Z-transform and machine learning techniques for the pricing of discretely monitored path-dependent options

May 2025

Benjamin Loveless

A thesis submitted for the degree of Doctor of Philosophy in Computer Science, University College London

I, Benjamin Loveless confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Supervisors

Professor Guido Germano, Dr Carolyn E. Phelan

Department of Computer Science

University College London

This thesis is dedicated to Danielle.

Truly, words are not enough.

Sometimes, none need be said.

Thank you.

Abstract

This thesis focuses on optimising exotic option pricing using integral and discrete transforms, with a primary emphasis on discrete monitoring of the option. It also explores the broader application of discrete transforms to general problems encountered in physics and signal processing.

Numerous methods for computing the inverse z-transform are compared to find an alternative to the more popular methods seen in financial applications. Popular modern techniques have an inherent error floor that cannot be improved; thus, a new approach is required and developed in this work. Furthermore, series acceleration is explored to improve CPU times, where the concern is not just raw processing time, but also CPU versus error performance.

Pricing algorithms for double barrier and α-quantile are presented, utilising a new methodology that focuses on the inverse z-transform to achieve a machine-accurate solution. This error level was previously unachievable due to the aforementioned error floor. Series acceleration techniques are further extended to enhance method performance. It can also be demonstrated that the pricing methodology is exponentially convergent in the case of α-quantile options, a result previously unverified.

Finally, a new approach is presented that utilises deep learning techniques to learn from synthetic data generated by numerical models and predict option prices, as well as the Greeks. An optimised method for parameter searching is also presented. The process offers high performance and is a viable alternative, particularly in scenarios with multiple monitoring dates or multiple repricing events.

Impact Statement

The motivating application for the methods explored and enhanced in this thesis is the fast, efficient and accurate pricing of financial option contracts. This topic is of resounding importance within a financial setting and a deep and interesting topic within academia. Options pricing is still vital for regulatory, trading, and risk purposes, with the legacy of the global economic crisis still weighing heavily on the industry. In addition, with recent economic turbulence rippling through markets, the need for optimised pricing techniques is more potent than ever. The works presented in this thesis will be published within a very short timeline following this submission. Several presentations have been made at international conferences to both academics and practitioners.

The works in this thesis have a broader audience beyond their direct financial applications. Fields such as signal processing, statistical physics, and fluid mechanics can all benefit from enhancements presented in the computation of the inverse z-transform presented here. The open-source code repositories accompanying this thesis are also helpful and immediately applicable to any field interested in the efficient computation of discrete integral transforms.

Acknowledgements

The support provided by the Engineering and Physical Sciences Research Council (EPSRC) in funding the UK Centre for Doctoral Training in Financial Computing and Analytics is gratefully acknowledged.

I sincerely thank my supervisor, Professor Guido Germano, for all his support and help. I am also very grateful to Dr Carolyn Phelan, whose comments, suggestions and time were invaluable throughout the programme.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This work is motivated by the requirement to value exotic options within financial markets. Specifically, the pricing of path-dependent options based upon a probabilistic distribution reflecting the underlying asset dynamics upon which the option is written is under examination. The seminal works by Black and Scholes [1973] and Merton [1973] created an initial closed-form framework that was mathematically tractable for evaluating vanilla European options. In this framework, mathematical tractability is favoured over accurate asset modelling with the restriction of the underlying model to geometric Brownian motion. As more complex options have entered the market since the early 1970s, and the desire to capture asset dynamics more accurately has grown, the industry has sought viable alternatives. One such alternative is Lévy processes. These models capture well-known asset behaviour features, such as fat tails and price jumps, while maintaining constant parameters.

As the range of options contracts has increased and the industry seeks to model more realistic asset dynamics, so has the need to explore alternatives to closed-form solutions. External regulations requiring specific exotic options to be handled discretely have been the standard for some time. This results in path-dependent options, such as barriers and Asian options, being monitored discretely rather than the more mathematically tractable continuous approach. Typical closed-form frameworks assume the ability to monitor continuously, rendering them obsolete. Invariably, these factors enforce the need for numerical approaches to pricing options. As such, recent literature primarily concerns accurate and efficient numerical methods for pricing.

Standard numerical pricing techniques, such as Monte Carlo and finite difference, are the most widely used within the industry. However, there are noted issues with both methods, which can make them suboptimal choices. Error convergence with respect to computational time is the most significant issue, where convergence is polynomial for finite difference methods and sub-polynomial for Monte Carlo methods, respectively. While this can be improved using various techniques, it is a particular issue for path-dependent options where computational time also increases linearly with the number of monitoring dates.

The alternative is the Fourier-z framework, but it has flaws. Two methods stand out as the most attractive: the technique developed by Feng and Linetsky [2008] and the one by Fusai et al. [2016]. The former produces excellent error results using sinc-based fast Hilbert transforms, applying the barrier at each monitoring step. The flaw is in the continued linear relationship between the number of monitoring dates and the speed of error convergence. The latter removes this dependency by utilising the z-transform to collapse the time domain, meaning the computational time is now independent of the monitoring dates but dependent on the numerical inversion of the inverse z-transform. Additionally, while Feng and Lintesky's method achieves machine accuracy, E-16, Fusai's method has a distinct error floor caused by the numerical inversion of the z-transform.

Therefore, this work focuses on the continued development of the Fourier-z framework for path-dependent option pricing. Within this focus, there are three overarching goals. The first is to remove the error floor present in Fusai's method. The second is a detailed error analysis and performance review of the numerical inverse z-transform, as well as the applied series acceleration techniques. This task has yet to be satisfactorily addressed within the literature. The final one is to provide access to these methods; typically, even well-written works can be impenetrable to practitioners. The aim is to address this with detailed, open-source codebases.

While the primary venue of exploration is the field of financial derivative pricing, the works in this thesis are sufficiently general to extend far beyond it. The Fourier-z and Spitzer frameworks Spitzer [1957] have applications in probability, stochastics, and queuing theory. The numerical z-transforms have a broad field of applicability, ranging from signal processing to fluid dynamics and statistical mechanics. Improvements herein benefit multiple disciplines. Likewise, at a later stage, we explore deep learning methods as universal approximators. While the application is derivative pricing, this extends naturally to any field requiring numerical approximation where synthetic data is available.

Chapter 2 provides a technical introduction to the topics explored in this thesis. It is written to be self-contained with sufficient information to understand the core concepts. Due to the breadth of the topics within the relevant literature, only details directly related to the work within this thesis have been provided. Where omissions occur, an extensive list of references has been included for the interested reader to explore further.

Chapter 3 presents detailed error and performance benchmarking for various numerical inverse z-transforms in the literature. Extensions that aim to expose selected inversion methods to series acceleration techniques are also provided, along with a comprehensive discussion and numerical results.

Chapter 4 presents a performance enhancement for the Fourier-z pricing framework presented by Feng and Linetsky [2008] and Fusai et al. [2016] for double barrier and α-quantile options. The inherent error floor is overcome, and monitoring date independence is retained. Additionally, the error convergence for the α-quantile option is demonstrated to be of exponential order. The work presented by Phelan et al. [2020] could only guarantee polynomial convergence prior to this point.

Finally, Chapter 5 presents a deep learning approach to pricing exotic options. Deep learning has grown over the past few years due to increased accessibility to open-source libraries and the democratisation of fairly powerful computer hardware. This approach is examined for its performance and benchmarked against the Fourier-z framework to quantify its computational performance, a rare occurrence in the literature. More typical is the comparison to unoptimised numerical methods, which is often unrevealing in examining the method's viability.

The chapters within this thesis constitute three working papers, one of which has been submitted for publication. Several presentations have also been made at conferences, including the Workshop on Stochastic and Partial Differential Equation Methods in Finance and Economics in Rome and the Workshop on Quantitative Finance in Zurich.

Chapter 2

Technical Background

This chapter provides the technical background to the work presented in this thesis and reviews current and relevant literature. Due to the expansive nature of the topics in this thesis, a curated selection of information is provided to make this thesis as self-contained as possible. Extensive references are provided to direct the interested reader to explore areas in further depth, where required. An initial presentation of the financial contracts of interest is followed by the relevant analytical background to the work contained in this thesis. A review of the existing literature on pricing methods is presented, including numerical methods, some of which are extended in later chapters.

2.1 Exotic options

Financial option contracts, also known as contingent claims, are those whose value is derived from an underlying asset, such as a stock, index, or commodity. A basic, or vanilla, option usually features standard payoff structures with maturities and strikes that follow well-defined patterns. A typical example is the vanilla call option, where the holder has the right but not the obligation to exercise the option to purchase the underlying at expiry. By contrast, exotic options add more intricate terms or payoff dependencies. They may incorporate multiple underlying assets, barriers that activate or nullify payoffs, or even path dependence, where the final value depends on a portion of the (or entire) history of the underlying asset rather than merely its final price at expiration. These structural differences make pricing, hedging, and risk management notably more complex than for vanilla options. Their usage and value derive from the need for complex positions, which can be created using exotic options, often in conjunction with other financial contracts. These positions typically arise from non-linear financial contracts, complex portfolios, or a general focus on market alpha or beta.

A significant subset of exotic options are path-dependent, meaning the option's payoff is influenced by the path the underlying asset takes over time. Historical averages or hitting certain price thresholds become relevant in these cases. For instance, an Asian option pays off based on the average price of the underlying over a specified period, while lookback and barrier options require the minimum, maximum, or boundary-tracking of prices throughout the option's life. Such complexity requires more advanced mathematical or computational techniques to value these products accurately. The following subsections will delve into the contracts of interest within this thesis and a detailed analysis of how the modelling of financial assets is approached.

In practice, path-dependent options are monitored discretely rather than continuously. This shift from continuous monitoring to realistic, periodic observations arises partly from regulatory requirements demanding regular, discrete-time assessments of positions for risk management and reporting. Under frameworks like Basel, institutions are required to demonstrate robust mark-to-market practices at specific intervals, rendering continuous monitoring impossible. As a result, pricing models that assume continuous observation can misstate the risk or value of these contracts if they do not adjust for discrete sampling. Accounting for discrete monitoring not only changes the mathematical treatment (since events can occur only at certain times rather than at any instant) but also affects numerical methods. For instance, Monte Carlo schemes must reflect the observation schedule under consideration. This reality makes the handling of discrete monitoring critical.

2.1.1 Contracts of interest

Two flavours of exotic options are frequently of interest within this thesis. First is the barrier option, the second is the α-quantile option.

Barrier options are a class of exotic derivatives whose payoff depends not only on the terminal price of the underlying asset but also on whether the price touches a specified barrier level during the contract's life. For example, "knock-out" barrier options immediately become worthless once the underlying hits the barrier, whereas "knock-in" options only become active if the barrier is breached. Variants such as up-and-out and down-and-in are distinguished by whether the barrier is set above or below the current price and the style of barrier applied. The introduction of such triggers allows for more customised payoff structures, typically resulting in lower premiums compared to standard (vanilla) options, given the added conditional element. Additionally, multiple barriers can be applied in a single contract.

A related variant, the α-quantile option, is designed to pay off depending on whether the underlying price distribution falls above or below a certain quantile threshold. These contracts can be seen as a specialised form of binary or digital option, but with a payoff keyed to specific quantile-based events, such as exceeding the 95th percentile of possible outcomes. Because α-quantile options isolate specific sections of the underlying asset's return distribution, they can be precise tools for capturing tail risk or exploiting asymmetric views on volatility. Their payoff structure can be implemented so that it is contingent not only on absolute price levels but also on the price ranking relative to other potential outcomes, allowing for more nuanced risk and return profiles.

Both barrier and α-quantile options for portfolio managers provide targeted ways to manage risk and enhance yields. Barrier options allow investors to reduce costs compared to vanilla options by pricing in the chance that the option will be rendered inactive if a barrier is breached. This feature is particularly appealing when an investor has strong views about the likelihood of the underlying remaining within a specific price range. Similarly, α-quantile options enable managers to position for extreme market scenarios or hedge particular portions of the return distribution. In doing so, they address the shortcomings of standard contracts, which often do not capture distributional nuances or tail risk.

2.1.2 Stochastic processes and modelling assets

The application of stochastic processes to model financial assets dates back to the pioneering work of Louis Bachelier, who, in his 1900 doctoral thesis, Bachelier [1900], introduced Brownian motion as a tool for describing fluctuations in stock prices. Later developments in the 1970s, particularly the Black-Scholes-Merton framework, Black and Scholes [1973], Merton [1973], solidified the relevance of randomness and probability in describing market dynamics.

Many advancements and improvements in asset modelling have been introduced since then. The original log-normal stochastic model governing the underlying asset evolution is given by the geometric Brownian motion (GBM)

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t), \tag{2.1}$$

where S is the underlying price, μ is the drift, σ is the asset volatility and $W(t)$ is the driving Wiener process or Brownian motion. To ensure a self-contained thesis, some of the fundamental characteristics of the Brownian motion are given here for completeness:

1. $X(0) = 0$.

2. The process is almost surely continuous.

3. Increments are independent, for any $0 < t_1 < t_2 < \ldots < t_n < \infty$, $X(t_n) - X(t_{n-1}), X(t_{n-1}) - x(t_{n-2}), \ldots$ are independent.

4. Increments are stationary, for any $s < t$, $X(t) - X(s)$ has the same probability distribution as $X(t - s)$.

To model financial assets, we typically use exponential processes of the form

$$S(t) = S_0 e^{X(t)}, \tag{2.2}$$

where S_0 is the initial value of $S(t)$. This exponential form is beneficial as it guarantees non-negative asset prices while modelling returns as (approximately) normally distributed in the log domain, aligning well with observable market behaviour.

A key limitation of the GBM is that it assumes constant volatility and log-normal returns, failing to capture empirically observed features such as heavy tails, volatility clustering, and sudden market price jumps. An alternative approach is to combine the diffusion process of a model, such as GBM, with a jump process. Adding a jump process allows the modelling of extreme price movements. An additional benefit is the production of excess kurtosis of the underlying process distribution, which better reflects market return distributions. The category of diffusion process and those combined with an additional jump process are referred to as Lévy processes; a comprehensive review of this topic is presented in Cont and Tankov [2004].

The most well-known of the Lévy process is the aforementioned geometric Brownian motion, which has increments governed by a normal distribution with mean 0 and variance $t - s$. Of similar renown is the Poisson process, whose increments are governed by the Poisson distribution and have mean and variance equal to $\lambda(t - s)$. Thus, the most recognisable jump-diffusion process is a combination of the two. The Merton jump-diffusion process combines a Brownian motion diffusion process with a compound Poisson process, resulting in jumps occurring at random intervals.

A different approach, with subsequent advancements in model accuracy at the expense of numerical simplicity, involves adding an additional stochastic process. Typical choices of which variable to model using this additional process are the risk-free rate r or the instantaneous volatility, often denoted σ. The Heston model Heston [1993] and SABR Hagan et al. [2002] are examples of stochastic volatility models. These models are well-known for their increased accuracy and the ability to capture volatility smiles, albeit at the cost of calibration complexity. These advantages are weighted by an additional random process to manage. To highlight the difference between a single-process model and a multi-process model, Heston's SDE is given below for brevity

$$dS(t) = \mu S(t)dt + \sqrt{v}S(t)dW_1(t), \tag{2.3}$$

$$dv(t) = \kappa(\theta - v(t))dt + \sigma\sqrt{v(t)}dW_2(t), \tag{2.4}$$

where θ is the long-run average variance of the price, κ is the mean reversion rate of $v(t)$ and σ is the volatility of the volatility. An additional ρ parameter governs the correlation between the two Wiener processes.

Due to the added complexity of a secondary process, stochastic volatility processes are generally considered an extension of most single-process models. Extensions and new avenues of research are applied to Lévy processes first and then extended to stochastic volatility processes second.

The true benefit of using Lévy processes is the accessibility of the underlying distribution's characteristic function. The characteristic function $\Psi(\xi)$ of a given random variable X with probability density function $\theta(x)$ is the Fourier transform

$$\Psi(\xi) = \int_{-\infty}^{+\infty} \theta(x)e^{i\xi x}dx. \tag{2.5}$$

.

One advantage of working with characteristic functions is that for two independent random variables, the characteristic function of their sum is simply the product of their characteristic functions, reflecting that the probability distribution of $X+Y$ is the convolution of their distributions, for independent random variables X and Y. Thus, the probability density function $\theta_2(y)$ of the sum of two independent and identically distributed random variables with corresponding densities $\theta_1(x)$ and $\theta_1(y)$ is the convolution

$$\theta_2(y) = \int_{-\infty}^{+\infty} \theta_1(x)\,\theta_1(y-x)\,dx. \tag{2.6}$$

Furthermore, the characteristic function $\Psi_2(x)$ for the sum is given by

$$\Psi_2(\xi) = [\Psi_1(\xi)]^2. \tag{2.7}$$

This leads to a recursive argument for the sum of N i.i.d. random variables

$$\Psi_N(\xi) = [\Psi_1(\xi)]^N. \tag{2.8}$$

This property enables the straightforward derivation of the distribution of a sum of independent random variables by simply taking the product of their characteristic functions (or convolving their probability densities), thereby greatly simplifying both theoretical analysis and practical computation of aggregate behaviour.

2.1.3 Lévy-Khinchine

In general, the Lévy processes considered here have a simplistic characteristic function of the form $\Psi(\xi, t) = e^{\Psi(\xi)t}$ where the characteristic exponent is given by the Lévy-Khinchine formula

$$\Psi(\xi) = ia\xi - \frac{1}{2}\sigma^2\xi^2 + \int_{\mathcal{R}} \left(e^{i\xi\eta} - 1 - i\xi\eta\mathbf{1}_{[-1,1]}(\eta)\right) \nu(d\eta). \tag{2.9}$$

The Lévy-Khincine triplet (a, σ, ν) uniquely defines the Lévy process, where a is the linear drift of the process, σ is the volatility of the diffusion and $\nu(\eta)$ signifies the intensity of an embedded Poisson jump process with jump size η. Under the risk-neutral measure, the parameters contained in the triplet can be related using the equation

$$a = r - q - \frac{1}{2}\sigma^2 - \int_{\mathbb{R}} \left(e^{\eta} - 1 - i\eta\mathbf{1}_{[-1,1]}(\eta)\right) \nu(d\eta), \tag{2.10}$$

where r is the risk-free interest rate and, depending on asset class, q represents the dividend rate or cost of carry (where the sign of q is negative). Obvious examples of Lévy processes are the Wiener process itself or the Poisson process. A further category division of Lévy processes is finite and infinite activity processes. Finite activity processes include those whose jumps occur according to a Poisson distribution

$$X(t) = \sum_{n=1}^{N} n(t)J(n). \tag{2.11}$$

Jump processes considered in this thesis assume that the jump sizes are double-exponentially distributed. When a jump occurs, it is negative with probability p and exponentially distributed with parameter η_1. Likewise, it is positive with probability $q = 1 - p$ and has an exponential distribution with parameter η_2. Thus, the probability density of jump sizes is given by

$$f(x) = p\left(\eta_1 e^{\eta_1 x} \chi_{(-\infty,0]}(x)\right) + q\left(\eta_2 e^{-\eta_2 x} \chi_{(0,\infty)}(x)\right). \tag{2.12}$$

For considerations relating to modelling, finite activity processes are often combined with an independent Brownian motion. An example used within this thesis was introduced by Kou [2002], the Kou double exponential (KDE) process. The KDE process was an upgrade to the Merton [1976] model, which featured normally distributed jumps. The improvement of the KDE process follows the financial intuition that market adjustments are caused by isolated events, such as a media article.

Infinite activity processes are built upon the foundations presented by Geman et al. [2001], describing the idea of a stochastic clock as a representative model of time. This, in turn, can be viewed as the rate of economic activity building jump processes of infinite activity gives added weight behind the Lévy market model. This approach is consistent with no-arbitrage principles since every price process can be represented as a time-changed Brownian motion. The method for building this stochastic clock is by adding a subordinator. A Lévy process is a subordinator if it is non-negative (almost surely) and non-decreasing. Subsequently, a Lévy process subordinated by an independent Lévy process is itself a Lévy process. For a more in-depth look into infinite activity processes, see Madan and Yor [2008], where the CGMY process is introduced. Processes such as CGMY or Meximer do not have simple subordinator processes. Processes such as Variance Gamma (VG) and the Normal Inverse Gaussian (NIG), however, have comparatively simple formulations. The VG process, Madan et al. [1998], can be constructed by adding a gamma-distributed variable $G_t \sim \Gamma_t(1/\nu, 1/\nu)$ to a Brownian motion W_t with a drift term

$$X_t = \theta G_t + \sigma W_{G_t}. \tag{2.13}$$

Similarly the NIG process, Barndorff-Nielsen [1998], can be constructed by adding an inverse-Gaussian distributed variable $I_t(t, \delta\sqrt{\alpha^2 - \beta^2})$ to a Brownian motion W_t with a drift term

$$X_t = \beta\delta^2 I_t + \delta W_{I_t}. \tag{2.14}$$

2.2 Option Pricing

Underpinning this entire thesis is the application of pricing models for financial options or contingent claims. This area has been a focus of intensive research within the economic literature for the last few decades. Fundamental and transformative works in the early 1970s, notably Black and Scholes [1973] and Merton [1973], introduced the concepts of pricing options under a risk-neutral measure, delta hedging, and portfolio replication. All of which remain fundamental topics today. The works by Black and Scholes [1973] presented a closed-form solution for the price of European options, which gained fame beyond financial circles. Similarly, Merton presented closed-form solutions for perpetual American and continuously monitored barrier options. The limitations of the papers in those early days were the use of log-normal price processes for relevant underlying assets and continuous monitoring of barriers. These limitations have since been removed with the increased sophistication of modelling techniques, resulting in more accurate approximations for pricing and hedging that capture market dynamics.

2.2.1 Vanilla options

Despite the triteness of including the Black-Scholes model for pricing vanilla options in academic works, no discussion on pricing options is complete without it. The model Black and Scholes [1973], Merton [1973] is a cornerstone of modern quantitative finance, providing a closed-form solution for pricing European-style vanilla calls. Underpinning this model is the assumption that the underlying asset follows a geometric Brownian motion with constant volatility and drift under the risk-neutral measure. The seminal insight by Black, Scholes, and Merton was that, through a continuously updated hedging strategy, one can replicate the payoff of a call option and thus price it by eliminating arbitrage opportunities. Their derivation leads to a partial differential equation, often referred to as the Black–Scholes PDE, whose solution yields the well-known closed-form formula.

In practice, the model expresses the fair value of a call option as

$$C(S,t) = S\,\Phi\big(d_1\big) - Ke^{-r(T-t)}\,\Phi\big(d_2\big), \tag{2.15}$$

where S is the current asset price, K is the strike price, r is the risk-free interest rate, T is the maturity date, t is the current time, and Φ is the cumulative distribution function of the standard normal distribution. The terms d_1 and d_2 are defined as

$$\begin{aligned} d_1 &= \frac{\ln\Big(\frac{S}{K}\Big) + \Big(r + \frac{\sigma^2}{2}\Big)(T-t)}{\sigma\sqrt{T-t}}, \\ d_2 &= d_1 - \sigma\sqrt{T-t}, \end{aligned}$$

where σ is the volatility. These expressions shed light on how sensitivity to volatility, interest rates, and time to maturity is embedded in the pricing formula Hull [2017].

Alternative methods for numerical pricing methods involve binomial or trinomial trees Cox et al. [1979], where a discrete model of the underlying price over several time steps converges to the Black–Scholes result. Simulation based techniques, such as Monte Carlo simulations Boyle [1977] are also widely employed. Their advantage lies in their ability to quickly adapt to various stochastic processes or option structures. However, these numerical approaches can be computationally demanding. For vanilla European options, the Black-Scholes model remains the industry standard.

2.2.2 Barrier options

A barrier option is a financial contract whose payoff is a function of the value of the underlying asset at expiry, but with an added condition that determines whether the contract is paid out or not, depending on whether the asset touches a predetermined barrier at some point in the option's life cycle. While there are other barrier forms, such as a target redemption barrier, the vanilla barrier of "knock-in" or "knock-out" is the most common form.

Barrier options are particularly popular in FX markets and high-frequency trading strategies. These contracts come in various formats with potentially mixed barriers layered into the option; some may be active for a specific time window or the entire option's lifetime. The most widely quoted barriers on the open market are single knock-in and knock-out barrier options; more exotic flavours tend to be handled OTC. A knock-in barrier requires the barrier to be touched to be active at expiry. A knock-out barrier starts as an active contract but is deactivated if the barrier is touched. An additional feature in determining the type of barrier option is the starting position of the barrier relative to the underlying asset value at the beginning of the option, S_0. For instance, if the barrier is located at a value higher than S_0 and is of the knock-out style, we refer to this as an up-and-out barrier option. For the most part, we consider the case of the down-and-out barrier option for simplicity; however, the techniques discussed are equally applicable to other types of barrier options.

Closed-form solutions for single barrier options were first presented by Merton [1973] and Goldman et al. [1979] in the case of a single barrier option. In both cases, the underlying process was driven by a GBM. Kunitomo and Ikeda [1992] introduced the natural extension to double barrier options using acceleration techniques to converge an infinite series. These early solutions are applicable only to continuously monitored barriers and are limited to a GBM process, which does not accurately reflect the realities of financial contracts. From a practical point of view, the continuous monitoring of a barrier is impossible. As such, barrier options have strict regulations built into the agreement for the monitoring process, from the number of dates to precisely what is monitored. The interested reader should consult Kou [2007] for more on this topic. The more interesting and practical problem is pricing discretely monitored barrier options with an extension to the general class of Lévy processes. To reflect this increased complexity, we must use numerical techniques to price such options.

A natural avenue of enquiry is adapting closed-form solutions from continuous monitoring to approximate the discrete monitoring case. This would have the advantage of utilising an existing literature base and result in monitoring dates being independent of the computational cost. While attractive in principle, Broadie et al. [1997] showed that the convergence rate of approximated discretely monitored solutions to the continuous case relative to the number of dates was $O(1/\sqrt{N})$, which is too slow. We also lack the extension to the general case of Lévy processes.

A helpful checklist of traits we wish our numerical methods to have includes five primary things. First, the method must be fast. Within a financial setting, speed is paramount; reasons for this range from regulatory requirements to the frequency of a trading strategy. Speed is also crucial, as a portfolio of options often needs to be priced or repriced for portfolio rebalancing and risk management. An extension to this first requirement is that the numerical pricing method should have exponential convergence; as the grid size or number of simulations increases, the error converges to machine zero, i.e. E$-$16 exponentially. A second desirable trait is that the method can be adapted simply to single and double barriers. An extension of this trait is the ability to extend to other forms of path-dependent options easily, but this is not a requirement. Third, we would prefer a method whose computational complexity is not dependent on the number of barrier monitoring dates. Fourth, a simple method is preferable to a complex one. While not always possible, given the prior three requirements, any simplicity that can be achieved is desirable, as complexity can build with ease. Lastly, we want a process-independent method. This is not possible. Thus, we settle to be able to choose from within a family of processes such as Lévy processes. This allows the method to fit a process to an asset rather than the other way around.

The bulk of numerical schemes for discretely monitored double barrier options seen in the literature are variations of quadrature methods, where the price process can be expressed as a recursive integral of the form

$$v(x,n) = \int_d^u k(x - x' \mid \Delta t) v(x', n-1) dx', \tag{2.16}$$

where d and u are the values of the down and upper barrier, $v(x,n)$ is the value of the option at the n^{th} monitoring date, $k(x - x' \mid \Delta t)$ is the transition density of the underlying process. The quadrature approach can be implemented in either a path-integration approach Chiarella et al. [2008] or by utilising interpolations Andricopoulos et al. [2003] and Fusai and Recchioni [2007]. The technique has the advantage of being usable for the general Lévy processes, and that discretising time does not add an error factor. The major drawback is the need for at least one integration step for each monitoring date; thus, the computational cost scales linearly with the number of monitoring dates. Combining this with the polynomial convergence inherent to the scheme renders the process suboptimal.

An obvious numerical candidate would be to utilise Monte Carlo techniques, which are industry standard when handling exotic options. The intrinsic problem with this approach is the speed of convergence and the overall time required to retrieve accurate results, which is exacerbated in this case by the linear scaling with the number of monitoring dates. While many multiple variance reduction techniques have been presented that allow for faster convergence, see Beaglehole et al. [1997] or more recently Dingeç and Hörmann [2012], this problem remains.

Another traditional approach employed in option pricing is finite difference methods. This has been explored by Boyle and Titan [1998] and more recently by Golbabai et al. [2014], who presented improved models with 4th-order accuracy but only in the case of a GBM process. The extension to the general class of Lévy models has not been done. Unfortunately, finite difference schemes have inherent drawbacks that make them less appealing when applied to pricing barrier options. In addition to the intrinsic stability associated with the more straightforward explicit methods, barrier discontinuities can introduce errors into implicit schemes, such as the Crank-Nicolson method, resulting in slow error decay or error divergence. Consequently, a much finer grid than otherwise preferable must be utilised, increasing the computational cost and eliminating the ideal of independence from the number of monitoring dates.

The last of three popular methods is tree approximations. The binomial tree can be applied to the barrier case, see Steiner and Wallmeir [1999]. As with other path-dependent options, the binomial tree approach suffers from requiring a high number of time steps due to the slow convergence. Trinomial trees achieve a superior convergence rate, but at the cost of increased computational complexity. An adapted version was presented by Figlewski and Gao [1999], which utilises a variable time mesh to add more time points near a barrier and fewer periods with no monitoring dates, thereby achieving superior convergence and accuracy.

All of these methods have advantages and disadvantages. Most importantly, when applied to discrete barrier options, they fail to meet all five desirable points discussed earlier for a reasonable pricing model.

The class of methods that this thesis focuses on is the use of integral transforms. A brief history of the technique is useful at this juncture to set the current landscape. The initial investigation into option pricing using integral transforms is credited to the famous work by Heston [1993], who was the first to use the characteristic functions of the probability of an option being in the money. This work builds upon the work by Chen and Scott (1992), who attempted the same outcome but used the Laplace transform instead. A major step forward in presenting a comprehensive pricing method was the work by Carr and Madan [1999], who published a Fourier transform approach to pricing European options, utilising both the characteristic function and the payoff in the Fourier domain. In the case of European options, this approach has been improved in recent years. Notably, Fang and Oosterlee [2008] devised a pricing technique using the Fourier-cosine expansion for the GBM process. This was extended to the general class of Lévy models by Lord et al. [2008] and more exotic options, including discretely monitored barrier options, by Fang and Oosterlee [2011]. The Fourier-cosine approach achieves satisfactory accuracy with exponential error convergence relative to the number of grid points in the log-price domain, provided that the transition density of the underlying asset is smooth for all orders of differentiation, i.e.

$$k(x - x' \mid \Delta t) \in C^{\infty}. \tag{2.17}$$

As discussed in Phelan et al. [2020], a drawback of the method is the linear relationship between computational cost and the number of monitoring dates. Additionally, the error converges with a polynomial degree rather than an exponential one for specific processes, such as the variance gamma process. This problem can be overcome with clever spectral filters, as demonstrated by Ruijter et al. [2015].

Simultaneously with the later works employing the Fourier transform, the Hilbert transform was explored and applied to price barrier options using a backwards induction scheme within Fourier space, see Feng and Linetsky [2008, 2009]. This method demonstrated exponential error convergence with the grid for log price and achieved machine accuracy of E-16. Similar to the Fourier-cosine method, a significant disadvantage of applying it to discrete monitoring cases is that performance increases linearly with the number of dates. This method was also improved for the variance gamma model using spectral filters by Phelan et al. [2019] to enhance the convergence from polynomial to exponential for transition densities of the form given by Eq. (2.17).

A common theme thus far is the linear relationship between the computational time and the number of monitoring dates applied to the discrete case. To circumvent this problem, Fusai et al. [2006] utilised the z-transform to collapse the time domain, transforming the option price into the Wiener-Hopf equation, which can be solved analytically for a log-normal price process. This results in both the need for a forward and inverse z-transform. The forward transform is reasonably straightforward, but the inverse z-transform is more complex and requires numerical approximation. This topic is explored in detail in Chapter 3. The first numerical approximation used for the inverse z-transform in this context was the version presented by Abate and Whitt [1992a,b]. The simplest version has a computational time dependent on the grid size, representing the number of monitoring dates. An acceleration technique can make the inverse transform date independent of monitoring dates. The natural candidate and the first used in the literature is the Euler acceleration, as seen in O'Cinneide [1997]. This technique was subsequently adapted by Green et al. [2007, 2010] to give the Fourier-z transform of the probability distribution at the extrema and points around the barriers. By now, utilising the Plancherel theorem given by Eq. (2.30), these transformed distributions can be used to price options. This process was codified and presented by Fusai et al. [2016], employing the Hilbert transform from Stenger [2011] in a manner similar to the process described by Feng and Linetsky [2008]. The resulting implementation results in a pricing process that is extendible to the general class of Lévy processes, is date-independent with respect to computational cost, and achieves exponential error convergence for single barriers where the transition densities satisfy Eq. (2.17). Unfortunately, this was not the case for double barrier options that could only achieve polynomial convergence. This slow convergence was eventually overcome using spectral filters by Phelan et al. [2019] to improve the Fourier-cosine method embedded in the process, thus achieving exponential convergence even in the double barrier case. The only downside to this method is an apparent error floor of E-10 resulting from the numerical inverse z-transform applied within the pricing process. Further discussion on this is given in Chapter 4.

Lian et al. [2017] proposed a slightly different approach, which circumvents the inverse z-transform by calculating the price at the N^{th} date through computing the $(N-1)^{\text{th}}$ power of a matrix. This methodology is usable for the general exponential Lévy process but requires the calculation for the N^{th} power of a matrix. While acceleration techniques are available for matrix power calculations, the authors do not employ such techniques. Consequently, the computational cost for calculating a matrix's N^{th} power depends on N and cannot be claimed as date independent.

An interesting avenue of investigation into Legendre polynomials, see Sobhani and Milev [2018]. While the method achieves date independence regarding computational cost, it has yet to be extended beyond GBM to the broader class of Lévy processes. CAS wavelets have recently been used to price discrete barrier options, Sobhani [2021]. While an interesting area of research, the results struggle with accuracy and speed and are only applied to the GBM process.

2.2.3 α-quantile options

Options with some referable hindsight are an increasingly popular class of OTC options traded. These options have a fixed point of expiry where the payoff depends on some historical feature of the underlying asset's path. The typical example is lookback options, whose payoff is determined by the underlying asset's maximum or minimum price over the option's lifetime, Fusai et al. [2016]. Typically, fixed strike lookback options result in a payoff of the form

$$V_C(S,t) = \max(S_M - K, 0), \tag{2.18}$$

$$V_P(S,t) = \max(K - S_m, 0), \tag{2.19}$$

for a call and a put, respectively, where $S_M = \max_{t\in[0,T]} S(t)$ and $S_m = \min_{t\in[0,T]} S(t)$ are the maximum and minimum price of the underlying, respectively. This can be discretised similarly to barrier options where the maximum and minimum are viewed over a selection of predetermined dates, i.e. $S_M = \max_{t_n\in[t_0,t_N]} S(t_n)$ and $S_m = \min_{t_n\in[t_0,t_N]} S(t_n)$.

It is often convenient to begin with lookback options since quantile options are often regarded as an adaptation of the former. The payoff of quantile options depends on a single maximum or minimum and the percentage of time above a certain threshold. This percentage is often called the α percentile, hence the name of α-quantile options. These options were originally introduced by Miura [1992] to mitigate sensitivity to extreme price spikes, particularly in volatile markets. Early analytical solutions were published in the mid-1990s by Akahori [1995] and Yor [1995], exclusively using the GBM process. Later that year, Dassios [1995] would adapt the works by Port [1963] and Wendel [1960] that created an identity that states the α-quantile of a Brownian motion over T has the same distribution as the sum of the infimum αT. The Dassios-Port-Wendel identity says that if $X_M = \max_{t\in[0,\alpha T]} X_1(t)$ and $X_m = \min_{t\in[0,(1-\alpha T)]} X_2(t)$ where $X_1(t)$ and $X_2(t)$ are independent Brownian motions then

$$X_\alpha \overset{\mathrm{d}}{=} X_M + X_m, \tag{2.20}$$

where X_α is the α-quantile of the Brownian motion. Dassios also showed in the paper that this could be extended to a general class of Lévy processes, although the result may not be unique. Using this work as a base, several pricing methods using general Lévy processes have been presented, such as the works by Cai et al. [2010] using the Kou double exponential process. A variant using Monte Carlo methods was presented by Ballotta [2002] utilising jump-diffusion methods.

The first closed-form solutions for discretely monitored quantile options were presented by Atkinson and Fusai [2007], utilising the z-transform to engineer the problem using a Wiener-Hopf equation. As with many previously discussed closed-form solutions, the price was solved for the GBM process. Later, this was extended to the general class of Lévy processes by Green [2009]. This problem was further extended by Phelan et al. [2020] using the Spitzer identities developed by Green [2009] to develop a pricing method which is extendible to general Lévy processes and is exponentially convergent. This method, too, suffers from an unsatisfactory error floor caused by the inverse z-transform.

2.2.4 Fourier transform

Before detailing the integral transform approach to pricing, we must introduce the integral transforms for completeness.

The Fourier transform has numerous applications in mathematics and physics, spanning multiple distinct yet interconnected knowledge domains. The most common setting in which the Fourier transform is applied is transforming a one-dimensional function of time into a function of frequency. The field of signal processing utilises this application of the Fourier transform extensively. Much of the foundation literature surrounding financial applications originated within operations research and signal processing. Various modelling considerations can be chosen, which largely depend upon the setting. For instance, it is typical for financial literature surrounding integral transforms to define the Fourier domain as its spectrum, predominantly due to historical reasons. This choice ensures general consistency with other areas of operations research; however, it can be confused with the more abstract understanding of transforming the price distribution or the asset's log-price. There is little physical sentiment, as one would find with other applications of the Fourier transform.

Therefore, although multiple definitions of the Fourier transform exist, we adopt the definition preferred in scientific computing and finance in general. Thus, the

Fourier transform and inverse Fourier transform are defined as follows

$$\widehat{f}(\xi) = \mathcal{F}_{t\to\xi}[f(t)] = \int_{-\infty}^{+\infty} f(t)e^{i\xi t}dt, \tag{2.21}$$

$$f(t) = \mathcal{F}^{-1}_{\xi\to t}[\widehat{f}(\xi)] = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \widehat{f}(\xi)e^{-i\xi t}d\xi. \tag{2.22}$$

This choice of definition is exceptionally comfortable to work with since the forward transform Eq. (2.21) is the equivalent of the characteristic function of a random variable with probability distribution $f(x)$.

While this definition is most natural, in a financial setting, there is typically a time origin where we are unconcerned with the behaviour of the option before $t = 0$. Thus, we also include the unilateral definition

$$\widehat{f}(\xi) = \mathcal{F}_{t\to\xi}[f(t)] = \int_{0}^{+\infty} f(t)e^{i\xi t}dt, \tag{2.23}$$

$$f(t) = \mathcal{F}^{-1}_{\xi\to t}[\widehat{f}(\xi)] = \frac{1}{2\pi}\int_{0}^{+\infty} \widehat{f}(\xi)e^{-i\xi t}d\xi. \tag{2.24}$$

There is a vast choice of literature concerning the Fourier transform and its long history. For a modern reference, see Kreyszig [2011], who gives a detailed technical description from first principles. For a thorough history of the development of the Fourier transform and its applications, see Dominguez [2016]. Despite the Fourier transform's rich detail, we restrict the information presented here to the specific application and approach to option pricing. Below is a brief introduction to some of the techniques relevant to pricing.

To begin this more specific discussion, the Fourier convolution of two functions $f(t)$ and $g(t)$ is given by

$$h(t) = (f * g)(t) = \int_{0}^{\infty} f(z)g(t-z)dz = \int_{-\infty}^{\infty} f(t-z)g(z)dz. \tag{2.25}$$

This can often be reduced to a simpler problem where the application of a forward transform to both $f(t)$ and $g(t)$ gives

$$\widehat{h}(\xi) = \widehat{f}(\xi)\widehat{g}(\xi) = \int_0^\infty f(x)e^{i\xi t}dt \int_0^\infty g(t)e^{i\xi t}dt, \tag{2.26}$$

in the Fourier domain. To obtain $h(t)$, an inverse Fourier transform is performed on $\widehat{h}(\xi)$. Convolutions can reduce many problems to a forward multiplication resulting from Eq. (2.25).

The Fourier shift theorem, which can be applied to either the state or the Fourier domain, describes a shift applied to the function by a fixed amount a. From the definitions given above, this shift can be formulated as

$$\mathcal{F}_{t\to\xi}[f(t)] = \widehat{f}(\xi) \quad \to \quad \mathcal{F}_{t\to\xi}[e^{-iat}f(t)] = \widehat{f}(\xi - a), \tag{2.27}$$

$$\mathcal{F}^{-1}_{\xi\to t}[\widehat{f}(t)] = f(\xi) \quad \to \quad \mathcal{F}^{-1}_{\xi\to t}[e^{iat}\widehat{f}(\xi)] = f(t - a). \tag{2.28}$$

An important tool that follows is Parseval's theorem, which is an isometry that can evaluate an inner product. For example, the expected value of a function of a random variable. Parseval's theorem preserves the inner product when computing the expected value and is given as

$$\int_{-\infty}^{\infty} f^*(t)g(t)dt = \frac{1}{2\pi}\widehat{f}^*(\xi)\widehat{g}(\xi)d\xi, \tag{2.29}$$

where f^* is the complex convolution of f.

The last theorem required to apply the Fourier transform to the field of option pricing is Plancherel's. This theorem states that if we have two functions $f(t)$ and $g(t)$ with Fourier transforms $\widehat{f}(\xi)$ and $\widehat{g}(\xi)$ respectively, then

$$\int_{-infty}^{\infty} f(t)g^*(t)dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} \widehat{f}(\xi)\widehat{g}^*(\xi)d\xi, \tag{2.30}$$

where g^* represents the complex conjugate of $g(t)$. While this was originally presented for real values of ξ only, it was extended by Paley and Wiener [1933] to the entirety of the complex plane. Consequently, this extension demonstrated that a function's upper and lower half-range Fourier transforms are holomorphic in the upper and lower planes of ξ, respectively. Thus

$$\widehat{f_+}(\xi) = \mathcal{F}_{t\to\xi}[f(t)\mathbf{1}_{\mathbb{R}^+}(t)](\xi) = \int_0^\infty f(t)e^{i\xi t}dt, \tag{2.31}$$

$$\widehat{f_-}(\xi) = \mathcal{F}_{t\to\xi}[f(t)\mathbf{1}_{\mathbb{R}^-}(t)](\xi) = \int_{-\infty}^0 f(t)e^{i\xi t}dt, \tag{2.32}$$

where $\mathbf{1}_A(t)$ is the indicator function on a given set A. Furthermore, the function $\widehat{f_+}(\xi)$ is analytical in the upper half plane of ξ, which includes the real line. Symmetrically, the function $\widehat{f_-}(\xi)$ is analytical in the lower half plane of ξ, which includes the real line. Therefore, the analytic regions of $\widehat{f_+}(\xi)$ and $\widehat{f_-}(\xi)$ overlap in a strip which includes the real line.

A key concept of the Fourier transform approach to option pricing is the use of the characteristic function. The Fourier transform of the probability density function (PDF) $p(x,t)$, referred to as the characteristic function, is defined as

$$\Psi(\xi,t) = \mathrm{E}[e^{i\xi X(t)}] = \int_0^\infty e^{i\xi t}p(x,t)dx = \mathcal{F}_{x\to\xi}[p(x,t)] = \widehat{p}(\xi,t). \tag{2.33}$$

A unique advantage in obtaining the characteristic function of a process is that $\Psi(x,\xi)$ uniquely describe the process in full. Moments of the process can also be obtained, resulting in a general preference for characteristic functions over moment-generating functions (MGFs). This is because MGFs do not exist for all random variables, for example, the log-normal process. An additional advantage is that the Fourier transform's dimensionality reduction attributes are exploited, resulting in PDFs with complicated forms simplifying to simple expressions in Fourier space. This is generally true for most Lévy processes where the characteristic function can be expressed as

$$\Psi(x,\xi) = \mathrm{E}[e^{i\xi X(t)}] = \exp\left(t\left(ai\xi - \frac{1}{2}\sigma^2\xi^2 + \int_{-\infty}^\infty (e^{i\xi x} - 1 - i\xi x\mathbb{I}_{|x|<1})\Pi(dx)\right)\right), \tag{2.34}$$

where $a \in \mathbb{R}$, $\sigma \geq 0$ and Π is the so-called Lévy measure.

Critical to calculating an option's value is the use of the expectation for a function of a random variable, making the properties above particularly useful. Given a function $\psi(X)$ of a random variable X with a corresponding PDF $p_X(t)$, the expectation can be expressed in the Fourier domain using the Plancherel relation Eq. (2.30)

$$\mathrm{E}[\psi(X)] = \int_0^\infty \psi(x)p_X(x)dx = \frac{1}{2\pi}\int_{-\infty}^{\infty} \widehat{\psi}^*(\xi)\Psi(\xi)d\xi. \tag{2.35}$$

2.2.5 z-transform

The z-transform is intrinsic to the first few sections of this thesis, and a more in-depth view of the transform and its inverse is given in Chapter 3. In keeping with the brief technical introductions of this chapter, a brief explanation is provided to help readers understand the pricing methodology discussed in this thesis and to allow them to skip directly to any chapter without needing to read it linearly. The z-transform is a discrete-time transform where the unilateral version is given by

$$\widetilde{f}(z) = \mathscr{Z}_{n\to z}[f(n)] = \sum_{n=0}^{\infty} f(n)z^{-n}, \tag{2.36}$$

where z is a complex number and $f(n)$ is a discretely sampled function over $n \in \{0, 1, 2, \ldots\}$. As with the Fourier transform, a bilateral z-transform exists where the range of the summation extends to $\pm\infty$, but as we typically reformulate to consider the options time to expiry $\tau = T - t$ where we are unconcerned with the behaviour at $\tau < 0$. The definition given in Eq. (2.36) is most commonly seen within discrete signal processing, where z-signals occupy an entire literature section. A potentially more convenient and interchangeable definition of the transform exists where z, where we set $q = z^{-1}$, so that $z = 1/q$

$$\widetilde{f}(q) = \mathscr{Z}_{n\to q}[f(n)] = \sum_{n=0}^{\infty} f(n)q^{n}, \tag{2.37}$$

This definition is more suitable for a probability setting, such as finance, because its form is the same as that of probability-generating functions. A common choice within the literature is to present the discrete function under consideration as a function of n. This choice is interchangeable.

Typically, within signal processing literature, the inverse of the z-transform is often handled by examining well-known z-transform pairs. Extensive tables of pairs with common functions and their analytic z-transform provide a convenient way of finding the forward or inverse transform by comparing functions to their closest pair. Many of these functions are simple trigonometric or exponential functions with various scalars. For more general functions, where a similar pair may not exist, the inverse z-transform is given by a Cauchy integral of the form

$$f(n) = \mathcal{Z}^{-1}_{z\to n}[\widetilde{f}(z)] = \frac{1}{2\pi i}\oint_C \widetilde{f}\,(z)\,z^{n-1}dz, \tag{2.38}$$

where C is a counter-clockwise closed contour within the region of convergence for the function $f(n)$. Equivalently, we can reformulate in terms of q

$$f(n) = \mathcal{Z}^{-1}_{q\to n}[\widetilde{f}(q)] = \frac{1}{2\pi i}\oint_C \widetilde{f}(q)q^{n}dq. \tag{2.39}$$

Unfortunately, this integral is often intractable for relatively simple and smooth functions. If a direct approach is possible, it may still be a poor approach to tackle the integral directly, considering complexity and speed. In most cases, employing approximation methods to solve the inversion problem is more natural. The option pricing methodology employs a Fourier-z transform and thus requires an inversion in both the Fourier and z domains. A pleasant feature is that the time and space domains are orthogonal for the Fourier-z transform; as such, the individual inversion procedures can be performed and implemented independently and in any order.

Despite much of the foundation literature regarding discrete-time transforms being housed within signal processing, a renewed interest has been taken in the subject by disciplines less associated with it. Many fields, such as probability, finance and chemical engineering, have modelling considerations where continuous transforms are incoherent or not viable beyond an academic or theoretical setting. A direct consequence of this is apparent when examining the difference in the number of papers concerned with continuous transforms compared with those interested in their discrete-time counterparts. For instance, since signals are continuous, the z-transform's continuous counterpart, the Laplace transform, is far more prevalent in the literature.

In an option pricing setting, it is more accurate to assume discrete monitoring as continuous monitoring is an academic exercise rather than a reality in markets, with the notable exception of American markets, where this is a practical consideration. This was investigated extensively by Kou [2007]. Thus, there is a desire to discretise the monitoring period to model and price a path-dependent option accurately. This requires a discrete transform instead of a continuous one. While forward z-transforms are often well-defined, the inverse can be non-trivial, as shown when the ability to compare to analytical pairs is removed. In such cases, a numerical approximation is required due to the intractability of the Cauchy integral governing the inverse z-transform seen in Eq. 2.38. A complete discussion on the inverse z-transform is given in Chapter 3.

2.2.6 Hilbert transform

Unique among the integral transforms presented in this thesis, the Hilbert transform does not alter the domain of the function it operates on. The Hilbert transform is a crucial component of the pricing methodology, making it worthwhile to provide a brief introduction. The Hilbert transform of a function of the Fourier domain is given by

$$\begin{aligned}\mathcal{H}[\widehat{f}(\xi)] &= \text{P.V.}\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\widehat{f}(\xi')}{\xi-\xi'}d\xi' \\ &= \lim_{\epsilon\to 0^+}\frac{1}{\pi}\left(\int_{-\infty}^{\xi-\epsilon}\frac{\widehat{f}(\xi')}{\xi-\xi'}d\xi' + \int_{\xi+\epsilon}^{\infty}\frac{\widehat{f}(\xi')}{\xi-\xi'}d\xi'\right),\end{aligned} \tag{2.40}$$

where P.V. denotes the Cauchy principal value. There is no need for an explicit inverse transform, as an inverse can be recovered using the Hilbert transform property

$$\mathcal{H}[\mathcal{H}[\widehat{f}(\xi)]] = -\widehat{f}(\xi). \tag{2.41}$$

The topic of the Hilbert transform is significant, with many diverse applications. As it is used in pricing an exotic option without modification, this thesis does not provide a detailed examination of it. The interested reader is directed to King [2009], where the author has written extensively about the topic.

2.2.7 Fourier methods for pricing

We now turn to a discussion on the methodology behind the Fourier pricing approach. The cornerstone of the method is calculating the PDF in the Fourier-z domain as a separate step, rather than embedding it in the process as a whole. The Plancherel relation is then applied along with a Fourier transform of the payoff function to calculate the price. Given the extensive discussion of this method, a technical introduction is provided in this section to establish the context. It is assumed that the payoff and the PDF are functions available in the Fourier domain. The underlying process follows an exponential random process, which is the default unless stated otherwise.

Let $S(t)$ be the price of the underlying asset and $x(t) = \log(S(t)/S_0)$ be the log-price. To calculate the option value at $t = 0$, we assume that the initial price of the underlying is $S(0) = S_0$ and consequently $x(0) = 0$. Furthermore, to find the price $v_0(x)$ at time $t = 0$ we need to discount the expected value of the damped payoff function $\phi(x(T))e^{-\alpha_d x(T)}$ at maturity $t = T$ with respect to an appropriate risk-neutral probability distribution function $p(x, T)$ with initial conditions $p(x, 0) = \delta(x)$. The direct approach assumes that the PDF of the underlying process at time T can be represented as a transition density given its value at some time $t < T$

$$p(x(T) \mid x(t)) = k(x(T) - x(t)), \tag{2.42}$$

where k is the transition density. The process distribution depends only on the difference between current and future values and not on absolute values. This allows the calculation of the price using the convolution theorem

$$\begin{aligned} v_0(x) &= e^{-rT}\mathrm{E}\left[\phi(x(T))e^{-\alpha_d x(T)} \mid x(0) = x\right] \\ &= e^{-rT}\int_{-\infty}^{\infty} \phi(x')e^{-\alpha_d x'} p(x' \mid x)dx' \\ &= e^{-rT}\int_{-\infty}^{\infty} \phi(x')e^{-\alpha_d x'} k(x - x')dx' \qquad (2.43) \\ &= e^{-rT}\mathcal{F}^{-1}_{\xi\to x}\left[\widehat{\phi}(\xi)\Psi_k^*(\xi - i\alpha_d, T)\right], \qquad (2.44) \end{aligned}$$

where $*$ denotes complex conjugation. We utilise that Eq. (2.43) is a convolution and thus a multiplication inside the Fourier domain. In this context, $\Psi(\xi, T)$ is the characteristic function of the transition density. As a result $\Psi_k^*(\xi - i\alpha_d, T)$ is the complex conjugate of the Fourier transform of $e^{-\alpha_d x}k(x, T)$. The next stage in the pricing process involves taking the Fourier transform of the damped payoff to perform computations using the underlying process's characteristic function. For illustration purposes, we use the damped payoff of a double barrier option, although other payoff options could be interchanged. The damped payoff for a double barrier option is

$$\phi(x) = e^{-\alpha_d x} S_0\big(e^x - \theta e^{k_S}\big)^+ \mathbf{1}_{[l,u]}(x), \qquad (2.45)$$

where $e^{\alpha_d x}$ is the damping factor, $\theta = 1$ for a call, $\theta = -1$ for a put and $\mathbf{1}_{[l,u]}(x)$ is the indicator function of the set $[l, u]$, where $l = \log(L/S_0)$ is the lower log-barrier and $u = \log(U/S_0)$ is the upper log-barrier. The values of U and L are the upper and lower barriers, respectively. For the log-strike, we use $k_S = \log(K/S_0)$, where the subscript distinguishes it from the transition density $k(\cdot, \Delta T)$, where K is the strike price. The Fourier transform of the damped payoff is available analytically

$$\widehat{\phi}(\xi) = S_0\left(\frac{e^{(1+i\xi-\alpha_d)a} - e^{(1+i\xi-\alpha_d)b}}{1 + i\xi - \alpha_d} - \frac{e^{k_S+(i\xi-\alpha_d)a} - e^{k_S+(i\xi-\alpha_d)b}}{i\xi - \alpha_d}\right), \qquad (2.46)$$

where for a call option $a = u$ and $b = \max(k_S, l)$ and for a put $a = l$ and $b = \min(k_S, u)$.

A breakthrough in the popularity of Fourier methods was the demonstration by Carr and Madan [1999] that the Black-Scholes formula and put-call parity can be recovered with transforms in Fourier space for European vanilla options. While not strictly necessary for this thesis, it is an important demonstration of understanding the literature. The initial step is to take the payoff functions of vanilla options with European exercise, in terms of log-prices, to be $\psi^c(x) = S_0(e^x - e^k)^+$ for a call and $\psi^p(x) = S_0(e^k - e^x)^+$ for a put. In this context, $k = \ln(K/S_0)$ is the strike price expressed in log form. Following risk-neutral valuation, the value of a call option is its expected payoff under $\mathcal{Q}$ is

$$v^c(x, K) = \int_k^\infty \psi^c(x) p(x, T) dx. \tag{2.47}$$

Employing Parseval's identity, the value of the call option expressed in Fourier space is

$$v^c(x, K) = \frac{e^{-rT}}{2\pi} \int_0^{\infty + iw} \widetilde{\psi}^c(\xi) \Psi(\xi, T) d\xi, \tag{2.48}$$

where $\widetilde{\psi}^c(\xi)$ is the Fourier transform of the payoff. The Fourier-transformed payoff can be calculated as

$$\begin{aligned}
\widetilde{\psi}^c(\xi) &= \int_k^\infty S_0 \left(e^x - e^k\right) e^{i\xi x}\, dx \\
&= \int_k^\infty S_0 e^{i(\xi - i)x} dx - S_0 e^k \int_k^\infty e^{i\xi x} dx \\
&= S_0 \left[\frac{e^{i(\xi-i)x}}{i(\xi - i)} - \frac{e^{i\xi x + k}}{i\xi} \right]_k^\infty \\
&= -S_0 \frac{e^{(1+i\xi)k}}{\xi(\xi - i)}.
\end{aligned} \tag{2.49}$$

Substituting Eq. (2.49) into Eq. (2.48) we find

$$v^c(x, K) = \frac{-Ke^{-rT}}{2\pi} \int_0^{\infty + iw} \frac{e^{-i\xi K} \widetilde{\psi}^c(\xi)}{\xi(i + \xi)} d\xi. \tag{2.50}$$

The integration path $w > 0$ is required for a call option and $w < 0$ for a put option. The formulation of w highly depends on the underlying model, where $w > 1$ suffices for the Gaussian distribution. The extra parametrisation is required to extend to more complicated distributions such as normal inverse Gaussian or variance gamma. The residues of the integrand at the poles $\xi = 0$ and $\xi = -i$ are $-Ke^{rT}/2\pi i$

and $S_0/2\pi i$, respectively. To recover the put-call parity, we ensure that $w > 0$ to encapsulate these residues, which produces

$$v^c(x, K) = v^p(x, K) + S_0 - Ke^{-r\tau}, \tag{2.51}$$

where $\tau = T - t$ and $v^p(x, K)$ is the value of vanilla put with the same expiry and strike price as the call. Partial fractions can then be employed to reformulate the call price into a format compatible with the Black-Scholes world, assuming the usage of the Gaussian distribution

$$v^c(x, K) = \frac{Ke^{-rT}}{2\pi i} \int_0^{\infty + iw} e^{i\xi k} \Psi(\xi, T) \left(\frac{1}{i + \xi} - \frac{1}{\xi} \right) d\xi, \tag{2.52}$$

where $k = \ln(K/S_0)$. This is equivalent to the original Black-Scholes formulation for a vanilla call.

2.3 Numerical Fourier pricing procedures

In Chapter 4, we improve upon the pricing procedure first presented by Fusai et al. [2016] and use Feng and Linetsky [2008] as a benchmark to judge error performance and computational speed. While the interested reader is directed to these papers and those by Phelan et al. [2019] to gain a deeper understanding of the method, the procedure is outlined here to make this thesis self-contained and for reference purposes. This includes brief discussions on the relevant techniques involved.

2.3.1 Plemelj-Sokhotsky relations, Wiener-Hopf technique and Spitzer identities

While not directly referenced again, an introduction to these topics is required to discuss the pricing methodology presented later. A fundamental concept to the pricing methodology is the concept of the plus and minus parts of a function, typically denoted $f_+(x) = f(x)\mathbf{1}_{\mathbb{R}_+}(x)$ and $f_-(x) = f(x)\mathbf{1}_{\mathbb{R}_-}(x)$. For the most part, the Fourier transform of these function components is required, i.e. $\widehat{f}_+(\xi) = \mathcal{F}_{x\to\xi}[f(x)\mathbf{1}_{\mathbb{R}_+}(x)]$ and $\widehat{f}_-(\xi) = \mathcal{F}_{x\to\xi}[f(x)\mathbf{1}_{\mathbb{R}_-}(x)]$. One can obtain these transforms directly by applying the Hilbert transform and calculating the Plemelj-Sokhotsky relations. For a full discussion, see Germano et al. [2018]. These relations utilise the fact that applying the Hilbert transform in the Fourier domain is equivalent to multiplying the function by $-i\mathrm{sgn}(x)$ in the state domain. Therefore, since $f(x)\mathrm{sgn}(x) = f_+(x) + f_-(x)$ and the Hilbert transform is a linear function we obtain

$$-i\mathcal{H}[\widehat{f}(\xi)] = \widehat{f}_+(\xi) - \widehat{f}_-(\xi). \tag{2.53}$$

Since the Fourier transform is likewise a linear function we can combine Eq. (2.53) with the fact that $\widehat{f}(\xi) = \widehat{f}_+(\xi) - \widehat{f}_-(\xi)$ to recover the Plemelj-Sokhotsky relations, which split a given function into its positive and negative parts through decomposition

$$\widehat{f}_+(\xi) = \frac{1}{2}\left(\widehat{f}(\xi) - i\mathcal{H}\widehat{f}(\xi)\right), \tag{2.54}$$

$$\widehat{f}_-(\xi) = \frac{1}{2}\left(\widehat{f}(\xi) + i\mathcal{H}\widehat{f}(\xi)\right). \tag{2.55}$$

In the case of applying the decomposition to a barrier option, it would be useful to have a generalised version of the decomposition applied to a function, which results in positive and negative parts of the function relating to its position above or below an arbitrary barrier of value b. To do this, we apply the Fourier shift theorem discussed in Section 2.2.4, resulting in the generalised form

$$\widehat{f}_{b+}(\xi) = \frac{1}{2}\left(\widehat{f}(\xi) - e^{ib\xi}i\mathcal{H}[e^{ib\xi}\widehat{f}(\xi)]\right), \tag{2.56}$$

$$\widehat{f}_{b-}(\xi) = \frac{1}{2}\left(\widehat{f}(\xi) + e^{ib\xi}i\mathcal{H}[e^{ib\xi}\widehat{f}(\xi)]\right). \tag{2.57}$$

These equations can be combined to produce the Fourier transform for the section of the function between two arbitrary barriers l and u in the case of a double barrier option,

$$\widehat{f}_{lu}(\xi) = \frac{1}{2}\left(e^{il\xi}i\mathcal{H}[e^{il\xi}\widehat{f}(\xi)] - e^{iu\xi}i\mathcal{H}[e^{iu\xi}\widehat{f}(\xi)]\right). \tag{2.58}$$

The numerical methods that follow in later parts of the thesis require the additional feature of factorisation, i.e. the elements required for the calculation $\widehat{f}(\xi) =$

$\widehat{f}_{\oplus}(\xi)\widehat{f}_{\ominus}(\xi)$. The exclusive use of $\oplus$ and $\ominus$ is used for factorisation and distinguishes it from decomposition. The process is performed by arithmetically decomposing the logarithm of a function $\widehat{h}(\xi) = \log \widehat{f}(\xi)$, i.e.

$$\widehat{h}_{+}(\xi) = \frac{1}{2}\left(\widehat{h}(\xi) - i\mathcal{H}[\widehat{h}(\xi)]\right) = \frac{1}{2}\left(\log \widehat{f}(\xi) - i\mathcal{H}[\log \widehat{f}(\xi)]\right), \qquad (2.59)$$

$$\widehat{h}_{-}(\xi) = \frac{1}{2}\left(\widehat{h}(\xi) + i\mathcal{H}[\widehat{h}(\xi)]\right) = \frac{1}{2}\left(\log \widehat{f}(\xi) + i\mathcal{H}[\log \widehat{f}(\xi)]\right). \qquad (2.60)$$

Due to its extensive use in option pricing, another topic to introduce is the Wiener-Hopf technique. Due to the breadth and depth of this topic and its limited use within this thesis, only a brief discussion is dedicated herein. An interested reader is encouraged to examine Wiener and Hopf [1931] and Polyanin and Manzhirov [1998] for a comprehensive look at the area. For a more modern reference, consult Lawrie and Abrahams [2007], who cover historical applications. A Wiener-Hopf equation is of the form

$$\lambda f(x) - \int_{a}^{b} k(x - x')f(x')dx' = g(x) \quad x \in (a, b), \qquad (2.61)$$

where we intend to solve for $f(x)$. The functions $k(x)$ and $g(x)$ are known and are often referred to as the kernel and forcing functions, respectively. If either $a = -\infty$ or $b = +\infty$, the expression Eq. (2.61) is a Wiener-Hopf equation. Alternatively, if a and b are both finite, it is referred to as a Fredholm equation. When $\lambda = 0$, it is an equation of the first kind. When $\lambda \neq 0$, it is an equation of the second kind. While many applications for the technique exist, we are primarily interested in its application to price discretely monitored path-dependent options. This is seen in two distinct approaches. A direct calculation to price the option directly, as seen in Fusai et al. [2006], or by finding the probability distributions of the underlying processes subject to discrete monitoring, as presented by Green et al. [2010] and Fusai et al. [2016].

Related to the Wiener-Hopf technique in probability problems are the Spitzer identities. Spitzer [1957] presented fluctuation identities, developed using combinatorial arguments, for the Fourier-z transform of the probability distributions for the maximum and minimum of a process under discrete monitoring and the final value of a process conditional on the crossing of a discretely monitored barrier. The identities were extended to the continuous and double barrier cases by Baxter and Donsker [1957] and by Kemperman [1963], respectively. The link to the Wiener-Hopf factorisation was initially discussed by Baxter [1961] and further investigated by Green et al. [2010], with a focus on pricing options. This work focused on using a variation on the method, referred to as Jones' method or lemma, Jones [1952], to apply the method using Fourier integrals rather than directly. This enables a transformation of the Spitzer identities into a numerically tractable form.

Similarly to the Wiener-Hopf technique, the field and applications of Spitzer's identities are well-established and widely known. An entire work could be dedicated to the breadth and scope of the applications. For brevity, we note the applications to queuing systems, see Cohen [1975] and more recently Bayer and Boxma [1996], fields in applied probability such as Markov chains, see Rogers [1994] and areas within mathematical finance outside of the scope of this thesis such as insurance, see Chi and Lin [2011]. Due to the complex nature of the Spitzer identities, a comprehensive discussion has been omitted. The interested reader looking for a comprehensive guide should consult Green [2009].

2.3.2 Feng and Linestky's method

As shown in section 2.2.4, we can use the Hilbert transform to obtain the Fourier transform of the part of a function above or below a barrier or between two barriers. Since this feature is fundamental to the method presented by Feng and Linetsky [2008], we refer to the method as the inverse Hilbert method (IH). This feature of the Hilbert transform allows the price to be calculated between two successive monitoring dates

$$v(x, t_{n-1}) = \int_l^u v(x', t_n) k(x - x', \Delta t) dx', \tag{2.62}$$

where $v(x, t_N) = \phi(x) e^{-\alpha_d x}$ is the damped payoff of the option, $k(\cdot, \Delta t)$ is the transition density of the underlying process with step size Δt and $\Psi(\xi, \Delta t)$ is its characteristic function. By applying first the Hilbert transform and then a convolution, the price at two successive dates can be expressed as

$$\widehat{v}(\xi, t_{n-1}) = \frac{1}{2}\Big(\Psi(\xi - i\alpha_d, \Delta t)\widehat{v}(\xi, t_n) - ie^{il\xi}\mathcal{H}\left[e^{-il\xi}\Psi(\xi - i\alpha_d, \Delta t)\widehat{v}(\xi, t_n)\right]\Big), \quad (2.63)$$

for a single down-and-out barrier option. For the double barrier knock-out case, this is rearranged to

$$\begin{aligned}\widehat{v}(\xi, t_{n-1}) = \frac{1}{2}\Big(& - ie^{il\xi}\mathcal{H}\left[e^{-il\xi}\Psi(\xi - i\alpha_d, \Delta t)\widehat{v}(\xi, t_n)\right] \\ & + ie^{iu\xi}\mathcal{H}\left[e^{-iu\xi}\Psi(\xi - i\alpha_d, \Delta t)\widehat{v}(\xi, t_n)\right]\Big).\end{aligned} \quad (2.64)$$

2.3.3 Spitzer based method for barrier options (ZS)

For brevity and without loss of generality, we describe the pricing procedure for a single-barrier down-and-out option. The Spitzer identities can be used to price any barrier option type and applied to lookback options. The methodologies modified in this paper were devised and explained in depth by Fusai et al. [2016], and the modification applies to the range of options contracts the methodology covers.

1. Set the number of monitoring dates to $N-2$ so that the characteristic function can be applied as a smoothing function for the first and last monitored dates.

2. Compute the characteristic function $\Psi(\xi - i\alpha, \Delta t)$, where α is the damping factor.

3. Use the Plemelj-Sokhotsky relations

$$\hat{f}_+(\xi) = \frac{1}{2}\{\hat{f}(\xi) - i\mathcal{H}[\hat{f}(\xi)]\}, \quad (2.65)$$

$$\hat{f}_-(\xi) = \frac{1}{2}\{\hat{f}(\xi) + i\mathcal{H}[\hat{f}(\xi)]\}, \quad (2.66)$$

with the sinc-based Hilbert transform to factorise

$$\Phi(\xi, q) = 1 - q\Phi(\xi - i\alpha, \Delta t) = \Phi_{\oplus}(\xi, q)\Phi_{\ominus}(\xi, q), \quad (2.67)$$

with q selected according to the chosen inverse z-transform.

4. Decompose

$$P(\xi,q)=\frac{e^{-il\xi}\Psi(\xi-i\alpha,\Delta t)}{\Phi_-(\xi,q)}=P_+(\xi,q)+P_-(\xi,q), \tag{2.68}$$

   and calculate

$$F(\xi,q)=\hat{\phi}^*(\xi)\Psi(\xi-i\alpha,\Delta t)e^{il\xi}\frac{P_+(\xi,q)}{\Phi_+(\xi,q)}. \tag{2.69}$$

5. Calculate the price

$$V(0,N)=e^{-rT}\mathcal{F}^{-1}_{\xi\to x=0}\left[\mathcal{Z}^{-1}_{q\to n=N-2}\left[F(\xi,q)\right]\right]. \tag{2.70}$$

The Spitzer identities provide the z-transform of the characteristic function. Thus, the inverse z and the Fourier transform must be applied to recover the price.

Many methods for using Spitzer identities to price options are direct. A notable extension is the case of the double-barrier option. It is similar to the single-barrier case in that it, too, requires Wiener-Hopf factorisation and decomposition to compute the corresponding Spitzer identities. However, these equations cannot be solved directly and need a fixed-point algorithm to numerically approximate the identities iteratively. The pricing procedure is identical to the single-barrier case, except that step 3 is replaced by the fixed-point algorithm.

3. Set $J(\xi,q)=0$ and decompose

$$\begin{aligned}P(\xi,q)&=\sigma\left(\frac{\xi}{\xi_{\max}}\right)\left[\frac{e^{-il\xi}}{\Phi_{X-}(\xi,q)}-\frac{e^{i(u-l)\xi}J_{X+}(\xi,q)}{\Phi_{X-}(\xi,q)}\right]\\&=P_{X_M}(\xi)+P_{X_m}(\xi),\end{aligned} \tag{2.71}$$

   and calculate $J_-(\xi,q)=P_-(\xi,q)\Phi_-(\xi,q)$.

b) Decompose

$$\begin{aligned}Q(\xi,q)&=\sigma\left(\frac{\xi}{\xi_{\max}}\right)\left[\frac{e^{-iu\xi}}{\Phi_+(\xi,q)}-\frac{e^{i(l-u)\xi}J_-(\xi,q)}{\Phi_+(\xi,q)}\right]\\&=Q_+(\xi,q)+Q_-(\xi,q),\end{aligned} \tag{2.72}$$

and calculate $J_+(\xi) = Q_+(\xi, q)\Phi_+(\xi, q)$.

c) Calculate

$$\hat{p}(\xi, q) = \sigma\left(\frac{\xi}{\xi_{\max}}\right)\left[\frac{1}{\Phi(\xi, q)} - \frac{e^{il\xi}J_-(\xi, q)}{\Phi(\xi, q)} - \frac{e^{iu\xi}J_+(\xi, q)}{\Phi(\xi, q)}\right]. \tag{2.73}$$

d) If the difference between the current and previous version of $\hat{p}(\xi, q)$ is higher than the predefined tolerance and the number of iterations is less than an agreeable threshold, return to step b). Otherwise, proceed to step 5 and Eq. (2.70), calculating the price of the double barrier option. Numerical tests indicate that an iteration threshold of five to six should be sufficient, as further iterations yield minimal improvements.

2.3.4 Spitzer based method for α-quantile options (ZS)

We present the general methodology for pricing α-quantile options following Phelan et al. [2020]. We price a discretely monitored α-quantile option with a uniform monitoring interval. Given the number of monitoring dates N, we express the predefined time beyond the barrier $j = \alpha N$ to the nearest integer. The pricing procedure is then given as follows.

1. Compute the characteristic function $\Phi(\xi - i\alpha_d, \Delta t)$ of the underlying transition density, where α_d is the damping parameter applied on the option payoff.

2. Using the Plemelj-Sokhotsky relations and the sinc-based Hilbert transform to factorise

$$\Phi(\xi, q) = 1 - q\sigma\left(\frac{\xi}{\xi_{\max}}\Psi(\xi - i\alpha_d, \Delta t)\right) = \Phi_+(\xi, q)\Phi_-(\xi, q), \tag{2.74}$$

where $\sigma(x)$ is an exponential filter and q is selected according to the chosen inverse z-transform.

3. Calculate the Fourier-z transform of the probability density function of the

maximum X_M and minimum X_m,

$$P_{X_M}(\xi, q) = \frac{1}{\Phi_+(\xi, q)\Phi_-(0, q)}, \tag{2.75}$$

$$P_{X_m}(\xi, q) = \frac{1}{\Phi_+(0, q)\Phi_-(\xi, q)}. \tag{2.76}$$

4. Apply the inverse z-transform for j and $N - j$ dates respectively,

$$P_{X_M}(\xi, j) = \mathcal{Z}^{-1}_{q\to j}\left[P_{X_M}(\xi, q)\right], \tag{2.77}$$

$$P_{X_m}(\xi, N - j) = \mathcal{Z}^{-1}_{q\to N-j}\left[P_{X_m}(\xi, q)\right]. \tag{2.78}$$

5. Eliminate the numerical error in the characteristic functions X_M and X_m that corresponds to the spurious imaginary part of the probability density function to avoid its propagation later in the method

$$\widehat{P}^{\text{Re}}_{X_M}(\xi, j) = \mathcal{F}_{x\to\xi}\left[\text{Re}P_{X_M}(\xi, j)\right] = \frac{1}{2}\left[\widehat{P}_{X_M}(\xi, j) + \widehat{P}^*_{X_M}(-\xi, j)\right], \tag{2.79}$$

$$\widehat{P}^{\text{Re}}_{X_m}(\xi, N-j) = \mathcal{F}_{x\to\xi}\left[\text{Re}P_{X_m}(\xi, j)\right] = \frac{1}{2}\left[\widehat{P}_{X_m}(\xi, N - j) + \widehat{P}^*_{X_m}(-\xi, N - j)\right]. \tag{2.80}$$

6. Calculate the characteristic function of X_α at the required monitoring date N,

$$\widehat{P}_{X_\alpha}(\xi, N) = \widehat{P}^{\text{Re}}_{X_M}(\xi, j)\widehat{P}^{\text{Re}}_{X_m}(\xi, N - j). \tag{2.81}$$

7. Calculate the price of the discretely monitored α-quantile option at time origin,

$$V(0, 0) = \mathcal{F}^{-1}_{\xi\to x}\left[\sigma\left(\frac{\xi}{\xi_{\max}}\right)\widehat{P}_{X_\alpha}(\xi, N)\widehat{\phi}^*(\xi)\right]. \tag{2.82}$$

2.4 Extensions outside of options pricing

Although this thesis primarily focuses on financial applications, the Fourier-z framework and the numerical methods it contains can be applied in various other disciplines. The core advantage stems from the ability to recover discrete-time or discrete-state solutions from a function expressed in terms of its z-transform, often leading to more transparent or computationally efficient procedures than direct inversion by contour integration or naive summation. This technique extends to any problem where time evolution or discrete convolution plays a pivotal role.

A prime example is the field of stochastics, where models evolve in discrete steps through Markov chains, random walks, or branching processes. In such settings, the Fourier-z framework aligns well with established results, such as the Spitzer identities, which connect random walk functionals to generating functions. When traditional techniques struggle because closed-form solutions do not exist or standard summations become numerically unstable, Fourier-z methods can deliver reliable computations for hitting times, overshoot probabilities, or state distributions. This benefit extends naturally into areas such as queueing theory or risk processes, where summing series for distributions is common but can be computationally intensive.

One of the key advantages of the Fourier-z approach in stochastic processes is its ability to handle discrete-time random walks and related sums of random variables. Suppose $\{X_k\}$ is a sequence of i.i.d. random variables with common probability mass function $p_X(x)$, and let

$$S(n) = \sum_{k=1}^{n} X(k), \tag{2.83}$$

where $n \geq 1$. In many applications, such as gambler's ruin, overshoot probabilities, or boundary-crossing times, the $S(n)$ distribution is critical. The z-transform of $S(n)$, often purely known as a generating function in probability, takes the product form

$$G_{S(n)}(z) = \left[G_X(z)\right]^n, \tag{2.84}$$

where

$$G_X(z) = \sum_{x} p_X(x) z^x, \tag{2.85}$$

where $p_X(x)$ is the probability mass function. Inverting this transform analytically or numerically reveals the full probability distribution of $S(n)$.

One of the central identities in this context is the Spitzer identities, which express certain random-walk functionals (e.g., the maximum of a partial sum) in terms of generating functions of the increment distribution. By combining such identities with robust z-transform inversion methods, problems that were once too cumbersome to tackle can be approached via naive summation or direct combinatorial arguments.

Furthermore, if one is interested in the probability of hitting a boundary, say $S(n) < 0$ for some n, the Spitzer identities and related theorems allow us to express the necessary quantities in a closed or semi-closed form involving z-transforms of $X(k)$. For instance, overshoot distributions, how far a random walk surpasses a boundary, can be easily recovered by tracking the series expansion within the z-domain and applying an inverse transform. More straightforward inversion schemes sometimes struggle with convergence or numerical instability. This is particularly evident when distributions have heavy tails or exhibit oscillatory behaviour. Though illustrated with option pricing, the methods developed in this thesis can likewise stabilise or accelerate the partial sums in these contexts. Thus, the Fourier-z framework, combined with other tools such as the Spitzer identities, provides a systematic method for deriving and approximating probability laws in discrete stochastic models.

Beyond probability and queueing systems, these methods have applications in signal processing and physics. The z-transform is a fundamental tool for analysing discrete-time filters and signals in signal processing. However, the usual practice often relies on transforms that fit neatly into standard tables. When faced with more unusual or intricate systems, an accurate numerical inversion of the z-transform is required for reconstructing time-domain signals. In physics, discrete lattice models similarly lend themselves to transform-based representations. This enables one to characterise complex state transitions or particle dynamics. Here, too, specialised inversion algorithms that stabilise or accelerate partial sums can enhance the precision and speed of solutions, making them valuable in scenarios where the standard toolbox proves inadequate.

While the methods introduced in this thesis are devised primarily to improve the inverse z-transform calculation for option pricing, the advancements have the potential to advance discrete modelling in multiple disciplines.

Chapter 3

Inverse z-transform

This chapter reviews the numerical approach to the inverse z-transform and explores series acceleration techniques to reduce the computational cost of an intensive procedure. The literature surrounding the inverse z-transform is sporadic at best. Since it is the cause of the error floor observed in the Fourier-z approach to option pricing, it is worthwhile to explore finding a more optimal solution. Here, we present a comprehensive review of the numerical methods for inverting the z-transform, as found in the literature, within a unified framework. Discussing the technical and historical relationships between the z-transform and its continuous counterparts, the Laplace and Fourier transforms, and the basic convergence of the z-transform is beneficial.

3.1 Convergence

The z-transform does not converge for all functions or all values of z in a convergent function. For any given function, the set of values of z for which the z-transform converges is known as the region of convergence (ROC). The ROC of the z-transform is expressed as a circular region or a disc within the complex z-plane and takes the form $\{z : |z| > a\}$, where it excludes all poles the function may have. Thus, Eq. 2.36 is convergent within the function's ROC and divergent outside. For finite duration causal systems, most commonly a right-sided sequence, the ROC is the entire z-plane with the exclusion of any poles and at $z = 0$. Conversely, for infinite duration causal systems, the ROC is the exterior of the circle with radius a, i.e. $|z| > a$. Other standard formulations exist; for example, if $f(n)$ is an anti-causal finite system, the ROC will be the circle's interior in the complex z-plane with radius a, i.e. $|z| < a$. In most numerical cases, the ROC may include some points on the circle with radius a acting as the boundary. Depending on the target function, it may be convergent or divergent at this boundary.

Most numerical validation cases presented in this paper are finite or infinite causal systems, resulting in right-sided sequences. With these boundaries in place, causality and stability are two natural conditions to formulate. A discrete-time system is causal when the ROC is outside the outermost pole, which one can attempt to satisfy in a computational sense by setting certain parameters. There are two general conditions for stability. Firstly, the function $\widetilde{f}(z)$ includes the unit circle $|z| = 1$. Secondly, all poles of the function lie inside the unit circle $|z| = 1$. For a more in-depth discussion on the ROC and related issues, consult Oppenheim [2010]. The case of discrete option pricing results in stable causal systems, but this is highly dependent on modelling considerations such as the chosen probability distributions governing asset price movements and contractual considerations such as the number of monitoring dates. A full discussion is presented in Chapter 4.

3.2 The Laplace transform

The z-transform can be understood as the discrete analogue of the Laplace transform, which extends to unilateral and bilateral cases. Historically, the Laplace transform pre-dates the z-transform and, along with the Fourier transform, was adopted into many fields in engineering and applied mathematics by the early 20th century. The utility of the Laplace transform in solving differential equations, analysing continuous-time systems and formulating solutions to initial value problems ensured its widespread adoption. By the mid-20th century, the Laplace transform had become a central tool in engineering mathematics, analysis of linear time-invariant systems and control theory, see Doetsch [1974], Titchmarsh [1948], Brown and Churchill [2014] and Erdélyi [1954].

Conversely, the z-transform emerged later as digital computation and sampled-data systems began to be used post-World War II. An early reference to discrete transforms concerning difference equations appears, Hurewicz [1958], but without formal mathematical foundations. The z-transform itself finally gained prominence in discrete-time control theory and digital signal processing during the 1960's where Jury [1964] presented it in the context of digital systems. Standard textbooks on the subject would finally appear in the 1970s, see Oppenheim and Schafer [1975], making the z-transform integral synonymous with the growing field of discrete-time signals.

One may be surprised at the abundance of literature surrounding the Laplace transform with respect to the comparatively minimal amount on the z-transform. This fact can be due to several factors, the first being the academic approach to the z-transform, which, in practical terms, was inferior until the emergence of digital technologies. Before the advent of digitised machines, fields such as engineering, mathematics, and physics were predominantly concerned with continuous systems, such as signals, fluids and particle dynamics. The Laplace transform is integral to the modelling and solution of differential equations that can occur in these fields. Typically, as these continuous systems require complex equations, it has become a widely researched topic to improve the Laplace transform's numerical inversion to remain within the realms of mathematical tractability. Furthermore, within stability analysis and PDES, methods such as the Routh-Hurwitz criterion, root locus and frequency response techniques use the Laplace transform and often rely on a numerical inversion for more complex problems.

Moreover, the popularity of Laplace-based approaches is reflected in the breadth of specialized inversion algorithms: Talbot's method and the Gaver Wynn Rho (GWR) algorithm, for instance, are widely known for their reliability and efficiency, while multi-precision Laplace transform inversion techniques, as discussed by Abate and Valkó [2004], underscore the continued refinement and sophistication of Laplace inversion methods. These well-established frameworks and the historical prevalence of continuous-time modelling help explain why the Laplace transform and its numerical inversion boast a more extensive literature base than the z-transform.

Until the use of the z-transform became clear and discrete systems became increasingly relevant, it was simply not as needed as its continuous counterpart. Additionally, the need for the transform's numerical inversion has only recently become worthy of increased attention. This rise in applicability can be partly due to many solutions being available semi-analytically by analysing known z-pairs, especially in signal processing, since many of the systems under study are trigonometric. As a consequence, only scattered research existed until the late 1970s on numerical inversion, see Cavers [1978] and Papoulis [1973], and only became prominent in the late 1990s where Abate and Whitt [1992a] and Abate and Whitt [1992b] began to produce coherent frameworks for the z-transform.

The z-transform can be mathematically converted to the Laplace transform by taking $f(t_n)$ as the discretely sampled adaptation of its continuous counterpart $f(t)$ at the sampling points $t_n = n\Delta t$, and setting

$$z = e^{s\Delta t}, \tag{3.1}$$

the Laplace transform of $f(t)$ can be expressed as the z-transform of $f(t_n)$ in the continuous limit as $\Delta t \to 0$,

$$\begin{aligned} \mathcal{L}_{t\to s}[f(t)] &= \int_0^\infty f(t)e^{-st}dt \\ &= \lim_{\Delta t\to 0} \Delta t \sum_{n=0}^{\infty} f(n\Delta t)(e^{s\Delta t})^{-n} \\ &= \lim_{\Delta t\to 0} \Delta t \mathcal{Z}_{t_n\to e^{s\Delta t}}\big[f(n)\big]. \end{aligned} \tag{3.2}$$

By definition, there is a conformal mapping between the z and s-space where the angles between contours on each plane are preserved for all functions within the space. An example of such a conformal mapping can be shown in Phelan et al. [2020].

3.3 The Fourier transform

Another crucial aspect of this thesis is the relation between the Fourier transform and the z-transform. This relation follows that the Fourier transform is a special case of the bilateral Laplace transform where $\mathrm{Re}(s) = 0$, i.e. $s = i\omega = 2\pi i\xi$. Without loss of generality, we give an alternative form to the Fourier transform seen within computational formats to enable a clearer comparison with the z-transform,

$$\widehat{f}(\xi) = \mathcal{F}_{t\to\xi}[f(t)] = \int_{-\infty}^{+\infty} f(t)e^{-2\pi i\xi t}dt, \tag{3.3}$$

and inverse

$$f(t) = \mathcal{F}_{\xi\to t}[\widehat{f}(\xi)] = \int_{-\infty}^{+\infty} \widehat{f}(\xi)e^{2\pi i\xi t}d\xi. \tag{3.4}$$

Discretising this equation leads to the discrete Fourier transform with an additional discretisation error given by Δt and a truncation error given by N. The frequency grid in Fourier space $\xi_n = n\Delta\xi,\ n = -N/2, \dots, N/2-1$ is linked to the time grid $t_n = n\Delta t,\ n = -N/2, \dots, N/2-1$ by the Nyquist relation,

$$\Delta\xi\Delta t = \frac{1}{N}. \tag{3.5}$$

The N grid point index can be chosen to run from 0 to $N-1$ or from 1 to N. Therefore, the discrete Fourier transform is defined as

$$\widehat{f}(\xi_n) = \Delta t \sum_{k=0}^{N-1} f(t_k)e^{-2\pi i\xi_n t_k}. \tag{3.6}$$

whose inverse is obtained by inverting the sign of the Fourier factor

$$f(t_n) = \Delta\xi \sum_{k=0}^{N-1} \widehat{f}(\xi_k)e^{2\pi i\xi_k t_n}, \tag{3.7}$$

Where $\Delta\xi\Delta t = 1/N$. This choice of definition results from the Fourier transform being commonly defined as a function of the pulsation ω, which introduces a factor $1/(2\pi)$ into the equation and is most consistent with scientific computing. In addition, a negative sign appears in the exponent of the Fourier factor in the forward transform, which makes the characteristic function $E[e^{i\xi X}]$ of a random variable X equal to the Fourier transform $\mathcal{F}_{t\to\xi}[f_X(t)]$ of its PDF $f_X(t)$.

Given the definition of the z-transform expressed by Eq. (2.36), if we express the variable z in polar form, i.e. $z = e^{i\omega} = e^{2\pi i\xi}$, the z-transform equates to the discrete Fourier transform, Eq. (3.6), on a positive infinite grid $t_n = 0, 1, \ldots$. Therefore, under the assumption of existence, the discrete Fourier transform $\widehat{f}(\xi_n)$ is equivalent to the z-transform $\widetilde{f}(z)$ with the substitution $z = e^{2\pi i\xi_n}$. A restriction of unit magnitude for z, i.e. $|z| = 1$, ensures correspondence between the two. This is equivalent to restricting z to the unit circle in the complex z-plane. Thus, Eq. (2.36) can be expressed as

$$\widetilde{f}(re^{2\pi i\xi}) = \sum_{t_n=0}^{\infty} f(t_n)(re^{2\pi i\xi})^{-n}, \tag{3.8}$$

which is equivalent to the discrete Fourier transform of the product of the exponential sequence r^{-n} with the original function $f(t_n)$. For the case $r = 1$, Eq. (3.8) reduces to the discrete Fourier transform applied to $f(t_n)$.

3.4 Probability Generating Functions

In probability theory, the z-transform of a non-negative random variable X is closely related to its probability-generating function (PGF). The PGF is defined as

$$G_X(z) = \mathcal{E}\left[z^X\right] = \sum_{n=0}^{\infty} p_X(n) z^n. \tag{3.9}$$

From a control theory perspective, the z-transform of the sequence $p_X(n)$ is often given by

$$F(z) = \sum_{n=0}^{\infty} p_X(n) z^{-n}, \tag{3.10}$$

adopting the conventions momentarily of Oppenheim [2010]. Despite the reversed exponent, both representations store the same coefficient information about the probability mass function (PMF) of X. This fact makes the PGF and the z-transform interchangeable in many cases in probability and engineering contexts.

To recover the PMF $p_X(n)$ from the generating function $G_x(z)$, one can extract the coefficient z^n. Thus, if

$$G_X(z) = \sum_{n=0}^{\infty} p_X(n)\, z^n,$$

then

$$p_X(n) = [z^n]\, G_X(z), \tag{3.11}$$

where $[z^n]$ denotes the coefficient-extraction operator (i.e., the coefficient of z^n). In the case of the z-transform $F(z)$, the inversion can proceed via contour integration

$$p_X(n) = \frac{1}{2\pi i} \oint_\Gamma F(z)\, z^{n-1}\, \mathrm{d}z, \tag{3.12}$$

where Γ is the contour encircling the origin, Davenport [1970].

This connection between the PGF and the z-transform allows many practical applications in probability and related fields, such as stochastic modelling, queuing theory, and option pricing.

3.5 Numerical inversion approaches

This section reviews the current methodology for numerically inverting the inverse z-transform. These methods are benchmarked and results are discussed in Section 3.8. These methods follow the general trend of a matrix decomposition and have been grouped accordingly.

3.5.1 Linear system of equations

Presented by Merrikh-Bayat [2014], in the case where $f(n)$ is a suitable function where $\widetilde{f}(z)$ is known or analytical, it is possible to form a linear system of equations approximating $f(n)$. This method takes the explicit assumption that $\widetilde{f}(z)$ has no poles at infinity, i.e. $\lim_{|z|\to\infty} \widetilde{f}(z) = 0$. The expression given in Eq. (2.36) holds for any value of z that lies within the region of convergence, defined in Section. (2.38). Taking a single point z_1 inside the region of convergence the inverse transform for z_1 can be given by

$$\widetilde{f}(z_1) \approx \sum_{n=0}^{N} f(n) z_1^{-n}. \tag{3.13}$$

Taking a series of such random points $z_1, z_2, \ldots, z_m$ within the region of convergence, it is possible to formulate the problem as a system of algebraic equations approximating $\widetilde{f}(z_1), \widetilde{f}(z_2), \ldots, \widetilde{f}(z_m)$ given by

$$\begin{bmatrix} \widetilde{f}(z_1) \\ \widetilde{f}(z_2) \\ \vdots \\ \widetilde{f}(z_m) \end{bmatrix} = \begin{bmatrix} 1 & z_1^{-1} & z_1^{-2} \ldots & z_1^{-N} \\ 1 & z_2^{-1} & z_2^{-2} \ldots & z_2^{-N} \\ \vdots & & \ddots & \vdots \\ 1 & z_m^{-1} & z_m^{-2} \ldots & z_m^{-N} \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ f(m) \end{bmatrix}, \tag{3.14}$$

where m is the number of points chosen from within the region of convergence. Thus, Eq. (3.14) can be written as a linear system of the form $\widetilde{\mathbf{f}}(z) = \mathbf{Af}(n)$ containing N variables and m equations where

$$\mathbf{A} = \begin{bmatrix} 1 & z_1^{-1} & z_1^{-2} \ldots & z_1^{-N} \\ 1 & z_2^{-1} & z_2^{-2} \ldots & z_2^{-N} \\ \vdots & & \ddots & \vdots \\ 1 & z_m^{-1} & z_m^{-2} \ldots & z_m^{-N} \end{bmatrix}. \tag{3.15}$$

The downside of such an approach consists mainly of ensuring that a unique solution is available when one performs matrix inversions. A unique solution is theoretically found where $m = n$ and the random points $z_1, z_2, \ldots, z_m$ are chosen to ensure that $\mathbf{A}$ is of full rank. Under these assumptions, the solution of Eq. (3.14) is found by minimising $\|\mathbf{Af}(n) - \widetilde{\mathbf{f}}(z)\|_2$. This can be done within environments such as Matlab using the backslash command, i.e. $\mathbf{f}(n) = \mathbf{A} \setminus \widetilde{\mathbf{f}}(z)$.

This method is awkward to implement. Furthermore, if any point z_i is chosen within the unit circle, the system becomes ill-conditioned, which gets worse as N increases. This method is more heuristic and unreliable, as simple functions like the Heaviside step and decaying exponentials do not work under this framework. While the author presents no specific error bounds, our experiments can not achieve an accuracy greater than E-03. This general low performance and the computational cost associated with inverting the matrix are omitted from our accompanying code.

3.5.2 Orthogonal decomposition

A less-widely known method proposed by Rajković et al. [2004] and originating from an earlier paper, Papoulis [1973], relies on finding orthogonal functions that can be decomposed to find the inverse transform. The method approximates $f(n)$ by

$$f_{N,q}(n) = \sum_{k=0}^{N} c_k \phi_r^k(n), \tag{3.16}$$

where

$$\phi_r^k(n) = \sum_{m=1}^{n} b_{r,n,m} r^m, \tag{3.17}$$

where r is an input parameter selected to be smaller than 1. The authors recommend setting $r = 5/6$ or $r = 3/4$. The coefficients $b_{r,n,m}$ are calculated using binomial calculations given as

$$b_{r,n,m} = (-1)^{n-m} r^p \binom{n}{m} \binom{n+k-1}{k-1}, \tag{3.18}$$

where $p = -\binom{n}{2} + \binom{m+1}{2} - mn$ and the c_n coefficients are calculated as

$$c_k = \frac{r^{k(1-k)}}{1-r^{2k}} \sum_{l=1}^{k} b_{r,k,l} \widetilde{f}(1/q^l). \tag{3.19}$$

The values $b_{r,n,m}$ are chosen such that $\phi_r^{(0)}(n), \phi_r^{(1)}(n), \ldots, \phi_r^{(N)}(n)$ form a set of orthogonal series. The author presents some error analysis and presents a minimum relative error of E-06 with double precision.

3.5.3 Matrix exponential

A method presented in Horváth et al. [2020] and a similar methodology to that presented by Lian et al. [2017] revolves around matrix exponentials. This relies upon a similar technique for inverting Laplace transforms, namely the concentrated matrix exponential (CME) method. Using the unified framework for the Laplace transform given by [Abate and Whitt, 2006], the inverse transform can be approximated by

$$f(T) = \int_0^\infty \delta(T-t) f(t) dt, \tag{3.20}$$

where $\delta(t)$ is a Dirac delta function. Under the framework, the function $\delta(T-t)$ is approximated by a weighted sum of exponentials consisting of n terms

$$\delta(T-t) \approx \frac{1}{T} \sum_{k=1}^{n} \nu_k e^{-\beta_k \frac{t}{T}}. \tag{3.21}$$

Therefore, the function $f(T)$ can be expressed as the linear combination of its Laplace transform at particular points

$$f(T) = \int_0^\infty \delta(T-t) f(t) dt \approx \frac{1}{T} \sum_{k=1}^{n} \nu_k e^{-\frac{\beta_k}{T} t} f(t) = \frac{1}{T} \sum_{k=1}^{n} \nu_k \tilde{f}\left(\frac{\beta_k}{T}\right). \tag{3.22}$$

A matrix exponential distribution's probability density function (pdf) is chosen as the function $f(t)$. The resultant class of matrix exponential distributions of order N contains positive random variables with a pdf of the form

$$f_X(t) = -\underline{\alpha} \boldsymbol{A} e^{\boldsymbol{A} t} \underline{\mathbf{1}}, \quad t \geq 0, \tag{3.23}$$

where $\underline{\alpha}$ is a row vector and $\boldsymbol{A}$ is a matrix of size $N \times N$. Since $f_X(t)$ is non-negative and under the assumption that $\boldsymbol{A}$ is diagonalisable with spectral decomposition

$$\boldsymbol{A} = \sum_{k=1}^{N} \lambda_k u_k \nu_k, \tag{3.24}$$

where λ_k are the eigenvalues, u_k are the right eigenvectors and ν_k are the left eigenvectors of $\boldsymbol{A}$ then $f_X(t)$ can be rewritten as

$$f_X(t) = \sum_{k=1}^{N} \left(-\alpha \, \boldsymbol{A} u_k \, \nu_k\right) e^{\lambda_k t} = \sum_{k=1}^{N} \nu_k \, e^{-\beta_k t}, \tag{3.25}$$

where $\beta_k = -\lambda_k$ and the weight $c_k = -\alpha \boldsymbol{A} u_k \nu_k$ is absorbed into ν_k in the final sum. This method is a discretised version of the above process where the Dirac delta function is approximated via a series of impulses. This method is satisfactory but suffers from high CPU times due to handling and inverting large matrices. It can be presented, however, as a very stable method due to the restriction on the range of parameters that require selection. The authors' numerical tests show the best absolute error of E-09, but for a wide range of inversion cases. Matrix power acceleration techniques could handle the computational inefficiencies, but this has yet to be explored.

3.6 Contour integration methods

At this stage, a reasonable question can be asked about why we cannot calculate Eq. (2.38) directly using numerical techniques to approximate an integral performed on a contour. This numerical integration can be performed on the contour $c = re^{2\pi i\nu}$ where $0 \leq \nu \leq 1$, under the assumptions that the region of convergence is equivalent to $|z| > a$ where r and a are arbitrarily chosen constants such that $r > a$. While feasible, the main barrier to this approach is the lack of accuracy and computational efficiency. Specifically for smaller values of n, the contour integration of Eq. (2.38) becomes less accurate, while conversely, higher values create high computational costs. This results from repeatedly recalculating the integral for each value of n.

This restriction results in a general array of methods based upon the discussion in Sec (2.3) where Fourier series and analysis can be applied to a discrete system and hence approximate the contour integral Eq. (2.38). Using this approach, one can delve into the extensive literature surrounding Fourier transforms and series. A more computationally convenient form for Eq. (2.38) is

$$f(n) = \mathcal{Z}^{-1}_{z \to n}[\widetilde{f}(z)] = \frac{r^n}{2\pi} \int_0^{2\pi} \widetilde{f}(re^{iu}) e^{-inu} du, \tag{3.26}$$

which corresponds to the case where C is a circle of radius r in the complex plane. The variable r often acts as a scaling parameter and can be considered the radius of the region of convergence. Computationally, this parameter can significantly impact this class of methods.

All contour-based methods rely on the use of Eq. (3.8) to relate the inverse z-transform to that of a discrete inverse Fourier transform, which in turn can be calculated with ease by either using the FFTW Frigo and Johnson [1997] or some other approximation technique. As seen previously in Section 3.3 the variable z can be expressed in polar form, $z = re^{i\theta}$ where $\theta = 2\pi k/N$. Thus, the inverse z-transform can be reduced to a discrete Fourier transform given by Eq. (3.6). If one takes N samples of $\widetilde{f}(re^{i\theta})$ and denotes this as the periodic series $\widetilde{F}[k]$ with period N, it is possible to prove that if $\widetilde{F}[k]$ is considered as the coefficients of a periodic function such as $\widetilde{f}(z)$, then $f(n)$ and $\widetilde{f}(z)$ are related through the following equation

$$\widetilde{f}(z) = \sum_{k=0}^{\infty} f(n - rkN). \tag{3.27}$$

Furthermore, the inverse discrete Fourier transform of the series $\widetilde{F}[k] = \widetilde{f}(re^{i\theta})$ is equal to the inverse z-transform applied to the function $f(n)$ under the Fourier convention chosen. The assumption is that all poles are contained strictly within the unit circle, with discontinuities only at zero. To proceed with calculating the inverse z-transform we calculate $\widetilde{f}(z)$ on the contour $C = e^{2\pi in/N}$, where $n = 1, \ldots, N$. The contour chosen is not the only choice, as any partition can be taken in theory. This particular partition is chosen for its simplicity and symmetry. Thus, the inverse z-transform becomes

$$f(n) \approx \frac{1}{N} \sum_{k=0}^{N-1} r^n \widetilde{f}(re^{2\pi ik/N}) e^{2\pi ikn/N}, \tag{3.28}$$

which is equivalent to an inverse discrete Fourier transform of the function $\widetilde{f}(re^{2\pi in/N})$ multiplied by the scaling factor r^n. The summation can be computed using an inverse Fourier transform multiplied by r^n. Several methods have been presented using this approach, and the two most notable ones are discussed below. These methods tend to result in the most accurate approximations and can be extended easily beyond the case of analytic z-transform pairs.

3.6.1 Abate and Whitt (1992)

In various papers, Abate and Whitt [1992a,b,c], developed an approximate inversion formula based upon a Fourier series approach, following from the discussion in

Section 3.3. The Cauchy integral form of the inverse z-transform, Eq. (2.38), can be approximated numerically by applying a trapezoidal rule

$$f(n) \approx \frac{1}{N\rho^n}\Big(\widetilde{f}(\rho) + 2\sum_{k=1}^{N}(-1)^k \widetilde{f}\big(\rho e^{ik\pi/N}\big) + (-1)^N \widetilde{f}(-\rho)\Big), \tag{3.29}$$

where the error is controlled only by the parameter ρ. The authors state that an accuracy of order $10^{-\lambda}$ requires $\rho = 10^{-\lambda/2k}$. For practical purposes one can consider an error bound ρ^{2k} since $\rho^{2k}/(1-\rho^{2k})$ is approximately equal to ρ^{2k} when ρ^{2k} is small. This error bound on the error often results in an error floor of around E-11. The resulting method is remarkably simple to program and can be done with minimal code.

3.6.2 Cavers (1978)

The following method is derived from Section 3.3 discussion. It exploits the relation to the discrete Fourier transform, which can be conveniently calculated using a fast Fourier transform (FFT) and is similar to the method by Abate and Whitt. The key difference is that a new free parameter N defines a secondary grid. Cavers [1978] produces the earliest known presentation of this method; it has, however, been presented in several other papers with varying degrees of completeness, for example, Merrikh-Bayat [2014], Brancik [2003] and Mills [1987].

Cavers' method employs the discrete Fourier transform approach to approximate the inverse z-transform Eq. (3.28) and applies the FFT directly

$$f(n) = r^n \mathcal{F}^{-1}_{k\to n}\Big[\widetilde{f}(re^{2\pi ik/N})\Big], \tag{3.30}$$

which can be rewritten as

$$f(n) = \frac{r^n}{N}\mathcal{F}_{n\to k}\Big[\widetilde{f}(re^{2\pi ik/N})\Big]. \tag{3.31}$$

The $\mathcal{F}^{-1}$ and $\mathcal{F}$ can be computed with library calls such as Matlab's ifft or fft, respectively, both of which utilise the FFTW. This method has the added advantage of increasing N as required; as such, it can achieve an error E-16, i.e. machine accuracy, with double precision. The computational cost of a one-dimensional problem is $\mathcal{O}(N \log N)$ due to the FFT. Extensive work has gone into developing a pruned FFT approach, where only a subset of the resulting output is required, to account for this, the interested should consult Sorensen and Burrus [1993].

Further development can be done by using shifted points, where the points of the contour integral are shifted by a half period. The discrete Fourier transform can then be reformulated

$$f(n) = \frac{r^n}{N} \mathcal{F}_{n \to k} \Big[\widetilde{f}(re^{2(k-1)\pi i/N}) \Big] \quad k = 1, 2, \ldots, N. \tag{3.32}$$

3.7 Series acceleration techniques

To address the potential computational cost of using fast Fourier libraries, many of the methods discussed thus far can benefit from applying series acceleration techniques because an infinite series is present. Under the assumption that the series converges, achieving the desired level of accuracy may require a large number of terms, resulting in a high computational cost. This issue is particularly noticeable in practical environments when the method must be called repeatedly. This is true within the lens of option pricing, where multiple calls must be made to the inversion procedure for different parameter sets, ultimately resulting in a CPU bottleneck. With such an application, series acceleration methods can make an inversion method practically usable. A broad class of methods exist aiming to improve the acceleration that attempts to transform the partial sums of the given series into a new sequence with a faster rate of convergence. For the interested reader, consult Smith and Ford [1979] and Weniger [1989] for comprehensive reviews.

3.7.1 Euler Transform

One of the oldest acceleration techniques is that of Euler, which has been applied directly to Abate and Whitt's method by the authors and discussed in depth in O'Cinneide [1997]. A brief discussion is given here for completeness. Following the same framework seen previously, let $c_n(f)$ denote the Fourier series of f defined as

$$c_n(f) = \frac{1}{2\pi}\int_0^{2\pi} e^{int} f(t)dt. \tag{3.33}$$

We can then define $f(t)$ as the sum of a Fourier series at t

$$f(t) = \sum_{-\infty}^{\infty} c_n(f)e^{-int}. \tag{3.34}$$

Since the Fourier series has a natural symmetry, we can frame this discussion around the symmetric partial sums of the series

$$S_N f(t) = \sum_{n=-N}^{N} c_n(f)e^{-int}. \tag{3.35}$$

The Euler summation can be applied to convergent alternating complex series and has been used in several publications, e.g. Fusai et al. [2006, 2012, 2016] and Chen et al. [2014]. To proceed, for integers $l \geq 1$ and $m_E \geq 0$ we define the Euler(l, m_E) transform of this sequence to be the resulting sequence

$$b_n = 2^{-m_E} \sum_{k=0}^{m_E} \binom{m_E}{k} a_{n+kl}, \tag{3.36}$$

where a_n is the sequence to be accelerated. This sequence is equivalent to the binomial$(m_E, 1/2)$ average of every l-th term of the original sequence, beginning with the n-th. If the terms a_n are partial sums of a sequence, moving from a_n to b_n is referred to as Euler(l, m_E) summation. The most ubiquitous case of $l = 1$ is the same as the Euler transform, known as Euler summation. The Euler acceleration technique works under the following two conditions. Firstly, an integer $k \geq n$ exists such that the signs of S_N are alternating. Secondly,

$$\frac{1}{2} < \left| \frac{S_{n+1}}{S_n} \right| \leq 1 \quad \text{for } n \geq k. \tag{3.37}$$

The original use case for this acceleration is due to a function or its derivatives having discontinuities at zero. However, it is also applicable to a function with discontinuities in general. Thus, for the set of 2π periodic functions with bounded derivatives, the lack of smoothness is associated with the slow convergence of a Fourier series and, consequently, the z-transform. More generally, from the "integration by parts coefficient bound" described by Boyd [2001], if the function is smooth up to its $(k-2)^{\text{th}}$ and its k^{th} derivative is integrable, then the speed of convergence is $\mathcal{O}(1/\xi^k)$. Similarly, Boyd [2001] shows that functions where all derivatives are integrable have exponential convergence. The Euler (l, m_E) summation improves the rate of convergence of the Fourier series of functions at certain values $t = t_{i,l}$ of the series,

$$t_{i,l} = t(i,l) = \frac{(2i-1)\pi}{l}, \quad i = 1, 2, \ldots, l. \tag{3.38}$$

This is extendible to derivatives and functions with up to k-bounded derivatives. When Euler(l, m_E) summation is applied to Eq. (3.35) for the points seen in Eq. (3.38) the resulting sequence can be denoted as

$$S_n f(l, m_E; t_{il}) = 2^{-m_E} \sum_{k=0}^{m_E} S_{n+kl} f(t_{il}). \tag{3.39}$$

When using Euler summation applied to Abate and Whitt's method, we have to manually control two more parameters, m_E and $n_E = k$. Extensive analysis has been done here; see Zong [2020] to find the optimal values of these parameters. The usual choice is to select $n_E = 12$ and $m_E = 20$.

The discussion on the Euler summation thus far has been from a computational and practical aspect. The more traditional and generalised formulation of the Euler summation can be given as

$$S_N f(t) = S_{n-1} + \frac{\lambda^n}{1-\lambda} \sum_{k=0}^{m_E} \left(\frac{\lambda}{1-\lambda} \right)^k \Delta^k \left(\frac{a_n}{\lambda^n} \right), \tag{3.40}$$

where the difference operator Δ is applied to the superscripts. For $\lambda = -1$, i.e. in the case of an alternating series, this reduces to the classical Euler transform. The Euler transform is applied to alternating series, and the generalised form can be applied to any convergent series that can be written as a power series where $\lambda \in [-1, 1)$.

There are two general conditions to ensure the Euler transform works as expected. Firstly, an integer $k \geq 1$ exists such that the signs of $S_N f(t)$ are alternating for $n \geq t$. The second condition is that for $t \geq k$, $0.5 < |S_N f(t+1)/S_N f(t)| \leq 1$. Applying a scalar variable k to ensure that the partial sums of Cavers' method oscillate to meet the first condition is possible. This can be adjusted so that the Euler acceleration works. More information on how this is done can be seen in Chapter 4.

3.7.2 Padé Approximants

It is worth noting here a brief introduction to Padé approximants, which feature heavily in acceleration techniques. Padé approximants are rational functions constructed from a power series, i.e. Taylor or Maclaurin series, expansion of a function and introduced by Padé [1892]. This technique has been developed into a valuable tool for approximation theory, analytic continuation and the summation of divergent or slowly convergent series. Unlike truncating a series to a polynomial, Padé approximants use ratios of polynomials to approximate a given function. This approach often yields far superior approximations, primarily when singularities exist.

To begin the definition of a Padé approximant, first, consider a function $f(z)$ analytic about $z = 0$ with a known Maclaurin series expansion

$$f(z) = \sum_{n=0}^{\infty} c_n z^n, \tag{3.41}$$

where

$$c_n = \frac{f^{(n)}(0)}{n!}. \tag{3.42}$$

A Padé approximant $P_{[m/n]}$ of order $[m/n]$ is a rational function

$$P_{[m/n]}(f, z) = \frac{P_m(z)}{Q_n(z)}, \tag{3.43}$$

where $P_m(z)$ and $Q_n(z)$ are polynomials of degree at most m and n respectively. The coefficients of this polynomial are determined by matching the Taylor series of $f(z)$ up to order $m + n$

$$f(z) - \frac{P_m(z)}{Q_n(z)} = \mathcal{O}(z^{m+n+1}). \tag{3.44}$$

This ensures that the first $m + n + 1$ coefficients in the series expansion of $f(z)$ and $P_m(z)/Q_n(z)$ coincide. In practice, one solves a linear system derived from the conditions

$$Q_n(z) \left(\sum_{k=0}^{m+n} c_k z^k \right) - P_m(z) = \mathcal{O}(z^{m+n+1}). \tag{3.45}$$

The uniqueness of the Padé approximant, assuming $c_0 \neq 0$ and a non-degenerate system, is specified by the theory of linear algebra applied to the corresponding matrices from the series coefficients $\{c_n\}$, see Baker and Graves-Morris [1996].

Under generic conditions, the Padé approximant $[m/n]$ exists and is unique. However, degeneracy can occur if the linear systems fail to have a unique solution. Such degeneracies often signal additional structure in the function $f(z)$, such as hidden algebraic relations among the coefficients. A known result here is that if the function $f(z)$ is rational of the form

$$f(z) = \frac{U(z)}{V(z)}, \tag{3.46}$$

for polynomials U and V and with $deg(U) = \alpha$ and $deg(V) = \beta$, then $[m/n]$ for sufficiently large m, n where $m > \alpha$ and $n > \beta$, will reproduce $f(z)$ exactly. This perfect approximation underscores the power of Padé approximants. The interested reader should consult C. Brezinski and Redivo-Zaglia [1991] or Baker and Graves-Morris [1996] for more information.

A remarkable feature of Padé approximants is the ability to achieve analytic continuation. Even if the original power series has a finite radius of convergence due to some singularity present at some $|z| = R$, the Padé approximant may provide meaningful values beyond values $|z| > R$. Thus, the approximant can avoid singularities in the complex plane if they are not poles.

Furthermore, this approach enjoys pleasant convergence properties in many classical scenarios. For instance, if $f(z)$ is meromorphic, i.e. it is analytic except for isolated poles, within a specific region of the complex plane, a sequence of Padé approximants can reconstruct the location and size of these poles from the given coefficients $\{c_n\}$. Indeed, for analytic and meromorphic functions, the theory of Padé approximants can yield exponential convergence rates; see Baker and Graves-Morris [1996] and Stahl [1997].

3.7.3 Wynn's Rho and Epsilon Algorithms

Two popular methods were presented by Wynn [1956] in his seminal paper, known as the epsilon and rho algorithms. The epsilon algorithm is a nonlinear sequence transformation designed to accelerate the convergence of a given sequence $\{S_n\}$, which generally represents the partial sums of an infinite sequence. The key concept behind the epsilon algorithm is to construct a triangular array of transformed values that lead to the approximation converging much faster than the original sequence, provided the original sequence satisfies certain analyticity conditions.

The algorithm gained popularity because it is straightforward and numerically stable for many problems. Crucially, it does not matter whether the sequence is alternating or non-alternating, and it is often applied in numerical quadratures and solutions to integral equations. The foundation of the method is simple to elucidate; consider a sequence $\{S_n\}$, with partial sums S_n where

$$S_n = \sum_{k=0}^{n} a_k. \tag{3.47}$$

The algorithm produces a two-dimensional array $\{\epsilon_k^{(n)}\}$ defined by the recurrence relation for $k \geq 1$,

$$\epsilon_k^{(n)} = \epsilon_{k-2}^{(n+1)} + \frac{1}{\epsilon_{k-1}^{(n+1)} - \epsilon_{k-1}^{(n)}}, \tag{3.48}$$

where

$$\epsilon_{-1}^{(n)} = 0, \quad \epsilon_0^{(n)} = S_n, \quad \text{for all } n \geq 0.$$

Provided that the denominator is non-zero. This recursion is typically written and computed in a tabular form. This table is often called the epsilon table, which is constructed as follows. The first column, where $k = 0$ is just the original sequence $S_0, S_1, S_2, \ldots$. In the second column, $k = 1$, these values produce improved approximations. Subsequent columns can produce further accelerations.

The key observation is that only the entries with even k often yield the best approximations to the $\{S_n\}$ limit. The even-indexed elements, i.e. $\epsilon_{2r}^{(0)}$ for $r = 0, 1, 2, \ldots$, frequently converge to the limit of the now accelerated sequence faster than the original.

The epsilon algorithm can be interpreted in several ways. A common interpretation mentioned by Wynn is that the method relates to continued fractions and Padé approximants of power series C. Brezinski and Redivo-Zaglia [1991], Weniger [1989]. Thus, the algorithm rapidly converges to the limit when the partial sums S_n correspond to an analytic function within a specific domain. The epsilon algorithm is beneficial when dealing with sequences that stem from expanded series involving factorial terms, binomial coefficients or generalised hyper-geometric series Smith and Ford [1979].

In practice, the user must be cautious about numerical stability. As the denominator in Eq. 3.48 indicates, division by small numbers can cause numerical issues. Further, the algorithm's efficiency can be improved by implementation where one would store only a few columns of the epsilon table at a time and overwrite the previous values to avoid extensive memory usage.

Within the same work, Wynn also produced a different acceleration based on a modification of the algorithm. The so-called rho algorithm is explicitly designed to address convergence by including an additional factor, k, in the recurrence relation. This minor adjustment often produces faster convergence than the epsilon algorithm and can be considered an accelerated version.

The rho algorithm constructs a similar two-dimensional array $\rho_k^{(n)}$, starting from the same sequence as previously $\{S_n\}$ with partial sums S_n, for $k \geq 1$

$$\rho_k^{(n)} = \rho_{k-2}^{(n+1)} + \frac{k}{\rho_{k-1}^{(n+1)} - \rho_{k-1}^{(n)}}, \tag{3.49}$$

where

$$\rho_{-1}^{(n)} = 0 \quad \rho_0^{(n)} = S_n \quad \text{for all } n \geq 0.$$

The factor k in the numerator generally improves the convergence rate. As with the epsilon algorithm, the best approximations are typically found at even indices of k.

This upgrade algorithm is a refinement that leverages information about how the transforms evolve with each step k. When applied to challenging sequences, the convergence behaviour is often superior to that of the epsilon algorithm. As before, the rho algorithm is connected intrinsically to Padé approximants. Typical usage for the algorithm is as an improvement upon the epsilon algorithm, where the secondary algorithm converges but not as fast as desired.

A distinct advantage of the epsilon and rho algorithms over Euler's acceleration approach is the lack of explicit parameters that require tuning. The criteria for applying this method are based solely on the convergence of successive approximations, which are typically used to slowly convergent or asymptotic expansions of special functions such as Bessel or hypergeometric functions encountered in physics.

3.7.4 The Shanks Transform

The Shanks transform, Shanks [1955], is derived from an earlier work presented by Schmidt [1941] based on nonlinear series acceleration methods. The Shanks transform is designed to improve the convergence rate of a sequence by targeting the partial sums typically of an infinite series. Unlike linear acceleration methods, such as the Euler acceleration, the Shanks transform uses rational approximations inferred from successive iterations, thus often achieving dramatic improvements in convergence and stability of the series.

Fundamentally, the Shanks transform approximates the underlying function or generating mechanism by constructing a ratio of two polynomials that is rational and whose values at specific points match the known partial sums. Similarly to Wynn's algorithms, this approach can be related to Padé approximants and continued fractions C. Brezinski and Redivo-Zaglia [1991], Sidi [2003].

As before, let $\{S_n\}$ be a sequence of partial sums defined as before in Eq. (3.47). Furthermore, suppose $\{S_n\}$ converges slowly to the limit L. The Shanks transform aims to produce a new sequence $\{\widehat{S}_n\}$ that converges to L more rapidly. This approach is commonly expressed as a discrete analogue of moment or determinant forms. A popular representation of the Shanks transform is

$$\mathcal{S}(S_n) = \frac{S_{n+1}S_{n-1} - (S_n)^2}{S_{n+1} - 2S_n + S_{n-1}}, \tag{3.50}$$

for $n \geq 1$ and under the assumption that the denominator is non-zero. This particular formulation represents a second-order nonlinear transform that attempts to remove the dominant part of the asymptotic error term. In practice, repeated application of the Shanks transform and nested Shanks transforms can be used analogously to iterative Euler summation or repeated Wynn's transforms.

The Shanks transform can be applied to three consecutive terms of any sequence $\{x_n\}$. Suppose $\{x_n\}$ is the sequence of partial sums or any other slowly converging sequence. In that case, the transform can produce a new sequence that converges faster under suitable conditions on the generating function and the nature of the asymptotic expansions, Weniger [1989].

The connection to Padé approximants allows the Shanks transform to use the input sequence's behaviour to infer a rational approximation to its generating function, giving rise to powerful error-reduction capabilities.

Suppose now that the original sequence $\{S_n\}$ can be associated with a function $F(z)$, which is analytic in some domain. In that case, the Shanks transform can be interpreted as a step in constructing a Padé approximant to $F(z)$. Furthermore, if $F(z)$ is analytic and has no singularities in the complex plane, the transformed sequence $\{\widehat{S}_n\}$ may display geometric convergence to the actual limit, C. Brezinski and Redivo-Zaglia [1991].

The general convergence properties, however, can depend heavily on the analytic properties of the underlying function. For instance, if the original sequence $\{S_n\}$ behaves asymptotically

$$S_n = c\lambda^n + \mathcal{O}(\lambda^{2n}), \quad \text{with } |\lambda| < 1, \tag{3.51}$$

the Shanks transform can often remove the dominant λ^n term resulting in a new sequence converging with $\mathcal{O}(\lambda^{2n})$.

Additionally, if $F(z)$ is the generating function for $\{S_n\}$ or $\{a_k\}$ and $F(z)$ can be continued analytically beyond the unit circle and has isolated singularities in the complex plane, the Shanks transform can detect singularities and adjusts accordingly with successive iterations. This self-correcting process makes the convergence substantially better.

As with many transforms, if the sequence $\{S_n\}$ is irregular, highly oscillatory or dominated by multiple competing error terms, the Shanks transform may require multiple applications of combined use with another transform to achieve stable results. Wynn's epsilon or rho algorithm is the standard choice in this case. Alternatively, preconditioning the sequence using Euler's summation to induce alternating signs or scaling transforms can result in more stable results, see Sidi [2003], Smith and Ford [1979].

Unlike linear transforms, the Shanks transform can suffer from numerical instability if the denominator of Eq. (3.50) is very small or if the sequence is highly noisy. Small denominators can be mitigated by scaling, smoothing or applying other transforms at carefully chosen indices, which can minimise these issues. Further, adaptive strategies that monitor the magnitude of denominators and the stability of successive approximations are recommended. With this in mind, more modern approaches combine the Shanks transform with other acceleration methods, such as Euler and Wynn's accelerations.

3.7.5 Richardson Extrapolation

It is worth a small introduction to Richardson extrapolation, which, while not directly an acceleration technique, is fundamental to acceleration approaches to follow. Initially presented by Richardson [1910], this technique is designed to improve the accuracy of numerical approximations by exploiting known asymptotic error expansions originally conceived for numerical calculus. The power of this approach lies in its simplicity and generality; if one knows or can guess how the error term depends on a discretisation parameter, for instance, a step size h or iteration count, then the extrapolation can systematically eliminate leading-order error terms and thus accelerate the convergence.

Mathematically, consider a sequence of approximations $\{A(h)\}$ to a limit L, where h is a discretisation parameter that tends to 0. Typically, $A(h)$ might represent an approximation of a derivative or a solution to a differential equation using a step size h. Further, suppose that the approximation $A(h)$ admits an asymptotic expansion

$$A(h) = L + \alpha h^p + \alpha_2 h^{p_2} + \dots, \tag{3.52}$$

where $p > 0$ is the leading order of the error, α is the leading error coefficient, and higher order terms are of a smaller magnitude as $h \to 0$. Richardson's extrapolation constructs a new approximation that cancels the leading-order error term.

Consider two approximations, $A(h)$ and $A(qh)$, for a chosen scaling factor $q > 1$.

Then

$$A(h) = L + \alpha h^p + \mathcal{O}(h^{p_2}), \tag{3.53}$$

$$A(qh) = L + \alpha (qh)^p + \mathcal{O}((qh)^{p_2}). \tag{3.54}$$

The idea is that by forming a suitable linear combination, one can eliminate αh^p. Thus,

$$A_{R_e} = \frac{q^p A(h) - A(qh)}{q^p - 1}, \tag{3.55}$$

yields a new approximation A_{R_e} closer to L, typically with the leading error term removed, resulting in a higher order of accuracy of terms of h.

The essence of this approach is the assumption of an asymptotic error expansion. If the behaviour of the error term is known or can be well approximated by a power law h^p, the extrapolation reduces the error order from $\mathcal{O}(h^p)$ to $\mathcal{O}(h^{p_2})$. For many standard problems in calculus, the leading error terms are often polynomial in h. By changing the underlying approximation scheme, Richardson extrapolation provides a straightforward and powerful method to enhance a low-order method into a higher-order one.

Two methods under discussion, the Levin and Sidi transforms, rely on knowledge or assumptions about the error structure. While Richardson extrapolation focuses on eliminating a single dominant error term, the following transforms can target more complicated error models.

3.7.6 Levin-Type Sequence Transforms

Levin-type sequence transformations are a class of nonlinear series acceleration methods inspired mainly by problems in integral convergence found in physics. First presented in Levin [1973], Levin acceleration methods are particularly effective for sequences asymptotically similar to a known reference function or for series with factorial, binomial or geometric-like terms. Unlike simpler approaches, this transform incorporates additional information about the asymptotic behaviour of the sequence terms, allowing for the handling of a broader class of problems and often yielding faster convergence. Despite relative obscurity at publication, this class of accelerations has gained attention due to its inherent robustness and flexibility.

As before, consider a sequence $\{S_n\}$, where S_n represents the partial sums of the sequence, i.e.

$$S_n = \sum_{k=0}^{n} a_k.$$

The Levin transform relies on an associated sequence $\{\omega_n\}$ that encapsulates known or assumed asymptotic behaviour of the sequence $\{a_n\}$ of their partial sums.

The most common form in which this transform is cited is the so-called Levin u-transform. Suppose again there is a sequence $\{a_n\}$ and a corresponding sequence $\{\lambda_n\}$ which approximates the behaviour of a_n as $n \to \infty$. Typically, λ_n is chosen such that $a_n\lambda_n$ approaches a constant or has a simple limiting form. For the sequence of partial sums S_n, the Levin u-transform is defined as follows. Begin with the simple sequence

$$u_n = \frac{a_n}{\lambda_n}. \tag{3.56}$$

This sequence forms the partial sums

$$U_n = \sum_{k=0}^{n} u_k, \tag{3.57}$$

the Levin transform of order m can be expressed as

$$S_n = \frac{\sum_{k=0}^{m} (-1)^k \binom{m}{k} U_{n+k}}{\sum_{k=0}^{m} (-1)^k \binom{m}{k} \lambda_{n+k}^{-1}}, \tag{3.58}$$

for appropriate sequence λ_n.

The power of Levin's transform stems from their explicit usage of the sequences of asymptotic information. If the asymptotic form of a_n is known or can be reasonably approximated, Levin-type transforms can drastically reduce dominant error terms which may be present. Under suitable analyticity conditions and correct approximation of asymptotic behaviour, Levin transforms achieve at least geometric convergence, Levin [1973], Sidi [1979]. The convergence can become exponential when combined with appropriate reference sequences λ_n.

As one may have diagnosed, the fundamental practical problem of implementation is correctly identifying asymptotic behaviour. This may not be trivial to determine. Inaccurate estimates of asymptomatic parameters can lead to numerical instability or slow convergence. Modern applications of this approach often try multiple transforms with variations of λ_n to determine which yields the most rapid convergence.

3.7.7 Sidi Accelerations

First presented in 1979 and extensively detailed in the seminal work Sidi [2003], this family of transforms encompass a family of acceleration techniques designed to handle various convergence issues. Unlike the previous accelerations that rely on specific patterns, for instance, an alternating series or asymptotic forms, Sidi's approach incorporates more flexible modelling of the error terms. The foundational idea is similar to that of Richardson Extrapolation. Still, here, the key idea is to approximate the underlying limit or function by exploiting the known or assumed behaviour of the error term.

Begin with the sequence $\{S(h)\}$ parametrised by a step size $h \to 0$. A classic example is $S(h) = S_n$ where $h = 1/n$. This is analogous to numerical integration, where h is the grid size. Thus, considering a sequence as before $\{S_n\}$ that converges to some limit L. If one can express the error $S_n - L$ in a form involving a known function, for instance

$$S_n - L \approx \sum_{k=1}^{N} c_k \phi_k(n), \tag{3.59}$$

where the sequence $\{\phi_k(n)\}$ is known or a reasonable estimation of functions of n that characterise the asymptotic behaviour of the error, then it is possible to construct transforms that eliminate these error terms successively. By choosing a suitable ϕ_k, this framework can simplify the problem of slow convergence. By doing so, it is possible to achieve geometric and exponential convergence.

A variant of this generalised transform is the so-called D-transforms, Sidi [2003], a class of extrapolation methods that generalise and unify existing techniques. Again, let $\{S_n\}$ be the sequence to be accelerated and $\{w_n\}$ a sequence of weights capturing the main asymptotic behaviour. Then one constructs a table of approximants $D_{m,n}$ by linear or nonlinear recurrences that combine the original sequence values and their finite differences, weighted by $\{w_n\}$. Under careful selection of the weights, the D-transform can eliminate the leading terms of the error expansion.

Furthermore, numerous variants tailored to different classes of problems have been developed. For instance, transforms designed for oscillatory integrals often model the error terms using trigonometric or oscillatory components. Similarly, some target factorial-type growth or geometric patterns.

This approach is firmly rooted in approximation theory and asymptotic analysis. If it is assumed that the error in the original sequence S_n has a known asymptotic expansion in terms of functions $\phi_k(n)$. The form of functions ϕ could take multiple forms depending on the application. To construct the transformed sequence, a new sequence $\{\tilde{S}_n\}$ is produced via a linear or nonlinear transform that eliminates the leading-order terms in the error expansion. Mathematically, this is often resolved by solving a linear equation system derived from conditions where the transformed sequence dampens specific error terms up to a given order. When the chosen modelled functions $\omega_k(n)$ match the actual asymptotic behaviour, the transforms can produce significant improvements in convergence. The author notes that if done correctly, a sequence that converges like n^{-p} into one that converges like n^{-q} where $q > p$, or even exponential convergence in some cases. For more detailed analysis, consult Sidi [2003].

The power of this approach is that the transforms can be seen as a framework that incorporates accelerations already discussed and utilises modern transformations as special or limiting cases. Primarily, Levin-type accelerations rely on known asymptotic information. Sidi's approach generalises this idea, allowing for a broader range of error models and making them applicable to a wider set of problems. Thus, Levin transforms are often regarded as a subset of Sidi's framework, where the error modelling is done via a single function.

Similarly to Levin's transformation approach, the implementation requires some insight into the error's asymptotic form. If this knowledge is unavailable, an iterative procedure will need to be followed, testing various models until the correct form is found, yielding an accelerated convergence.

3.7.8 Salzer Summation

The Salzer summation is a classical series acceleration technique introduced initially to improve the convergence of infinite series, particularly those arising in numerical Laplace transform inversion, see Wimp [1981]. The method re-weights partial sums of an underlying series to mitigate the higher-order terms. These tend to be the terms contributing the bulk of the oscillations and slow convergence. By applying weights, the approach can reduce the number of terms needed to achieve a desired accuracy, particularly if the series has an alternating or slowly convergent structure. This method has been used extensively in the case of the numerical Laplace inversion, see Valkó and Abate [2004], where accelerations have been made to the traditional Gaver-Stehfest method by reorganising the sums involved in extracting inverse transforms.

Although Salzer's summation has been most commonly applied to the Laplace transform domain, its principle of re-weighting partial sums can also be adapted to the z-transform framework. In discrete-time or discrete-state settings, the numerical inversion of a z-transform typically involves summing a series expansion of the inverse transform, which, like its Laplace counterpart, can suffer from slow or oscillatory convergence. By implementing a Salzer-style summation, one can introduce selective weighting schemes to stabilise the partial sums and possibly reduce the number of terms needed.

3.8 Results

For benchmarking purposes, it is convenient to use analytical z-pairs, i.e. pairs of functions with well-defined $\widetilde{f}(z)$ and $f(n)$. Such pairs are simple to compute and provide an excellent benchmark to measure accuracy and computational performance. The number of sampling or option monitoring points is set to 20 for readability, but is arbitrary. Table 1 shows the numerical error and CPU time results for each method over a series of analytical pairs.

Function	$f(n)$	$F(z)$
Heaviside step	1	$\dfrac{z}{z-1}$
Polynomial	n	$\dfrac{z}{(z-1)^2}$
Decaying exp	e^{-an}	$\dfrac{1}{1-e^{a\Delta n}z^{-1}}$
Sinusoidal	$\sin(\omega n)$	$\dfrac{z^{-1}\,\sin(\omega\Delta n)}{1-2z^{-1}\cos(\omega\Delta n)+z^{-2}}$
Geometric Distribution	$p\,(1-p)^{\,n}$	$\sum_{n=0}^{\infty} p\,(1-p)^{\,n} z^n$
Poisson Distribution	$\dfrac{e^{-\lambda}\,\lambda^n}{n!}$	$\sum_{n=0}^{\infty}\dfrac{e^{-\lambda}\,\lambda^n}{n!}\,z^n$
Negative Binomial Distribution	$\binom{n+r-1}{n}\,(1-p)^{\,r}\,p^n$	$\sum_{n=0}^{\infty}\binom{n+r-1}{n}\,(1-p)^{\,r}\,p^n\,z^n$
Catalan	$\dfrac{1}{n+1}\binom{2n}{n}$	$\sum_{n=0}^{\infty} C_n\,z^n$

Table 3.1: List of transform pairs used in the numerical tests to follow. A wide range has been chosen to benchmark the methods against different features.

Table 3.8 details the evaluated methods' average, maximum errors and CPU time. The CPU time was assessed using an Intel(R) Core i7-9750H 8 core 2.60GHz processor and 32 GB of RAM averaged over four successive runs to avoid interference from background processes. The tests were performed with MATLAB using Windows 10. As we are primarily interested in an application such as option pricing, any error value greater than 10^{-2} is omitted as unusable. The analytical pairs used are evaluated over a time grid $t = 0, \ldots, 1 = T$, where the number of discretisation points $n = 20$ for Table 2 and $n = 100$ for Table 3. An additional column is provided for any constants in the method. As a benchmarking choice, no accelerated methods were tested for $n = 20$ since series acceleration is largely redundant at this number of intervals.

AW denotes the accelerated version of Abate and Whitt's approach for reference. C refers to the Cavers method. Cavers' method with Euler, Shanks, and epsilon accelerations is given by CEuler, CShanks, and CEpsilon, respectively. Orthogonal decomposition using $r = 5/6$ is denoted with OD. Finally, the matrix exponential approach is denoted by ME. Of note is the generally poorer performance when significantly increasing the number of discretisation points.

As the results show, the Cavers' method can achieve a machine-accurate solution of E-16, but at the cost of computational time. This CPU deficiency can be improved with an acceleration that performs well, but at the expense of some numerical accuracy. Another issue with Cavers' approach is the selection of the parameter r. We have set r in the $1.01 \leq r \leq 1.6$ range for all experiments. Horváth et al. [2020] also notes the difficulty in handling this parameter and witnesses the loss in precision if this parameter is set incorrectly. With an acceleration applied, the method is comparable to Abate and Whitt's methods. The orthogonal decomposition and matrix exponential approaches are the least attractive methods due to the high error floor and inconsistent nature.

As expected, all methods achieved their lowest errors for the polynomial, probability-generating function, or the decaying exponential. Many methods had a dramatic and often volatile increase in maximum error when handling the Heaviside step function, because of the jump discontinuity in the step function. This is particularly interesting as a function that could perform in a superior fashion for functions with discontinuities has its advantages.

To broaden the study of series acceleration beyond simple one-dimensional z-pairs, we benchmark the techniques to a two-dimensional binomial distribution and the generating function for Catalan numbers. The former case is interesting with our subconscious task of option pricing since the functions required for inversion are two-dimensional, i.e. $F(t, z)$, where t is the time domain. An extensive array for t was incorporated into the function to mirror this requirement, resulting in a large two-dimensional array. The Catalan generator is known for its rapidly growing coefficients, making series convergence more difficult. These two examples test whether the numerical inversion and acceleration methods can handle more intricate combinatorial forms and maintain reliable accuracy in higher complexity.

We apply series acceleration exclusively to the Abate and Whitt and Cavers' methods because, in unaccelerated form, these two approaches already demonstrated substantial flexibility and precision on simpler pairs and are under scrutiny within option pricing. In addition to the numerical acceleration techniques used previously, Cavers' method with Levin and Sidi transforms are also tested, marked CLevin and CSidi, respectively.

The general trend of Table 3.8 exhibits the same accuracy-versus-cost trade-off observed earlier. A clear trade-off between numerical accuracy and computational efficiency is present across the methods tested. The two-dimensional binomial case, in particular, slowed down the Cavers' method despite its continued excellent accuracy. The Levin- and Sidi-based accelerations perform on par with Euler acceleration. Interestingly, the epsilon and Shanks transform experienced a dip in performance, which can be attributed to incomplete control over the number of iterations the accelerations perform. In particular, Shanks' transform requires specifying a fixed number of iterations, and each iteration directly processes the partial sums. This results in a direct trade-off between numerical performance and the number of acceleration iterations the user wishes to provide.

In contrast, the Sidi and Levin transform circumvents this iteration-based overhead by applying a single-pass reorganisation of the partial sums using difference operators. Specifically, the user specifies a transform order $W(m)$, which determines how many successive differences (and corresponding weights) are applied. This systematic approach removes leading error terms from the partial sums more flexibly, often resulting in more stable and efficient convergence than strictly iterative methods.

Function	Method	Average error	CPU time
Heaviside step	AW	3E-10	0.00044
	C	3E-16	0.00218
	CEuler	5E-15	0.00033
	OD	0	0.00842
	ME	2E-04	0.01913
Polynomial	AW	2E-11	0.00034
	C	2E-16	0.00351
	CEuler	2E-16	0.00064
	OD	3E-06	0.00330
	ME	1E-05	0.01354
Decaying exp	AW	3E-10	0.00064
	C	4E-16	0.00648
	CEuler	3E-15	0.00196
	OD	1E-06	0.00658
	ME	2E-05	0.03749
Sinusoidal	AW	6E-10	0.00110
	C	8E-16	0.00891
	CEuler	5E-16	0.00076
	OD	2E-06	0.00545
	ME	4E-05	0.06541

Table 3.2: Error values for numerical tests on single variable z-pairs with number of discretisation points $n = 20$. Note the expected excellent performance of AW and C.

Function	Method	Average error	CPU time (sec)
Heaviside step	AW	2E-09	0.00125
	C	3E-15	0.02156
	CEuler	6E-15	0.00248
	CShanks	4E-11	0.00141
	CEpsilon	2E-10	0.00111
	CSidi	5E-10	0.0057
	CLevin	4E-09	0.0043
	OD	3E-09	0.00842
	ME	2E-06	0.01913
Polynomial	AW	1E-08	0.00188
	C	3E-15	0.005681
	CEuler	3E-15	0.00112
	CShanks	4E-12	0.00255
	CEpsilon	8E-12	0.00174
	CSidi	2E-11	0.0276
	CLevin	4E-12	0.0571
	OD	3E-06	0.00551
	ME	2E-02	0.01354
Decaying exp	AW	3E-12	0.00190
	C	8E-16	0.00871
	CEuler	3E-15	0.00301
	CShanks	3E-10	0.00141
	CEpsilon	3E-10	0.00159
	CSidi	2E-10	0.0357
	CLevin	4E-10	0.0600
	OD	1E-06	0.00663
	ME	6E-02	0.02235
Sinusoidal	AW	3E-10	0.00203
	C	3E-16	0.00993
	CEuler	6E-15	0.00105
	CShanks	3E-10	0.00154
	CEpsilon	2E-13	0.00215
	CSidi	3E-11	0.0424
	CLevin	1E-11	0.0488
	OD	1E-05	0.00481
	ME	3E-04	0.04513

Table 3.3: Error values for numerical tests of single variable z-pairs with number of discretisation points $n = 100$. This increase of discrete points is designed to test the method's stability. Again, AW and C perform well with reasonable performance from the accelerated versions of C.

Function	Method	Average error	CPU time (sec)
Geometric	AW	4E-11	0.0048
	C	6E-16	0.0200
	CEuler	1E-13	0.0045
	CShanks	3E-10	0.0036
	CEpsilon	2E-10	0.0051
	CSidi	6E-14	0.0424
	CLevin	8E-14	0.0488
Poisson	AW	1E-10	0.0013
	C	8E-15	0.0026
	CEuler	3E-14	0.0013
	CShanks	4E-10	0.0064
	CEpsilon	8E-11	0.0072
	CSidi	1E-14	0.0024
	CLevin	1E-14	0.0068
Neg Binomial	AW	7E-11	0.0005
	C	8E-16	0.0097
	CEuler	8E-15	0.0010
	CShanks	3E-10	0.0046
	CEpsilon	3E-10	0.0033
	CSidi	3E-15	0.0034
	CLevin	4E-15	0.0047

Table 3.4: Error values for numerical tests of probability generating functions with number of discretisation points $n = 120$. The methods, M and OD, have been omitted from this part of the analysis since their performance was not comparable to the contour integration approaches.

Function	Method	Average error	CPU time (sec)
Catalan	AW	3E-10	0.0231
	AWEuler	1E-07	0.0038
	C	1E-15	0.2322
	CEuler	1E-13	0.0097
	CShanks	2E-07	0.0612
	CEpsilon	3E-06	0.0455
	CSidi	2E-12	0.0776
	CLevin	4E-11	0.0264
Multi-Dim Binomial	AW	1E-10	0.0567
	AWEuler	5E-09	0.0064
	C	7E-16	0.4320
	CEuler	2E-13	0.0105
	CShanks	2E-08	0.0789
	CEpsilon	4E-09	0.0648
	CSidi	8E-12	0.0940
	CLevin	6E-11	0.0567

Table 3.5: Error values for numerical tests of series acceleration with slowly converging or multi-dimensional z-pairs. With the number of discretisation points $n = 100$.

3.9 Conclusion

In this chapter, we reviewed and implemented a selection of numerical inverse z-transform techniques to survey and benchmark proposed techniques from various applications within the more general literature. Additionally, we reviewed and implemented series acceleration techniques for the computationally intensive Cavers method to achieve machine accuracy and have a computational cost independent of the number of monitoring points. This was achieved with success using the modified Euler technique and, to a lesser extent, employing the Shanks, Epsilon, Levin and Ford-Sidi accelerations, all of which drastically improve the CPU times with varying costs to the accuracy.

Chapter 4

Option Pricing

In this chapter, we present an improvement to the Fourier-z Sptizer approach to option pricing presented by Fusai et al. [2016] and improved by Phelan et al. [2019] with a specific focus on the double barrier and α-quantile options. Since the method uses the Spitzer identities and z-transform, this method will be abbreviated by ZS. For benchmarking and analysis reasons, the technique presented by Feng and Linetsky [2008] is employed to gauge the performance of the ZS method. This secondary approach uses the inverse Hilbert transform as a fundamental building block. Thus, this method will be abbreviated by IH. The procedure for both of these methods is given in Section 2.3. This method iterates on the N monitoring dates, and thus, its time grows linearly with N; we refer to it as iterative Hilbert (IH). The convergence of the error with respect to the grid size is exponential until machine accuracy, i.e. $2^{-52} \approx$E-16 with IEEE 754 double precision representation of floating point numbers, thanks to a sinc function expansion of the Hilbert transform.

4.1 Analysis of inverse z-transform

A noted shortcoming of the ZS method is its inability to achieve machine accuracy, restricted to an error floor of E-10. Contrastingly, the IH method produces machine-accurate solutions in all scenarios except for the variance gamma process. Whilst machine-accurate solutions are not of concern to practitioners, they are still an unsatisfactory limitation of the ZS method. The reason for this limitation is in the computation of the inverse z-transform, which is required to recover the transformed probabilities produced by the Spitzer identities. Candidates for the z-transform are not as plentiful as for its continuous cousin, the Laplace transform. The inverse z-transform was first presented by Abate and Whitt (AW) (1992) and has proved to be a popular choice in financial applications. The AW method has the limitation of not fully controlling the parameter governing the error, so its accuracy can only be varied by a limited amount before losing numerical control. This chapter presents a modified version of the ZS method, which eliminates the increased error floor and achieves machine-accurate solutions for barrier and alpha quantile options at the expense of computational cost in the case of barrier options.

The numerical inverse z-transform employed in the ZS method was devised by Abate and Whitt (1992) and approximates the inverse z-transform by

$$f(n) \approx \frac{1}{2n\rho^n}\left[\widetilde{f}(\rho) + 2\sum_{j=1}^{n-1}(-1)^j \mathrm{Re}\widetilde{f}\left(\rho e^{i\pi j/n}\right) + (-1)^n \widetilde{f}(-\rho)\right]. \tag{4.1}$$

Following the definition of the inverse z-transform given in Sec.(3),

$$\widetilde{f}(\xi) = \mathcal{F}_{t\to\xi}[f(t)] = \int_{-\infty}^{\infty} f(t)\, e^{-2\pi i\xi t}\, dt \tag{4.2}$$

$$f(t) = \mathcal{F}^{-1}_{\xi\to t}[f(\xi)] = \int_{-\infty}^{+\infty} \widetilde{f}(\xi) e^{2\pi i\xi t} d\xi. \tag{4.3}$$

This definition coincides with that of the FFTW library, see Frigo and Johnson [1998], which is included in Matlab and is Python-wrapped in pyFFTW.

Dividing the integration range $[0,1]$ of ξ into J equal intervals $\Delta\xi = 1/J$ and introducing the grid $\xi_j = j\Delta\xi = j/J,\ j = 0,\ldots,J-1$, the Fourier transform in

Eq. (4.3) is approximated by a discrete Fourier transform,

$$f(n) = \mathcal{Z}^{-1}_{z\to n}\big[\widetilde{f}(z)\big] \approx \frac{r^n}{J}\sum_{j=0}^{J-1} \widetilde{f}\big(re^{i2\pi j/J}\big)e^{i2\pi nj/J}, \quad n = 0, \dots, N, \quad J \geq 2N. \tag{4.4}$$

In the case $n = N$, Eq. (4.4) becomes

$$f(N) = \mathcal{Z}^{-1}_{z\to N}\big[\widetilde{f}(z)\big] \approx \frac{r^N}{2N}\sum_{j=0}^{2N-1} \widetilde{f}\big(re^{i\pi j/N}\big)e^{i\pi j} \tag{4.5}$$

$$= \frac{r^N}{N}\left[\frac{1}{2}\widetilde{f}(r) + \sum_{j=1}^{N-1}(-1)^j \mathrm{Re}\,\widetilde{f}\Big(\frac{1}{re^{i\pi j/N}}\Big) + \frac{(-1)^N}{2}\widetilde{f}(-r)\right]. \tag{4.6}$$

There is a natural constraint, $J \geq 2N$, arising from the Nyquist-Shannon sampling theorem, which dictates the setting of J as stated in Press et al. [2007], for instance, Abate and Whitt [1992a,b] set $J = 2N$ since it is "convenient". This is evident in Figure 4.3, where the error jumps when the sampling theorem is broken at $n = J/2 + 1$. Two natural symmetries are present in this particular setting: symmetry in the integrand of the DFT and symmetry in the conjugation of the complex circle upon which z is defined. For each point $(-1)^j\mathrm{Re}[\widehat{f}(re^{i\pi j/J})]$ on the upper half-circle, there exists a conjugate point on the lower half circle.

Abate and Whitt [1992a,b] proved that the absolute error $|\varepsilon|$ of the inverse z-transform given by Eq. (4.4) has the bound

$$|\varepsilon| < \frac{1}{r^J - 1} \approx \frac{1}{r^J} \tag{4.7}$$

(the approximation holds when r^J is large), so setting $r = 10^{\gamma/J}$ should yield an accuracy of $10^{-\gamma}$. However, Abate and Whitt [1992a, Remark 5.8] pointed out that if the machine accuracy is $\varepsilon_{\mathrm{m}} = 10^{-\gamma_{\mathrm{m}}}$, rounding errors cause this bound to break down for about $\gamma > \frac{2}{3}\gamma_{\mathrm{m}}$; see also Abate and Whitt [1992b, Remark 6]. With IEEE 754 double precision representation of floating point numbers, $\varepsilon_{\mathrm{m}} = 2^{-52} \approx 10^{-15.65}$, and thus it makes little sense to set $\gamma > 10.5$, which yields the IZT up to about 10 or 11 significant digits. Abate and Whitt spent only a few lines of text about the rounding error and concluded that one should use at least $3\gamma/2$ digits of precision to compute the IZT with γ significant digits.

On the contrary, we aim to exploit as much as possible the available digits of precision, which can be single, double, quadruple, or half, to compute the IZT with an error as close as possible to machine accuracy. Using the function pair

$$f(n) = e^{-\psi n \Delta t}, \tag{4.8}$$

$$\widetilde{f}(z) = \frac{z}{z - e^{\psi \Delta t}} \tag{4.9}$$

where $\psi \in \mathbb{C}$, a numerical study of the error of the IZT confirms that the error bound of r^{-J} given by Eq. (4.7) holds as a function of $\gamma = J \log_{10} r$ until it is swamped by the rounding error that grows proportionally to γ; see Figure 4.1. This can also be seen when applying the full pricing method to a single or double barrier option; see Figure 4.2. The turning point depends on the function parameter ψ, on the monitoring interval ΔT and the value of N, but for $J = 2N$ indeed lies around or slightly above $\frac{2}{3}\gamma_{\mathrm{m}} \approx 10.5$ as we use double precision, confirming Abate and Whitt's "3/2 rule for the ratio of decimal places to final accuracy". This rule originates from a programming nuance in many scientific computing languages where there is a fixed number of decimal places for each number.

To further this analysis, several plots examine how the core integrand, resulting from the IDFT procedure, behaves under different discretisation settings when valuing barrier options. Figure 4.4 is a two-dimensional slice of the real and imaginary parts, overall magnitude and phase-shifted real part of $\widetilde{f}(z)$, revealing how the integrand oscillates and how its magnitude evolves. This gives some sense of the stability and resolution of the IDFT contour integration. The comparison between 52 and 252 monitoring dates demonstrates how increasing the number of points refines the sampling, potentially improving numerical accuracy and stability.

The three-dimensional figures, Figures 4.5 and 4.6, offer a more detailed perspective by plotting the real and imaginary parts of $\widetilde{f}(z)$ across both j and the magnitude r. These plots demonstrate that the IDFT approach, combined with a well-chosen discretisation, yields an integrand $\widetilde{f}$ whose behaviour can be controlled.

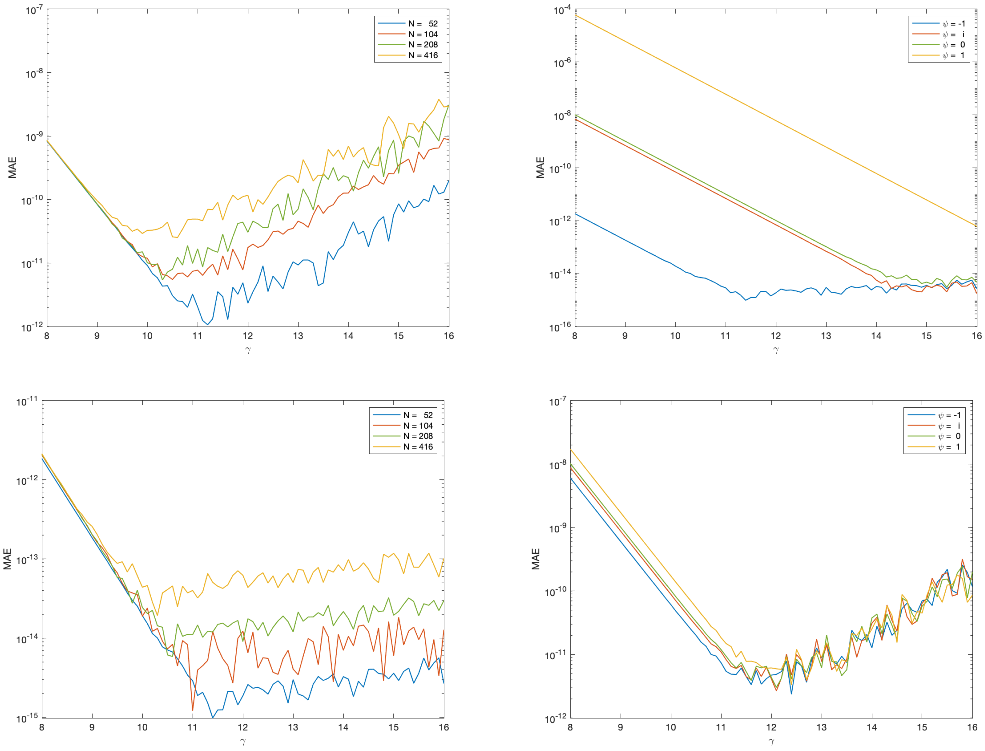

Figure 4.1: Mean absolute error of the inverse z-transform as a function of $\gamma = J\log_{10} r$, computed with Eq. (4.4) for the analytic pair given in Eqs. (4.8) and (4.9); $n = 0, \ldots, N$, $J = 2(N+1)$. Top: $\Delta t = 1/N$; bottom: $\Delta t = 1/252$; ; left-hand plots vary the number of monitoring dates N with $\psi = -1$ fixed; right-hand plots vary ψ (here $\psi = -1$ and $\psi = -2$) with $N = 52$ fixed.

4.2 Alternative inverse z-transform and modified Euler acceleration

The proposed alternative presented in Chapter 3 is the contour integration method presented by Cavers (1979). Under this approach the inverse z-transform can be approximated by

$$f(n) \approx \frac{1}{N}\sum_{k=1}^{N} r^n \widetilde{f}(re^{2\pi ik/N})e^{2\pi ikn/N}. \tag{4.10}$$

This representation is equivalent to an inverse discrete Fourier transform of the function $\widetilde{f}(re^{2\pi in/N})$ multiplied by the scaling factor r^n. We can thus apply an FFT algorithm directly

$$f(n) = r^n \mathcal{IFFT}\left(\widetilde{f}(re^{2\pi ik/N})\right), \tag{4.11}$$

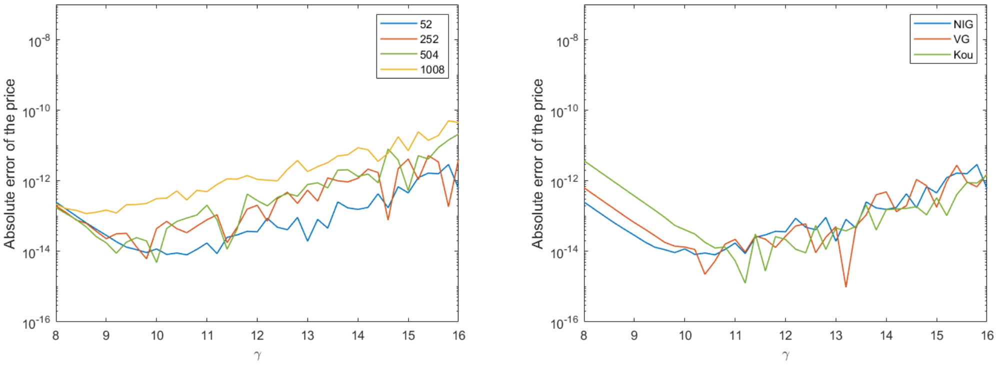

Figure 4.2: Absolute error of the price of a single barrier option with N monitoring dates and various stochastic models for the underlying. Left panel: normal–inverse-Gaussian (NIG) model; right panel: variance-gamma (VG) model, each evaluated at $N = 52$.

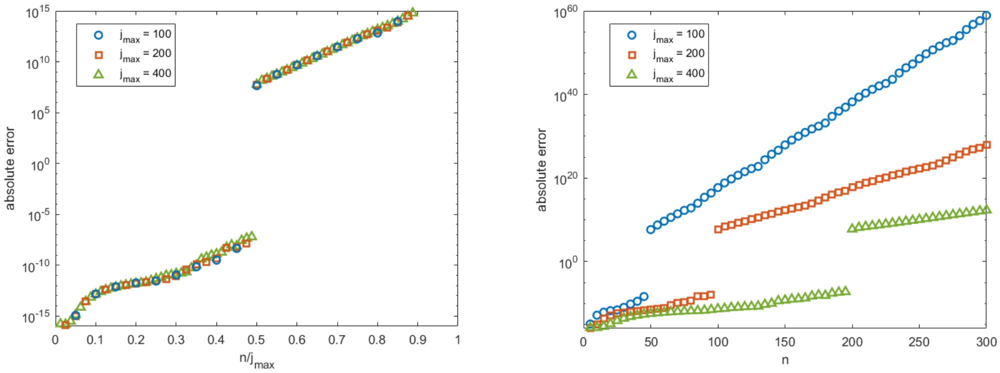

Figure 4.3: Error explosion where Shannon-Nyquist relation is broken. In this case the function is the same as Figure 4.1 i.e. $\widetilde{f} = 1/(1 - q\exp(\psi\Delta t))$ where $\psi = -1$ and $N = 300$, where $n = 0 : N$.

where $\mathcal{IFFT}$ is a computational IFFT operator and is the inverse of the forward Fourier transform. Due to having access to a secondary grid, the method can achieve machine accuracy with double precision.

The cost of improving the accuracy is a noticeable speed reduction since a large time domain requires enough Fourier points to ensure machine accuracy. FFT libraries are highly optimised, and while there is some speed reduction due to this array, the computation of the FFT is not the leading cause of computational costs. This is due to other parts of the ZS method, such as the Hilbert transform and fixed point algorithm in the case of the double barrier option. Furthermore, series

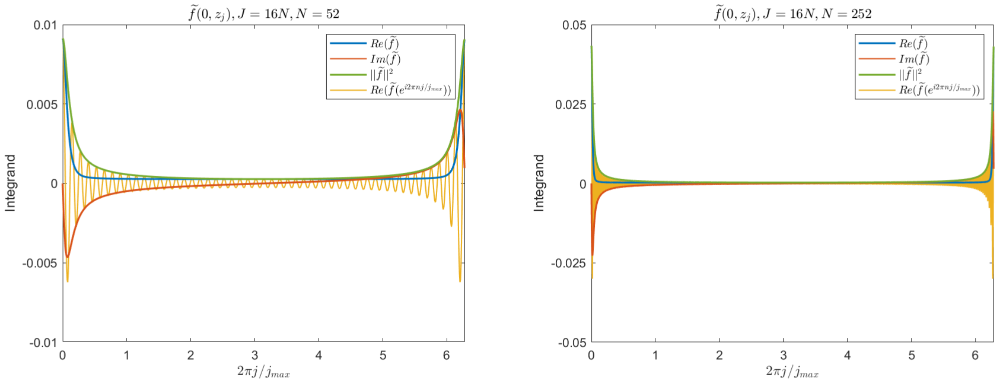

Figure 4.4: Integration envelope for IDFT method over 52 and 252 monitoring dates respectively using NIG distribution.

Figure 4.5: Surface for the real part of the characteristic function for IDFT method over 52 and 252 monitoring dates respectively using NIG distribution.

acceleration techniques can be applied to make the method date-independent in the same fashion as the original inverse z-transform. Due to the success of the Euler acceleration, as used in the Abate and Whitt inverse z-transform, we also introduced a modified version for use with other inversion methods where the number of points on the integral contour is not confined to a value of $2N$.

The Euler acceleration is designed to be used with convergent series that have alternating or nearly alternating terms. The design of the Abate and Whitt method with $2N$ terms means that there is a $(-1)^j$ term in Eq. (4.6), which gives us this property.

However, when we increase the number of points to a higher multiple of N, for example, 8, we lose this alternating property. This effect is illustrated in Figure 4.7, and the result of this is that the Euler acceleration cannot be directly used with the methods with several terms greater than $2N$. Each subplot shows the real and imaginary parts of $\widetilde{f}$, its magnitude, and the real part after introducing a phase factor $\exp(i2\pi j/J)$. As J increases beyond $2N$, the basic $(-1)^j$ pattern that drives the alternating-series convergence no longer holds. In particular, the case $J = 8N$ demonstrates a repeating sequence of four positive and four negative terms. While convergence holds, it does not strictly alternate each successive term.

Figure 4.6: Surface for the imaginary part of the characteristic function for IDFT method over 52 and 252 monitoring dates respectively using NIG distribution.

Figure 4.8 focuses specifically on the $2N$ and $8N$ cases, overlaying the sampling points to highlight the convergence behaviour and the emerging pattern in the higher-resolution case. The figure underscores that the series remains well-behaved, but the “four-on–four-off” sign pattern diverges from the classic $(-1)^j$ alternation. Notably, in the $8N$ case, the sequence oscillates between $||\widetilde{f}||$ and $\mathrm{Im}(\widetilde{f})$, whereas in the $2N$ case it primarily alternates between $\mathrm{Re}(\widetilde{f})$ and $\mathrm{Im}(\widetilde{f})$. While the series is still convergent and ultimately suitable for numerical evaluation, it illustrates why applying Euler acceleration techniques becomes more nuanced or infeasible once the number of summation terms exceeds $2N$ without the use of additional steps.

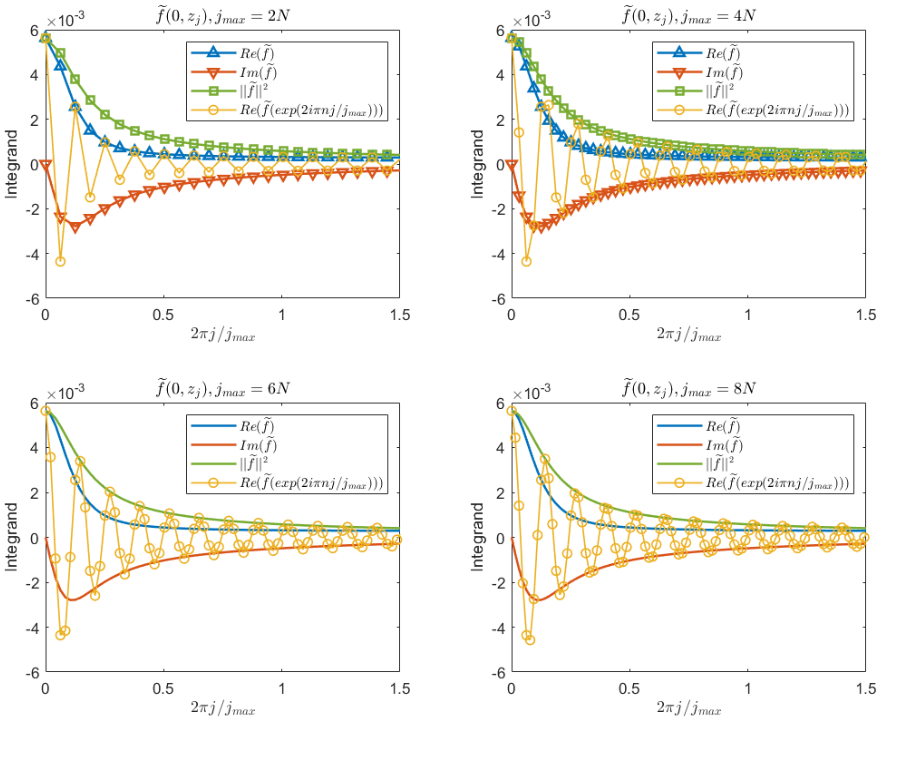

Figure 4.7: The input to the summation for a numerically performed contour integration of $z^j\widetilde{f}(0, z)$ with $2N$, $4N$, $6N$ and $8N$ integration points on the contour. Notice that the series is convergent and follows the pattern of 4 positive terms followed by four negative terms in the case of $8N$, with downsampling applied.

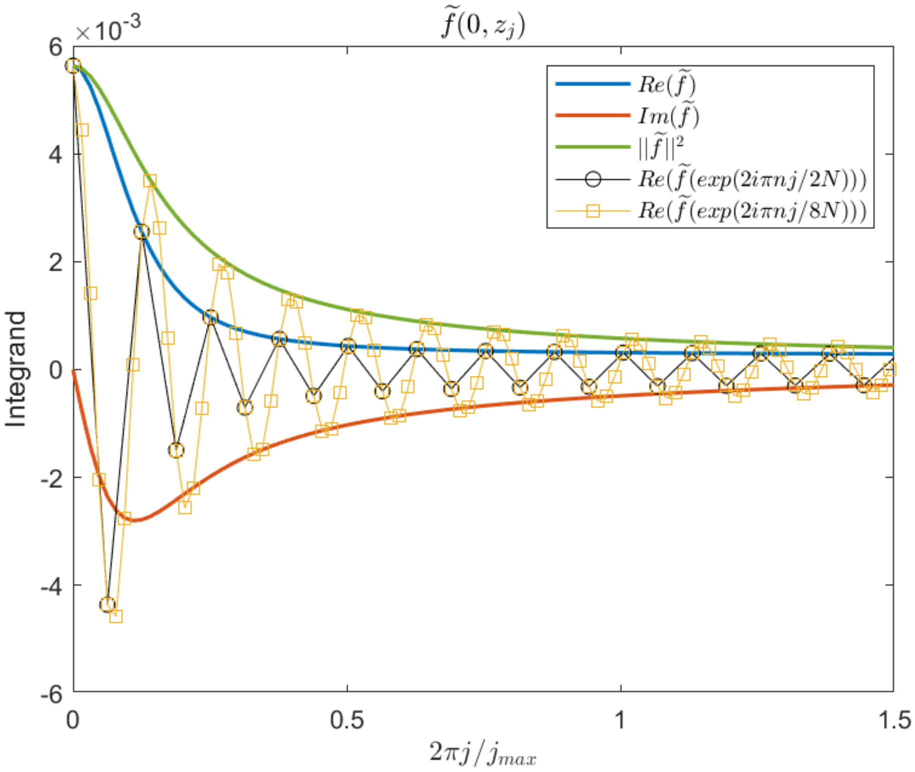

Figure 4.8: The input to the summation for a numerically performed contour integration of $z^j \widetilde{f}(0, z)$ with the number of points on the contour integral is $2N$ and $8N$ with the sampling points overlayed. Notice that the series is convergent and follows the pattern of 4 positive terms followed by four negative terms in the case of $8N$.

Notwithstanding this limitation, it is nonetheless true that the Euler acceleration is a widely used and effective method. Therefore, we looked at preprocessing methods to make the series more suitable for use with this method. The simplest method is to sum the terms in groups of β, where the number of terms on the contour integral is $2\beta N$. This then produces an alternating series with the number of terms reduced by a factor of β. This reduced series can then be used with the Euler acceleration.

The preprocessing and acceleration procedure for accelerating the summation of such a series a_n is then as follows,

1. For the first β terms in the series, calculate

$$a_1^\beta = 0.5a_1 + a_2 + \cdots + a_\beta, \tag{4.12}$$

where the 0.5 coefficient on the first term is because we are using the half circle's integral to calculate the full circle's integral.

2. For $k = 1, 2, \ldots, nE + mE$, calculate

$$a_k^\beta = a_{k\beta+1} + a_{k\beta+2} + \cdots + a_{(k+1)\beta}. \tag{4.13}$$

3. For $k = 1, 2, \ldots, nE + mE$, compute the partial sums

$$b_k^\beta = 2\sum_{j=1}^{k} a_j^\beta. \tag{4.14}$$

4. Take the binomial average of b_k^β between the $k = n_E$ and $k = n_E + m_E$,

$$f_n \approx \frac{1}{2^{m_\mathrm{E}} n\rho^n} \sum_{j=0}^{m_\mathrm{E}} \binom{m_\mathrm{E}}{j} b_{n_\mathrm{E}+j}^\beta. \tag{4.15}$$

4.3 Results

We present the results from numerical tests using the pricing schemes described in Chapter 2, with the different numerical inverse z-transforms presented in Chapter 3. Results for both double barrier options and α-quantile options are presented. Details of the contract and the model parameters are included in 7. The numerical results were produced using MATLAB R2020a running Windows 10, an Intel(R) Core i7-9750H 8 core 2.60GHz processor and 32 GB of RAM.

We present the results for the ZS method for discretely monitored double barrier options for the normal inverse Gaussian (NIG), Kou and Variance Gamma (VG) process. We show that applying the new inverse z-transform reaches machine accuracy for both options at the expense of computational cost. This is particularly evident in the double barrier case, as the non-accelerated version is required for machine accuracy. The method presented by Feng and Linetsky [2008] is labelled 'IH' and used as a numerical benchmarking for the methods. All ZS methods and one implementation of IH, labelled 'IHF', have the exponential filter applied. For further discussion on this point, consult Phelan et al. [2019]. The exponential filter improves the error convergence from polynomial to exponential. The 'ZSAW' method has the inverse z-transform performed using Abate and Whitt. The method labelled 'ZSC' has the inverse z-transform performed using the approximate contour integration method using MATLAB's IFFT algorithm. This has been replaced with a numerical summation over the grid points performed on a unit half-circle in z-space. Accordignly, we evaluate Eq. (4.11) as

$$f(n) = r^n \sum_{k=1}^{N} \widetilde{f}\left(re^{-\pi ik/N}\right) e^{i\pi kn/N}, \tag{4.16}$$

which is labelled by 'ZSCH'. The Cavers' method with Euler's acceleration, discussed in Chapter 4.2, is labelled 'ZSCE'. The latter computation is performed on the half circle using Eq. (4.16). The number of monitoring dates chosen is 52, 252 and 504. This represents weekly, daily and bi-daily monitoring.

Figures 4.9 through 4.16 plot the error convergence of the price against different grid sizes M for a double barrier option with increasing monitoring dates, typically $N = 52$, $N = 252$ and $N = 504$. Each plot utilises a different underlying distribution, namely Kou, normal inverse Gaussian or Variance Gamma, or changes the option's expiry from three months to two years.

Conversely, Figures 4.17, 4.18, and 4.19 plot the error convergence of the price against the CPU time. In this case, the dependence on the expiry of the contract is not interesting; rather, the number of monitoring dates is. Thus, only the underlying distribution is changed.

Comparing the results for error across the different processes, we see a general improvement in the model using Cavers' method, especially utilising Euler's acceleration. As the number of monitoring dates increases, performance is slightly improved, as seen from the process using FFT libraries. This is expected as these libraries tend to be highly optimised. Both methods successfully achieve accuracies of E$-$16, with some slight variations resulting in error values of E$-$15. A noted downside of using the ZSC and ZSCH methods is a considerable decrease in CPU time. Since the accelerated version achieves (ZSCE) machine accuracy with the same number of grid points, it is advised to use that method instead if one requires a machine-accurate solution.

This advantage is seen only in the error plots, as examining the CPU plots shows a marked disadvantage of the raw contour integration. The method is not slow in general, but with the contextual performance of comparative methods, there is a trade-off between method accuracy and method performance. Once Euler acceleration is switched on (ZSCE), the contour method's speed aligns with the Abate–Whitt baseline. For weekly monitoring it remains marginally slower than the IH family, but from daily (252) points onward the Euler-accelerated contour scheme overtakes IH, becoming the quickest route to machine accuracy—though it is still a shade slower than the unaccelerated Abate–Whitt (ZSAW) when only 6–7 significant digits are required.

Next, we present the results for a discretely monitored α-quantile option with a uniform monitoring interval $\Delta t = T/N$ between N dates, which is required to apply the inverse z-transform. This ensures that the ZS method remains independent of

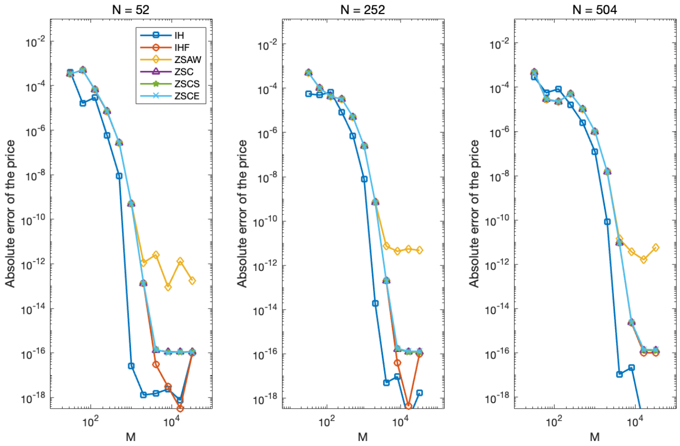

Figure 4.9: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the Kou process. The expiry for the contract is 3 months. The general performance of the ZSC methods is on par with that of the IHF method, and machine accuracy is achieved. The ZSAW method bottoms out at around E−11, which is coherent with the previous papers noting the error floor.

the number of monitoring dates. This restriction can be dropped to use the contour integration approach. For more information regarding this, consult Phelan et al. [2020].

Figures 4.20 and 4.21 plot the absolute price error against the grid size M and the CPU time, respectively. Instead of using different distributions or option features, such as the number of monitoring dates, a direct comparison of different pricing methods is used to check model error convergence. In this case, the Kou distribution is used. Whereas, for Figures 4.22 and 4.23, the normal inverse Gaussian model is used.

Results are provided with $\alpha = 0.667, 0.75, 0.833$ for $N = 252$ monitoring dates using the NIG process. The error convergence is notably fast, with full convergence achieved with a CPU time of 10^{-2} or less. Most notably, the accelerated version of the approximate contour integration achieves machine accuracy along with its non-accelerated sibling. This is due to a looser relationship between the Spitzer identities and the monitoring dates than in a barrier setting. This factor, combined with the lack of a fixed-point algorithm, results in a much smaller cost for making the method date-independent. As such, the unaccelerated version is omitted from

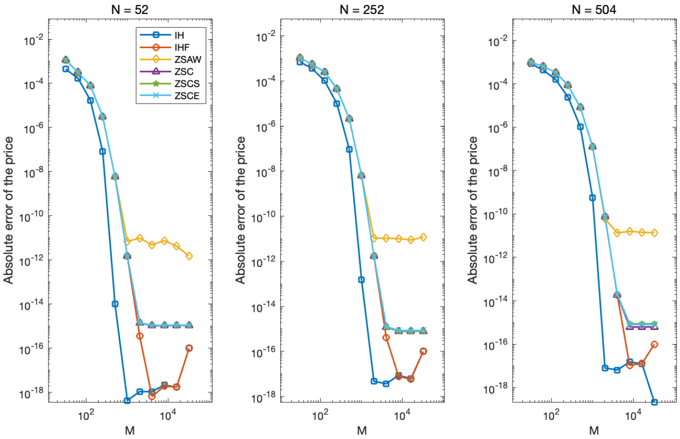

Figure 4.10: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the Kou process. The contract expires in 1 year. The ZSAW method converges to the error floor at a smaller grid size. The general performance of the ZSC methods matches the IHF method.

this section as its lack of speed makes it redundant when the accelerated version achieves machine accuracy.

An additional conclusion that can be drawn from our analysis is that exponential error convergence is suggested by plotting the error on a more extensive range of grid sizes M. An exponential error convergence conclusion can be drawn by increasing the range and noting the slower rate of decrease at the lower values of M. In the analysis performed by Phelan et al. [2020], the error floor caused by Abate and Whitt's method was too high to conclude this claim substantially. As such, it was only suspected in the original paper that the error convergence was exponential; a nice consequence of using Cavers' numerical integration approach can confirm this statement.

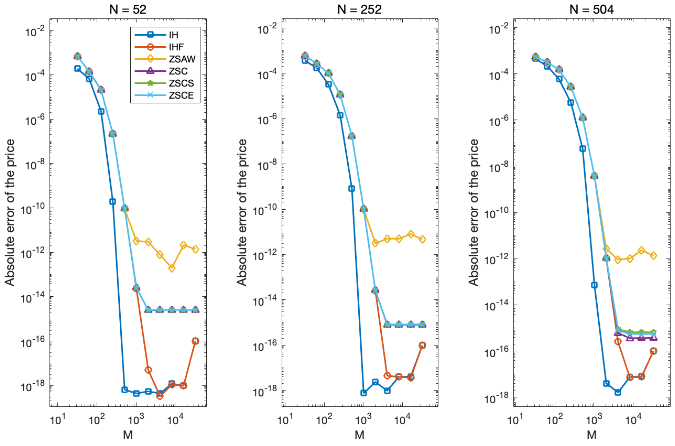

Figure 4.11: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the Kou process. The contract expires in 2 years. The performance of the ZSC methods under the Kou process depends more on the parameters used. For further discussion on these parameters, consult Schoutens [2003].

4.4 Conclusion

In this chapter, we showed that the z-Spitzer methodology described originally by Fusai et al. [2016] and extended by Phelan et al. [2019] can be improved with respect to the error floor by replacing the numerical inverse z-transform. The original numerical inversion was performed using the method proposed by Abate and Whitt (1992), which has insufficient freedom to select its error parameter, resulting in an error floor of approximately E−10. We replaced the inverse z-transform with an approximate contour integration technique, which removes this artificially high error floor and can result in an options pricing methodology with machine accuracy, i.e. E−16 with double precision. Results were presented for path-dependent options, namely double barrier and α-quantile options, which achieve error values of E−16. This is of particular value to the literature on α-quantile options as the error convergence rate could not be accurately determined from the paper by Phelan et al. [2020]. The results of Cavers' numerical integration method suggest that the error is exponentially convergent. The CPU performance is comparable to that of the original method and excels as the number of monitoring dates increases.

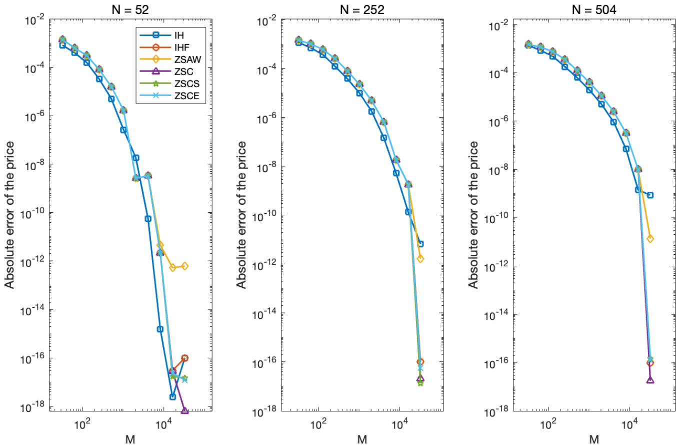

Figure 4.12: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the NIG process. The expiry for the contract is 3 months. The convergence of the price is notably slower due to the time step $\Delta t = T/N$.

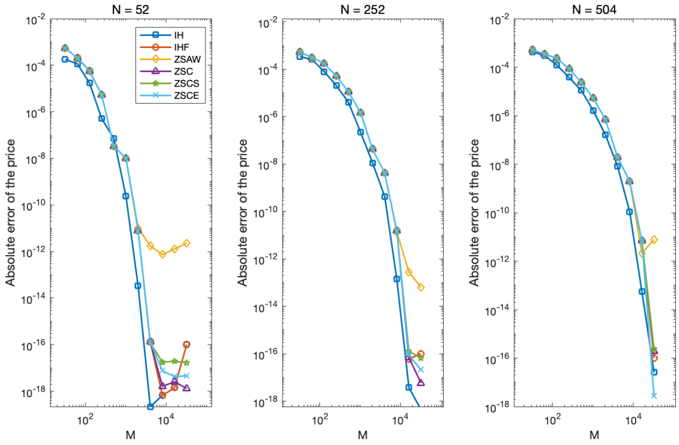

Figure 4.13: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the NIG process. The contract expires in 1 year. The convergence is faster at a lower number of monitoring dates. The performance of the methods follows the general scene of the ZSC methods matching the performance of IHF.

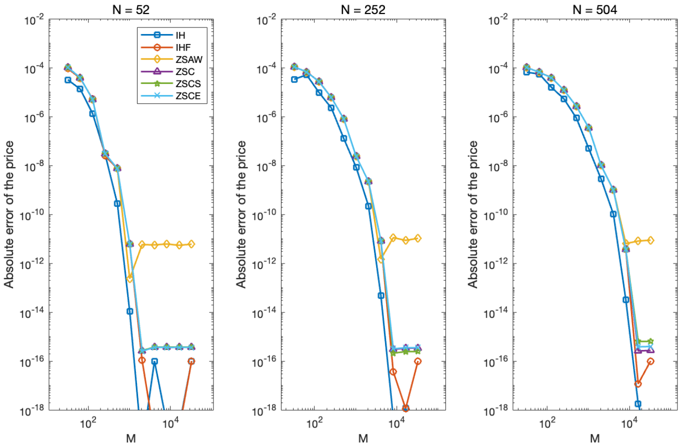

Figure 4.14: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the NIG process. The contract expires in 2 years.

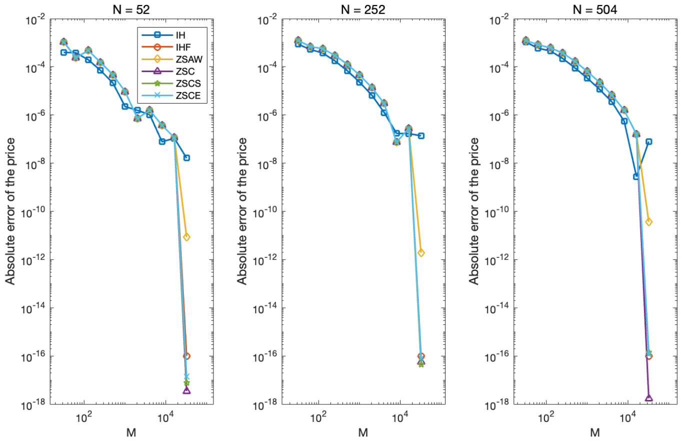

Figure 4.15: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the VG process. The expiry for the contract is 3 months. The variance gamma process is the most variable in terms of error convergence due to the time-changed nature of the embedded Brownian motion. The performance of the IH method without an exponential filter is notably poor under the VG process, due to the lack of an exponential filter.

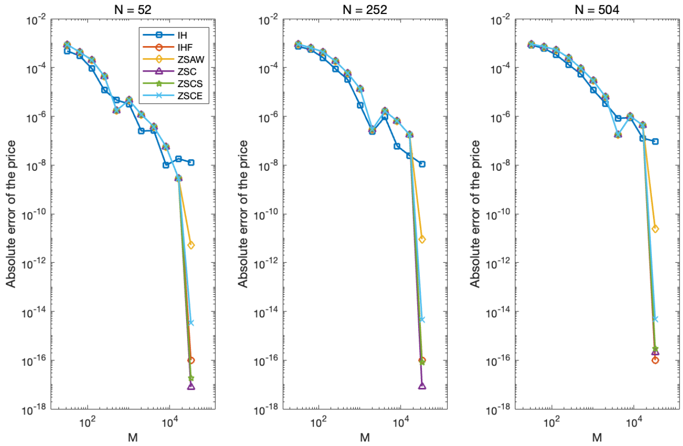

Figure 4.16: Error convergence against grid size M for a double barrier option with increasing monitoring dates using the VG process. The contract expires in 1 year.

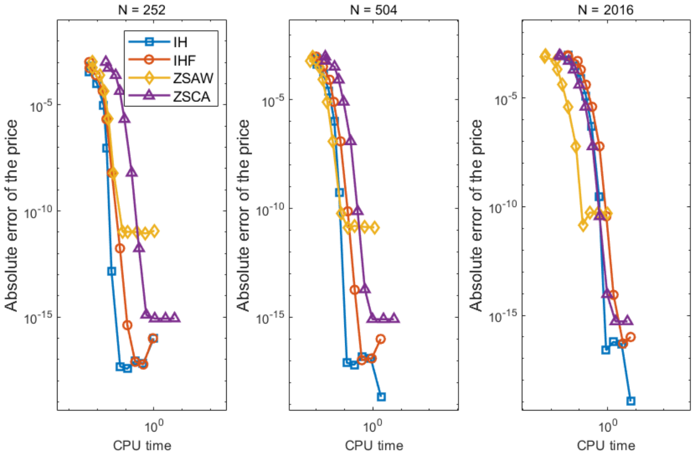

Figure 4.17: Error convergence against CPU time for a double barrier option with increasing monitoring dates using the Kou process. This shows the clear downside of the non-accelerated ZSC method. Due to the slowness of the method, it is removed from the higher monitoring dates as the method becomes practically unusable. The ZSCE method begins to perform best at the highest monitoring dates when the IH methods begin to slow down as monitoring dates increase.

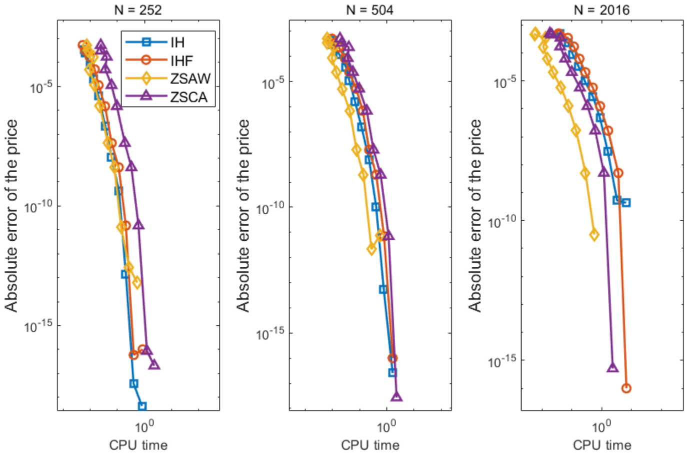

Figure 4.18: Error convergence against CPU time for a double barrier option with increasing monitoring dates using the NIG process.

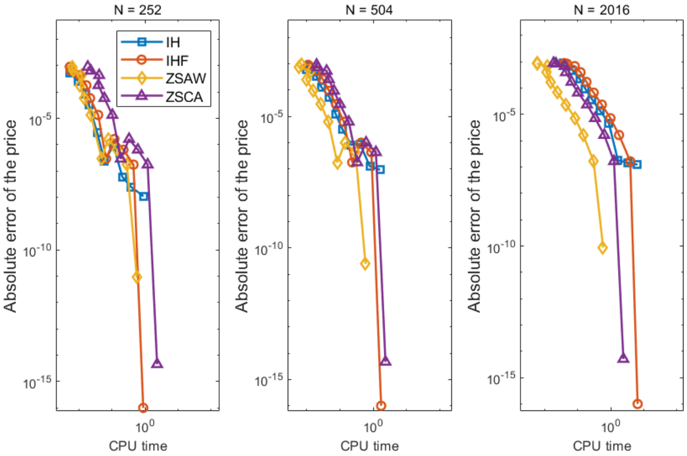

Figure 4.19: Error convergence against CPU time for a double barrier option with increasing monitoring dates using the VG process.

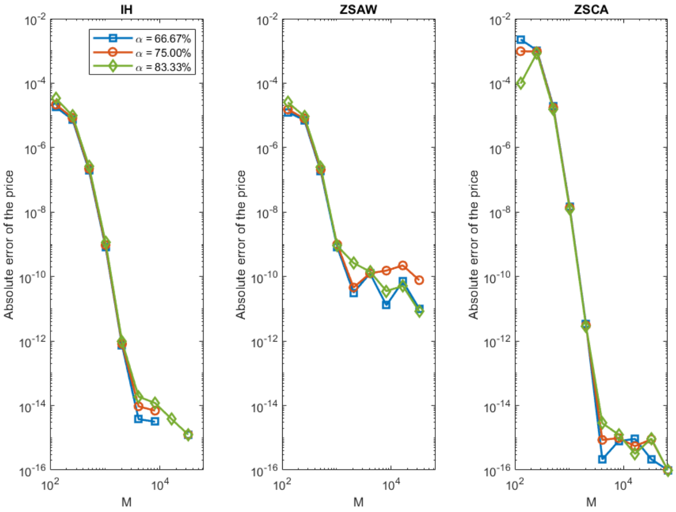

Figure 4.20: Error convergence against grid size M for an α-quantile option with various levels for α using the Kou process. The IH method eventually reaches machine accuracy at higher grid points. The ZSCE method performs very well in this context with exponential convergence.

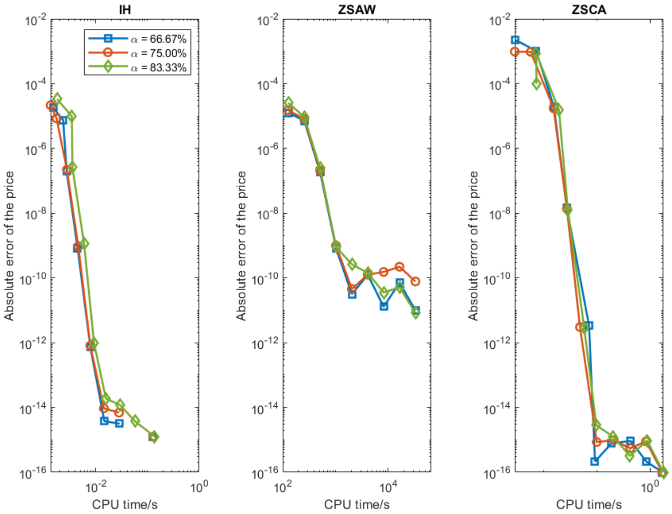

Figure 4.21: Error convergence against CPU time for an α-quantile option with various levels for α using the Kou process. The method with the best speed is the IH method, but the ZSCE method outperforms the original ZSAW.

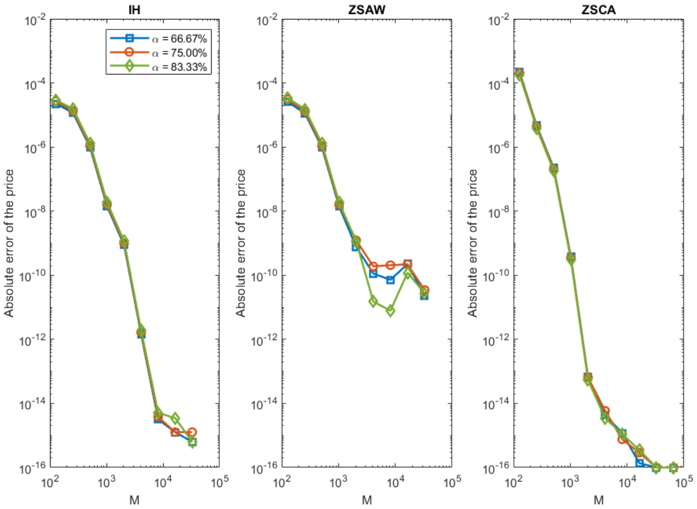

Figure 4.22: Error convergence against grid size M for an α-quantile option with various levels for α using the NIG process.

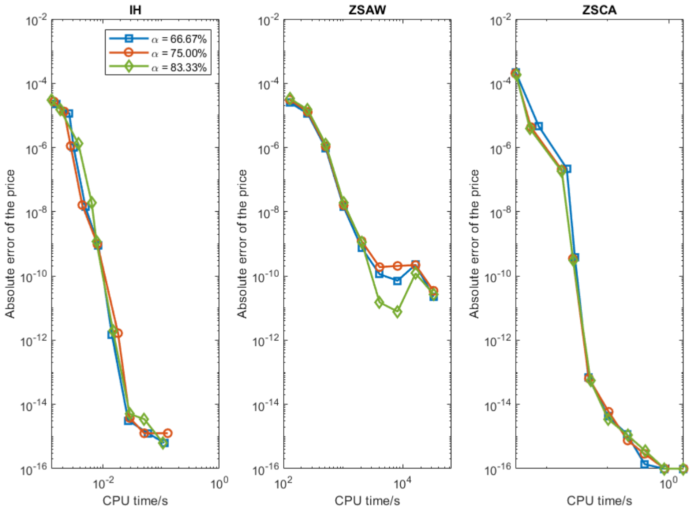

Figure 4.23: Error convergence against CPU time for an α-quantile option with various levels for α using the NIG process.

Chapter 5

Machine learning for the pricing of path-dependent options

In this chapter, we discuss and present results for fitting a deep neural network to the ZS method discussed in previous chapters. The motivation is to explore the usage of neural networks in pricing an exotic derivative and to compare the methodology to that of a sophisticated pricing method. We fit a 4-layered MLP process to synthetic pricing data generated via repeated pricing calculations using a range of randomised parameters for the options contract.

5.1 Machine learning

Despite its notable advantages, the Fourier pricing framework has been adopted by practitioners minimally compared to other methodologies. A potential solution is to use Artificial Neural Networks (ANN) to train a model to predict option prices using the Fourier methodology as an underlying engine. Since performance is a key metric, the computational expense of using machine learning must be compared to state-of-the-art pricing methods.

Deep learning was introduced as a numerical method in a financial setting by White [1988] for stock price prediction, following the same author's work regarding single-layer neural networks, White [1989]. This work was extended to option pricing by Malliaris and Salchenberger [1993], where the authors applied neural networks to vanilla options for price estimation using equity price data. The authors demonstrate that for some instances, notably those with short time horizons, the mean-squared error is lower when compared to the Black Scholes calculated price used for validation. A common theme of these papers is symptomatic of the early days of personal computing: the lack of computational power available to train and store models. This lack of computational performance had the notable effect of rendering machine learning methods broadly unusable. While an issue, it did not stop research into the area. With time and computing power, researchers have returned to machine learning as a viable model to use in practice. This can be easily witnessed in the numerous advancements in computer vision, medicine, image, text and speech recognition techniques.

Many works have been published with varying degrees of completeness on machine learning applied in some financial context, as detailed by Ruf and Wang [2020]. As noted by Hahn [2013], most of this work is focused on vanilla options, where daily stock data is typically the format of choice. Exotic derivatives have not been explored to the same degree. Early notable works in exotics include Lu and Ohta [2003a] for complex power options and Lu and Ohta [2003b] for rainbow options.

One of the key papers by Hutchinson and Poggio [1994] used Black Scholes simulated option prices to provide an alternative pricing methodology when the underlying price dynamics are unknown. The authors tested multiple models, including MLP networks and Ordinary Least Squares models, with varying success; notably more successful was the accuracy when predicting price data rather than Δ-hedging predictions. Subsequently, this is one of the key ideas for using simulated pricing data rather than the more unreliable market data.

An interesting approach was presented by Lajbcygier and Connor [1997] that proposed to learn the difference in prices between the Black Scholes calculated price and the market observed price. Similarly, Anders et al. [1998] compared ANN-predicted option prices with the Black Scholes model when using different volatility estimates. This approach demonstrated some promise regarding explanatory statistics for variables within the network but was not extended beyond European call options.

Up until this point, canonically, ANNs had been built with naivety. Notably, Garcia and Gençay [1998] applied a homogeneity hint to the ANN to embed financial knowledge into the network construction. This homogeneity hint reduces out-of-sample mean squared prediction error compared with the naive version. This method is not widely copied since reducing market data to genuinely homogeneous requires many transforms.

While the focus of most research is limited to prices, it is to be noted that other key factors of derivatives markets can be the subject of prediction. For instance, Carverhill and Cheuk [2003] takes a different approach and expands upon option prices to examine hedging parameters in depth to predict a more efficient hedging strategy. Dugas et al. [2009] presents a model that enforces no-arbitrage conditions, such as the convexity of option prices, at the cost of model simplicity. A noted difficulty in both approaches is the degree of hyperparameter tuning required to establish a working model.

A common problem with many of the mentioned papers is the availability, reliability and reproducibility of the market data used to train and test the models. As such, the investigation into the performance of ANNs trained on synthetic data has continued with the complexity of the training model progressing beyond vanilla options and the Black Scholes model. This avenue of research has begun to combine with utilising ANNs to approximate partial differential equations in a wide range of areas, most notably fluid dynamics, Kochkov et al. [2021].

Liu et al. [2019b] produced an optimised ANN, which was trained and tested on synthetic data sets generated by the Black-Scholes model to value options and calculate implied volatilities. The same paper then extended this work to a dataset based on the Heston model and calculated using the COS method, see Heston [1993] and Fang and Oosterlee [2009]. Furthermore, ANNs were beginning to be applied to more complex models and contracts by McGhee [2021] and Horváth et al. [2020], where the synthetic data sets were based on the SABR stochastic volatility model, the rough Bergomi model and exotic options were considered. Neural networks estimate a model by learning from input data while minimising a cost function, typically based on the MSE or MAE between the predicted value and the training data. This follows from Cybenko [1989] and K.Hornik et al. [1989], who show that ANNs are universal approximators; given complete control over the number of neurons and their configuration, neural networks can approximate any function to a given level of accuracy.

However, a continuing challenge of ANN design and configuration is that we are limited to a given number of neurons by the size of our data set and available computing power, and finding their optimum configuration is not straightforward. Indeed, we can optimise our ANN configuration heuristically by selecting the best-performing candidate out of several candidates. However, it cannot be guaranteed that this is the optimal configuration for all options. An additional concern is when real-world data overfits the model due to the noise in the data. However, one significant advantage of using ANNs trained on synthetic data generated by a deterministic numerical method is that we are no longer concerned about overfitting, as the data is noise-free.

Other approaches have also been explored more recently. Andreou et al. [2010] proposed an ANN that returns implied model parameters. This approach comprises an informal mapping of options to a parametric model used to determine option prices. This significantly improves the computational cost of calibration as the process can be moved away from a live data environment. More recent efforts by

Hernandez [2016] result in an ANN calibrated to a single-factor Hull-White model. Dimitroff et al. [2018] and Liu et al. [2019b] calibrate stochastic volatility models and Stone [2019] and Bayer et al. [2019] calibrate rough volatility models. Additional avenues of investigation are using ANNs to solve partial differential equations such as the Black Scholes model; see Han et al. [2018] and Beck et al. [2021]. Their approaches reformulate a PDE using backward stochastic differential equations, where ANNs approximate the gradient of the unknown solution. The results suggest that the method is effective for various problems with multiple dimensions.

5.1.1 Brief introduction to Artificial Neural Networks (ANNs)

Artificial neural networks (ANNs) were a primitive attempt to capture the human brain's learning process to solve complex, multi-layered algorithmic problems. ANNs learn to perform tasks through experience in data form and are generally not explicitly programmed with instructions during the learning process. A series of simulated nodes, more commonly referred to as neurons, are constructed in layers to form the structure of a neural network. Typically, these layers come in three distinct types. An initial input layer handles the initial data for the network and generally processes data taken as input. An output layer handles the results of the trained network and has an equivalent number of neurons to the number of outputs required. For example, if one seeks a single option price as a prediction, the output layer requires only a single neuron. In between the input and output layers are known as hidden layers. These intermediate layers consist of neurons that cannot be accessed during the training phase in a vanilla neural network. The number of hidden layers depends primarily on how much we wish to train the network. There is a linear relation between the number of layers (and neurons) and the computational cost of training the network. However, the more layers included, the more accurate the results. This last statement is generally correct, but it is not a guarantee. Each hidden layer can be programmed to perform different transformations depending on the use case and thus learn slightly differently from the other networks. An example construction is given in the appendix Fig. (7.1).

Neurons within each layer connect to other neurons in adjacent layers, with each link having a set weight. A neuron receives input, performs some transformation, which can be minimal, applies the weight and passes the value to the activation function. Letting the input of a neuron be the n-value vector $X_n = (x_1, x_2, \ldots, x_n)$ and the output be an m-value vector $Y_m = (y_1, y_2, \ldots, y_m)$, we set a bias node $x_0 = 1$ which is analogous to the intercept in a regression model and let the weights between neurons be the matrix $\mathbf{W}_{m\times n}$. Then the output of the j^{th} neuron can be written as

$$y_j = \phi\left(\sum_{i=0}^{n} w_{ji}x_i\right), \tag{5.1}$$

where $w_{j0} = b_k$ is the offset for the input and $\phi(\cdot)$ is the activation function. Fig. (7.2) shows a visual example of the neuron learning process. This learning process is often called a perceptron, where an artificial neuron is a generalisation. The term perceptron was coined by Rumelhart et al. [1986] to mirror the terminology present in psychology due to the comparison between the human brain. This is naturally extended to multi-layer perceptrons (MLP), which consist of multiple layers of perceptrons composed into a network.

The learning process for MLP networks is an initial forward propagation consisting of several steps. Initially, inputs are multiplied by starting weights, and their sum is calculated, including the bias node. These weights are fine-tuned with every pass of the learning process to produce a more accurate solution. This weighted sum is applied to the activation function to map inputs to outputs. The choice of activation function is an important issue in MLP training, and many choices are available. These choices range in complexity to allow the fitting of MLP networks to more complicated data. The two most commonly used historically are the tanh and sigmoid activation functions, given by

$$y(v_i) = \tanh(v_i), \tag{5.2}$$

and

$$y(v_i) = (1 + e^{-v_i})^{-1}, \tag{5.3}$$

respectively. The output from the activation function forms the output data, which is subsequently refined through multiple training iterations. Optimising the required weights is known as backward propagation, where a reverse pass from output to input fine-tunes the required weights to improve the output. This is achieved by calculating the derivative of the activation function for each successive neuron and solving for the minimisation problem that results. Other practical issues, such as epoch and batch size, determine the training size supplied to the network at any one time by partitioning the data into more manageable sizes.

Numerous problems exist when handling MLP networks. An initial barrier to entry is the amount of data required. Deep learning is notorious for requiring large amounts of data for a network to be trained correctly. Compounding this is the need for quality data as well as quantity. As such, this approach is highly sensitive to data. An additional consideration is the problem of over- and under-fitting. An over-fitted model is one in which the model is trained to a high degree on a specific set of data but subsequently struggles with accuracy on unseen data presented to the model.

In contrast, under-fitting occurs when the network cannot accurately predict the data on which it was trained. Cross-validation is a tool to minimise the over-fitting effect, validating the prediction accuracy for the dataset on which the network is trained. The most common form of cross-validation is K-Fold validation, which splits the data into K partitions used as a test set and $K-1$ as a training set. The error estimation performed by the validation is averaged over K trials to get the total effectiveness of the model.

The field of MLP networks is vast, and the literature is dense. The discussion given in this thesis up until this point is enough to keep it self-contained, but does not scratch the surface of the topic. For a detailed look into MLP networks, the interested reader should consult Dreiseitl and Ohno-Machado [2002].

5.2 Data Generation

A known disadvantage of deep learning solutions is that they require vast amounts of data for consumption in the training and testing process. This has led to the use of synthetic data, which can be generated quickly and efficiently. Further, synthetic data has the noted advantage of being reproducible and verifiable. While this ultimately means any model trained on synthetic data loses the randomness associated with messier data, synthetic data is hugely advantageous for focused tasks such as option pricing.

We thus begin by generating a sufficiently large data set using the Fourier-z model with parameters set at randomised values as input to build a database of prices and Greeks. In this case, 50,000 prices for each distribution is a good benchmark. The two chosen distributions to use are the normal inverse Gaussian (NIG) and variance gamma (VG) models. The distributions chosen provide a difference in stability, where the VG model has well-documented stability issues, and the NIG model is widely more stable than the available models.

The data generation process requires using previously discussed Fourier pricing approaches applied to double barrier options, utilising the Abate and Whitt approach to inverting the z-transform. This gives confidence that even though what is being proposed is a black box technique, it ultimately attempts to predict based on a well-known and established methodology. To generate the pricing data required to train the ANN, we repeatedly run the pricing process while selecting randomised market and contract data, which is extended to distribution model parameters. This process results in multiple data sets with varying price accuracies, which the Fourier grid size can control.

The two data sets generated use the NIG and VG processes, respectively, with the highest number of grid points, 2^{32}, required to produce stable lowest errors, namely E-12 and E-08 for each process, respectively. Several parameters were chosen to generate the pricing data in Table 5.1. These parameters typically reflect the options listed in a limit order book, e.g. those with different strikes, barriers and the number of monitoring dates. For completeness, we also vary the spot price and the risk-free rate. Some parameters for the underlying distributions are also varied to emulate different calibrations one may see for the underlying processes once fitted to market data. For an introduction to the distribution processes and their parameters, consult [Schoutens, 2003]. All parameters are normalised to optimal training results.

Additionally, the option Greeks are calculated to provide a complete view of the option in the data. The Greeks are calculated by applying a small bump to the relevant underlying contract parameter.

The advantage of artificial data in this context is the ability to generate more data if required, a useful feature considering deep learning's requirement for excessive amounts of data. This approach also allows us to be far more liberal with the test/train split one must choose when training models. Typical choices for this split are 80/20, with the justification being that 20% of the data can reflect the underlying features and patterns. With synthetic and reproducible data, one has the luxury to push this split far higher due to the observation that small portions of the data will have the same macro features as more significant subsets of the whole. Furthermore, the data is wholly deterministic. This means that when using it, noise is eliminated from the dataset. Noise and error can significantly diminish the performance of ANNs and their training, a problem not typically present with artificial data.

Parameter	Range
spot	$40 - 160$
monitoring dates	$5 - 550$
lower barrier	$40 - 100$
upper barrier	$100 - 160$
risk free rate	$0.0 - 0.1$
strike	$40 - 160$
α - (NIG)	$3 - 18$
δ - (NIG)	$0.01 - 3$
ν - (VG)	$1 - 5$
θ - (VG)	$3 - 14$
stddev - (VG)	$12 - 18$

Table 5.1: Market and parameter ranges for the generation of artificial data using the Fourier-z method. A noted potential quirk in the truly randomised approach is that many contracts can be widely meaningless, thus resulting in many prices of essentially 0. This is still useful, however, as worthless contracts are present in the market and can help with model pattern recognition.

5.3 Methodological Framework

Once data is generated, the standard procedure is followed, which is to split the data into a training and validation set. We employ a general grid search for various hyperparameters to find the optimal ANN model via K-fold cross-validation for our particular problem. We then use this data to train the resulting optimised ANN model with the training set and test the resulting model with the validation set.

There are two types of parameters involved in training a neural network. The first set is model parameters, which are adjusted during the training process, and the second is hyperparameters, which are set before the process begins and dictate the neural network structure and training process. While setting model parameters is a relatively straightforward problem, finding optimised hyperparameters is not trivial.

Typically considered a multi-dimensional optimisation problem, one aims to find the optimal combination of parameters that achieves the smallest error. This optimisation type is often considered more of an art than a science Claesen and Moor [2015]. As the application of deep learning continues to become more constrained on focused problems rather than a more significant generic problem, the focus is shifting to better methods of optimising hyperparameter tuning, for example, Kumar et al. [2021] and Zhang et al. [2021]. Avoiding a computationally expensive grid search is preferred in such a problem. However, modern methods still struggle to compete with the accuracy of the hyperparameters from an exhaustive and iterative process. Parallel programming makes this more palatable, as the grid search is not subject to data races. The evaluation of parameters can be done independently and is limited by CPU power only.

Since our approach also involves predicting the options' Greeks, we require something more advanced than the typical single DL model. Two choices are apparent: a single model with multi-head architecture or separate models for each desired output. Since the data is generally causal, i.e., the data will reproduce the same valuations each time, and those output parameters are linked mathematically, a multi-head model seems a plausible solution.

In this design, the early layers form a shared representation, a series of transformations that extract general features of the input data, while separate heads branch out from this shared backbone to produce distinct outputs. The network leverages the commonality among the various outputs by training all heads simultaneously, given that option price and its sensitivities often share underlying dependencies. However, each head can learn task-specific parameters that fine-tune the predictions for its particular target. This setup also helps manage differing loss functions, enabling each head to optimise its performance without interfering with others. In practice, multi-head models have proven especially beneficial when several closely related tasks can be learned jointly, enhancing overall data efficiency and capturing complex interrelationships between the variables of interest.

Since the data sets are synthetic, we perform an extensive grid search evaluating almost all possible combinations of hyperparameters. The selections that are ignored are considered unreasonable due to excessively high prediction errors. A notable downside of using such an extensive search is the problem of overfitting. However, we apply cross-validation, where samples are divided into K-folds, to mitigate this issue.

The hyperparameters selected for the grid search procedure follow the currently accepted norms in the literature, see [McGhee, 2021, Liu et al., 2019b, Horváth et al., 2020], and are detailed as follows. A natural starting point is the number of hidden layers composing the network, where there is a theoretical tradeoff between model accuracy and computational cost. Typically, the more layers in the network, the more accurate the prediction results are, at the expense of a longer training time. From experimentation, there is a distinct zenith where the accuracy of the network is barely affected by increasing the number of layers, and the computational cost for this increase becomes unrewarding. A result that has been demonstrated [Bayer et al., 2019] is that adding additional layers beyond the initial four does not consistently reduce errors in the validation set.

Furthermore, the scenario where adding additional layers increases the model accuracy is unclear. Due to this inconsistency, we tested three, four, five, and six hidden layers using the grid search process. The typical choice within the literature is four hidden layers where the final layer represents the output, [Liu et al., 2019b, Horváth et al., 2020]. Since the model presented here is multi-output, three distinct output nodes are required.

Other parameters under consideration for the grid search are the number of neurons for each hidden layer. The batch size dictates the number of samples the network will pass through at each process. The number of epochs refers to the number of times the algorithm processes the dataset. Furthermore, ensuring that the training data size is divided by the batch size to eliminate the possibility of a final batch with fewer samples is good practice. The type of optimiser will be restricted to well-known examples and popular within the literature to avoid unnecessary complications; those chosen are Adam, SGD and RMSprop, [Hoffer et al., 2017].

The other parameters required are set using the current literature and the widely agreed-upon best practice. Because we are handling an option pricing problem, we choose the activation function to be the Rectified Linear Unit (ReLU) function, i.e. $f(x) = \max(0, x)$. This function is advantageous over other typical choices, such as sigmoid and tanh, because ReLU is a non-negative function that is similar to an option payoff function. The extremes for x in a ReLU function have a higher learning rate. Other non-negative and smooth alternatives, namely Elu and Softplus, were tested, as suggested in McGhee [2021] and Horváth et al. [2020]. ReLU generally outperforms these other methods, aligning with our testing and is shown in Ramachandran et al. [2017].

The final two parameters are the weight initialisation method and the dropout rate. The initialisation of weights is a deep topic of discussion in the deep learning literature, as the consequences of poorly initialised weights can render a deep learning model redundant. For more information consult Glorot and Bengio [2010] and He et al. [2015]. The dropout rate is designed to prevent overfitting. Overfitting is difficult to diagnose in certain settings where the underlying problem is additional parameters, such as an option pricing environment, [Liu et al., 2019b]. This hypothesis is supported by our experimentation with a range of dropout values, which did not improve the model's accuracy to any notable degree. The general algorithm for the hyperparameter search is displayed in Algorithm 1.

To evaluate the model following each successive run of the parameter grid search, we examine the mean squared error and R^2 score for both the training and validation samples. Using the parameters displayed in Tables 5.2 and 5.3, which constituted the optimal parameter settings following the grid search, we achieve the error values displayed in Table 5.4.

Parameters	Settings
No. of hidden layers	4
No. of epochs	500
Activation function	ReLU
Weights initialization	he_uniform
Dropout rate	0.2
Batch normalization	Standard

Table 5.2: The hyper-parameters set based on the current standards within the literature. While brief testing was done after the systematic hyperparameter search to assess other options, no meaningful progress was made. Thus justifying the choice to stick to the accepted norms.

Parameters	Test ranges	Optimal value
No. of neurons in 1st hidden layer	$20, 100, 400, 800$	800
No. of neurons in 2nd hidden layer	$20, 100, 400, 800$	20
No. of neurons in 3rd hidden layer	$10, 100, 200, 400$	100
No. of neurons in 4th multi layer	$10, 20, 50, 100$	50
Batch size	$200, 500$	500
Optimizer	SGD, Adam	Adam

Table 5.3: The range of parameter selections chosen at the start of the iterative search for the optimal model hyper-parameters. The 4th layer consists of three separate output layers, one for prices, one for predicting the options' δ and Γ and a final one for ρ, ν and $Theta$.

	Mean-squared error	R^2
Training set	0.2112	0.9976
Test set	1.5959	0.9822

Table 5.4: K-fold cross-validation results for the multi-head model indicate strong predictive accuracy, as evidenced by low mean-squared errors (MSE) and high R^2 values on both training and test sets. Although the MSE is higher for the test set than the training set, the test R^2 remains near 0.98, suggesting the model generalises well.

A key consideration of the machine learning approach is that the learning process acts as a black box. Optimisation typically relies on gradient-based methods (e.g., stochastic gradient descent and its variants), but high-dimensional parameter spaces can contain numerous local minima or other pathological features. As a result, the iterative search for an optimal network configuration may sometimes settle on seemingly arbitrary or unstable parameter settings, a phenomenon known to produce inconsistent or unexpected model behaviours. Early stopping, careful regularisation, and multiple random weight initialisations can mitigate these apparent issues, but none can fully guarantee a globally optimal solution. Furthermore, the interplay between hyperparameter choices (learning rates, batch sizes, network depth) and data availability complicates the training pipeline, emphasising robust validation methods like K-fold cross-validation. The mean-squared error table for the neuron structures with the optimal neuron sizes is given in Table 5.5.

n_2/n_1	20	100	400	800
20	2.83 (0.51)	2.08 (0.33)	2.13 (0.22)	1.75 (0.17)
100	2.84 (0.31)	2.26 (0.21)	1.97 (0.23)	1.93 (0.09)
400	2.98 (0.26)	2.32 (0.11)	2.07 (0.12)	2.14 (0.26)
800	3.13 (0.56)	2.25 (0.22)	2.11 (0.15)	1.96 (0.20)

Table 5.5: The mean-squared error and corresponding standard deviation in brackets for different combinations of the number of neurons in the first (n_1) and second hidden layer (n_2) and fixed hyperparameters.

The complete model setup, post K-folds search, can be seen in the Appendix Table 7.4 for the normal inverse Gaussian model and Appendix Table 7.5 for the variance gamma model.

5.4 Results

Due to the differences in underlying statistical distribution, a new ANN was trained to determine the performance of the model's predictive power for option pricing data and the options Greeks. The tables below display the results for each distribution with a range of contract parameters, each evaluating a new scenario. As we intend to compare the process of predicting a price against a notable state-of-the-art method, the CPU and error performance will be benchmarked against the fastest baseline method. For monitoring frequencies up to daily monitoring, $N = 252$, the inverse Hilbert (IH) method [Feng and Linetsky, 2008] performs best. For contracts with a higher number of monitoring dates, it has been shown that the z-Spitzer method has a lower computation time, as demonstrated in Chapter 4.1.

After the extensive hyperparameter search, the optimal model, within the search ranges, was found. The details of which are presented in the Appendix of this thesis, namely Table 7.4 for the normal inverse Gaussian model and Table 7.5 for the variance gamma model. Both models' seemingly odd node structure is noteworthy to the untrained eye. This is a consequence of the exhaustive search and is one of the hindrances to mass adoption, as it is unclear why such a node structure should result in optimal predictive power. Traditional heuristics might suggest a monotonic funnel, strictly decreasing or increasing layer sizes.

This phenomenon arises from how neural networks reconcile high representational capacity with regularisation. The network can explore a vast space of potential feature interactions by initially expanding the dimensionality from layer 1 (140 neurons) to layer 2 (400 neurons). This effectively teaches a richer set of basis functions for the data. Similarly, compressing this from layer 2 to layer 3 (60 neurons) enforces a bottleneck that forces the network to discard redundant or less important features. This process of expand–compress–expand pattern allows the model to both learn a complex structure of the data and avoid overfitting, aligning with the principles underlying auto-encoders, which use a bottleneck to enhance the generalisation power of a model.

Moreover, automated search strategies like the K-folds search or other evolutionary algorithms are driven purely by performance objectives rather than human intuition about monotonic layer sizing. These methods can discover architectures that maximise performance for a given dataset and task, even if such architectures deviate from common heuristics. Ultimately, this structure reflects a nuanced balance between capturing rich interactions in high dimensions, enforcing meaningful compression, and re-expanding to accommodate the final prediction space.

Several dummy contracts have been created to test these models and various parameters and market conditions. The whole table can be seen in Table 7.1. We first present the results from the ANN trained on the normal inverse Gaussian distribution.

contract(method)	1(IH)	2(IH)	3(IH)	4(AW)
price	7.2207	3.4637	0.8765	0.1970
δ	0.4514	0.4335	0.3586	0.0468
Γ	-0.0004	0.0327	-0.0140	-0.0439
ν	48.2822	563.7460	-44.3214	-115.3606
Θ	0.8049	6.1557	-0.3983	-0.0957
ρ	36.1498	18.8073	5.7372	-0.0393
CPU(s)	0.2775	0.2592	0.2597	1.0609

Table 5.6: Numerical valuations of option contracts 1-4 using the normal inverse Gaussian distribution. The fastest method is taken as benchmark, denoted by IH or AW for Feng & Lintesky or Abate & Whitt based on the Fourier-z framework.

contract	1	2	3	4
price	7.2340	3.7520	0.7915	0.1934
δ	0.4800	0.5741	0.3654	0.1146
Γ	-0.0325	0.1458	-0.0088	-0.0013
ν	48.2654	520.0154	-42.1324	-113.5240
Θ	0.9651	6.6654	-0.3481	-0.0423
ρ	38.0154	17.1623	5.8315	-0.0467
CPU(s)	0.0094	0.0106	0.0114	0.0132

Table 5.7: Predicted valuations of option contracts 1-4 using the neural network trained on synthetic data produced by the normal inverse Gaussian distribution. The predictions based on less volatile sensitivities are accurate enough for usage. However, more volatile Greeks such as ν seem to suffer accuracy loss.

contract(method)	5(AW)	6(AW)	7(AW)	8(IH)
price	3.3204	3.2832	1.1745	2.5197
δ	0.1232	0.2824	-0.0737	-0.3158
Γ	-0.0222	0.0021	-0.0111	-0.0040
ν	-2240.6138	1.0524	10.0147	-136.7948
Θ	-4.1431	0.9037	-2.3914	-1.4053
ρ	-0.7134	21.5991	-3.6851	-18.4062
CPU(s)	1.2276	2.4992	2.5003	1.2436

Table 5.8: Numerical valuations of option contracts 5-8 using the normal inverse Gaussian distribution.

contract	1	2	3	4
price	3.3647	3.6501	1.1658	2.5475
δ	0.1102	0.3014	-0.0445	-0.3144
Γ	-0.0015	0.0146	-0.0047	-0.0222
ν	-1650.01	2.0147	10.0014	-65.15
Θ	-3.0457	1.1125	-3.2470	-1.8870
ρ	-0.4135	23.0014	-3.0014	-16.1470
CPU(s)	0.0047	0.2270	0.3042	0.3101

Table 5.9: Predicted valuations of option contracts 5-8 using the neural network trained on synthetic data produced by the normal inverse Gaussian distribution.

The numerical results provided in Tables 5.6 and 5.8 serve as our benchmark for pricing and Greeks computation under the normal inverse Gaussian (NIG) model. The neural network (ANN) results, shown in Tables 5.7 and 5.9, are mathematically close for the option price and most of the Greeks, underscoring the ANN's ability to capture salient features of the option payoff and dynamics. Despite minor deviations in a few specific Greeks, such as vega, the ANN predictions exhibit competitive performance in terms of accuracy. While not good enough to rely on as a definitive pricer, it is certainly close enough to be considered practically correct.

A closer look at Contracts 1 through 4 shows the ANN achieves price estimates within a small margin of the numerical benchmark. For instance, Contract 1 has a numerical price of 7.2207 (IH) compared to an ANN estimate of 7.2340, a discrepancy of less than 0.2%. Similar tight convergence is observed for Contract 4, where both methods yield prices around 0.19–0.20. Contract 2 demonstrates a slightly more significant difference, but it remains within an acceptable error band given the complexity of barrier features and the underlying NIG dynamics. The ANN reliably reproduces the price for these short to medium-term maturities and relatively frequent barrier monitoring. Furthermore, δ is closely matched, while Γ and ν show some natural variability. This is a known challenge, given that higher-order sensitivities tend to be more sensitive to local curvature effects.

Moreover, we see the same pattern for contracts 5 through 8. We often put this down to ANN data limitations and a lack of complete coverage for more significant discrepancies. Notably, contract 6 is the worst-performing prediction. The mild error discrepancy propagates to the contract sensitivities, where Γ and ν demonstrate divergences. Despite these differences, the ANN still captures the correct sign and order of magnitude, indicating partial success in learning higher dimensionality with respect to option sensitivity patterns.

The key takeaway is the stark difference in CPU times. The numerical benchmark is taken from the beginning of the pricing process to the total Greek calculation, removing standard parameter set-up time. The numerical approach requires an order of 0.26 to 2.50 seconds (or slightly more for high-frequency barrier monitoring), reflecting the cost of discretising the path-dependent payoff. In contrast, the ANN inference stage takes mere milliseconds (0.01 to 0.30 seconds in most reported scenarios), showcasing an order-of-magnitude acceleration. This computational advantage is crucial for real-time risk management and trading desk applications where rapid scenario analyses are paramount. Moreover, once the network is trained, a one-time offline cost, the marginal cost of pricing additional contracts with different parameters is minimal. Of note is that the loading time of the model has been removed. This accurately reflects real-world scenarios, as the model would be in a perpetual loaded state inside CPU memory for pricing.

Contracts 2 and 6, however, highlight the difficulty of ANNs. Even after the process of hyperparameter exhaustive tuning, further calibrations and refinements can be made. This makes the process perpetually one of trial and error, something practitioners may find intolerable.

contract(method)	1(IH)	2(IH)	3(IH)	4(AW)
price	6.9091	3.1716	1.1668	0.2335
δ	0.4655	0.4122	0.5354	1.1988
Γ	0.0057	0.0373	-0.5580	-21.8491
ν	32.9088	223.4726	-41.3224	-68.8552
Θ	1.4085	5.9037	-0.4539	-0.8128
ρ	38.4869	17.9447	10.4559	0.2391
CPU(s)	1.2419	0.2634	0.2576	1.2417

Table 5.10: Numerical valuations of option contracts 1-4 using the variance gamma distribution. This has been produced using the numerical methods discussed in Chapter 4. The fastest calculation was taken as a benchmark, denoted by IH or AW for Feng & Lintesky or Abate & Whitt based Fourier-z.

contract(method)	1	2	3	4
price	6.9101	3.3150	1.1040	0.1945
δ	0.4610	0.4080	0.5641	1.0018
Γ	0.0150	0.0347	-0.5101	-20.2151
ν	33.0154	225.02	-42.0250	-66.0114
Θ	1.6201	5.9820	-2.0215	-1.0250
ρ	45.2515	17.0240	11.0214	0.3548
CPU(s)	0.0055	0.0017	0.3088	0.3122

Table 5.11: Predicted valuations of option contracts 1-4 using the neural network trained on synthetic data produced by the variance gamma distribution. In general, the price and δ are acceptably accurate. The prediction seems to lose accuracy with higher variability. For instance, vega, ν, can vary wildly compared to δ.

contract(method)	5(AW)	6(IH)	7(IH)	8(AW)
price	4.6637	3.0731	1.9408	3.3319
δ	0.1665	0.2770	-0.0965	-0.3868
Γ	-0.0258	0.0082	-0.0189	-0.0141
ν	-670.1712	188.3655	-459.7268	-108.9641
Θ	-4.5947	1.4601	-3.2366	-1.8904
ρ	3.7989	22.5386	-5.8261	-28.6401
CPU(s)	1.2469	2.4701	2.4899	1.2464

Table 5.12: Numerical valuations of option contracts 5–8 using the variance gamma distribution.

contract(method)	5	6	7	8
price	4.6411	3.1007	1.9414	3.3110
δ	0.1324	0.2889	-0.0925	-0.3564
Γ	-0.0247	0.0101	-0.0152	-0.0125
ν	-672.0114	186.1201	-454.0131	-110.244
Θ	-4.2141	1.4215	-3.2658	-0.9910
ρ	3.6244	22.3210	-5.0245	-27.0244
CPU(s)	0.0020	0.1886	0.2985	0.3001

Table 5.13: Predicted valuations of option contracts 5-8 using the neural network trained on synthetic data produced by the normal inverse Gaussian distribution.

The results presented in Tables 5.10 through 5.13 provide a comparative analysis of numerical valuations versus neural network predictions for a set of eight barrier option contracts under the variance gamma (VG) model. The network demonstrates a generally tight alignment with the numerical solutions for the option price and the Greeks for these initial contracts. An example is Contract 1, which exhibits price estimates of 6.9091 (numerical) and 6.9101 (ANN), reflecting an error on the order of 0.01%. Sensitivities such as delta are also closely tracked. However, higher-order Greeks (e.g. Γ, ν) can sometimes show a more pronounced deviation in absolute terms, likely due to the complexity of learning curvature effects in path-dependent VG processes.

The CPU times reinforce the computational advantages of the ANN approach. Numerical valuation under VG typically requires anywhere from 0.26 to 1.24 seconds to accurately capture path dependencies and barrier conditions. In contrast, the neural network's inference time is a few milliseconds (0.0017 to 0.0122 seconds), indicating a remarkable speed-up of two to three orders of magnitude.

As with the normal inverse Gaussian process, disparities are present in higher-order sensitivities. Again, Γ and ν suffer, reflecting the challenge of extrapolating model-specific dynamics. Nevertheless, the ANN has a valuable contribution to practical applications where the order of magnitude of error and the computational performance are key.

5.5 Conclusion

These findings underscore the remarkable potential of neural networks to price complex, exotic derivatives rapidly and with high fidelity to benchmark methods. As seen in our numerical experiments across a variety of barrier option contracts, once a network is properly trained on synthetic data, even under advanced Lévy processes like the variance gamma (VG) or normal inverse Gaussian (NIG) distributions, it can generate accurate prices and Greeks in a fraction of a second.

Nevertheless, while speed and computational efficiency gains are palpable, the benefits are partly contingent on the exact problem setting. In particular, the neural network's advantage becomes more pronounced as the complexity of the contract escalates, for instance, when the number of monitoring dates or the dimensionality of the parameter space grows. Our tables illustrate how CPU times can plummet from over one second per contract to milliseconds once the network has been trained. However, the upfront cost of constructing and training the model, which might involve generating large volumes of synthetic data and tuning hyperparameters, can be substantial and needs to be factored into the overall feasibility of this approach. For instance, a true K-fold or hyperband set-up evaluating combinations of parameters can take days to complete, even for relatively simple cases such as this.

An additional consideration is that networks, although powerful black-box approximators, still rely on representative training data that mirrors the target pricing model. Reliance on synthetic data can also seriously prevent adoption due to the long wait times required to produce thousands of data points.

It is also important to note that while deep learning undoubtedly shines in scenarios where numerical methods incur a considerable runtime penalty, the gap narrows when these conventional methods are highly optimised. In other words, the neural network approach is not a panacea. It is most advantageous in settings where frequent re-pricing under multiple scenarios is required or where the complexity of direct simulation methods becomes prohibitive. When these conditions are met, the small but non-negligible gains in CPU time, on the order of 0.1 seconds to several seconds saved per contract, can accumulate into a substantial advantage for latency-critical applications where numerical accuracy is secondary.

Additionally, the ability to hide model complexity behind a trained black-box approximator remains a powerful proposition for practitioners seeking near-instantaneous valuations. The caveat comes with a trade-off between training overhead, real-time deployment benefits, and the potential for minor errors.

Chapter 6

Further work

This thesis has been primarily interested in pricing exotic options and the integral transforms intrinsic to the process. Many avenues of further research exist within these fields.

6.1 Inverse z-transform

The inverse z-transform is remarkably under-researched compared with continuous transforms such as the Fourier or Laplace transform. As a result, the number of usable numerical approaches to transforms is surprisingly low. At present, research adapts more general methods, such as the inverse discrete Fourier transform or matrix exponentials. Most intriguingly, there is no discrete adaptation for a numerical inverse Laplace method, despite the z-transform being the discrete analogue of the Laplace transform. An avenue of further work would be an attempt to find an accurate and computationally efficient inverse Laplace transform from the vast literature on the topic, which can be adapted as required for the inverse z-transform.

The Cavers' inverse z-transform is remarkably accurate in specific cases where the number of monitoring points is reasonably sized. However, it has a high computational cost, linearly dependent on the number of monitoring points. Chapter 3 employed several series acceleration techniques to remove this linear dependence. In most cases, the acceleration techniques worked as intended, but some accuracy may be lost. Most of the methods within the literature have no concern with a slight accuracy loss, as the more important feature should be the minimisation of computational cost. A series acceleration technique that mathematically ensures minimal accuracy loss would be advantageous in many disciplines as the focus on machine-accurate solutions increases. Additionally, one could apply matrix acceleration techniques to the matrix exponential scheme to improve the computational cost required.

Furthermore, the Cavers' method has the disadvantage of needing careful handling with the parameter r embedded in the process. An extension of the method would be to remove this dependence in some manner to make the technique more stable.

An additional avenue of research would be to explore sinc-based approximations for integral transforms, leveraging the same strengths that sinc functions offer in the Fourier setting.

6.2 Pricing exotic options

The pricing procedure using the z-Spitzer (ZS) approach has improved accuracy and computational cost satisfactorily. The natural extension is to look beyond Lévy models and explore stochastic volatility processes such as Heston and SABR. This approach would give rise to a matrix problem, an additional complication.

Many more exotic options can be considered under the Wiener-Hopf ZS approach. An interesting avenue would be to consider touch options, where a payment is made if a barrier has been touched. Conversely, a no-touch barrier makes a payment if the barrier is not touched within a time window. Additionally, Asian options would be of interest where the path dependency takes the form of an average over the underlying asset's life rather than maximums or minimums. Many exotic options like these are traded regularly within markets, and extensions to handle each contract type should be considered. This would lead to a framework capable of handling many forms of options. This has always been the advantage of Monte Carlo techniques because it is a framework for handling many options.

The formulation of the barrier option pricing problem assumes that the barrier is always active and that the monitoring dates upon which the asset and barrier are checked are discrete. This is not the norm for market conditions, where barriers often operate in time windows. This feature should be built into the method to make it more appealing from a modelling point of view.

6.3 Machine learning approach to option pricing

Generally, within the industry, there is little appetite to integrate neural networks into the methodology for pricing options as a primary method. While many reasons exist, one of the main ones is the black box problem. Many networks have no way of explaining how or why it concluded that it did. In the use case of option pricing, when modelling the prices of a single method as is done in Chapter 5, this is not so much of a problem since we are fitting to a known model. Traditional approaches model market data, which is far more unpredictable regarding structure and cleanliness. As such, many practitioners are unwilling to use a method they cannot explain. This problem has been felt in several industries and explains the drive for explainable AI research in recent years. This would be a welcome addition to the literature in finance and other applications.

Furthermore, the concern exists that the methodology may not outperform straightforward multi-dimensional interpolation. This may be hard to test with the number of parameters chosen in Chapter 5, but it could be investigated using two or three select parameters. One could then fit a three-dimensional spline to find the price and compare this to an optimal neural network, with comparisons to speed and accuracy.

An additional aspect of growing interest in the literature is performing an inverse classification procedure. In this procedure, we take a trained neural network and ask it to identify the stochastic model or pricing process from which a given set of pricing data comes. This is of interest since we would generally be interested in whether the parameters from one pricing model can be translated accurately to the parameters of another.

A further avenue of work involves a systematic approach to calculating optimal hyperparameters. The current crop of methods, K-fold cross-validation, hyperbanding, or Bayesian optimisation, all demand long computing times. Any methodology that reduces this search process to hours rather than days would be a welcome step towards widespread adoption.

One can also begin to look into more informative neural networks, such as physics-informed neural networks (PINNs). These networks decrease the amount of data needed by learning higher-order derivatives of the underlying technique. If applied correctly over the standard ANN model, this can drastically improve training and accuracy.

Chapter 7

Conclusion

This thesis concerns the problem of pricing path-dependent options under the assumption of employing general Lévy processes within a Fourier framework and the associated techniques therein. The methods discussed have a far broader scope than option pricing, which is used as the setting in this thesis. Indeed, the Spitzer identities and resulting requirement for an inverse z-transform have wider applications beyond financial mathematics.

The primary objective of this thesis was to address various problems associated with the Fourier-z pricing methodology, where error floors in the method presented by Fusai et al. [2016] prevented a true machine-accurate solution. Additionally, new frameworks have been presented as an alternative approach to option pricing via machine learning, requiring further exploration and benchmarking against state-of-the-art methods.

A significant body of existing literature regarding the numerical pricing of path-dependent and, more broadly, exotic options now exists. The most relevant and quoted in this thesis are the works by Feng and Linetsky [2008], Fusai et al. [2016], Phelan et al. [2019] and Phelan et al. [2020].

In Chapter 3, we present a full benchmarking of the numerical inverse z-transform using analytical pairs and employing numerous methods found in an extensive literature search. Only one other paper, Horváth et al. [2020], has performed such a benchmarking. This is surprising, considering the numerous works on the inverse Laplace transform.

In Chapter 4, a detailed error analysis of the inverse z-transform presented by Abate and Whitt [1992a] and used in an option pricing framework by Fusai et al. [2016] and extended by Phelan et al. [2020] is presented. An alternative inverse z-transform proposed by Cavers [1978] is then implemented in the pricing framework, achieving a consistent, machine-accurate solution at the cost of increased processing time. This is overcome by implementing a modified Euler series acceleration to fit a non-alternating series resulting from the new inverse z-transform. This analysis was performed upon α-quantile and double barrier options. In the case of the α-quantile option, we also show exponential error convergence for the option's price.

In Chapter 5, attention is turned to implementing a machine learning solution for pricing a double barrier option. Most of the literature surrounding machine learning solutions for option pricing focuses primarily on vanilla options and comparisons with notoriously slow methods, such as Monte Carlo and finite difference methods. We present a machine learning solution that outperforms optimised modern methods when the number of monitoring dates is suitably large, albeit at the expense of extended training time. There are notable problems, however, particularly a lack of explainability, which could continue to be a significant obstacle to using the method in practice.

Bibliography

J. Abate and P. Valkó. Multi-precision Laplace transform inversion. *International Journal for Numerical Methods in Engineering*, 60:979–993, 2004. doi: 10.1002/nme.995.

J. Abate and W. Whitt. Numerical inversion of probability generating functions. *Operations Research Letters*, 12(4):245–251, 1992a. doi: 10.1016/0167-6377(92)90050-D.

J. Abate and W. Whitt. Numerical inversion of probability generating functions. *Operations Research Letters*, 12(4):245–251, 1992b. doi: 10.1016/0167-6377(92)90050-D.

J. Abate and W. Whitt. The Fourier-series method for inverting transforms of probability distributions. *Queueing Systems*, 10:5–87, 1992c. doi: 10.1007/BF01158520.

J. Abate and W. Whitt. Numerical inversion of Laplace transform of probability distributions. *ORSA Journal on Computing*, 7:36–43, 1995. doi: 10.1287/ijoc.7.1.36.

J. Abate and W. Whitt. A unified framework for numerically inverting Laplace transforms. *INFORMS Journal on Computing*, 18(4):407–530, 2006. doi: 10.1287/ijoc.1050.0137.

J. Akahori. Some formulae for a new type of path-dependent option. *The Annals of Applied Probability*, 5(2):383–388, 1995. doi: 10.1214/aoap/1177004769.

C. Albanese, S. Crépey, R. Hoskinson, and B. Saadeddine. XVA analysis from the balance sheet. *Quantitative Finance*, 21(1):99–123, 2021. doi: 10.1080/14697688.2020.1817533.

U. Anders, O. Korn, and C. Schmitt. Improving the pricing of options: a neural network approach. *Journal of Forecasting*, 17(5-6):369–388, 1998. doi: 10.1002/(SICI)1099-131X(1998090)17:5/6⟨369::AID-FOR702⟩3.0.CO;2-S.

P. C. Andreou, C. Charalambous, and S. H. Martzoukos. Generalized parameter functions for option pricing. *Journal of Banking and Finance*, 34(3):633–646, 2010. doi: 10.1016/j.jbankfin.2009.08.027.

A. D. Andricopoulos, M. Widdicks, P. W. Duck, and D. P. Newton. Universal option valuation using quadrature methods. *Journal of Financial Economics*, 67 (3):447–471, 2003. doi: 10.1016/S0304-405X(02)00257-X.

C. Atkinson and G. Fusai. Discrete extrema of Brownian motion and pricing of exotic options. *Journal of Computational Finance*, 10(3):1–43, 2007. doi: 10.21314/JCF.2007.174.

L. Bachelier. Théorie de la spéculation. *Annales scientifiques de l'École Normale Supérieure*, 3e série, 17:21–86, 1900. doi: 10.24033/asens.476.

G. A. Baker and P. Graves-Morris. *Padé Approximants.* Cambridge University Press, 2 edition, 1996. doi: 10.1017/CBO9780511530074.

L. Ballotta. *α-quantile option in a jump-diffusion Economy*, chapter 5. Springer, 2002. doi: 10.1007/978-1-4757-5226-7_5.

O. Barndorff-Nielsen. Processes of normal inverse Gaussian type. *Finance and Stochastics*, 2:41–68, 1998. doi: 10.1007/s007800050032.

G. Baxter. An analytic approach to finite fluctuation problems in probability. *Journal d'Analyse Mathématique*, 9:31–70, 1961. doi: 10.1007/BF02795339.

G. Baxter and M. D. Donsker. On the distribution of the supremum functional for processes with stationary independent increments. *Transactions of the American Mathematical Society*, 85:73–87, 1957. doi: 10.2307/1992962.

C. Bayer, B. Hovarth, A. Muguruza, B. Stemper, and M. Tomas. On deep calibration of (rough) stochastic volatility models. *arXiv preprint arXiv:1908.08806*, 2019. doi: 10.48550/arXiv.1908.08806.

N. Bayer and O. J. Boxma. Wiener-Hopf analysis of an M/G/1 queue with negative customers and of a related class of random walks. *Queueing Systems*, 23:301–316, 1996. doi: 10.1007/BF01206563.

D. R. Beaglehole, P. H. Dybvig, and G. Zhou. Going to extremes: correcting simulation bias in exotic option valuation. *Financial Analysts Journal*, 53(1):62–68, 1997. doi: 10.2469/faj.v53.n1.2057.

A. Beck, S. Becker, P. Cheridito, A. Jentzen, and A. Neufeld. Deep splitting method for parabolic PDEs. *SIAM Journal on Scientific Computing*, 43(5):A3135–A3154, 2021. doi: 10.1137/19M1297919.

F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81:637–654, 1973. doi: 10.1086/260062.

J. P. Boyd. *Chebyshev and Fourier Spectral Methods.* Springer, Heidelberg, 2001. doi: 10.1002/zamm.19910710715.

P. P. Boyle. Options: A monte carlo approach. *Journal of Financial Economics*, 4 (3):323–338, 1977.

P. P. Boyle and Y. Titan. An explicit finite difference approach to the pricing of barrier options. *Applied Mathematical Finance*, 5(1):17–43, 1998. doi: 10.1080/135048698334718.

L. Brancik. Numerical inversion of z-transforms and its application in linear system analysis. pages 7–10, 2003.

M. Broadie, M. P. Glasserman, and S. Kou. A continuity correction for discrete barrier options. *Mathematical Finance*, 7(4):325–349, 1997. doi: 10.1111/1467-9965.00035.

J. W. Brown and R. V. Churchill. *Complex Variables and Applications.* McGraw Hill, 2 Penn Plaza, New York, 2014.

C. C. Brezinski and M. Redivo-Zaglia. *Extrapolation Methods: Theory and Practice.* North-Holland, 1991.

N. Cai, N. Chen, and X. Wan. Occupation times of jump-diffusion processes with double exponential jumps and the pricing of options. *Mathematics of Operations Research*, 35(2):412–437, 2010. doi: 10.1287/moor.1100.0447.

P. Carr and D. Madan. Option valuation using the fast Fourier transform. *Journal of Computational Finance*, 2(4):61–73, 1999. doi: 10.21314/JCF.1999.043.

A. P. Carverhill and T. H. F. Cheuk. Alternative neural network approach for option pricing and hedging. *Available at SSRN 480562*, 2003. doi: 10.2139/ssrn.480562.

J. K. Cavers. On the fast Fourier transform inversion of probability generating functions. *IMA Journal of Applied Mathematics*, 22(3):275–282, 1978. doi: 10.1093/imamat/22.3.275.

D. Chen, H. J. Härkönen, and D. P. Newton. Advancing the universality of quadrature methods to any underlying process for option pricing. *Journal of Financial Economics*, 114(3):600–612, 2014. doi: 10.1016/j.jfineco.2014.07.014.

R. R. Chen and L. Scott. Pricing interest rate options in a two-factor Cox-Ingersoll Ross model of the term structure. *The Review of Financial Studies*, 5(4):613–636, 1992. doi: 10.1093/rfs/5.4.613.

Y. S. Chi and X. Lin. On the threshold dividend strategy for a generalized jump-diffusion risk model. *Insurance: Mathematics and Economics*, 48:326–337, 2011. doi: 10.1016/j.insmatheco.2010.11.006.

C. Chiarella, N. El-Hassan, and A.Kucera. *Evaluation of discrete barrier options in* *a path integral framework*, pages 117–144. Springer, Berlin, Heidelberg, 2008. doi: 10.1007/978-3-540-77958-2_7.

M. Claesen and B. D. Moor. Hyperparameter search in machine learning. *arXiv preprint arXiv:1502.02127*, 2015. doi: 10.48550/arXiv.1502.02127.

A. M. Cohen. *Numerical Methods for Laplace Transform Inversion.* Springer, 233 Spring Street, New York, 2007.

J. W. Cohen. The Wiener-Hopf technique in applied probability. *Perspectives in probability and statistics: papers in honour of M. S. Bartlett on the occasion of his sixty-fifth birthday*, pages 145–156, 1975.

R. Cont and P. Tankov. *Financial Modelling with Jump Processes.* Chapman and Hall/CRC Press, London, 1st edition, 2004.

J. C. Cox, S. A. Ross, and M. Rubinstein. Option pricing: A simplified approach. *Journal of Financial Economics*, 7(3):229–263, 1979.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. doi: 10.1007/BF02551274.

A. Dassios. The distribution of the quantile of a Brownian motion with drift and the pricing of related path-dependent options. *The Annals of Applied Probability*, 5(2):389–398, 1995. doi: 10.1214/aoap/1177004770.

W. B. Davenport. Probability and random processes: An introduction for applied scientists and engineers. 1970.

G. Dimitroff, D. Röder, and C. P. Fries. Volatility model calibration with convolutional neural networks. *Available at SSRN 3252432*, 2018. doi: 10.2139/ssrn.3252432.

K. D. Dingeç and W. Hörmann. A general control variate method for option pricing under Lévy processes. *European Journal of Operational Research*, 221(2):368–377, 2012. doi: 10.1016/j.ejor.2012.03.046.

G. Doetsch. *Introduction to the Theory and Application of the Laplace Transformation.* Springer, New York, 1974.

G. Doetsch. *Introduction to the Theory and Application of the Laplace Transform.* Springer Science and Business Media, Berlin, 2012.

A. Dominguez. Highlights in the history of the Fourier transform. *IEEE Pulse*, 7: 53–61, 2016. doi: 10.1109/MPUL.2015.2498500.

S. Dreiseitl and L. Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics*, 35(5):352–359, 2002. doi: 10.1016/S1532-0464(03)00034-0.

C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia. Incorporating functional knowledge in neural networks. *Journal of Machine Learning Research*, 10:1239–1262, 2009. doi: 10.1109/ICNN.1997.611676.

A. Erdélyi. *Tables of Integral Transforms.* McGraw-Hill, 1954.

J. H. K. F. R. de Hoog and A. N. Stokes. An improved method for numerical inversion of Laplace transforms. *SIAM Journal on Scientific and Statistical Computing*, 3(3), 1982. doi: 10.1137/0903022.

F. Fang and C. W. Oosterlee. A novel pricing method for European option based on the Fourier-cosine series expansions. *SIAM Journal on Scientific Computing*, 31(2):826–848, 2008. doi: 10.1137/080718061.

F. Fang and C. W. Oosterlee. Pricing early-exercise and discrete barrier options by Fourier-cosine series expansions. *Numerische Mathematik*, 114(1):27–62, 2009. doi: 10.1007/s00211-009-0252-4.

F. Fang and C. W. Oosterlee. A Fourier-based valuation method for Bermudan and barrier options under Heston's model. *SIAM Journal on Financial Mathematics*, 2(1):439–463, 2011. doi: 10.1137/100794158.

L. Feng and V. Linetsky. Pricing discretely monitored barrier options and default-able bonds in Lévy process models: a Hilbert transform approach. *Mathematical Finance*, 18(3):337–384, 2008. doi: 10.1111/j.1467-9965.2008.00338.x.

L. Feng and V. Linetsky. Computing exponential moments of the discrete maximum of a Lévy process and lookback options. *Finance and Stochastics*, 13(4):501–529, 2009. doi: 10.1007/s00780-009-0096-x.

S. Figlewski and B. Gao. The adapative mesh model: a new approach to efficient option pricing. *Journal of Financial Economics*, 53(3):313–351, 1999. doi: 10.1016/S0304-405X(99)00024-0.

M. Frigo and S. Johnson. The fastest Fourier transform in the West. Technical report, Massachusetts Inst of Tech Cambridge, 1997.

M. Frigo and S. Johnson. FFTW: An adaptive software architecture for the FFT. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 1381–1384. IEEE, 1998. doi: 10.1109/ICASSP.1998.681704.

G. Fusai and M. C. Recchioni. Analysis of quadrature methods for pricing discrete barrier options. *Journal of Economic Dynamics and Control*, 31(3):826–860, 2007. doi: 10.1016/j.jedc.2006.03.002.

G. Fusai, I. D. Abrahams, and C. Sgarra. An exact analytical solution for discrete barrier options. *Finance and Stochastics*, 10:1–26, 2006. doi: 10.1007/s00780-005-0170-y.

G. Fusai, D. Marazzina, M. Marena, and M. Ng. Z-transform and preconditioning techniques for option pricing. *Quantitative Finance*, 12(9):1381–1394, 2012. doi: 10.1080/14697688.2010.538074.

G. Fusai, G. Germano, and D. Marazzina. Spitzer identity, Wiener-Hopf factorization and pricing of discretely monitored exotic options. *European Journal of Operational Research*, 251(1):124–134, 2016. doi: 10.1016/j.ejor.2015.11.027.

R. Garcia and R. Gençay. Option pricing with neural networks and a homogeneity hint. *Decision Technologies for Computational Finance*, 2:195–205, 1998. doi: 10.1007/978-1-4615-5625-1_15.

H. Geman, D. B. Madan, and M. Yor. Time changes for Lévy processes. *Mathematical Finance*, 11:79–96, 2001. doi: 10.1111/1467-9965.00108.

G. Germano, C. Phelan, D. Marazzina, and G. Fusai. Solution of Weiner-Hopf and Fredholm integral equations by fast Hilbert and Fourier transforms. *arXiv:2106.05326*, 2018. doi: 10.48550/arXiv.2106.05326.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL `https://proceedings.mlr.press/v9/glorot10a.html`.

A. L. Golbabai, V. Ballestra, and D. Ahmadian. A highly accurate finite element method to price discrete double barrier options. *Computational Economics*, 44: 153–173, 2014. doi: 10.1007/s10614-013-9388-5.

B. M. Goldman, H. B. Sosin, and M. A. Gatto. Path dependent options: "buy at the low, sell at the high". *The Journal of Finance*, 34:1111–1127, 1979. doi: 10.2307/2327238.

R. Green. *Application of the Wiener-Hopf technique to derivative pricing.* PhD thesis, 2009.

R. Green, D. I. Abrahams, and G. Fusai. Pricing financial claims contingent upon an underlying asset monitored at discrete times. *Journal of Engineering Mathematics*, 59:373–384, 2007. doi: 10.1007/s10665-007-9176-0.

R. Green, G. Fusai, and D. I. Abrahams. The Wiener-Hopf technique and discretely monitored path-dependent option pricing. *Mathematical Finance*, 20(2):259–288, 2010. doi: 10.1111/j.1467-9965.2010.00397.x.

P. S. Hagan, D. D. Kumar, A. S. Lesiewski, and D. E. Woodward. Managing smile risk. *The Best of Wilmott*, 1:249–296, 2002.

J. T. Hahn. Option pricing using artificial neural networks: an australian perspective, 2013.

I. Halperin. Qlbs: Q-learner in the black-scholes (-merton) worlds. *The Journal of Derivatives*, 28(1):99–122, 2020. doi: 10.3905/jod.2020.1.108.

J. Han, A. Jentzen, and E. Weinan. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018. doi: 10.1073/pnas.1718942115.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015. doi: 10.1109/ICCV.2015.123.

A. Hernandez. Model calibration with neural networks. *Available at SSRN 2812140*, 2016. doi: 10.2139/ssrn.2812140.

S. L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The Review of Financial Studies*, 6: 327–343, 1993. doi: 10.1093/rfs/6.2.327.

E. Hoffer, I. Hubara, and D. Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *arXiv preprint arXiv:1705.08741*, 2017. doi: 10.48550/arXiv.1705.08741.

T. Honsono. Numerical inversion of the Laplace transform and some applications to wave optics. *Radio Science*, 16:1015–1019, 1981. doi: 10.1002/EEJ.4390990506.

B. Horvath, A. Muguruza, and M. Tomas. Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. *Quantitative Finance*, 21(1):11–27, 2021. doi: 10.1080/14697688.2020.1817974.

I. Horváth, A. Mészáros, and M. Telek. Numerical inverse transformation methods for z-transform. *MDPI Mathematics*, 8(4), 2020. doi: 10.3390/math8040556.

J. C. Hull. *Options, Futures, and Other Derivatives.* Pearson, 10th edition, 2017.

A. Hurewicz. *Lectures on Ordinary Differential Equations.* Cambridge Technology Press of the Massachusetts Institute of Technology, Cambridge, Massachusetts, 1958.

J. M. Hutchinson and T. Poggio. A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, 49(3):851–889, 1994. doi: 10.1111/j.1540-6261.1994.tb00081.x.

D. S. Jones. A simplifying technique in the solution of a class of diffraction problems. *The Quarterly Journal of Mathematics*, 3(1):189–196, 1952. doi: 10.1093/qmath/3.1.189.

E. Jury. *Theory and Application of the Z-transform Method.* Wiley, 1964.

J. H. B. Kemperman. A Wiener-Hopf type method for a general random walk with a two-sided boundary. *Annals of Mathematical Statistics*, 34:1168–1193, 1963. doi: 10.1214/aoms/1177703855.

K.Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. doi: 10.1016/0893-6080(89)90020-8.

F. W. King. *Hilbert Transforms.* Cambridge University Press, Cambridge, 1st edition, 2009. doi: 10.1017/CBO9780511721458.

D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer. Machine learning-accelerated computational fluid dynamics. *PROCEEDINGS OF THE NATIONAL ACADEMY OF SCIENCES OF THE UNITED STATES OF AMERICA*, 118(21), 2021. doi: 10.1073/pnas.2101784118.

S. G. Kou. A jump-diffusion model for option pricing. *Management Science*, 48(8): 1086–1101, 2002. doi: 10.1287/mnsc.48.8.1086.166.

S. G. Kou. Chapter 8 discrete barrier and lookback options. *Handbooks in Operations Research and Management Science*, 15:343–373, 2007. doi: 10.1016/S0927-0507(07)15008-8.

E. Kreyszig. *Advanced Engineering Mathematics*. Wiley, New York, 10th edition, 2011.

P. Kumar, S. Batra, and B. Raman. Deep neural network hyper-parameter tuning through twofold genetic approach. *Soft Computing*, 25:8747–8771, 2021. doi: 10.1007/s00500-021-05770-w.

N. Kunitomo and M. Ikeda. Pricing options with curved boundaries. *Mathematical Finance*, 2:275–298, 1992. doi: 10.1111/j.1467-9965.1992.tb00033.x.

P. R. Lajbcygier and J. T. Connor. Improved option pricing using artificial neural networks and bootstrap methods. *International Journal of Neural Systems*, 8(4): 457–471, 1997. doi: 10.1142/S0129065797000446.

J. B. Lawrie and I. D. Abrahams. A brief historical perspective of the Wiener-Hopf technique. *Journal of Engineering Mathematics*, 59:351–358, 2007. doi: 10.1007/s10665-007-9195-x.

D. Levin. Development of non-linear transformations for improving convergence of sequences. *International Journal of Computer Mathematics*, 3:371–388, 1973. doi: 10.1080/00207167308803075.

G. Lian, S. Zhu, R. J. Elliott, and Z. Cui. Semi-analytical valuation for discrete barrier options under time-dependent Lévy processes. *Journal of Banking and Finance*, 75:167–183, 2017. doi: 10.1016/j.jbankfin.2016.11.012.

S. Liu, A. Borovykh, L. A. Grzelak, and C. W. Oosterlee. A neural network-based framework for financial model calibration. *Journal of Mathematics in Industry*, 9 (9), 2019a. doi: 10.1186/s13362-019-0066-7.

S. Liu, C. W. Oosterlee, and S. M. Bohte. Pricing options and computing implied volatilities using neural networks. *Risks*, 7(1), 2019b. doi: 10.3390/risks7010016.

R. Lord, F. Fang, F. Bervoets, and C. W. Oosterlee. A fast and accurate FFT-based method for pricing early-exercise options under Lévy processes. *SIAM Journal on Scientific Computing*, 30(4):1678–1705, 2008. doi: 10.1137/070683878.

J. Lu and H. Ohta. A data and digital-contracts driven method for pricing complex derivatives. *Quantitative Finance*, 3(3):212–219, 2003a. doi: 10.1088/1469-7688/3/3/307.

J. Lu and H. Ohta. Digital contracts-driven method for pricing complex derivatives. *Journal of the Operational Research Society*, 54(9):1002–1010, 2003b. doi: 10.1057/palgrave.jors.2601597.

D. B. Madan and M. Yor. Representing the CGMY and Meixner Lévy processes as time changed Brownian motion. *Journal of Computational Finance*, 12:27–47, 2008.

D. B. Madan, P. P. Carr, and E. C. Chang. The variance gamma process and option pricing. *Review of Finance*, 2:79–105, 1998. doi: 10.1023/A:1009703431535.

M. Malliaris and L. Salchenberger. A neural network model for estimating option prices. *Applied Intelligence*, 3:193–206, 1993. doi: 10.1007/BF00871937.

W. A. McGhee. An artificial neural network representation of the SABR stochastic volatility model. *Journal of Computational Finance*, 25(3):1–27, 2021. doi: 10.21314/JCF.2021.007.

F. Merrikh-Bayat. Two methods for numerical inversion of the z-transform. *arXiv preprint arXiv:1409.1727*, 2014. doi: 10.48550/arXiv.1409.1727.

R. C. Merton. Theory of rational option pricing. *The Bell Journal of Economics and Management Science*, 4:141–183, 1973. doi: 10.2307/3003143.

R. C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3:125–144, 1976. doi: 10.1016/0304-405X(76)90022-2.

P. L. Mills. Numerical inversion of z-transform with application to polymerisation kinetics. *Computers and Chemistry*, 11(2):137–151, 1987. doi: 10.1016/0097-8485(87)80032-3.

R. Miura. A note on look-back options based on order statistics. *Hitotsubashi Journal of Commerce and Management*, 27(1):15–28, 1992. doi: 10.15057/5748.

C. A. O'Cinneide. Euler summation for Fourier series and Laplace transform inversion. *Stochastic Models*, 13(2):315–337, 1997. doi: 10.1080/15326349708807429.

A. Oppenheim. *Discrete Time Signal Processing.* Pearson Higher Education, New Jersey, 3rd edition, 2010.

A. Oppenheim and R. Schafer. *Digital Signal Processing.* Prentice-Hall, 1975.

H. Padé. Sur la représentation approchée d'une fonction par des fractions rationnelles. *Annales scientifiques de l'École Normale Supérieure*, 9:3–93, 1892.

R. E. A. C. Paley and N. Wiener. Notes on the theory and application of $f$ourier transforms. iii, iv, v, vi, vii. *Transactions of the American Mathematical Society*, 35:761–791, 1933. doi: 10.1090/S0002-9947-1933-1501716-9.

A. Papoulis. Numerical inversion of the z-transform. *IEEE Transactions on Circuit Theory*, 20(4):419–420, 1973. doi: 10.1109/TCT.1973.1083713.

C. E. Phelan, D. Marazzina, G. Fusai, and G. Germano. Hilbert transform, spectral filters and option pricing. *Annals of Operations Research*, 282:273–298, 2019. doi: 10.1007/s10479-018-2881-4.

C. E. Phelan, D. Marrazzina, and G. Germano. Pricing methods for alpha-quantile and perpetual early exercise options based on Spitzer identities. *Quantitative Finance*, 20(6):899–918, 2020. doi: 10.1080/14697688.2020.1718192.

A. D. Polyanin and A. V. Manzhirov. *Handbook of Integral Equations.* CRC Press, Boca Raton, 1st edition, 1998.

S. C. Port. An elementary probability approach to fluctuation theory. *Journal of Mathematical Analysis and Applications*, 6(1):109–151, 1963. doi: 10.1016/0022-247X(63)90097-0.

W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing.* Cambridge University Press, New York, 3rd edition, 2007.

P. M. Rajković, M. S. Stanković, and S. D. Markinković. A method for numerical evaluating of inverse Z-transform. *Facta universitatis-series: Mechanics, Automatic Control and Robotics*, 4(16):133–139, 2004.

P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. doi: 10.48550/arXiv.1710.05941.

L. F. Richardson. On the approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Royal Society*, 83(563), 1910. doi: 10.1098/rspa.1910.0020.

L. C. G. Rogers. Fluid models in queueing theory and Wiener-Hopf factorization of Markov chains. *Annals of Applied Probability*, 4:390–413, 1994. doi: 10.1214/aoap/1177005065.

J. Ruf and W. Wang. Neural networks for option pricing and hedging: a literature review. *Journal of Computational Finance*, 24(1):1–46, 2020. doi: 10.21314/JCF.2020.390.

M. J. Ruijter, M. Versteegh, and C. W. Oosterlee. On the application of spectral filters in a Fourier option pricing technique. *Journal of Computational Finance*, 19(1):75–106, 2015. doi: 10.21314/JCF.2015.306.

D. Rumelhart, G. Hinton, and R. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986. doi: 10.1038/323533a0.

R. Schmidt. On the numerical solution of linear simultaneous equations by an iterative method. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 32(214):369–383, 1941. doi: 10.1080/14786444108520797.

W. Schoutens. *Lévy Process in Finance: Picing Financial Derivatives.* Wiley, New York, 2003. doi: 10.1002/0470870230.

D. Shanks. Non-linear transformations of divergent and slowly convergent sequences. *Journal of Mathematics and Physics*, 34:1–42, 1955. doi: 10.1002/sapm19553411.

A. Sidi. Convergence properties of some nonlinear sequence transformations. *Mathematics of Computation*, 33:315–326, 1979. doi: 10.1090/S0025-5718-1979-0514827-6.

A. Sidi. *Practical Extrapolation Methods: Theory and Applications.* Cambridge University Press, 2003. doi: 10.1017/CBO9780511546815.

D. A. Smith and W. F. Ford. Acceleration of linear and logarithmic convergence. *SIAM Journal on Numerical Analysis*, 16(2):223–240, 1979. doi: 10.1137/0716017.

A. Sobhani. A numerical method for pricing discrete barrier option by CAS wavelet. *The 51st Annual Iranian Mathematics Conference*, 2021.

A. Sobhani and M. Milev. A numerical method for pricing discrete double barrier option by Legendre multiwavelet. *Journal of Computational and Applied Mathematics*, 328:355–364, 2018. doi: 10.1016/j.cam.2017.07.033.

H. V. Sorensen and C. S. Burrus. Efficient computation of the DFT with only a subset of input or output points. *IEEE Transactions on Signal Processing*, 41(3): 1184–1200, 1993. doi: 10.1109/78.205723.

F. Spitzer. The Wiener-Hopf equation whose kernel is a probability density. *Duke Mathematical Journal*, 24(3):327–343, 1957. doi: 10.1215/S0012-7094-57-02439-0.

H. Stahl. The convergence of padé approximants to functions with branch points. *Journal of Approximation Theory*, 91:139–204, 1997. doi: 10.1006/JATH.1997.3141.

H. Stehfest. Algorithm 368: Numerical inversion of Laplace transforms. *Communications of the ACM*, 13(1):47–49, 1970. doi: 10.1145/361953.361969.

M. Steiner and M. Wallmeir. Pricing near the barrier: the case of discrete knock-out options. *Journal of Computational Economics*, 3(1):69–90, 1999. doi: 10.21314/JCF.1999.038.

F. Stenger. *Handbook of Sinc Numerical Methods.* CRC Press, Boca Raton, 2011. doi: 10.1201/b10375.

H. Stone. Calibrating rough volatility models: a convolutional neural network approach. *Quantitative Finance*, 20(3):379–392, 2019. doi: 10.1080/14697688.2019.1654126.

A. Tagliani. Inverse z transform and moment problem. *Probability in the Engineering and Informational Sciences*, 14(3):393–404, 2000. doi: 10.1017/S0269964800143098.

A. Talbot. The accurate numerical inversion of Laplace transforms. *IMA Journal of Applied Mathematics*, 23(1):97–120, 1979. doi: 10.1093/imamat/23.1.97.

E. C. Titchmarsh. *Introduction to the Theory of Fourier Integrals.* Oxford at the Clarendon Press, Oxford, England, 1948.

P. Valkó and J. Abate. Comparison of sequence accelerators for the Gaver method of numerical Laplace transform inversion. *Computers & Mathematics with Applications*, 48(3):629–636, 2004. ISSN 0898-1221. doi: 10.1016/j.camwa.2002.10.017.

J. A. C. Weideman. Optimizing Talbot's contours for the inversion of the Laplace transform. *SIAM Journal of Numerical Analysis*, 44(6):2342–2362, 2006. doi: 10.1137/050625837.

J. G. Wendel. Order statistics of partial sums. *The Annals of Mathematical Statistics*, 31(4):1034–1044, 1960. doi: 10.1214/aoms/1177705676.

E. J. Weniger. Nonlinear sequence transformations for the acceleration of convergence and the summation of divergent series. *Computer Physics Reports*, 10(5): 189–371, 1989. doi: 10.1016/0167-7977(89)90011-7.

H. White. Economic prediction using neural networks: The case of IBM daily stock returns. *IEEE 1988 International Conference on Neural Networks*, pages 451–458 vol.2, 1988. doi: 10.1109/ICNN.1988.23959.

H. White. Some asymptotic results for learning in single hidden-layer feedforward network models. *Journal of the American Statistical Association*, 84(408):1003–1013, 1989. doi: 10.2307/2290076.

D. V. Widder. What is the Laplace transform? *The American Mathematical Monthly*, 52(8):419–425, 1945. doi: 10.1080/00029890.1945.11991601.

D. V. Widder. *The Laplace Transform.* Princeton University Press, Princeton, 1946.

N. Wiener and E. Hopf. Über eine Klasse singulärer integralgleichungen. *Sem–Ber Preuss Akad Wiss*, 31:696–706, 1931.

J. Wimp. Sequence transformations and their applications. 1981.

P. Wynn. On a Procrustean technique for the numerical transformation of slowly convergent sequences and series. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52(4):663–671, 1956. doi: 10.1017/S030500410003173X.

T. Ye and L. Zhang. Derivatives pricing via machine learning. *Journal of Mathematical Finance*, 9:561–589, 2019. doi: 10.4236/jmf.2019.93029.

M. Yor. The distribution of Brownian quantiles. *Journal of Applied Probability*, 32 (2):405–416, 1995. doi: 10.2307/3215296.

M. Zhang, H. Li, S. Pan, J. Lyu, S. Ling, and S. Su. Convolutional neural networks-based lung nodule classification: A surrogate-assisted evolutionary algorithm for hyperparameter optimization. *IEEE Transactions on Evolutionary Computation*, 25(5):869–882, 2021. doi: 10.1109/TEVC.2021.3060833.

C. Y. Zong. Optimisation of the parameters of the Euler summation rule and its application in the Fourier-z pricing method for barrier options and other path-dependent derivatives. Master's thesis, UCL, 2020.

Appendix A - Illustrative figures

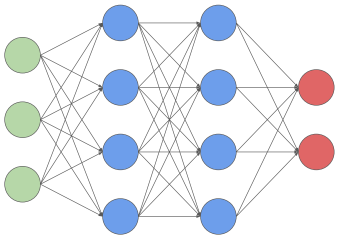

Figure 7.1: Example ANN construction using one output layer in red, two hidden layers in blue and an output layer in red.

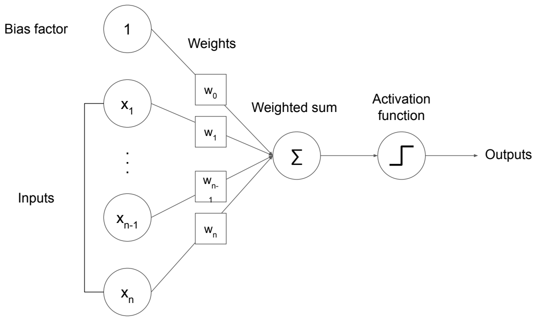

Figure 7.2: Example neuron learning process with bias factor.

Appendix B - Example contract parameters

Contract Number	1	2	3	4	5	6	7	8
Maturity (years)	1	0.5	2	1	1.5	1	0.5	2
Monitoring dates	52	52	52	256	256	514	512	52
Risk free rate	0.01	0.02	0.01	0.01	0.02	0.05	0.05	0.01
Volatility	0.02	0.02	0.03	0.01	0.02	0.015	0.02	0.02
Spot	100.0	95.0	90.0	105.0	110.0	100.0	100.0	150.0
Strike	100.0	100.0	100.0	100.0	100.0	110.0	90.0	100.0
Lower barrier	85.0	80.0	90.0	105.0	100.0	90.0	85.0	90.0
Upper barrier	150.0	160.0	140.0	120.0	140.0	150.0	110.0	155.0

Table 7.1: Parameters for contracts under evaluation during deep learning testing.

Appendix C - Process parameters

Process	Characteristic function $\Phi(\xi,t) = e^{-t\psi(\xi)}$
Normal inverse Gaussian	$e^{-t\sqrt{\alpha^2-i\beta\xi^2}-\sqrt{\alpha^2-\beta^2}}$
Variance gamma	$(1 - i\nu\theta\xi + \frac{1}{2}\nu\sigma^2\xi^2)^{-t/\nu}$
Kou double exponential	$e^{-t\left(-\frac{1}{2}\sigma^2\xi^2+\lambda((1-p)\eta_2/(\eta_2+i\xi)+p\eta_1/(\eta_1-i\xi)-1)\right)}$

Table 7.2: Example of parameters used in numerical tests where $\Phi(\xi, t)$ is the characteristic function.

Process	Values
Normal inverse Gaussian	$\alpha = 15, \beta = -5, \delta = 0.5$
Variance gamma	$\theta = \frac{1}{9}$
Kou double exponential	$s = 0.1, \lambda = 3, p = 0.3, \eta_1 = 40, \eta_2 = 12$

Table 7.3: Example of parameters used in numerical tests where $\Phi(\xi, t)$ is the characteristic function.

Appendix D - Hyper-parameter optimisation algorithm

Algorithm 1 Hyper-parameters optimizations with K-fold cross-validation

```
procedure HYPER-PARAMETERS OPTIMIZATIONS(training samples: $X_t$, $y_t$)
    Split the training samples into $K$ different subsets
    for each combination of hyperparameters do
        for each different $K - 1$ subsets do
            for each epoch do
                for each batch do
                    Update internal parameters with samples in the batch
                end for
            end for
            Calculate the metric by evaluating on the remaining (validation) subset
        end for
        Averaging over $K$ cases to obtain the final metric
    end for
    Compare and rank the final metrics for all combinations of the hyperparameters
    return the best combination of hyper-parameters
end procedure
```

Appendix E - Deep learning model structures

Parameter	Result
Model - Hidden layers	4
Model - Batch size	512
Model - Activation	ReLu
Model - Early Stopping Patience	6
Layer 1 - node count	140
Layer 1 - dropout rate	0.1
Layer 1 - regularisation	L1L2
Layer 1 - regularisation value	0.001
Layer 2 - node count	400
Layer 2 - dropout rate	0.1
Layer 2 - regularisation	L1
Layer 2 - regularisation value	0.00001
Layer 3 - node count	60
Layer 3 - dropout rate	0.1
Layer 3 - regularisation	L1
Layer 3 - regularisation value	0
Layer 4 - node count	140
Layer 4 - dropout rate	0.2
Layer 4 - regularisation	L2
Layer 4 - regularisation value	0.0001
Layer Price - node count	32
Layer δ/Γ - node count	64
Layer $\nu/\Theta/\rho$ - node count	64

Table 7.4: Optimal model for the normal inverse Gaussian distribution obtained after hyper-parameter K-folds search.

Parameter	Result
Model - Hidden layers	4
Model - Batch size	512
Model - Activation	Glorot Uniform
Model - Early Stopping Patience	7
Layer 1 - node count	420
Layer 1 - dropout rate	0.4
Layer 1 - regularisation	None
Layer 1 - regularisation value	
Layer 2 - node count	160
Layer 2 - dropout rate	0.1
Layer 2 - regularisation	L1L2
Layer 2 - regularisation value	0.00001
Layer 3 - node count	700
Layer 3 - dropout rate	0.2
Layer 3 - regularisation	L1
Layer 3 - regularisation value	0.01
Layer 4 - node count	760
Layer 4 - dropout rate	0.2
Layer 4 - regularisation	None
Layer 4 - regularisation value	
Layer Price - node count	32
Layer δ/Γ - node count	64
Layer $\nu/\Theta/\rho$ - node count	128

Table 7.5: Optimal model for the variance gamma distribution obtained after hyperparameter K-folds search.