Joni Kaipainen

Path Integral Calculations for Option Pricing

School of Accounting and Finance

Master's thesis in Economics

Master's Degree Programme in Economics

Vaasa 2023

VAASAN YLIOPISTO
School of Accounting and Finance
Author: Joni Kaipainen
Thesis title: Path Integral Calculations for Option Pricing
Degree: Master of Science in Economics and Business Administration
Supervisor: Jaana Rahko
Year of graduation: 2023 **Number of pages:** 72

ABSTRACT:
Since the initiation of options trading by the Chicago Board Options Exchange in 1973, the financial markets have experienced substantial growth in options trading. As of 2022, the trading volume reached an astounding 10.32 billion contracts, with the gross market value of over-the-counter derivatives, including options, amounting to $20.7 trillion and a notional value of $618 trillion. This growth underscores the critical role of options trading in modern finance.

The pricing of options is a highly mathematical task, influenced by multiple factors such as asset volatility, time until expiration, interest rates, and market unpredictability. Accurate pricing is essential not only for profit maximization but also for mitigating systemic risks, as evidenced by the 2007-2008 financial crisis where mispriced mortgage derivatives played a significant role. Consequently, there's an increasing demand for more detailed and computationally efficient pricing methodologies.

This study explores the application of the quantum mechanical path integral method introduced by R. Feynman to option pricing. This approach combines the probabilistic foundations of quantum mechanics with financial modeling. Traditionally used in physics to calculate particle transition probabilities with astonishing accuracy, path integrals offer also a method to model the paths of asset prices as a function of time. Numerical integration of path integrals with Monte Carlo simulations provides an interesting multidisciplinary method for simulating complex processes inherent in financial markets.

A significant aspect of this research is in the comparison of the quantum mechanical path integral Monte Carlo simulation framework with the traditional option pricing methods. The results indicate that the path integral formalism can replicate well-known results, and can be easily extended to valuate more complicated options. Furthermore, the results of this research clarify the quantum mechanical aspects of option pricing and present both the theoretical framework and efficient numerical solutions in a comprehensible manner. Through this, the study aims to contribute to the advancement of financial modeling and risk management strategies, marking a step forward in the intersection of quantum physics and financial economics.

Keywords: financial instruments, derivatives, options, quantum theory, quantum mechanics

VAASAN YLIOPISTO
Laskentatoimen ja rahoituksen yksikkö
Tekijä: Joni Kaipainen
Tutkielman nimi: Path Integral Calculations for Option Pricing
Tutkinto: Kauppatieteiden maisteri
Työn ohjaaja: Jaana Rahko
Valmistumisvuosi: 2023 **Sivumäärä:** 72

TIIVISTELMÄ:
Optiokaupankäynnin osuus finanssimarkkoinoilla on kasvanut merkittävästi Chicago Board Options Exchangen aloitettua kaupankäynnin optioilla vuonna 1973. Vuonna 2022 optioilla tehtyjen kauppojen määrä ylitti 10,32 miljardia. Näiden johdannaiskauppojen bruttoarvo ylitti 20,7 biljoonaa ja nimellisarvo 618 biljoonaa dollaria. Tämä kasvu korostaa optiokaupan merkistystä nykyaikaisilla rahoitusmarkkinoilla.

Optioiden hinnoittelu on matemaattisesti haastava tehtävä, johon vaikuttaa monta tekijää, kuten volatiliteetti, voimassaoloaika, korot sekä markkinoiden arvaamaton luonne. Tarkka hinnoittelu on voittojen maksimoinnin lisäksi tärkeää järjestelmäriskien välttämiseksi, kuten opittiin vuosien 2007-2008 finanssikriisistä, jossa väärin hinnoitellut asuntolainajohdannaiset olivat merkittävässä roolissa. Näin ollen kysyntää entistä tarkemmille ja laskennallisesti tehokkaammille optioiden hinnoittelumenetelmille on runsaasti.

Tässä tutkielmassa tarkastellaan R. Feynmanin esittämän kvanttimekaanisen polkuintegraalimenetelmän soveltamista optioiden hinnoitteluun. Menetelmä yhdistää kvanttimekaniikan probabilistiset lähtökohdat ja rahoitusteorian. Polkuintegraalimenetelmää on käytetty menestyksekkäästi hiukkasten tilasiirtymien laskemiseen, minkä lisäksi menetelmää voidaan soveltaa mallintamaan rahoitusintrumenttien arvon muutosta ajan funktiona. Polkuintegraalien numeerinen ratkaiseminen Monte Carlo -simulaatioita käyttämällä luo vahvan pohjan poikkitieteelliselle menetelmälle rahoitusmarkkinoiden monimutkaisten prosessien simuloimiseksi.

Tutkielmassa vertaillaan kvanttimekaanista polkuintegraali-Monte Carlo -menetelmää perinteisiin optioiden hinnoittelussa käytettyihin menetelmiin. Tuloksista nähdään, että polkuintegraalimenetelmällä voidaan päästä samoihin tuloksiin tunnettujen mallien kanssa, ja että menetelmää voidaan soveltaa hyvin monimutkaisempien optioiden hinnoittelussa. Työn tarkoituksena on selventää menetelmän kvanttimekaanisista luonnetta ja esitellä taustalla oleva teoreettinen viitekehys, sekä selkeä ja tehokas menetelmä numeerisen ratkaisun mahdollistamiseksi. Tutkielman tavoitteena on edistää rahoitusmallinnuksen ja riskienhallinnan menetelmiä, ja vahvistaa kvanttimekaniikan ja taloustieteen välisiä yhteneväisyyksiä.

Avainsanat: rahoitusinstrumentit, johdannaiset, optiot, kvanttiteoria, kvanttimekaniikka

Contents

Figures

Tables

1 Introduction

Since the Chicago Board Options Exchange started trading call options in 1973, financial markets have witnessed a meteoric rise in the trading of options (Hull, 2021). According to Saito-Chung (2023), in the first year of options trading in the U.S., a total of 1.12 million options contracts were made, and in 2022, a staggering 10.32 billion contracts. Options grant the right but not the obligation to buy or sell an asset, such as a stock, at a predetermined price, in a specified time frame. These financial instruments have become central to the strategic planning of both individual investors and large institutions. According to a report published by the Bank for International Settlements (*OTC derivatives statistics at end-December 2022*, 2023), the sheer gross market value of traded over-the-counter derivatives, including options, stood at a staggering $20.7 trillion in 2022 with the notional value as high as $618 trillion. Finally, according to (Maverick, 2022), the total market size of derivatives is sometimes estimated to be over 10 times the size of the total global world domestic product (GDP). These figures not only underscore the importance of options in modern finance but also highlight the immense complexity and dynamism that characterize today's financial landscapes.

Option pricing is a nuanced and complicated mathematical task that depends on several key factors: the volatility of the underlying asset, the time remaining until expiration, current interest rates, and the unpredictability of market events. These elements interact in complex ways, necessitating the use of advanced mathematical models that can navigate the probabilistic nature of financial markets. The objective of such models is to accurately reflect the risks and potential rewards embedded in options, thereby informing investment decisions and risk management. Given the complexity of these variables, option pricing represents a sophisticated blend of analytical precision and market insight. (Hull, 2021)

The difficulty in pricing options is further compounded by the need for timeliness and precision. Inaccurate pricing not only eats away possible profits but can also lead to sys-

temic risks, as seen in various financial crises. For example, it has been suggested that in the 2007-2008 financial crisis, mispriced mortgage derivatives played a catalytic role (Greenberger, 2010). It is therefore crucial to employ analytical models that can navigate the probabilistic nature of markets, accommodate the multifaceted determinants of option pricing, and provide accurate valuations in real-time. This need has propelled the search for more detailed and computationally efficient pricing methodologies, which is the overarching objective of this work.

Quantum theory, with its roots firmly planted in the domain of physical sciences, has emerged as an unexpected ally in the quest to unravel the complexities of financial markets (Kleinert, 2009). The success of quantum theory in explaining phenomena at the microscopic level has inspired financial theorists, economists, as well as physicists to explore its potential in economic modeling. The theory's probabilistic foundation and its approach to dealing with systems of many particles make it a tempting choice for modeling the stochastic random nature of financial markets.

One of the most promising intersections of quantum theory and finance is the application of quantum mechanical path integral methods (Feynman, 1948; Kleinert, 2009). Traditionally employed in physics to compute the probabilities of particle transitions, path integrals offer a way to model the probabilistic pathways of asset prices. The idea is to consider the infinitely many paths that an asset price could take over time, much like visualizing the various trajectories a particle might travel through space.

The union of quantum path integrals and Monte Carlo simulations (Thomopoulos, 2013) represents a cutting-edge approach to option pricing. Monte Carlo simulations are a cornerstone in the financial industry, providing a way to model the probability distributions of potential outcomes by generating a multitude of random samples (Hull, 2021). Accordingly, the integration of quantum path integrals with Monte Carlo methods offers a novel and fruitful interdisciplinary way to simulate the more complex processes inherent in financial markets.

A critical factor that has made such sophisticated financial simulation approaches feasible is the remarkable growth in computing power, especially via parallel computing and the rapid development of Graphics Processing Units (GPUs) (Dehal, Munjal, Ansari, & Kushwaha, 2018; Owens et al., 2008). Originally designed for rendering computer graphics, GPUs have found a new purpose in high-performance computing applications, thanks to their ability to perform multiple calculations simultaneously.

In the modern era of computer simulations, parallel computing has unlocked the potential for significant performance boosts (Kessler & Keller, 2007). This work also explores the implementation of quantum mechanical path integral Monte Carlo simulations (Thomopoulos, 2013) on both parallel Central Processing Unit (CPU), and GPU architectures by constructing a simple Python framework. By utilizing modern-day parallel processing capabilities, this work demonstrates a substantial acceleration in computational speed, making the path integral method more feasible to use in practice than previously.

Prior research on financial engineering and mathematics has presented numerous option pricing models. However, the pricing of complex options is lacking coherent and generally used industry standard methods. The proposed use of quantum mechanical path integrals in option pricing, together with computationally efficient techniques results in the work having two research questions:

1. *How can the quantum mechanical path integral method be adapted and applied to option pricing in financial markets?*

2. *What computational methods can be implemented to enhance the efficiency of quantum mechanical path integral Monte Carlo simulations in option pricing?*

The first research question explores the theoretical and numerical foundations of the method: Firstly, it elaborates on the adaptability of the quantum mechanical framework to the modeling of financial markets. Secondly, it addresses the applicability of the method for option pricing in practice. The second research question, however, has more

practical implications in trading as the computational cost is currently a limiting factor in utilizing the more refined financial modeling techniques.

This work consists of five sections: introduction, theory, methodology, results, and conclusions. After this introduction, the work continues with a theory section describing the fundamentals of options. It provides an overview of different types of options, as well as describes the underlying principles that govern their trading and valuation. Additionally, traditionally used option pricing methods are discussed in detail. Finally, the path integral formulation is introduced together with basic principles of quantum mechanics to demystify the complex mathematical notation. In the third section, the methodology used in numerical simulations is presented, starting with a friendly introduction to Monte Carlo simulations, after which the simulation procedure is described. Lastly, the performance-enhancing techniques, namely the use of parallel computing and GPU acceleration, are discussed. In the results section first the achieved performance increases are presented. After that, the simulation procedure is verified by using the software to price European options, after which, the software is used to price exotic options. Finally, in the conclusions section the work is summarised along with a discussion of the results are future implications of the research.

2 Theory

In this section, the necessary theoretical background is introduced. First, the features and various types of financial options are discussed. Then, the most commonly used methods used in option pricing, which are also used in this work for verification and comparison of results, are presented, as well as their strengths and drawbacks. Firstly, the Cox-Ross-Rubinstein model, also known as the binomial model, is presented. Secondly, the famous Black-Scholes model is presented, and its benefits and core assumptions are discussed. Finally, a thorough introduction to the path integral framework used in this work for option pricing is presented.

2.1 Financial Options

It is worth our while to briefly revisit the fundamental financial terms and concepts used throughout this work.

Options play a crucial role in the financial markets today. They began as a simple tool for farmers to protect themselves against unpredictable prices for their crops. Over time, these instruments have developed into advanced financial tools, allowing investors to employ a range of strategies and manage their risks effectively. (Hull, 2021)

In essence, holding an option gives the holder a choice to buy or sell a specific asset at a set price within a certain time period. This feature sets options apart from other financial derivatives like forwards and futures. While both options and forwards base their value on another asset, a forward contract binds the holder to the agreed transaction. The unique structure of options means the most an investor can lose is the initial cost or premium, but the potential profit can be much larger. (Kwok, 2008)

There are two primary places where options are traded: on exchanges and in over-the-counter (OTC) markets. Exchange-traded options are standard contracts with defined

terms, and they're bought and sold on major platforms like the Chicago Board Options Exchange (CBOE) or Euronext. These platforms ensure clear trading rules, reliable pricing, and reduced risk of one party defaulting. On the other hand, OTC options are more flexible and can be customized to fit individual needs, but they come with a higher risk since there's no central system overseeing the trades. (Witzany, 2020)

Options attract different types of traders, each with their unique objectives. Hull (2021) recognises three types:

Hedgers: Hedgers are traders who use options to guard against potential losses. For example, a company that relies on fuel might buy options that become valuable if fuel prices rise, offsetting their increased costs.

Speculators: These traders aim to profit from price changes. If they believe the price of an asset will go up, they might buy a call option; if they think it will drop, they might buy a put option. Speculators can introduce more money into the market but can also face big losses if their predictions don't pan out.

Arbitrageurs: Traders who look for price differences in similar assets across different markets and try to profit from those differences, ensuring prices remain consistent across platforms.

Despite the many advantages of options, as always, there are risks involved. The leverage that increases potential gains can also lead to greater losses. Some option strategies can be complex, and without proper understanding, can lead to unexpected results, highlightning the importance of setting risk limits. Additionally, in some markets or with certain options, there might not be enough buyers or sellers, making it difficult to trade. Also, options have a set expiry date, which means they can end up being worthless if not used within the specified period. (Hull, 2021)

Options are powerful financial instruments that cater to a wide range of investment

strategies and risk management needs. However, like all investments, they require a thorough understanding to be used effectively.

2.1.1 Put and Call Options

Put and call options are the two fundamental types of options. A put option gives the holder the right, but not the obligation, to sell an underlying asset at the strike price within the specified period. The payoff for a put option P depends on the underlying asset's price S at the time of expiration T. The formula for the put option payoff is given by:

$$P = \max\{0,\ K - S\}, \tag{1}$$

where K denotes the strike price. If the underlying asset's price at expiration is below the strike price, the put option holder can exercise the option and sell the asset at the strike price, resulting in a profit. If the underlying asset's price exceeds the strike price, the put option becomes worthless, resulting in a zero payoff as indicated by the zero inside the maximum function in Equation (1). (Hull, 2021)

A call option grants the holder again the right, but not the obligation, to buy an underlying asset at the strike price within the specified period. Similarly to the put option, the payoff for a call option C depends on the underlying asset's price at expiration. The formula for the call option payoff is given by

$$C = \max\{0,\ S - K\}, \tag{2}$$

where, again K denotes the strike price. Now in the case of a call option, if the underlying asset's price at expiration is above the strike price, the call option holder can exercise the option and buy the asset at the strike price, resulting in a profit. Similarly, if the underlying asset's price is below the strike price, the call option becomes worthless, resulting in a zero payoff. A simplified example on the use of options is provided in Figure 1. (Hull, 2021)

Example

Let us consider a simple example illustrating the use of options: Suppose an investor expects the price of a particular stock, XYZ Corp., to increase in the next three months. To capitalize on this anticipated price rise, the investor purchases call options on XYZ Corp with a three-month expiration date. In case the stock price indeed rises, the investor can exercise the call options and buy the stock at the predetermined strike price, even if the market price is higher. By doing so, the investor benefits from the price difference and potentially earns a profit as indicated by Equation (2).

Figure 1. Simplified conceptual example on the use of options.

Furthermore *long* and *short* positions represent two fundamental concepts that describe the type of trade that has been made. Long and short positions on put and call options are summarized in Table 1.

A long position in the financial markets denotes the acquisition and holding of a security with the anticipation of a future price increment. This position is predicated on the premise that the asset's value will appreciate over time, providing the investor with the opportunity for capital gains. In the context of equities, taking a long position implies the purchase and ownership of stock shares. Similarly, in derivatives markets, one may hold a long position in options contracts, such as calls or puts, which conveys the right to buy or sell the underlying asset at a predetermined price within a specific duration. (Witzany, 2020)

Conversely, a short position in the financial markets is established through the borrowing and immediate selling of a security, with the objective of repurchasing it at a reduced cost in the future. This transaction is based on the assumption that the security's price will

experience a downturn, thus allowing the short-seller to procure it at a diminished price, return the borrowed securities to the lender, and retain the price differential as profit. (Witzany, 2020)

Table 1. Summary for Long and Short positions in Call and Put options.

	Call	**Put**
Long	Right to buy asset	Right to sell asset
Short	Obligation to sell asset	Obligation to buy asset

In options trading, a long call or put position represents the purchase of a call or put option, reflecting an expectation of an upward or downward movement in the underlying asset's price, respectively. A short call or put position, on the other hand, implies the selling of these options, based on the anticipation that the underlying asset's price will not surpass the strike price by an amount greater than the premium received.

2.1.2 Other Types of Options

Furthermore, there are various types of options apart from put and call that all have their distinctive features. *European options* can only be exercised at expiration. The option holder has the right to exercise the option on the expiration date but not before that.

American options on the other hand provides the holder with the flexibility to exercise the option at any time until expiration. This feature allows the option holder to capture potential gains earlier, depending on market conditions. For this reason, the price of the American option is usually higher than that of an otherwise similar European option. In the case of European and American options, the payoff is determined by Equations (1) and (2) for put and call options, respectively. European and American options are the most commonly traded options and are sometimes referred to as *vanilla options* with American options being the most traded type. Options other than vanilla options are

often referred to as *exotic options* and are traded considerably less compared to vanilla options. (Witzany, 2020)

Asian options[1] derive their value from the average price $A(0,T)$ of the underlying asset over a specific period, rather than the asset's price at expiration. The average is then deduced as

$$A(0,T) = \frac{1}{T}\int_0^T S(t)dt \qquad \text{or} \qquad A(0,T) = \frac{1}{N}\sum_{i=1}^{N} S(t_i), \tag{3}$$

for continuous and discrete times (of N time steps), denoted by t and t_i respectively. Then the payoffs for put and call options are given similarly to Equations (1) and (2), using the averages as

$$P = \max\{0,\ K - A(0,T)\} \tag{4}$$

$$C = \max\{0,\ A(0,T) - K\}. \tag{5}$$

These average-based options are particularly useful when investors want to reduce the impact of short-term price fluctuations. (Witzany, 2020)

Barrier options come with an additional feature known as a barrier level. The option's payoff, and whether it can be exercised or not, depends on whether the underlying asset's price reaches or exceeds the barrier level during the option's lifetime. Barrier options can be either knock-in or knock-out options, depending on whether they become active or extinguished when the barrier level is hit. (Hull, 2021)

A double barrier option has a lower barrier B_l, and an upper barrier B_u and is illustrated in Figure 2. In the figure the other possible path stays inside the limits given by the barriers and can be executed. However, the other path for the stock price exceeds the upper barrier, becoming worthless in the case of a knock-out barrier option.

[1] In this work only *fixed strike* Asian options with predetermined strike prices are considered.

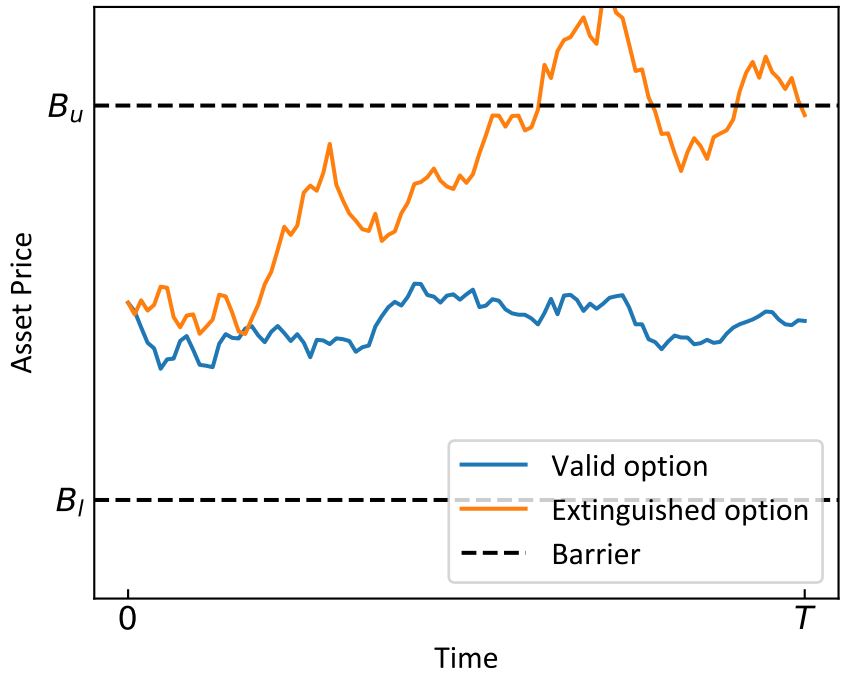


Figure 2. A schematic illustration of a double knock-out barrier option. The blue asset price stays inside the barriers, but the orange options exceeds the upper barrier before expiration and is knocked out, rendering it extinguished.

A fundamental problem in finance comes from the pricing of these options with no possible arbitrage ensuring a state of equilibrium in the market. In the next sections some commonly used closed-form and numerical pricing models, along with the path integral method are presented.

In this section, a general overview of financial options was provided, emphasizing their importance, users, and applications. Put and call options were revisited along with their respective key formulas. Additionally, European, American, Asian, and barrier options were discussed, highlighting their unique characteristics. It should be noted that there are numerous other types of options, and only the surface of exotic options was scratched.

2.2 The Cox-Ross-Rubinstein Model

The Cox-Ross-Rubinstein model provides a powerful, yet still intuitive method for pricing options and is thus first introduced in this work. It was first suggested by the Nobel-winning economist Sharpe (1978) and further generalized into the form used today by Cox, Ross and Rubinstein (1979). The model is to this day used extensively for option pricing as it can be utilized for the pricing of virtually any type of option easily.

The key idea is that in a discrete time lattice, with a spacing of Δt, the price of an asset[2] can either move up or down, giving rise to the alternatively used name *binomial model*. This procedure then generates paths that further yield a distribution of stock and options prices.

Consider a stock with initial price S that either moves up in value by a factor $u > 1$, or alternatively goes down by a factor $0 < d < 1$. Furthermore, the probability of S going up by is denoted by p, and the probability of S going down by $(1 - p)$, so that the total probability of possible outcomes adds up to unity. This general idea is illustrated in Figure 3. In this example, the expiration time $T = \Delta t$ as only one period is considered. (Witzany, 2020)

[2]The asset will be called stock from now on.

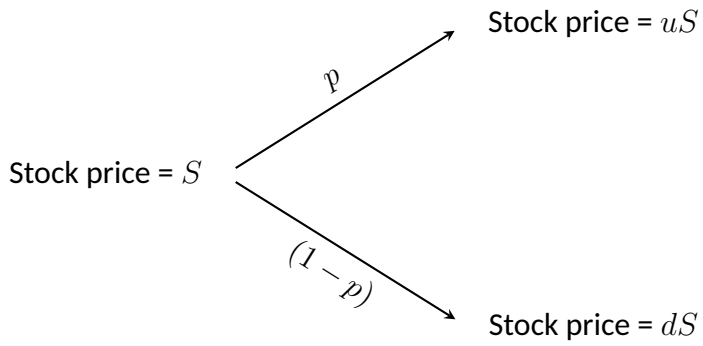


Figure 3. Illustration of a one period binomial tree. The initial stock price S goes up to uS with a probability p, or down to dS with a probability $(1-p)$.

Now, let us consider a more realistic *example*: an investor has a shares of stock S and amount B in a risk-free investment[3], with a growth factor R. The value of the investor's portfolio after one period, that is, $T = \Delta t$ is then as visualized in Figure 4.

[3]Typical convention is $R = e^{r\Delta t}$. It should be noted that this quantity is dimensionless as r has dimensions of time^{-1}, signifying the risk-free rate per unit time.

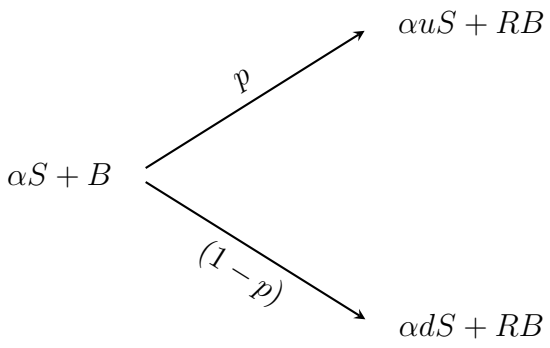


Figure 4. Illustration of a one period binomial tree with a risk-free investment B. The initial stock price S goes up to uS with a probability p, or down to dS with a probability $(1-p)$. The value of risk-free investment is RB in both cases.

The values of the parameters p, u and d must still be considered. Simple reasoning with a replicating portfolio results in that the value of the call can be written in terms of the binomial tree parameters as

$$\begin{aligned} C &= \frac{\frac{R-d}{u-d}C_u + \frac{u-R}{u-d}C_d}{R} \\ &= \frac{pC_u + (1-p)C_d}{e^{r\Delta t}}, \end{aligned} \tag{6}$$

where C_u and C_d are the values of the call option after one move up and down, respectively, and

$$p = \frac{e^{r\Delta t} - d}{u - d} \tag{7}$$

is defined as the risk-neutral probability (Kwok, 2008). That is, as the probabilities for up and down moves are given by p and $(1-p)$ as in Equation (7), the stock price is expected

to move as determined by discounting at the risk-free rate (Witzany, 2020):

$$\begin{aligned} E(S_{\Delta t}) &= pSu + (1-p)Sd \\ &= \frac{e^{r\Delta t} - d}{u - d} Su + \left(1 - \frac{e^{r\Delta t} - d}{u - d}\right) Sd \\ &= S\frac{ue^{r\Delta t} - ud + ud - de^{r\Delta t}}{u - d} \\ &= Se^{r\Delta t} \end{aligned} \tag{8}$$

The values of u and d are still to be determined. A common convention is to set $d = 1/u$ (Hull, 2021; Kwok, 2008). The values are found by matching the volatility of the one-period with a log-normally distributed result so that u is then written as

$$\begin{aligned} &\underbrace{pu^2 + (1-p)d^2 - [pu + (1-p)d]^2}_{\text{= one-period variance}} = \underbrace{e^{2r\Delta t}\left(e^{\sigma^2\Delta t} - 1\right)}_{\text{= normally distributed variance}} \\ \implies \quad & u \approx 1 + \sigma\sqrt{\Delta t} + \frac{\sigma^2}{2}\Delta t + \dots \end{aligned} \tag{9}$$

where a Taylor expansion of powers $\sqrt{\Delta t}$ was used. The first terms are the same as the terms of $e^{\sigma\sqrt{\Delta t}}$ up to $\mathcal{O}(\Delta t)$, making parameters

$$u = e^{\sigma\sqrt{\Delta t}} \tag{10}$$

$$d = e^{-\sigma\sqrt{\Delta t}} \tag{11}$$

a convenient choice. Additionally, it is required that $u > R > d$ to ensure there are no arbitrage possibilities. (Kwok, 2008)

Similarly then the price of, for example, a call option with two periods and three grid points, can be easily determined and visualized using a similar tree structure as illustrated in Figure 5:

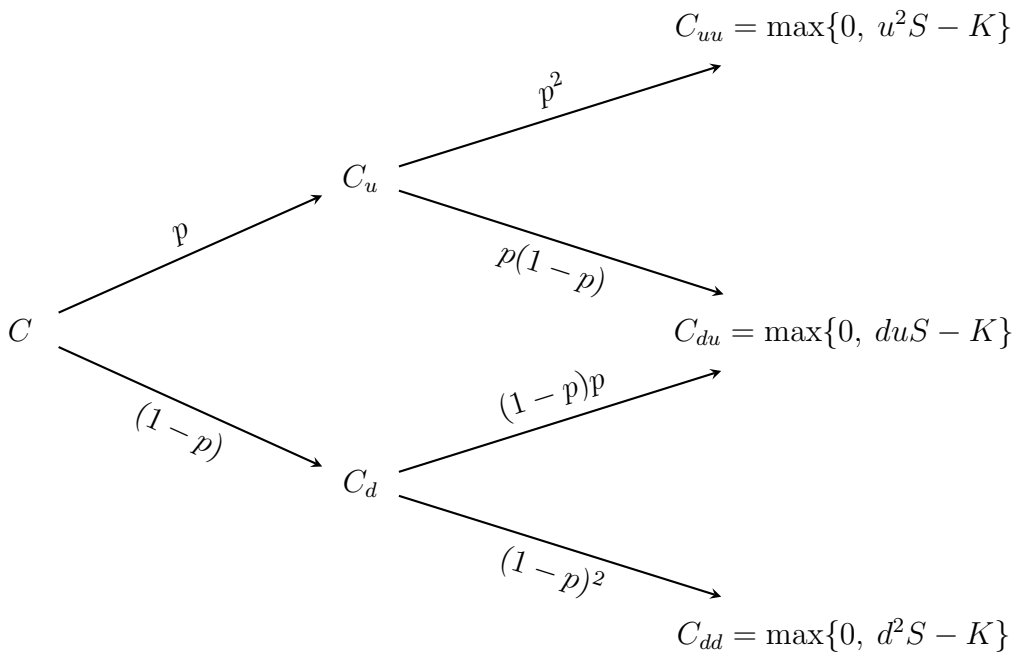


Figure 5. Two period binomial tree for a call option.

In the figure, C_u denotes the value of the option after a move up in the underlying stock price, and C_d similarly after a move down. Additionally, since two periods were considered, C_{uu}, C_{du} and C_{dd} denote the option prices after two moves up, a move down and up[4] and after two moves down, respectively. Furthermore, it is worth recalling that the option price is determined exactly from the underlying asset.

The results can be further generalized as the payoff can be written with n binomial steps with j moves up and $n - j$ moves down as (Witzany, 2020)

$$C_j = e^{-nr\Delta t}\left(\sum_{j=0}^{n}\frac{n!}{j!(n-j)!}p^j(1-p)^{n-j}\ \max\left\{0, u^j d^{n-j}S - K\right\}\right). \tag{12}$$

There is no real limit as to how many periods can be utilized: unsurprisingly a denser

[4]Or the other way around C_{ud}: first a move up and then a move down. It should be noted that clearly $C_{ud} = C_{du} = C$.

grid yields more accurate results, but comes with an increased computational cost. Furthermore, due to the Taylor expansion used in Equation (9), the spacing of grid points should be as close to zero as computationally reasonable. This in turn means that the computational cost also depends on the expiration date of the option. A simple example calculation using the one period binomial tree model for valuating a call option is shown in Figure 6. (Kwok, 2008)

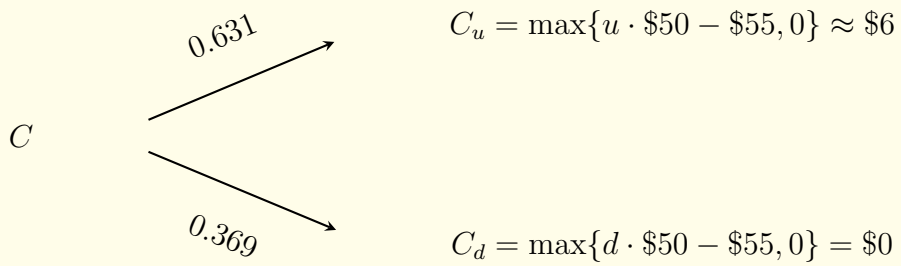
Example

Consider a European call option with a strike price of $\$55$, current price of the underlying stock of $\$50$ and expiration date at 1 year. The risk-free rate is 7% and volatility is 20%. Estimate the price of the option within a one period binomial scheme.

The key parameters can be calculated as

$$
\begin{aligned}
u &= e^{\sigma\sqrt{\Delta t}} = e^{0.20\cdot\sqrt{1}} \approx 1.221 \\
d &= e^{-\sigma\sqrt{\Delta t}} = e^{-0.20\cdot\sqrt{1}} \approx 0.819 \\
p &= \frac{e^{r\Delta t} - d}{u - d} = \frac{e^{0.07\cdot 1} - 0.819}{1.221 - 0.819} \approx 0.631
\end{aligned}
$$

Now the process for the call value can be visualized as follows:



And finally, the option value is given by Equation (12)

$$
\begin{aligned}
C &= e^{-r\Delta t} p \cdot C_u \\
&\approx e^{-0.07\cdot 1} \cdot 0.631 \cdot \$6 \\
&\approx \$3.53
\end{aligned}
$$

Figure 6. Example calculation using one period binomial tree.

It is evidently clear that the method is not exact: Firstly, the spacing of the time steps can cause errors. Secondly, the stock price can be realistically expected to move up or down by different values or stay exactly the same. However, the discrete approach of the model allows for a straightforward and adaptable methodology in valuing options, accommodating a variety of conditions and contractual features. The iterative process is numerically simple to solve and the method provides a clear picture of the option's value at each point in time, making it a powerful tool for valuing, not only vanilla options, but some exotic options as well. Additionally, it should be noted that all the methods used in this section are applicable to put options by adjusting the payoff as indicated in Equation (1).

2.3 The Black-Scholes Model

The Black-Scholes-Merton (BSM) model, widely regarded as one of the most influential breakthroughs in the realm of modern finance, represents a pivotal point in the evolution of financial theory and its practical applications. Introduced in the early 1970s by the Nobel-winning economists Fischer Black and Myron Scholes (Black & Scholes, 1973), and further formalised by Robert Merton (Merton, 1973), the BSM model revolutionized the way financial markets perceive, value, and manage options. Over the decades, it has become hugely important in the field of quantitative finance, offering a theoretical framework and a simple closed-form solution that facilitates the pricing and risk assessment of financial derivatives.

A rather simple derivation of the famous Black-Scholes-Merton equation is presented, continuing from the Cox-Ross-Rubinstein model discussed previously in Section 2.2. It should be noted that there are numerous ways of presenting the derivation, such as the use of Itô's lemma (Hull, 2021), by utilizing a hedging argument (Black & Scholes, 1973), by replicating portfolio (Merton, 1973) and purely mathematical analysis of partial differential equations (Olver, 2013), to name a few.

2.3.1 Derivation

The derivation presented in this work for the BSM equation follows directly the Cox-Ross-Rubinstein model and thus, is a natural choice here. The method is not as commonly used today as e.g., using Itô's lemma, but similar steps can be found in literature, e.g., in Atzberger (2006) and the work of Cox et al. (1979). The derivation is done by considering the one period case in the continuum limit, that is, as $\Delta t \to 0$ following Kwok (2008). Then the call can be written as

$$C(S, t - \Delta t) = e^{-r\Delta t}(pC_u + (1-p)C_d) \tag{13}$$

Then, by taking the Taylor expansion around (S, t), the left-hand side of Equation (13) can be written as

$$C(S, t - \Delta t) \approx (1 - e^{-r\Delta t})C - \frac{\partial C}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 C}{\partial t^2}\Delta t^2 + \dots \tag{14}$$

and similarly for the right-hand side:

$$\begin{aligned} e^{-r\Delta t}(pC_u + (1-p)C_d) \approx & \underbrace{e^{-r\Delta t}(p(u-1) + (1-pp)(d-1))}_{=r\Delta t + \mathcal{O}(\Delta t^2)} S\frac{\partial C}{\partial S} \\ & + \frac{1}{2}\underbrace{e^{-r\Delta t}(p(u-1)^2 + (1-p)(d-1)^2)}_{=\sigma^2\Delta t + \mathcal{O}(\Delta t^2)} S^2\frac{\partial^2 C}{\partial S^2} \\ & + \frac{1}{6}\underbrace{e^{-r\Delta t}(p(u-1)^2 + (1-p)(d-1)^2)}_{=\mathcal{O}(\Delta t^2)} S^2\frac{\partial^2 C}{\partial S^2} + \dots \end{aligned} \tag{15}$$

where the fact that $1 - e^{-r\Delta t} = r\Delta t + \mathcal{O}(\Delta t^2)$ was used. Putting Equations (13) to (15) together, the resulting equation yields

$$\begin{aligned}
\left[\frac{\partial C}{\partial t} + rS\frac{\partial C}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 C}{\partial S^2} - rC\right]\Delta t + \mathcal{O}(\Delta t^2) &= 0 \\
\frac{\partial C}{\partial t} + rS\frac{\partial C}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 C}{\partial S^2} - rC + \underbrace{\mathcal{O}(\Delta t)}_{\xrightarrow[\Delta t\to 0]{}0} &= 0 \\
\frac{\partial C}{\partial t} + rS\frac{\partial C}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 C}{\partial S^2} - rC &= 0, \qquad (16)
\end{aligned}$$

that is, via Taylor expansions around the continuum limit of the Cox-Ross-Rubinstein model one arrives to the celebrated Black-Scholes-Merton equation. (Kwok, 2008)

2.3.2 Solution

The BSM equation is a somewhat complicated partial differential equation (PDE) and it would be tedious trying to solve it as is. Luckily, the equation can be simplified greatly by turning it into the form of a heat equation as done in most textbooks, such as (Olver, 2013) which is followed in the following algebraic manipulation. Now let us consider the BSM equation for a European call option

$$\frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 C}{\partial S^2} + rS\frac{\partial C}{\partial S} - rC = 0, \qquad (17)$$

where the familiar terminal and initial *boundary conditions* are imposed:

$$C(S,T) = \max\{0, S-K\} \qquad (18)$$

$$C(0,t) = 0, \qquad (19)$$

respectively.

By transforming the variables so that $S = e^y$ and $t = T - \tau/\sigma^2$, so that $S\partial/\partial S \to \partial/\partial y$

and $\partial/\partial t \rightarrow -\partial/\partial\tau$, the BSM equation can be written as

$$\frac{\partial C}{\partial \tau} - \frac{1}{2}\sigma^2 \frac{\partial^2 C}{\partial y^2} - \left(r - \frac{1}{2}\sigma^2\right)\frac{\partial C}{\partial y} = 0. \tag{20}$$

Now, by the change of variables $u = e^{r\tau}C$, it can be shown that

$$\frac{\partial u}{\partial \tau} - \frac{1}{2}\sigma^2 \frac{\partial^2 u}{\partial y^2} - \left(r - \frac{1}{2}\sigma^2\right)\frac{\partial u}{\partial y} = 0. \tag{21}$$

Finally, the first order term in the PDE vanishes as a substitution $x = y + (r - \sigma^2/2)\tau$ is made. This can be interpreted as transforming the PDE into stationary "steady-state" problem instead of considering a moving frame of reference. This change results in

$$\begin{aligned} \frac{\partial u}{\partial \tau} &= \frac{1}{2}\sigma^2 \frac{\partial^2 u}{\partial x^2} \\ &= k\frac{\partial^2 u}{\partial x^2}, \end{aligned} \tag{22}$$

which has the familiar shape of the well-known heat equation with $k = \sigma^2/2$. Keeping in mind the changes of variables, the terminal condition is now expressed as the initial condition

$$u(x, 0) = \max\left\{0, e^{x(k+1)/2} - e^{x(k-1)/2}\right\}. \tag{23}$$

It should be noted that for a put option, as described in Equation (1), the initial condition is similarly set as

$$u(x, 0) = \max\left\{0, e^{x(k-1)/2} - e^{x(k+1)/2}\right\}. \tag{24}$$

Now that the BSM equation is successfully transformed into the form of a heat equation, the well-known solutions of this specific form can be directly used. The heat equation is frequently encountered in many disciplines such as physics and engineering, and thus, its solutions are also well documented in e.g. (Olver, 2013). The general solution is written

as

$$u(x,t) = \frac{1}{2\sqrt{\pi t}} \int_0^\infty \left(\left(e^{x(k-1)/2} - e^{x(k+1)/2} \right) e^{-(x-s)^2/4t} \right) ds. \tag{25}$$

Equation (25) can be solved with yet another change of variables, $x' = (s-x)/\sqrt{2t}$ using the cumulative Normal distribution $\mathcal{N}$ as

$$u(x,t) = e^{x(k+1)/2+t(k+1)^2/4}\mathcal{N}(d_1) - e^{x(k-1)/2+t(k-1)^2/4}\mathcal{N}(d_2), \tag{26}$$

where

$$d_1 = \frac{x}{\sqrt{2t}} + \frac{\sqrt{2t}}{2}(k+1) \tag{27}$$

$$d_2 = \frac{x}{\sqrt{2t}} + \frac{\sqrt{2t}}{2}(k-1). \tag{28}$$

By transforming the variables back to the original ones, the final closed-form solution is found as

$$C = S\mathcal{N}(d_1) - Ke^{-rT}\mathcal{N}(d_2), \tag{29}$$

where

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \tag{30}$$

$$d_2 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} = d_1 - \sigma\sqrt{T}. \tag{31}$$

An example calculation using Equation (29) is shown in Figure 7. Additionally, by following similar steps for the value of European put option, P, the closed-form Black-Scholes formula is quite similarly written as (Hull, 2021):

$$P = Ke^{-rT}\mathcal{N}(-d_2) - S\mathcal{N}(-d_1) \tag{32}$$

Example

Let us continue from the same example that was used to illustrate the use of the Cox-Ross-Rubinstein formulation. That is, the strike price of the underlying stock is $\$55$, current price is $\$50$ and expiration date at 1 year. The risk-free rate is 7% and volatility is 20%. Now the Black-Scholes equation can be used directly to price the European call option. First, the parameters d_1 and d_2 can be calculated as

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} = \frac{\ln\left(\frac{50}{55}\right) + \left(0.07 + \frac{0.20^2}{2}\right)\cdot 1}{0.20\cdot\sqrt{1}} \approx -0.027$$
$$d_2 = d_1 - \sigma\sqrt{T} = -5.572 - 0.20\cdot\sqrt{1} \approx -0.227$$

Now the option price is simply

$$\begin{aligned} C &= S\mathcal{N}(d_1) - Ke^{-rT}\mathcal{N}(d_2) \\ &= \$50\cdot 0.489 - \$55\cdot e^{-0.07\cdot 1}\cdot 0.410 \\ &\approx \$3.425 \end{aligned}$$

Figure 7. An example calculation using the Black-Scholes formula.

The Black-Scholes formula is a useful and easy-to-utilize closed-form tool for option valuation. Witzany (2020) summarizes some of the assumptions underlying the BSM model:

- The price dynamics of the asset are postulated to adhere to a geometric Brownian motion with invariant drift and volatility parameters, yielding log-normally distributed returns.
- The underlying asset is assumed not to distribute any form of income, such as dividends or interest.

- The risk-free rate r is assumed to remain constant throughout the temporal horizon of the option.
- The financial model presupposes the availability of lending and borrowing at the risk-free rate without the presence of any constraints or differing conditions.
- Transactional costs, including but not limited to brokerage fees and taxes, are considered to be non-existent.
- The model assumes assets to be perfectly divisible, allowing for transactions of fractional shares or units.
- It is posited that the short selling of securities can be conducted with no impediments or additional requirements.
- The absence of arbitrage opportunities is a foundational assumption, ensuring that the pricing model remains free from the possibility of riskless profit.
- The model is predicated upon the continuous trading of securities, with the capability to execute trades at infinitesimally small intervals of time.

It is clear that these rather ideal conditions do not take place in financial markets, and more realistic models should be utilized for more accurate pricing of options.

2.4 Path Integral Formulation

The path integral[5] formulation of quantum mechanics is a powerful and elegant theoretical framework that provides a unique perspective on the behavior of quantum systems. The path integral formulation offers a different, perhaps a more intuitive way, to understand the connections between classical and quantum systems, and has been used with great success in theoretical physics, and perhaps a bit surprisingly, financial mathematics

[5]Not to be confused with a line integral encountered in, e.g., vector calculus.

(Kleinert, 2009). The method was formally put together by the Nobel-winning theoretical physicist Richard Feynman (Feynman, 1948) building on the mathematical ideas of Norbert Wiener (Wiener & Masani, 1976) and Paul Dirac[6] (Dirac, 1933).

At its core, the path integral formulation takes a departure from the more traditional formulation involving the famous Schrödinger equation and Heisenberg's matrix mechanics, and offers a more intuitive and visual way to describe the evolution of quantum systems (Sakurai & Napolitano, 2020). Instead of focusing on wave functions, operators and matrix algebra, it treats quantum processes as the result of all possible paths or trajectories a particle or system could take to move from one state[7] to another. This method invokes the concept of a "sum over paths", where every conceivable path contributes to the quantum amplitude of the final state, and the total probability is obtained by summing these contributions.

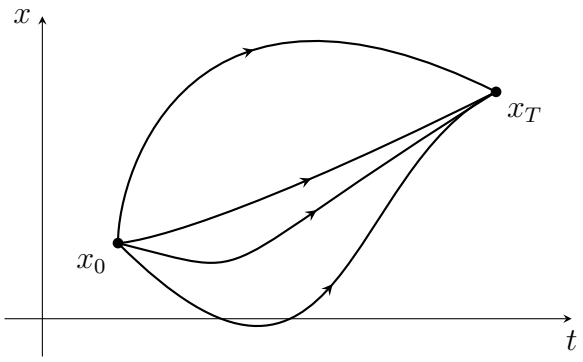


Figure 8. Illustration of some possible paths a system can take from state x_0 to x_T.

The quantum nature of following mathematical formulation can easily be overwhelming. The interested readers are referred to the excellent books (Griffiths & Schroeter, 2018;

[6] Another Nobel-winning quantum theorist.
[7] A quantum state is a complex-valued vector that embeds all information of the system.

Sakurai & Napolitano, 2020) that go a long way in understanding quantum mechanics.

The path integral formulation has been utilized in option pricing many times in previous theoretical and numerical studies. The pioneering work on path integrals in finance can be traced back to the late 20th century when physicists began exploring financial markets. The papers by Dash (1988, 1989) introduced the idea that financial markets could be modeled using stochastic processes similar to those found in quantum mechanics. This analogy opened the door for the use of path integral methods in the valuation of options.

Various theoretical and numerical studies replicating known results of vanilla options with good success were published in the following years (Linetsky, 1997; Montagna, Nicrosini, & Moreni, 2002). Additionally, the power of the method was quickly harnessed as the flexibility of the path integral formulation becomes particularly evident when dealing with time-dependent variables. The traditional path integral approach was then quickly extended to accommodate time-dependent volatility, providing a more accurate representation of real-life market conditions (Baaquie, 1997).

Additionally, numerous studies on path-dependent Asian options (Bormetti, Montagna, Moreni, & Nicrosini, 2006; Linetsky, 1997), as well as barrier options (Chen & Guo, 2023; Guarch Termens, 2022; Linetsky, 1997) have been conducted. Barrier options have been studied using both a modified Hamiltonian theoretically[8] (Chen & Guo, 2023), and a modified numerical scheme (Guarch Termens, 2022). Various other types of studies have been conducted more recently, such as advanced numerical analysis on using the path integral formulation to identify arbitrage (Contreras, Pellicer, Villena, & Ruiz, 2010), and the study of auto-correlations for long maturity options and non-Gaussian dynamics within the paths which the stock price can take (Capuozzo, Panella, Schettini Gherardini, & Vvedensky, 2021) with interesting results. These results clearly highlight the potential advantages of the path integral formulation when compared to the more traditional methods.

[8] Essentially, double barrier options are analogous to quantum mechanical particles in potential wells.

2.4.1 The Black-Scholes Hamiltonian

A key quantity in quantum mechanics is the *Hamiltonian operator* $\hat{H}$. Essentially it describes the total energy and can be written as the sum of kinetic energy operator $\hat{T}$, and potential energy operator $\hat{V}$ (Griffiths & Schroeter, 2018):

$$\hat{H} = \hat{T} + \hat{V}. \tag{33}$$

The eigenvalues of the Hamiltonian give the energy levels that can be measured from the system. Most formulations of quantum mechanics, such as the path integral formulation, use the Hamiltonian, or a closely related Lagrangian function to describe the dynamics of the system. Thus, it is essential that a way to represent the dynamics of the option pricing problem is found utilizing a Hamiltonian function. (Sakurai & Napolitano, 2020)

The time-dependent *Schrödinger equation* can be written in terms of the Hamiltonian as a PDE in the form

$$i\hbar\frac{\partial\psi}{\partial t} = \hat{H}\psi, \tag{34}$$

where i the imaginary unit, $\hbar$ is the Planck's constant which is set to unity for clarity, and ψ is the *wave function* of the system (Griffiths & Schroeter, 2018). Clearly, ψ must be complex-valued. The solution can be written as an integral equation

$$\psi(x,t) = \int_{-\infty}^{\infty} p(x,t|x_0,0)\psi(x_0,0)dx, \tag{35}$$

where p is the *kernel* or the *propagator* of the system, essentially describing the transition probability[9] for the transition from an initial state $(x_0, 0)$ to a later state (x_t, t) (Bustamante & Contreras, 2016; Utama & Purqon, 2016).

[9] This whole process is often called the *time evolution* of the system. Using the quantum mechanical bra-ket notation (Griffiths & Schroeter, 2018), the propagator would be expressed as $p(x,t|x_0,0) = \langle x|e^{-t\hat{H}}|x_0\rangle$ (Baaquie, Corianò, & Srikant, 2004).

By performing a Wick rotation[10] $t \rightarrow -it$ the Schrödinger equation, (34), can be transformed into the exact form of a heat equation (Contreras et al., 2010):

$$-\frac{\partial \psi}{\partial t} = \hat{H}\psi, \tag{36}$$

The general shape of the Wick rotated Schrödinger equation, Equation (36), greatly resembles the Black-Scholes equation after change of variables $x = \ln S$, so that the initial stock price $x_0 = \ln S_0$ and the stock price at expiration is $x_T = \ln S_T$:

$$\frac{\partial C}{\partial t} = \left[-\frac{\sigma^2}{2}\frac{\partial^2}{\partial x^2} + \left(\frac{\sigma^2}{2} - r \right) \frac{\partial}{\partial x} + r \right] C, \tag{37}$$

from which the *Black-Scholes-Schrödinger Hamiltonian* can now be identified as

$$\begin{aligned} \hat{H}_{\text{BS}} &= -\frac{\sigma^2}{2}\frac{\partial^2}{\partial x^2} + \left(\frac{\sigma^2}{2} - r \right) \frac{\partial}{\partial x} + r \\ &= -\frac{\sigma^2}{2}\frac{\partial^2}{\partial x^2} - \mu \frac{\partial}{\partial x} + r, \end{aligned} \tag{38}$$

where $\mu = r - \sigma^2/2$. Analogous to Equation (35), the solution to the option pricing problem (in terms of S) can be expressed as (Utama & Purqon, 2016)

$$C(S_T, T) = \int_{-\infty}^{\infty} p(S_T|S_0) C(S, t) dS. \tag{39}$$

2.4.2 Feynman-Kac Formula

The Black-Scholes Schrödinger equation, Equation (37) with a terminal condition has a unique solution in the form of the Feynman-Kac equation:

$$C(S_0, 0) = e^{-r(T)} E[C(S_T, T)], \tag{40}$$

[10] The name rotation naturally arises from the fact that the multiplication by an imaginary unit $i = \sqrt{-1}$ is performed, yielding a 90° counterclockwise rotation on the imaginary plane.

where E denotes the expected value, in this case for the option value at $t = T$ with initial stock price S_0 at $t = 0$ (Capuozzo et al., 2021; Utama & Purqon, 2016). Luckily, a solution already exists in the form of Equation (39), and all that's left to do is to find an expression for the propagator.

Essentially, this is where the quantum mechanical path integral formulation takes place as it is this expectation value that can be expressed as a path integral. The mathematically rigorous derivation of the propagator is a somewhat tedious task, and the final result for the logarithmic price of the option is taken from literature (Capuozzo et al., 2021; Montagna et al., 2002; Utama & Purqon, 2016):

$$C(x_0, 0) = e^{-rT} \int_{-\infty}^{\infty} \int_{x_0}^{x_T} C(e^{x_T}) e^{-S_{\text{BS}}[x(t)]} \mathcal{D}[x_T] dx_T, \tag{41}$$

where S_{BS} is the Black Scholes *action*, defined in terms of the Black Scholes *Lagrangian* $\mathcal{L}_{\text{BS}}$ as

$$S_{\text{BS}} = \int_0^T \mathcal{L}_{\text{BS}} dt. \tag{42}$$

The Black Scholes Lagrangian is closely related to the Black Scholes Hamiltonian in Equation (38). However, the connection again requires quantum mechanical formulation beyond the scope of this work, and the form[11] is taken as described by Capuozzo et al. (2021)

$$\begin{aligned} \mathcal{L}_{\text{BS}} &= \frac{1}{2\sigma^2} \left[\frac{dx}{dt} + \left(\frac{\sigma^2}{2} - r \right) \right]^2 \\ &= \frac{1}{2\sigma^2} \left(\dot{x} - \mu \right)^2, \end{aligned} \tag{43}$$

[11] It comes as no surprise that the Lagrangian has the form of the *Onsager-Machlup Lagrangian*, which gives the most probable path of a diffusion process (Dürr & Bach, 1978). Indeed, by recalling that the classical kinetic energy is written as $mv^2/2$, it can be deduced from Equation (43) that the option pricing problem is analogous to a free particle with a mass $m = 1/\sigma^2$ and velocity $v = dx/dt - \mu$.

and thus, the action integral can be solved simply as

$$S_{\text{BS}} = \int_0^T \frac{1}{2\sigma^2}\left[\frac{dx}{dt} + \left(\frac{\sigma^2}{2} - r\right)\right]^2 dt \tag{44}$$

$$= \int_0^T \frac{1}{2\sigma^2}\left(\frac{dx}{dt}\right)^2 dt - \frac{\mu}{\sigma^2}(x_T - x_0) + \frac{\mu^2 T}{2\sigma^2}. \tag{45}$$

In essence, the action is a measure of how ”costly” it is for a physical system to evolve. In the path integral formulation, the action of the individual path determines the probability of a certain path. Classically, a system evolves in what is known as the path of least action, meaning that the smaller the action of a certain path, the more likely the system is to take said path. (Goldstein, Poole, & Safko, 2002)

The path integral equation, Equation (41), may look rather intimidating so it is worth our while to further break it down, now that the meaning of the action term is clarified.

The very first integral takes into account every possible final stock price a path can end up in. The outer integral is the integral of all possible stock prices S_T at maturity. The inner integral is the integral over all possible paths and is linked to the propagator as

$$p(x_T, T|x_0, 0) = \int_{x_0}^{x_T} \mathcal{D}[x_T] e^{-S_{\text{BS}}}$$

$$= \lim_{N\to\infty}\left[\prod_{k=0}^{N-1}\int_{-\infty}^{\infty}\frac{dx_k}{\sqrt{2\pi\sigma^2\delta t_k}}\right] e^{-S_{\text{BS}}}, \tag{46}$$

that is, a continuum limit of all possible paths is taken and their respective actions are calculated, aligning with the previous remarks on the nature of the action specifying the probability of the path. (Capuozzo et al., 2021)

Finally, $C(e^{x_T})$ is a payoff function for the path with a specific outcome. This is deduced differently for different option types, as indicated in Equations (1) and (2). Note that for calculating the payoff, the logarithmic stock price is converted back to the normal value, using the exponential function.

With this level of theoretical background, we can now start looking for ways to do numerical calculations based on the path integral framework.

3 Methodology

In this section, the numerical procedures that enable the use of path integral formulation in practical applications are presented. Firstly, the Monte Carlo integration method which is especially useful in multidimensional integrals is introduced. Secondly, the path integral equations are discretized into a form that can be numerically solved similarly to the path integral Monte Carlo (PIMC) scheme. Finally, the software used in this study to perform the numerical calculations is presented along with some performance-enhancing techniques.

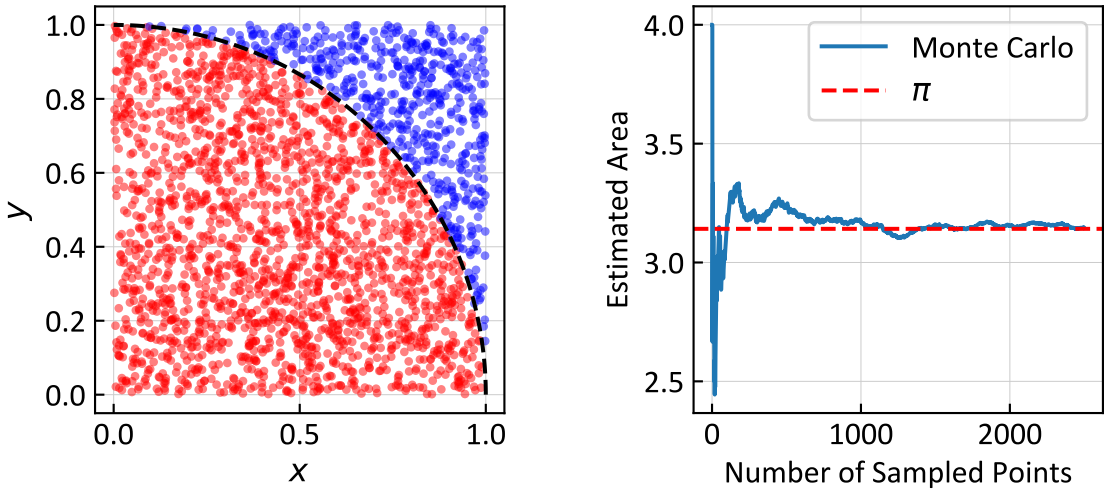
3.1 Monte Carlo Simulations

Monte Carlo simulations are a powerful and versatile computational technique that has revolutionized the way we approach complex problems across a wide range of fields, from physics and finance to engineering and biology. Named after the Monte Carlo Casino in Monaco, known for its games of chance, Monte Carlo simulations give means to solve deterministic problems with probabilistic predictions. (Harrison, 2010)

In a Monte Carlo simulation, random sampling and statistical methods are used to approximate solutions to problems that might be too intricate or analytically intractable to solve directly. This approach is especially valuable when dealing with systems that involve a high degree of uncertainty, randomness, or a multitude of interacting variables, and is thus highly applicable to be used with the path integral formalism. (Thomopoulos, 2013)

Let us illustrate the method with a classic example of calculating the area of a unit circle using Monte Carlo simulations. In this case, a set of randomly sampled points in the x, y-plane between 0 and 1 are generated, so that the points can only be inside the first quarter of the circle. Then, we check if the sampled point is within the radius of a circle or not. That is, we calculate $r = \sqrt{x^2 + y^2}$ and see if $r < 1$. The area of the circle is finally given by the ratio of points inside the circle multiplied by four, as only a quarter of

the circle was considered. This method is illustrated in Figure 9. Clearly, as the number of sampled points is increased, the value of the integral becomes more and more accurate and converges towards π.



(a) Random sampled points. Points that are inside the unit circle (black dashed line) are colored red, and those outside the circle are marked blue. The area is then obtained as the fraction of red points in the simulation multiplied by 4 as only one quarter of the circle is shown.

(b) Convergence of Monte Carlo simulation as a function of sampled data points. Initially there are large fluctuations. Increasing the number of sampled points increases the accuracy.

Figure 9. Monte Carlo example simulation of calculating the area of a circle. In this example, a quarter of the circle is considered, and thus the end result is multiplied by 4.

It comes as no surprise that Monte Carlo integration is not the most efficient method for calculating two-dimensional integrals. Indeed, the error is proportional to $N^{-1/2}$, where N is the number of sampled points. For comparison, other, deterministic integration methods such as trapezoid and Simpson's method, have error terms proportional to $N^{-\alpha/d}$, where α is a constant depending on the method and d is the dimension of the integral (Capuozzo et al., 2021). From this, it can be immediately noticed that the error in Monte Carlo integration does not depend on the number of dimensions used. The exact formulation of the path integral method involves infinitely many dimensions, making Monte Carlo methods a clear choice for the numerical tackling of path integral problems.

Monte Carlo methods have been widely successful in numerical simulations of complex systems. Especially with the fast increase in computational capabilities, these simulations have made complex problem-solving possible, providing a powerful tool for researchers, engineers, and decision-makers to tackle real-world challenges in a wide array of disciplines. The increase in computing capabilities means that more and more problems can be solved with high accuracy by sampling a sufficient amount of statistics to make conclusions. This section acted as a friendly introduction and a simple overview of the method. However, more details on performing numerical simulations using Monte Carlo methods can be found in e.g. the book (Thomopoulos, 2013).

3.2 Markov Chain Monte Carlo Algorithm

For the numerical implementation of the path integral formulation, some numerical approximations must take place. The implementation has been constructed using selected ideas from references (Capuozzo et al., 2021; Devreese, Lemmens, & Tempere, 2010). Most notably, the time lattice is discretized from 0 to T into N lattice points so that the lattice spacing is given by

$$\Delta t = \frac{N}{T}. \tag{47}$$

This time discretization of paths is illustrated in Figure 10.

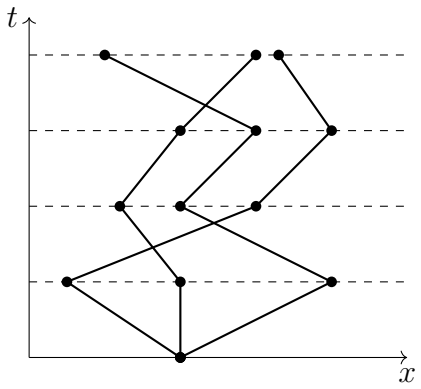


Figure 10. Illustration of the time lattice discretization Monte Carlo method for three paths. Note that all the paths start from the same initial price.

Then, paths are generated for the logarithmic stock price x, starting from x_0. This is done by randomly sampling stock prices for each lattice point from geometric *Brownian motion* as suggested by (Capuozzo et al., 2021) so that

$$\Delta S = \mu S \Delta t + \sigma S \epsilon \sqrt{\Delta t}, \tag{48}$$

where ϵ represents noise sampled from a normal distribution with a mean of zero and standard deviation of unity (Hull, 2021).

The numerical implementation follows a Metropolis-Hastings style Markov chain Monte Carlo algorithm (Hastings, 1970; Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 2004). The key idea is to propose changes to the path at random time steps. In this method, first, the action S, of the path is calculated. Now that a discrete lattice with N lattice points is used, the continuous action in Equation (44) is written using *finite differences* as a summation

$$S = \sum_{k=0}^{N-1} \frac{\Delta t}{2\sigma^2} \left(\frac{x_{k+1} - x_k}{\Delta t} - r + \frac{\sigma^2}{2} \right)^2 \tag{49}$$

After proposing a change $x \to x'$, a new action, S', is calculated for the path. If the new action is lower than the previously calculated action, the change is accepted (Devreese

et al., 2010). By doing this, the algorithm converges towards a path with less action, and thus, a more probable path. The algorithm can be presented by the following pseudocode:

```
Initialize path with some configuration x
while run optimization routine do
    Calculate the action of the current path S
    Propose a change at one time step: x → x′
    Calculate the action of the new path S′
    if S_f − S_i < 0 then
        Accept the change: x ← x′
    end if
end while
```

The path optimization routine in action[12] is illustrated in Figure 11 where an initial path is optimized on a time lattice of $N = 30$ points, and the action is reduced by approximately 83%. In a sense, the action optimization routine can be thought of as a smoothing filter on the path. This is clear since the finite difference terms in the expression for discrete action, Equation (49), increase the total action if there are large fluctuations present.

[12] Pun intended.

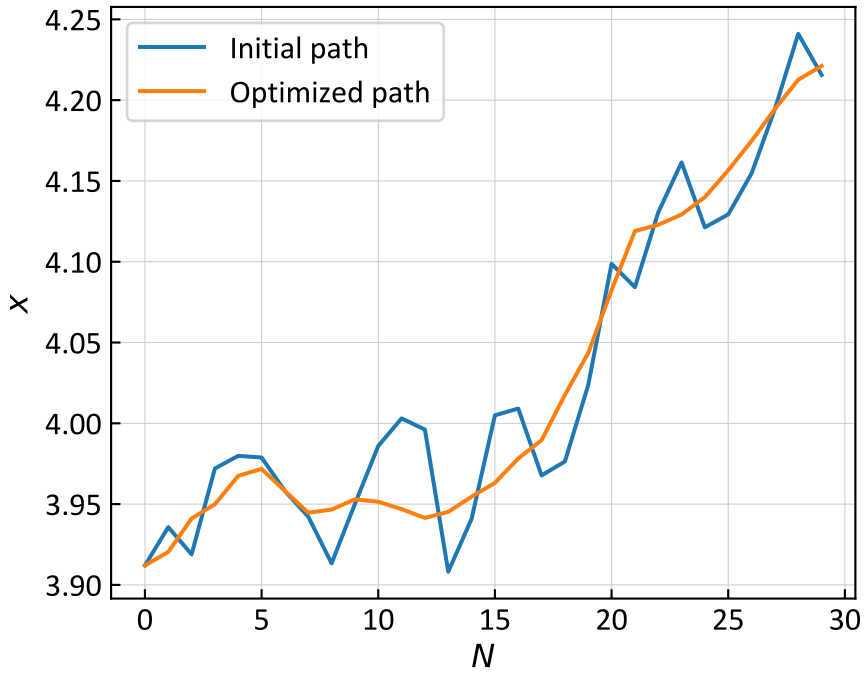


Figure 11. Example of the action optimization routine. Simulation parameters: $S_0 = \$50$, $\sigma = 0.2, r = 0.07, T = 1, N = 30$.

Repeating the calculation for multiple paths with different initial conditions as illustrated in Figure 10 and weighting them based on their action, an average price can be computed for the option. The payoff can then be calculated for various options as discussed in Section 2. For European options, the valuation is done based on the strike prices S_T of paths as indicated by Equations (1) and (2). On the other hand, for Asian options, the valuation is based on the averages of paths based on Equation (4).

Finally, a method for valuating barrier options is proposed. The implementation of a barrier can be done in two ways: Firstly, by imposing a potential function well into the Hamiltonian. Secondly, by limiting the proposed changes to the path of the option to be within the barriers. Both methods are implemented. The potential function can be defined to be analogous to that of a particle in a square potential well $V(x)$ between a lower barrier B_l and an upper barrier B_u (Baaquie et al., 2004; Chen & Guo, 2023). Then the double

barrier Hamiltonian $\hat{H}_{\mathsf{DB}}$, is simply written as

$$\hat{H}_{\mathsf{DB}} = -\frac{\sigma^2}{2}\frac{\partial^2}{\partial x^2} + \left(\frac{\sigma^2}{2} - r\right)\frac{\partial}{\partial x} + r + V(x), \tag{50}$$

where the square potential well is defined as

$$V(x) = \begin{cases} 0, & B_l \leq x \leq B_u \\ V_0 \to \infty, & \text{otherwise.} \end{cases} \tag{51}$$

The second method of limiting the range of proposed changes is done following the work of Guarch Termens (2022). For a double knock out barrier with a lower barrier $B_l = e^{x_{\mathsf{min}}}$ and an upper barrier $B_u = e^{x_{\mathsf{max}}}$, the change is proposed from an uniform distribution U, so that

$$x' \sim U\big(\max\{x - L_n, x_{\mathsf{min}}\},\ \min\{x + L_n, x_{\mathsf{max}}\}\big), \tag{52}$$

where L_n is a reduced interval based on the iteration n of the algorithm at each time step:

$$L_n = \frac{x_{\mathsf{max}} - x_{\mathsf{min}}}{2}\left(\frac{N - n}{N}\right). \tag{53}$$

The reduction of the interval is done to increase the accuracy of the algorithm and is implemented in all calculations (Guarch Termens, 2022). While the barrier option implementations presented here are for double barriers, they can be straightforwardly converted to single barrier cases by adjusting the other barrier so that it does not affect the calculation, i.e., by setting the barrier location far from the expected prices, or in the case of modified Hamiltonian by defining a potential barrier instead of a well.

3.3 Simulation Software

For the numerical calculations, a Python 3 (Van Rossum & Drake, 2009) program is utilized. Python was chosen for its ease of implementation and readability. Additionally, Python is an open-source software, making it a compelling choice compared to some other commonly used options used in simulations, such as Matlab. Additionally, industry standard Python libraries NumPy (Harris et al., 2020) and Matplotlib (Hunter, 2007) are used extensively for performing numerical computations and visualizing the results, respectively.

In Monte Carlo simulations, where a sufficient amount of statistical data is crucial, optimizing simulation performance is essential to ensure the necessary statistics can be effectively acquired. While Python is not as fast as some other open-source alternatives, such as C++ and Fortran, there are numerous methods that can be easily used to improve the performance of the simulation. In order to achieve relatively easy performance boosts, Numba (Lam, Pitrou, & Seibert, 2015) is used for enabling parallel computing within the simulation program and utilizing machine-level code, and CuPy (Okuta, Unno, Nishino, Hido, & Loomis, 2017) for GPU (Graphics Processing Unit) acceleration of floating point operations.

3.3.1 Parallel Computing

Parallel computing emerges from the foundational idea of conducting several computations concurrently, or *in parallel*. By exploiting the decomposition of larger problems into smaller subproblems, these subproblems can be addressed simultaneously, often leading to considerable reductions in computational time. The architecture underlying this paradigm has evolved from single processors to multi-core CPUs (Central Processing Unit) and GPUs, with capabilities for executing thousands of threads in parallel. (Kessler & Keller, 2007)

While parallelism offers impressive speed-ups, it brings forth challenges, including (Kessler & Keller, 2007):

Synchronization Ensuring tasks execute in a specific order or manage shared resources without conflict.

Data Dependencies Ensuring that tasks that rely on the results of other tasks have their dependencies resolved before execution.

Load Balancing Equitably distributing tasks among processors to avoid computational bottlenecks. These challenges make it essential to adopt appropriate tools and techniques to harness the potential of parallel computing.

Python, though recognized for its simplicity and readability, is not inherently suited for high-performance computing due to its interpreted nature. Numba (Lam et al., 2015), as a Just-In-Time (JIT) compiler that translates Python and NumPy code into machine-level instructions provides a powerful tool for greatly enhancing the performance of Python programs. This allows Python programs to achieve performance comparable to languages like C++ or Fortran. All following technical details on the usage of Numba follow from *Numba documentation* (2023).

One of the groundbreaking features of Numba is its capability to parallelize Python code with minimal intervention. By merely applying the @jit decorator, functions can be greatly optimized. For more explicit parallelization, Numba provides many tools such as @vectorize for element-wise operations on arrays, and the prange function for parallel loop operations.

When leveraging Numba for parallelism, it's essential to (*Numba documentation*, 2023):

Evaluate Performance Gains: Simply adding parallelism does not guarantee improved performance. Measurement with scalability benchmarks is crucial.

Consider Architectural Constraints: The underlying hardware (number of cores, memory bandwidth, etc.) plays a pivotal role in determining performance.

Address Compatibility: Numba doesn't support the full Python language spectrum. Adherence to its subset ensures optimal functioning. NumPy arrays and functions are generally well supported.

However, in practice, simply using the @jit decorator with a function utilizing NumPy can improve performance greatly. Additionally, Numba offers a no-python mode with either using the @njit decorator, or setting `nopython=True`. This ensures that the function does not access the Python C API, leading to faster execution. Parallelization of the @njit function is carried out by setting `parallel=True`. Finally, the `Fastmath` mode is used, which yields faster floating point operations that do not necessarily comply with IEEE 754 standard. However, this is not important to us, as a random number generation is already utilized in the numerical Monte Carlo implementation. (*Numba documentation*, 2023)

In conclusion, while Python's nature might not innately align with high-performance parallel computing, tools like Numba exemplify how the ecosystem is evolving, providing researchers and developers with potent tools to extract maximum computational efficiency.

3.3.2 GPU Acceleration

In the ever-evolving landscape of computational paradigms, Graphics Processing Units (GPUs) have carved a unique niche. Originally designed for rendering graphics, GPUs have found a renewed purpose in general-purpose computing due to their parallel processing capabilities. Unlike traditional CPUs that excel in serial processing tasks, GPUs consist of thousands of small cores suited for handling multiple tasks simultaneously, making them especially beneficial for tasks like data analysis, machine learning, and numerical simulations. (Owens et al., 2008)

The strength of GPU computing lies in its ability to perform a vast number of straightforward calculations in parallel. For operations that can be broken down and executed simultaneously, GPUs offer significant speed-ups. For instance, matrix operations, which are central to many scientific and engineering applications, can be parallelized efficiently on GPUs. (Dehal et al., 2018)

While GPUs offer compelling computational benefits, leveraging their capabilities requires a shift from traditional programming paradigms. This is where libraries like CuPy come into play (Okuta et al., 2017). CuPy is an open-source matrix library that's syntactically very similar to NumPy, but it operates on NVIDIA GPUs using CUDA (Cook, 2012; NVIDIA, Vingelmann, & Fitzek, 2020). With CuPy, the computing power of GPUs can be easily utilized in the Python program without going deep into the details of GPU programming. All following technical details on the usage of CuPy follow from CuPy documentation (*CuPy documentation*, 2023).

Key Considerations with CuPy and GPU Acceleration include:

Memory Management: Unlike traditional CPU memory, GPU memory is often more limited. Developers must be mindful of the memory footprint of their operations to avoid overflows.

Data Transfer Overhead: Moving data between CPU and GPU can introduce latency. For optimal performance, it's advantageous to perform as many operations as possible on the GPU before transferring results back to the CPU.

Compatibility and Dependencies: CuPy relies on NVIDIA's CUDA platform. This means it's primarily compatible with NVIDIA GPUs and requires appropriate CUDA drivers and toolkit installations.

CuPy has substantially simplified the landscape of GPU computing within the Python ecosystem. By providing a familiar interface to developers acquainted with NumPy, it

has smoothed the access to GPU acceleration. As we continue to grapple with increasingly complex computational challenges, tools like CuPy will be instrumental in ensuring Python remains at the forefront of scientific and high-performance computing.

4 Results

In this section the results are presented. First, the obtained performance gains from using parallel computing and GPU acceleration to reduce the computing time are presented. The option calculations are started with the so-called vanilla options, more specifically European call and put options are considered. The obtained results are compared with those obtained from the well-known Cox-Ross-Rubinstein method and Black-Scholes formula as introduced in Section 2 to verify the numerical implementation. Finally, the simulation program is used to calculate prices of Asian options and double barrier options to illustrate the usefulness of the method with more complex options.

4.1 Simulation Performance Benchmark

In this section, the achieved performance boosts are presented using a simple European call option pricing calculation as an example. Benchmarking of three distinct Python-based implementations of PIMC simulations is presented, each varying in computational approach and efficiency as discussed in Section 3. First, a naive Python implementation utilizing a single CPU core and NumPy functions. Second, a Numba parallelized and optimized implementation that utilizes multiple CPU cores with NumPy functions. Third, a GPU accelerated implementation that uses Cuda and CuPy to utilize the computational capabilities of GPUs.

The simulations were benchmarked using the following parameters: 200 time steps, 50,000 paths, and 10,000 iterations for the action functional optimization. The remaining simulation parameters are the same as in the examples throughout the work so far: $S_0 = \$50$, $K = \$55$, $T = 1$, $r = 0.07$ and $\sigma = 0.2$. For the single CPU core run the results were extrapolated by counting the iterations per second of the algorithm from two separate runs to ensure sufficient statistics for an adequate guess on the total simulation runtime.

The results of this benchmark run are presented in Table 2 and further illustrated in Figure 12. From the results it can be seen that the naive serial CPU implementation is highly inefficient, taking hours to complete. The parallelized Numba-based implementation achieves great performance increases at roughly a 577 time speedup. It is worth noting, that while the CPU used has a total of 12 CPU cores, the achieved performance increase greatly surpasses this level of scaling. This is due to the fact that Numba compiles the program into machine-level code as discussed in Section 3, and is thus faster to execute than typical Python programs.

Additionally, the GPU implementation achieves a staggering 1960 time increase in computational speed compared to the naive implementation, and is roughly 4 times faster to execute in the chosen benchmark setting than the parallelized Numba implementation. All of these results clearly highlight the importance of performance optimization in the used simulation code.

Table 2. Performance comparison of different PIMC Python implementations. The parallel CPU implementation achieves approximately a 577 time speedup, and the GPU implementation an impressive 1960 time speedup.

Implementation	**Time (seconds)**	**Speedup**
Naive Single CPU (NumPy)	147000	1x
Parallel CPU (Numba)	255	576.5x
GPU Accelerated (CuPy)	75	1960x

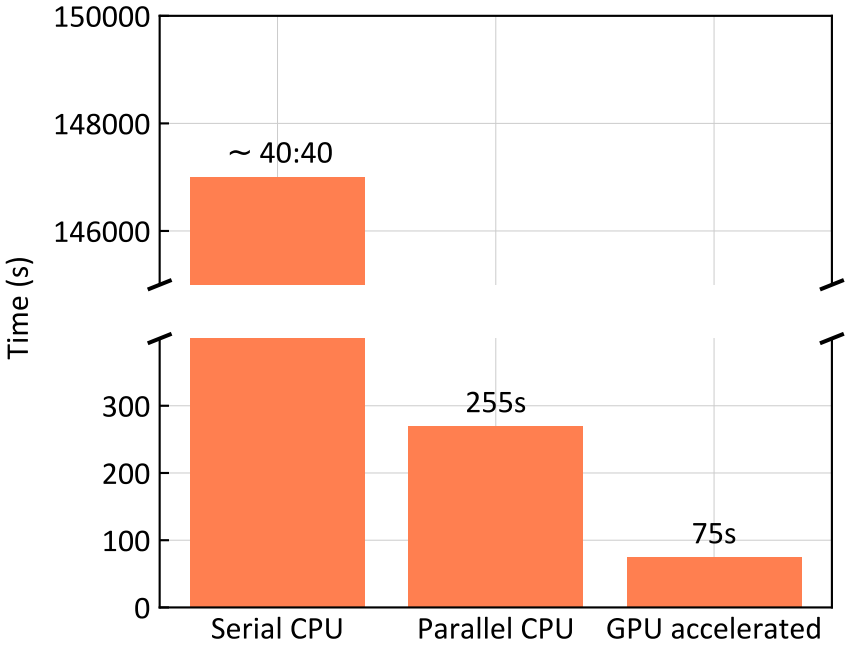


Figure 12. Bar chart showing the runtimes of various Python implementations.

It should be emphasized that in this case study only one set of simulation parameters was considered, and the chosen simulation was quite large in terms of the number of paths, time steps and action optimization iterations. In a smaller simulation, the performance increases won't necessarily be as impressive, especially with the GPU implementation due to the high degree of parallelism (Owens et al., 2008).

As the spec sheets Tables 6 and 7 given in Appendix 2 suggest, the system used in this work is fairly competent, but greatly falls short of supercomputers such as clusters that have thousands of CPUs and GPUs. That being said, for example, the scalability of the parallel CPU implementation can't be tested extensively. However, the simple benchmark works as a guideline and a case study as to why it is important to utilize these more refined programming techniques in numerical simulations. These results also indicate the fact that with today's computing capabilities, more and more refined option pricing methodologies are accessible to traders.

4.2 European Options

In this section, it is verified that the paths generated using the numerical algorithm indeed replicate the Black-Scholes results for European call and put options. This equivalence is rigorously shown in Appendix 1. The simulations are performed using the same parameters as used in the previous benchmarking section with the exception of increasing the number of lattice points on the time domain to 500. Thus, the exception value for the number of proposed changes per lattice point with 10,000 iterations is 20. This set of parameters was found to produce fairly consistent results, keeping in mind the probabilistic nature of the Monte Carlo method.

First of all, the distribution of stock paths, i.e., the propagator in Equation (46) should follow that of the analytical result shown in Appendix 1:

$$p(x_T, T|x_0, 0) = \frac{1}{\sqrt{2\pi\sigma^2 T}} \exp\left\{-\frac{(x_T - x_0)^2}{2\sigma^2 T}\right\}. \tag{54}$$

The logarithmic final stock price distribution from a simulation used in benchmarking is shown in Figure 13. In the figure, the analytical distribution of Equation (54) is plotted with a dashed line. The results agree extremely well, verifying the correctness of the numerical implementation for the simple European option.

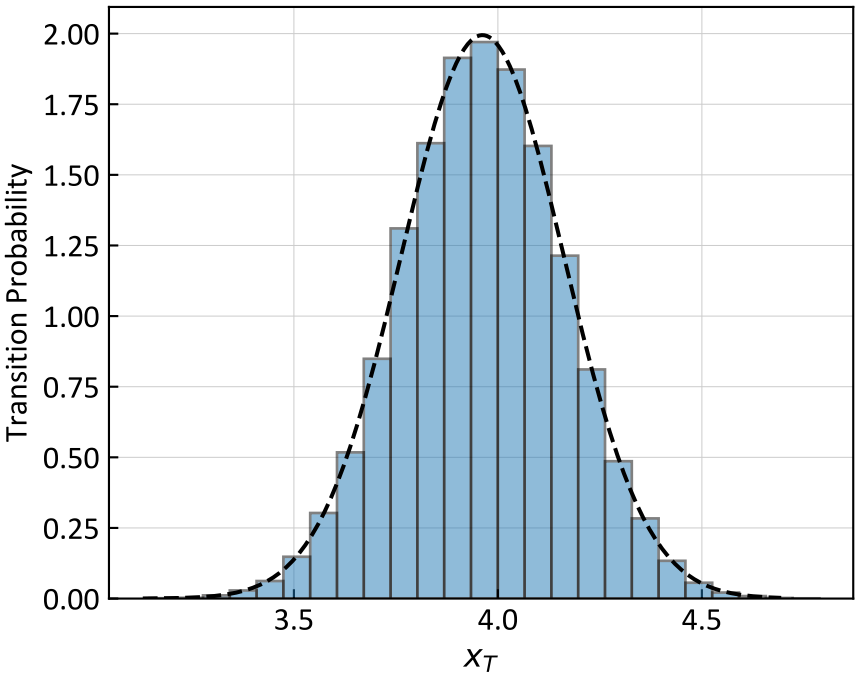


Figure 13. Comparison of the final stock price distribution with the analytical Black-Scholes result. The transition probability illustrated with a histogram shows good agreement with the closed-form analytical solution plotted as a dashed line. Simulation parameters: $S_0 = \$50$, $K = \$55$, $\sigma = 0.2$, $r = 0.07$, $T = 1$, $N = 200$.

Now let us compare the results to those obtained with the closed-form Black-Scholes formulae, Equations (29) and (32). Comparison to Cox-Ross-Rubinstein model as introduced in Section 2 is also carried out using the DerivaGem Excel-based calculator supplied with reference (Hull, 2021). The maximum number of 500 binomial tree steps are used to ensure sufficiently accurate results. Comparison of the call option pricing example with $S_0 = \$50$, $K = \$55$, $\sigma = 0.2$, $r = 0.07$, $T = 1$ using the different methods is shown in Table 3.

Table 3. Comparison of European option values (call and put) using the Black-Scholes formula, Cox-Ross-Rubinstein model, and the path integral approach.

Model	Call option price ($)	Put option price ($)
Black Scholes	3.425	4.709
Cox-Ross-Rubinstein	3.426	4.708
This work	3.405	4.676

It can be seen from the Table 3 that the Black-Scholes and Cox-Ross-Rubinstein agree well as expected in both cases. The path integral method slightly underestimates the value of the call option, but is still reasonably close ($C = 3.405\$$) to the analytical solution given by the Black-Scholes formula with a relative error of 0.57%.

Considering the put option, again, the Black-Scholes and Cox-Ross-Rubinstein are well aligned, and the path integral method slightly underestimates the value at $P = 4.767\$$. Still, the relative error is relatively small at 0.74%.

The obtained results for the European options are aligned with previous numerical studies (Lemmens, Wouters, Tempere, & Foulon, 2008; Montagna et al., 2002) where good, or sometimes even exact agreement has been achieved. Notably, the result agrees with the findings of Lemmens et al. (2008), where the path integral result was found to slightly underestimate the Black-Scholes result for the case when the ratio $K/S_0 \approx 1.1$. It should be noted that the Monte Carlo based results are never exact in practice, and there is always some error present. Additionally, the use of `Fastmath` mode in Numba results in some error.

4.3 Asian Options

The pricing calculations of Asian fixed strike options are carried out similarly to the European options with the exception that the average prices of paths are used in the cal-

culations instead of the final stock prices as indicated by Equation (3). Since there is no general closed-form formula, the traditional Monte Carlo solver from DerivaGem (Hull, 2021) is used for the comparison of results with the maximum values of 200 time steps and 10,000 simulation runs as parameters. For the path integral implementation, the same simulation parameters are used as in the previous calculations and the general parameters are kept the same: $S_0 = \$50$, $K = \$55$, $\sigma = 0.2$, $r = 0.07$, $T = 1$.

It should be emphasized that for Asian options, there are no standardized methods as universally recognized and as widely used as the Black-Scholes model for European options. Thus, there is no single correct value to compare the obtained results with. Commonly used pricing methods include traditional Monte Carlo methods and finite difference methods (Hull, 2021). In Table 4, the comparison between the traditional Monte Carlo method and the path integral approach in pricing Asian call and put options is shown.

Table 4. Comparison of Asian option values (call and put) using Monte Carlo method and the path integral approach.

Model	**Asian call option price ($)**	**Asian put option price ($)**
Monte Carlo	1.122	4.162
This work	1.051	4.240

The methods agree fairly well in both cases. Interestingly, the deviation of the path integral approach is not systematic in comparison to traditional Monte Carlo: for the call option the path integral result is lower than the result obtained using Monte Carlo whereas in the case of put option the path integral approach prices the option higher. However, both methods give results in the same regime, and the error can be induced simply by the fact that both methods are probabilistic in nature and thus not exact.

4.4 Barrier Options

In this section, the calculations for double knock-out barrier options are presented using both implemented methods discussed in Section 3: Firstly, a modified Hamiltonian of a particle in a square potential well. Secondly, by limiting the range of proposed changes during the sampling phase in the action optimization.

Similarly to Asian options, there is a lack of universally recognized standard pricing model. The results obtained in this study are thus compared to those obtained using the numerical scheme and Excel application provided in reference (Kunitomo & Ikeda, 1992), which is made for the case of a curved barrier boundary. However, by setting curvature terms of the model equal to 0, the method can be straightforwardly used for the standard barrier options.

For the double barrier calculation, a European option with the following parameters is used: $S_0 = \$100$, $K = \$90$, $r = 0.1$, $\sigma = 0.3$, $T = 0.5$, $B_l = \$50$ and $B_u = \$150$. The path integral calculations are performed again with the benchmark settings. The results are presented in Table 5.

Table 5. Comparison of double barrier option values (call and put) using the numerical scheme of Kunitomo and Ikeda (1992), the modified Hamiltonian and the restricted sampling.

Model	**Call option price ($)**	**Put option price ($)**
Kunitomo and Ikeda (1992)	12.8311	2.6187
Modified Hamiltonian	16.0584	2.6348
Restricted sampling	13.0717	2.5722

For the put option prices, all methods agree reasonably well. However, in call option valuation the results suggest that the modified Hamiltonian approach overvalues the option.

5 Conclusions

This study embarks on an intriguing exploration of the intersection between quantum mechanical path integrals and option pricing in finance. Utilizing Path Integral Monte Carlo (PIMC) simulations, an innovative approach rooted in the first principles of quantum mechanics, this work opens new avenues in financial modeling (Feynman, 1948; Kleinert, 2009). The method particularly excels in pricing various types of options, such as European, Asian, and barrier options, demonstrating its versatility and robustness. The path integral method offers a viable alternative in complex market scenarios where traditional methods can fall short.

While a comparative analysis with traditional models such as the Black-Scholes model (Black & Scholes, 1973; Merton, 1973) and the Cox-Ross-Rubinstein model (Cox et al., 1979) has been instrumental in establishing the accuracy and reliability of the path integral method, the true potential lies in the method's demonstrated adaptability and effectiveness in pricing more complex options. In this study, the method was introduced theoretically and applied successfully in the pricing of European, Asian and double barrier options, also answering the first research question. These results underscore the path integral method's potential as an adaptable and comprehensive tool in financial modeling, a critical asset in today's rapidly evolving financial markets.

The practical implications of this work are profound, particularly in financial modeling and option pricing. With financial instruments growing in complexity, the ability to precisely price these instruments using advanced simulations is paramount. This study highlights the vital role of computational methods in understanding market dynamics, enhancing risk management, and shaping effective investment strategies. In a broader context, these advancements in financial modeling could be instrumental in identifying and mitigating risks that lead to financial crises. By offering more accurate and flexible pricing tools, this approach could help in better forecasting market trends and valuing complex financial products, thus contributing to a more stable and resilient financial system.

In terms of performance optimization, the study demonstrates the significant computational advancements achievable through parallel computing and GPU acceleration, answering the second research question. These methods are shown to play a crucial role in efficiently managing large-scale simulations. By integrating Python with tools like Numba (Lam et al., 2015) and CuPy (Okuta et al., 2017), the study successfully bridges the gap between high-level programming ease and computational intensity, marking a significant leap forward in computational finance. These advancements enable the utilization of previously considered inefficient methods, leveraging increased computational capabilities.

This study was limited to verifying the method using European options and illustrating the method's capability on pricing Asian and double barrier options. Looking forward, this study smooths the path for further research in refining the path integral approach for more complex derivatives and incorporating sophisticated risk factors. The potential for additional computational optimizations and the advent of emerging technologies hold promising prospects for the future of financial engineering. The methods presented in this study could be trivially extended to include, for example, the stochastic nature of volatility and risk-free rate.

In summary, this study presents a promising methodology in financial engineering, showcasing the efficacy of path integral methods in option pricing, as well as the importance of computational optimization. The method offers a fresh perspective and valuable insights that bridge quantum mechanics and financial modeling, with far-reaching implications for both academic research and practical finance applications.

Furthermore, this study exemplifies the increased significance of interdisciplinary research, particularly in the realm of econophysics. The merging of physics, especially quantum mechanics, with economics and finance potentially opens up the ground for increasingly innovative approaches and solutions. This mixture of different disciplines can lead to a deeper understanding of complex market dynamics, transcending the limitations of traditional economic theories. The insights from physics can introduce new concep-

tual frameworks and analytical tools, potentially leading to breakthroughs in predicting market behaviors and managing financial risks. The success of such multidisciplinary approaches could additionally inspire more collaborative research endeavors, harnessing the strengths of various fields to tackle some of the difficult challenges in the financial world. In a speculative light, the continued evolution of econophysics may well end up redefining the landscape of financial analysis and risk management, underscoring the transformative power of interdisciplinary research.

Bibliography

Atzberger, P. J. (2006). The black-scholes-merton approach to pricing options.. Retrieved from https://api.semanticscholar.org/CorpusID:1726590

Baaquie, B. E. (1997). A Path Integral Approach to Option Pricing with Stochastic Volatility: Some Exact Results. *Journal de Physique I*, *7*(12), 1733-1753.

Baaquie, B. E., Corianò, C., & Srikant, M. (2004). Hamiltonian and potentials in derivative pricing models: exact results and lattice simulations. *Physica A: Statistical Mechanics and its Applications*, *334*(3), 531-557. https://doi.org/https://doi.org/10.1016/j.physa.2003.10.080

Black, F., & Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, *81*(3), 637–654.

Bormetti, G., Montagna, G., Moreni, N., & Nicrosini, O. (2006). Pricing exotic options in a path integral approach. *Quantitative Finance*, *6*(1), 55-66. https://doi.org/10.1080/14697680500510878

Bustamante, M., & Contreras, M. (2016). Multi-asset black–scholes model as a variable second class constrained dynamical system. *Physica A: Statistical Mechanics and its Applications*, *457*, 540-572. https://doi.org/https://doi.org/10.1016/j.physa.2016.03.063

Capuozzo, P., Panella, E., Schettini Gherardini, T., & Vvedensky, D. D. (2021). Path integral monte carlo method for option pricing. *Physica A: Statistical Mechanics and its Applications*, *581*, 126231. https://doi.org/https://doi.org/10.1016/j.physa.2021.126231

Chen, Q., & Guo, C. (2023). *Path integral method for pricing proportional step double-barrier option with time dependent parameters*.

Contreras, M., Pellicer, R., Villena, M., & Ruiz, A. (2010). A quantum model of option pricing: When Black-Scholes meets Schrödinger and its semi-classical limit. *Physica A Statistical Mechanics and its Applications*, *389*(23), 5447-5459.

Cook, S. (2012). *Cuda programming: A developer's guide to parallel computing with gpus (1st edition)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Cox, J. C., Ross, S. A., & Rubinstein, M. (1979). Option pricing: A simplified approach. *Journal of Financial Economics*, *7*(3), 229-263. https://doi.org/https://doi.org/10.1016/0304-405X(79)90015-1

Cupy documentation. (2023). Retrieved from [2023-11-04]https://docs.cupy.dev/en/stable/

Dash, J. (1988). *Path integrals and options, part i* (Tekn. rap. nro CPT-88/PE.2206). Centre National de la Recherche Scientifique (CNRS).

Dash, J. (1989). *Path integrals and options, part ii* (Tekn. rap. nro CPT-89/PE.2333). Centre National de la Recherche Scientifique (CNRS).

Dehal, R. S., Munjal, C., Ansari, A. A., & Kushwaha, A. S. (2018). Gpu computing revolution: Cuda. *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, 197-201. https://doi.org/10.1109/ICACCCN.2018.8748495

Devreese, J., Lemmens, D., & Tempere, J. (2010). Path integral approach to asian options in the black–scholes model. *Physica A: Statistical Mechanics and its Applications*, *389*, 780–788.

Dirac, P. A. M. (1933). The Lagrangian in quantum mechanics. *Phys. Z. Sowjetunion*, *3*, 64–72.

Dürr, D., & Bach, A. (1978). The onsager-machlup function as lagrangian for the most probable path of a diffusion process. *Communications in Mathematical Physics*, *60*(2), 153-170. https://doi.org/10.1007/BF01609446

Feynman, R. P. (1948). Space-time approach to non-relativistic quantum mechanics. *Rev. Mod. Phys.*, *20*, 367–387. https://doi.org/10.1103/RevModPhys.20.367

Goldstein, H., Poole, C., & Safko, J. (2002). *Classical mechanics (3th edition)*. Addison Wesley.

Greenberger, M. (2010). *The Role of Derivatives in the Financial Crisis –*. Retrieved from [2023-11-04]https://digitalcommons.law.umaryland.edu/cong_test/37/

Griffiths, D. J., & Schroeter, D. F. (2018). *Introduction to quantum mechanics (3)*. Cambridge University Press.

Guarch Termens, M. E. (2022). *Pricing methods for barrier options*. Retrieved from http://hdl.handle.net/2445/189204

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*, 357–362.

Harrison, R. (2010). Introduction to monte carlo simulation. *AIP conference proceedings*, *1204*, 17-21.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, *57*(1), 97-109. https://doi.org/10.1093/biomet/57.1.97

Hull, J. (2021). *Options, futures, and other derivatives, global edition*. Pearson Education. Retrieved from https://books.google.fi/books?id=sdg2EAAAQBAJ

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Kessler, C. W., & Keller, J. (2007). Models for parallel computing : Review and perspectives.. Retrieved from https://api.semanticscholar.org/CorpusID:15463555

Kleinert, H. (2009). *Path integrals in quantum mechanics, statistics, polymer physics, and financial markets (5th edition)*. WORLD SCIENTIFIC. https://doi.org/10.1142/7305

Kunitomo, N., & Ikeda, M. (1992). Pricing options with curved boundaries. *Mathematical Finance*, *2*(4), 275-298. https://doi.org/https://doi.org/10.1111/j.1467-9965.1992.tb00033.x

Kwok, Y. (2008). *Mathematical models of financial derivatives*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-68688-0

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler. *Proc. Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6. https://doi.org/10.1145/2833157.2833162

Lemmens, D., Wouters, M., Tempere, J., & Foulon, S. (2008). Path integral approach to closed-form option pricing formulas with applications to stochastic volatility and interest rate models. *Phys. Rev. E*, *78*, 016101. https://doi.org/10.1103/PhysRevE.78.016101

Linetsky, V. (1997). The Path Integral Approach to Financial Modeling and Options Pricing. *Computational Economics*, *11*(1-2), 129-163. Retrieved from http://dx.doi.org/10.1023/A:1008658226761

Maverick, J. (2022). *How big is the derivatives market?* Investopedia. Retrieved from [2023-11-04]https://www.investopedia.com/ask/answers/052715/

how-big-derivatives-market.asp

Merton, R. C. (1973). Theory of rational option pricing. *The Bell Journal of Economics and Management Science*, *4*(1), 141–183.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (2004). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, *21*(6), 1087-1092. https://doi.org/10.1063/1.1699114

Montagna, G., Nicrosini, O., & Moreni, N. (2002). A path integral way to option pricing. *Physica A: Statistical Mechanics and its Applications*, *310*(3-4), 450–466. https://doi.org/10.1016/s0378-4371(02)00796-3

Numba documentation. (2023). Retrieved from [2023-11-04]https://numba.readthedocs.io/en/stable/

NVIDIA, Vingelmann, P., & Fitzek, F. H. (2020). *Cuda, release: 10.2.89*. Retrieved from [2023-11-04]https://developer.nvidia.com/cuda-toolkit

Okuta, R., Unno, Y., Nishino, D., Hido, S., & Loomis, C. (2017). Cupy: A numpy-compatible library for nvidia gpu calculations. *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*. Retrieved from http://learningsys.org/nips17/assets/papers/paper_16.pdf

Olver, P. (2013). *Introduction to partial differential equations*. Springer. https://doi.org/10.1007/978-3-319-02099-0

Otc derivatives statistics at end-december 2022. (2023). Retrieved from [2023-11-04]https://www.bis.org/publ/otc_hy2305.htm

Owens, J. D., Houston, M., Luebke, D., Green, S., Stone, J. E., & Phillips, J. C. (2008). Gpu computing. *Proceedings of the IEEE*, *96*(5), 879-899.

Saito-Chung, D. (2023). *Options trading hits 50-year mark; can industry innovation help more investors grow their wealth?* Retrieved from [2023-11-04]https://www.investors.com/research/options/options-trading-today-after-50-years-of-growth/

Sakurai, J. J., & Napolitano, J. (2020). *Modern quantum mechanics (3th edition)*. Cambridge University Press.

Sharpe, W. (1978). *Investments*. Prentice-Hall. Retrieved from https://books.google.fi/

books?id=4UgWAQAAMAAJ

Thomopoulos, N. (2013). *Essentials of monte carlo simulation: Statistical methods for building simulation models*. Springer. https://doi.org/10.1007/978-1-4614-6022-0

Utama, B., & Purqon, A. (2016). Feynman path integral application on deriving black-scholes diffusion equation for european option pricing. *Journal of Physics: Conference Series*, *739*(1), 012021. https://doi.org/10.1088/1742-6596/739/1/012021

Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley, CA: CreateSpace.

Wiener, N., & Masani, P. (1976). *Norbert wiener - collected works: Mathematical philosophy and foundations - potential theory - brownian movement, wiener integrals, ergodic and chaos theories, turbulence and statistical mechanics*. MIT Press.

Witzany, J. (2020). *Derivatives, theory and practice of trading, valuation, and risk management (1st edition)*. Springer.

Appendices

Appendix 1. Derivation of the Black-Scholes Formula from Path Integrals

In this section, it is shown that the Feynman-Kac formula with Black-Scholes Lagrangian yields exactly the Black-Scholes closed-form result for European options following closely the work of Utama and Purqon (2016). In Section 2, the Feynman-Kac formula Equation (41) for a European call option was defined as

$$C(x_0, 0) = e^{-rT} \int_{-\infty}^{\infty} \int_{x_0}^{x_T} C(e^{x_T}) e^{-S_{\text{BS}}[x(t)]} \mathcal{D}[x_T] dx_T, \tag{55}$$

where the Black-Scholes action functional was written as

$$S_{\text{BS}} = \underbrace{\int_0^T \frac{1}{2\sigma^2} \left(\frac{dx}{dt}\right)^2 dt}_{=S_0[x(t)]} - \frac{\mu}{\sigma^2}(x_T - x_0) + \frac{\mu^2 T}{2\sigma^2}, \tag{56}$$

and where S_0 represents the action of a *zero drift process*.

From Equation (56), the constant terms can be further separated, so that the option value is written as

$$C(x_0, 0) = e^{-rT} \int_{-\infty}^{\infty} \int_{x_0}^{x_T} C(e^{x_T}) e^{\frac{\mu}{\sigma^2}(x_T - x_0) - \mu^2 T/2\sigma^2} e^{-S_0[x(t)]} \mathcal{D}[x_T] dx_T, \tag{57}$$

Recalling the discretization of the transition probability and expressing the integral over

paths using Equation (46) it can be show that

$$\begin{aligned}
p(x_T, T|x_0, 0) &= \int_{x_0}^{x_T} \mathcal{D}[x_T] e^{-S_{\text{BS}}} \\
&= \lim_{N\to\infty} \left[\prod_{k=0}^{N-1} \int_{-\infty}^{\infty} \frac{dx_k}{\sqrt{2\pi\sigma^2\delta t_k}} \right] e^{-S_{\text{BS}}} \\
&= \lim_{N\to\infty} \underbrace{\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty}}_{N-1} \exp\left\{ -\sum_{k=0}^{N-1} \frac{\Delta t}{2\sigma^2} \left(\frac{x_{k+1} - x_k}{\Delta t} \right)^2 \right\} \\
&\qquad \times \frac{1}{\sqrt{2\pi\sigma^2\delta t}} \times \frac{dx_1}{\sqrt{2\pi\sigma^2\delta t}} \times \cdots \times \frac{dx_{N-1}}{\sqrt{2\pi\sigma^2\delta t}} \\
&= \lim_{N\to\infty} \frac{1}{\sqrt{2\pi\sigma^2 N\delta t}} \exp\left\{ -\frac{(x_N - x_0)^2}{2\sigma^2 N\delta t} \right\} \\
&= \frac{1}{\sqrt{2\pi\sigma^2 T}} \exp\left\{ -\frac{(x_T - x_0)^2}{2\sigma^2 T} \right\},
\end{aligned} \tag{58}$$

where the following Gaussian integral was used

$$\int_{-\infty}^{\infty} e^{-a(p-x)^2 - b(x-q)^2} dx = \sqrt{\frac{\pi}{a+b}} \exp\left\{ -\frac{ab}{a+b}(p-q)^2 \right\}. \tag{59}$$

Putting the pieces from Equations (57) and (58) together, the price of the option can be expressed as

$$C(x_0, t) = e^{-rT} \int_{-\infty}^{\infty} C(e^{x_T}) \frac{1}{\sqrt{2\pi\sigma^2 T}} \exp\left\{ -\frac{(x_T - x_0 - \mu T)^2}{2\sigma^2 T} \right\} dx_T, \tag{60}$$

from which the Gaussian integral can again be exactly solved, yielding exactly the Black-Scholes formula, Equation (29)

$$C = S\mathcal{N}(d_1) - Ke^{-rT}\mathcal{N}(d_2), \tag{61}$$

where

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right) T}{\sigma\sqrt{T}} \tag{62}$$

$$d_2 = d_1 - \sigma\sqrt{T} \tag{63}$$

and the payoff function

$$C(e^{x_T}) = \max\{0, e^{x_T} - K\} \tag{64}$$

was used.

Appendix 2. System Specifications

The system used in the calculations of this work has an Intel CPU and an Nvidia GPU. The CPU used in this work is a 12-core Intel Core i7-12700K. The spec sheet of the CPU is given in Table 6.

Table 6. Intel Core i7-12700K CPU Specifications.

Total Cores	12
Number of Performance-cores	8
Number of Efficient-cores	4
Total Threads	20
Max Turbo Frequency	5.00 GHz
Intel Turbo Boost Max Technology 3.0 Frequency	5.00 GHz
Performance-core Max Turbo Frequency	4.90 GHz
Efficient-core Max Turbo Frequency	3.80 GHz
Performance-core Base Frequency	3.60 GHz
Efficient-core Base Frequency	2.70 GHz
Cache	25 MB Intel Smart Cache
Total L2 Cache	12 MB

The GPU used in this work is an Nvidia GeForce RTX 3070, the spec sheet of which is presented in Table 7.

Table 7. GeForce RTX 3070 GPU Specifications.

Device	GeForce RTX 3070 GPU
Amount of Memory	10 GB
Number of CUDA Cores	5888
CPU Boost Clock Speed	1725 MHz
Effective Memory Clock Rate	14 GHz
Memory Bus Width	256-bit
Memory Bandwidth	448 GB/s
Transistor Count	17.4 billion