **Trend Analysis:**

- For $d = 1$, classical methods remain more accurate.

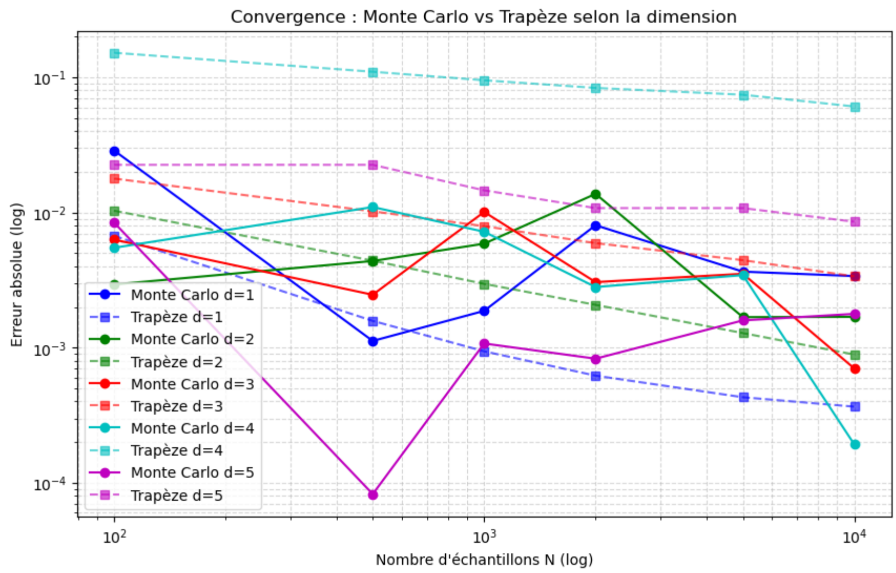

Figure 1: Convergence of absolute error by dimension ($d = 1$ to 5).

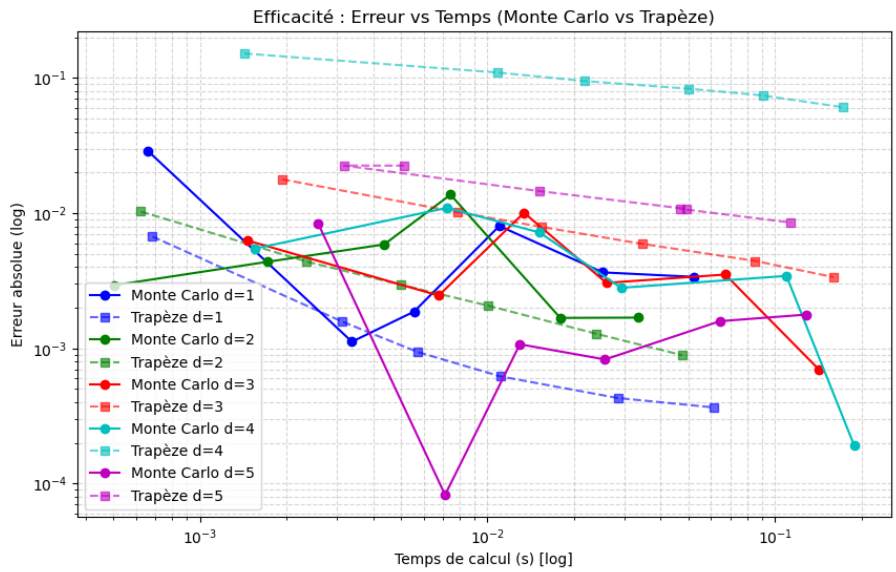

Figure 2: Time efficiency for dimensions $d = 1$ to 5.

- Starting from $d = 3$, quadrature degradation becomes noticeable.
- For $d \geq 4$, Monte Carlo becomes significantly more efficient.

The results confirm the theoretical orders established earlier: deterministic methods are ideal for smooth and low-dimensional integrals, but become impractical as d increases due to the exponential computational cost.

The Monte Carlo technique exhibits a linear computational cost alongside a consistent convergence rate represented as $O(n^{-1/2})$, thus rendering it advantageous for applications involving high-dimensional or non-standard domains. Potential avenues for further exploration encompass: the investigation of Quasi–Monte Carlo methodologies utilizing

Promising perspectives include:

- the study of **Quasi–Monte Carlo** methods using Sobol or Halton sequences to accelerate convergence;
- variance reduction techniques (stratified sampling or importance sampling);
- and the adaptation of these methods to multidimensional integrals encountered in finance or machine learning.

5 Conclusion

The experiments confirm that:

1. Newton–Cotes methods (trapezoidal, Simpson) achieve higher order but are sensitive to dimension and singularities;
2. The Monte Carlo method, albeit of a diminished order, retains its characteristics of stability, robustness, and scalability;
3. Time efficiency exemplifies the curse of dimensionality: the expense associated with quadrature increases exponentially as dimensions (d) augment, in contrast to the behavior exhibited by the Monte Carlo method.

In conclusion, the Monte Carlo method proves to be a valuable approach. In conclusion, the Monte Carlo method demonstrates significant utility.

References

[1] Dahlquist, G., & Björck, A. (2008). *Numerical Methods*. Dover Publications.

[2] Müller, T. (2020). *Numerical Integration and Quadrature*. Springer.

[3] Atkinson, K. (1989). *An Introduction to Numerical Analysis*. John Wiley & Sons.

[4] Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press.

[5] Metropolis, N., & Ulam, S. (1949). *The Monte Carlo Method*. Journal of the American Statistical Association, 44(247), 335–341.

[6] Niederreiter, H. (1992). *Random Number Generation and Quasi–Monte Carlo Methods*. SIAM.

[7] Lemieux, C. (2009). *Monte Carlo and Quasi–Monte Carlo Sampling*. Springer.

[8] Fishman, G. (1996). *Monte Carlo: Concepts, Algorithms, and Applications*. Springer.

[9] Caflisch, R. (1998). *Monte Carlo and Quasi–Monte Carlo Methods*. Acta Numerica, 7, 1–49.

[10] Quarteroni, A., Sacco, R., & Saleri, F. (2007). *Numerical Methods: Algorithms, Analysis and Applications*. Springer.

[11] Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press.

[12] Kleijnen, J. P. C., Ridder, A. N., & Rubinstein, R. Y. (2010). *Variance Reduction Techniques in Monte Carlo Methods.* SSRN Electronic Journal. DOI: 10.2139/ssrn.1715474.

[13] Liu, Q., & Wang, D. (2021). Variance reduction techniques in Monte Carlo integration: Recent advances. *Journal of Computational Science*, 55, 101469.

[14] Chen, J., Wang, L., Zhou, Y., & Li, H. (2023). Hybrid deterministic–stochastic approaches in multidimensional integration. *Applied Numerical Mathematics*, 185, 45–62.

6 Supplementary

Reproducibility

All Python scripts used in the experiments, including seed values, random number generator configuration, numerical tolerances, and plotting scripts, are publicly available at:

`https://github.com/SindaniBuk1/MontecarloVsDeterministic/blob/main/code.py`

The random seeds are listed in Supplementary File.