Hackathon 2025

proposed by:

1: Introduction to the exercise: Tuning an "artificial" Monte Carlo code in finance

2: Monte Carlo methods: A universal logic for solving complex problems

3: Black-Scholes method

Introduction to the exercise: Tuning an "artificial" Monte Carlo code in finance In this exercise, you will work on optimizing a Monte Carlo code applied to finance. Although this code simulates a typical option valuation strategy, its purpose is to familiarize you with general Monte Carlo simulation techniques, widely used in many fields, such as physics, biology, engineering, and, of course, finance.

The goal of this exercise is not to solve a mathematical problem or to compute the exact price of a financial product. Instead, it is an exercise in code optimization on a specific hardware architecture. Your task will be to tune and optimize this code to improve its performance, taking into account the underlying architecture. You will manipulate several aspects of programming, such as parallelism, efficient use of processor (CPU) resources, and vectorization techniques.

Why is this exercise important? Monte Carlo methods are powerful but computationally expensive. In finance, these methods are used to simulate thousands or even millions of stochastic paths to value options or other derivatives. Typically, these calculations need to be performed quickly and efficiently, especially when working with high-performance computing (HPC) architectures or embedded systems.

However, this simulation technique applies to many other fields. What you will learn here is how to improve the performance of a Monte Carlo code on a given architecture. The optimization strategies you will explore will be applicable in many situations, whether in numerical physics, computational biology, or complex systems engineering.

Tips and recommendations Here are a few points to keep in mind to succeed in this exercise:

1. **Focus on optimization, not mathematical precision**:
   - This exercise is not an assessment of your ability to solve a mathematical problem or to develop a sophisticated financial model. The model you will use is artificial and intentionally simplified. Your goal is to obtain fast and efficient code, maximizing the use of hardware resources.
2. **Use parallelism techniques**:
   - Monte Carlo calculations are highly parallelizable. Explore techniques such as threading, SIMD (Single Instruction Multiple Data) parallelism, and take advantage of all the capabilities of your hardware architecture (multi-core CPUs, GPUs, etc.).

3. **Adopt vectorization**:
   - If you are working on an architecture that supports vectorization (like NEON for ARM), implement vectorized operations to speed up calculations. Monte Carlo is an ideal candidate for vectorization, as repetitive calculations on independent data sets can be parallelized.
4. **Minimize bottlenecks**:
   - Identify sections of the code that slow down execution (use profiling tools). These bottlenecks may include costly function calls, frequent memory reads/writes, or poorly optimized loops.
5. **Random number management**:
   - A critical point in Monte Carlo is random number generation. The random number generator can become a bottleneck. Consider using optimized generators and avoid shared concurrent access when dealing with parallel computations.
6. **Pay attention to the hardware architecture**:
   - Each architecture has its peculiarities. For example, cache memory management, SIMD register width, and arithmetic operation latencies may vary from one processor to another. Optimize your code based on the specifics of the architecture you are working on.
7. **Measure your performance gains**:
   - After each optimization, measure the execution time and check that your changes indeed bring improvements. Ensure that optimization does not harm the expected calculation accuracy, and compare it to the non-optimized version.
8. **Use Google or ChatGPT for help**:
   - This exercise is also a strategy and knowledge enhancement task. You don't need to reinvent everything, but knowing how to leverage modern tools like Google or ChatGPT can help you solve difficult problems in constrained timeframes. The ability to search for solutions effectively is a key skill in optimizing code in a rapidly evolving environment.

Conclusion What you will accomplish here is directly applicable to real-world situations, where code optimization is crucial for reducing computation times and increasing productivity. Whether for complex financial simulations or physical simulations, the tuning techniques you will learn will help you write more efficient codes, capable of taking advantage of modern architectures. Remember: This is not a math problem, but rather a code optimization challenge on a particular architecture.

Monte Carlo methods: A universal logic for solving complex problems Monte Carlo methods form a powerful family of statistical algorithms, used to solve complex problems that would be difficult to tackle using classical analytical methods. Despite their diversity, all Monte Carlo codes follow the same fundamental logic and are massively used in many scientific fields, from finance to physics, biology, and engineering.

1. **Fundamental logic of Monte Carlo methods** The logic behind the Monte Carlo method is based on random simulation to solve problems that involve uncertainty or stochastic (random) phenomena. The main idea is to model a complex problem as a random process, then simulate a large number of possible outcomes to obtain an approximate statistical solution.

The method generally follows these steps:

1. **Problem modeling**: The phenomenon of interest is represented using random variables that follow probabilistic distributions.
2. **Simulations**: A large number of random scenarios (samples) are generated using a random number generator.
3. **Result calculation**: For each sample, a result is computed based on rules defined by the problem (e.g., the gain of a financial option or the energy of a particle).
4. **Statistical averaging**: The average of the results is calculated to obtain an overall estimate.
5. **Precision improvement**: The more simulations are performed, the more precise the estimate becomes, thanks to the law of large numbers.

This structure is shared by all Monte Carlo algorithms, whether in financial simulations, particle physics modeling, or other fields.

2. **Applications in various scientific fields** Monte Carlo methods are used in almost all scientific disciplines. Here are some notable examples.

a) **Finance I**n finance, Monte Carlo methods are widely used for the valuation of financial options and other derivative instruments. The Black-Scholes model (as mentioned earlier) can be solved using Monte Carlo when the model's assumptions are not fully met, or when dealing with more complex financial products, such as American or exotic options.

The logic here is to simulate the stochastic evolution of financial asset prices (often modeled by Wiener processes or Brownian motion) and calculate the associated gains or losses.

b) **Physics I**n physics, Monte Carlo methods are ubiquitous, particularly in particle physics, nuclear physics, and statistical mechanics. For example, they are used to simulate particle trajectories in accelerators, model nuclear reactions, or calculate thermodynamic properties in complex systems.

A famous example is the Metropolis-Hastings method, a Monte Carlo variant used in simulations of physical systems like spin networks in Ising-type models, where it helps compute the most probable energy states of a system.

c) **Biology and Medicine** In biology and medicine, Monte Carlo methods are used to model uncertain biological processes, such as population evolution or disease spread. They are also used in radiotherapy to calculate the effect of radiation on human tissues by simulating particle interactions with cells.

In clinical trials, Monte Carlo helps evaluate the effectiveness of treatments by simulating various patient response scenarios to medications.

d) **Chemistry and Materials Science** In quantum chemistry and materials science, Monte Carlo is used to simulate the behavior of electrons, molecules, or crystals at the atomic scale. This allows for the prediction of properties like electronic structure, binding energy, or chemical reactions under different conditions.

Quantum Monte Carlo is a key method in this field, where it helps solve problems that require probabilistic modeling of interactions between quantum particles.

e) **Engineering and Robotics** In engineering, Monte Carlo is used to analyze complex systems subject to uncertainties, for example in structural mechanics to predict material strength under various random loads. In robotics, Monte Carlo simulations help model sensor uncertainty and improve decision-making algorithms for autonomous navigation.

---

3. **Why is Monte Carlo so effective?** Monte Carlo methods are so widely used because they are:

- **Flexible**: They can be applied to almost any type of system, from financial models to physical phenomena.
- **Scalable**: The more simulations are performed, the more accurate the results. Thanks to modern technologies such as parallel computing and powerful processors, it is possible to run millions or even billions of simulations.
- **Non-restrictive**: Unlike analytical methods, which often require simplifications or strong assumptions, Monte Carlo can handle very complex systems with minimal simplifications.

4. **Conclusion** Monte Carlo methods are ubiquitous in the scientific world. Their logic is based on simple principles of random simulation and statistics, but they are powerful enough to solve some of the most complex problems faced by researchers and engineers today. Their ability to handle uncertainty and provide precise approximate solutions makes them an indispensable tool in almost all branches of science.

---

1. **Introduction to the Black-Scholes method**
2. **Black-Scholes model for a European option**
3. **Underlying assumptions**
4. **Mathematical formulation**
5. **Explanation of key terms**
6. **Black-Scholes differential equation**
7. **Monte Carlo approach for the Black-Scholes method**

---

1. **Introduction to the Black-Scholes method** The Black-Scholes model (or Black-Scholes-Merton) was developed in 1973 by Fischer Black, Myron Scholes, and Robert Merton. It is designed to assess the price of European options, which are options that can only be exercised on their expiration date.

An option is a financial contract that gives the right (but not the obligation) to buy or sell an asset (such as a stock) at a set price (called the strike price) at a certain date (called the expiration date). The Black-Scholes model provides a formula to calculate the value of a call option (right to buy) and a put option (right to sell).

---

2. **Black-Scholes model for a European option** The Black-Scholes model is used to determine the price of a European Call option, which gives the right to buy the underlying asset at a set price at maturity. The Black-Scholes formula for a European Call option is:

$$C(S_0, K, T, r, \sigma) = S_0 N(d_1) - K e^{-rT} N(d_2)$$

where:

- S0 is the initial price of the underlying asset.
- K is the strike price of the option.
- T is the time to maturity.
- r is the risk-free interest rate.
- σ is the volatility of the underlying asset.
- N(d1)) and N(d2) are the cumulative distribution function values of the standard normal distribution.
- d1 and d2 are defined by:

$$d_1 = \frac{\ln(S_0/K) + (r + 0.5\sigma^2)T}{\sigma\sqrt{T}}$$

$$d_2 = d_1 - \sigma\sqrt{T}$$

---

3. **Underlying assumptions of the Black-Scholes model** The model is based on several assumptions:
4. **Frictionless market**: There are no transaction fees or taxes, and buying or selling can be done without restriction.
5. **No arbitrage**: It is impossible to make a risk-free profit using a combination of assets.
6. **Random walk model**: The price of the underlying asset follows a diffusion process called geometric Brownian motion (or Wiener process).
7. **Constant interest rate**: The risk-free interest rate remains constant during the option's lifetime.
8. **Constant volatility**: The volatility σ of the underlying asset remains constant during the option's lifetime.
9. **European option**: The option can only be exercised on the expiration date.

---

4. **Mathematical formulation** The Black-Scholes model is based on the idea that the price of an asset follows a stochastic evolution, modeled by geometric Brownian motion. This means that the price of the asset evolves randomly, but with a certain trend related to the risk-free interest rate and volatility.

The stochastic differential equation describing the price evolution of the asset S(t) is:

$$dS(t) = S(t)(\mu dt + \sigma dW(t))$$

where:

- S(t) is the price of the asset at time t.

- μ is the expected rate of return of the asset.
- σ is the price volatility of the asset.
- W(t) is a Wiener process or Brownian motion.

---

5. **Explanation of key terms**
6. **Volatility (σ)**: Volatility measures the fluctuation of the underlying asset's price. High volatility means that the asset's price can vary widely, while low volatility indicates more stable fluctuations.
7. **Risk-free rate (r)**: The risk-free interest rate is usually the rate of government bonds, considered risk-free. This rate allows comparing the return on a risky investment (like an option) to a safe investment.
8. **Cumulative normal distribution function (N(d1),N(d2))**: N(d) represents the probability that the asset's price will be above the strike price at expiration. This reflects the market's uncertainty about the asset's future price.
9. **Payoff of a Call option**: The gain realized by exercising a Call option is max(ST−K,0), where ST is the asset price at expiration, and K is the strike price. If ST>K, the option is "in the money" and generates a profit. Otherwise, the option expires worthless.

---

6. **Black-Scholes differential equation:** the key to the Black-Scholes model is solving a partial differential equation (PDE) that describes the option's price dynamics. This equation is:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

Where V(S,t) is the option's value as a function of the asset price S and time t. This equation can be solved to derive the Black-Scholes formula.

---

7. **Monte Carlo approach for the Black-Scholes method:** In addition to the analytical solution, the Monte Carlo method is commonly used to evaluate option prices, especially when the analytical solution is difficult to obtain, or the Black-Scholes assumptions are not met (for example, for American options).

Monte Carlo method:

1. **Generating simulated paths**: The Monte Carlo method involves simulating many possible trajectories for the price of the underlying asset using stochastic dynamics (geometric Brownian motion).
2. **Calculating the payoff**: For each simulated path, the payoff of the option at maturity is calculated (e.g., max(ST−K,0) for a Call option).
3. **Averaging the payoffs**: The option's value is obtained by averaging the payoffs across all simulations and discounting this value using the risk-free rate r.
4. **General formula**:

$$C_{\mathrm{MC}} = e^{-rT} \times \frac{1}{N} \sum_{i=1}^{N} \max(S_T^{(i)} - K, 0)$$

Where N is the number of simulations, and ST(i) is the simulated price of the underlying asset at expiration for the i-th simulation.

Conclusion The Black-Scholes model is a powerful and widely used method for option pricing, but it relies on simplifying assumptions. The Monte Carlo approach, on the other hand, is more flexible and allows for the modeling of more complex scenarios, though it is more computationally expensive.