2023

A Dynamic parallel algorithm for derivatives pricing and hedging on GPU-CPU heterogeneous systems.

Muganda, Brian Wesley
Strathmore Institute of Mathematical Sciences
Strathmore University

A Dynamic Parallel Algorithm for Derivatives Pricing and Hedging on GPU-CPU Heterogeneous Systems

By

BRIAN WESLEY MUGANDA

72559

Doctor of Philosophy in Mathematical Sciences

July 2023

A Dynamic Parallel Algorithm for Derivatives Pricing and Hedging on GPU-CPU Heterogeneous Systems

By

BRIAN WESLEY MUGANDA

72559

A Thesis Submitted to the Strathmore Institute of Mathematical Sciences in partial fulfillment of the requirements for the award of Doctor of Philosophy in Mathematical Sciences.

Strathmore Institute of Mathematical Sciences

Strathmore University

July 2023

Declaration and Approval

I, the undersigned, declare that this project is my original work and has not been submitted for registration and examination in any other institution.

Brian Wesley Muganda

Sign:  **Date: 08/06/2023**

Approval

This research project has been submitted for examination with my approval as the university supervisor

Dr. Bernard Shibwabo Kasamani

Senior Lecturer in Computing,

School of Computing and Engineering Sciences,

Strathmore University

Dr. Ioannis Kyriakou

Reader in Actuarial Finance,

Faculty of Actuarial Science & Insurance,

Bayes Business School (formerly Cass), City, University of London

Dr. Godfrey Madigu,

Dean, Institute of Mathematical Sciences,

Strathmore University

Dr. Bernard Shibwabo,

Director of Graduate Studies,

Strathmore University

Table of Contents

List of Figures

List of Tables

Abstract

The use of artificial intelligence in the financial services industry has the potential to transform the sector through greater efficiencies, cost reductions and better tools to draw intelligence from large datasets. The access to computing power which is scalable, accurate and reliable has consequently become a major requirement for the industry due to increased competition, increased products and complexity in models, increased volume of data, stricter regulatory environment and desire for competitive advantage. In this regard, this research provides methodological solutions that would result in accurate and fast system throughput, cost saving and speed acceleration for a financial institution's financial engineering system by adoption of heterogeneous CPU-GPU parallel architecture with algorithms which are freshly created by drafting from dynamic copula framework for option pricing. This price estimation of options and the assessment of their risk sensitivities under stochastic dynamics namely: stochastic interest rate, stochastic volatility and jumps for varying strikes, maturity and asset classes is a computationally intensive task given the complex nature of the pricing methodologies applied. Models that are much more fully analytical and less complex for pricing derivatives under stochastic dynamics are desirable for much more accurate pricing, investment portfolio construction and risk analysis; and with it an associated system prototype that would provide real-time results. This thesis formulated dynamic parallel algorithms for derivative security pricing and hedging on GPU-CPU heterogeneous systems. This was achieved through the design and implementation of a real-time derivative pricing system prototype supported by a parallel and distributed architecture. The parallel architecture was implemented using hybrid parallel programming on CPU and GPU in OpenCL C, Python and R to provide computational acceleration. The GPU implementation resulted to a peak speed acceleration of 541 times by reducing compute time from 46.24 minutes to 5.12 seconds with the dynamic models under stochastic volatility and stochastic interest rates improving pricing accuracy by an aggregate of 46.68% over the Black-Scholes framework. This adopted approach in this thesis is of practical importance in the harnessing idle processor power, reducing the financial institution's computational resources requirements and provision of accurate and real-time results necessary in trading, hedging, risk assessment and portfolio optimization processes.

Keywords: *dynamic copula, empirical dependence, stochastic volatility, stochastic interest rates, jumps, hybrid GPU acceleration, parallelism*

Acknowledgement

To my supervisors Dr. Bernard Shibwabo and Prof. Ioannis Kyriakou, I would like to appreciate your support through my doctoral studies. Without your insights, guidance and encouragement, this work would not be possible. From the initial ideas to the final documents and papers, I have really benefited from your advice and comments. I have also learnt a lot from both of you and drawn inspirations to achieve greater. I owe you an immense debt of gratitude. Thank you. It has been a great pleasure and a very rewarding experience to work with you.

My most sincere thanks also go to the Dean and Research Director of SIMS, Dr. Godfrey Madigu and Dr. Mary Achieng respectively. I thank you for the administrative support and encouragement that I have received from you. I am also thankful for the comments and suggestions I received from members of defense committees, conference participants and journal referees. The suggestions and input you provided has been highly valuable.

I am very grateful for the financial support from DAAD in the In-Country Scholarship Programme. This support has immensely facilitated the production of this work. Thank you.

Chapter 1: Introduction

1.1 Background

1.1.1 Derivatives Pricing and Parallelism

Financial instruments traded on financial markets are essentially contractual agreements between two parties that involve the calculation and delivery of quantities of monetary currency or its economic equivalent. These financial instruments include stocks, bonds and derivatives such as options, which is a financial instrument that gives the buyer the right but not obligation to buy or sell an underlying asset such as stock at an agreed price in the future. When the buyer is allowed to exercise this right at any time before or on expiration, it is an American option, if they are allowed only to exercise this right at maturity it is a European option. Market participants use these options for hedging, arbitrage and speculation (Hull, 2015).

Complexities and novelties have recently arisen in the definitions and calculation of the options payoffs and the theoretical considerations used to price these options, some deviating from the assumptions put forth by Black and Scholes (1973) for pricing options. As much as the development of log-normal model of Black and Scholes was quite successful in demonstrating not only a prescient framework for derivative pricing, but also the importance of tractable models in the practical application of risk-neutral pricing theory, at a time when computational facilities were primitive by modern standards. The assumptions postulated have been established not to hold in practice. Developments since then in quantitative finance have been accompanied by relaxation of some of these assumptions and by simultaneous advancement in computing techniques and computing power, and this has opened the door to alternative computational methods such as Monte Carlo, PDE discretization, and Fourier methods, which have greatly increased the ability to price derivatives with complex payoffs and optionality (Rosen et al., 2016).

Significant progress, especially in risk pricing, has been made due to the development of these mathematical models and the use of state-of-the-art workstations for their solution. Some of the more complex risk pricing models would exhaust the computer capabilities of workstations, and analysts then resort to the use of clusters of workstations. We would also see an occasional use of massively parallel or high-performance computers for the prototyping and testing of some pricing model, before the model would be deployed in practice on a cluster of workstations (Zenios, 1999).

1.1.2 Real-time Derivatives Pricing Systems

In a report on high performance computing in finance, Vynckier (2015) opines that the real benchmark for calling a valuation method "efficient" is the ability to perform calibration to market, valuation, risk sensitivities and implement stress tests, back tests and performance tests for all strikes and all maturities on the entire trading book in real-time, enabling traders and risk managers need to make proper investment and hedging decisions in real-time.

The choice of inputs into the pricing models will affect quoted option prices and consequently investment or hedging decisions made. The accuracy of pricing models and a reliable computing system are essential for real-time decision making. This quick real-time decision making could present a profit potential of investment and strategies adopted, consequently improving quotes and position taken up. Having a real-time system and accurate pricing models that capture true market dynamics is a source of competitive advantage for an investor of trading company.

1.1.3 Performance Improvements by Parallelism

The access to computing power that is scalable, accurate and reliable has quickly become a major requirement for the financial services industry due to increased competition, increased products and complexity in models, increased volume of data, and a stricter regulatory environment (Harris, 2008). Regulation authorities' requirements amplify the need for fast and reliable computing capacities by imposing higher standards for availability, system integrity and security on financial institutions. At the same time, financial institutions feel the pressure to cut development times and costs for the introduction of new products and models (Basermann et al., 2008).

The need to improve business efficiency and cut costs has become more important than ever before. By implementing parallel and distributed architecture for their front or back offices, they are able to achieve a great deal of flexibility and reliability and increase the performance-to-cost ratios for the institution. Since they will have a system of shared workstations that can solve a large complex task in a fraction of time that a single computer would, and thus the need to have constant upgrading to large computing hardware eliminated. Previously large tasks that require constant recomputation can be accelerated by harnessing parallel architectures which will enable achieving the low latency requirements of the institutions' computationally extensive and expensive problems, solving them within microseconds (Basermann et al., 2008).

1.2 Problem Statement

The famous Black-Scholes (1973) model in their assumptions asserts the existence of a flat implied volatility surface profile. It has been empirically observed however that this surface exhibits dependence on maturity and strike price at a given cross-section and time-series. This empirical implied volatility surface has thus been noted to exhibit stylized characteristics of 'smile', 'skew' or 'smirk' which changes over time as a result of options market dynamics. Empirical studies on the behavior of implied volatilities have pointed to some commonality of these characteristics across markets particularly on the smile patterns, the term structure and the time series pattern which displays high autocorrelation and mean-reverting behavior. Further, the stylized facts of asset market returns have indicated the presence of non-constant volatility which is corroborated by volatility clustering and conditional heavy tails, correlations asymmetry in stress market conditions, leverage effects of negative correlation of asset return and volatility, and the underlying tail dependence with gain-loss asymmetry where downsize movements are disproportionately larger than to upward movements. Consequently, the assumption of asset return normality is violated in practice given these heavy tails and skewness of return distribution (Bakshi, 2010; Cont, 2001; Cont, Da Fonseca, 2002; Dumas, Fleming & Whaley, 1998; Alexander, 2001; Wallmeier & Hafner, 2000; Fouque, Sircar & Papanicolaou, 1999: Bates, 2001).

Previous approaches that have been applied to capture these stylized facts of financial markets into the pricing of options have been semi-analytical approximations, simulations or fully numerical implementations. The Fourier approach is semi-analytical and involves use of characteristic function inversion, the partial differential equations approach requires numerical approximation by finite differences which however suffer numerical problems of dimensionality, other numerical methods such as nested monte carlo simulation have been utilized as generic tools to approximate functional stochastic processes and option prices but are however time consuming and computationally demanding if we introduce various stochastic processes and multi-dependence structures. The approaches under stochastic volatility feature for option pricing such as models by Hull and White (1987), Johnson and Shanno (1987), Scott (1987), Wiggins (1987), Stein and Stein (1991), Heston (1993), Ball and Roma (1994), Bates (1996, 2001), Scott (1997), Bakshi and Chen (1997), Sabanis (2003), Lee (2004) and Bakshi (2010) have developed closed form equity option formulas that admit both stochastic volatility and stochastic interest rates using Fourier inversion of characteristic function but these solutions are semi-analytical thereby computationally

involving for a pricing a portfolio of many assets across strike, maturity and obtaining their risk sensitivities.

Therefore, a model that captures better these stochastic dynamics in interest rates, volatility and jumps in the pricing and is fully analytical and less complex is desirable; and with it an associated risk system that will test its robustness in a real-time practical financial market environment. In practice price estimation of options and risk sensitivities under stochastic parameters for all strikes, maturities and payoffs is demanding and their constant revaluation within a financial institution's risk assessment, trading, hedging and portfolio optimization processes significantly increases the computational resource requirements. This amplifies the need for use of a parallel and accelerated system deployed on Central Processing Unit (CPU) and the Graphics Processing Unit (GPU). The control of risk exposure is an impossible task for the risk managers without appropriate methodology, advanced analytical tools, and such a first-class technological infrastructure. Therefore, in order to have an efficient financial engineering system we need to explore more complex valuation methodologies that can perform calibrations for market valuation and risk sensitivities as well as implement stress tests, back tests and performance tests for all strikes and all maturities on the entire trading book in real-time. This would enable traders and risk managers to make proper investment and hedging decisions real-time. Such a fast and efficient system requires a bespoke technological infrastructure that is distributed, parallelized and accelerated (Chorafas, 2007; Kanniainen, Piché & Mikkonen, 2009; Kanniainen & Koskinen, 2017).

1.3 Research Objectives

1.3.1 General Objective

The aim of this research is to formulate dynamic parallel algorithms for derivative security pricing and hedging. This will involve the implementation of novel dynamic option pricing closed-form algorithms in a heterogeneous parallel GPU-CPU system architecture.

1.3.2 Specific Objectives

i. To develop a dynamic option pricing closed-form fully analytical formulae under stochastic volatility, stochastic interest rates and finite proportional jumps and their respective risk sensitivities, namely: delta, gamma, dependence delta and dependence gamma.

ii. To develop an accelerated financial engineering prototype system by using hybrid programming methodologies and parallel computing techniques that utilize both the Central Processing Unit (CPU) and the Graphics Processing Unit (GPU) for the formulated pricing approaches.

iii. To test the acceleration performance of the developed prototype in the pricing of derivatives securities.

iv. To conduct a comprehensive comparative empirical study demonstrating the extent of each stochastic feature in improving option pricing and hedging performance.

1.4 Research Questions

i. How can option pricing theory and dynamic copula theory be applied to develop a dynamic option pricing closed-form fully analytical formulae under stochastic volatility, stochastic interest rates and finite proportional jumps? And are the associated risk sensitivities obtainable?

ii. How can a high performance financial engineering system be developed through by using hybrid programming and heterogeneous CPU-GPU parallel computing technique to implement the developed pricing formulae?

iii. How efficient are the developed CPU-GPU parallel algorithms in acceleration of the pricing and hedging of derivatives across strikes, maturities and assets?

iv. How does allowing for stochastic features in the option pricing formulation improve pricing and hedging performance?

1.5 Justification

1.5.1 Contribution to Society

Risk Managers: For the risk managers the departure for joint normality assumption may cause exposures that may lead to huge losses. Thus, the choice of inputs into the pricing algorithm and the pricing model may lead to risk exposures in options trading. The risk manager strives to choose the best method and the best set of parameters in order to report exposures. Low model risk is also ensured also by the risk manager having risk analytics in desired time given his large portfolio position and complex pricing models. If the pricing model and computing infrastructure are not

efficient then the option prices quoted, and positions consequently taken may lead to significant losses.

Financial Engineers: Origination of new financial products such as exotic derivatives, portfolio positions and strategies are affected by the inputs in the valuation models. In the valuation of these new products, financial engineers will desire an array or statistical techniques together with derivative valuation knowledge, more so on incorporating stochastic dynamic, to come up with better fair values of these new products. Quantitative analysis has brought innovation, efficiency and rigor to financial markets and to the investment process. And thus, with reliance on these novel modelling techniques and understanding the dependence structure of the underlying stochastic processes of these new products may be useful to ascertain profitable business and meet the client's portfolio needs for investment banks, insurance companies and hedge funds.

Academia: Knowledge of copulas for pricing options may be extended to pricing other derivatives. There is at present a lack in literature on use of martingale approach and copula theory to value options whose result is not by simulation or by use of a semi-analytical approximation. This research will address this by applying copula theory and martingale theory to pricing options under stochastic volatility, stochastic interest rates and jumps. It will also help us understand dependence risk in stochastic pricing environment. The implementation of such algorithms in a heterogeneous parallel system is also of interest in the design of fault tolerant and low latency algorithms.

Traders: Traders in options desk typical activity is to price and then sell the option if it meets a client's payoff requirement, or hedge position fully or part off the risk involved. The trader is thus concerned with being wrong in the pricing of the option when he sells it below its value and being wrong in the hedging position consequently taken. Poor correlation, dependence estimates and copula model selection may lead the trader to misprice the options, leading to taking inaccurate hedging positions. Correlations and dependence in the underlying stochastic processes may change through time and these changes in the level of dependence may produce substantial change in the options' prices and consequently problematic hedging if sensitivity is not assessed. If the trader is capable of using more sophisticated methods in computing infrastructure and pricing models, at least in order to assess his risk exposures and fair value of the assets that he is buying and selling; then he can have substantially profitable defined-risk business.

1.5.2 Contribution to Knowledge

This research presents several contributions to the fields of computing and mathematical finance, substantially transforming the way we approach derivatives pricing and hedging and its associated technologies. A reflection of these contributions are:

- An introduction of a dynamic conditional pair copula under multi-conditionality framework. This approach would advance the way we approach the pricing of derivatives under stochastic dynamics, providing an alternative framework to build pricing models. When applied in this research, apply it to develop a dynamic option pricing analytical model that incorporates stochastic volatility, stochastic interest rates and finite proportional jumps. This novel approach enhances the accuracy of option pricing, fostering more informed investment decision-making process and risk management.

- An introduction of a hybrid programming approach for the data and computation intensive application in a heterogeneous computing environment with varying CPU and GPU hardware architecture where the compute-intensive and data-intensive algorithms are written in a mixed language deriving from Python, C sourced R-implementation and OpenCL C. This approach enhances the efficiency and versatility of computational finance applications.

- An implementation of dynamic kernel offloading to the GPU to provide significant performance per watt and parallelization for performance acceleration of compute-intensive portion of the prototype. This provides a major acceleration and leads to faster derivatives pricing and the effective use of computational resources. This is a significant stride in the optimization of computational power and an advancement in high-performance computing in finance with implication for energy efficiency, processing speed and computational resource management.

Chapter 2: Closed-form Pricing of European Options under Stochastic Volatility and Stochastic Interest Rates using Dynamic Conditional Copula

2.1 Chapter Overview

In this chapter, a novel closed-form formula for pricing European options under stochastic volatility is presented and its empirical performance evaluated. A dynamic conditional copula approach is proposed for the payoff probability distribution. To stress the simplicity and flexibility of the proposed approach to option pricing under stochastic volatility - the stochastic volatility is first constructed as following a mean-reverting Gaussian process and subsequently a square-root stochastic volatility process. The empirical tests using S&P 500 option prices indicated that in comparison to the Heston (1993) model - the dynamic models were found to outperform significantly and consistently across moneyness levels, maturity terms and as well as in various copula functions specifications. The dynamic pricing model with square-root stochastic volatility was the best ranking performer across moneyness at 13% overall performance improvement and 10.6%, 9.8% and 25.4% performance improvement for In-the-Money (ITM), At-the-Money (ATM) and Out-the-Money (OTM) call option pricing respectively under the empirical copula specification. Extension of the model by incorporating stochastic interest rate resulted to further performance improvement of an aggregate of 33.44% across moneyness and copula specification over the Heston model, with cross-copula deviation being low at only 0.001614 was achieved for ITM options, and 0.04574 for ATM options and 0.026502 for OTM options was computed. Accurate empirical performance in the pricing of options in addition to uniquity of being pliable and analytically tractable demonstrates this framework transcendence over previous approaches.

2.2 Introduction

The Black and Scholes (1973) model in its assumptions assert the existence of a flat implied volatility surface profile. Implied volatilities are obtained by inversion of the Black-Scholes formula and are widely used options market indicators. It has been empirically observed however that this surface exhibits dependence on maturity and strike price at a given cross-section and time-series. This empirical implied volatility surface has thus been noted to exhibit stylized characteristics of ‘smile’, ‘skew’ or ‘smirk’ which changes over time as a result of options market dynamics. Empirical studies on the behavior of implied volatilities have pointed to some

commonality of these characteristics across markets particularly on the smile patterns, the term structure and the time series pattern which displays high autocorrelation and mean-reverting behavior. Further, stylized facts of asset market returns have also indicated presence of non-constant volatility corroborated by presence of volatility clustering and conditional heavy tails after correcting for volatility clustering, presence of correlations asymmetry in stress market conditions where leverage effects of negative correlation of asset return and volatility exists, and underlying tail dependence with gain-loss asymmetry where downsize movements are disproportionately larger than to upward movements. Consequently, the assumption of asset return normality is also violated in practice given these heavy tails and skewness of return distribution (Cao et al., 2010; Cont, 2001; Cont et al., 2002; Dumas et al., 1998; Alexander, 2001; Wallmeier and Hafner, 2000; Fouque et al., 1999; Bates, 2001).

In its implementation in practice, the Black-Scholes algorithm albeit assumes a constant volatility and constant interest rates, is implemented as though volatility and interest rates are time varying being extracted by algorithm inversion from daily prices and current yield curve (Cao et al., 2010). This treatment further substantiates the weakness of these static assumptions of Black-Scholes model and inherently implying presence of stochasticity in volatility and interest rates by construction. There have been several suggested approaches that attempt to explicitly define and capture these features into the pricing of options. Models to capture the instantaneous profile of the implied volatility have been proposed such as local volatility models, non-linear diffusion, jump-diffusion models, GARCH models and stochastic volatility models (Cao et al., 2010; Cont, 2001).

The approaches under stochastic volatility feature for option pricing include models by Hull and White (1987), Johnson and Shanno (1987), Scott (1987), Wiggins (1987), Stein and Stein (1991), Heston (1993), Ball and Roma (1994), Bates (1996), Bates (2001), Scott (1997), Bakshi et al. (1997), Sabanis (2003), Lee (2004) and Cao et al. (2010). In these approaches, except for Hull and White (1987), Fourier inversion techniques were applied to develop a semi-analytical expression for pricing of options. Hull and White (1987) make use of conditional density function and Taylor series expansion, suggesting an option pricing expression under stochastic volatility framework with independent driving Brownian motions as the expected Black-Scholes price where the expected value is taken over the probability distribution of the integrated squared volatility. They

computed moments of the integrated volatility process to obtain approximate option prices. In the Fourier inversion approach, the probability density function is obtained by inversion of the corresponding Fourier transform and an integral is taken over the option payoff function to obtain the option prices by numerical integration using approximation methods such as Gauss Lagendre or Gauss Lobatto integration. Although, the Fourier inversion approach and the Fast Fourier Transform approaches by Carr and Madan (1999) and Lee (2004) have been widely used and noted to be much faster than finite difference methods for Partial Differential Equations; their complexities, accuracy and computational performance has become a critical factor when calibrating the models to observed market prices and in obtaining Greeks and hedging strategies.

A new methodological development for a less complex fully analytical closed-form formula under stochastic volatility framework is still deficient given that the trade-off between flexibility and tractability is particularly delicate. A model that is flexible enough and analytically tractable to capture the stochastic volatility dynamics and associated heavy tail and skewness in stock price distributions in the pricing of options needs to be considered. In this section, a novel dynamic copula-based approach for option pricing under stochastic volatility and stochastic interest rates is proposed and presented. A conditional copula h-function is suggested to capture the existent non-linear dependence structure between the stock price process and the stochastic volatility and stochastic interest rates process. This function also captures the joint conditional probability distribution of the option expiring in the money. To stress the simplicity and flexibility of the proposed copula approach to pricing under stochastic volatility, the stochastic volatility is constructed as following a log-normal process, a mean-reverting Gaussian process and in the subsequent construction the strictly non-negative property is invoked to make negative volatility strictly inaccessible by defining a mean-reverting square-root process for the stochastic volatility. The stochastic interest processes considered are of the CIR and Vasicek types.

Closed-form formulae of the European option prices under these different configurations of stochastic volatility and stochastic interest rates are analytically derived by application of martingale approach. This pricing approach under stochasticity is termed as being dynamic as it allows for several alternative dependence structures between the driving processes to be captured by a malleable conditional copula h-function whose formulation is flexible to variant

configurations of the stochastic volatility and stochastic interest rates processes and their resultant integrals. This approach is an important development over the Fourier inversion techniques and other numerical approximation methods by simulation or partial differential equations that are being applied currently to pricing options under stochastic volatility, since it demonstrates much more accurate performance in the pricing of options in addition to its uniquity of being pliable, extensible and analytically tractable in pricing applications for stochastic volatility, stochastic interest rates and jumps.

Each configuration developed in this section was implemented and evaluated on the pricing performance for S&P 500 Index call options contract across moneyness and maturity category in comparison to the Heston and Black-Scholes estimates. These configurations were found to result in an aggregate respective performance improvement of 5% and 13% across all moneyness levels and terms to expiry under empirical copula specifications over the Heston (1993) model. Particularly, the model under mean-reverting square-root stochastic volatility resulted in 10.6%, 9.8% and 25.4% performance improvement for ITM, ATM and OTM; performing better than that under the mean-reverting Gaussian configuration under empirical copula. Copula-specific performance improvement for OTM options of up to 47% by Gaussian copula and 36% by frank copula was achieved for the respective dynamic model configurations. Performance improvement was uniform for ITM and ATM options at 6% and 10%, and 5% and 9% for both models respectively across copula function specifications. The subsequent sections are structured as follows: Section 2.2 present review of copula functions, Section 2.3 presents dynamic pricing model under stochastic volatilities, Section 2.4 presents the development of the closed-form dynamic formulae for European options under stochastic volatility, stochastic interest rates and jumps, Section 2.4.1 presents the closed-form formulae for the risk sensitivities. Section 2.5 and Section 2.5 presents the numerical results of the performance evaluation of the closed-form dynamic pricing formula with the respective discussions.

2.2 Models Specification

2.2.1 Joint probability and Marginals

If X and Y are absolutely continuous, then X and Y are independent *iff* the joint density is equal to the product of the marginal ones, that is *iff*

$$\varphi_{X,Y} = \varphi_X(x) \cdot \varphi_Y(x)$$

Where $f_X(x)\ and\ f_Y(x)$ are the marginal densities. By the application of Fubini's theorem and partial integration it holds that the joint cumulative distribution function in the case of independence becomes

$$\begin{aligned}\phi_{X,Y}(x,y) &= \int_{-\infty}^{x}\int_{-\infty}^{y} \varphi_{X,Y}(w,s)dw\ ds \\ &= \int_{-\infty}^{x}\int_{-\infty}^{y} \varphi_X(w)\cdot\varphi_Y(s)dw\ ds \\ &= \int_{-\infty}^{x} \varphi_X(w)dw \int_{-\infty}^{y} \varphi_X(s)ds \\ &= \phi_X \cdot \phi_Y\end{aligned}$$

2.2.2 Copula

Definition: A copula function is defined in the bivariate case as follows:

A two-dimensional copula is a function C: [0,1] × [0,1] → [0,1] with the following properties:

i) For every $u, v\ \in [0,1]$:

$$C(u,0) = C(0,v) = 0$$

ii) For every $u, v\ \in [0,1]$:

$$C(u,1) = u\ and\ C(1,v) = v$$

iii) For every $u_1, u_2, v_1, v_2 \in [0,1]$ with $u_1 \le u_2\ and\ v_1 \le v_2$:

$$C(u_2,v_2) - C(u_2,v_1) - C(u_1,v_2) + C(u_1,v_1) \ge 0$$

Given the copula C for uniform distributed variables *u* and *v* we have that

$$\begin{aligned}C(\phi_X(x),\phi_X(y) &= C(u,v) \\ &= \Pr\big(u \le \phi_X(x), v \le \phi_Y(y)\big) \\ &= \Pr(\phi_X^{-1}(u) \le x, \phi_Y^{-1}(v) \le y) \\ &= Pr(X \le x, Y \le y) = \phi_{X,Y}(x,y)\end{aligned}$$

According to Sklar's theorem, while writing

$$\phi_{X,Y}(x,y) = C(\phi_X(x),\phi_Y(y))$$

joint probability is split into the marginals and a copula so that the copula only represents the "association" between X and Y.

2.2.3 Copula density and h-functions

If a copula is sufficiently differentiable the copula density can be defined as follows:

$$c(u,v) = \frac{\partial^2 C(u,v)}{\partial u \partial v}$$

For continuous random vectors the copula density is related to the density of the distribution, denoted as φ. The copula density is equal to the ratio of the joint density $\varphi_{x,y}$ to the product of the marginal densities $\varphi_x \; and \varphi_y$, given as

$$c(u,v) = \frac{\varphi_{x,y}(x,y)}{\varphi_x\,(x)\varphi_y\,(y)}$$

By Sklar's theorem the following canonical representation holds

$$\varphi_{x,y}(x,y) = c(u,v)\varphi_x\,(x)\varphi_y\,(y)$$

The partial derivatives of a copula distribution have a probabilistic meaning. More precisely, if $u_{1,}u_2$ has the copula *C* as joint distribution function, then the function

$$h_m : u_1 \to \frac{\partial}{\partial m} C(u_1, m), m \in (0,1)$$

can typically be computed for almost every m ∈ (0,1). It equals the distribution function of $u_1$conditional on the event $\{u_2 = m\}$. For some bivariate copulas, the function h_x can be computed, and even inverted, in closed form. This function is known as the h-function and represents the cumulative distribution function X conditional on Y.

$$\Pr(X \le x | Y = y) = C_{X|Y}\big(\phi_X(x), \phi_Y(y)\big) = C_{X|Y}(u,v) = \frac{\partial C(u,v)}{\partial v} = h_v$$

The h functions $\frac{\partial C(u,v)}{\partial v} \; and \frac{\partial C(u,v)}{\partial u}$ exist for almost all *u* and *v*, as shown by Nelsen (2006) we have that $0 \le \frac{\partial C(u,v)}{\partial v} \le 1$ and $0 \le \frac{\partial C(u,v)}{\partial u} \le 1$. This then suggests an alternative way to writ the copula function

$$\phi_{X,Y}(x,y) = C(u,v) = \int_0^v \frac{\partial C(u,\omega)}{\partial \omega} d\omega$$

2.2.4 Parametric Estimation of Pair Copula

The methodology employed to estimate the pair copula parameters is the Maximum likelihood method. The maximum-likelihood method starts from the assumption of having a parametric model for the distribution in concern. It then estimates all parameters of the marginals by the maximum likelihood method (by maximizing the log-likelihood function). The joint density is the represented in the canonical form as a product of the copula density and the marginal densities. Log transform is then applied and splits the problem into a sum of terms that are alike in the univariate maximum-likelihood procedure (for each of the marginal laws) and a term that depends on both: parameters of the marginal laws and parameters of the dependence structure (log of copula density).

2.2.5 Martingale Option Pricing Approach

The pricing of a European Call Option $g(S,t)$ which expires at time T with a strike price of K by risk-neutral valuation under constant volatility and constant interest rate is obtained by

$$g(S,t) = e^{-r(T-t)}\mathbb{E}_{\mathbb{Q}}(h(S_T))$$

Where $h(S_T)$ is the payoff function of $S_T - K$. Dufresne, Keirstead and Ross (1996) show by using the martingale approach and auxiliary probability measure adopted by Girsanov Theorem that the Black-Scholes equation is derived with computation simplicity as:

$$C(S,t) = e^{-r(T-t)}\mathbb{E}_{\mathbb{Q}}(S_T - K)^+ = \mathbb{E}_{\mathbb{Q}}(S_T e^{-r(T-t)}\mathbb{1}_{S_T>K}) - \mathbb{E}_{\mathbb{Q}}(Ke^{-r(T-t)}\mathbb{1}_{S_T>K})$$

$$= A_1 - A_2$$

By assuming that S_T follows a geometric Brownian motion under martingale measure $\mathbb{Q}$, and follows a log-normal distribution such that $ln\frac{S_T}{S_t} \sim N\left(\left(r - \frac{1}{2}\sigma^2\right)(T-t), \sigma^2(T-t)\right)$ and $dW^{\mathbb{Q}} \sim N(O, T-t)$. From the definition of and indicator function, $\mathbb{E}_{\mathbb{Q}}\left(\mathbb{1}_{S_T>K}\right) = \mathbb{Q}(S_T > K)$. We have therefore that second integral as:

$$A_2 = \mathbb{E}_{\mathbb{Q}}(Ke^{-r(T-t)}\mathbb{1}_{S_T>K}) = Ke^{-r(T-t)}\mathbb{Q}(S_t e^{\left(r-\frac{1}{2}\sigma^2\right)(T-t)+\sigma dW^{\mathbb{Q}}} > K)$$

$$A_2 = Ke^{-r(T-t)}\mathbb{Q}\left(\xi \leq \frac{ln\frac{S_t}{K}+\left(r-\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}\right) = \mathrm{Ke}^{-\mathrm{r}(\mathrm{T}-\mathrm{t})}\phi(\mathrm{d}_2) \text{ Where } \mathrm{d}_2 = \frac{\ln\frac{\mathrm{S_t}}{\mathrm{K}}+\left(\mathrm{r}-\frac{1}{2}\sigma^2\right)(\mathrm{T}-\mathrm{t})}{\sigma\sqrt{\mathrm{T}-\mathrm{t}}}$$

Theorem 2.1 Girsanov Theorem

Two probability measures $\mathbb{F}$ and $\mathbb{Q}$ on *(*Ω*,* F_T*)* are said to be *equivalent* if, for any event $A \in FT$, the equality $\mathbb{F}(A) = 0$ holds if and only if $\mathbb{Q}(A) = 0$. In other words, $\mathbb{F}$ and $\mathbb{Q}$ are equivalent on *(*Ω*,* F_T *)* if they have the same set of null events in the σ -field F_T . Given *a one-dimensional standard Brownian motion on a probability space (*Ω*,* F*,* $\mathbb{Q}$ *). We define the process X by setting the wiener process of the geometric brownian motion to* $W^{\mathbb{F}} = W^{\mathbb{Q}} + \lambda t$ *for t* $\in$ [0,*T*]*. Let the probability measure* $\mathbb{F}$ *, equivalent to* $\mathbb{Q}$ *on (*Ω*,* F_T*), be defined through the formula*

$$\frac{d\mathbb{F}}{d\mathbb{Q}} = \exp\left(\lambda \cdot W^{\mathbb{Q}}{}_T - \frac{1}{2}\lambda^2 T\right) = \eta_T$$

This is the *Radon-Nikodým density* of $\mathbb{F}$ with respect to $\mathbb{Q}$ is defined as the unique F_T - measurable random variable η_T such that we have, for any event $A \in F_T$.

By virtue of the above proposition, setting $W^{\mathbb{F}} = W^{\mathbb{Q}} - \sigma(T-t)$, the *Radon-Nikodým density is:*

$$\frac{d\mathbb{F}}{d\mathbb{Q}} = \exp\left(\sigma \cdot W^{\mathbb{Q}}{}_T - \frac{1}{2}\sigma^2(T-t)\right) = \eta_T$$

We therefore have that $\mathbb{E}_{\mathbb{F}}\left(\mathbb{1}_{S_T>K}\right) = \mathbb{E}_{\mathbb{Q}}\left(\eta_T \mathbb{1}_{S_T>K}\right)$

The solution of the first integral A_1 yields,

$$A_1 = \mathbb{E}_{\mathbb{Q}}\left(S_T e^{-r(T-t)}\mathbb{1}_{S_T>K}\right) = \mathbb{E}_{\mathbb{Q}}\left(\eta_T \mathbb{1}_{S_T>K}\right) = S_t \mathbb{E}_{\mathbb{F}}\left(\mathbb{1}_{S_T>K}\right)$$

$$A_1 = S_t\mathbb{F}\left(S_t e^{\left(r-\frac{1}{2}\sigma^2\right)(T-t)+\sigma d\bar{W}+\sigma^2(T-t)} > K\right) = S_t\mathbb{F}\left(\xi \leq \frac{ln\frac{S_t}{K}+\left(r+\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}\right) = S_t\phi(d_1)$$

Where $d_1 = \frac{ln\frac{S_t}{K}+\left(r+\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$. The $\mathbb{F}$ measure is the equivalent martingale measure when the stock mutual fund $X_t = S_t$ is used as numeraire, as opposed to the $\mathbb{Q}$ measure which is the risk neutral e measure when the money market fund e^{rt} is used as numeraire. Therefore have the price of a European call option as

$$g(S,t) = S_t\phi(d_1) - \mathrm{Ke}^{-\mathrm{r(T-t)}}\phi(d_2)$$

2.3 Dynamic Option Pricing Under Stochastic Volatility

In this section, we develop a closed-form fully analytical European option pricing formula under stochastic volatility. In the subsequent sub-section we extend this developed dynamic stochastic volatility (DSV) model to allow for stochastic interest rates and jumps, i.e a dynamic stochastic volatility & stochastic interest rates (DJSVSI) model from which the models under stochastic volatility and stochastic interest rates are special cases.

We model the stock price by a geometric Brownian motion with a variance parameter that follows an Ornstein–Uhlenbeck (OU) process. This setup is described under risk neutral probability measure $\mathbb{Q}$ as

$$dS(t) = S(t)rdt + S(t)\sigma(t)d\widetilde{w}_s(t) \qquad 2.3.1$$

$$d\sigma^2(t) = k(\alpha - \sigma^2(t)) + \delta d\widetilde{w}_v(t) \qquad 2.3.2$$

We specify $d\widetilde{w}_s(t)$ and $d\widetilde{w}_v(t)$ to be independent standard Brownian motions on the filtered probability space (Ω,F, $\mathbb{Q}$,F_t). The correlation $\rho \in [1,-1]$ is the instantaneous correlation between the stock price and the instantaneous variance processes. r is the continuously compounded risk-free rate, $\{S_t, t > 0\}$ is the price process of the stock that pays no dividends, $\{\sigma^2(t), t > 0\}$ is the instantaneous variance process described as observing a Gaussian mean-reverting stochastic volatility with k, α and δ as the mean-reversion speed, the long-run mean and volatility rate respectively.

The solution to equation (2.3.1) follows as

$$ln\,\frac{S_T}{S_t} = \int_t^T rds - \frac{1}{2}\int_t^T \sigma^2(s)ds + \int_t^T \sigma(s)d\widetilde{w}_s(t)$$

$$ln\ \frac{S_T}{S_t} = r(T-t) - \frac{1}{2}V_T(T-t) + \xi_T\sqrt{E(V_T)T-t}$$

$$ln\ \frac{S_T}{S_t} = (r - \frac{1}{2}V_T)(T-t) + d\tilde{w}_s^v\sqrt{E(V_T)} \qquad 2.3.3$$

Where $V_T = \frac{1}{T-t}\int_t^T \sigma^2(s)ds$ and $\xi_T = [E(V_T)]^{-\frac{1}{2}}\int_t^T \frac{\sigma(s)d\tilde{w}_s(t)}{\sqrt{T-t}}$ and $\xi_T \sim N(0,1)$, $d\tilde{w}_s^v = \xi_T\sqrt{T-t}$ and $d\tilde{w}_s^v \sim N(0, T-t)$.

The solution to equation (2.3.2) follows that for $s \geq t$, we have that for

$$\sigma^2(s) = \sigma^2(t)e^{-k(s-t)} + \alpha\left(1 - e^{-k(s-t)}\right) + \delta\int_t^s e^{-k(s-u)}\, d\tilde{w}_v(u) \qquad 2.3.4$$

Theorem 2.3.1: Fubini Theorem

Three main results may be derived from Fubini's theorem, these are:

(I) $\int_0^\tau \left(\int_0^t g(u,v,\omega)dW(u)\right)dv = \int_0^t\left(\int_0^\tau g(u,v,\omega)dv\right)dW(u)$ for all $0 \leq t \leq \tau$

(II) $\int_t^T\left(\int_0^\tau g(u,v,\omega)dW(u)\right)dv = \int_0^\tau \left(\int_t^T g(u,v,\omega)dv\right)dW(u)$ for all $0 \leq t \leq \tau \leq T$

(III) $\int_0^t\left(\int_0^v g(u,v,\omega)dW(u)\right)dv = \int_0^t \left(\int_u^t g(u,v,\omega)dv\right)dW(u)$

By applying Fubini Theorem, we proceed from equation 2.3.4 the integrated aggregate variance process becomes:

$V_T = \frac{1}{T-t}\int_t^T \sigma^2(s)ds = \frac{(\sigma^2(t)-\alpha)B(t,T)}{T-t} + \alpha + \frac{\delta}{T-t}\int_t^T B(u,T)d\tilde{w}_v(u)$ Where $B(t,T) = \frac{1}{k}(1 - e^{-k(T-t)})$

The distribution of the integrated variance process is conditionally normal with mean and variance,

$$M_V(t) = E[V_T] = E\left[\frac{1}{T-t}\int_t^T \sigma^2(s)ds\right] = \frac{(\sigma^2(t)-\alpha)B(t,T)}{T-t} + \alpha \qquad 2.3.5$$

$$V_V^2(t) = Var(V_T) = E\left[\frac{\delta^2}{(T-t)^2}\int_t^T B(u,T)^2du\right] = \frac{\delta^2}{k^2(T-t)^2}\int_t^T (1 - 2e^{-k(T-u)} + e^{-2k(T-u)})du$$

$$V_V^2(t) = \frac{\delta^2}{k^2(T-t)^2}\left((T-t) - 2B(t,T) + \frac{1}{2k} - \frac{e^{-2k(T-t)}}{2k}\right) \qquad 2.3.6$$

The stock price is conditionally log-normally distributed on $\sigma^2(t)$ with

$$M_S(t) = E\left[ln\ \frac{S_T}{S_t}\right] = \left(r - \frac{1}{2}E[V_T]\right)(T-t) = \left(r - \frac{1}{2}M_V(t)\right)(T-t) \qquad 2.3.7$$

$$V_S^2(t) = Var\left[ln\frac{S_T}{S_t}\right] = \frac{(T-t)^2}{4}Var(V_T) + E(V_T)Var\left(d\tilde{w}_s^v\right) - (T-t)\sqrt{E(V_T)}cov(V_T, d\tilde{w}_s^v)$$

$$V_S^2(t) = Var\left[ln\frac{S_T}{S_t}\right] = \frac{(T-t)^2}{4}V_V^2(t) + M_V(t)(T-t) - \rho_{V_T,d\tilde{w}_s^v}(T-t)\sqrt{M_V(t)V_V^2(t)(T-t)} \qquad 2.3.8$$

Theorem 2.3.2: Patton (2001) Conditional copula theorem

Let $F_{X|Z}(\cdot|z)$ be the conditional distribution of X given Z=z and $F_{Y|Z}(\cdot|z)$ be the conditional distribution of Y given Z=z. And let $F_{XY|Z}(\cdot,\cdot|z)$ be the joint conditional distribution of (X,Y) given Z=z. Assume that $F_{X|Z}(\cdot|z)$ and $F_{Y|Z}(\cdot|z)$ are continuous in x and y for all z∈Z.

Then there exists a unique conditional copula $C(\cdot|z)$ such that $F_{XY|Z}(x,y|z)=C(F_{X|Z}(x|z), F_{Y|Z}(y|z)|z)$. Conversely, if we let $F_X|Z(\cdot|z)$ be the conditional distribution of X given $Z = z$, $F_Y|Z(\cdot|z)$ be the conditional distribution of Y given $Z=z$, and $C(\cdot, \cdot|z)$ be a family of conditional copulas that is measurable in z, then the function $F_{XY}|Z(\cdot,\cdot|z)$ is a conditional bivariate distribution function with conditional marginal distributions $F_X|Z(\cdot|z)$ and $F_Y|Z(\cdot|z)$. The key point in this "conditional" version of Sklar's theorem is that the conditioning variable Z must be the same for both marginal distributions and the copula, and this is a fundamental condition.

Theorem 2.3.3: Darsow et al (1992) Conditional Probability Theorem

$$P(X \le x|Y = y) = \lim_{y\to 0} P(X \le x|y \le Y \le y + y) = \lim_{y\to 0}\frac{F(x, y + \Delta y) - F(x, y)}{F_Y(y + \Delta y) - F_Y(y)}$$

$$= \lim_{y\to 0}\frac{C(F_X(x), F_Y(y + \Delta y)) - C(F_X(x), F_Y(y))}{F_Y(y + \Delta y) - F_Y(y)} = \frac{\partial C(F_X(x), v)}{\partial v}$$

Proposition 2.3.1: Conditional copula h-function

Given Theorem 2.3.2 and Theorem 2.3.3, we develop here a proposition.

Let $F_{X|Z}(\cdot|z)$ be the conditional distribution of X given Z=z and $F_{Y|Z}(\cdot|z)$ be the conditional distribution of Y given Z=z. And let $F_{XY|Z}(\cdot,\cdot|z)$ be the joint conditional distribution of (X,Y) given Z=z. Assume that $F_{X|Z}(\cdot|z)$ and $F_{Y|Z}(\cdot|z)$ are continuous in x and y for all z∈Z.

Then there exists a unique conditional copula $C(\cdot|z)$ such that $F_{XY|Z}(x,y|z)=C\ (F_{X|Z}(x|z), F_{Y|Z}(y|z)\ |z)$. Then by application of Sklar's theorem, the joint conditional distribution of X on Y=y given prior conditioning on Z we have. $F_{\{X|Y\}|Z\}}(\{X \leq x|Y = y\}|z) = \frac{\partial C(F_{X|Z}(x|z), F_{Y|Z}(y|z))}{\partial F_{Y|Z}(y|z)}$. The value $\frac{\partial C(F_{X|Z}(x|z), F_{Y|Z}(y|z))}{\partial F_{Y|Z}(y|z)}$ is the conditional copula h-function on Z.

The probability that the option will expire in the money at maturity is given as

$$P(S_T > K|\sigma_t^2) = P(S_T > K|V_T|\sigma_t^2)$$

The stock price process is intermediately conditionally dependent on the average volatility. We thereby have a case of double conditioning on V_T and σ_t^2 such that

$$P(S_T > K|V_T|\sigma_t^2) = P(\{S_T > K|V_T\}|\sigma_t^2)$$

By the application of the Proposition 2.3.1 above

$$P(\{S_T > K|V_T\}|\sigma_t^2) = \frac{\partial C(F_{S|\sigma_t^2}(S_T > K|\sigma_t^2),\ F_{V|\sigma_t^2}(V_T|\sigma_t^2); \theta)}{\partial F_{V|\sigma_t^2}(V_T|\sigma_t^2)}$$

Let $u = F_{S|\sigma_t^2}(S_T > K|\sigma_t^2)$ and $v = F_{V|\sigma_t^2}(V_T|\sigma_t^2)$, where u is the conditional log-normal distribution function with parameters as in equation 2.3.5 and 2.3.6 and v is the conditional normal distribution function with parameters as in equation 2.3.7 and 2.3.8. We have therefore that the probability that the option is going to expire in the money under stochastic volatility being:

$$P(S_T > K) = \frac{\partial C(u,\ v, \theta)}{\partial v} = h_v(u, v; \theta)$$

Therefore, the price of a European call option $g(S,t)$ which expires at time T with a strike price of K under risk-neutral valuation under stochastic volatility is

$$g(S,t) = \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds} h(S_T)\right]$$

For a call option with payoff of $h(S_T) = S_T - K$

$$g(S,t) = \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds} (S_T - K)\right]$$

$$= \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds} S_T \mathbb{1}_{S_T>K}\right] - \mathbb{E}^{\mathbb{Q}}\left[Ke^{-\int_t^T r(s)ds}\mathbb{1}_{S_T>K}\right]$$

From the definition of and indicator function, $\mathbb{E}_{\mathbb{Q}}\left(\mathbb{1}_{S_T>K}\right) = \mathbb{Q}(S_T > K)$. Since the interest rates r(s) are constant,

$$= S_t \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds} e^{(r-\frac{1}{2}V_T)(T-t)+d\tilde{w}_s^v\sqrt{E(V_T)}}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] - K\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right]$$
$$= B_1 - B_2$$

From the definition of the payoff probability and distribution properties of the log-stock price we have that for B_2,

$$B_2 = K\,\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]\mathbb{E}_{\mathbb{Q}}(\mathbb{1}_{S_T>K})$$

Given the money market account as the numeraire and under the risk-neutral probability measure $\mathbb{Q}$, $\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]$ is the current price of a bond P_t

$$B_2 = KP_t\mathbb{Q}(S_T > K) = KP_t h_v(u, v; \theta)$$

Where the probability $\mathbb{Q}(S_T > K)$ is the conditional probability under stochastic volatility under measure $\mathbb{Q}$ represented as a pair copula as in Proposition 2.3.1 where $u = F^{\mathbb{Q}}_{S|\sigma_t^2}(S_T > K|\sigma_t^2)$ and $v = F^{\mathbb{Q}}_{V|\sigma_t^2}(V_T|\sigma_t^2)$, are the conditional distribution functions under risk neutral measure $\mathbb{Q}$

$$u = F^{\mathbb{Q}}_{S|\sigma_t^2}(S_t e^{(r-\frac{1}{2}V_T)(T-t)+d\tilde{w}_s^v\sqrt{E(V_T)}} > K)$$

$$u = F^{\mathbb{Q}}_{S|\sigma_t^2}\left(\xi \leq \frac{\ln\frac{S_t}{K}+r(T-t)-\frac{1}{2}M_V(t)(T-t)}{\left(\frac{(T-t)^2}{4}V_V^2(t)+M_V(t)(T-t)-\rho_{V_T,d\tilde{w}_s^v}(T-t)\sqrt{M_V(t)V_V^2(t)(T-t)}\right)^{0.5}}\right)$$

$$u = \phi(d_2^S) \text{ where } d_2^S = \frac{\ln\frac{S_t}{K}+r(T-t)-\frac{1}{2}M_V(t)(T-t)}{\left(\frac{(T-t)^2}{4}V_V^2(t)+M_V(t)(T-t)-\rho_{V_T,d\tilde{w}_s^v}(T-t)\sqrt{M_V(t)V_V^2(t)(T-t)}\right)^{0.5}}$$

For conditional distribution function *v*,

$$v = F^{\mathbb{Q}}_{V|\sigma_t^2}\left(\frac{\left(\sigma^2(t)-\alpha\right)B(t,T)}{T-t} + \alpha + \frac{\delta}{T-t}\int_t^T B(u,T)d\tilde{w}_v(u) \leq \hat{\sigma}_c^2\right)$$

$$v = F^{\mathbb{Q}}_{V|\sigma_t^2}\left(\xi \leq \frac{\hat{\sigma}_c^2 - M_V(t)}{\sqrt{V_V^2(t)}}\right) = \phi(d_2^V) \quad \text{where } d_2^v = \frac{\hat{\sigma}_c^2 - M_V(t)}{\sqrt{V_V^2(t)}}$$

We thereby have the value for B_2 as:

$$B_2 = KP_t h_2(\phi(d_2^S), \phi(d_2^V); \theta)$$

Where we have $h_2(\phi(d_2^S), \phi(d_2^V); \theta)$ as the pair copula h-function with respect to the second argument, i.e, $v = \phi(d_2^V)$ and with dependence parameter θ. For B_1, applying the Girsanov's Theorem and defining the indicator function under the equivalent martingale measure $\mathbb{F}$ which corresponds to the numeraire S, as $\mathbb{E}_{\mathbb{Q}}(\mathbb{1}_{S_T>K}) = \mathbb{E}_{\mathbb{F}}(\eta_T \mathbb{1}_{S_T>K}) = \mathbb{F}\,(S_T > K)$

$$B_1 = S_t \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds} e^{(r-\frac{1}{2}V_T)(T-t)+d\tilde{w}_s^v\sqrt{E(V_T)}}\right] \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right]$$

Setting $\Sigma_t = \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds} e^{(r-\frac{1}{2}V_T)(T-t)+d\tilde{w}_s^v\sqrt{E(V_T)}}\right] = e^{-\frac{1}{2}M_V(t)(T-t)+\frac{1}{2}V_S^2\,(t)}$ which is a constant.

We proceed as follows:

$$\mathrm{B_1} = \mathrm{S_t}\Sigma_t\, \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] = \mathrm{S_t}\, \Sigma_t \mathbb{E}_{\mathbb{F}}\left(\eta_T \mathbb{1}_{S_T>K}\right)$$

$$\mathrm{B_1} = \mathrm{S_t}\Sigma_t\, \mathbb{F}\,(\mathrm{S_T} > \mathrm{K})$$

$$\mathrm{B_1} = \mathrm{S_t}\Sigma_t\, \mathrm{h_v}(\tilde{\mathrm{u}}, \tilde{\mathrm{v}}; \theta)$$

Where the probability $\mathbb{F}\,(S_T > K)$is the conditional probability under stochastic volatility under measure $\mathbb{F}$. $\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2}(S_T > K|\sigma_t^2)$ and $\tilde{v} = F^{\mathbb{F}}_{V|\sigma_t^2}(V_T|\sigma_t^2)$, are the conditional distribution functions under measure $\mathbb{F}$. By setting that $d\tilde{w}_s^{v,\mathbb{Q}} = d\tilde{w}_s^{v,\mathbb{F}} + \sqrt{E(V_T)}dt$

$$ln\,\frac{S_T}{S_t} = (r - \frac{1}{2}V_T)dt + d\tilde{w}_s^{v,\mathbb{Q}}\sqrt{E(V_T)}$$

$$ln\,\frac{S_T}{S_t} = (r + E(V_T) - \frac{1}{2}V_T)dt + d\tilde{w}_s^{v,\mathbb{F}}\sqrt{E(V_T)}$$

After integration to obtain the solution of the geometric Brownian motion and its conditional moments under probability measure $\mathbb{F}$, we have that:

$$\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2}(S_t e^{(r+E(V_T)-\frac{1}{2}V_T)(T-t)+d\tilde{w}_s^{v,\mathbb{F}}\sqrt{E(V_T)}} > K)$$

$$\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2}\left(\xi \leq \frac{\ln\frac{S_t}{K}+r(T-t)+E(V_T)(T-t)-\frac{1}{2}E(V_T)(T-t)}{\sqrt{\frac{(T-t)^2}{4}Var(V_T)+E(V_T)Var(d\tilde{w}_S^v)-(T-t)\sqrt{E(V_T)}cov(V_T,d\tilde{w}_S^v)}}\right) = \phi(d_1^S)$$

Where $d_1^S = \frac{\ln\frac{S_t}{K}+\left(r+\frac{1}{2}M_V(t)\right)(T-t)}{\left(\frac{(T-t)^2}{4}V_V^2(t)+M_V(t)(T-t)-\rho_{V_T,d\tilde{w}_S^v}(T-t)\sqrt{M_V(t)V_V^2(t)(T-t)}\right)^{0.5}}$

Further, by setting that $d\tilde{w}_v^{\mathbb{Q}}(t) = d\tilde{w}_v^{\mathbb{F}}(t) + \delta dt$. We have that under probability measure $\mathbb{F}$

$$d\sigma^2(t) = k(\alpha - \sigma^2(t))dt + \delta(d\tilde{w}_v^{\mathbb{F}}(t) + \delta dt)$$

$$d\sigma^2(t) = k(\beta - \sigma^2(t))dt + \delta d\tilde{w}_v^{\mathbb{F}}(t)$$

Where $\beta = \alpha + \frac{\delta^2}{k}$, thus the conditional distribution function *v* under probability measure $\mathbb{F}$

$$\tilde{v} = F^{\mathbb{F}}_{V|\sigma_t^2}\left(\frac{(\sigma^2(t)-\beta)B(t,T)}{T-t}+\beta+\frac{\delta}{T-t}\int_t^T B(u,T)d\tilde{w}_v(u) \leq \hat{\sigma}_c^2\right) = F^{\mathbb{F}}_{V|\sigma_t^2}\left(\xi \leq \frac{\hat{\sigma}_c^2-\frac{(\sigma^2(t)-\beta)B(t,T)}{T-t}-\beta}{\sqrt{\frac{\delta^2}{(T-t)^2}\int_t^T B(u,T)^2du}}\right) = \phi(d_2^V)$$

where $d_1^v = \frac{\hat{\sigma}_c^2-M_V(t)+\frac{\delta^2}{k(T-t)}\left(B(t,T)-\frac{\alpha k}{\delta^2}-1\right)}{\sqrt{V_V^2(t)}}$

We thereby have the value for B_1 as:

$$B_1 = S_t\Sigma_t h_2(\phi(d_1^S), \phi(d_1^V); \theta)$$

Where we have $h_2(\phi(d_1^S), \phi(d_1^V); \theta)$ as the pair copula h-function with respect to the second argument, i.e, $v = \phi(d_1^V)$ and with dependence parameter θ.

Therefore the price of a European call option by the dynamic approach under stochastic volatility (DSV) as is obtained as

$$C(S,t) = S_t\Sigma_t h_2(\phi(d_1^S), \phi(d_1^V); \theta) - KP_t h_2(\phi(d_2^S), \phi(d_2^V); \theta) \qquad 2.3.9$$

Assuming constant interest rates $P_t = e^{-r(T-t)}$ and $\Sigma_t = e^{-r(T-t)+M_S(t)+\frac{V_S^2(t)}{2}}$. We assume that h_2 is the h-function of a normal copula and that $\theta = \rho_{S,V}$. This approach is dynamic as it is possible to allow the use of empirical pair copula of Gumbel, Clayton, Student-t or Frank dependence and their respective h-function making this approach pleasantly dynamic.

2.4 Dynamic Option Pricing under Jumps, Stochastic Volatility and Stochastic Interest rates

In this section, we allow discrete proportional jumps in the asset price process. We apply martingale approach to develop an option pricing formula that allows discrete proportional jumps in the asset price process. Here we consider discretely occurring jumps and not continuous jumps as in Merton (1982). We also incorporate stochastic volatility to develop a dynamic jumps, stochastic volatility and stochastic interest rates formula, developing a DJSVSI model which we can term as a generalized option pricing model under stochastic dynamics. We see here that the Black-Scholes (1973) / Boness (1964) and formulae under stochastic volatility and stochastic interest rates are a special cases of this generalized option pricing model under stochastic dynamics.

The arrival of the events causing a price jump is assumed to follow a Poisson process. Thus, the stochastic differential equation of the asset return process incorporates these proportional jumps and is described under risk neutral probability measure $\mathbb{Q}$ by the following set up given stochastic volatility and stochastic interest rates:

$$dS(t) = S(t)(r(t) - k\lambda)dt + S(t)\sigma(t)d\tilde{w}_s(t) + S(t)(Y_i - 1)d\tilde{N} \qquad 2.4.1$$

$$d\sigma^2(t) = k(\alpha - \sigma^2(t)) + \delta d\tilde{w}_v(t) \qquad 2.4.2$$

$$dr(t) = \theta(\vartheta - r(t)) + \Lambda d\tilde{w}_r(t) \qquad 2.4.3$$

We make use of the compensated poisson process by having the compensator $-\lambda dt$ in the drift term. This gives the desired property that the mean of the increment *dS* remains the same as under the pure diffusion process and under risk neutral probability measure $\mathbb{Q}$.

The poisson process has the following properties and short-term behavior

$$P(N_{dt} = 0) = e^{-\lambda dt} = 1 - \lambda dt + O(dt), \qquad dt \to 0,$$

$$(N_{dt} = 1) = \lambda dt e^{-\lambda dt} \approx \lambda dt, \qquad dt \to 0,$$

$$P(N_{dt} = c) = \frac{(dt)^c \lambda^c}{c!} = O(dt), \qquad dt \to 0,$$

$$P(N_T = n) = e^{-\lambda t} \frac{(\lambda t)^n}{n!}$$

$$P(N_T \leq n) = e^{-\lambda t} \sum_{n=0}^{\lfloor N_T \rfloor} \frac{(\lambda t)^n}{n!}$$

The stock price process, by applying Ito's lemma, has the following solution given stochastic interest rates and stochastic volatility,

$$ln\ \frac{S_T}{S_t} = \int_t^T r ds - k\lambda(T-t) - \frac{1}{2}V_T(T-t) + d\widetilde{w}_s^v\sqrt{E(V_T)} + \sum_{i=1}^{N_T} \ln Y_i$$

$$ln\ \frac{S_T}{S_t} = (R_T - k\lambda - \frac{1}{2}V_T)(T-t) + d\widetilde{w}_s^v\sqrt{E(V_T)} + \sum_{i=1}^{N_T} \ln Y_i \qquad 2.4.4$$

Where $V_T = \frac{1}{T-t}\int_t^T \sigma^2(s)ds$ and $\xi_T = [E(V_T)]^{-\frac{1}{2}}\int_t^T \frac{\sigma(s)d\widetilde{w}_s(t)}{\sqrt{T-t}}$ and $\xi_T \sim N(0,1)$, $d\widetilde{w}_s^v = \xi_T\sqrt{T-t}$ and $d\widetilde{w}_s^v \sim N(0, T-t)$ and that $\ln Y \sim N(\pi, \varpi^2)$.

The stock price is conditionally Poisson mixed log-normally distributed on $\sigma^2(t)$, *r(t)and N(t)* assuming that arrival of jumps is independent from other driving processes such as volatility, interest rates. We have that:

$$\ddot{M}_S(t) = E\left[ln\ \frac{S_T}{S_t}\right] = \left(E[R_T] - k\lambda - \frac{1}{2}E[V_T]\right)(T-t) + \pi\lambda(T-t)$$
$$= \left(M_R(t) - k\lambda + \pi\lambda - \frac{1}{2}M_V(t)\right)(T-t)$$

$$\ddot{M}_S(t) = \left(M_R(t) + \lambda(\pi - k) - \frac{1}{2}M_V(t)\right)(T-t) \qquad 2.4.5$$

Y is drawn at time t_i from a jump-size probability distribution $\mathbb{Q}_J$and the set of Y from successive events is assumed to be independently and identically distributed with $k = E_J^{\mathbb{Q}}(Y-1)$. $(Y-1)$ is the percentage change in the asset price if the Poisson jump event occurs and $E_J^{\mathbb{Q}}$ is the expectation operator taken over the probability distribution of the random variable Y.

$$\ddot{V}_S^2(t) = Var\left[ln\ \frac{S_T}{S_t}\right] = (T-t)^2 Var(R_T) + \frac{(T-t)^2}{4}Var(V_T) + E(V_T)Var(d\widetilde{w}_s^v) - (T-t)\sqrt{E(V_T)}cov(V_T, d\widetilde{w}_s^v)$$
$$+ 2(T-t)\sqrt{E(V_T)}cov(R_T, d\widetilde{w}_s^v) - (T-t)^2 cov(V_T, R_T) + \varpi^2\lambda(T-t)$$

$$\ddot{V}_S^2(t) = Var\left[ln\ \frac{S_T}{S_t}\right] = (T-t)^2 V_R^2(t) + \frac{(T-t)^2}{4}V_V^2(t) + M_V(t)(T-t) - \rho_{V_T, d\widetilde{w}_s^v}(T-t)\sqrt{M_V(t)V_V^2(t)(T-t)} +$$
$$2(T-t)\rho_{R_T, d\widetilde{w}_s^v}\sqrt{M_V(t)V_R^2(t)(T-t)} - (T-t)^2\rho_{V_T,R_T}\sqrt{V_V^2(t)V_R^2(t)} + \varpi^2\lambda(T-t) \qquad 2.4.6$$

The log return is assumed to follow a conditional Poisson mixture of normal distribution as:

$$ln\frac{S_T}{S_t} \sim e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}\emptyset(\ddot{M}_S(t), \ddot{V}_S^2(t))$$

This conditional distribution is leptokurtic (i.e. has fat tails) and therefore, captures an important feature of asset price movements that is not well captured by the simple geometric Brownian motion process.

The solution for the stock price at maturity T becomes:

$$S_T = \prod_{i=1}^{N_T} Y_i S_t e^{(R_T - k\lambda - \frac{1}{2}V_T)(T-t) + d\tilde{w}_S^{\vartheta}\sqrt{E(V_T)}} \qquad 2.4.7$$

The Y_i are assumed to independently and identically distributed, and N_T the number of jumps in (t,T) is again drawn from the Poisson distribution with parameter λ.

By Fubini Theorem, the integrated aggregate interest process becomes:

$$R_T = \frac{1}{T-t}\int_t^T r(s)ds = \frac{(r(t)-\vartheta)D(t,T)}{T-t} + \vartheta + \frac{\Lambda}{T-t}\int_t^T D(u,T)d\tilde{w}_r(u)$$

$$\text{Where } D(t,T) = \frac{1}{\theta}(1 - e^{-\theta(T-t)})$$

The distribution of the integrated interest rate process is conditionally normal with mean and variance,

$$M_R(t) = E[R_T] = E\left[\frac{1}{T-t}\int_t^T r(s)ds\right] = \frac{(r(t)-\vartheta)D(t,T)}{T-t} + \vartheta \qquad 2.4.8$$

$$V_R^2(t) = Var(R_T) = E\left[\frac{\Lambda^2}{(T-t)^2}\int_t^T D(u,T)^2 du\right] = \frac{\Lambda^2}{\theta^2(T-t)^2}\int_t^T (1 - 2e^{-\theta(T-u)} + e^{-2\theta(T-u)})du$$

$$V_R^2(t) = \frac{\Lambda^2}{\theta^2(T-t)^2}\left((T-t) - 2D(t,T) + \frac{1}{2\theta} - \frac{e^{-2\theta(T-t)}}{2\theta}\right) \qquad 2.4.9$$

Similarly by Fubini Theorem, the integrated aggregate variance process becomes:

$$V_T = \frac{1}{T-t}\int_t^T \sigma^2(s)ds = \frac{(\sigma^2(t)-\alpha)B(t,T)}{T-t} + \alpha + \frac{\delta}{T-t}\int_t^T B(u,T)d\tilde{w}_v(u)$$

$$\text{Where } B(t,T) = \frac{1}{k}(1 - e^{-k(T-t)})$$

The distribution of the integrated variance process is conditionally normal with mean and variance,

$$M_V(t) = E[V_T] = E\left[\frac{1}{T-t}\int_t^T \sigma^2(s)ds\right] = \frac{(\sigma^2(t)-\alpha)B(t,T)}{T-t} + \alpha \qquad 2.4.10$$

$$V_V^2(t) = Var(V_T) = E\left[\frac{\delta^2}{(T-t)^2}\int_t^T B(u,T)^2 du\right] = \frac{\delta^2}{k^2(T-t)^2}\int_t^T (1 - 2e^{-k(T-u)} + e^{-2k(T-u)})du$$

$$V_V^2(t) = \frac{\delta^2}{k^2(T-t)^2}\left((T-t) - 2B(t,T) + \frac{1}{2k} - \frac{e^{-2k(T-t)}}{2k}\right) \qquad 2.4.11$$

We define the probability that the option is going to expire in the money given occurrence of n jumps within the options life as:

$$P(S_T > K, N(t) \le n) = P(N(t) \le n)\, P(S_T > K|n)$$

$$= e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}P(S_T > K|n)$$

Given that we have incorporated stochastic volatility and stochastic interest rate in this formulation we have the payoff probability as being:

$$P(S_T > K, N(t) \le n) = e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}P(\{S_T > K|(\sigma_t^2, r_t)\}|n)$$

Where

$$P(\{S_T > K|(\sigma_t^2, r_t)\}|n) = P(\{S_T > K|(V_T, R_T)|(\sigma_t^2, r_t)\}|n) = P(\{\{S_T > K|(V_T, R_T)\}|(\sigma_t^2, r_t)\}|n)$$

By applying the Proposition 2.3.1, we have that

$$P(\{S_T > K|(V_T, R_T)\}|(\sigma_t^2, r_t)\}|n) = \frac{\partial C(F_{S|\sigma_t^2,r_t}(\{S_T > K|(\sigma_t^2, r_t)\}|n),\ F_{V,R|\sigma_t^2 r_t}(V_T, R_T|\sigma_t^2, r_t); \theta)}{\partial F_{V,R|\sigma_t^2 r_t}(V_T, R_T|\sigma_t^2, r_t)}$$

We hereby have that:

$$P(S_T > K, N(t) \le n) = e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}P(\{S_T > K|(V_T, R_T)\}|(\sigma_t^2, r_t)\}|n)$$

$$= e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}\,\frac{\partial C(F_{S|\sigma_t^2,r_t}(\{S_T > K|(\sigma_t^2, r_t)\}|n),\ F_{V,R|\sigma_t^2 r_t}(V_T, R_T|\sigma_t^2, r_t); \theta)}{\partial F_{V,R|\sigma_t^2 r_t}(V_T, R_T|\sigma_t^2, r_t)}$$

Let $\hat{u} = F_{S|\sigma_t^2,r_t}(\{S_T > K|\sigma_t^2, r_t\}|n)$ and $\hat{v} = F_{V,R|\sigma_t^2,r_t}(V_T, R_T|\sigma_t^2, r_t)$, where $\hat{u}$ is the conditional log-normal distribution function for the stock price under stochastic volatility and stochastic interest rate with mean and variance described above; and $\hat{v}$ is the conditional joint

distribution/pair copula function with the respective mean and variance for the volatility process and interest rate process as described above.

We have thus that the probability that the option is going to expire in the money under stochastic volatility, stochastic interest rates and n possible jumps being:

$$P(S_T > K, N(t) \leq n) = e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}\frac{\partial C(\hat{u},\ \hat{v},\theta)}{\partial \hat{v}} = e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}h_{\hat{v}}(\hat{u},\hat{v};\theta)$$

Using the money market $A_t = e^{\int_o^t r(s)ds}$ as the numeraire, which follows the deterministic ordinary differential equation:

$$dA = r(t)A(t)dt$$

The price of a derivative $g(S,t)$ under stochastic volatility, stochastic interest rates and jumps becomes

$$g(S,t) = \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}(S_T - K)\right]$$

$$= \mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}S_T\mathbb{1}_{S_T>K}\right] - \mathbb{E}^{\mathbb{Q}}\left[Ke^{-\int_t^T r(s)ds}\mathbb{1}_{S_T>K}\right]$$

$$= S_t\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\prod_{i=1}^{N_T}Y_i e^{\int_t^T r(s)ds+(-k\lambda-\frac{1}{2}V_T)(T-t)+d\widetilde{w}_s^v\sqrt{E(V_T)}}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] - K\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right]$$

$$= S_t\mathbb{E}^{\mathbb{Q}}\left[\prod_{i=1}^{N_T}Y_i e^{(-k\lambda-\frac{1}{2}V_T)(T-t)+d\widetilde{w}_s^v\sqrt{E(V_T)}}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] - K\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right]$$

Let $X_N = \prod_{i=1}^{N_T}Y_i$, and $\mathbb{E}^{\mathbb{Q}_\lambda,\mathbb{Q}_{y'}}(X_N) = e^{n(\pi+\frac{\varpi^2}{2})} = e^{\lambda(\pi+\frac{\varpi^2}{2})(T-t)}$ and $var(ln\ X_N) = n\varpi^2 = \lambda\varpi^2(T-t)$

Thus we have that

$$= S_t e^{\lambda(\pi-k+\frac{\varpi^2}{2})(T-t)}\mathbb{E}^{\mathbb{Q}}\left[e^{(-\frac{1}{2}V_T)(T-t)+d\widetilde{w}_s^v\sqrt{E(V_T)}}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] - K\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right]$$

Setting $\Sigma_t = \mathbb{E}^{\mathbb{Q}}\left[e^{(-\frac{1}{2}V_T)(T-t)+d\widetilde{w}_s^v\sqrt{E(V_T)}}\right] = e^{-\frac{1}{2}M_V(t)(T-t)+\frac{1}{2}V_S^2(t)}$ which is a constant, where

$$V_S^2(t) = \frac{(T-t)^2}{4} V_V^2(t) + M_V(t)(T-t) - \rho_{V_T, d\tilde{w}_s^v}(T-t)\sqrt{M_V(t)V_V^2(t)(T-t)}$$

We proceed as:

$$= S_t e^{\lambda(\pi - k + \frac{\varpi^2}{2})(T-t)} \Sigma_t \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] - K\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]\mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right]$$

Having $\mathbb{E}^{\mathbb{Q}}\left[e^{-\int_t^T r(s)ds}\right]$ being the current price of a bond P_t and settting $\Pi_{\lambda,t} = e^{\lambda(\pi - k + \frac{\varpi^2}{2})(T-t)}$

$$= S_t \Pi_{\lambda,t} \Sigma_t \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] - KP_t \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] = \hat{Z}_1 - \hat{Z}_2$$

From the definition of the payoff probability and indicator function we have that for $\hat{Z}_2$

$$\hat{Z}_2 = KP_t \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T>K}\right] = KP_t \mathbb{Q}(N(t) \le n)\, \mathbb{Q}(S_T > K|n)$$

$$\hat{Z}_2 = KP_t e^{-\lambda t} \sum_{n=0}^{\lfloor N_T \rfloor} \frac{(\lambda t)^n}{n!} \mathbb{Q}(S_T > K|n) = KP_t e^{-\lambda t} \sum_{n=0}^{\lfloor N_T \rfloor} \frac{(\lambda t)^n}{n!}\, h_v(\hat{u}, \hat{v}; \theta|n)$$

From the definition of $\hat{u} = F_{S|\sigma_t^2, r_t}(\{S_T > K|\sigma_t^2, r_t\}|n)$ above we have that:

$$\hat{u} = F^{\mathbb{Q}}_{S|\sigma_t^2, r_t}\left(\prod_{i=1}^{N_T} Y_i\, S_t e^{(R_T - k\lambda - \frac{1}{2}V_T)(T-t) + d\tilde{w}_s^v\sqrt{E(V_T)}} > K\right)$$

$$\hat{u} = F^{\mathbb{Q}}_{S|\sigma_t^2, r_t}\left(\sum_{i=1}^{N_T} \ln Y_i + R_T(\mathrm{T-t}) - \frac{1}{2}V_T(T-t) + d\tilde{w}_s^v\sqrt{E(V_T)} > \ln\frac{K}{S_t} + k\lambda(T-t)\right)$$

$$\hat{u} = F^{\mathbb{Q}}_{S|\sigma_t^2, r_t}\left(\xi \le \frac{\ln\frac{S_t}{K} + (M_R(t) + \lambda(\pi - k) - \frac{1}{2}M_V(t))(T-t)}{\sqrt{\bar{V}_S^2(t)}}\right) = \phi\left(\hat{d}_2^S\right)$$

Where $\hat{d}_2^S = \frac{\ln\frac{S_t}{K} + (M_R(t) + \lambda(\pi - k) - \frac{1}{2}M_V(t))(T-t)}{\sqrt{\bar{V}_S^2(t)}}$

For the joint distribution (pair copula) function $\hat{v} = F_{V,R|\sigma_t^2, r_t}(V_T, R_T|\sigma_t^2, r_t) = C(a, b, \theta)$

Where for conditional distribution function *a*,

$$a = F^{\mathbb{Q}}_{V|\sigma_t^2}\left(\xi \le \frac{\hat{\sigma}_c^2 - M_V(t)}{\sqrt{V_V^2(t)}}\right) = \phi(d_2^v) \quad \text{where } d_2^v = \frac{\hat{\sigma}_c^2 - M_V(t)}{\sqrt{V_V^2(t)}}$$

Where for conditional distribution function *b*,

$$b = F^{\mathbb{Q}}_{R|r_t}\left(\frac{(r(t)-\vartheta)D(t,T)}{T-t} + \vartheta + \frac{\Lambda}{T-t}\int_t^T D(u,T)d\tilde{w}_r(u) \le r_c\right)$$

$$b = F^{\mathbb{Q}}_{R|r_t}\left(\xi \le \frac{r_c - M_R(t)}{\sqrt{V_R^2(t)}}\right) = \phi(d_2^R) \quad \text{where } d_2^R = \frac{r_c - M_R(t)}{\sqrt{V_R^2(t)}}$$

We thereby have the value for $\hat{Z}_2$ as:

$$\hat{Z}_2 = e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}KP_t h_2\left(\phi\left(\hat{d}_2^S\right); C(\phi(d_2^V), \phi(d_2^R), \theta_2); \theta_1 | n\right)$$

Where we have $h_2\left(\phi\left(\hat{d}_2^S\right); C(\phi(d_2^V), \phi(d_2^R), \theta_2); \theta_1 | n\right)$ as the pair copula h-function with respect to the second argument, i.e, $\ddot{v} = C(\phi(d_2^V), \phi(d_2^R), \theta_2)$ and with dependence parameter θ_1 in the instance that we have n jumps. And P_t is the price of the bond under stochastic interest rate.

For $\hat{Z}_1$, applying Theorem 3.1 (girsanov theorem) and defining the indicator function under the equivalent martingale measure $\mathbb{F}$ which corresponds to the numeraire S, as $\mathbb{E}_{\mathbb{Q}}\left(\mathbb{1}_{S_T>K}\right) = \mathbb{E}_{\mathbb{F}}\left(\eta_T \mathbb{1}_{S_T>K}\right) = \mathbb{F}\,(N(t) \le n)\,\mathbb{F}\,(S_T > K|n)$ where $\mathbb{F}\,(N(t) \le n) = \mathbb{Q}(N(t) \le n)$ as shown by Theorem 2.8.1 below .

Theorem 2.4.1: Girsanov Theorem for jump processes

Let $(N, \mathbb{F}_{\lambda_1})$ *and* $(N, \mathbb{Q}_{\lambda_2})$ *be Poisson processes on* (Ω, F_T) *with intensities* λ_1 *and* λ_2 *and jump sizes* a_1 *and* a_2. *If* $a_1 = a_2$ *then* $\mathbb{F}_{\lambda_1}$ *is equivalent to* $\mathbb{Q}_{\lambda_2}$ *with Radon-Nikodym density:*

$$\frac{d\mathbb{F}_{\lambda_1}}{d\mathbb{Q}_{\lambda_2}} = exp\left[(\lambda_2 - \lambda_1)T - N_T \ln\frac{\lambda_2}{\lambda_1}\right]$$

Else if $a_1 \neq a_2$ *then the measures* $\mathbb{F}_{\lambda_1}$ *and* $\mathbb{Q}_{\lambda_2}$ *are not equivalent.*

The result above of *Theorem 2.4.1* means that changing the intensity of jumps amounts to "reweighting" the probabilities on paths: no new paths are generated by simply shifting the intensity. The intensity of a Poisson process can be modified without changing the "support" of the process, but changing the size of jumps generates a new measure which assigns nonzero probability to some events which were impossible under the old one (Cont &Tankov,2004).

From the Theorem 2.8.1 above, setting $\lambda_2 = \lambda_1$ we have that

$$\frac{d\mathbb{F}_{\lambda_1}}{d\mathbb{Q}_{\lambda_2}} = exp[-N_T \ln 1] = 1$$

Thus, given assumption that $\lambda_2 = \lambda_1$ then $d\mathbb{F}_{\lambda_1} = d\mathbb{Q}_{\lambda_2}$.

We proceed and show that the value of $\hat{Z}_1$ becomes

$$\hat{Z}_1 = S_t \Pi_{\lambda,t} \Sigma_t \mathbb{E}^{\mathbb{Q}}\left[\mathbb{1}_{S_T > K}\right]$$
$$\hat{Z}_1 = \mathrm{S_t}\, \Pi_{\lambda,t} \Sigma_t \mathbb{E}_{\mathbb{F}}\left(\eta_T \mathbb{1}_{\mathrm{S_T > K}}\right)$$

$$\hat{Z}_1 = \mathrm{S_t} \Pi_{\lambda,t} \Sigma_{\mathrm{t}} \mathbb{F}\,(N(t) \le n)\,\mathbb{F}\,(S_T > K|n)$$

$$\hat{Z}_1 = \mathrm{S_t} \Pi_{\lambda,t} \Sigma_{\mathrm{t}} e^{-\lambda t} \sum_{n=0}^{\lfloor N_T \rfloor} \frac{(\lambda t)^n}{n!}\,\mathbb{F}\,(S_T > K|n)$$

$$\hat{Z}_1 = \mathrm{S_t} \Pi_{\lambda,t} \Sigma_{\mathrm{t}} e^{-\lambda t} \sum_{n=0}^{\lfloor N_T \rfloor} \frac{(\lambda t)^n}{n!}\,\mathrm{h_v}(\tilde{u}, \tilde{v}; \theta)$$

Where $\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2, r_t}((S_T > K|\sigma_t^2, r_t)|n)$ and $\tilde{v} = F^{\mathbb{F}}_{V,R|\sigma_t^2, r_t}((V_T, R_T|\sigma_t^2, r_t)|n) = C(\mathrm{a}, b, \theta_2)$ are the conditional distribution functions under risk neutral measure $\mathbb{F}$.

Setting $d\tilde{w}_s^{v,\mathbb{Q}} = d\tilde{w}_s^{v,\mathbb{F}} + \sqrt{E(V_T)}dt$

$$ln\,\frac{S_T}{S_t} = (R_T - k\lambda - \frac{1}{2}V_T)dt + d\tilde{w}_s^{v,\mathbb{Q}}\sqrt{E(V_T)} + \sum_{i=1}^{N_T} \ln Y_i$$

$$ln\,\frac{S_T}{S_t} = (R_T - k\lambda - \tfrac{1}{2}V_T)dt + \left(d\tilde{w}_s^{v,\mathbb{F}} + \sqrt{E(V_T)}dt\right)\sqrt{E(V_T)} + \sum_{i=1}^{N_T} \ln Y_i$$

$$ln\,\frac{S_T}{S_t} = (R_T - k\lambda + E(V_T) - \tfrac{1}{2}V_T)dt + d\tilde{w}_s^{v,\mathbb{F}}\sqrt{E(V_T)} + \sum_{i=1}^{N_T} \ln Y_i$$

We have that the conditional distribution function $\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2, r_t}(S_T > K|\sigma_t^2, r_t)$,

$$\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2 r_t}(\prod_{i=1}^{N_T} Y_i\, S_t e^{(R_T - k\lambda + E(V_T) - \frac{1}{2}V_T)(T-t) + d\tilde{w}_s^{v,\mathbb{F}}\sqrt{E(V_T)}} > K)$$

$$\tilde{u} = F^{\mathbb{F}}_{S|\sigma_t^2, r_t}\left(\sum_{i=1}^{N_T} \ln Y_i + R_T(\mathrm{T} - \mathrm{t}) - \tfrac{1}{2}V_T(T-t) + d\tilde{w}_s^{v,\mathbb{F}}\sqrt{E(V_T)} > \ln\tfrac{K}{S_t} + k\lambda(T-t) - E(V_T)(T-t)\right)$$

$$\tilde{u} = F^{\mathbb{F}}_{S|\sigma^2_{t,}r_t}\left(\xi \leq \frac{\ln\frac{S_t}{K}+M_R(t)(T-t)+\lambda(\pi-k)(T-t)+\frac{1}{2}M_V(t)(T-t)}{\sqrt{\bar{V}^2_S(t)}}\right)$$

$$u = \phi\left(\hat{d}_1^S\right) \text{ where } \hat{d}_1^S = \frac{\ln\frac{S_t}{K}+\left(M_R(t)+\lambda(\pi-k)+\frac{1}{2}M_V(t)\right)(T-t)}{\sqrt{\bar{V}^2_S(t)}}$$

We have for the joint/pair copula distribution function $\hat{v} = F^{\mathbb{F}}_{V,R|\sigma^2_t,r_t}(V_T, R_T|\sigma^2_t, r_t) = C(\mathrm{a}, b, \theta_2)$, where $a = F^{\mathbb{F}}_{V|\sigma^2_t}(V_T|\sigma^2_t)$ and $b = F^{\mathbb{F}}_{R|r_t}(R_T|r_t)$. Thus for the stochastic volatility process, setting that $d\tilde{w}^{\mathbb{Q}}_v(t) = d\tilde{w}^{\mathbb{F}}_v(t) + \delta dt$ and under probability measure $\mathbb{F}$

$$d\sigma^2(t) = k(\alpha - \sigma^2(t))dt + \delta(d\tilde{w}^{\mathbb{F}}_v(t) + \delta dt)$$

$$d\sigma^2(t) = k(\beta - \sigma^2(t))dt + \delta d\tilde{w}^{\mathbb{F}}_v(t) \quad \text{where } \beta = \alpha + \frac{\delta^2}{k}$$

Similarly setting $d\tilde{w}^{\mathbb{Q}}_r(t) = d\tilde{w}^{\mathbb{F}}_r(t) + \Lambda dt$ for the stochastic interest rates process:

$$dr(t) = \theta(\omega - r(t))dt + \Lambda d\tilde{w}^{\mathbb{F}}_r(t) \quad \text{Where } \omega = \vartheta + \frac{\Lambda^2}{\theta}$$

The conditional distribution function a under probability measure $\mathbb{F}$

$$a = F^{\mathbb{F}}_{V|\sigma^2_t}\left(\frac{\left(\sigma^2(t)-\beta\right)B(t,T)}{T-t} + \beta + \frac{\delta}{T-t}\int_t^T B(u,T)d\tilde{w}_v(u) \leq \hat{\sigma}^2_c\right)$$

$$a = F^{\mathbb{F}}_{V|\sigma^2_t}\left(\xi \leq \frac{\hat{\sigma}^2_c - M_V(t) + \frac{\delta^2}{k(T-t)}\left(B(t,T) - \frac{\alpha k}{\delta^2} - 1\right)}{\sqrt{V^2_V(t)}}\right)$$

$$v = \phi(d_2^V) \quad \text{Where } d_1^v = \frac{\hat{\sigma}^2_c - M_V(t) + \frac{\delta^2}{k(T-t)}\left(B(t,T) - \frac{\alpha k}{\delta^2} - 1\right)}{\sqrt{V^2_V(t)}}$$

The conditional distribution function b under probability measure $\mathbb{F}$

$$b = F^{\mathbb{F}}_{R|r_t}\left(\frac{(r(t)-\omega)D(t,T)}{T-t} + \omega + \frac{\Lambda}{T-t}\int_t^T D(u,T)d\tilde{w}_r(u) \leq r_c\right)$$

$$b = F^{\mathbb{F}}_{R|r_t}\left(\xi \leq \frac{r_c - M_R(t) + \frac{\Lambda^2}{\theta(T-t)}\left(D(t,T) - \frac{\vartheta\theta}{\Lambda^2} - 1\right)}{\sqrt{V^2_R(t)}}\right)$$

$$v = \phi(d_2^V) \quad \text{Where } d_1^R = \frac{r_c - M_R(t) + \frac{\Lambda^2}{\theta(T-t)}\left(D(t,T) - \frac{\vartheta\theta}{\Lambda^2} - 1\right)}{\sqrt{V^2_R(t)}}$$

Thus the joint (pair copula) distribution function $\mathrm{C}(\mathrm{a},\mathrm{b},\theta_2)$ becomes $\mathrm{C}(\phi(\mathrm{d}_1^{\mathrm{R}}),\phi(\mathrm{d}_1^{\mathrm{V}}),\theta_2|\mathrm{n})$

We thereby have the value for $\hat{Z}_1$as:

$$\hat{Z}_1 = \mathrm{S_t}\Pi_{\lambda,t}\Sigma_{\mathrm{t}}e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}h_2\left(\phi\left(\hat{d}_1^S\right);C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)$$

The price of a European call option by the dynamic approach under jumps, stochastic volatility and stochastic interest rate is:

$$g(S,t) = \mathrm{S_t}\Pi_{\lambda,t}\Sigma_{\mathrm{t}}e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}h_2\left(\phi\left(\hat{d}_1^S\right);C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)$$
$$- e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}KP_t h_2\left(\phi\left(\hat{d}_2^S\right);C(\phi(d_2^V),\phi(d_2^R),\theta_2);\theta_1|n\right)$$

We can simplify this to have the price of a European call option under jumps, stochastic volatility and stochastic interest rates as:

$$g(S,t) = \mathrm{S_t}\Pi_{\lambda,t}\,\Sigma_{\mathrm{t}}\,F(n;\lambda)h_2\left(\phi\left(\hat{d}_1^S\right);C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)$$
$$- KP_t F(n;\lambda)h_2\left(\phi\left(\hat{d}_2^S\right);C(\phi(d_2^V),\phi(d_2^R),\theta_2);\theta_1|n\right)$$

2.7.12

Where $F(n;\lambda)$ is the poisson cumulative distribution function with number of jumps n and arrival of jumps rate is λ and $\Pi_{\lambda,t} = e^{\lambda(\pi-k+\frac{\varpi^2}{2})(T-t)}$ and $\Sigma_t = e^{-\frac{1}{2}M_V(t)(T-t)+\frac{1}{2}V_S^2\,(t)}$ and h_2 is the copula h-function and P_t is the current price of a bond under stochastic interest rates.

Using alternative notation we can have the formula above as being

$$g(S,t) = e^{-\lambda t}\sum_{n=0}^{\lfloor N_T \rfloor}\frac{(\lambda t)^n}{n!}\mathbb{E}^n\left[\mathrm{S_t}\Pi_{\lambda,t}\Sigma_{\mathrm{t}}h_2\left(\phi\left(\hat{d}_1^S\right);C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1\right)\right.$$
$$\left. - KP_t h_2\left(\phi\left(\hat{d}_2^S\right);C(\phi(d_2^V),\phi(d_2^R),\theta_2);\theta_1\right)\right]$$

This formula under jumps, stochastic volatility and stochastic interest rates expresses the European call option value as a Poisson weighted sums of n-fold expectations of Dynamic Stochastic Volatility-Stochastic Interest rates (DSVSI) model values conditional on n-jumps. When we

restrict the pricing formulation of equation 2.7.12, we can obtain the pricing formula for the conditions of stochastic interest rates, jumps and stochastic volatility respectively to yield pricing models under the various restrictions.

2.4.1 Risk Sensitivities under jumps, stochastic volatility and stochastic interest rates (DJSVSI)

Below, we present closed-form formulas for delta, gamma, dependence delta and dependence gamma for European call options under jumps, stochastic volatility and stochastic interest. The results for can similarly be obtained for DSV, DSI, DSVSI, DJSV, DJSI models by restricting the relevant parameters.

a) Delta (Call Option)

$$\frac{\partial C(S,t)}{\partial S} = \Pi_{\lambda,t}\,\Sigma_{\mathrm{t}}\,KP_t F(n;\lambda)\left[\frac{h_2\left(\phi\left(\hat{d}_1^S\right); C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)}{KP_t}\right.$$
$$+\frac{c\left(\phi\left(\hat{d}_1^S\right); C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right) f\left(\hat{d}_1^S\right)}{KP_t\sqrt{\ddot{V}_S^2}}$$
$$\left.-\frac{c\left(\phi\left(\hat{d}_2^S\right); C(\phi(d_2^R),\phi(d_2^V),\theta_2);\theta_1|n\right) f\left(\hat{d}_2^S\right)}{S_t\Pi_{\lambda,t}\,\Sigma_{\mathrm{t}}\sqrt{\ddot{V}_S^2}}\right]$$

b) Gamma (Call Option)

$$\frac{\partial^2 C(S,t)}{\partial S^2} = \frac{\Pi_{\lambda,t}\,\Sigma_{\mathrm{t}}\,F(n;\lambda)}{S_t\ddot{V}_S^2}\left[f\left(\hat{d}_1^S\right)\sqrt{\ddot{V}_S^2}\,c\left(\phi\left(\hat{d}_1^S\right); C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)\right.$$
$$\left.+\,f\left(\hat{d}_1^S\right)^2\frac{\partial c}{\partial\phi\left(\hat{d}_1^S\right)}-\hat{d}_1^S f\left(\hat{d}_1^S\right)c\left(\phi\left(\hat{d}_1^S\right); C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)\right]$$
$$-\frac{KP_tF(n;\lambda)}{S_t^2\ddot{V}_S^2}\left[f\left(\hat{d}_2^S\right)^2\frac{\partial c}{\partial\phi\left(\hat{d}_2^S\right)}\right.$$
$$\left.-\hat{d}_2^S f\left(\hat{d}_2^S\right)c\left(\phi\left(\hat{d}_2^S\right); C(\phi(d_2^R),\phi(d_2^V),\theta_2);\theta_1|n\right)\right]$$

c) Dependence Delta

We have that the sensitivity of the option price to changes in the dependence assuming jumps, stochastic volatility and stochastic interest rates is given by:

$$\frac{\partial C(S,t)}{\partial \theta_1} = \mathrm{S_t}\Pi_{\lambda,t}\,\Sigma_{\mathrm{t}}\,F(n;\lambda)\frac{\partial h_2\left(\phi\left(\hat{d}_1^S\right);C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)}{\partial \theta_1} - KP_tF(n;\lambda)\frac{\partial h_2\left(\phi\left(\hat{d}_2^S\right);C(\phi(d_2^V),\phi(d_2^R),\theta_2);\theta_1|n\right)}{\partial \theta_1}$$

Where $\frac{\partial h_2}{\partial \theta_1}$ is the first derivative of the h –function with respect to the dependence parameter θ_1.

d) Dependence Gamma

The convexity of the option price to changes in the dependence assuming jumps, stochastic volatility and stochastic interest rate is given by:

$$\frac{\partial^2 C(S,t)}{\partial {\theta_1}^2} = \mathrm{S_t}\Pi_{\lambda,t}\,\Sigma_{\mathrm{t}}\,F(n;\lambda)\frac{\partial^2 h_2\left(\phi\left(\hat{d}_1^S\right);C(\phi(d_1^R),\phi(d_1^V),\theta_2);\theta_1|n\right)}{\partial {\theta_1}^2} - KP_tF(n;\lambda)\frac{\partial^2 h_2\left(\phi\left(\hat{d}_2^S\right);C(\phi(d_2^V),\phi(d_2^R),\theta_2);\theta_1|n\right)}{\partial {\theta_1}^2}$$

Where $\frac{\partial^2 h_2}{\partial {\theta_1}^2}$ is the second derivative of the h –function with respect to the dependence parameter θ_1.

2.5 Empirical Results: Dynamic Option Pricing under Stochastic Volatility

This section examines the relative empirical performance of the models. The data used was of daily closing prices obtained from *Thomson Reuters DataStream* for the S&P 500 Index Call Options of maturity of 3 Years with expiry of 18th December 2019 and 70 Strikes price levels from 100 to 4200 USD. The underlying S&P 500 Index Prices were obtained for a 5-year period from 19-August-2015 to 19-August-2020. Table 2.1 reports the summary statistics of the data used in the analysis and Figure 2.1 plots the S&P Index Daily Prices from 19-August-2015 to 19-August-2020.

Table 2.1: Summary Statistics of S&P 500 Index Prices, log-returns and stochastic variance from 19-August-2015 to 19-August-2020

	Min	Median	Mean	Max	Std. Dev	Skewness	Kurtosis
S&P 500 Index Prices	1829	2636	2575	3390	388.6346	-0.004384	2.003978
Daily Log-returns S&P 500 Index	-0.1277	0.0003734	0.0003710	0.08968	0.01206	-1.0909	25.1340
Stochastic Variance	0.0000	0.01554	0.02361	0.1092	0.02337	1.1193	3.4949

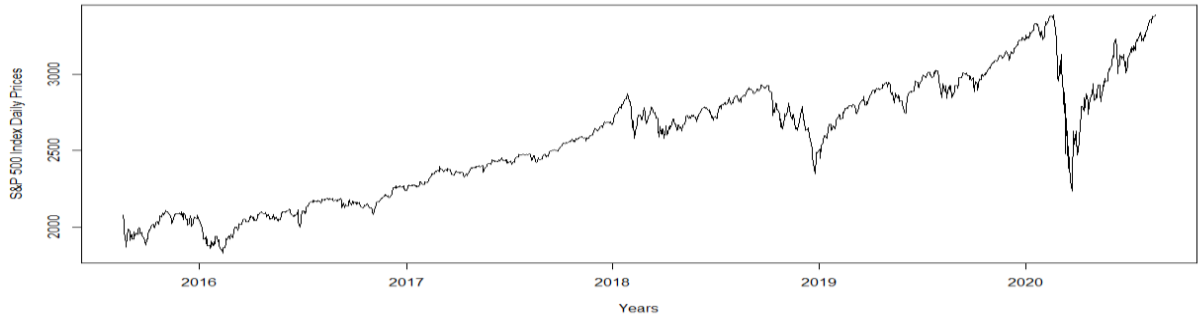

Figure 2.1: Plot of S&P Index Daily Prices from 19-August-2015 to 19-August-2020

In this analysis, the following pair copula h-functions are fitted in addition to the empirically estimated Joe Copula, these are the Gaussian, Student, Clayton, Gumbel and Frank copula specifications. The two models applied to European option pricing the Gaussian mean-reverting stochastic volatility model and the mean-reverting square-root stochastic volatility model are analysed. Table 2.2 present summaries of the estimated parameters of the stochastic volatility process under Gaussian mean-reversion and square-root mean reversion, the moments of the

integrated variance process and parameter estimates of the copula h-function with the empirically fitted copula specification being the Joe Copula, and the other specifications fitted on assumption of dependence structure between the underling price process and the stochastic volatility in a bid to understand the dynamism of the developed models. Figure 2.2 present Plots of S&P Index Daily Prices and the S&P Index Stochastic Volatility from 19-August-2015 to 19-August-2020. The empirical copula dependence plot and the empirical integrated variance cumulative distribution plots are presented in Figure 2.3 and Figure 2.4 next.

Table 2.2: Parameters of Stochastic Volatility and Estimated Empirical Copula Dependence

Stochastic Volatility Parameter Estimation		**OU Stoch. Vol**	**Feller Stoch. Vol.**
Parameters	k	2.974829	1.20891
	α	0.02658892	0.03473743
	δ	0.06323379	0.4013498
Integral Moments	$\hat{V}_c$	0.01903415	0.01903415
	M_V	0.02340127	0.02364749
	V_V	0.000187169	0.0004108785
Copula h-function Parameter Estimation	**Joe Copula**	4.885082 (0.17)	
	Gaussian Copula	0.6546281 (0.02)	
	Student t	0.7850045 (0.02) 2.101107 (0.19)	
	Clayton	0.5813352 (0.06)	
	Gumbel	2.557011 (0.08)	
	Frank	6.930227 (0.32)	

Table 2.2: The table presents the estimated parameters *k*, α and δ of the stochastic volatility processes and the corresponding integrated process moments - the mean M_V , the variance V_V and the aggregated estimate $\hat{V}_c$ of the integrated variance process across time-to-expiry horizons. The empirical joe copula parameter θ and the other fitted copula parameters are also presented with their respective standard errors shown in brackets.

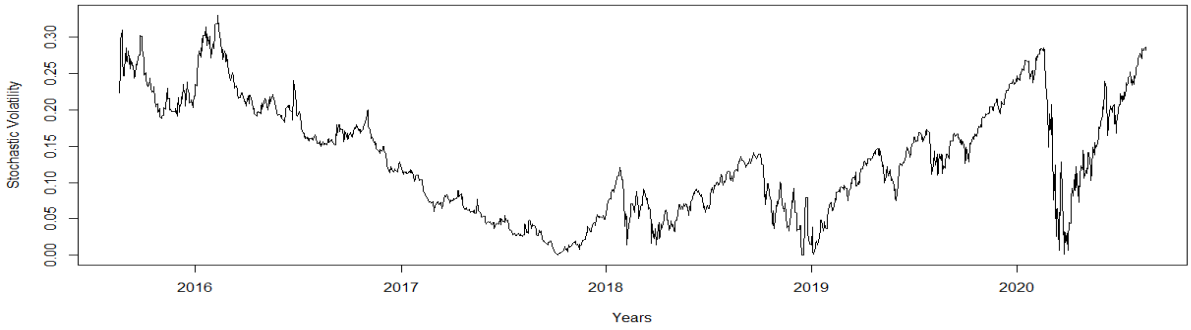

Figure 2.2: S&P Index Stochastic Volatility Plot from 19-August-2015 to 19-August-2020

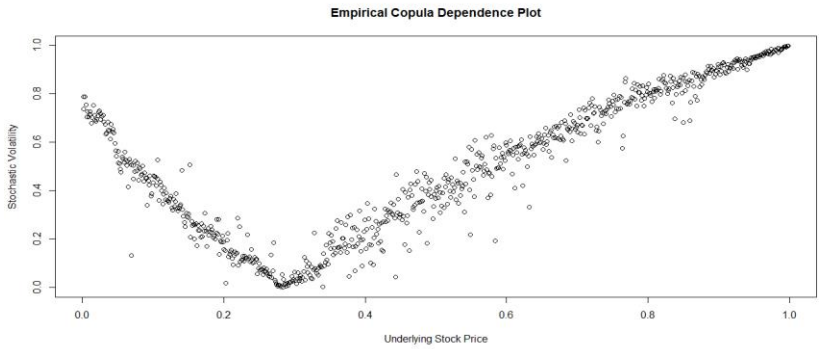

Figure 2.3: Empirical Copula Dependence Plot of Stochastic Volatility and Underlying Stock Prices

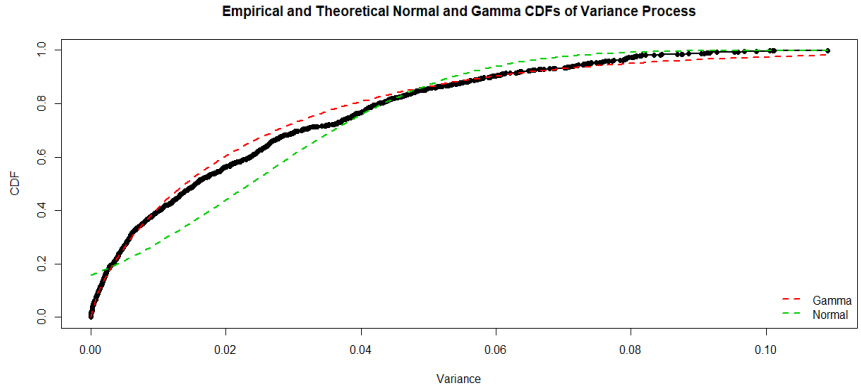

Figure 2.4: Empirical Vs Normal and Gamma Cumulative Distribution Functions of Stochastic Variance Process .

Figure 2.4: Figure shows the Empirical integrated variance cumulative distribution function in comparison to the fitted theoretical gamma and normal cumulative distribution functions displayed in dashed red and dashed green respectively. As noted also by Prayoga and Privault (2017) that both the marginal and conditional density of the integrated square-root process are approximated better by a gamma density than a lognormal approximation. They showed that in their implementation for yield options in the CIR model, the stratified gamma approximation performs consistently better than the lognormal approximations.

As shown in Figure 2.4, it is evidenced that the empirical cumulative distribution of the integrated variance process is asymptotically approximate to the gamma cumulative distribution function rather than a Gaussian distribution. The gamma approximation of the integrated variance process distribution was thereby found to be empirically more accurate than the normal approximation for the integrated variance cumulative distribution. This impacted the performance of the dynamic models, as seen in the Table 2.3 and Table 2.4 presenting summaries of the Pricing

Errors by RMSE and the relative percentage performance improvements obtained by the Dynamic Pricing Models under Gaussian mean-reverting stochastic volatility and square-root mean-reverting stochastic volatility given different copula h-function specifications. Findings reveal that the pricing errors are reduced by the Heston Model by 3% across strike prices with respect to the Black-Scholes model, generally the Heston Model specification improves the pricing performance by 18% and 2% for OTM and ATM options above the Black-Scholes model.

Performance improvement for ITM call options by Heston model is not witnessed. The dynamic stochastic volatility model with mean-reverting Gaussian stochastic volatility (DSV1) improves the pricing performance by 5% across moneyness levels over the Heston Model. Specifically, performance is improved by this model under the Joe (empirical) h-function specification by 6%, 5% and 2% for ITM, ATM and OTM call options over the Heston Model. Given Gaussian copula specification the DSV 1 model, reduces the pricing errors by an aggregate of 12.5% across moneyness and for ITM, ATM and OTM by 5.9%, 3.2% and 46.5% respectively. The dynamic stochastic volatility model with mean-reverting square-root stochastic volatility (DSV2) improves pricing performance by 13% over the Heston model. More particularly, 10.6%, 9.8% and 25.4% performance improvement for ITM, ATM and OTM respectively for DSV 2 model under empirical joe copula h-function over the Heston Model. This pricing model with mean-reverting square-root stochastic volatility under Frank copula specification improves the pricing performance on aggregate by 14.8% across moneyness and for ITM, ATM and OTM call options the pricing performance is improved by 10.5%, 10.6% and 36%. These results are in Table 2.3 and Table 2.4.

Table 2.3: Relative Percentage (%) Performance Improvement in RMSE of Dynamic Option Pricing Models under Stochastic Volatility

Moneyness	DSV Model 1 (Joe Copula)	DSV Model 1 (Gaussian Copula)	DSV Model 2 (Joe Copula)	DSV Model 2 (Frank Copula)	Black-Scholes Model
ALL	5.29%	12.54%	13.04%	14.83%	-3.10%
ITM	6.05%	5.89%	10.60%	10.49%	0.13%
ATM	5.03%	3.21%	9.80%	10.64%	-1.69%
OTM	2.05%	46.55%	25.40%	35.98%	-17.92 %

Table 2.3: The table reports the relative percentage (%) performance improvement in RMSE of Dynamic Option Pricing Models under best performing copula configurations against the Heston (1993) Model across ALL moneyness levels and ITM, ATM and OTM levels. The relative percentage performance improvements are obtained from summarizing deviations from RMSE values obtained from the models as indicated in Table 2.4 below.

The daily prices of European call options were obtained for a 3-year maturity period from 19-December-2016 to 18-December 2019. These prices of the European call options by the developed models are aggregate across moneyness and summarized in Table 2.4 in comparison to the actual prices. The standard deviation of the price estimates are also summarized across moneyness. OTM options are seen to have the lowest prices variation under the dynamic pricing model with square-root stochastic volatility (DSV2) and for ITM call options have the highest prices variation for this model. This is also the case for the other dynamic pricing model with Gaussian mean-reverting stochastic volatility as well as the Heston and Black-Scholes models. Generally, across moneyness levels, the dynamic pricing model with square-root stochastic volatility given empirical copula h-function specification has the lowest price variation.

Table 2.4: Dynamic Stochastic Volatility Option Pricing Models under Empirical Joe Copula h-function Performance Comparison across Moneyness for Call Options (Contracts =70, Total Observations =35000)

	Moneyness	Obs	Dynamic SV Model 1 (Empirical Joe Copula h-function)	Dynamic SV Model 1 (Gaussian Copula)	Dynamic SV Model 2 (Empirical Joe Copula h-function)	Dynamic SV Model 2 (Frank Copula)	Heston (1993) Model	Black-Scholes Model
Mean	ALL	35000	60.88	56.22	55.90	54.75	64.28	66.27
	ITM	21000	74.63	74.76	71.02	71.11	79.44	79.34
	ATM	6500	29.27	29.83	27.80	27.54	30.82	31.34
	OTM	7500	49.79	27.17	37.92	32.54	50.83	59.94
Std. Dev	ALL	35000	22.62	26.26	23.08	24.67	24.46	24.01
	ITM	21000	16.40	16.18	16.21	16.06	16.50	16.55
	ATM	6500	4.37	5.50	3.60	4.60	6.11	5.59
	OTM	7500	9.50	4.75	8.92	14.43	14.00	18.04
Min	ALL	35000	24.80	23.44	24.29	23.25	24.82	25.71
	ITM	21000	40.71	41.97	37.79	38.60	45.61	44.91
	ATM	6500	24.80	23.63	24.29	23.25	24.82	25.71
	OTM	7500	32.27	23.44	29.54	23.53	29.16	32.38
Max	ALL	35000	95.32	95.32	91.59	91.59	100.15	100.15
	ITM	21000	95.32	95.32	91.59	91.59	100.15	100.15
	ATM	6500	38.31	39.83	35.53	36.48	42.49	42.38
	OTM	7500	59.46	38.72	63.68	69.24	71.20	86.90

Table 2.4: The table reports the pricing errors (RMSE) of the dynamic option pricing models under the empirical (joe) copula configuration and estimated lowest pricing errors copula configurations (i.e Gaussian copula for model under gaussian stochastic volatility and Frank copula for model under square-root stochastic volatility). The pricing errors are compared across ITM, ATM, OTM and ALL moneyness levels. The mean, min, max and std. dev of the pricing errors within moneyness levels are also summarized. A total of 70 S&P 500 Index European call options contracts are used, with total observations of 35000.

Table 2.5: Dynamic Stochastic Volatility Option Pricing Models Price Comparison across Moneyness for Call Options under empirical copula h-function (Contracts =70, Total Observations =35000)

		Obs	DSV Model 1 (Empirical Copula)	DSV Model 2 (Empirical Copula)	Heston Model	Black-Scholes Model	Actual Price
Mean Price	ALL	35000	1095.849	1099.17	1099.58	1097.83	1061.76
	ITM	21000	1466.148	1462.69	1470.96	1470.92	1397.30
	ATM	6500	709.5949	707.09	714.64	713.84	699.91
	OTM	7500	393.7664	421.10	393.31	385.94	435.87
Price Std. Dev	ALL	35000	170.31	166.59	169.82	171.66	177.80
	ITM	21000	178.73	180.47	177.61	177.62	196.48
	ATM	6500	170.64	171.10	169.24	170.25	166.34
	OTM	7500	146.46	123.81	148.50	156.18	135.42

Table 2.5: The table reports the estimated S&P 500 Index European call option prices obtained by the Dynamic model under gaussian mean-reverting stochastic volatility and the Dynamic model under square-root stochastic volatility under empirical (joe) copula configurations. These estimates are compared to those obtained by the Heston and Black-Scholes Models. The actual price is also indicated. These prices are aggregated across moneyness levels, with a total of 70 S&P 500 Index European option contracts and 35000 prices estimated.

In Figure 2.5 as shown, the pricing errors are compared for the dynamic option pricing model under Gaussian mean-reverting stochastic volatility (DSV Model 1) with copula h-function specifications of Gaussian, Student, Gumbel, Frank, Joe and Clayton types. The pricing errors are evaluated across strike prices and moneyness levels. Generally, this model performs consistently better than the Heston and Black-Scholes models across moneyness for all copula h-function definitions expect the Clayton specification under which it does not outperform for the ATM call options only. This model outperforms consistently the Heston model for all ITM and OTM call option under all copula h-functions definitions.

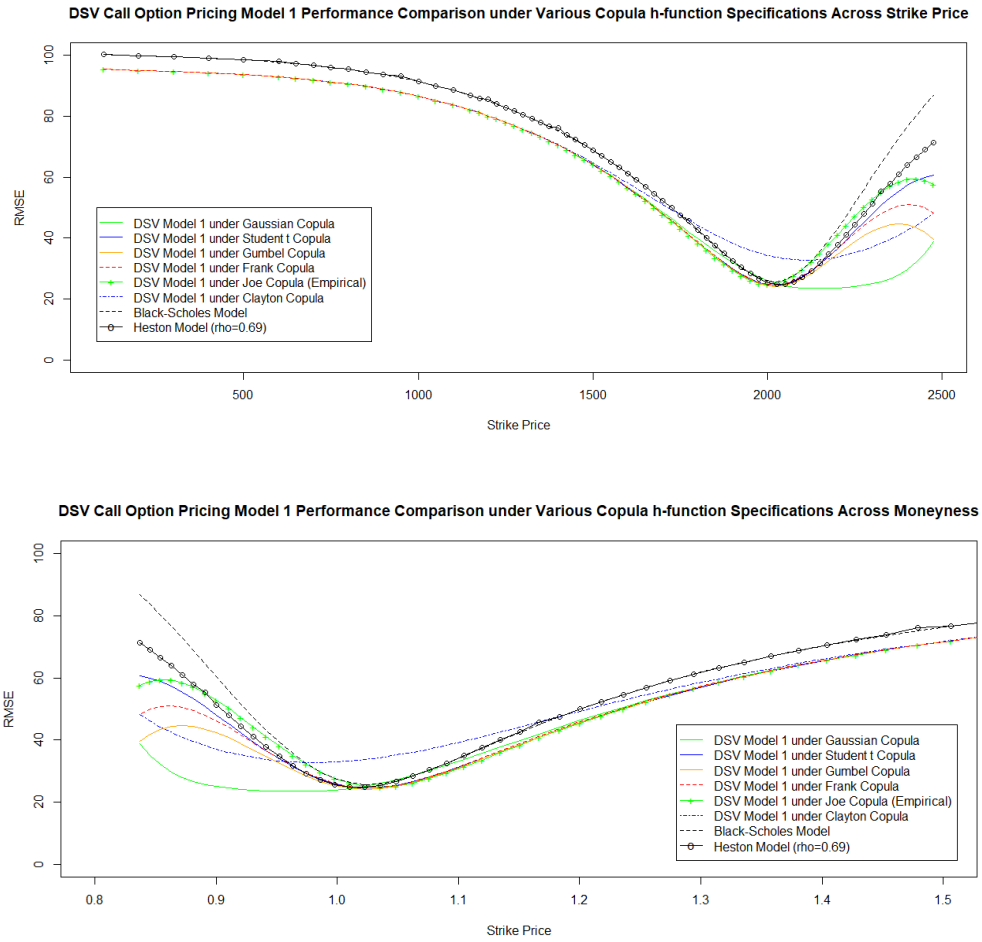

Figure 2.5 (a), (b): Dynamic option pricing model under Gaussian mean-reverting stochastic volatility pricing performance comparison (by RMSE) under various copula h-function configurations across (a) Strike Price and (b) Moneyness in comparison with Black-Scholes and Heston Model

In Figure 2.6, the pricing errors are compared for the dynamic option pricing model under square-root mean-reverting stochastic volatility (DSV Model 2) with copula h-function specifications of Gaussian, Student, Gumbel, Frank, Joe and Clayton types. The pricing errors are evaluated across strike prices and moneyness levels. Generally, this model performs consistently better than the Heston and Black-Scholes models across moneyness for all copula h-function definitions expect the Clayton and Gaussian specification under which it does not outperform for the ATM call options only. This model outperforms consistently the Heston model for all ITM and OTM call option under all copula h-functions definitions.

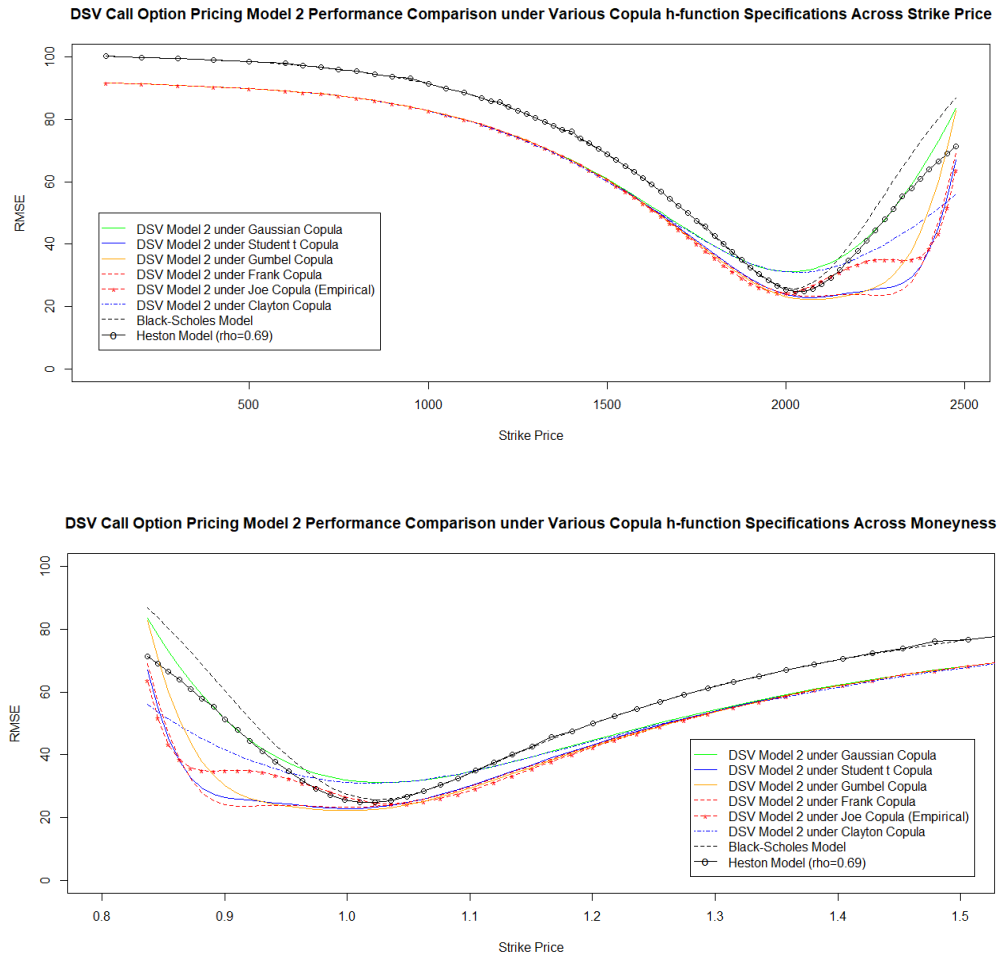

Figure 2.6 (a),(b): Dynamic option pricing model under mean-reverting square-root stochastic volatility pricing performance comparison (by RMSE) under various copula h-function configurations across (a) Strike price and (b) Moneyness in comparison with Black-Scholes and Heston Models

Variation in performance by various copula h-function specifications was majorly observed for OTM call options. ITM call options pricing errors were uniform under these various copula h-function specifications for both the dynamic models configurations. As seen in Figure 2.7, the dynamic option pricing model under square-root mean-reverting stochastic volatility (DSV 2) consistently outperforms the dynamic option pricing model under Gaussian mean-reverting stochastic volatility (DSV 1) configuration across all moneyness levels given the specification of the copula h-function being the empirical one of Joe type. This dynamic model under square-root configuration outperforms consistently the dynamic model under Gaussian stochastic volatility configuration, the Heston and the Black-Scholes models for all ITM and OTM call options.

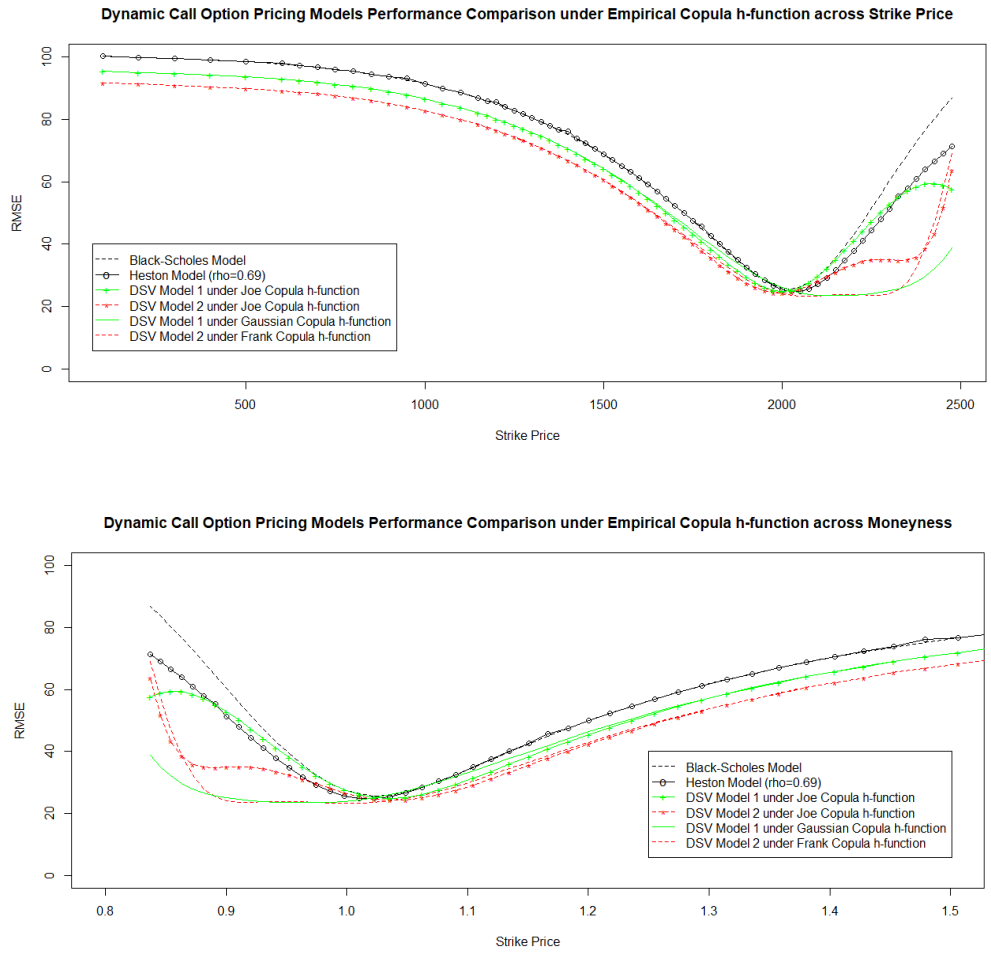

Figure 2.7 (a),(b): Dynamic option pricing models pricing performance comparison (by RMSE) under empirical Joe Copula h-function and best performing copula configurations (i.e Gaussian copula for model under Gaussian mean-reverting stochastic volatility and frank copula for model under mean-reverting square-root stochastic volatility) across (a) Strike price and (b) Moneyness levels

2.5.1 Discussions: Dynamic Option Pricing under Stochastic Volatility

The objective of this section was to develop a novel closed-form European option pricing formula under stochastic volatility by apply dynamic conditional copula techniques for the payoff probability definition and evaluate its empirical performance given mean-reverting gaussian stochastic volatility and mean-reverting square-root stochastic volatility configurations. The obtained pricing errors (RMSE) for the 3-year European Call options obtained by the constructed dynamic models under the two stochastic volatility configurations gaussian mean-reversion and square-root mean-reversion - indicate that in comparison to the Heston (1993) and BlackScholes (1973) models, the dynamic models were found to perform significantly better consistently across

moneyness levels and as well as in various copula h-functions specifications – An average of 6% and 10% performance improvement for ITM options and 5% and 9% for ATM options for the respective stochastic volatility configurations across copula h-function specifications was obtained. ITM and ATM call options pricing errors were uniform under these various copula h-function specifications for both the dynamic models configurations. Variation in performance improvement by the various copula h-function specifications was majorly observed for OTM call options with a range from 2% (min) by Joe copula to 47% (max) by Gaussian copula for OTM option under mean-reverting gaussian stochastic volatility; and under meanreverting square-root stochastic volatility - a performance improvement from up to 36% by the Frank copula h-function for OTM options. This exposed presence of copula specific performance variability for OTM options and copula invariability for ITM and ATM options.

Broadly, the dynamic model under mean-reverting square root configuration performs better than that under the mean-reverting gaussian configuration for ITM and ATM options. OTM options prices are sensitive to the choice of copula h-function applied but generally a majority of the specifications still perform better under the mean-reverting square-root configuration than its compatriot. An aggregate of 5.3% and 13% performance improvement is obtained across all moneyness levels by the two dynamic models configurations respectively under empirical copula specifications. The dynamic pricing model with square-root stochastic volatility is the best ranking performer across moneyness, considering the empirical copula dependence structure, at 13% overall performance improvement and 10.6%, 9.8% and 25.4% performance improvement for ITM, ATM and OTM call option pricing respectively. This was a strong endorsement for the dynamic pricing model under square-root stochastic volatility.

As noted that OTM option prices derived from the various copula specifications were sensitive to the choice of the copula fitted, with ITM and ATM option prices being rigid to this choice. However, remarkably, given whichever specification of the type of copula h-function Gaussian, Student t, Clayton, Gumbel and Frank types - both the dynamic pricing model configurations under mean-reverting gaussian and mean-reverting square-root stochastic volatility performed better than the Heston (1993) and the Black-Scholes (1973) models across moneyness levels and terms to expiration. The selection herein of the copula h-function, the stochastic volatility process, corresponding moments and cumulative distribution impact the general performance of the

models. The empirical cumulative distribution of the integrated variance process was found to asymptotically match that of gamma distribution rather than a gaussian distribution.

Therefore, the gamma approximation was more accurate than the normal approximation for the integrated variance cumulative distribution, explaining the triumph of the dynamic model under mean-reverting square root stochastic volatility over the dynamic model under mean-reverting gaussian stochastic volatility. The dynamism of this conditional copula approach for pricing options under stochastic volatility, allows for liberty in the specification of the stochastic volatility model such that only the first two moments of the integrated variance process need to be defined and its associated cumulative distribution. It also allows for liberty in the choice of copula h-function such that no restrictive modelling assumption is postulated about the nature of dependence between the stochastic volatility process and the underlying price process allowing for a myriad of alternative implementations under variant copula and stochastic volatility configurations - an algorithmic panacea of its sort for option pricing under stochastic dynamics. It is shown that this closed-form pricing formula is practically implementable being fully analytical and straightforward in its considerations exhibiting transcendence and superiority over previous pricing models under stochastic volatility - thereby providing a suitable alternative for option pricing under stochastic volatility and substantive framework for application of the dynamic copula approach for other option classes under varied stochastic dynamics.

2.6 Empirical Results: Dynamic Option Pricing under Stochastic Volatility and Stochastic Interest Rates

In this sub-section, analysis of the dynamic European option pricing model under stochastic volatility and stochastic interest rates is presented. A Gaussian pair copula is fitted to the mean-reverting stochastic volatility process and the Vasicek stochastic interest rates, and also fitted between the mean-reverting stochastic volatility and the CIR stochastic interest rate processes. These two configurations are used in the pricing of European options under the closed-form dynamic pricing model. Various copula h-functions specifications are fitted in the model, these are - the Joe Copula, Gaussian, Student t, Clayton, Gumbel and Frank Copula. Figure 2.8 presents the plot of US Treasury Bill rate from 19-August-2015 to 19-August-2020. Table 2.6 presents the parameter estimates of the stochastic volatility and stochastic interest rates processes, the moments

of the integrated stochastic variance and stochastic interest rates processes and the fitted copula functions parameters.

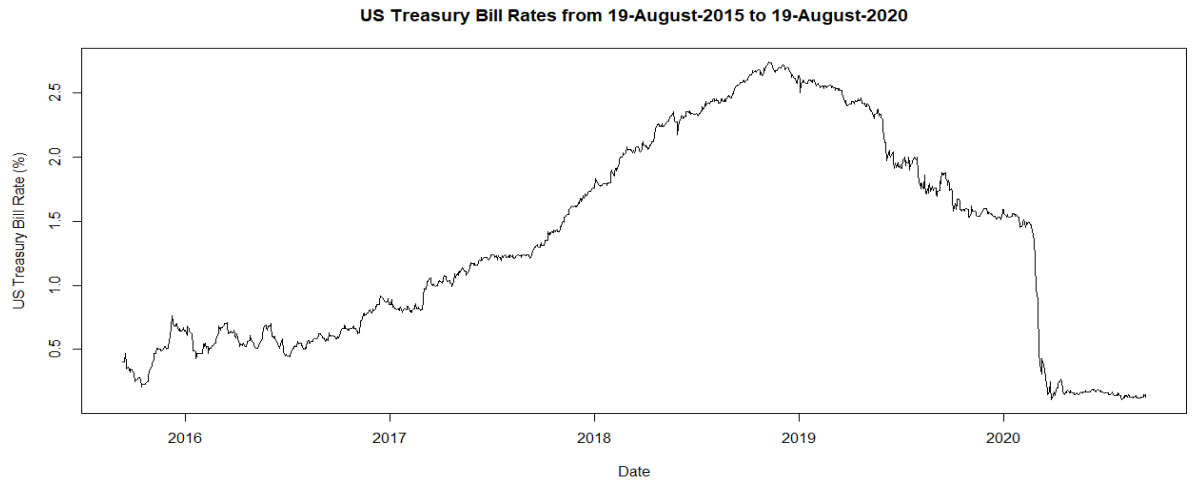

Figure 2.8: Plot of US Treasury Bill rates from 19-August-2015 to 19-August-2020.

Table 2.6: Parameters of Stochastic Volatility and Estimated Empirical Copula Dependence

		OU Stochastic Interest Rates Process	**CIR Stochastic Interest rates Process**	**Mean-reverting Square-root Stochastic Volatility**
Parameters	Speed of Mean Reversion	0.4867521	0.3741669	1.2089
	Long-term Mean Level	0.02198485	0.0241659	0.0347
	Instantaneous Volatility	0.004354768	0.04297848	0.4013
Integral Moments	Mean of Integrated Process	0.01399426	0.01485665	0.0236
	Variance of Integrated Process	0.000005018539	0.0008312896	0.00041
Fitted Copula	SV-SI Gaussian Copula	0.03569974 (0.04)		
	Gaussian	0.464697 (0.02)		
	Student-t	0.4627899 (0.02)		
		30 (0.00)		
	Clayton	0.5268558 (0.04)		
	Gumbel	1.431645 (0.04)		
	Frank	3.395189 (0.22)		
	Joe	1.452394 (0.07)		

Table 2.6: The table presents the estimated parametes k, α and δ of the stochastic volatility process and β, ϑ and Λ of the stochastic interest rates processes representing the speed of mean-reversion, the longterm mean level and intantaneous volatiliy of the stochastic interest rates and stochastic volatility processes. The aggregated first and second moments of the integrated stochastic variance M_V and V_V and integrated stochastic interest rates M_R and $V_R$are also presented. The parameters of the gaussian copula fitted of the stochastci volatility and stochastic interest rates (SV-SI gaussian copula) are shown. Also, the other fitted copula h-functions parameter estimates and the respective parameter standard errors (in brackets) are also presented.

Table 2.7: Correlation Matrix between the Stock Price (SP), Stochastic Volatility (SV) and Stochastic Interest rates (SI) processes

	SP	SV	SI
SP	1.0000	0.6911894	0.596156
SV	0.6911894	1.000	0.02772152
SI	0.596156	0.02772152	1.0000

The Dynamic Stochastic Volatility Model under CIR interest rates, as shown in Table 2.8 and Table 2.9 next, records significant improvement of the Dynamic Square-root Stochastic volatility model of Sub-section 2.2 herein and also a significant improvement over the Heston (1993) model. The dynamic stochastic volatility model with mean-reverting square root stochastic volatility with CIR stochastic interest rates (DSVSI Model 1) improves the pricing performance by on aggregate 31.64% across moneyness levels and copula specifications. Specifically, performance is especially improved over the Heston model by the Clayton copula specification at 41.8% pricing error reduction for OTM options and 31.38% pricing error reduction for ITM options which represents an aggregate of 32.86% performance improvement across moneyness levels over the Heston (1993) Model. The choice of the copula specification is also witnessed to affect the pricing performance of the model albeit all specifications reporting a significant improvement over the Heston Model at an aggregate of 31.64% across specifications. The pricing errors are greatly reduced for OTM options at an aggregate of 36.15% across copula specifications over the Heston Model, and consequently an aggregate of 31.03% for ITM options and by an aggregate of 28.13% for ATM options over the Heston Model.

In Table 2.9, a significant improvement by incorporation of stochastic interests to the dynamic square-root stochastic volatility model under the Joe copula specification is witnessed. The Dynamic Stochastic Volatility Model under CIR interest rates under Clayton copula specification exhibits the highest performance improvement over the DSV2 model under Joe Copula formulation. Under this Clayton copula, the dynamic stochastic volatility model under CIR interest rates reports an aggregate of 22.79% performance improvement across moneyness level, with the highest improvement witnessed for the ITM options at 23.25% improvement and a higher deviation at 21.99% for OTM options in comparison to the aggregate of the other copula specifications of 12.89% over the DSV2 model. Generally, this pricing model under square-root

stochastic volatility and CIR interest rates exhibits an improvement of 22.85% for ITM options, 20.33% for ATM options and 14.41% for OTM options across copula specifications and an aggregate of 21.40% across moneyness levels and copula specifications over the pricing model under square-root stochastic volatility.

Table 2.8: Relative Percentage (%) Performance Improvement of the Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model under Gaussian SV-SI Copula over Heston (1993) Model

	Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model						
	Joe Copula	**Gaussian**	**t-Copula**	**Clayton**	**Gumbel**	**Frank**	**DSV Model 2 under Joe**
ALL	31.89%	31.47%	31.39%	32.86%	31.11%	31.13%	13.04%
ITM	30.94%	30.90%	30.95%	31.38%	30.96%	31.07%	10.60%
ATM	30.17%	29.69%	29.66%	28.12%	29.76%	21.43%	9.80%
OTM	36.95%	34.92%	34.23%	41.80%	32.48%	36.50%	25.40%

Table 2.8: The table reports the percentage performance improvement in RMSE of the Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates under Joe, Gaussian, Student, Clayton, Gumbel and Frank copula specifications across ALL moneyness levels and for ITM, ATM and OTM levels and the Dynamic Mean-reverting Square-root Stochastic Volatility under empirical Joe Copula over the Heston (1993) Model. The relative performance improvement values are computed from the RMSE values obtained and shown in Table 5 below.

Table 2.9: Relative Percentage (%) Performance Improvement of the Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model under Gaussian SV-SI Copula Performance over the DSV Model 2 under empirical Joe Copula

	Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model							
	Joe Copula	**Gaussian**	**t-Copula**	**Clayton**	**Gumbel**	**Frank**	**Aggregate across Copula**	**Deviation across Copula**
ALL	21.68%	21.20%	21.11%	22.79%	20.78%	20.81%	21.40%	0.006912489
ITM	22.76%	22.71%	22.76%	23.25%	22.78%	22.89%	22.85%	0.001834318
ATM	22.59%	22.05%	22.01%	20.31%	22.13%	12.90%	20.33%	0.033993549
OTM	15.49%	12.77%	11.84%	21.99%	9.49%	14.88%	14.41%	0.039231068

Table 2.9: The table reports the percentage performance improvement in RMSE of the Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates under Joe, Gaussian, Student, Clayton, Gumbel and Frank copula specifications across ALL moneyness levels and for ITM, ATM and OTM levels over the Dynamic Mean-reverting Square-root Stochastic Volatility under empirical Joe Copula. The relative performance improvement values are computed from the RMSE values obtained and shown in Table 5 below.

Table 2.10: Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model under Gaussian SV-SI Copula Performance Comparison (RMSE) across Moneyness for Call Options (Contracts =70, Total Observations =35000)

			Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model						**DSV Model 2 (Joe)**	**Heston (1993) Model**	**Black-Scholes Model**
	Moneyness	Obs	**Copula**	**Norm**	**t**	**Clayton**	**Gumbel**	**Frank**			
Mean	ALL	35000	43.78	44.05	44.10	43.16	44.28	44.27	55.90	64.28	66.27
	ITM	21000	54.86	54.89	54.85	54.51	54.84	54.76	71.02	79.44	79.34
	ATM	6500	21.52	21.67	21.68	22.15	21.65	24.21	27.80	30.82	31.34
	OTM	7500	32.05	33.08	33.43	29.58	34.32	32.28	37.92	50.83	59.94
Std. Dev	ALL	35000	20.71	20.71	20.73	20.88	20.69	20.52	23.08	24.46	24.01
	ITM	21000	19.26	19.20	19.24	19.54	19.26	19.29	16.21	16.5	16.55
	ATM	6500	1.24	1.41	1.40	0.83	1.37	0.82	3.60	6.11	5.59
	OTM	7500	7.15	9.25	9.80	6.03	10.24	11.27	8.92	14	18.04
Min.	ALL	35000	20.36	20.40	20.41	21.42	20.40	22.28	24.29	24.82	25.71
	ITM	21000	25.23	25.57	25.52	24.87	25.44	26.24	37.79	45.61	44.91
	ATM	6500	20.36	20.40	20.41	21.42	20.40	22.85	24.29	24.82	25.71
	OTM	7500	22.44	21.66	21.53	22.56	21.72	22.28	29.54	29.16	32.38
Max.	ALL	35000	89.89	89.89	89.89	89.89	89.89	89.89	91.59	100.15	100.15
	ITM	21000	89.89	89.89	89.89	89.89	89.89	89.89	91.59	100.15	100.15
	ATM	6500	24.27	24.65	24.62	24.08	24.55	25.69	35.53	42.49	42.38
	OTM	7500	44.46	49.92	51.39	41.26	52.91	55.56	63.68	71.2	86.9

Table 2.10: The table reports the pricing errors (RMSE) of the Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates Model under Joe, Gaussian, Student, Clayton, Gumbel and Frank copula specifications. The pricing errors from the DSV 2 model under joe copula, and those obtained by the Heston and Black-Scholes model are also presented. The pricing errors are compared across ITM, ATM, OTM and ALL moneyness levels. The mean, min, max and std. dev of the pricing errors within the respective moneyness levels is also summarized. A total of 70 S&P 500 Index European call options contracts were evaluated, with a time series of 500 price observations for each contracts, making the total number observations in the RMSE evaluated to be 35000.

In Figure 2.9 next as shown, the pricing errors (RMSE) are plotted against the strike price and the moneyness levels for the Dynamic Mean-reverting Square-root Stochastic Volatility with CIR Stochastic Interest rates (DSVSI Model 1) with different copula specifications, which are then compare with the Dynamic Mean-reverting Square-root Stochastic Volatility Model (DSV Model 2) under Joe copula. Given that the DSV Model 2 performed best in comparison to other model specifications under stochastic volatility it was taken as a benchmark for the performance of the DSVSI Model 1. Generally, the DSVSI Model 1 performs consistently better than the benchmark DSV Model 2 across strike levels. Under different copula function

specifications, the performance is consistent for ITM options. There is variation in performance trend for ATM and OTM options although still better performing than the DSV Model 2. For ITM options, the model performance is invariant to the copula function specifications with a reported cross-copula deviation in performance of only 0.001834318. However, the performance variation due to the copula function choice is low at 0.033993549 and 0.039231068 for the ATM and OTM options as shown in Table 2.9 above.

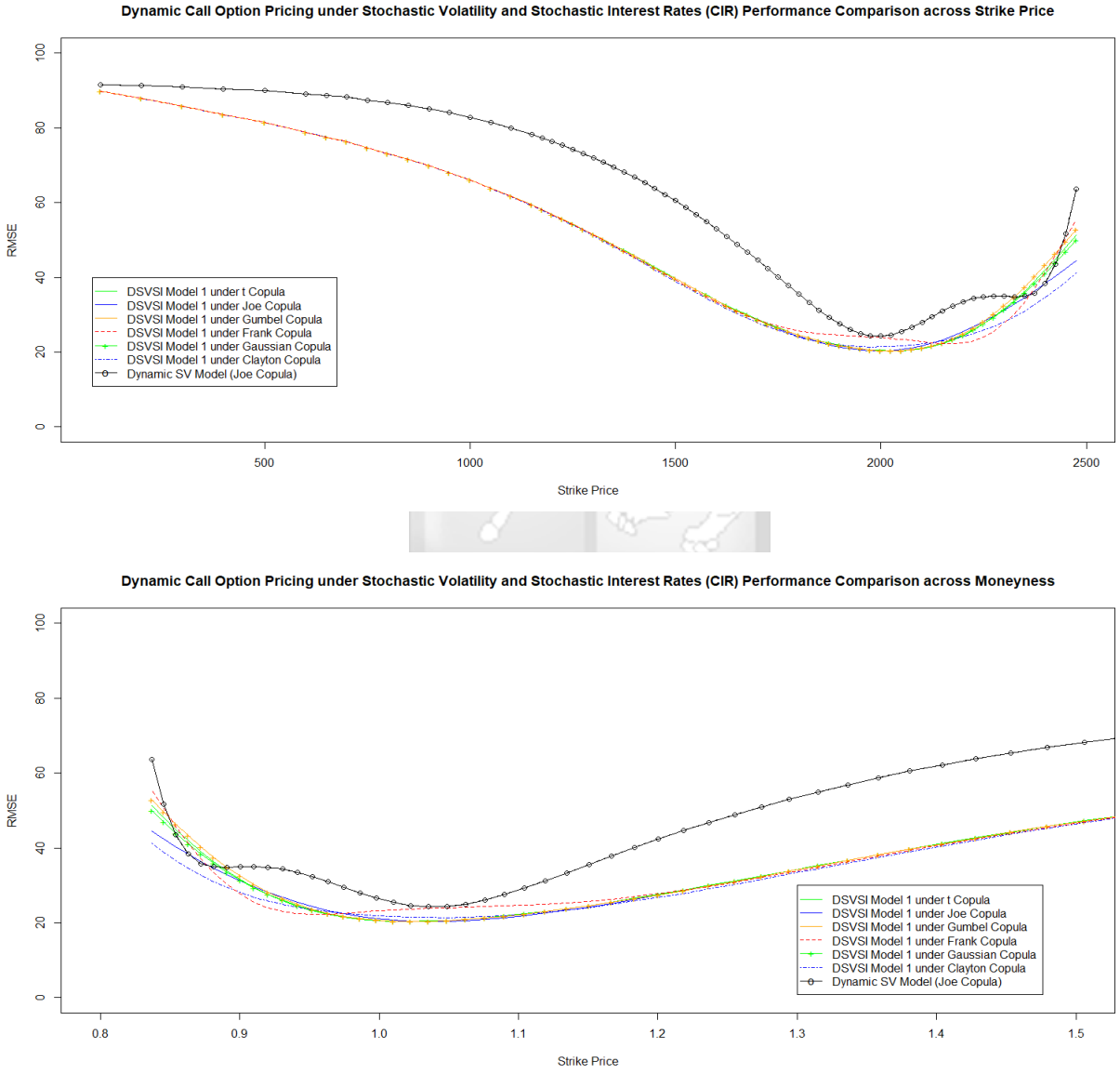

Figure 2.9 (a), (b): Dynamic option pricing model under mean-reverting square-root stochastic volatility and CIR stochastic interest rates (DSVSI Model 1) pricing performance comparison (by RMSE) under various copula h-function configurations across (a) Strike price and (b) Moneyness in comparison with the DSV Model 2 under Joe Copula

The empirical tests findings for the dynamic mean-reverting square-root stochastic volatility model with vasicek stochastic interest rates (DSVSI Model 2) in Table 2.11 and Table 2.12 next, reveal that in comparison to the Heston (1993) Model, the model has a performance improvement over the Heston of an aggregate of 33.44% across copula function specifications. The clayton specification performs best with an aggregate of 34.14% performance improvement across moneyness levels. OTM options under the clayton specification display a performance improvement of 40.43% followed by the ITM options at 33.61% for this clayton specification. The choice of copula specification affects the performance of the model only slightly with deviation in performance across copula specification being minimal. The performance improvement recorded for OTM options over the Heston model is an aggregate of 36.91% across copula specifications, and an aggregate performance improvement of 33.29% and 28.06% for ITM and ATM options respectively over the Heston (1993) Model.

In Table 2.12, the dynamic mean-reverting square-root stochastic volatility model with vasicek stochastic interest rates (DSVSI Model 2) performance is compared to that of the dynamic mean-reverting square-root stochastic volatility model under joe copula (DSV Model 2). The clayton copula specification for the DSVSI 2 Model exhibits the highest performance improvement over the DSV2 Model under joe copula at 24.27% across moneyness levels, with the highest witnessed for ITM options at 25.74% and 20.15% for OTM options. Generally, this pricing model under the square-root stochastic volatility and vasicek interest rates exhibits a performance improvement of 25.39% for ITM options, 20.24% for ATM options and 15.43% for OTM options across copula specifications, an aggregate of 23.46% performance improvement across moneyness and copula specification was achieved by the DSVSI 2 Model over the DSV 2 Model under Joe copula.

Table 2.11: Relative Percentage (%) Performance Improvement of the Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model under Gaussian SV-SI Copula over Heston (1993) Model

	Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model						**DSV Model 2 under Joe**
	Joe	**Gaussian**	**t-Copula**	**Clayton**	**Gumbel**	**Frank**	
ALL	33.85%	33.50%	33.37%	34.14%	33.09%	32.68%	13.04 %
ITM	33.24%	33.18%	33.22%	33.61%	33.23%	33.27%	10.60 %
ATM	31.18%	30.70%	30.31%	26.61%	30.09%	19.45%	9.80 %
OTM	37.94%	36.35%	35.63%	40.43%	34.07%	37.04%	25.40 %

Table 2.11: The table reports the percentage performance improvement in RMSE of the Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates under Joe, Gaussian, Student, Clayton, Gumbel and Frank copula specifications across ALL moneyness levels and for ITM, ATM and OTM levels and the Dynamic Mean-reverting Square-root Stochastic Volatility under empirical Joe Copula over the Heston (1993) Model. The relative performance improvement values are computed from the RMSE values obtained and shown in Table 2.6.8 below.

Table 2.12: Relative Percentage (%) Performance Improvement of the Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model under Gaussian SV-SI Copula Performance over the DSV Model 2 under empirical Joe Copula

	Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model							
	Joe	**Gaussian**	**t-Copula**	**Clayton**	**Gumbel**	**Frank**	**Aggregate across Copula**	**Deviation across Copula**
ALL	23.94%	23.53%	23.39%	24.27%	23.06%	22.59%	23.46%	0.005497171
ITM	25.32%	25.26%	25.31%	25.74%	25.32%	25.36%	25.39%	0.001614
ATM	23.70%	23.18%	22.74%	18.63%	22.49%	10.70%	20.24%	0.04574
OTM	16.81%	14.68%	13.72%	20.15%	11.63%	15.61%	15.43%	0.026502

Table 2.12: The table reports the percentage performance improvement in RMSE of the Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates under Joe, Gaussian, Student, Clayton, Gumbel and Frank copula specifications across ALL moneyness levels and for ITM, ATM and OTM levels over the Dynamic Mean-reverting Square-root Stochastic Volatility under empirical Joe Copula. The relative performance improvement values are computed from the RMSE values obtained and shown in Table 2.13.

Table 2.13: Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model under Gaussian SV-SI Copula Performance Comparison (RMSE) across Moneyness for Call Options (Contracts =70, Total Observations =35000)

			Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model						**DSV Model 2 (Joe)**	**Heston (1993) Model**	**Black-Scholes Model**
	Moneyness	Obs	**Joe**	**Norm**	**t**	**Clayton**	**Gumbel**	**Frank**			
Mean	ALL	35000	42.52	42.75	42.83	42.33	43.01	43.27	55.90	64.28	66.27
	ITM	21000	53.03	53.08	53.05	52.74	53.04	53.01	71.02	79.44	79.34
	ATM	6500	21.21	21.36	21.48	22.62	21.55	24.83	27.80	30.82	31.34
	OTM	7500	31.54	32.35	32.72	30.28	33.51	32.00	37.92	50.83	59.94
Std. Dev	ALL	35000	20.46	20.47	20.45	20.31	20.39	20.00	23.08	24.46	24.01
	ITM	21000	19.71	19.65	19.68	19.93	19.69	19.66	16.21	16.50	16.55
	ATM	6500	0.83	0.89	0.86	0.54	0.82	0.20	3.60	6.11	5.59
	OTM	7500	6.45	8.49	9.00	5.18	9.35	10.15	8.92	14.00	18.04
Min.	ALL	35000	20.33	20.54	20.71	22.04	20.80	23.46	24.29	24.82	25.71
	ITM	21000	23.70	24.10	24.11	23.75	24.08	25.31	37.79	45.61	44.91
	ATM	6500	20.33	20.54	20.71	22.04	20.80	24.28	24.29	24.82	25.71
	OTM	7500	22.87	21.98	21.95	24.12	22.23	23.46	29.54	29.16	32.38
Max.	ALL	35000	89.71	89.71	89.71	89.71	89.71	89.71	91.59	100.15	100.15
	ITM	21000	89.71	89.71	89.71	89.71	89.71	89.71	91.59	100.15	100.15
	ATM	6500	22.91	23.35	23.40	23.70	23.37	25.01	35.53	42.49	42.38
	OTM	7500	42.93	48.00	49.48	40.37	50.86	53.65	63.68	71.20	86.90

Table 2.13: The table reports the pricing errors (RMSE) of the Dynamic Mean-reverting Square-root Stochastic Volatility with Vasicek Stochastic Interest rates Model under Joe, Gaussian, Student, Clayton, Gumbel and Frank copula specifications. The pricing errors from the DSV 2 model under joe copula, and those obtained by the Heston and Black-Scholes model are also presented. The pricing errors are compared across ITM, ATM, OTM and ALL moneyness levels. The mean, min, max and std. dev of the pricing errors within the respective moneyness levels is also summarized. A total of 70 S&P 500 Index European call options contracts were evaluated, with a time series of 500 price observations for each contracts, making the total number observations in the RMSE evaluated to be 35000.

In Figure 2.10 as shown, the pricing errors (RMSE) are plotted against the strike price and the moneyness levels for the dynamic mean-reverting square-root stochastic volatility model with vasicek stochastic interest rate (DSVSI Model 2) with different copula specifications and compared against the DSV Model 2 under Joe copula formulation. From the plots, it can be noted that generally the DSVSI Model 2 has lower pricing errors by RMSE than that of the DSV Model 2 under Joe copula. This performance is consistently good for ITM options and is invariant (at a cross-copula deviation of only 0.001614) to the choice of copula specification. The pricing errors were seen to be still lower for ATM and OTM option than those achieved

by the DSV Model 2 under joe copula but there was variability of the performance based on the choice of copula for these ATM and OTM options, however this variation due the choice of copula is low at only 0.04574 and 0.026502 respectively for the ATM and OTM options as shown in Table 2.6.7. The highest variability of performance due to copula specification is witnesses for the ATM options at 0.04574.

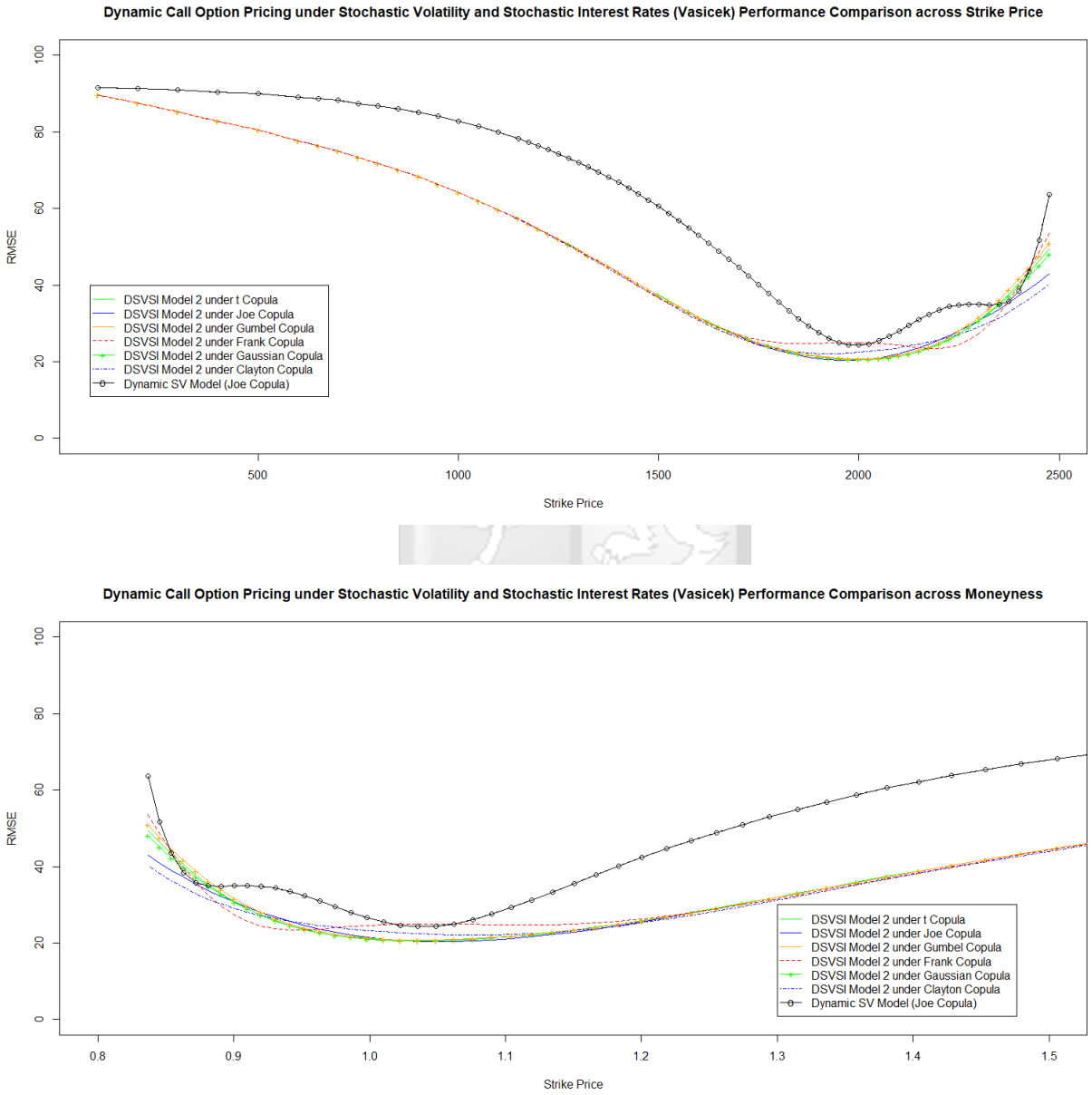

Figure 2.10 (a), (b): Dynamic option pricing model under mean-reverting square-root stochastic volatility and vasicek stochastic interest rates (DSVSI Model 2) pricing performance comparison (by RMSE) under various copula h-function configurations across (a) Strike price and (b) Moneyness in comparison with the DSV Model 2 under Joe Copula

2.6.1 Discussions: Dynamic Option Pricing under Stochastic Volatility and Stochastic Interest Rates

The objective of this sub-section was to develop a closed-form dynamic option pricing model under stochastic volatility and stochastic interest rates, with the incorporation of CIR and vasicek stochastic interest rates to the dynamic mean-reverting square-root stochastic volatility model (DSV Model 2) - the two dynamic model specifications under CIR (DSVSI Model 1) and under vasicek stochastic interest rates (DSVSI Model 2) were constructed in a bid to examine the extent of performance improvement yielded in pricing of 3-Year European S&P 500 Index Options by further allowing that interest rates be stochastic. The two models – the dynamic mean-reverting square-root stochastic volatility model with CIR interest rates (DSVSI Model 1) and the dynamic mean-reverting square-root stochastic volatility model with vasicek interest rates (DSVSI Model 2) were evaluated empirically and the pricing errors (RMSE) for the 3-year European Call options were obtained and compared to those achieved by the DSV Model 2 and the Heston model. The results indicated that generally by the inclusion on stochastic interest rates, the two models performed significantly better than the Heston (1993) and the DSV Model 2 across moneyness levels and across the various copula (Joe, Gaussian, Student t, Clayton, Gumbel and Frank) specifications used.

The dynamic stochastic volatility model with mean-reverting square root stochastic volatility with CIR stochastic interest rates (DSVSI Model 1) improves the pricing performance by on aggregate 31.64% across moneyness levels and copula specifications over the Heston Model. In comparison to the DSV Model 1 it exhibits an improvement of 22.85% for ITM options, 20.33% for ATM options and 14.41% for OTM options across copula specifications and an aggregate of 21.40% across moneyness levels and copula specifications. For ITM options, this model performance is invariant to the copula function specifications with a reported cross-copula deviation in performance of only 0.001834318. However, the performance variation due to the copula function choice is low at 0.033993549 and 0.039231068 for the ATM and OTM options.

The DSVSI Model 2 with vasicek stochastic interest rates performed on aggregate better than the DSVSI Model 1 with CIR stochastic interest rates, in comparison to the Heston Model the DSVSI Model 1 achieved an aggregate performance improvement of 33.44% across moneyness and

copula specification whereas the DSVSI Model 1 achieved 31.64% performance improvement across copula specifications and moneyness levels over the Heston Model.

This pricing model under the square-root stochastic volatility and vasicek interest rates exhibits a performance improvement of 25.39% for ITM options, 20.24% for ATM options and 15.43% for OTM options across copula specifications, an aggregate of 23.46% performance improvement across moneyness and copula specification was achieved by the DSVSI Model 2 over the DSV Model 2 under Joe copula. A cross-copula deviation of only 0.001614 was achieved for ITM options, and 0.04574 for ATM options and 0.026502 for OTM options was computed. The highest variability of performance due to copula specification in the two models was witnessed for the DSVSI Model 2 at 0.04574 and the lowest was 0.001614 for ITM options. This indicated the sensitivity of the dynamic approach to the specification of the stochastic interest rates process and copula function. The combination of modelling the stochastic interest rates as following a gaussian mean-reverting process and the stochastic volatility as following a mean-reverting square-root process yielded 1.8% performance gain for the DSVSI Model 2.

Albeit the gain is not much large but empirical results indicate that the specification of the stochastic interest rate process could potentially lead to further improvement of performance. For the two models, however the difference is small enough (1.8% aggregate difference) for us to consider the models as alternatives and not competitors since at some copula specifications and moneyness levels the DSVSI Model 1 with CIR stochastic interest outperformed the DSVSI Model 2 with vasicek interest rates, for example under Clayton specification for OTM options the former model achieved 41.80% while the latter achieved 40.43%. In both the models however, the Clayton copula specification brought the highest performance improvement at 22.79% and 24.27% for the DSVSI Model 1 and DSVSI Model 2 respectively over the dynamic mean-reverting square-root stochastic volatility (DSV Model 1 under joe copula).

The dynamism of the copula approach allows for pricing under stochasticity of interest rates and volatility in a tractable and easily implementable manner. The underlying processes of the stochastic interest rates and stochastic volatility are easily malleable in this dynamic pricing approach where a myriad of alternative combinations can be made for these underlying processes under various copula configurations. This approach generates a closed-form formula that is easily and practically implementable, being fully analytical for pricing options under stochastic interest

rates and stochastic volatility providing liberty for the specification of the underlying processes and the copula function definition. The consideration of stochastic interest rates in the pricing of option reduces the model pricing errors significantly in this case by 33.44% across moneyness levels and across copula function configurations in the best performing version of the model.

Chapter 3: Acceleration by Parallelization of Dynamic Derivatives Pricing Algorithms on GPU-CPU Heterogeneous Systems

3.1 Chapter Overview

This chapter utilizes stochastic global optimization specifically genetic algorithm and simulated annealing for the inverse problem of calibrating the dynamic stochastic volatility model under square-root stochastic volatility and the dynamic stochastic volatility and stochastic interest rates models to the market prices. The performance of the models in pricing of European options under these obtained optimal parameters is also discussed. To enhance model throughput and reduce latency, a heterogeneous hybrid programming model on GPU which was explored emphasized a data parallel implementation of the developed option pricing algorithm under stochastic volatility on a GPU-based prototype system. Kernel offloading to the GPU of the compute intensive segments of the pricing algorithms was done in OpenCL. An investigation of this GPU accelerated approach to option pricing algorithms deployment over a parallel approach was provided. The GPU approach was found to significantly reduce latency by an optimum of 541 times faster than a parallel implementation approach on the CPU.

3.2 Introduction

Over the past decade, the computing needs within a typical financial institution have increased significantly. The pricing, calibration and risk management tasks must now be delegated to dedicated computational servers to be completed in a sufficiently short time. This increase in computing demands has been brought about by the tremendous growth in both the scope and size of problems being addressed and the complexity of models being used. To accommodate the observed implied volatility skew, models with stochastic volatility, stochastic interest rates and jump diffusions have been deployed. This has then brought about even further complexities. This ever-increasing complexity and scope of problems being tackled in the area of computational finance brings about a need for efficient pricing architectures that are both powerful and flexible (Basermann et al., 2008; Surkov, 2010; Zeng, Lin & Pan, 2022).

The advent into the use of powerful Graphics Processing Units (GPUs) has suggested the idea of developing algorithms to price financial options that are highly performing when executed on these

processors. Since GPUs are ubiquitous, they can be cheaper platforms to do parallel computing. GPUs are present in personal computers and also in smaller devices such as tablets and smartphones. The use of GPU in finance can result in substantial savings in the evaluation times for financial products in comparison to those obtain when using the CPU. These savings could be used to speed up the decision process in financial institutions such as banks, insurance companies and hedge funds since these institutions usually evaluate a large number of contracts of the same type with varying parameter values and varying strike, maturity and volatility levels. This feature of this decision process in the financial institutions is thus very well suited for parallel and/or distributed computing. The inherently parallel, multi-processor structure of the GPUs makes them especially attractive for the parallel solution of multiple problems. Parallelization of multiple computations allows for the streamlining of the memory transfer and saturate the processor cores with work, thus increasing the overall throughput of the algorithm (Fatone, Giacinti & Mariani, 2012).

The access to computing power that is scalable, accurate and reliable has therefore become a major requirement due to increased competition, increased products and complexity in models and increased volume of data. Requirements by regulation authorities have also amplified the need for fast and reliable computing capacities by imposing higher standards for availability, system integrity and security on financial institutions. At the same time also, financial institutions have felt the pressure to cut development times and costs for the introduction of new products and models, and improve business efficiency (Basermann et al., 2008; Harris, 2008). Some of the more complex analytical models would exhaust the computer capabilities of the workstations, and analysts would then ideally resort to the use of clusters of workstations. By implementing parallel and distributed architecture for their front or back offices, they are able to achieve a great deal of flexibility and reliability and increase the performance-to-cost ratios for the institution. Since they will have a system of shared workstations that can solve a large complex task in a fraction of time that a single computer would, and thus the need to have constant upgrading to large computing hardware eliminated. Previously large tasks that require constant re-computation can be accelerated by harnessing parallel architectures which will enable achieving the low latency requirements of the institutions' computationally extensive and expensive problems, solving them within microseconds. In order to have this efficient system - it is therefore desirable to have complex optimization and forecasting methodologies in addition to ability to perform fast

calibrations to market valuation, risk sensitivities and implement stress tests, back tests and performance tests for all strikes and all maturities on the entire trading book in real-time - enabling financial markets participants to make timely proper investment and hedging decisions (Basermann et al., 2008).

In computational finance area, the compute time is always an important factor as most financial models are supposed to be used in real financial markets. GPUs have therefore been exposed to offer a superior level of performance over CPUs, both in achievable computational power and in memory bandwidth (Peng et al., 2010). Many financial measures are stochastic and require a high number of simulations. Option pricing, Value-at Risk calculations and Hedging strategies require rapid adaptation with suitable techniques and automated trading algorithms need fast execution to make gain opportunities. Financial establishments using grid computing with CPU have pointed out excessive cost in hardware and electricity consumption. As it was in the case of Aon Benfield, a world's leading insurance company which for a bond pricing service spent $4 million in a grid architecture using CPUs and $1.2 million in electricity a year. On the contrary, a GPU based pricing engine would only have cost $144,000 and $31,000 in electricity a year. For instance, for J.P. Morgan's Equity Derivatives Group, the equity derivative focused risk computation is performed on hybrid GPU-based systems, increasing performance by 40x compared to only CPU-based systems for the same electric power (Benguigui & Baude, 2012)

The complexities of the developed dynamic models under stochastic volatility and stochastic interests, and the large size of the computing problem would require the designing of a much efficient pricing system using alternative approaches and adoption of better suited model calibration algorithms. By implementing a parallel GPU accelerated architecture for such a pricing system that makes use of the dynamics models and uses stochastic optimization for the model calibration process, then a great deal of efficiency, flexibility, reliability and increased performance-to-cost ratios in the implementation of these models can be achieved. This section thereby makes use of stochastic optimization algorithms – Genetic Algorithm and Simulated Annealing – to calibrate the developed dynamic models to market and consequently implements the dynamic models on a hybrid parallel GPU architecture. The performance of these models under the optimal parameters is also evaluated in addition to the compute performance of the parallel GPU architectural deployment. The objective of this section is therefore: (i) to examine the use of

stochastic optimization algorithms in the calibration of the dynamic models and (ii) to evaluate the performance of the dynamic pricing models on the parallel GPU architecture.

This section is therefore arranged as follows: Subsection 3.3 presents literature on GPU architectures for financial derivatives pricing, Subsection 3.4 presents the model calibration approach, Subsection 3.5 presents the stochastic optimization methods, and Subsection 3.6 and 3.7 presents the system development, appraisal and implementation of the GPU based approach.

3.3 Related Literature

3.3.1 Derivatives Pricing on Distributed Computing Architecture

Kanniainen, Piché and Mikkonen (2009) studied two different ways to run Asian option algorithm coded in Matlab in a distributed environment, on PC grid and cluster. The PC grid was using the surplus processor time of computers allocated to other tasks, whereas the cluster had computers that were dedicated to scientific computation. The computations were completed on the cluster at best only 25% faster than on the PC grid with the same number of cores; at around 80 cores, however, the computing times became nearly equal. They note that the PC grid can be considered competitive when the number of nodes increases. The PC grid, they state, is an attractive alternative for computation-intensive tasks that require little or no communication between parallel tasks since clusters are specialized for large computational problems and faster data transfer rates. They opine that the PC grid albeit slightly slower than the cluster, is more attractive in an organization setting given the low cost of set up and availability of extensive computational resources. They concluded that the dedicated cluster and the PC grid both offer comparable performance, scalability, and ability to operate heterogeneous environments. The choice of system, they state, however will in the end depend on management issues such as overall cost of ownership, open-source vs. commercial software, data security, and compatibility with the organization's existing computing resources

They present the following challenges and opportunities in the design of distributed systems for pricing derivatives: (i) usability – an easy to use integrated system with other mathematical tools like MATLAB with accurate and fast results (ii) Gridification of the mathematical problem (i.e. how the computation is modified to enable distributed computing). It should be possible to 'gridify' the code without great efforts and deep software engineering skills (iii) Availability and

reliability - The distributed computing solution must be robust, i.e. it must be always available 24/7/365, the system must operate trustworthily, and the system has to be able to recover from abnormal situations. (iv) Data and infrastructure security - Fine grained security must be involved in the distributed computing solution. (v) Administration and management ability - administration and management of the distributed computing system must be simple and easy. They note also that although distributed computing can, for some problems, produce impressive speedups; so can intelligent modelling, algorithm development, and coding. Therefore, intelligent algorithm design for a distributed architecture can yield the desired low-latency or near real-time solutions.

Lee and Kim (2012) value 108 options simultaneously on networked cluster computer system which used the Black-Scholes partial differential equation by the finite element method to solve the option prices and the greeks. They found that adoption of the distributed approach yielded accurate values of options and the Greeks with reasonable computational times when was executed on the cluster computer system rather than on the single node. They state that for large amounts options, the distributed computing approach will be a highly attractive alternative to devising hedging strategies or developing new pricing models. Their cluster computer system (*Pegasus)* consists of 260 nodes, 520 Intel Xeon 2.2/2.4/2.8/3.06 GHz processors. Each node in Pegasus runs dual Intel Xeon processors on the E7500 chipset motherboard with 2 (or 3, 4, 6) GB DDR RAM and 80/160 GB HDD. Nortel Baystack 380-24T L2 switches were connected to 260 nodes and to Passport 8600 Routing switch at the core of Gigabit Ethernet network system. The tuned-up network bandwidth of two nodes is about 920 Mbps and the latency time is about 21 μsec. They perform one hundred options' valuations were performed simultaneously on Pegasus. The total elapsed time of the computation for 108 option prices was 15.024 seconds. They state option pricing on the distributed computing is a very economical way, compared to other high-performance systems such as the supercomputer or the GPU adopting system.

Kanniainen and Koskinen (2017) in their study of distributed calibration of option pricing models with multiple contracts written on different underlying assets, demonstrate by the use of Techila how multiple independent calibration tasks (on different underlyings) can be efficiently accelerated, reducing the computational time from 7 hours to a few minutes. They consider instances where a financial institution has multiple options positions on different underlying securities. The calibration and recalibration of models using data on these several underlying

securities, they opined to take too long, especially, if advanced non-affine models without analytical solutions are used for pricing the options and when the price processes are affected within the trading day by news arrival.

They calibrated the volatility surfaces for 100 assets. In their computing infrastructure computations were performed in Microsoft Azure cloud using Techila Distributed Computing Engine. The computational capacity consisted of 25 D3v2 virtual machines, which each have 4 CPU Cores, a total 0f 100 CPU cores. The CPU time used to calibrate the model for 100 volatility surfaces was CPU time of 7 h 50 m 10 s, on a single CPU core single threaded machine. By using 100 CPU cores in Techila Distributed Computing Engine, the computations were completed in wall clock time of 5 minutes 19 seconds. They found that by increasing the number of CPUs per job and using multithreaded functionality in Techila Distributed Computing Engine, the CPU time increased to 12 h 33m whereas the wall clock time reduced significantly to 1 m 55 s. with 800 CPU cores.

Benguigui and Baude (2012) present a GPU adaptation of a specific Monte Carlo and classification based method for pricing American basket options. They propose a dynamic strategy of kernel calibration. In their parallel algorithm for GPU each thread performs the following operations at every discrete time: generates random uniform variables, applies Gaussian transformation, correlates them if specified, simulates asset prices, predicts exercise situation through a classify function and stores actualized payoff. They note however that contrary to a distributed architecture, all cores on the GPU are identical and no load balancing is needed. They had challenges with implicit synchronization barrier since threads of the same warp (smallest quantity of threads that are issued with a SIMT instruction; for NVIDIA architecture or a wavefront for AMD). They overcome this by distributing the same number of simulations per thread, those performing short-length simulations would wait for the others of the same warp. They advocate for use of distributed computing infrastructures such as Grids and Clouds, equipped with GPUs, in order to benefit for even more parallelism in solving computationally intensive problems.

3.3.2 Derivatives Pricing on GPU architecture

Among the works done on GPU acceleration in option pricing, most have focused on numerical implementation on GPU based on lattice methods such as in Peng et al. (2010), Ganesan et al.(2009), Solomon et al. (2009), Zhang et al. (2012); Suo et al. (2015), or based on monte carlo

simulation such as in Podlozhnyuk and Harris (2007) , Abbas-Turki and Lapeyre (2009) , Suo et al. (2015), Trainor and Crookes (2013) and Yu et al. (2010) or fourier methods / PDE approximation such as in Dang (2011), Dang, Christara and Jackson (2010), Surkov (2010). These implementations were done for a particular option pricing model and option type such as American, Asian, European or multi-asset options. Among those that apply lattice methods for option pricing as such as Peng et al. (2010) where they used a non-linear BSDE model for the option price and solved it numerically using a binomial lattice method on GPU. In their implementation in order to reduce the global memory access frequency and avoid the extra transferring cost, kernel invocation was avoided at each time step. They found that running time experiments on the GPU implementation achieved almost 200x speedup compared with the CPU-only case; with the growing the number of time steps for the problem, the CPU time increased much more sharply than the GPU time. Solomon et al. (2009) implement a trinomial lattice pricer for European options and a binomial lattice pricer for American lookback options. They find that, for large numbers of time-steps (between 1000 and 30,000), the GPU gives speedups which increase, almost linearly, up to 101x (for 30,000 steps). This shows that GPUs can also be useful when using other types of options pricing methods.

Peng et al. (2010) noted that the influence of thread blocks number and load balance on the performance was also examined whereby when increasing the number of blocks, the amount of computing tasks allocated to each thread decreased, therefore the computation time decreases. But when the number of blocks was increased to a certain degree, the amount of computation tasks on each thread became too small, which made the extra scheduling costs significant, thus the computation time begin to increase. Their study noted that naive implementation is adapted better for problems with small number of time steps and load-balanced version suits better problems with large number of time steps.

Zhang and Oosterlee (2009) implemented the COS algorithm on the GPU and compared the GPU based prices with those obtained by CPU implementations. They used the COS method for pricing European options where they achieved 5 times acceleration. For both European and Bermudan options, as the number of terms in the Fourier cosine expansion increases, the acceleration factor of the GPU improves. For Bermudan options, they found that the number of exercise dates does not influence the acceleration factor of GPU very much. They noted GPU implementation

challenges such as a time-consuming memory transfer, and requiring additional computation resulting from unscaled inverse Fourier transformation; it however still outperformed the CPU due to its parallelization

Suo et al. (2015) implemented lattice and Monte-Carlo methods for option pricing using GPU in CUDA and OpenCL for individual options. They suggest parallelization of the problem for large option portfolio. They opine that the parallel implementations achieve significant performance improvement over serial implementations. Yu et al. (2010) present monte carlo pricing acceleration on GPU based on the SABR stochastic volatility model. They focused on European option pricing using quasi-Monte Carlo with the Brownian bridge method and American option pricing using the least squares Monte Carlo method. They then implemented a GPU-based program for pricing European options and a hybrid CPU-GPU program for pricing American options. The numerical results indicated around 100× speedup in European option pricing and 10× speedup in American option pricing can be achieved by GPU computing while maintaining satisfactory pricing accuracy.

Trainor and Crookes (2013) in their study of pricing complex derivatives using monte carlo simulation, adopt GPU implementation of a Basket Option pricing engine and an analysis of the timing and memory resources for different algorithm parameters. Their results showed that, on an NVidia GTX670 card using NVidia's proprietary CUDA programming platform, a speedup of over 250 can sometimes be achieved compared with a parallel C implementation. They went ahead and compared CUDA and OpenCL implementations noting OpenCL portability of code to a range of different architectures. They reported that with all things being equal, the performance of OpenCL and CUDA implementations are within approximately 10% of each other, with the OpenCL implementation sometimes being the faster. They opine that OpenCL is a viable programming platform for GPU acceleration of pricing engines across architectures.

Grauer-Gray et al. (2013) accelerate QuantLib financial applications, in their study Black-Scholes, Monte-Carlo, Bonds, and Repo code paths in QuantLib are accelerated using hand-written CUDA and OpenCL codes specifically targeted for the GPU. Their results demonstrated a significant speedup for each code using each parallelization method. The speedup for all GPU implementations is over 100 times over the parallel CPU implementation and over 10 times over the multicore OpenMP implementation when running over 20,000 options. They found further that

the manual OpenCL implementation is over 40% faster than the hand-written CUDA implementation. This difference in speedup could be because of differences in how the code in each environment is compiled. They also examine GPU acceleration for monte carlo financial applications and found that using over 5,000 samples showed a speedup of at least 75 times over the single-core CPU implementation and at least 10 times over the multi-core CPU implementation.

Fatone, Giacinti and Mariani (2012) note that due to the properties of GPUs an effective implementation of a numerical algorithm on these units must satisfy the following requirements: (i) low number of "communications" between the "processors" (i.e., between the cores) contained in the unit, (ii) well balanced distribution of the computational load among the "processors," (iii) no use (when possible) or almost no use of nested functions and/or of nested subroutine calls. This last requirement is due to the fact that in the GPUs the memory addressable by each "processor" is rather small and the communications between the "processors" take place through the global memory of the GPU card that is shared among the "processors" of the GPU. In their study they derive semi-integral formulae to evaluate the derivative prices and use the formulae to develop efficient parallel algorithms for pricing barrier options and realized variance options. They opine that the ubiquitous presence and low cost of GPU deployments is an area of further interest in mathematical finance and could be exploited to further enrich contexts in this area.

Generally, in these studies the acceleration algorithms exhibit tremendous speedup over the parallel CPU implementation and therefore suitable for real-time application. It is noted that the highly parallel structure of GPUs makes the more effective than traditional CPUs in computationally intensive programs.

3.4 Model Calibration

Calibration of the model is a crucial process and a price to pay with more complex model is the increased complexity of the calibration process. The industry standard approach is to minimize the difference between the observed prices and the model prices. A difficulty in solving the calibration problem is that observed information from market data is insufficient to exactly fit the parameters and as such several sets of parameters may perform well and even produce model prices that are consistent with the observed market prices, thereby causing an illposedness of the problem (Mrazek, 2017).

The problem of calibration the dynamic models is formulated as an optimization problem with the aim of minimizing the objective function $G(\Theta)$ between the model prices and the market prices for a set of the traded options.

$$\inf_{\Theta} G(\Theta), \quad G(\Theta) = \sum_{i=1}^{N} w_i |C_i^{\Theta}(t, S_t, T_i, K_i) - C_i^*(T_i, K_i)|^2$$

where N denotes the number of observations of options used for parameter estimation, w_i is an equal weight for ATM options, $C_i^*(T_i, K_i)$ is the market price of the call options observed at time t. $C_i^{\Theta}(t, S_t, T_i, K_i)$ denotes the model prices computed using vector of model parameters from the dynamic option pricing under square-root stochastic volatility model where the set of parameters $\Theta = (v_0, \kappa, \alpha, \theta)$ for the dynamic model under square-root stochastic volatility. This function $G(\Theta)$ is not of any particular structure or form and is not necessarily convex such that it could be having more than one local minimum or both local and global minima, and it is not clear whether a gradient based algorithm would achieve a unique minimum, or whether if a local optimizer such as the nonlinear least square optimization is used, the solution would not be stuck at a local minimum. Therefore, due to this and the possible sensitivity of this function to the initial parameters, the use of both the local and stochastic optimization algorithms is suggested. The local optimization algorithm that was used was the nonlinear least squares algorithm and the stochastic global optimization algorithms that was considered was the genetic algorithm and the simulated annealing algorithm.

3.5 Stochastic Optimization Algorithms

As held to be true in practice that the estimation method of a model is as crucial as the model itself - in this section we investigate the model calibration techniques of the developed dynamic models namely the dynamic square-root stochastic volatility model and the dynamic stochastic interest rate and stochastic volatility model. Determining the set of model parameters that would match the market prices of a set of options is the calibration process, which is ideally an inverse problem to the option pricing problem. When the market prices are used to calibrate the models then more complex exotic options can be priced using the models and risk sensitivities and hedge ratios can be obtained accurately. However, a price to pay for having a more realistic model is the increased

complexity of this model calibration process. The variation between the model prices and the market prices could result as from the choice of the local or global optimizer methods (Mrazek ,2017).

Optimization schemes can be broadly categorized into two groups — local and global schemes. Both have their merits and drawbacks. Local optimization schemes tend to start from their initial parameter estimates and then choose new parameter estimates such that the value of the objective function always moves towards an optimum value. The schemes will continue until a stationary point is found, and as such, tend to locate local optimums rather than global ones. Simple gradient based methods are good examples of local optimizers. The choice of initial parameters in these schemes, especially in non-convex situations, is very important since a poor choice can cause the algorithm to settle in a local optimum instead of a global one. Their simplicity, however, tends to make them faster and easier to implement than global schemes. Global optimization schemes, on the other hand, are much less sensitive to initial parameter estimates. Many of these methods also fall into the category of stochastic optimization schemes since they generate and use random variables to assist in locating an optimum. Stochastic optimization schemes use random variates to generate and accept new parameter values. This helps the algorithms form being stuck in a region of a local optimum rather than being at the global optimum. This increased flexibility however comes at a computational cost (Mikhailov & Nogel, 2003). The stochastic optimization schemes applied to the calibration of the dynamic stochastic volatility models are the Genetic Algorithm and the Simulated Annealing Algorithm.

3.5.1 Genetic Algorithm

The Genetic Algorithm is inspired by evolutionary biology, hence the procedure starts with a population of solutions with the objective function being a fitness function which is to be maximized and the iterations are the generations. The new candidate solutions which are called children or off springs are created by crossover, which refers to a mixing of the existing solutions, and mutation which refers to the randomly changing of component solutions. A selection among parents and children solutions takes place at the end of each generation. The survival of a solution may be stochastic with the probability of survival proportional to a solution's fitness.

Algorithm

Randomly generate initial population P of solutions

While stopping criteria not met **do**

select $P'cP$, $intitialize\ P'' = \emptyset\ (set\ of\ children)$

for *i=1 to n* **do**

randomly select individuals x^a and x^b from P'

apply crossover to x^a and x^b to produce x^{child}

randomly mutate produce child x^{child}

$$P'' = P'' \cup x^{child}$$

end for

$$P = survive\ (P', P'')$$

end while

return best solution

3.5.2 Simulated Annealing Algorithm

The Simulated Annealing Algorithm starts with a random solution x^c and creates a new solution x^n by adding a small perturbation to x^c. If the new solution is better that the current one ($\Delta < 0$), it is accepted and replaces x^c. In the case, x^n is worse, the algorithm does not reject it right away but applies a stochastic acceptance criterion, thus there would still be a chance that the new solution will be accepted, albeit only with a certain probability. This probability is a decreasing function of both the order of magnitude of deterioration and the time the algorithm has already run. This time factor is controlled by the temperature parameter *T* which is reduced over time; hence, impairments in the objective function become less likely to be accepted and, eventually, the algorithms turns into a standard local search. The algorithm stops after a predefined number of iterations R_{max}.

Algorithm

Set R_{max} and T

Randomly generate current solution x^c

For r=1 to R_{max} **do**

While stopping criterion not met d**o**

Generate $x^n \in \aleph(x^c)$ (neighbor to current solution)

Compute \delta- $\Delta = f(x^n) - f(x^c)$ and generate u (uniform random variable)

If $(\Delta < 0)$ or $\left(e^{-\Delta/T} > u\right)$ then $x^c = x^n$

3.6 System Development Methodology

According to Sommerville (2006), with Rapid Prototyping, also known as Rapid E-learning, learners or subject matter experts interact with prototypes and instructional designers in a continuous review and revision process. The rapid prototype creates an early iteration loop that provides valuable feedback on technical issues, creative treatment, and effectiveness of instruction. Figure 3.1 next shows the RAD development cycle.

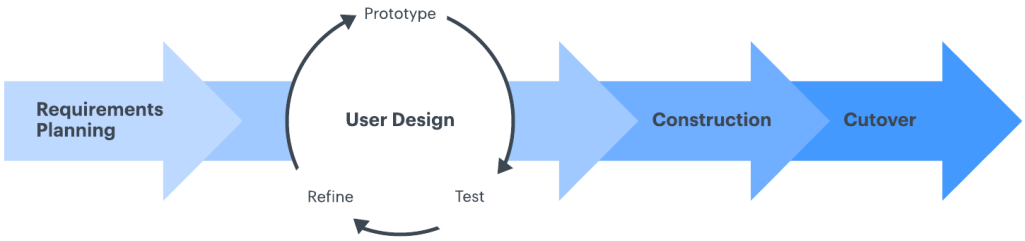

Figure 3.1: RAD Development Cycle (Sommerville, 2006)

How RAD Development Methodology was applied:

The Rapid Application Development (RAD) methodology was applied in this research as follows:

(i) **Requirements Planning:** In this phase, the key needs and specifications for the prototype system were gathered. This included understanding the parameters required in the dynamic option pricing models and computations that needed to be made more efficient through the use of a CPU-GPU parallel architecture. The architecture and algorithms needed for these processes, particularly those derived from the dynamic copula framework for option pricing were also explored. The scalability, accuracy, and reliability of computing power was a major consideration in this stage.

(ii) **Prototyping phase:** In this stage, the prototype of the system was built, modeling how it would use parallel processing to optimize the option pricing computations. The real-time derivative pricing prototype system supported by a parallel and distributed architecture was implemented. This was done using OpenCL C, Python, and R, to implement the hybrid parallel programming methodology on CPU and GPU to provide computational acceleration.

(iii) **Testing:** During this testing phase, the system was extensively evaluated for accuracy and efficiency. The option pricing models under different parameters and stochastic volatility considerations were investigated in performance over the Black-Scholes and Heston models. Further, the system's performance was assessed in acceleration of computation of the option pricing models.

(iv) **Cutover:** In the cutover phase, the prototype system was fully implemented. This involved a transition from the existing deployment to the newly developed system. Further the deployed methods were continuously monitored for any issues and necessary adjustments were made.

3.7 Programming Model

3.7.1 GPU Architecture

A CPU consists of four to eight CPU cores, while the GPU consists of hundreds of smaller cores. Unlike CPU that works at really high frequency to achieve high speed, GPUs have a parallel architecture composed of numerous streaming multiprocessors that work at a lower frequency allowing for lower power consumption and faster speed if the algorithm is parallelizable. Together, they operate to crunch through the data in the application. This massively parallel architecture is what gives the GPU its high compute performance. The architecture of the GPU as compared to the CPU is illustrated in Figure 3.2 next.

In the GPU architecture, which is optimized for parallel tasks and has significantly more compute resources; we can leverage on the high throughput and high performance per watt that the GPU can provide. This can be done in our prototype by GPU acceleration. The GPU accelerates applications running on the CPU by offloading some of the compute-intensive and time-

consuming portions of the code. The rest of the application still runs on the CPU. From a user's perspective, the application runs faster because it is using the massively parallel processing power of the GPU to boost performance (NVidia, 2018).

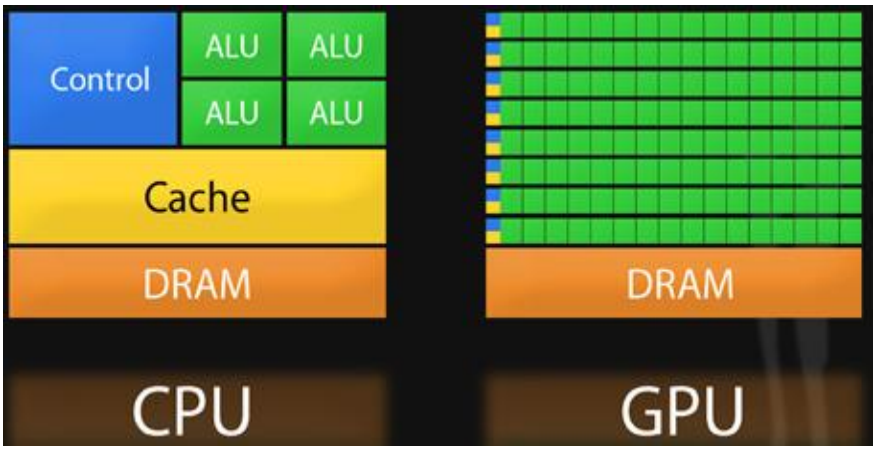

Figure 3.2: Inside a CPU and the GPU (Nvidia, 2018)

A GPU has multiple streaming multiprocessors (SM) that contain memory registers for threads to use, several memory caches (shared memory, constant cache, texture memory, L1 cache), thread schedulers. Each core also consists of an Arithmetic logic unit (ALU) that handles integer and single precision calculations and a Floating-point unit (FPU) that handles double precision calculations. The number of multiprocessor cores depends on microarchitecture generation, they may be different for NVIDIA, Intel and AMD graphic cards (Duke, 2015; NVidia, 2018).

3.7.2 GPU Acceleration

GPU acceleration enables use of highly optimized function to perform 2X-10X faster that the CPU-only parallelization alternatives. For optimal performance, the programmer generally has to juggle finding enough parallelism to use all streaming multiprocessors and optimize the use of registers and shared memory. GPU acceleration within Python can be performed by use of acceleration libraries adopting NVIDIA CUDA or OPENCL implementation. OpenCL is supported by multiple vendors - NVidia, AMD, Intel IBM, ARM, Qualcomm, while CUDA is only supported by NVidia. OpenCL (Open Computing Language) is the open, royalty-free standard for cross-platform, parallel programming of diverse processors found in personal computers, servers, mobile devices and embedded platform. Within Python, GPU acceleration libraries such as PyOpenCL can be used and wrappers that can be used to generate C code within python to be supported by OpenCL such as py2opencl. OpenCL is the only framework for writing portable

applications that work on CPU, GPUs, and mobile devices. Yet programming into native OpenCL code embedded into C/C++ code is very complex and is a major barrier to entry for scientists. pyOpenCL solves this by letting scientists combine the best of both worlds: using Python for the overall workflow, input/output, and plotting; and using OpenCL for the code's computing-intensive parts. pyOpenCL is built on top of the Python library for efficient array manipulations (that is, numerical Python, or NumPy) and OpenCL. It lets programmers embed OpenCL code into a Python program in the form of a string containing the kernel code. It also provides APIs to load a numPy array on a computing device (GPU or CPU), queue and run the kernel, and retrieve the computation results. In pyOpenCL, object cleanup is tied to the lifetime of objects, which makes it easier to write correct leak- and crash-free code (Massimo Di Pierro, 2014). This GPU acceleration illustration is show in the Figure 3.3 next.

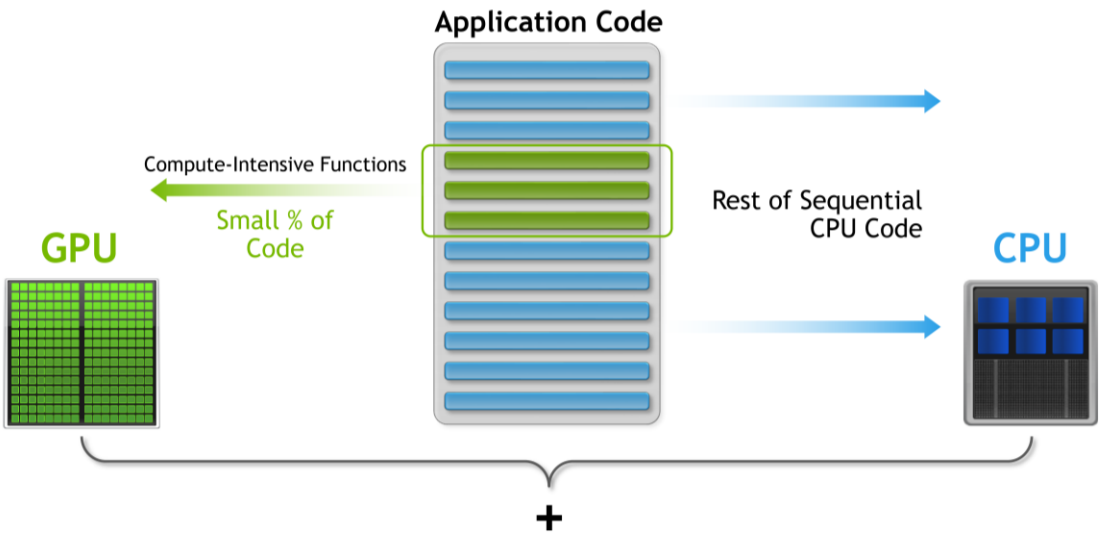

Figure 3.3: How GPU Acceleration Works (Nvidia, 2018)

3.7.3 Parallel Execution Model

Large data sets and algorithms can be processed in parallel using many processors. Modern computers' CPU are multiprocessing, which means that they can run multiple programs/multiple processes concurrently and also at the same time on the different cores. In Figure 3.4, the parallel execution model is illustrated. A primary benefit of OpenCL is substantial acceleration in parallel processing. OpenCL takes all computational resources, such as multi-core CPUs and GPUs, as

peer computational units and correspondingly allocates different levels of memory, taking advantage of the resources available in the system (AMD, 2010).

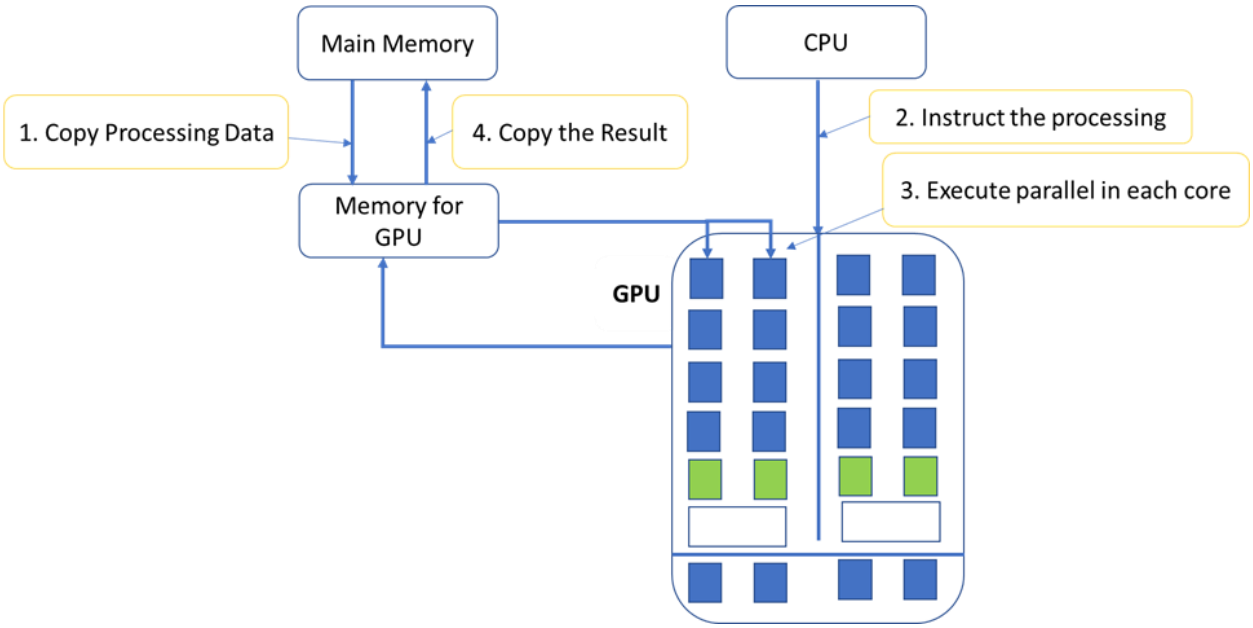

Figure 3.4 : Parallel Execution Model Implemented

The OpenCL execution model is defined in terms of two distinct units of execution: kernels that execute on one or more OpenCL devices and a host program that executes on the host. With regard to OpenCL, the kernels are where the "work" associated with a computation occurs. This work occurs through work-items that execute in groups (work-groups).

A kernel executes within a well-defined context managed by the host. The context defines the environment within which kernels execute. It includes the following resources:

i. Devices: OpenCL GPU Device which was 'Intel(R) HD Graphics 630' on 'Intel(R) OpenCL' on the host 'Intel(R) Core (TM) i7-8705G CPU @ 3.10GHz'. Graphics Base Frequency 350 MHz, Graphics Max Dynamic Frequency 1.10 GHz, Graphics Video Max Memory 64 GB, Max Resolution (HDMI) 4096 x 2160 @30Hz

ii. CPU Host: 'Intel(R) Core (TM) i7-8705G CPU @ 3.10GHz'. with 4 total Cores , 8 total threads; Max Turbo Frequency , 4.10 GHz, Processor Base Frequency 3.10 GHz, Cache 8 MB, Memory Types DDR4-2400, OS Windows 10

The host is any computer with a CPU running a standard operating system. The OpenCL devices can be a GPU, DSP, or a multi-core CPU. An OpenCL device consists of a collection of one or more compute units (cores). A compute unit is further composed of one or more processing elements. Processing elements execute instructions as SIMD (Single Instruction, Multiple Data) or SPMD (Single Program, Multiple Data). SPMD instructions are typically executed on general purpose devices such as CPUs, while SIMD instructions require a vector processor such as a GPU or vector units in a CPU. The host program uses the OpenCL API to create and manage the context. Functions from the OpenCL API enable the host to interact with a device through a command-queue. Each command-queue is associated with a single device. The commands placed into the command queue fall into one of three types:

i. Kernel-enqueue commands: Enqueue a kernel for execution on a device.
ii. Memory commands: Transfer data between the host and device memory, between memory objects, or map and unmap memory objects from the host address space.
iii. Synchronization commands: Explicit synchronization points that define order constraints between commands.

3.7.4 Programming Tools

The model was also implemented for the parallelization on the CPU by defining the platform device to be 'Intel(R) Core (TM) i7-8705G CPU @ 3.10GHz'. This was possible since OpenCL provides a hardware-agnostic programming model, allowing code to run on a variety of CPU architectures as well as other types of accelerators such as GPUs. The performance of the parallel GPU acceleration was compared to the parallel CPU implementation. Further, a hybrid programming model approach was used for the stochastic optimization algorithms, Genetic Algorithm and Simulated Annealing, which were embedded in the prototype from R but within python using the python wrapper rpy2 which was used to interface the R-based stochastic optimization algorithms within the Python application development environment before the parallel execution strategy was adopted for the dynamic option pricing models.

Chapter 4: System Architecture and Design

4.1 Overview

After conducting a thorough review of literature on parallel GPU derivatives pricing systems and algorithms implementation, and an examination of the benefits and challenges of GPU hybrid implementation and architecture for a derivatives pricing prototype. The following functional and non-functional requirement of the parallel GPU-CPU hybrid derivatives pricing system were summarized.

4.2 Functional Requirements

These functional requirements are descriptions of the inputs, behavior, services and output that the prototype will offer. The following are the functional requirements of the prototype:

a) The prototype should allow upload of user data in csv form with daily stock data, strike price series, call option price series, evaluation period (time-to-maturity), interest rate series, target maturity interest rate, target maturity volatility.
b) The prototype should allow the user to perform calibration of the dynamic option pricing models in parallel on CPU using the Genetic Algorithm and Simulated Annealing Algorithm.
c) The prototype should price the dynamic option pricing model under stochastic volatility and the dynamic option pricing model under stochastic volatility and stochastic interests parallel on GPU, allowing user to set the copula specifications.
d) The prototype should be able to perform performance back tests by RMSE of the dynamic option pricing models.
e) The prototype should allow the user to download the predicted European call options values in a csv file.
f) The prototype should allow the user to create visualization of the predicted European call option values using line plots.
g) The prototype should present the total computational time taken to complete computation on GPU and on CPU.
h) The prototype should allow the user to go back and load a different set of data in csv form.

4.3 Non-functional Requirements

4.3.1 Usability

The graphical user interface (GUI) of the system is designed to be intuitive and easy to navigate, minimizing learning curves and allowing users to efficiently perform tasks. Simplicity and ease of use is also prioritized, with clear labels and prompts to guide the user. Given the complexity of the financial computations and the intricacy of the stochastic models utilized, it is essential that the user interface of the prototype is user-friendly. It simplifies the process of inputting necessary parameters for derivative pricing and risk analysis, and the results are displayed in an easily understandable manner.

4.3.2 System Performance and Reliability

The system consistently perform at high efficiency levels, maintaining its functionality even under high workloads. This involves the efficient use of the parallel GPU-based prototype with a multilanguage programming paradigm on the GPU-CPU architecture, leveraging the latency benefits of Python, the acceleration capabilities of OpenCL, and the powerful R-based libraries to ensure optimal performance. The system's reliability also pertains to its ability to handle errors gracefully and recover quickly from any failure, ensuring uninterrupted operation.

4.3.3 System Scalability and Modification

The system is designed with scalability in mind, allowing it to handle an increasing amount of work by adding resources and complexities. The dynamic pricing algorithm, in particular, is adaptable and able to incorporate more stochastic processes and jumps, as well as more copula specifications for longer time-to-maturity durations and more strike price levels. This adaptability ensures the system can respond to changing requirements of the analyst. Additionally, changes can be made, new features added, and the system updates made with minimal effort and without impacting its overall functionality.

4.3.4 Portability

This prototype system was implemented using OpenCL C, Python, and R. These programming languages are widely used and compatible with a variety of hardware platforms and Windows operating systems. This is beneficial because financial institutions have varying hardware and software environments. The ability to easily deploy the prototype system across different platforms

is valuable in a sector where the use of diverse hardware setups and environments is the norm. Further, given the iterative nature of model development and validation in financial engineering, the prototype is ideally be designed with modular components. This allows for sections of the system, such as the pricing algorithms, to be swapped out or modified without having to alter the entire system.

4.4 System Architecture

The system architecture is described by multilanguage programming approach and the parallel GPU-CPU architecture that was used where the algorithms were developed in a hybrid computing environment with the dynamic pricing algorithms being implemented on GPU using OpenCL and the dynamic model calibration by Genetic Algorithm and Simulated Annealing Algorithms are implemented in parallel on CPU in the hybrid R and Python programming framework. The calibration and pricing algorithms are deployed in OpenCL and R to leverage on GPU acceleration and optimization libraries and to leverage on Python speed and application development utilities. This system architecture allows for efficient development of the application of in multilanguage and multicore architectural paradigm that would ensure optimal optimization algorithm performance as well as application performance.

4.5 System Design

The design modelling of the parallel pricing prototype was done using UML (Unified Modelling Language). The tools for modelling used are use case diagrams, sequence diagrams, data flow diagrams, activity diagrams and entity relationship diagrams.

4.5.1 Use Case Diagram

Use case diagrams show the interactions between the system and its external actor. It is used to model requirements and contexts of a system. Use case diagrams are composed of use cases, actors, associations and a system boundary. Each use case describes a requirement that exists in the system such as input csv stock price series, call price series, strike price series data, input target stochastic volatility, perform calibration and perform pricing. These requirements are shown to relate to other requirements or actors using associations. An association describes a relationship between two diagram elements that represents communication or interaction between them. The interaction between Actors and the proposed prototype have been described in Figure 4.1.

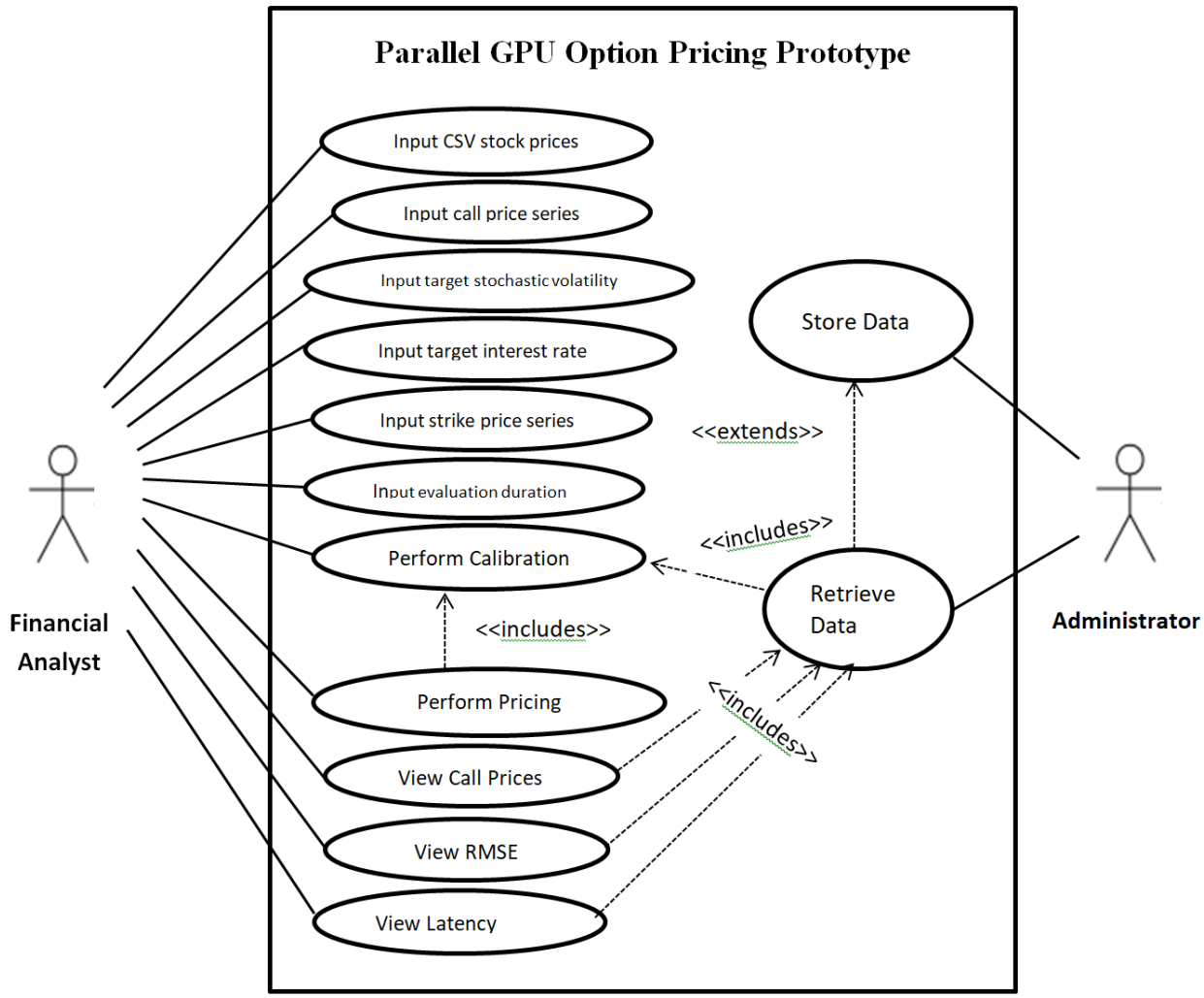

Figure 4.1: Use case diagram of parallel GPU-based option pricing prototype

4.5.2 Data Flow Diagram

The data flow diagram in Figure 4.2 represents the flow of a data from processes in the system prototype. The financial analyst would be able to input the data and the variables. These inputs would then be used in the performing calibration and in the pricing of the European call options using the dynamic option pricing models.

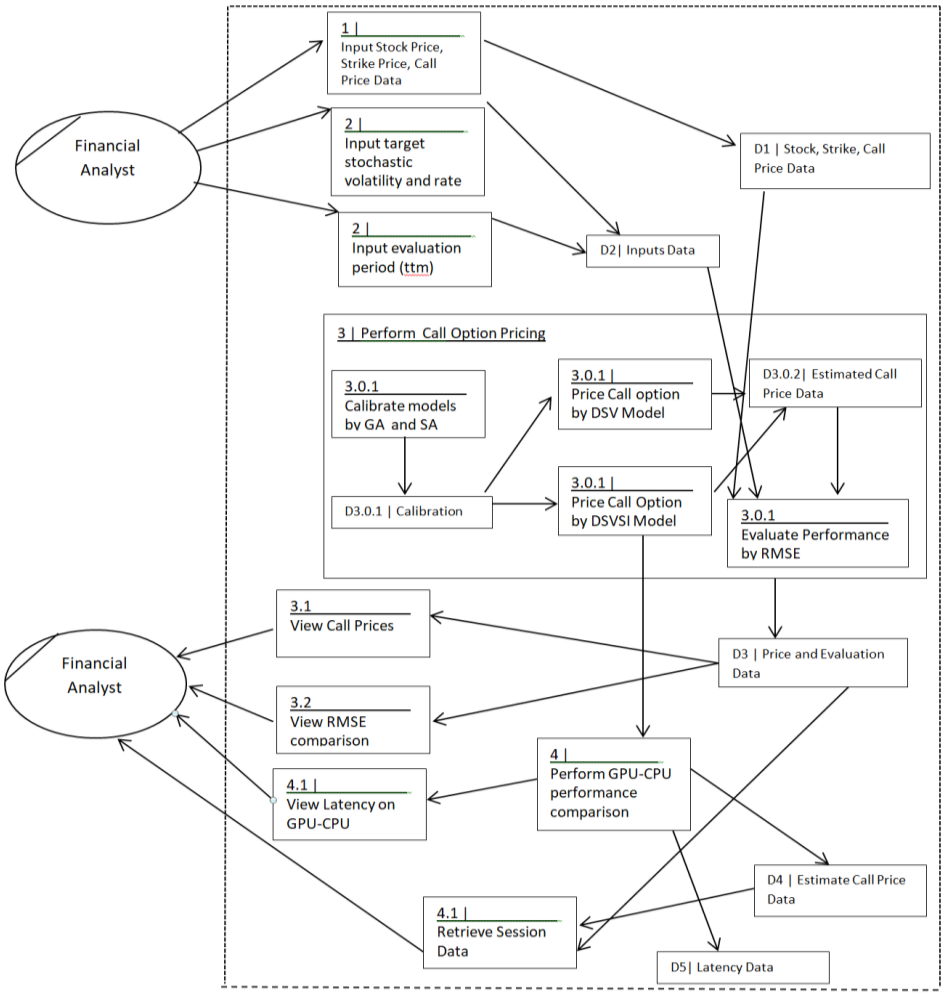

Figure 4.2: Data Flow Diagram

4.5.3 Sequence Diagram

Figure 4.3 presents the sequence diagram which describes the interaction between the prototype system objects in a parallel manner depicting the flow and duration of execution. The financial analyst inserts data of the stock price series, the call prices, the strike price series which is in CSV form. The data is the store in the SQLite database. To perform the calibration and the call option pricing the data is applied to the calibration algorithms and optimal parameters obtained, these optimal parameters are then used in the call option pricing using the dynamic option pricing models under stochastic volatility and stochastic interest rate on GPU. The performance evaluation by RMSE is also performed. The results are then saved.

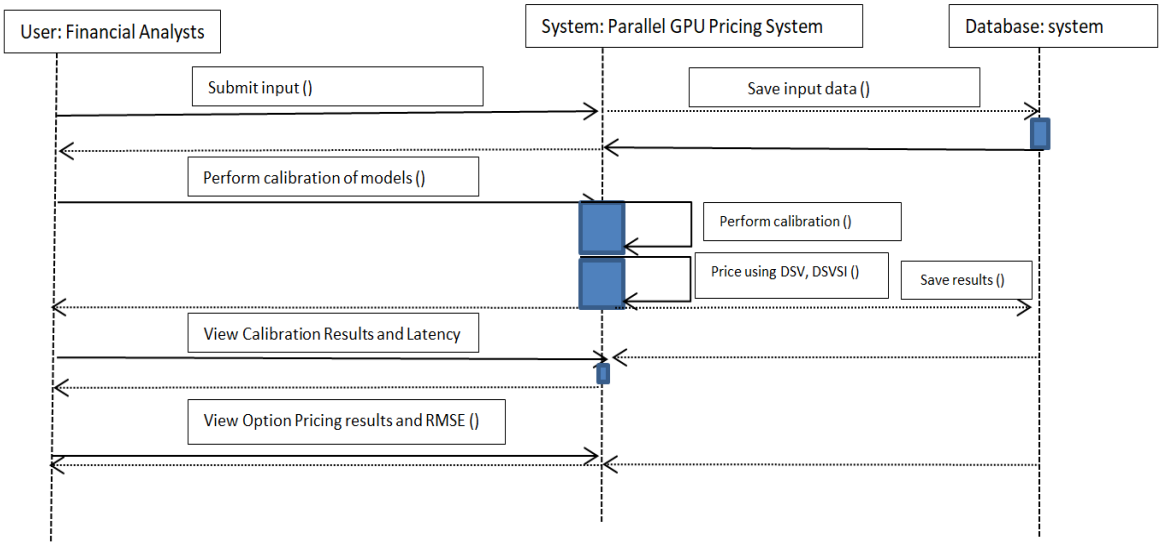

Figure 4.3: Sequence Diagram

4.5.4 User Interface Design

Figure 4.4 describes the user interface for the parallel GPU option pricing prototype. The user is required to input values of target volatility and target interest rate, the evaluation period (time-to-maturity), the prices series for the stock, call prices and strike prices. The user can then proceed to do the calibration with the inputs of the price series for the stock and strike price to obtain optimal parameters by using the Genetic Algorithm or the Simulated Annealing Algorithm or using market estimated parameters for the call option pricing using the dynamic models under stochastic volatility and stochastic interest rates.

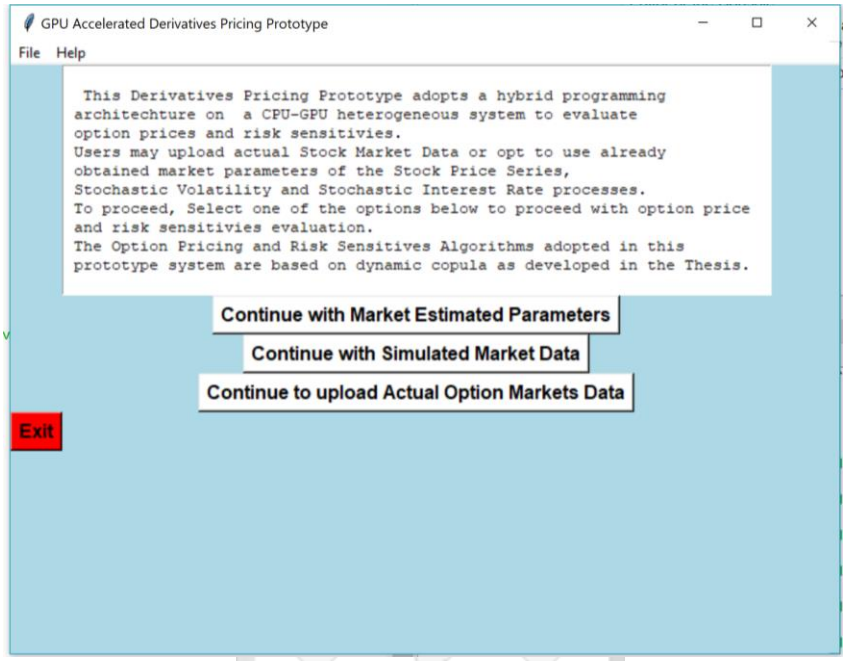

Figure 4.4: User Interface Design

Chapter 5: Implementation and Testing Results

5.1 Overview

This chapter illustrates the implementation, testing and results from the parallel GPU-CPU option pricing prototype. The prototype was developed by using the dynamic option pricing models under stochastic volatility and stochastic interest rates on a parallel GPU-CPU architecture. The data used in the algorithms is that of the 3-Year S&P 500 Index Call Options.

5.2 System Requirements

5.2.1 Hardware Requirements

The hardware requirements for running the prototype have been summarized in Table 5.1.

Table 5.1: Hardware Requirements

Hardware	**Minimum Requirements**
Processor	i5 3rd Generation (used i7 8th Generation)
Number of Processor Cores	8
Cycle Speed	2.5 GHz
Hard Disk Space	250 GB
RAM	4 GB
Graphic Card	Intel, AMD Radeon RX Vega M
Display Resolution	1080x1920

5.2.2 Software Requirements

The software requirement to run the prototype have been summarized in Table 5.2.

Table 5.2: Software Requirements

Software	**Minimum Requirements**
Operating System	Windows 7 (Windows 10 Pro Used)
Anaconda Python	Version 3.7
R	Version 3.6
Windows Excel	2010 Suite
Database	SQLite

5.3 Implementation of Prototype

The Parallel GPU-CPU Pricing Prototype was developed using a hybrid and heterogenous programming approach in Python, R and OpenCL. The pricing kernel functions were offloaded to the GPU and writing in OpenCL language for parallel execution of the dynamic option pricing algorithms. The Genetic Algorithm and Simulated Annealing calibration algorithms were hybridly interfaced into the prototype using the rpy2 library. This integration enabled execution of the dynamic pricing algorithms in parallel on the CPU and GPU and enabled us to leverage the benefits of each programming language, the R computational libraries and Python application development and speed.

5.3.1 Data Inputs

The data inputs that were required for the prototype to perform the pricing and evaluation were loaded in a dataset in CSV form, where the variables were the S&P 500 Index Prices, the Strike Price series, the evaluation period, the US Treasury bill series and the target volatility.

5.3.2 Interface Design

The graphical interface was designed to be easy to use and quite direct where the user is meant to input data in a csv file and then upload the stock price series, the interest rates series, the constant interest rate target, the target volatility at maturity, the evaluation period (time-to-maturity and the strike price series. Once this data has been stored into the prototype then the user can then select to perform calibrations using the Genetic Algorithm or Simulated Annealing algorithm, Price the European options using the Dynamic Option Pricing Model under Square-root stochastic volatility or the Dynamic Option Pricing model under square-root stochastic volatility and CIR interest rates and perform back-tests.

5.4 Implementation Results

In this section, we compared the results obtained from the stochastic global optimization methods for the calibration of the developed dynamic pricing model under square-root stochastic volatility. To overcome the problem with the local minima, the use of the stochastic global optimizer algorithms, in particular simulated annealing and genetic algorithm, were used. The parameters obtained from this optimization training process of the dynamic models to market prices were used in the evaluation of the performance in pricing and hedging of the dynamic model under stochastic volatility. In a bid to accelerate this process of evaluation, a heterogeneous GPU-based architecture

was adopted. A variety of speed and pricing back-tests were performed within this architecture for ATM, ITM and OTM options across maturity levels and strike levels. The results were then compared to the efficiency with a parallel CPU-based implementation.

This study was conducted using the data of daily closing prices for the S&P 500 Index Call Options of maturity of 3 Years with expiry of 18th December 2019 and 85 Strikes price levels from 100 to 3400 USD. The cross-sectional calibration of the models was done at time-to-maturity interval of 3 years, 2 years and 1 year. The evaluation of the performance was performed on the remainder of the option data series. The interest rate of 1.855895% used in the evaluation was obtained as the aggregate of US Treasury Bill rates. Single parameter copula specifications were used in the calibration of the dynamic pricing models.

5.5 Optimization Results for Model Calibration

Table 5.3: Dynamic Models Calibration with Genetic Algorithms

		Copula Specification				
	Parameter	**Gaussian**	**Clayton**	**Gumbel**	**Frank**	**Joe**
3 Years to Maturity	v_0	0.02414154	0.03769998	0.03161835	0.03442013	0.03745968
	κ	0.1572368	0.2332099	0.09984733	0.04210516	0.2982385
	α	0.03713236	0.04218899	0.0555952	0.0358582	0.05743699
	δ	0.5894475	0.4829601	0.6411132	0.5704188	0.7625923
	θ	0.7981766	0.7989597	2.469944	7.145878	4.505357
		Iterations = 127 Value = -2428.926	Iterations = 1000 Value = -1987.245	Iterations = 578 Value = -912.062	Iterations = 716 Value = -1933.692	Iterations = 393 Value = -2688.322
2Years to Maturity	v_0	0.02019178	0.02593533	0.03330974	0.03572781	0.03866438
	κ	0.328062	0.2162289	0.06091254	0.1213961	0.04077598
	α	0.05741426	0.05972117	0.05678468	0.05041373	0.05747357
	δ	0.5728889	0.4651076	0.7094659	0.6716168	0.7859795
	θ	0.7301591	0.6438983	2.14394	6.515574	4.500436
		Iterations = 1000 Value = -2197.222	Iterations = 1000 Value = -2332.688	Iterations = 1000 Value = -2197.222	Iterations = 408 Value = -2181.163	Iterations = 528 Value = -5367.235
1 Year to Maturity	v_0	0.03877616	0.03977456	0.03842599	0.03718965	0.01314576
	κ	0.04140862	1.465543	0.02656108	0.494201	1.436593
	α	0.0547508	0.05938493	0.05951688	0.05939876	0.03001564
	δ	0.3967781	0.3223667	0.7785402	0.7031962	0.2075656
	θ	0.6207339	0.6479486	2.024779	6.500668	4.501837
		Iterations = 147 Value = -2159.424	Iterations = 517 Value = -2362.583	Iterations = 321 Value = -2403.208	Iterations = 995 Value = -2678.133	Iterations = 540 Value = -808.472

Table 5.3: The table above reports the optimal parameters obtained from the calibration of the dynamic stochastic volatility model under square-root stochastic volatility with single parameter copula specifications. The calibration was performed in 3 year, 2 year and 1 year intervals. The optimal parameter

values, iterations and obtained fitness function value of the genetic algorithm are presented for the dynamic stochastic volatility model under various copula specifications.

Table 5.4: Dynamic Models Calibrations with Simulated Annealing Algorithm

		Copula Specification				
	Parameter	**Gaussian**	**Clayton**	**Gumbel**	**Frank**	**Joe**
3 Years to Maturity	v_0	0.029045806	0.017730002	0.034661810	0.035995755	0.03476542
	κ	0.002116922	0.000188893	0.004120945	0.002339804	0.38952781
	α	0.015059194	0.059898017	0.060000000	0.044581672	0.06000000
	δ	0.554791626	0.369703908	0.606154567	0.565962861	0.80000000
	θ_1	0.800000000	0.800000000	2.435638279	7.222355311	4.50000000
		Value=2294.49 Counts= 10717	Value =3375.711 Counts = 13786	Value =1886.956 Counts= 10794	Value =1900.952 Counts = 10959	Value = 2506.292 Counts = 10585
2Years to Maturity	v_0	0.0297977722	0.0253874693	0.03359923	0.03951423	0.04000000
	κ	0.0001395898	0.0001904983	0.0000060063	0.0000459313	0.05870343
	α	0.0581383080	0.0598607803	0.04772559	0.05991218	0.06000000
	δ	0.6017014005	0.3304405396	0.06827138	0.6556445	0.80000000
	θ_1	0.7636318913	0.6999738969	2.134004	6.500049	4.50000000
		Value = 2316.002 Counts = 24874	Value = 2690.865 Counts = 16877	Value= 2062.774 Counts= 14754	Value= 2117.548 Counts=16448	Value=5026.754 Counts= 11080
1Year to Maturity	v_0	0.0400000000	0.0400000	0.0398467608	0.0400000	0.01000000
	κ	0.0000372667	1.4570275	0.0003512607	0.2810997	1.50000000
	α	0.0184498586	0.0600000	0.0598959993	0.0600000	0.03411684
	δ	0.3899583146	0.3145964	0.7866269103	0.8000000	0.20000000
	θ_1	0.6091919042	0.7000000	2.0000000000	6.5000000	4.50000000
		Value = 2147.955 Counts=14391	Value =2345.25 Counts = 10200	Value=2372.65 Counts=13115	Value = 2707.4 Counts = 11058	Value =4756.27 Counts = 10145

Table 5.4: Calibration Results of Dynamic Option Pricing Models with Simulated Annealing Algorithm. The table above reports the optimal parameters obtained from the calibration of the dynamic stochastic volatility model under square-root stochastic volatility with single parameter copula specifications using the simulated annealing algorithm. The calibration was performed in 3 year, 2 year and 1 year intervals. The optimal parameter values, iterations and obtained fitness function value of the simulated annealing algorithm are presented for the dynamic stochastic volatility model under various copula specifications.

The optimal parameters obtained by the Genetic Algorithm and Simulated Annealing calibrations were then used in the pricing of ITM 3-Year S&P Index European Options. The results are shown in Table 5.5 next. The Dynamic Option Pricing Model under square-root stochastic volatility was evaluated under these optimal parameters for ITM options. The RMSE obtained were then compared under various copula specifications (Gaussian, Clayton, Gumbel, Frank and Joe) with the RMSE values obtained by the best performing empirical joe copula formulation. Under the

optimal parameters obtained by the Genetic Algorithms, the performance improvement was found to be higher than under the optimal parameters obtained by Simulated Annealing for ITM options. The Genetic Algorithm parameters had an aggregate of 31.43% performance improvement whereas the Simulated Annealing optimal parameters lead to an aggregate of 27.31% performance improvement across copula specification over the DSV Model 2 under empirical Joe copula. The highest performance improvement was 40.44% under optimal Genetic Algorithm parameter for the Joe copula specification over the DSV Model 2. These optimal parameters were as follows $v_0 = 0.03745968, \kappa = 0.2982385, \alpha = 0.05743699, \delta = 0.7625923$ and $\theta = 4.505357$. The lowest performance improvement under optimal parameter over the DSV Model 2 was under the clayton copula specification at 6.87%. Generally, for ITM option pricing use of optimal parameters obtained by the stochastic optimization algorithms yields a significant performance improvement (Aggregates of 31.43 % for GA and 27.31% for SA) over the DSV Model 2. Cross-copula performance variation is lowest for the implementations using Genetic Algorithms optimal parameters at 0.064629 performance standard deviation and highest for the implementations using Simulated Annealing optimal parameters at 0.121265 performance standard deviation. The Genetic Algorithm optimal parameters yield the best performance for all copula specifications

Table 5.5: Performance Evaluation (RMSE) under optimal parameters obtained by Genetic Algorithm and Simulated Annealing Algorithm

		Dynamic Option Pricing Model under Square-root Stochastic Volatility					
		Gaussian	Clayton	Gumbel	Frank	Joe	Empirical Joe
Genetic Algorithm	*ITM*	56.48826	49.26966	48.58376	46.86639	42.30125	71.02
		20.46 %	30.63%	31.59%	34.01%	40.44%	-
Simulated Annealing	*ITM*	56.501	66.14229	47.41183	45.31924	42.74703	71.02
		20.44%	6.87%	33.24%	36.19%	39.81%	-

Table 5.5: Performance evaluation (RMSE) of DSV Model 2 under optimal parameters by Genetic Algorithm and Simulated Annealing for ITM options. The performance improvement over the DSV Model under empirical joe copula was also computed and presented.

5.6 Parallel Implementation of Dynamic Stochastic Volatility Model on GPU

The Dynamic Option Pricing Model under square-root stochastic volatility was implemented on GPU using OpenCL. The kernel function on the GPU consisted on the pricing model and related

single parameter copula h-function written in OpenCL and offloaded to the GPU for evaluation. For parallel implementation on the CPU program was written hybridly in Python using appropriate *VineCopula* Library within R and multicore processing. The number of contracts and observations was increased from 85 to 10,000 representing 66,300 and 7,800,000 price observations respectively. The time-to-maturities in the evaluation was set to 3 years. These results are presented in the table below, comparing the speeds of the parallel GPU implementation and the parallel CPU-based implementation.

Table 5.6: Evaluation of Parallel Implementation on GPU and CPU of Dynamic Stochastic Volatility Model

Number of Contracts	Obs	Implementation with Time-to-maturity of 3 Years	DSV Model	Acceleration factor on GPU
85	66,300	Parallel on CPU	3.7811 mins	-
		Parallel on GPU	1.3905 secs	x163
100	78,000	Parallel on CPU	4.9540 mins	-
		Parallel on GPU	1.7919 secs	x166
300	234,000	Parallel on CPU	6.6807 mins	-
		Parallel on GPU	2.2205 secs	x181
500	390,000	Parallel on CPU	22.8970 mins	-
		Parallel on GPU	3.1855 secs	x431
1000	780,000	Parallel on CPU	46.24554 mins	-
		Parallel on GPU	5.1269 secs	x541
2000	1,560,000	Parallel on CPU	1.118698 hours	-
		Parallel on GPU	9.4050 secs	x428
5000	3,900,000	Parallel on CPU	1.751028 hours	-
		Parallel on GPU	22.7924 secs	x277
10000	7,800,000	Parallel on CPU	5.968399 hours	-
		Parallel on GPU	85.1690 secs	x252

Table 5.6: Dynamic Stochastic Volatility Model under Square-root Stochastic Volatility Parallel Implementation on GPU vs Parallel Implementation on CPU Performance Comparison on European Options with 3 Years time-to-maturity. The aggregate performance of the parallel implementation on GPU was 273 times faster across number of contracts. The number of options contracts analyzed was increased from 85 to 10,000 throughout this increment of number of contracts the speed of parallel implementation on GPU increase to a maximum of 541 time faster for 1000 contracts where the acceleration factor decreased steadily albeit still being high at 252 times faster for 10,000 option contracts.

5.7 Algorithms Results Discussion

In this sub-section the dynamic stochastic volatility model was calibrated to market prices of 3-Year S&P 500 Index Call Option using stochastic optimization algorithms, Genetic Algorithm and

Simulated Annealing Algorithm. The Genetic Algorithm optimal parameters were found to yield the best performance of the dynamic option pricing model under square-root stochastic volatility where the cross-copula performance deviation was lower at 0.064629 for the Genetic Algorithm as compared to the Simulated Annealing cross-copula performance deviation of 0.121265. The Genetic Algorithm optimal parameter set of $v_0 = 0.03745968, \kappa = 0.2982385, \alpha = 0.05743699, \delta = 0.7625923$ and $\theta = 4.505357$ yielded the best performance improvement over the DSV Model 2 under empirical joe copula at 40.44% performance improvement. Generally, for the ITM option pricing under optimal parameter sets obtained by the stochastic optimization algorithms a significant performance improvement of aggregates of 31.43% and 27.31% were achieved under the Genetic Algorithm and Simulated Annealing Algorithm. The Genetic Algorithm applied to the calibration of the Dynamic Option pricing model under square-root stochastic volatility yielded the best performance under the joe copula configuration of 40.44% improvement over the empirical joe specification for the model.

From the GPU implementation results of the dynamics model under square-root stochastic volatility model, we find that the GPU implementation performance is at least 163 times faster than the parallel CPU implementation. As the number of observations and the number of contracts under observation is increased from 85 contracts to 1000 contracts, the GPU implementation performance peaks at 541 times faster than that of the parallel implementation on the CPU, this reduces the implementation time from 46.24 minutes to 5.12 seconds. When with a maximum number of contracts under evaluation is set to 10,000 contracts the implementation time on the GPU becomes 85.17 seconds from 5.97 hours. This indicates a 252 times faster evaluation speed for the GPU based implementation over the parallel CPU approach. The aggregate performance of the parallel implementation on GPU was 273 times faster across number of contracts. As the number of option contracts under implementation is increased from 85 contracts to 1000 contracts the computation speeds are reduced from 3.78 minutes to 1.39 seconds and from 46.24 minutes to 5.13 seconds, this represents an acceleration of 163 times and 541 respectively. As the number of contracts are increased from 2,000 through to 10,000, the acceleration decreases gradually albeit still being high from1.12 hours to 9.40 seconds and 5.97 hours to 85.17 seconds representing a computational speed acceleration 428 times and 252 times respectively.

An optimal peak in acceleration performance of 541 times is witnessed, and then a slow decline is seen in the acceleration as the number of contracts and observations are increased. This could have resulted from the overhead of kernel launches as the parallelism is increased such that the number of launches in the compute GPU processing units also grow. This can also be explained by the fact that as the degree of parallelism increases, the available computing resources such as the CPU cores, GPU compute units and memory bandwidth may have become saturated, leading to diminishing returns and suboptimal acceleration.

This experimental evaluation of the implementation of the dynamic stochastic volatility model on the GPU shows that the proposed approach is efficient and scalable, which achieves almost a linear speedup across number of option contracts and observations. There were no differences in accuracy of the evaluation between the GPU implementation and those achieved by the parallel CPU approach. The highly parallel structures of GPUs make them more effective than the traditional CPU for the compute intensive segments of the application. This results favor considerable the use of GPU in acceleration of financial applications more so when model throughput and speed are desired for novel complex pricing models. The efficient utilization of GPUs and alternative high-performance architectures and techniques in the pricing of financial derivatives using various numerical algorithms presents a research area worth further exploration whose benefits would be incredible in the pricing, hedging, trading and analytical and energy efficiencies and cost savings for the relevant financial institutions and their divisions.

Chapter 6: Conclusion and Recommendations

6.1 Summary of Conclusions

This thesis proposes a copula-based framework to develop dynamic option pricing algorithms under stochastic volatility and stochastic interest rates. This framework allows for pricing options under stochastic dynamics and jumps. The framework allows for different specifications of the driving stochastic processes and different specifications of the copula function. It is generic in the view that various specifications of the spot price, volatility and interest rates processes can be used and copula function specifications can be applied.

The novel models developed and presented in this framework were calibrated using the Genetic Algorithm and Simulated Annealing algorithm and the optimal parameters were obtained. The Genetic Algorithm under the optimal parameters in joe copula specification for the dynamic model under stochastic volatility was found to be the best performing over the dynamic option pricing model under stochastic volatility under empirical inputs. Incorporation of stochastic interest rates feature improves the performance of the dynamic model under stochastic volatility which had shown significant performance improvement over the Heston Model and the Black-Scholes Model by an aggregate of 33.44% across moneyness and copula specifications.

More specifically following the thesis objectives the following conclusions are drawn:

a) Dynamic Option Pricing Models under Stochastic Dynamics

The developed closed-form dynamic option pricing formula under Gaussian mean-reverting stochastic volatility and square-root stochastic volatility were empirically evaluated and their performance compared to the performance of the Heston (1993) Model and the static Black-Scholes (1973) Model. The dynamic models were found to perform significantly better and consistently across moneyness levels and as well as across copula specifications. A performance improvement of an aggregate of 6% and 10% for ITM option, 5% and 9% for ATM options. Across all moneyness levels and empirical copula specifications an aggregate of 5.3% and 13% performance improvement was achieved respectively above the Heston Model. The dynamic pricing model with square-root stochastic volatility (DSV Model 2) was the best ranking perform across moneyness at 13% overall performance improvement. OTM option prices derived from the

various copula specifications were sensitive to the choice of the copula fitted, with ITM and ATM option prices being rigid to this choice.

Under the stochastic interest rates feature, the two models – the dynamic mean-reverting square-root stochastic volatility model with CIR interest rates (DSVSI Model 1) and the dynamic mean-reverting square-root stochastic volatility model with vasicek interest rates (DSVSI Model 2) were evaluated empirically and the pricing errors (RMSE) for the 3-year European Call options were obtained and compared to those achieved by the DSV Model 2 and the Heston model. The results indicated that generally by the inclusion on stochastic interest rates, the two models performed significantly better than the Heston (1993) and the DSV Model 2 across moneyness levels and across the various copula (Joe, Gaussian, Student t, Clayton, Gumbel and Frank) specifications used. The DSVSI Model 2 with vasicek stochastic interest rates performed on aggregate better than the DSVSI Model 1 with CIR stochastic interest rates, in comparison to the Heston Model the DSVSI Model 1 achieved an aggregate performance improvement of 33.44% across moneyness and copula specification whereas the DSVSI Model 1 achieved 31.64% performance improvement across copula specifications and moneyness levels over the Heston Model.

The highest variability of performance due to copula specification in the two models was witnessed for the DSVSI Model 2 at 0.04574 and the lowest was 0.001614 for ITM options. This indicated the sensitivity of the dynamic approach to the specification of the stochastic interest rates process and copula function. The combination of modelling the stochastic interest rates as following a gaussian mean-reverting process and the stochastic volatility as following a mean-reverting square-root process yielded 1.8% performance gain for the DSVSI Model 2. Under the Clayton copula specifications, the models achieved the best performance with DSVSI Model 1 reporting at 41.80% performance and DSVSI Model 2 reporting a 40.43% performance improvement for OTM options. Therefore at some copula specifications the DSVSI Model 1 performed better than the DSVSI Model 2. In both the models however, the Clayton copula specification brought the highest aggregate performance improvement at 22.79% and 24.27% for the DSVSI Model 1 and DSVSI Model 2 respectively over the dynamic mean-reverting square-root stochastic volatility (DSV Model 1 under Joe copula).

The dynamism of the proposed copula framework for pricing options allows for pricing under stochastic of interest rates, stochastic volatility and jumps in a tractable and easily practically

implementable and fully analytical manner. It exhibits transcendence and superiority over previous pricing models and frameworks under stochastic volatility, stochastic interest rates and jumps. . It also allows for liberty in the choice of copula h-function such that no restrictive modelling assumption is postulated about the nature of dependence between the stochastic volatility process and the underlying price process. This allows for a myriad of alternative implementations under variant copula and stochastic volatility configurations – a modelling panacea of its sort for option pricing under stochastic dynamics. It thereby provides a substantive and suitable alternative framework for option pricing under stochastic volatility, stochastic interest rates and jumps for vanilla or exotic options under varied dependence structures and stochastic dynamics. The underlying processes of the stochastic interest rates and stochastic volatility are easily malleable in this dynamic pricing approach where a myriad of alternative combinations can be made for these underlying processes under various copula configurations.

However, while this dynamic copula framework presents a robust solution for option pricing under stochastic dynamics, it is essential to consider the potential limitations of the stochastic processes such as for the DSV 1 model that considers a Gaussian mean-reverting model that may theoretical imply possibility of negative values. Further, the framework's performance and applicability under various market conditions, along with its scalability and computational efficiency, should be thoroughly investigated under different types of option contracts in different markets, underlying assets and time-to-maturity. Also, the potential of integrating emerging machine learning algorithms to further optimize the parameters of the dynamic copula framework could also be explored, potentially enhancing its accuracy and efficiency. In summary, the proposed dynamic copula framework represents a significant leap forward in option pricing under stochastic dynamics, offering flexibility, practicality, and analytical tractability.

b) Calibration of Dynamic Option Pricing Models by Stochastic Optimization

These results of this investigation reveal a significant performance improvement when using optimal parameters obtained through Genetic Algorithm (GA) and Simulated Annealing (SA) in the pricing of In-The-Money (ITM) 3-Year S&P 500 Index European options. Notably, the optimization process resulted in a higher performance enhancement when the parameters were derived from the GA compared to those obtained from SA for ITM options. Specifically, the GA-optimized parameters resulted in an aggregate performance improvement of 31.43%, while the

SA-optimized parameters led to a slightly lower improvement of 27.31% across copula specifications over the DSV Model 2 under empirical Joe copula. The maximum performance improvement achieved was 40.44%, observed under the optimal GA parameters for the Joe copula specification over the DSV Model 2.

Despite these promising findings, several limitations warrant mention. Firstly, the current analysis focused solely on ITM 3-Year S&P 500 Index European options. The performance of these optimization methods may vary when applied to different option types, durations, or underlying assets. Further research is required to investigate the robustness of these findings across a broader range of financial derivatives. Secondly, the study utilized only GA and SA for optimization. While these techniques have shown promising results, other optimization methods such as Particle Swarm Optimization, Ant Colony Optimization, or even newer, more advanced algorithms could potentially yield even better results. Future research could therefore aim to compare the performance of various optimization techniques in this context.

Thirdly, the study did not explore the impact of varying the parameters within the GA or SA algorithms themselves. The performance of these algorithms can often be significantly influenced by their internal parameters, such as mutation rate in GA or temperature schedule in SA. An exploration of how variations in these parameters affect the optimization results could be a valuable avenue for future research. In terms of policy implications, these results suggest that financial institutions could significantly improve their option pricing models by incorporating optimization techniques like GA and SA. However, the choice of optimization technique and the setting of its internal parameters should be carefully considered based on the specific requirements of each application. As always, while the adoption of these techniques can lead to improved performance, institutions must also be aware of the computational resources required and ensure the robustness and stability of the algorithms in their specific use case.

In summary, while this research provides strong evidence of the benefits of using GA and SA for parameter optimization in dynamic option pricing models, further investigation is needed to explore the potential of other optimization techniques, the impact of varying algorithm parameters, and the applicability of these findings to other types of financial derivatives.

c) Implementation of Dynamic Option Pricing model in a Parallel GPU-CPU System Architecture

The results of this study elucidate the considerable potential of GPU-accelerated computation in the financial sector, specifically in the application of dynamic option pricing models. The empirical evidence suggests that GPU implementations, is efficient and scalable, outperforming the parallel CPU implementations by achieving nearly linear speedup across varying numbers of option contracts and observations. Moreover, the GPU method maintains the same level of accuracy as the CPU approach while dramatically reducing computation time. A peak performance improvement of 541-fold was observed, reducing calculation times from 46.24 minutes to a mere 5.12 seconds. The highly parallel structures of GPUs make them more effective than the traditional CPU for the compute intensive segments of the application. These results favour considerably the use of GPU in acceleration of financial applications more so when model throughput and speed are desired for novel complex pricing models.

In the GPU implementation of the dynamic option pricing model under stochastic volatility, when the number of contracts was gradually increased from 85 to 10,000 contracts which represented 66,300 and 7,800,000 price observations with the time-to-maturity in the evaluation being 3 years. The parallel GPU implementation was at least 163 faster than the parallel CPU implementation. As the number of observations and the number of contracts under observation is increased from 85 contracts to 1000 contracts, the GPU implementation performance peaks at 541 times faster than that of the parallel CPU implementation on the CPU, this reduces the implementation time from 46.24 minutes to 5.12 seconds. When with a maximum number of contracts under evaluation is set to 10,000 contracts the implementation time on the GPU becomes 85.17 seconds from 5.97 hours. This indicates a 252 times faster evaluation speed for the GPU based implementation over the parallel CPU approach.

An increasing number of financial problems could be solved with GPU acceleration efficiently and accurately. A natural extension of the parallel GPU implementation of the dynamic models would be consideration on architecture that are as heterogeneous as possible with distributed infrastructures composed of CPU/GPU and Clouds, and since OpenCL enables use of a diverse range of accelerators including multi-core CPUs, GPUs, DSPs, FPGAs and dedicated hardware, this distributed infrastructure could also be used. The implementation on GPU has the benefit of

being less an expensive architecture, but limited in terms of memory as the GPU does not offer much memory as the CPU. The efficient utilization of GPUs and alternative high performance architectures and techniques in the pricing of financial derivatives using various numerical algorithms presents a research area worth further exploration whose benefits would be incredible in the pricing, hedging, trading and analytics efficiencies and savings for the relevant divisions.

However also, it is important to acknowledge the limitations of this study. Firstly, the results were based solely on the implementation of a specific dynamic option pricing model under square-root stochastic volatility. It is thus uncertain whether the significant speed improvements and the maintenance of accuracy would hold for different models or under different volatility structures. Further research is needed to validate these results across a broader range of financial models and scenarios. Secondly, the study was conducted using OpenCL to facilitate GPU programming. While OpenCL is widely supported, it might not be the optimal choice for all hardware configurations or application requirements within a financial institution. Future research could consider exploring alternative programming interfaces, such as CUDA, to compare their performance with that of OpenCL. Thirdly, the current study did not examine the energy efficiency of GPU implementations. Considering that power consumption is a crucial factor in the operation of large-scale financial systems, future studies should investigate this aspect to provide a more comprehensive picture of the benefits and costs associated with GPU-accelerated computation.

In terms of policy implications, the findings of this research suggest that financial institutions should consider investing in GPU technologies to accelerate computationally-heavy tasks such as dynamic option pricing within their financial engineering systems. As competition in the financial sector continues to increase, the ability to quickly and accurately price derivatives can offer a significant competitive advantage. However, policymakers should also be aware of the potential risks associated with reliance on GPU technologies, such as vulnerability to hardware failures or cyber-attacks that may develop. In conclusion, this study presents compelling evidence for the benefits of GPU-accelerated computation in the financial sector. Nevertheless, further research is required to fully understand the potential and limitations of this technology in different contexts and scenarios. Policymakers and financial institutions should carefully consider these factors when planning their future investments in computational resources.

References

Abbas-Turki, L. A., & Lapeyre, B. (2009). American Options Pricing on Multi-Core Graphic Cards. *IEEE BIFE.*

Abbas-Turki, L. A., Vialle, S., Lapeyre, B., & Mercier, P. (2009). High Dimensional Pricing of Exotic European Contracts on a GPU Cluster, and Comparison to a CPU Cluster. *Parallel and Distributed Computing in Finance, IEEE.*

Alexander, C. (2001). Principles of the Skew. *Risk, 14*(1), 529-532.

Amin , K., & Ng, V. (1993). Option Valuation with Systematic Stochastic Volatility. *Journal of Finance, 48*, 881-910.

Amin, K., & Jarrow, R. (1992). Pricing Options on Risky Assets in a Stochatic Interest Rate Economy. *Mathematical Finance, 2*, 217-237.

Bailey, W., & Stulz, R. (1989). The Pricing of Stock Index Options in a General Equilibrium Model. *Journal of Financial and Quantitative Analysis, 24*, 1-12.

Bakshi, G., Cao, C., & Chen, Z. (1997). Empirical Performance of Alternative Option Pricing Models. *Journal of Finance, 52*(5), 2003-2049.

Bakshi, G., Cao, C., & Chen, Z. (2010). Option Pricing and Hedging Performance Under Stochastic Volatility. In C. F. Lee, A. C. Lee, & J. Lee (Eds.), *Handbook of Quantitative Finance and Risk* (pp. 547-572). USA: Springer.

Bakshi, G., & Madan, D. (2000). Spanning and Derivative Security Valuation. *Journal of Financial Economics, 55*, 205-238.

Ball, C. A., & Roma, A. (1994). Stochastic Volatility Option Pricing. *Journal of Financial and Quantitative Analysis, 29*(4), 589-607.

Bates, D. (1996a). Jumps and Stochastic Volatility:Exchange Rate Process Implicit in Deutschemark Options. *Review of Financial Studies, 9*, 69-108.

Bates, D. S. (1991). The Crash of 87: Was it Expected? The Evidence from Options Markets. *Journal of Finance, 46*(3), 1009-1044.

Bates, D. S. (1995). Testing Option Pricing Models. *National Bureau of Economic Research (NBER) Working Papers 5129*.

Bates, D. S. (2001). Empirical Option Pricing: A Retrospection. *University of Iowa and the National Bureau of Economic Research*.

Basermann, A., Kohring, G. A., & Neff, C. (2008). Derivative Pricing as a Business Grid Application using NextGRID Technology. In M. Constantino, A. Brebbia, & M. Larran (Eds.), *Computational Finance and Its Applications III.* WIT Press: Transactions on Information and Communication Technologies.

Bates, D. (1996b). Jumps and Stochastic Volatility: Exchange Rate Processes Implicit in Deutschemark Options. *Review of Financial Studies, 9(1),* 69-108.

Benmammar, B. (2018). *Concurrent, Real-Time and Distributed Programming in Java.* London, United Kingdom: Wiley and Sons, ISTE.

Benguigui, M., & Baude, F. (2012). Towards Parallel and Distributed Computing on GPU for American Basket Option Pricing. *Cloud Computing Technology and Science , IEEE*.

Benmammar, B. (2018). *Concurrent, Real-Time and Distributed Programming in Java.* London, United Kingdom: Wiley and Sons, ISTE.

Black, F., & Scholes, M. (1973). The Pricing of Options and Corporate Liabilities. *Journal of Political Economy, 81*, 37-54.

Boness, A. J. (1964). Elements of Theory of Stock Option Value. *Journal of Political Economy*, 72-163.

Cao, C., Bakshi, S., & Chen, Z. (2010). Option Pricing and Hedging Performance under Stochastic Volatility and Stochastic Interest Rates. In H. o. Management, C. F. Lee, J. Lee, & A. C. Lee (Eds.). Springer.

Carr, P., & Madan, D. (1997). Option Valuation Using the fast Fourier Transform. *Journal of Computational Finance, 2*(4), 61-73.

Cherubini, U., Gobbi, F., Mulinacci, S., & Romagnoli, S. (2012). *Dynamic Copula Methods in Finance.* UK: Wiley and Sons.

Cherubini, U., Luciano, E., & Vecchiato, W. (2004). *Copula Methods in Finance.* West Sussex, UK: John Wiley and Sons.

Chiarella, C., He, X. Z., & Sklibosios Nikitopoulos, C. (2015). *Derivative Security Pricing: Techniques, Methods and Applications.* Springer: Dynamic Modeling and Econometrics in Economics and Finance.

Chorafas, D.N. (2007). *Risk Management Technology in Financial Services: Risk Control, Stress Testing, Models and IT Systems and Structures*. Burlington, MA; Oxford: Butterworth-Heinemann

Cont, R. (2001). Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues. *Quantitative Finance, 1*, 223-236.

Cont, R., & da Fonseca, J. (2002). Dynamics of Implied Volatility Surfaces. *Quantitative Finance, 2*, 45-60.

Cont, R., da Fonseca, J., & Durrleman, V. (2002). Stochastic Models of Implied Volatility Surfaces. *Economics Notes, 31*(2), 361-377.

Cont, R., & Tankov, P. (2004) *Financial Modelling with Jump Processes*. Chapman and Hall/CRC.

Cox, J., & Ross, S. (1976). The Valuation of Options for Alternative Stochastic Processes. *Journal of Financial Economics, 3*, 145-166.

Dang, D.M., Christara, C., & Jackson, K. R. (2011). An Efficient GPU-Based Parallel Algorithm for Pricing Multi-Asset American Options. *SSRN*.

Dang, D.M. (2011). Modelling Multifactor Financial Derivatives by a Partial Differential Equation Approach with Efficient Implementation on Graphics Processing Units. *PhD Thesis, University of Toronto.*

Darsow, W., Nguyen, B., & Olsen, E. (1992). Copulas and Markov Processes. *Illinois Journal of Mathematics, 36*(4), 600-642.

Di Pierro, M. (2014): Portable Parallel Programs with Python and OpenCL. *Computing in Science & Engineering*, 16, 34–40.

Dufresne, P. C., Kierstead , W., & Ross, M. P. (1996). Pricing Derivatives the Martingale Way. *Institute of Business and Economics Research*.

Dumas, B., Fleming, J., & Whaley, R. (1998). Implied Volatility Function: Empirical Tests. *Journal of Finance, 53*(6), 2059-2106.

Fatone, L., Giacinti, M., Mariani, F., Recchioni, M. C., & Zirilli, F. (2012): Parallel option pricing on GPU: Barrier Options and Realized Variance Options. *Journal of Supercomputing*, 62 (3), 1480–1501.

Fouque, P., Papanicolaou, G., & Sircar, K. R. (1999). Financial Modelling in a Fast Mean-Reverting Stochastic Volatility Environment. *Asia-Pacific Financial Markets, 6*, 37-48.

Fouque, J. P., Papanicolau, G., Sircar, R., & Solna, K. (2011). *Multiscale Stochastic Volatility for Equity.* New York, USA: Cambridge University Press.

Ganesan, N., Chamberlain, R.D., & Buhler, J. (2009). Acceleration of Binomial Options Pricing via Parallelizing along Time-axis on a GPU. *Proceedings of Symposium on Application Accelerators in High Performance Computing*.

Grauer-Gray, S., Killian, W., Searles, R., & Cavazos J. (2013). Accelerating financial applications on the GPU. *Proceedings of the 6th Workshop on General Purpose Processor Using Graphics Processing Units*, 127–136.

Gupta, A., Chandra, A. K., & Luksch, P. (2016). *Real-Time and Distributed Real-Time Systems: Theory and Applications.* UK: CRC Press, Taylor and Francis Group.

Harris, M. (2008, March 11). Grid Computing Adoption in the Financial Services Sector. Retrieved from https://www.theglobaltreasurer.com/2008/03/11/grid-computing-adoption-in-the-financial-services-sector/

Harrison, J. M., & Kreps, D. M. (1979). Martingale and Arbitrage in Multiperiod Securities Markets. *Journal of Economic Theory, 20*, 381-408.

Harrison, J. M., & Pliska, S. R. (1981). Martingales and Stochastic Integrals in the Theory of Continuous Trading. *Stochastic Processes and their Applications, 11*(3), 215-260.

Heston, S. (1993). A Closed-form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. *Review of Financial Studies, 6*, 327-343.

Howes, L. (2010). OpenCL: Parallel Computing for CPUs and GPUs. *AMD Advanced Micro Devices*.

Hull, J. C. (2015). Options, Futures and Other Derivatives.

Hull, J., & White, A. (1987a). The Pricing of Options with Stochastic Volatilties. *Journal of Finance, 42*, 281-300.

Johnson, H., & Shanno, D. (1987). Option Pricing when the Variance is Changing. *Journal of Financial and Quantitative Analysis*, 143-151.

Kanniainen, J., Piche, R., & Mikkonen, T. (2009). Use of Distributed Computing in Derivative Pricing. *International Journal of Electronic Finance, 3*, 270-283.

Kanniainen, J., & Koskinen, M. (2017). Distributed Calibration of Option Pricing Models with Multiple Contracts. *Working Paper, Tampere University of Technology, Finland*.

Khanna, A. (2008). *Straight Through Processing for Financial Services.* USA: Elsevier.

Lee, R. W. (2004). Option Pricing by Transform Methods: Extensions, Unification and Error Control. *Journal of Computational Finance, 7*(3), 51-86.

Lee, J., & Kim, S. (2012). Simulation of Multi-Option Pricing on Distributed Computing. *CMES*, 93-112.

Lewis, A. L. (2001). A Simple Option Formula For General Jump-Diffusion and Other Exponetial Levy Processes. *Envision Financial Systems and OptionCity.net*.

Merton, R. C. (1973). Theory of Rational Option Pricing. *The Bell Journal of Economics and Management Science, 4*(1), 141-183.

Merton, R. C. (1974). On the pricing of corporate debt: The risk structure of interest rate. *Journal of Finance*, 29, 449–469.

Merton, R. C. (1976). Option pricing when underlying stock returns are discountinuous. *Journal Financial Economy*, 3, 125–144.

Merton, R. C. (1982). *On the Mathematics and Economics Assumptions of Continuous-Time Models*, in Essays in Honor of Paul Cootner, Prentice Hall, Englewood Cliffs, NJ.

Mikhailov, S., & Nögel, U. (2003). Heston's Stochastic Volatility Model Implementation, Calibration and Some Extensions. *Wilmott Magazine*, 74–79

Milner, R. (1980). *A Calculus of Communicating Systems.* Springer.

Mrázek, M. & Pospíšil, J. (2017). Calibration and Simulation of Heston model. *Open Mathematics*, *15*(1), 679-704.

Musiela, M., & Rutkowski, M. (2005). *Martingale Methods in Financial Modelling.* Springer: Stochastic Modelling and Applied Probability.

Nages Sieslack (4/21/2015): Insider's View of Financial Modelling on HPC Systems. Interview with Erik Vynckier.

Nandi, S. (1996). Pricing and Hedging Index Options under Stochastic Volatility. *Working Paper, Federal Reserve Bank of Atlanta*.

Nelsen, R. B. (2006). *An Introduction to Copulas.* New York: Springer.

Patton, A. (2001). Modelling Time Varying Exchange Rate Dependence Using the Conditional Copula. *UCSD Discussion Paper*.

Peng, Y., Gong, B., Liu, H., & Dai, B. (2010). Parallel Option Pricing with BSDE Method on GPU. *Proceedings of the 9th International Conference on Grid and Cloud Computing*, 191-195.

Podlozhnyuk, V., & Harris M. (2008) Monte Carlo Option Pricing. NVIDIA.

Prayoga, A., & Privault, N. (2017). Pricing CIR Yield Options by Conditional Moment Matching. *Asia-Pacific Financial Markets*, *24*, 19-38.

Renault, E., & Touzi, N. (1996). Option Hedging and Implied Volatilities in Stochastic Volatility Model. *Mathematical Finance, 6*, 279-302.

Sabanis, S. (2003). Stochastic Volatility and the Mean-reverting Process. *Journal of Futures Markets, 23*, 33-47.

Saxena, S., & Gupta, S. (2017). *Practical Real-Time Data Processing and Analytics.* Packt Publishing.

Schepsmeier, U., & Stober, J. (2014). Derivatives and Fisher Information of bivariate copulas. *Statistical Papers,55(2),* 525-542.

Schmelzle, M. (2010). Option Pricing Formulae using Fourier Transform: Theory and Application. *Pfad Integral*. Retrieved from http://pfadintegral.com

Schober , P., Schroder, P., & Wittum, G. (2015). Efficient Parallel Solution Methods for High-dimensional Option Pricin Problems. *Journal of Computational Finance*.

Scott, L. (1987). Option Pricing when the Variance Changes Randomly: Theory, Estimators and Applications. *Journal of Financial and Quantitative Analysis, 22*, 419-438.

Scott, L. (1997). Pricing Stock Options in a Jump-diffusion Model with Stochastic Volatility and Interest Rate: Application of Fourier Inversion Methods. *Mathematical Finace, 7*, 413-426.

Solomon, S., Thulasiram, R.K., & Thulasiraman, P. (2010). Option Pricing on the GPU. *2010 IEEE 12th International Conference on High Performance Computing and Communications (HPCC)*, 289-296.

Sommerville, I. (2006). *Software Engineering.* Pearson International.

Stein, E., & Stein, J. (1991). Stock Price Distributions with Stochastic Volatility. *Review of Financial Studies, 4*, 727-752.

Surkov, V. (2010) Parallel option pricing with Fourier space time-stepping method on graphics processing units. *Parallel Computing*, 36 (7), 372-380.

Suo, S., Zhu, R., Attridge, R., & Wan, J. (2015): GPU Option Pricing. *Proceedings of the 8th Workshop on High Performance Computational Finance.*

Trainor, S., & Crookes, D. (2013). *GPU Acceleration of a Basket Option Pricing Engine. World Congress on Engineering.*

Trobec, R., Vajtesic, M., & Zinterhof, P. (2009). *Parallel Computing: Numerics, Applications and Trends.* London, United Kingdom: Springer.

Wallmeier, M., & Hafner, R. (2000). Dynamics of DAX Implied Volatilities. *University of Augsburg Working Paper*.

Wiggins, J. (1987). Option Values under Stochastic Volatilities. *Journal of Financial Economics, 19*, 351-372.

Yu, J., Sugano, H., Miyamoto, R., & Onoye, T. (2010). GPU Implementation of Efficient Pedestrian Detection based on MCMC. *Joint 5th International Conference on Soft Computing and Intelligent Systems and 11th International Symposium on Advanced Intelligent Systems*, 1624-1629, Okayama, Japan.

Zenios, S. A. (1999). High-perfomance Computing in Finance: The Last 10 Years and the Next. *Parallel Computing*, *25*, 2149-2175.

Zhang, N., Lei, C., & Man, K.L. (2012). Binomial American Option Pricing on CPU-GPU Hetergenous System. *Engineering Letters*.

Zhang, B., & Oosterlee, C.W. (2009). Option pricing with COS method on graphics processing units. *IEEE International Symposium on Parallel & Distributed Processing*, 1-8.

Appendices

Appendix A: Similarity Report

Curiginal

Document Information

Analyzed document	THESIS_MUGANDA_PHD_UPDATED.docx (D127815729)
Submitted	2022-02-14T08:47:00.0000000
Submitted by	
Submitter email	bshibwabo@strathmore.edu
Similarity	2%
Analysis address	bshibwabo.strath@analysis.urkund.com

Sources included in the report

W	URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.829.6508&rep=rep1&type=pdf Fetched: 2022-01-21T14:25:22.2430000	1
SA	**Mémoire_Jumeau_Nicolas.pdf** Document Mémoire_Jumeau_Nicolas.pdf (D13089227)	2
W	URL: http://www.kunitomo-lab.sakura.ne.jp/papers/Kim.pdf Fetched: 2021-09-11T19:02:01.4570000	5
W	URL: http://science.sut.ac.th/mathematics/finance/images/pdf/paper/28-nontiya.pdf Fetched: 2021-12-16T17:32:05.4800000	1
W	URL: https://digitalcommons.bard.edu/cgi/viewcontent.cgi?article=1212&context=senproj_s2019 Fetched: 2020-05-20T04:10:37.0400000	1
W	URL: https://www.chicagofed.org/~/media/publications/working-papers/2009/wp2009-04-pdf.pdf Fetched: 2020-05-24T11:08:10.5200000	2
W	URL: https://arxiv.org/pdf/1511.01460 Fetched: 2021-06-30T20:27:17.9070000	1
SA	**7656899.pdf** Document 7656899.pdf (D34396783)	1
SA	**Jiøí Brejcha- diplomová práce.docx** Document Jiøí Brejcha- diplomová práce.docx (D3976592)	2
SA	**2246075_Dissertation.docx** Document 2246075_Dissertation.docx (D30230210)	1
SA	**draft13.pdf** Document draft13.pdf (D30560512)	5
W	URL: https://repub.eur.nl/pub/26086/2011-1254.pdf Fetched: 2019-12-20T21:08:25.8770000	2

Appendix B: Ethical Clearance

Strathmore UNIVERSITY

28th June 2022

Mr Muganda, Brian
bw.muganda@gmail.com

Dear Mr Muganda,

RE: A Dynamic, Parallel and Distributed Algorithm for Derivatives Pricing and Hedging on GPU-CPU Heterogeneous Systems

This is to inform you that SU-ISERC has reviewed and **approved** your above **SU Masters'** research proposal. Your application reference number is **SU-IERC1216/21.** The approval period is **28th June 2022 to 27th June 2023.**

This approval is subject to compliance with the following requirements:

i. Only approved documents including (informed consents, study instruments, MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-ISERC.
iii. Death and life-threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-ISERC within 48 hours of notification
iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-ISERC within 48 hours
v. Clearance for export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days upon completion of the study to SU-ISERC.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology, and Innovation (NACOSTI) https://research-portal.nacosti.go.ke/ and obtain other clearances needed.

Yours sincerely,

for: **Dr Ben Ngoye,**
Secretary; SU-ISERC

Cc: Prof Fred Were,
Chairperson; SU-ISERC

STRATHMORE UNIVERSITY INSTITUTIONAL
ETHICS REVIEW COMMITTEE
(SU-IERC)
28-Jun-2022
Email:ethicsreview@strathmore.edu
P.O BOX 59857-00200
NAIROBI-KENYA