CHALMERS
UNIVERSITY OF TECHNOLOGY

Multi-Asset Options

A Numerical Study

Master's thesis in Engineering Mathematics and Computational Science

VILHELM NIKLASSON
FRIDA TIVEDAL

Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg
Gothenburg, Sweden 2018

Master's thesis 2018

Multi-Asset Options

A Numerical Study

VILHELM NIKLASSON
FRIDA TIVEDAL

Department of Mathematical Sciences
Division of Analysis and Probability Theory
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2018

Multi-Asset Options
A Numerical Study
VILHELM NIKLASSON
FRIDA TIVEDAL

Supervisor: Docent Simone Calogero, Department of Mathematical Sciences
Examiner: Docent Simone Calogero, Department of Mathematical Sciences

Master's Thesis 2018
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2018

Multi-Asset Options
A Numerical Study
VILHELM NIKLASSON
FRIDA TIVEDAL
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg

Abstract

This thesis compares three methods for numerically pricing multi-asset options, assuming the underlying assets follow a multi-dimensional geometric Brownian motion with constant coefficients. The considered methods are the binomial pricing model, the Monte Carlo method, and the finite element method (FEM) applied to the pricing PDE (the PDE method). It is shown that the binomial model can be used to price both European and American multi-asset options. It is also concluded that the binomial model has a rather fast convergence rate and the results can be further improved by using adaptive mesh refinements. However, the binomial model performs worse for large volatilities. Furthermore, it is found that the Monte Carlo method converges very fast and that the results can be improved by using variance reduction techniques. This method also works well for pricing Asian options due to its simple formula. Even though the Monte Carlo method is shown to be the fastest and most reliable out of the three methods, it does not perform well for larger volatilities. While the binomial pricing model and the Monte Carlo method seem to underestimate the price for large volatilities, the PDE method is shown to be the only method out of the three that gives reliable estimates. However, the method also has the slowest convergence rate out of the three methods when the volatilities are low. This method also needs the most adaptation for each new option.

Keywords: Multi-asset options, Binomial model, Adaptive mesh refinement, Monte Carlo, Variance reduction, Finite element method, Python, FEniCS.

Acknowledgements

First of all, we would like to thank our supervisor Simone Calogero for coming up with the idea for this thesis and for supporting us throughout the process. We are very grateful for all the time that he has spent together with us discussing questions and for giving us valuable feedback and reading suggestions. We would also like to thank our families and friends for always supporting us.

Vilhelm Niklasson and Frida Tivedal, Gothenburg, May 2018

Contents

1 Introduction

An option is a type of financial derivative, meaning that its value depends on the performance of one or several underlying assets. More precisely, an option is a contract between two parties that gives the holder the right (but not the obligation) to buy or sell some amount of the underlying assets at a future time. Options can generally be categorised into being either European or American depending on when they can be exercised. European options can only be exercised at the time of maturity T whereas American options can be exercised at any time prior to maturity.

Both European and American options are heavily traded on the financial markets, both on exchanges and over-the-counter. The buyer (holder) of an option contract pays a premium to the seller (writer) when the contract is stipulated. The size of the premium depends on the prices of the underlying assets $S(t) = (S_1(t), \ldots, S_d(t))$ at the current time $t < T$ and what kind of rights the option entitles the owner. The rights can often be expressed in terms of a payoff function Y. This means that price of an option can be denoted as $V_Y(S(t), t)$, $V(S(t), t)$, $V_Y(t)$ or just V depending on if the payoff, stock prices and current time are emphasised or not.

This thesis presents and compares different pricing methods for multi-asset options in order to determine how well they perform for different types of options and market parameters. Especially the variances and correlations of the underlying assets are varied in order to study the impact on the price computations. To the authors knowledge, there is a lack of research in how different pricing methods compare in terms of speed and precision for multi-asset options. In this thesis, the focus is on European two-asset options but it is possible to generalize many of the ideas to options with even more underlying assets. American multi-asset options are also studied to some extent.

The fundamental idea behind all the option pricing methods is the arbitrage-free principle. This principle implies that neither the holder nor the writer of an option should be able to make a risk-free profit. Compared to single-asset options, multi-asset options are more complicated to price since there are more sources of randomness to take into consideration. In this thesis it is generally assumed that the underlying assets follow correlated geometric Brownian motions with constant parameters. This makes it possible to use the multi-dimensional Black-Scholes formula which is a well-known generalization of the usual single-asset Black-Scholes formula. However, the consistency of the Black-Scholes framework with the real world has been questioned by many researchers. The usage of more sophisticated models have been suggested mainly for single-asset options and they have not been widely used for multi-asset options.

Mainly three different pricing methods are considered in this thesis: the binomial pricing model, the Monte Carlo method, and the finite element method (FEM) applied to the pricing PDE. The trinomial model is also discussed shortly in the chapter about the binomial model. Python, and especially the NumPy package, is used for numerical computations. Moreover, the FEniCS project in Python is used to implement the finite element method. The European two-asset correlation call option is studied throughout the thesis as a reference option. The advantage of using this option as a reference is that there exists a closed pricing formula for it which is easy to evaluate, so the numerical models can be checked before they are implemented for other options. The American two-asset correlation put option, the Asian two-asset call option, and the European two-asset maximum call option are other options that are thoroughly studied for at least one of the pricing methods.

The rest of this thesis is structured as follows: Chapter 2 gives an introduction to financial theory and multi-asset options. Chapter 3 describes the binomial pricing method and different refinement techniques are presented and used to price options. The trinomial model is also introduced in the same chapter. Chapter 4 focuses on the Monte Carlo method and variance reduction techniques are implemented. Chapter 5 is about the pricing PDE and it contains a derivation of the multi-dimensional Black-Scholes formula. Moreover, the finite element method is presented and used to price options. Chapter 6 contains a comparison of all of the pricing methods and we discuss their advantages and disadvantages.

2
Financial Theory and Multi-Asset Options

The first part of this chapter gives a brief introduction to financial theory, especially risk-neutral pricing. The second part of the chapter focuses on multi-dimensional geometric Brownian motions with constant parameters. This is the model that will be used to describe the stock dynamics in this thesis. The final part of this chapter gives a brief introduction to all of the multi-asset options that are studied in the rest of the thesis.

2.1 Risk-neutral pricing

The theory of risk-neutral pricing constitutes the foundation of option pricing. The key idea behind this theory is that neither the buyer nor the seller of an option shall be able to make a risk-free profit. This is also known as the arbitrage-free principle. In general, an investment strategy represented by the portfolio process $\{\Pi(t)\}_{t\geq 0}$ is an arbitrage if it satisfies the conditions in the following definition.

Definition 1 (Arbitrage portfolio)
A portfolio is an arbitrage if its value $\{\Pi(t)\}_{t\geq 0}$ satisfies the following properties for some $T > 0$ [1]:

1. $\Pi(0) = 0$ *almost surely,*
2. $\Pi(T) \geq 0$ *almost surely,*
3. $\mathbb{P}(\Pi(T) > 0) > 0.$

In order to understand the logic behind risk-neutral pricing, assume that an option is sold for $V_Y(t)$ at time t and that the seller invests this amount in a $d+1$ dimensional market consisting of the d underlying assets and a risk-free asset (for example a bond). Moreover, assume that this portfolio is self-financing, i.e., no money is added or withdrawn, and that the investments can be made in such a way that the portfolio value is equal to the payoff Y of the option at the time of maturity. This means that the portfolio is hedging the option and the price $V_Y(t)$ can therefore be considered "fair" (or risk-neutral) since no party is guaranteed to make a profit.

Based on this logic it becomes reasonable to look for the price of a self-financing hedging portfolio when pricing options. In order to find the price of such a portfolio it is necessary to introduce the multi-dimensional Girsanov's theorem. For this purpose, let

$$\{W_1(t)\}_{t\geq 0}, \ldots, \{W_d(t)\}_{t\geq 0}$$

be independent Brownian motions on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and let $\{\mathcal{F}_W(t)\}_{t\geq 0}$ be a filtration generated by $\{W_1(t)\}_{t\geq 0}, \ldots, \{W_d(t)\}_{t\geq 0}$.

Theorem 2.1.1 (Multi-dimensional Girsanov's theorem)
Let $\{\theta_1(t)\}_{t\geq 0}, \ldots, \{\theta_d(t)\}_{t\geq 0}$ be adapted to $\{\mathcal{F}_W(t)\}_{t\geq 0}$ and define

$$Z(t) = \exp\left(-\sum_{j=1}^{d}\int_0^t \theta_j(s)dW_j(s) - \frac{1}{2}\int_0^t ||\theta(s)||^2 ds\right), \tag{2.1}$$

$$\widetilde{W}_k(t) = W_k(t) + \int_0^t \theta_k(s)ds \quad k = 1, \ldots, d, \tag{2.2}$$

where $||\theta(t)||^2 = \theta_1(t)^2 + \cdots + \theta_d(t)^2$. Moreover, let T be a fixed positive time and assume that

$$\mathbb{E}\int_0^T ||\theta(s)||^2 Z^2(s)ds < \infty. \tag{2.3}$$

Under the equivalent probability measure $\widetilde{\mathbb{P}}$ given by

$$\widetilde{\mathbb{P}}(A) = \mathbb{E}[Z(T)I_A], \quad A \in \mathcal{F} \tag{2.4}$$

the processes $\{\widetilde{W}_1(t)\}_{t\geq 0}, \ldots, \{\widetilde{W}_d(t)\}_{t\geq 0}$ are independent Brownian motions and the filtration $\{\mathcal{F}_W(t)\}_{t\geq 0}$ is a non-anticipating filtration for $\{\widetilde{W}_1(t)\}_{t\geq 0}, \ldots, \{\widetilde{W}_d(t)\}_{t\geq 0}$.

A proof of the multi-dimensional Girsanov's theorem is presented in [2]. Before this theorem can be applied to price options it is necessary to make assumptions about how the underlying assets behave and what the risk-free interest rate is. The latter is used to discount values, i.e., to take into consideration the time-devaluation of money. Assume a general model for the stock prices with possibly non-constant parameters and independent Brownian motions described by

$$dS_i(t) = \mu_i(t)S_i(t)dt + \sum_{j=1}^{d}\sigma_{ij}(t)S_i(t)dW_j(t), \quad t \geq 0, \ \ i = 1, \ldots, m, \tag{2.5}$$

where the drift processes $\{\mu_i(t)\}_{t\geq 0}$ and the volatility matrix $[\{\sigma_{ij}(t)\}_{t\geq 0}]$ are adapted to $\{\mathcal{F}_W(t)\}_{t\geq 0}$. Also define the discount process $\{D(t)\}_{t\geq 0}$ as

$$D(t) = \exp\left(-\int_0^t R(s)ds\right), \tag{2.6}$$

where it is assumed that the interest rate process $\{R(t)\}_{t\geq 0}$ is adapted to $\{\mathcal{F}_W(t)\}_{t\geq 0}$. The differential for the discounted stock price becomes

$$d\left(D(t)S_i(t)\right) = D(t)\left[dS_i(t) - R(t)S_i(t)dt\right] \tag{2.7}$$

$$= D(t)S_i(t)\left[(\mu_i(t) - R(t))dt + \sum_{j=1}^{d}\sigma_{ij}(t)dW_j(t)\right], \quad i = 1, \ldots, m. \tag{2.8}$$

Now the definition of a risk-neutral probability measure can be presented which is very much related to the multi-dimensional Girsanov's theorem and the discounted stock process introduced above. The risk-neutral measure has great significance when pricing options and other financial derivatives.

Definition 2 (Risk-neutral measure)
$\widetilde{\mathbb{P}}$ is called a risk-neutral probability measure if

1. *$\widetilde{\mathbb{P}}$ and $\mathbb{P}$ are equivalent (i.e., they agree on which events that have zero probability)*
2. *The discounted stock price $D(t)S_i(t)$ is a martingale under $\widetilde{\mathbb{P}}$ for every $i = 1, \ldots, m$.*

It follows from Theorem 2.1.1 and (2.8) that the conditions in Definition 2 hold if it is possible to find so called market price of risk processes $\{\theta_1\}_{t\geq 0}, \ldots, \{\theta_d\}_{t\geq 0}$ such that

$$d\left(D(t)S_i(t)\right) = D(t)S_i(t)\sum_{j=1}^{d}\sigma_{ij}(t)\left[\theta_j(t)dt + dW_j(t)\right]. \tag{2.9}$$

This is motivated by the possibility to construct an equivalent risk-neutral probability measure $\widetilde{\mathbb{P}}$ as in (2.4) under which $\{\widetilde{W}_1(t)\}_{t\geq 0}, \ldots, \{\widetilde{W}_d(t)\}_{t\geq 0}$ given by (2.2) are Brownian motions. Thus (2.9) can in that case be reduced to

$$d\left(D(t)S_i(t)\right) = D(t)S_i(t)\sum_{j=1}^{d}\sigma_{ij}(t)d\widetilde{W}_j(t), \tag{2.10}$$

which implies that $D(t)S_i(t)$ is a martingale under $\widetilde{\mathbb{P}}$. Equations for finding the market price of risk processes are obtained from equating (2.8) and (2.9), and they are called the market-price of risk equations.

Definition 3 (Market price of risk equations)
The market price of risk equations are given by:

$$\mu_i(t) - R(t) = \sum_{j=1}^{d}\sigma_{ij}(t)\theta_j(t), \quad i = 1, \ldots, m. \tag{2.11}$$

If the market price of risk equations have no solution, then there will be arbitrage opportunities in the market and the model should not be used [2]. Conversely, the market is free of arbitrage if there exists a solution to these equations.

Next consider a self-financing portfolio where investments are made in the stocks $S_i(t)$, $i = 1, \ldots, m$, and a bond with interest rate process $R(t)$. Let $\{\Delta_i(t)\}_{t\geq 0}$ be adapted processes representing the number of shares invested in each stock. The portfolio process $\{\Pi(t)\}_{t\geq 0}$ then satisfies

$$\begin{aligned} d\Pi(t) &= \sum_{i=1}^{m}\Delta_i(t)dS_i(t) + R(t)\left(\Pi(t) - \sum_{i=1}^{m}\Delta_i(t)dS_i(t)\right)dt \qquad (2.12)\\ &= R(t)\Pi(t)dt + \sum_{i=1}^{m}\Delta_i(t)\left(dS_i(t) - R(t)S_i(t)dt\right) \qquad (2.13)\\ &= R(t)\Pi(t)dt + \sum_{i=1}^{m}\frac{\Delta_i(t)}{D(t)}d\left(D(t)S_i(t)\right). \qquad (2.14) \end{aligned}$$

Thus the differential of the discounted portfolio value becomes

$$d\left(D(t)\Pi(t)\right) = D(t)\left(d\Pi(t) - R(t)\Pi(t)dt\right) = \sum_{i=1}^{m}\Delta_i(t)d\left(D(t)S_i(t)\right), \tag{2.15}$$

and since $\{D(t)S_i(t)\}_{t\geq 0}$, $i = 1, \dots, m$, are martingales under $\widetilde{\mathbb{P}}$, the following theorem holds.

Theorem 2.1.2
If $\widetilde{\mathbb{P}}$ is a risk-neutral probability measure, then the discounted self-financing portfolio process $\{D(t)\Pi(t)\}_{t\geq 0}$ is a martingale under $\widetilde{\mathbb{P}}$.

It is now straightforward to give a definition of the fair price of an option. Recall the discussion in the beginning of this section where it was argued that the option price should be equal to the value of a self-financing hedging portfolio. Since the discounted value of any self-financing portfolio is a martingale under a risk-neutral probability measure $\widetilde{\mathbb{P}}$, it holds that

$$D(t)\Pi(t) = \widetilde{\mathbb{E}}[D(T)\Pi(t)|\mathcal{F}_W(t)], \tag{2.16}$$

where $\widetilde{\mathbb{E}}$ denotes the expectation under $\widetilde{\mathbb{P}}$. Since the value of the portfolio should be equal to the payoff Y of the option at maturity T it follows that

$$\begin{aligned}\Pi(t) &= \frac{1}{D(t)}\widetilde{E}\left[D(T)Y|\mathcal{F}_W(t)\right] = \widetilde{E}\left[\frac{D(T)}{D(t)}Y|\mathcal{F}_W(t)\right] &(2.17)\\ &= \widetilde{E}\left[Y\exp\left(-\int_t^T R(s)ds\right)|\mathcal{F}_W(t)\right], &(2.18)\end{aligned}$$

Thus the fair price of a European option (or some other European derivative) can be defined in the following way.

Definition 4 (Risk-neutral pricing formula)
The risk-neutral price at time $t \in [0,T]$ of a European option with payoff Y and time of maturity T satisfies

$$V_Y(t) = \widetilde{\mathbb{E}}\left[Y\exp\left(-\int_t^T R(s)ds\right)|\mathcal{F}_W(t)\right]. \tag{2.19}$$

Generally it is difficult to find an explicit expression of the risk-neutral price. However, in numerical computations it is often useful to take advantage of the fact that each of the stocks has drift rate equal to the rate of return $R(t)$ of the money market under a risk-neutral probability measure $\widetilde{\mathbb{P}}$ [2].

2.2 Multi-dimensional geometric Brownian motions

In order to price multi-asset options it is necessary to decide upon a model for the underlying assets. In this thesis it is assumed that the underlying assets are described by multi-dimensional geometric Brownian motions with constant parameters. Two different ways to represent these motions and the corresponding correlations are commonly used in the literature. For some pricing methods, one of the two representations turn out to be easier to use for numerical implementations.

Definition 5 (Representation 1)
Consider a d-dimensional stock market. Assume that the stock prices $\{S_1(t)\}_{t\geq 0}$, $\ldots$, $\{S_d(t)\}_{t\geq 0}$ *satisfy the following stochastic differential equations (SDEs):*

$$dS_i(t) = \mu_i S_i(t)dt + \sum_{j=1}^{d} \sigma_{ij} S_i(t) dW_j(t), \quad t > 0, \;\; i = 1, \ldots, d, \tag{2.20}$$

where μ_i *are the drift parameters,* $[\sigma_{ij}]$ *is the constant volatility matrix, and* $\{W_i(t)\}_{t\geq 0}$ *are independent Brownian motions in the probability space* $(\Omega, \mathcal{F}, \mathbb{P})$*. That is, it must hold that*

$$Cov[dW_i(t)dW_j(t)] = 0, \quad i,j = 1, \ldots, d, \;\; i \neq j. \tag{2.21}$$

Equation (2.20) is a common way to describe the multi-dimensional geometric Brownian motion with constant parameters. Another way to describe the motions, which is equivalent in distribution, is as follows.

Definition 6 (Representation 2)
Consider a d-dimensional stock market. Assume that the stock prices $\{S_1(t)\}_{t\geq 0}$, $\ldots$, $\{S_d(t)\}_{t\geq 0}$ *satisfy the following stochastic differential equations:*

$$dS_i(t) = \mu_i S_i(t)dt + \sigma_i S_i(t) dW_i^{(\rho)}(t), \quad t > 0, \;\; i = 1, \ldots, d, \tag{2.22}$$

where μ_i *are the drift parameters,* $\sigma_i > 0$ *are the constant volatilities, and* $\{W_i^{(\rho)}(t)\}_{t\geq 0}$ *are Brownian motions in the probability space* $(\Omega, \mathcal{F}, \mathbb{P})$*. Further, the Brownian motions are assumed to be correlated with covariances given by*

$$Cov[dW_i^{(\rho)}(t)dW_j^{(\rho)}(t)] = \rho_{ij}dt, \quad i,j = 1, \ldots, d, \;\; i \neq j. \tag{2.23}$$

For now, we restrict the discussion to the two-asset case, i.e., $d = 2$. It is possible to integrate the systems in Representation 1 and Representation 2 in order to get explicit expressions for the stock prices. It can be shown that the system (2.20) can be integrated to give

$$S_1(t) = S_1(0) \exp\left[\left(\mu_1 - \frac{\sigma_{11}^2 + \sigma_{12}^2}{2}\right)t + \sigma_{11}W_1(t) + \sigma_{12}W_2(t)\right], \tag{2.24}$$

$$S_2(t) = S_2(0) \exp\left[\left(\mu_2 - \frac{\sigma_{21}^2 + \sigma_{22}^2}{2}\right)t + \sigma_{21}W_1(t) + \sigma_{22}W_2(t)\right]. \tag{2.25}$$

Similarly, the system (2.22) corresponding to Representation 2 can be integrated to give

$$S_1(t) = S_1(0) \exp\left[\left(\mu_1 - \frac{\sigma_1^2}{2}\right)t + \sigma_1 W_1^{(\rho)}(t)\right], \tag{2.26}$$

$$S_2(t) = S_2(0) \exp\left[\left(\mu_2 - \frac{\sigma_2^2}{2}\right)t + \sigma_2 W_2^{(\rho)}(t)\right]. \tag{2.27}$$

It is also straightforward to convert between the two representations in this case. First note that the variances for any fixed $t \geq 0$ of the linear combinations of the Brownian motions in Representation 1 are given by

$$\mathrm{Var}[\sigma_{11}W_1(t) + \sigma_{12}W_2(t)] = (\sigma_{11}^2 + \sigma_{12}^2)t, \tag{2.28}$$

$$\mathrm{Var}[\sigma_{21}W_1(t) + \sigma_{22}W_2(t)] = (\sigma_{21}^2 + \sigma_{22}^2)t. \tag{2.29}$$

It is also well known that the sum of independent normal random variables follows a normal distribution. Therefore, for any fixed t in Representation 1, it holds that

$$\sigma_{11}W_1(t) + \sigma_{12}W_2(t) \sim \mathcal{N}(0,(\sigma_{11}^2 + \sigma_{12}^2)t), \quad \forall t \geq 0, \tag{2.30}$$
$$\sigma_{21}W_1(t) + \sigma_{22}W_2(t) \sim \mathcal{N}(0,(\sigma_{21}^2 + \sigma_{22}^2)t), \quad \forall t \geq 0, \tag{2.31}$$

where $\mathcal{N}(\mu,\sigma^2)$ denotes a normally distributed random variable with mean μ and variance σ^2. It follows from the independent increments of $\{W_1(t)\}_{t\geq 0}$ and $\{W_2(t)\}_{t\geq 0}$ that

$$\sigma_{11}W_1(t) + \sigma_{12}W_2(t) = \sqrt{\sigma_{11}^2 + \sigma_{12}^2}W_1^{(\rho)}(t), \tag{2.32}$$
$$\sigma_{21}W_1(t) + \sigma_{22}W_2(t) = \sqrt{\sigma_{21}^2 + \sigma_{22}^2}W_2^{(\rho)}(t), \tag{2.33}$$

where $\{W_1^{\rho}(t)\}_{t\geq 0}$ and $\{W_2^{\rho}(t)\}_{t\geq 0}$ are correlated Brownian motions. Moreover, since $\{W_1(t)\}_{t\geq 0}$ and $\{W_2(t)\}_{t\geq 0}$ are independent in Representation 1, it follows that

$$\mathrm{Cov}[\sigma_{11}W_1(t) + \sigma_{12}W_2(t), \sigma_{21}W_1(t) + \sigma_{22}W_2(t)] = (\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22})t \tag{2.34}$$

and thus

$$\mathrm{Cor}[\sigma_{11}W_1(t) + \sigma_{12}W_2(t), \sigma_{21}W_1(t) + \sigma_{22}W_2(t)] = \frac{\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}}{\sqrt{(\sigma_{11}^2 + \sigma_{12}^2)(\sigma_{21}^2 + \sigma_{22}^2)}}. \tag{2.35}$$

Hence in the case of two assets, given the covariance matrix $[\sigma_{ij}]$ in Representation 1, the corresponding parameters in Representation 2 are given by

$$\sigma_1 = \sqrt{\sigma_{11}^2 + \sigma_{12}^2}, \tag{2.36a}$$
$$\sigma_2 = \sqrt{\sigma_{21}^2 + \sigma_{22}^2}, \tag{2.36b}$$
$$\rho = \frac{\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}}{\sqrt{(\sigma_{11}^2 + \sigma_{12}^2)(\sigma_{21}^2 + \sigma_{22}^2)}}. \tag{2.36c}$$

Conversely, given the parameters σ_1, σ_2 and ρ in Representation 2, the corresponding parameters in Representation 1 are obtained by solving a system of equations. This system will be underdetermined since it has four unknowns and three equations.

2.3 Examples of multi-asset options

There are many types of multi-asset options. Some of these options are path-dependent whereas other only depend on the final prices of the underlying assets. Generally, an option can also take the form of either a call or a put depending on if the holder makes a profit when the prices of the underlying assets increase or decrease. Options can also be classified into being either European or American depending on when they can be exercised.

The European two-asset correlation call option is studied throughout the thesis in order to compare the different pricing methods. There are a couple of advantages of using this option as a reference: (i) there exists a closed formula which is fast to evaluate so the numerical computations can easily be checked; (ii) most of the pricing methods discussed in this thesis can be implemented without any difficulties for this option.

Other options are studied along with the European two-asset correlation call option in this thesis. This includes American two-asset correlation put options, Asian two-asset call options, and European two-asset maximum call options. These options are not implemented for all of the pricing methods but rather used to see how the pricing methods work for different options. This section gives a brief introduction to each of these options.

2.3.1 Two-asset correlation option

The European and American two-asset correlation options are well-known examples of multi-asset options that depend on two underlying assets. Let $S(t) = (S_1(t), S_2(t))$ be the prices at time t of the two underlying assets and denote the strike prices by K_1 respectively K_2. The payoff Y_{call} at the time of maturity T of the call option is given by

$$Y_{\text{call}}(S(T)) = \begin{cases} \max\{S_2(T) - K_2, 0\} & \text{if } S_1(T) > K_1, \\ 0 & \text{otherwise,} \end{cases} \tag{2.37}$$

and the corresponding payoff Y_{put} of the put option is given by

$$Y_{\text{put}}(S(T)) = \begin{cases} \max\{K_2 - S_2(T), 0\} & \text{if } S_1(T) < K_1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.38}$$

The difference between a European and an American two-asset correlation option is that the former can only be exercised at the time of maturity whereas the latter can also be exercised at any time prior to maturity.

There exist closed formulas for computing the prices of European call and put versions of this option. Assume that the underlying assets are described by Representation 2. Then it can be shown that the fair price $V_{Y_{\text{call}}}$ of the call option is given by the formula [3]

$$V_{Y_{\text{call}}} = S_2(t) M\left(y_2 + \sigma_2\sqrt{T-t}, y_1 + \rho\sigma_2\sqrt{T-t}; \rho\right) - K_2 e^{-r(T-t)} M(y_2, y_1; \rho) \tag{2.39}$$

where M is the cumulative bivariate normal distribution function given by

$$M(a, b; \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \int_{-\infty}^{a} \int_{-\infty}^{b} \exp\left[-\frac{x^2 - 2\rho xy + y^2}{2(1-\rho^2)}\right] dxdy, \tag{2.40}$$

r is the constant risk-free interest rate, ρ is the correlation coefficient between the two underlying assets, and

$$y_1 = \frac{\log(S_1(t)/K_1) + (r - \sigma_1^2/2)(T-t)}{\sigma_1\sqrt{T-t}}, \quad y_2 = \frac{\log(S_2(t)/K_2) + (r - \sigma_2^2/2)(T-t)}{\sigma_2\sqrt{T-t}}. \tag{2.41}$$

Similarly, the price $V_{Y_{\text{put}}}$ of the European put option is given by

$$V_{Y_{\text{put}}} = K_2 e^{-r(T-t)} M(-y_2, -y_1; \rho) - S_2(t) M\left(-y_2 - \sigma_2\sqrt{T-t}, -y_1 - \rho\sigma_2\sqrt{T-t}; \rho\right). \tag{2.42}$$

The pricing formulas (2.39) and (2.42) need to be modified if it is assumed that the underlying assets are described by Representation 1 instead of Representation 2. In the case of a call option, the price will then be given by

$$\begin{aligned} V_{Y_{\text{call}}} = S_2(t)M\left(y_2 + \sqrt{\sigma_{21}^2+\sigma_{22}^2}\sqrt{T-t}, y_1 + \rho\sqrt{\sigma_{21}^2+\sigma_{22}^2}\sqrt{T-t}; \rho\right) \\ -K_2e^{-r(T-t)}M(y_2, y_1; \rho), \end{aligned} \tag{2.43}$$

where

$$\rho = \frac{\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}}{\sqrt{(\sigma_{11}^2+\sigma_{12}^2)(\sigma_{21}^2+\sigma_{22}^2)}}, \tag{2.44}$$

and σ_1 and σ_2 in y_1 and y_2 are replaced by $\sqrt{\sigma_{21}^2+\sigma_{22}^2}$ respectively $\sqrt{\sigma_{21}^2+\sigma_{22}^2}$ in (2.41). A similar transformation also holds for the pricing formula of the put option.

2.3.2 Asian call option

For an Asian option, the payoff depends on an average of the stock prices up to time of maturity. Assuming an Asian call option with only one underlying asset and fixed strike price K, the payoff Y_d in the discrete arithmetic case is given by

$$Y_d = \left(\frac{1}{N+1}\sum_{j=0}^{N} S(t_j) - K\right)^+ \tag{2.45}$$

where $S(t_j)$ is the price of the asset at time t_j and $0 = t_0 < \cdots < t_N = T$ are a discrete set of monitoring dates. In the continuous arithmetic case, the payoff Y_c becomes

$$Y_c = \left(\frac{1}{T}\int_0^T S(\tau)d\tau - K\right)^+ \tag{2.46}$$

where the average is computed over an interval $[0, T]$ [4].

It is possible to generalise the single asset Asian call option to the case when there are several underlying assets. Following the approach in [5], assume d underlying assets and monitoring dates $0 = t_0 < \cdots < t_N = T$. Let $S_i(t_j)$ be the price at time t_j of asset S_i. The arithmetic payoff Y_d in the discrete case is then given by

$$Y_d = \left(\frac{1}{N+1}\sum_{j=0}^{N}\left[\sum_{i=1}^{d}\alpha_i S_i(t_j)\right] - K\right)^+ \tag{2.47}$$

where α_k are weights satisfying $\sum_{i=1}^{d}\alpha_i = 1$. The corresponding continuous arithmetic case for the interval [0,T] has payoff

$$Y_c = \left(\frac{1}{T}\int_0^T\left[\sum_{i=1}^{d}\alpha_i S_i(\tau)\right]d\tau - K\right)^+ . \tag{2.48}$$

It is not possible to reduce the Asian option pricing problem to a one-dimensional problem, thus there exists no simple pricing formula for this option.

2.3.3 Maximum and minimum call option

Maximum call options and minimum call options are European or American multi-asset options that offer its holder the right to get the maximum respectively minimum payoff of a set of standard call options on the underlying assets. Thus the payoff $Y_{\max}$ of the maximum call option with d underlying assets with prices $S(t) = (S_1(t), \dots, S_d(t))$ is given by

$$Y_{\max}(S(T)) = \max\left\{(S_1(T) - K_1)^+, \dots, (S_d(T) - K_d)^+\right\}, \tag{2.49}$$

where K_i can be thought of as the strike price of the corresponding underlying asset S_i. Analogously, the payoff of a minimum call option is given by

$$Y_{\min}(S(T)) = \min\left\{(S_1(T) - K_1)^+, \dots, (S_d(T) - K_d)^+\right\}. \tag{2.50}$$

To the best of the authors knowledge, there exists no way to reduce the pricing problem of a maximum or minimum call option to a one-dimensional problem. However, in the case of two underlying assets $S(t) = (S_1(t), S_2(t))$, we can derive a "max-min parity" of the option prices. Consider a European maximum call option and a European minimum call option with the same two underlying assets. At the time of maturity T it holds that

$$Y_{\max}(S(T)) + Y_{\min}(S(T)) = (S_1(T) - K_1)^+ + (S_2(T) - K_2)^+. \tag{2.51}$$

This means that the value at the time of maturity of a maximum and a minimum call option is equal to the value of two standard European call options (one on each underlying asset). It follows from the arbitrage-free principle that the same relationship must hold at any time prior to maturity. Let $V_{Y_{\max}}(S(t), t)$ and $V_{Y_{\min}}(S(t), t)$ denote the values of the maximum respectively minimum call option at time t. Denote also the values at time t of European call options on S_1 and S_2 with strike prices K_1 respectively K_2 and time of maturity T by $C(S_1(t), t, K_1, T)$ respectively $C(S_2(t), t, K_2, T)$. It holds that

$$V_{Y_{\max}}(S(t), t) + V_{Y_{\min}}(S(t), t) = C(S_1(t), t, K_1, T) + C(S_2(t), t, K_2, T) \tag{2.52}$$

for all $t \in [0,T]$.

The values of the call options on the right hand side of (2.52) can easily be calculated by the Black-Scholes formula assuming that the underlying assets are described by geometric Brownian motions with constant parameters [2].

3
The Binomial Pricing Model

The binomial pricing model has been widely used to find the price of single-asset options due to its simplicity. It is also possible to use a generalisation of this model to price both European and American multi-asset options. The generalised model has not been studied as extensively, but is has been examined in for example [6].

Consider a European option with underlying asset prices $S(t) = (S_1(t), \dots, S_d(t))$ and assume that these assets are described by Representation 2. In order to simplify future numerical implementations, it is useful to introduce the processes

$$X_k(t) = \log(S_k(t)) - \left(\mu_k - \frac{1}{2}{\sigma_k}^2\right) t, \tag{3.1}$$

which can be rewritten by using Itô's formula as

$$X_k(t) = X_k(0) + \sigma_k W_k^{\rho}(t), \quad t > 0, \;\; k = 1, \dots, d. \tag{3.2}$$

It follows from the arbitrage-free principle that the value $V_Y(S(t), t)$ of the option is given by the discounted conditional expectation of the payoff under the risk-neutral probability measure. Using the notation $X(t) = (X_1(t), \dots, X_d(t))$ and setting $V_Y(S(t), t) = V_Y(X(t), t)$ gives

$$V_Y(X(t), t) = e^{-r(T-t)} \mathbb{E}[Y(X(T)) | X(t) = x], \tag{3.3}$$

where $Y(X(T))$ denotes the payoff function at time of maturity T. The constant drift rate is set to be equal to r for the process $\{X(t)\}_{t \geq 0}$ in order to satisfy the risk-neutral condition.

3.1 Uniform method for two underlying assets

Consider the case of a European option with two underlying assets and time to maturity T. Let the interval $[0, T]$ be partitioned into $N \in \mathbb{N}$ equally sized sub-intervals of length $\Delta t = T/N$ and set $t_n = n\Delta t$, $n = 0, 1, \dots, N$. In the binomial pricing model adapted to this framework it is assumed that the combined price movement of the underlying assets will move in one out of four directions at each t_n, $n = 1, \dots, N$. The prices of both assets can either go up (uu), down (dd), or go in opposite directions (ud and du). The corresponding probabilities for each of these four directions are denoted by $\mathbb{P}_{uu}, \mathbb{P}_{dd}, \mathbb{P}_{ud}$, and $\mathbb{P}_{du}$. By denoting the increments for the two assets by h_1 respectively h_2, the four possible branches from t_n to t_{n+1}

can be expressed as

$$X^{n+1} = \Big\{(X^n_{1,(i,j)} + h_1, X^n_{2,(i,j)} + h_2), (X^n_{1,(i,j)} - h_1, X^n_{2,(i,j)} - h_2),$$
$$(X^n_{1,(i,j)} + h_1, X^n_{2,(i,j)} - h_2), (X^n_{1,(i,j)} - h_1, X^n_{2,(i,j)} + h_2)\Big\},$$

where $X^n_{k,(i,j)}$, $k = 1, 2$, denotes the transformed asset price corresponding to (3.1) at time t_n. Observe that this X represents a discrete version of the process introduced in the previous section. By introducing a $(n+1) \times (n+1)$ rectangular mesh with all possible stock price combinations at time t_n, the index (i, j) is used to distinguish the current stock prices at different nodes. In such a matrix, it is assumed that the price of the first underlying asset only increases when increasing the column index (j) whereas the price of the second asset only increases when decreasing the row index (i). The possible movements, their corresponding probabilities, and the rectangular mesh are illustrated for the first two steps in Figures 3.1 and 3.2.

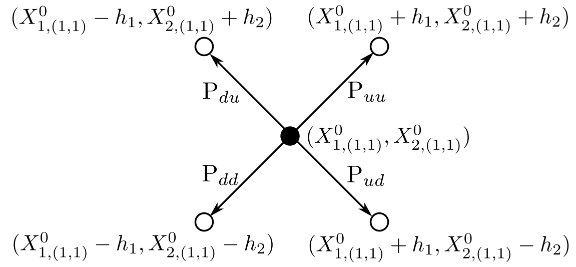

$(X^0_{1,(1,1)} - h_1, X^0_{2,(1,1)} + h_2)$
$(X^0_{1,(1,1)} + h_1, X^0_{2,(1,1)} + h_2)$
P_{du}
P_{uu}
$(X^0_{1,(1,1)}, X^0_{2,(1,1)})$
P_{dd}
P_{ud}
$(X^0_{1,(1,1)} - h_1, X^0_{2,(1,1)} - h_2)$
$(X^0_{1,(1,1)} + h_1, X^0_{2,(1,1)} - h_2)$

Figure 3.1: Possible paths in the two-asset binomial model and corresponding probabilities at t_1, i.e., the first step. The filled circle is the present state (t_0) and the empty circles are the possible new states at t_1.

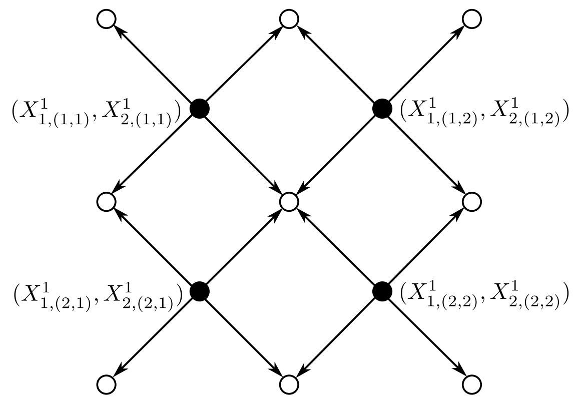

$(X^1_{1,(1,1)}, X^1_{2,(1,1)})$
$(X^1_{1,(1,2)}, X^1_{2,(1,2)})$
$(X^1_{1,(2,1)}, X^1_{2,(2,1)})$
$(X^1_{1,(2,2)}, X^1_{2,(2,2)})$

Figure 3.2: Possible paths in the two-asset binomial model at t_2, i.e., the second step. The filled circles are states at t_1 and the empty circles are the possible new states at t_2 depending on the state at t_1.

The unknown parameters $\mathbb{P}_{uu}, \mathbb{P}_{dd}, \mathbb{P}_{ud}$, $\mathbb{P}_{du}$ and the increments h_1 and h_2 can be determined by examining the moments and the covariance of $X^{n+1} - X^n$. In order for the binomial model to converge to the stochastic process described by (3.2) it must hold that

$$\mathbb{E}[X^{n+1} - X^n] = (0, 0) \tag{3.4}$$

and

$$\mathrm{Var}[X^{n+1} - X^n] = (\sigma_1^2 \Delta t, \sigma_2^2 \Delta t). \tag{3.5}$$

This gives the equations

$$\mathbb{P}_{uu} h_1 + \mathbb{P}_{ud} h_1 - \mathbb{P}_{du} h_1 - \mathbb{P}_{dd} h_1 = 0, \tag{3.6a}$$
$$\mathbb{P}_{uu} h_2 - \mathbb{P}_{ud} h_2 + \mathbb{P}_{du} h_2 - \mathbb{P}_{dd} h_2 = 0, \tag{3.6b}$$

and

$$\mathbb{P}_{uu} h_1^2 + \mathbb{P}_{ud} h_1^2 + \mathbb{P}_{du} h_1^2 + \mathbb{P}_{dd} h_1^2 = \sigma_1^2 \Delta t, \tag{3.7a}$$
$$\mathbb{P}_{uu} h_2^2 + \mathbb{P}_{ud} h_2^2 + \mathbb{P}_{du} h_2^2 + \mathbb{P}_{dd} h_2^2 = \sigma_2^2 \Delta t. \tag{3.7b}$$

Moreover, it follows from (2.23) that

$$\mathbb{P}_{uu} h_1 h_2 - \mathbb{P}_{ud} h_1 h_2 - \mathbb{P}_{du} h_1 h_2 + \mathbb{P}_{dd} h_1 h_2 = \rho \sigma_1 \sigma_2 \Delta t. \tag{3.8}$$

The probabilities for all of the four branches must obviously also sum to 1, i.e.,

$$\mathbb{P}_{uu} + \mathbb{P}_{ud} + \mathbb{P}_{du} + \mathbb{P}_{dd} = 1 \tag{3.9}$$

must hold. Having six equations and six unknowns, it is easy to solve (3.6)-(3.9) with respect to the unknown parameters and obtain

$$\mathbb{P}_{uu} = \mathbb{P}_{dd} = \frac{1}{4}(1+\rho), \quad \mathbb{P}_{ud} = \mathbb{P}_{du} = \frac{1}{4}(1-\rho), \tag{3.10a}$$
$$h_1 = \sigma_1 \sqrt{\Delta t}, \quad h_2 = \sigma_2 \sqrt{\Delta t}. \tag{3.10b}$$

This choice of the parameters makes the binomial model converge to the continuous model of the stock prices given by Representation 2. A formal proof of the convergence is presented in [7].

Now it is possible to compute the value of an option recursively by using the fact that the value of an option is equal to its payoff at the time of maturity. Let $V_Y(X(t_n), t_n)$ be the value at time t_n and let $V_Y^{uu}(X(t_{n+1}), t_{n+1})$, $V_Y^{ud}(X(t_{n+1}), t_{n+1})$, $V_Y^{du}(X(t_{n+1}), t_{n+1})$, and $V_Y^{dd}(X(t_{n+1}), t_{n+1})$ denote the values of the derivative at time t_{n+1} given that the prices of the underlying assets both go up (uu), the first one goes up and the second one goes down (ud), the first one goes down and the second goes up (du), and both go down (dd), respectively, at time t_{n+1}. It holds that

$$V_Y(X(T), T) = Y(X(T)) \tag{3.11}$$

and

$$\begin{aligned} V_Y(X(t_n), t_n) = e^{-r\Delta t}\Big(& V_Y^{uu}(X(t_{n+1}), t_{n+1}) \mathbb{P}_{uu} + V_Y^{ud}(X(t_{n+1}), t_{n+1}) \mathbb{P}_{ud} \\ & + V_Y^{du}(X(t_{n+1}), t_{n+1}) \mathbb{P}_{du} + V_Y^{dd}(X(t_{n+1}), t_{n+1}) \mathbb{P}_{dd} \Big). \end{aligned} \tag{3.12}$$

This is the standard (uniform) method to compute the binomial price of a European two-asset option. It is possible to extend this model to more than two assets in a straightforward manner. In the case of d underlying assets there will be 2^d possible directions at each new step and the probabilities and the increments can be computed in a similar way as above by imposing symmetry on the probabilities [6].

3.2 Refinement method for two underlying assets

There are ways to improve the uniform binomial method by using certain refinement techniques. In [8], an adaptive mesh model for the single asset binomial mode was introduced which was later on adapted to the multi-asset binomial model in [6]. These refinement methods have shown to give a significant improvement in computational speed for some options.

The algorithm introduced in [6] imposes a finer mesh in the interval between $T-\Delta t$ and T for nodes in the binomial model which can expire both in the money *and* out of the money in the finer mesh. In the case of level one refinement, asset prices corresponding to nodes in the refinement region will take four steps instead of one step in the interval between $T - \Delta t$ and T. Moreover, considering the transformed asset prices given by (3.1), it is assumed that the increments in each step are $h_1/2$ and $h_2/2$ for the first respectively second asset. This means that the prices of the underlying assets in the refinement region can change by maximum $2h_1$ respectively $2h_2$ in the interval between $T - \Delta t$ and T.

In order to illustrate this method, consider a European two-asset correlation call option with underlying asset prices $S(t) = (S_1(t), S_2(t))$ and payoff (2.37). That is,

$$Y(S(T)) = \begin{cases} \max\{S_2(T) - K_2, 0\} & \text{if } S_1(T) > K_1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

Following the notation in the previous section, define $i_{\min}$, $i_{\max}$, $j_{\min}$, and $j_{\max}$ for level one refinement as

$$i_{\min} = \min_i \left\{ (i,j) \in \{(1,\ldots,N)^2\} : X^{N-1}_{2,(i,j)} - 2h_2 \le \log(K_2) - \left(r - \frac{\sigma_2^2}{2}\right)T \right\}, \tag{3.14a}$$

$$i_{\max} = \max_i \left\{ (i,j) \in \{(1,\ldots,N)^2\} : X^{N-1}_{2,(i,j)} + 2h_2 > \log(K_2) - \left(r - \frac{\sigma_2^2}{2}\right)T \right\}, \tag{3.14b}$$

$$j_{\min} = \min_j \left\{ (i,j) \in \{(1,\ldots,N)^2\} : X^{N-1}_{1,(i,j)} + 2h_1 > \log(K_1) - \left(r - \frac{\sigma_1^2}{2}\right)T \right\}, \tag{3.14c}$$

$$j_{\max} = \max_j \left\{ (i,j) \in \{(1,\ldots,N)^2\} : X^{N-1}_{1,(i,j)} - 2h_1 \le \log(K_1) - \left(r - \frac{\sigma_1^2}{2}\right)T \right\}, \tag{3.14d}$$

where (3.1) has been used to find the right hand side of the inequalities in (3.14). The refinement region for the two-asset correlation option consists of all coarse nodes at time t_{N-1} with indices (i,j) that satisfy

$$(i_{\min} \le i \le i_{\max} \wedge j \ge j_{\min}) \vee (i \le i_{\max} \wedge j_{\min} \le j \le j_{\max}). \tag{3.15}$$

This refinement region will be called A and it usually has the shape of an L. An example of such a region is shown i Figure 3.3.

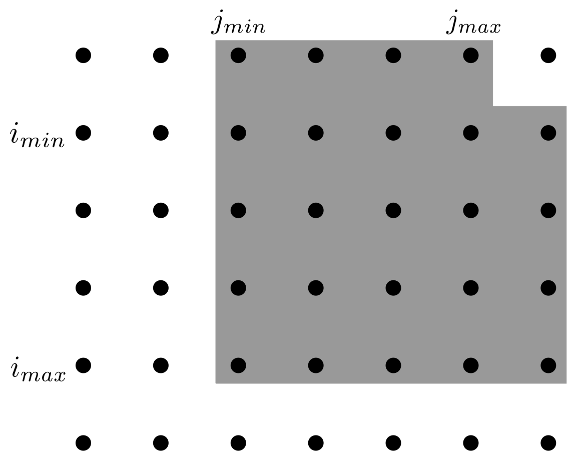


Figure 3.3: An example of refinement region A. Nodes in the shaded area should be assigned a finer mesh.

After the refinement region has been specified, the finer mesh is imposed at each one of the nodes in the refinement region. This is illustrated in Figure 3.4 for one node in the refinement region in the case of level one refinement.

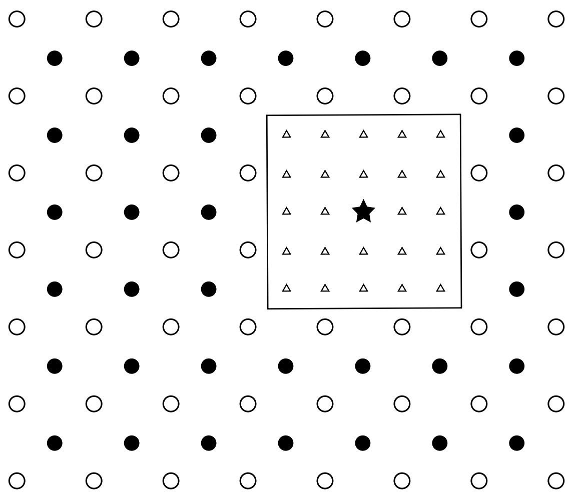


Figure 3.4: Finer mesh used for level one refinement marked by empty triangles corresponding to a node marked by a filled star. Filled circles correspond to other coarse nodes in the binomial model at time t_{N-1} and empty circles correspond to possible new states at time t_N without refinement. Note that some of the coarse nodes overlap with the finer mesh and thus have been omitted.

The option price can now be computed similar to the way it was done in the uniform binomial method, i.e., by equations (3.11) and (3.12). The option prices at time t_{N-1} in the refinement region are computed by discounting the expected payoff in the finer mesh whereas option prices outside the refinement region are computed by discounting the expected payoff just like in the uniform method. After the option prices in each node at time t_{N-1} have been computed, the computations become exactly the same as in the uniform case.

It is possible to use other level of refinements than level one which has been illustrated above. Let α be any positive integer, then level α refinement is obtained analogous to level one refinement but by imposing a finer mesh with time steps $\Delta t/4^\alpha$ and price steps of size $h_1/2^\alpha$ and $h_1/2^\alpha$. Thus $i_{\min}$, $i_{\max}$, $j_{\min}$, and $j_{\max}$ in (3.14) are determined by changing $2h_1$ and $2h_2$ to $2^\alpha h_1$ respectively $2^\alpha h_2$ in each of the inequalities. The conditions in refinement region A given by (3.15) remains the same and the option price is computed in the same way as for the level one case.

The usage of refinement region A can be argued to give better estimates of the option price than the uniform method when an average estimate is considered for some large enough interval of N. By imposing a finer mesh at some nodes, the estimated option price should become more accurate in those positions of the matrix at time t_{N-1}. Thus even the initial option price will become more precise since it is calculated in a recursive way. However, the convergence of the binomial model tends to oscillate [9]. Hence it is not certain that the usage of refinement region A will give a better estimate for any fixed N.

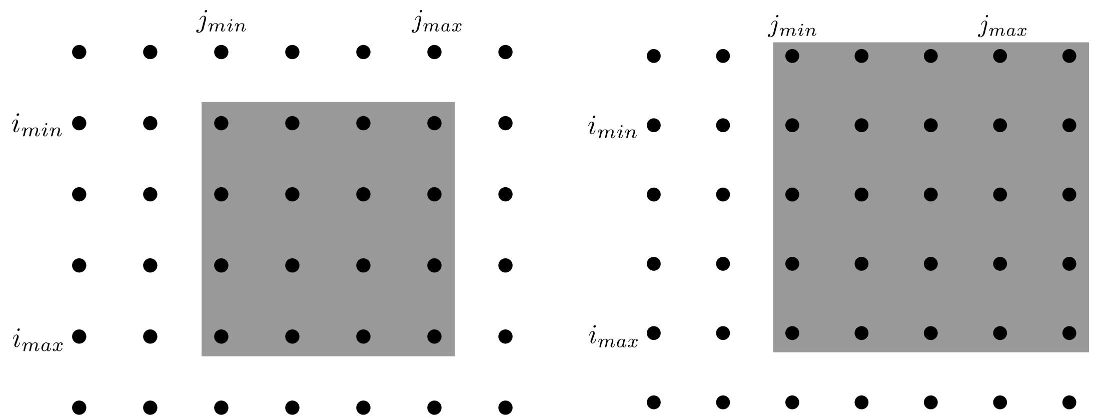
It seems possible that other refinement regions than the one give by (3.15) could be used to obtain improvements in accuracy. Of course it is never interesting to impose a finer mesh on a node that can never expire in the money, but it might still be beneficial to use a smaller or a larger refinement area than in (3.15). Let $i_{\min}$, $i_{\max}$, $j_{\min}$, and $j_{\max}$ be defined as in (3.14) and consider the set of indices (i, j) that satisfy

$$i_{\min} \le i \le i_{\max} \wedge j_{\min} \le j \le j_{\max}, \tag{3.16}$$

and

$$i \le i_{\max} \wedge j \ge j_{\min}. \tag{3.17}$$

The refinement regions given by (3.16) and (3.17) will possibly be smaller respectively larger than the one given by (3.15), and they will be referred to as refinement region B respectively C. Two examples of the two new regions are shown in Figure 3.5.



(a) Refinement region B. **(b)** Refinement region C.

Figure 3.5: Examples of refinement regions B and C. Nodes in the shaded area should be assigned a finer mesh.

The motivation behind introducing refinement region B is that it might be the case that some of the nodes in the top and right edges of refinement region A turn out to have a small contribution to the initial option price. Thus it could be more efficient to impose a finer mesh for a smaller number of nodes in the matrix. In opposite to this reasoning, one could argue that all nodes at t_{N-1} that can expire in the money in the finer mesh should be included in the refinement region. This is the logic behind refinement region C.

3.3 Adaption to American options

It is easy to adapt the binomial pricing model to price American options. The recurrence formula (3.12) only has to be slightly changed in order to take into consideration that an American option can be exercised at any time prior to maturity. Let $\widehat{V}_Y(X(t_n), t_n)$ denote the fair price of an American option whose European counterpart is $V_Y(X(t_n), t_n)$. It holds that

$$\widehat{V}_Y(X(T), T) = V_Y(X(T), T) = Y(X(T)), \tag{3.18}$$

and the recurrence formula for an American option becomes

$$\begin{aligned}\widehat{V}_Y(X(t_n), t_n) = e^{-r\Delta t}\max\Big(&\widehat{V}_Y^{uu}(X(t_{n+1}), t_{n+1})\mathbb{P}_{uu} + \widehat{V}_Y^{ud}(X(t_{n+1}), t_{n+1})\mathbb{P}_{ud} \\ &+\widehat{V}_Y^{du}(X(t_{n+1}), t_{n+1})\mathbb{P}_{du} + \widehat{V}_Y^{dd}(X(t_{n+1}), t_{n+1})\mathbb{P}_{dd}, Y(X(t_n))\Big),\end{aligned} \tag{3.19}$$

where $Y(X(t_n))$ denotes the so called intrinsic value of the American option at time t_n. By this recurrence formula it should be clear that $\widehat{V}_Y(X(t_n), t_n) \geq V_Y(X(t_n), t_n)$ and $\widehat{V}_Y(X(t_n), t_n) \geq Y(X(t_n))$. This makes intuitively sense since otherwise the market would not be free of arbitrage.

It is possible to apply the same refinement methods as discussed in the previous section but with the American recurrence formula in the refinement region. The finer mesh is found by using the same conditions as in the previous section.

An important concept concerning American options is the so called optimal exercise boundary. This boundary is the interface between the continuation region and the stopping region [4]. The continuation region is where it is not optimal to exercise the American option and it consists of stock prices that satisfy

$$\widehat{V}_Y(X(t_n), t_n) > Y(X(t_n)), \quad t_n < T. \tag{3.20}$$

The stopping region is where it is optimal to exercises the American option and it consists of stock prices that satisfy

$$\widehat{V}_Y(X(t_n), t_n) = Y(X(t_n)), \quad t_n < T. \tag{3.21}$$

3.4 The trinomial model

The trinomial model is an extension of the binomial model. In the trinomial model, each stock price can move in three different directions at each time step. Just like the binomial model, the trinomial model can be used to approximate the continuous model of the stock prices [10]. However, some authors have written that the trinomial model can also be considered a complete and arbitrage-free market model when there are two risky assets [11]. We take a closer look whether or not this holds by considering two stocks described by Representation 1, i.e.,

$$S_1(t) = S_1(0)e^{\alpha_1 t+\sigma_{11}W_1(t)+\sigma_{12}W_2(t)}, \tag{3.22}$$

$$S_2(t) = S_2(0)e^{\alpha_2 t+\sigma_{21}W_1(t)+\sigma_{22}W_2(t)}, \tag{3.23}$$

where

$$\alpha_1 = \mu_1 - \frac{\sigma_{11}^2+\sigma_{12}^2}{2}, \tag{3.24}$$

$$\alpha_2 = \mu_2 - \frac{\sigma_{21}^2+\sigma_{22}^2}{2}. \tag{3.25}$$

In order to implement the trinomial model it is necessary to introduce discrete versions of (3.22) and (3.23). For this purpose, let $\{X_i^{(1)}\}_{i\in\mathbb{N}}$ and $\{X_i^{(2)}\}_{i\in\mathbb{N}}$ be two independent and identically distributed (i.i.d.) stochastic processes satisfying

$$X_i^{(j)} = \begin{cases} 1 & \text{with probability } p_u \\ 0 & \text{with probability } p_m \\ -1 & \text{with probability } p_d \end{cases} \tag{3.26}$$

for $j = 1, 2$. The physical probabilities p_u, p_m, p_d are defined such that

$$\mathbb{E}\left[X_i^{(j)}\right] = 0 \implies p_u = p_d = p, \tag{3.27}$$

which means that

$$\mathrm{Var}\left[X_i^{(j)}\right] = \mathbb{E}\left[\left(X_i^{(j)}\right)^2\right] = p_u + p_d = 2p \tag{3.28}$$

for $1 = 1, \dots, N$ and $j = 1, 2$. Let $t_0 = 0 < t_1 < \dots < t_N = t$ and take $t_{i+1} - t_i = h$ with $N = t/h$. Define

$$\widetilde{S}_1(t_i) = \widetilde{S}_1(t_{i-1})e^{\widetilde{\alpha}_1+\widetilde{\sigma}_{11}X_i^{(1)}+\widetilde{\sigma}_{12}X_i^{(2)}}, \tag{3.29}$$

$$\widetilde{S}_2(t_i) = \widetilde{S}_2(t_{i-1})e^{\widetilde{\alpha}_2+\widetilde{\sigma}_{21}X_1^{(1)}+\widetilde{\sigma}_{22}X_i^{(2)}}, \tag{3.30}$$

which implies that

$$\widetilde{S}_1(t) = \widetilde{S}_1(0)e^{\widetilde{\alpha}_1 N+\widetilde{\sigma}_{11}M_N^{(1)}+\widetilde{\sigma}_{12}M_N^{(2)}}, \tag{3.31}$$

$$\widetilde{S}_2(t) = \widetilde{S}_2(0)e^{\widetilde{\alpha}_2 N+\widetilde{\sigma}_{21}M_N^{(1)}+\widetilde{\sigma}_{22}M_N^{(2)}}, \tag{3.32}$$

where

$$M_N^{(j)} = \sum_{i=0}^{N} X_i^{(j)}. \tag{3.33}$$

It is shown in Appendix A that the parameters $\widetilde{\alpha}_1$, $\widetilde{\alpha}_2$, $\widetilde{\sigma}_{11}$, $\widetilde{\sigma}_{12}$, $\widetilde{\sigma}_{21}$, $\widetilde{\sigma}_{22}$ must satisfy the following equations in order for (3.31) and (3.32) to converge to (3.22) respectively (3.23) when N goes to infinity:

$$\widetilde{\alpha}_1 = h\alpha_1, \quad \widetilde{\alpha}_2 = h\alpha_2, \tag{3.34}$$

$$\sigma_{11}^2 + \sigma_{12}^2 = \frac{2p}{h}(\widetilde{\sigma}_{11}^2 + \widetilde{\sigma}_{12}^2), \tag{3.35}$$

$$\sigma_{21}^2 + \sigma_{22}^2 = \frac{2p}{h}(\widetilde{\sigma}_{21}^2 + \widetilde{\sigma}_{22}^2), \tag{3.36}$$

$$\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22} = \frac{2p}{h}(\widetilde{\sigma}_{11}\widetilde{\sigma}_{21} + \widetilde{\sigma}_{12}\widetilde{\sigma}_{22}). \tag{3.37}$$

By imposing the martingale property, it is shown in Appendix A that there might not exist a risk-neutral probability measure for the trinomial model. This means that the trinomial model is not arbitrage-free and it is not a complete market model. This result is different from what some authors have written [11]. Even though the trinomial model is not a valid market model, it could still be used to approximate the continuous model if the parameters are specified correctly [10]. However, we use the binomial model for this purpose since it is easier to implement.

3.5 Results of the binomial pricing model

In this section, the uniform binomial pricing model is compared to the binomial pricing models with refinement regions A, B and C. Both options of European and American types are considered. In addition to computations of the option price, the dependence of parameters such as the number of time steps, correlation, and volatilities is studied.

The Python code for the numerical computations in this section is presented in Appendices C and D. Below, and throughout the rest of the thesis, the relative error is calculated as the absolute difference between the estimated price and the exact price divided by the exact price. Also, all computations are carried out on a computer with 3.40 GHz Intel Core i5 processor and 8 GB of RAM.

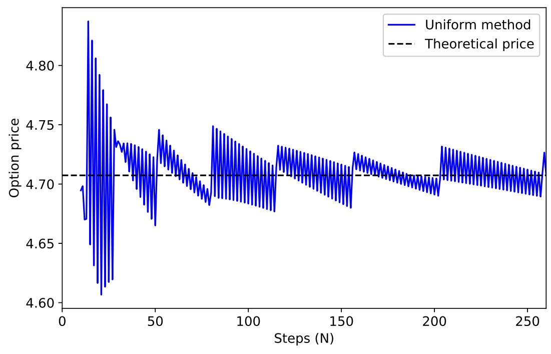
3.5.1 European two-asset correlation call option

Consider a European two-asset correlation call option with payoff as in (2.37), i.e.,

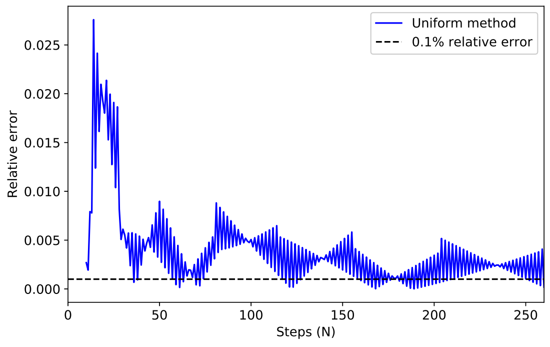
$$Y(S(T)) = \begin{cases} \max\{S_2(T) - K_2, 0\} & \text{if } S_1(T) > K_1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.38}$$

Assume the initial stock prices are $S_1(0) = 52$ and $S_2(0) = 65$, and let the strike prices be $K_1 = 50$ respectively $K_2 = 70$ at time of maturity $T = 0.50$. Moreover, let the correlation be $\rho = 0.75$ and let the risk-free interest rate be $r = 0.10$. If the volatilities of the stocks are $\sigma_1 = 0.20$ respectively $\sigma_2 = 0.30$ and the underlying assets pay no dividends, then it can be shown that the theoretical option price is approximately 4.7073 [3].

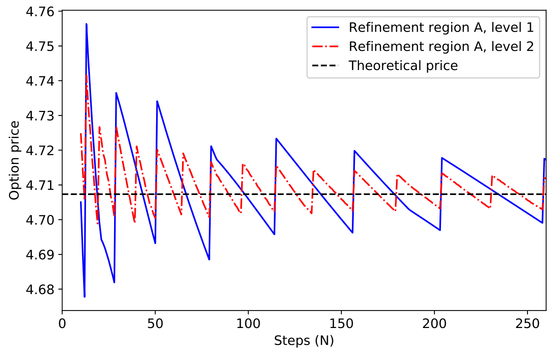
Figure 3.6 shows how the option prices converge to the theoretical price in the binomial pricing model when using the uniform method and three different refinement regions (A, B, and C) with two different levels of refinement.



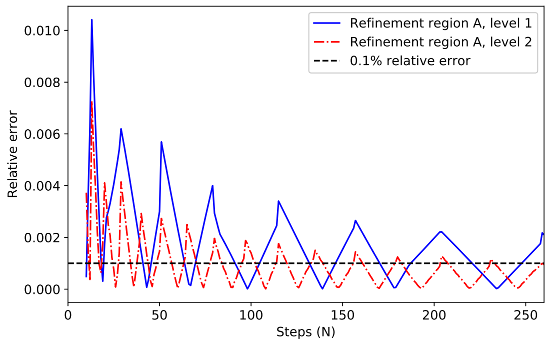
(a) Uniform method.



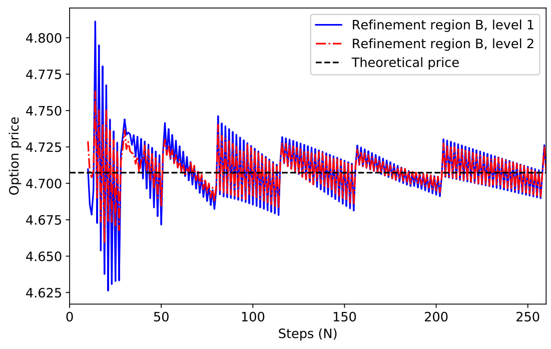
(b) Uniform method.



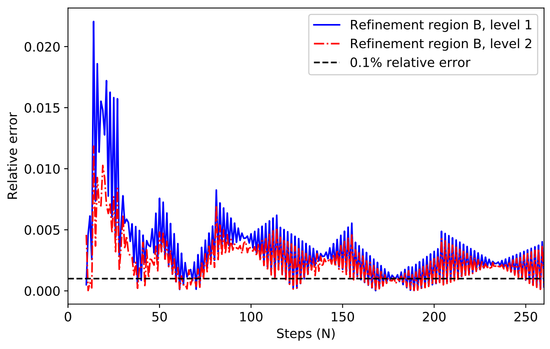
(c) Refinement region A.



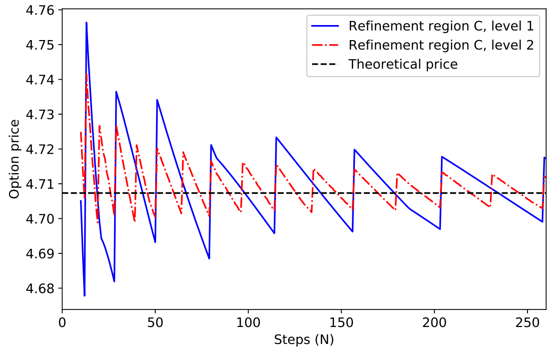
(d) Refinement region A.



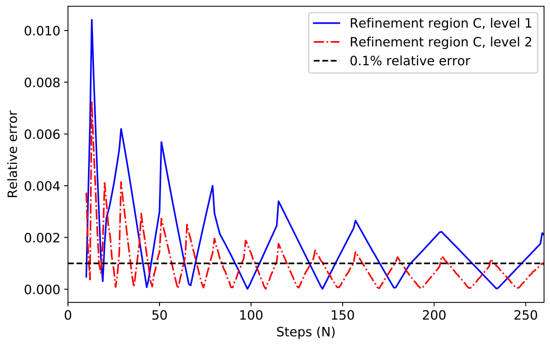
(e) Refinement region B.



(f) Refinement region B.



(g) Refinement region C.



(h) Refinement region C.

Figure 3.6: Option prices (left) and relative errors (right) for the European two-asset correlation call option computed with the binomial model. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

In a similar way, the computational time for the uniform method and the different refinement regions are shown in Figure 3.7.

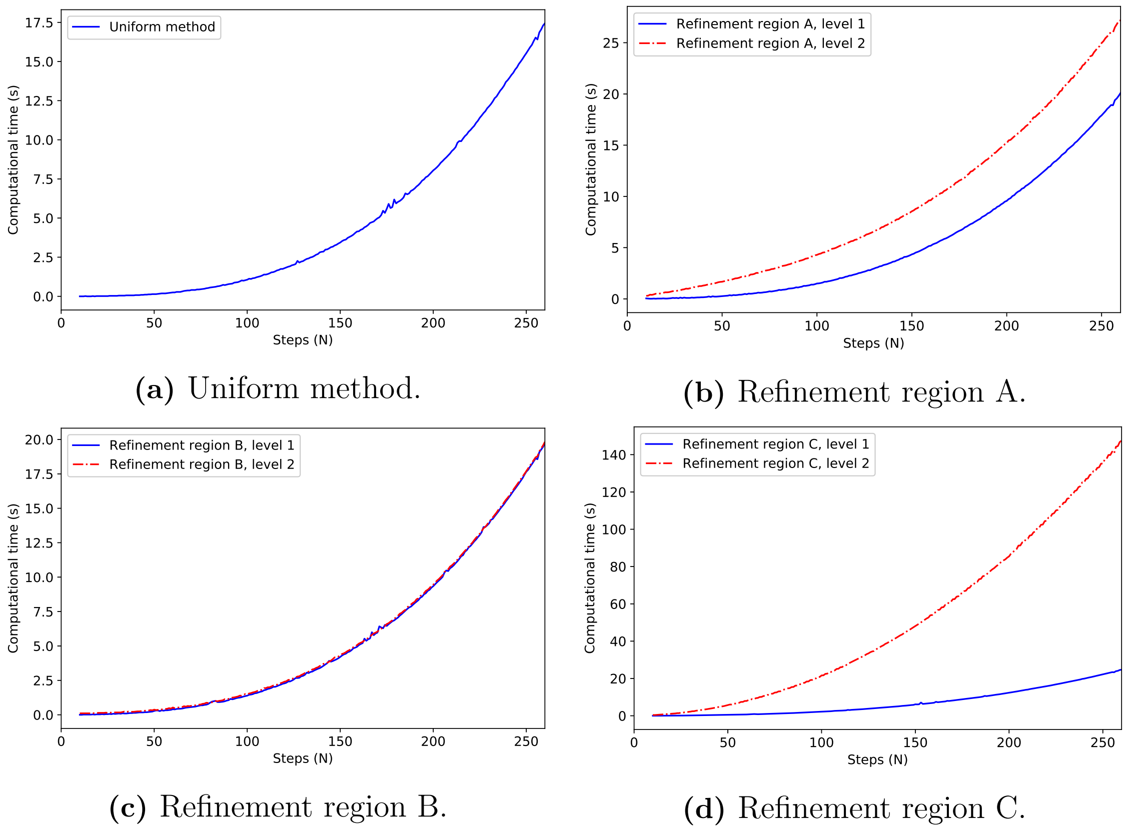


Figure 3.7: Computational times when using the binomial model for pricing the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

Observing the converge rate with respect to the number of steps N is not enough to conclude which refinement region and method that is the best one. Thus, from the previous figures the number of steps N in the binomial model is extracted from when the computational time is about 15 seconds. A comparison between the option prices for these times and an average for the 50 last N up to this computational time is shown in Table 3.1.

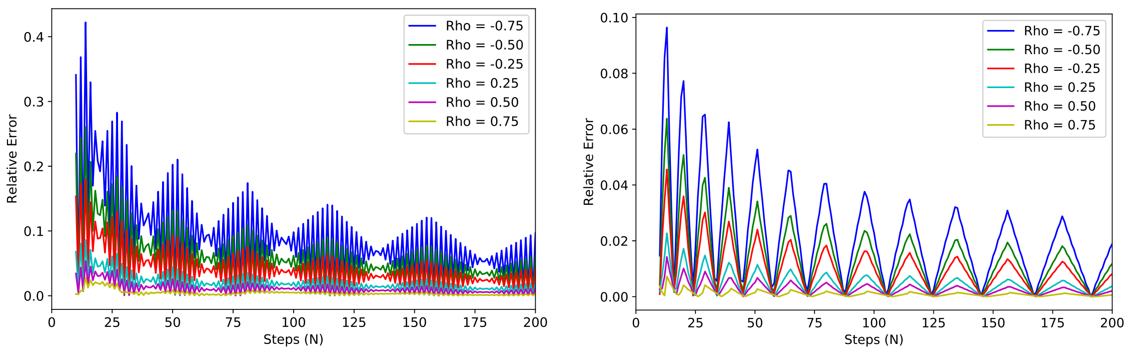
Table 3.1: Relative errors for the price of a European two-asset correlation call option computed with the binomial model when the computational time is shorter than 15 seconds. Results are presented for the final N satisfying the time limit as well as for the 50 steps prior to the final step. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$, and $\sigma_2 = 0.30$.

Method	Final N	Estimated price for final N	Mean price of 50 last N	Standard deviation of 50 last N	Error of final N	Mean error of 50 last N
Uniform	250	4.7122	4.7095	0.01313	0.001034	0.002523
Region A, Level 1	237	4.7064	4.7082	0.006650	0.0002020	0.001250
Region A, Level 2	201	4.7040	4.7080	0.003343	0.0007128	0.0006093
Region B, Level 1	237	4.6956	4.7084	0.01206	0.002481	0.002181
Region B, Level 2	236	4.7166	4.7081	0.01016	0.001972	0.001818
Region C, Level 1	214	4.7144	4.7073	0.006721	0.001498	0.001261
Region C, Level 2	83	4.7128	4.7099	0.006098	0.001163	0.001177

Higher levels of refinement than two could be used to give possibly even better approximations. However, the computational time increases drastically when increasing the level of refinement, e.g., level 3 refinement in region A takes more than 25 seconds already at $N = 10$.

Table 3.1 suggests that refinement region A with level 2 refinement is the best choice for this kind of option since it has the lowest average error and standard deviation. From here on, only the uniform method and refinement region A with refinement level 2 is used.

To further study the two-asset correlation option, it may be of interest to determine how the relative error depends on the correlation ρ between the two underlying assets. Using the same parameters as above but with six different values of the correlation, Figure 3.8 shows that the relative error seems to decrease when the correlation increases.



(a) Uniform method. **(b)** Refinement region A, level 2.

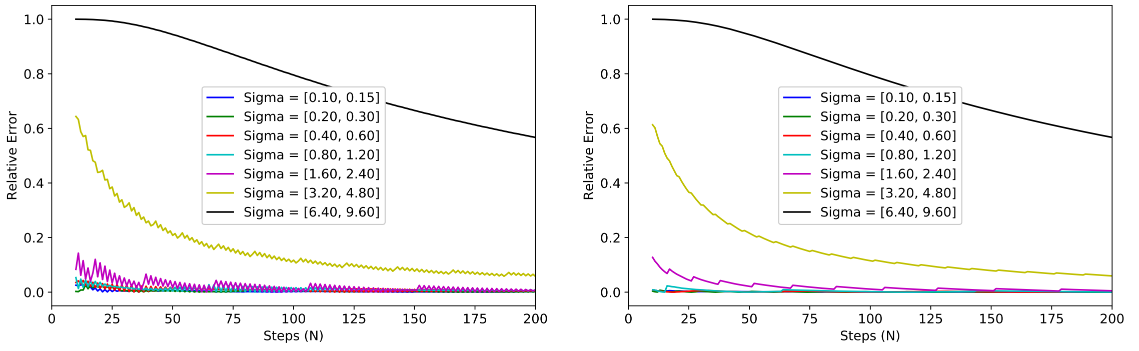
Figure 3.8: Dependence on the correlation ρ for the price of a European two-asset correlation call option computed with the binomial model. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\sigma_1 = 0.20$, and $\sigma_2 = 0.30$.

Fixing the number of steps N which yields a computational time of 15 seconds, the estimated price and relative errors for each value of the correlation are collected in Table 3.2 below.

Table 3.2: Option prices when using the binomial model for a European two-asset correlation call option with different correlations ρ, and for N such that the computational time is about 15 seconds. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\sigma_1 = 0.20$, and $\sigma_2 = 0.30$.

Correlation ρ	Method	N	Estimated price	Exact price	Relative error
-0.75	Uniform	250	1.3159	1.2981	0.01370
	Region A, Level 2	201	1.2703		0.02142
-0.50	Uniform	250	2.1106	2.0914	0.009162
	Region A, Level 2	201	2.0633		0.01343
-0.25	Uniform	250	2.8134	2.7948	0.006629
	Region A, Level 2	201	2.7689		0.009281
0.25	Uniform	250	3.9734	3.9599	0.0003394
	Region A, Level 2	201	3.9431		0.004244
0.50	Uniform	250	4.4105	4.4010	0.002149
	Region A, Level 2	201	4.3906		0.002370
0.75	Uniform	250	4.7122	4.7073	0.001034
	Region A, Level 2	201	4.7040		0.0007128

It is also of interest to study the dependence on the volatilities. Observations are made for seven different volatilities σ_1 and σ_2 in Figure 3.9 and the results are summarised in Table 3.3.



(a) Uniform method. **(b)** Refinement region A, level 2.

Figure 3.9: Dependence on the volatilities σ_1 and σ_2 for the price of a European two-asset correlation call option computed with the binomial model. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$.

Table 3.3: Option prices when using the binomial model for a European two-asset correlation call option with different volatilities σ_1, σ_2, and for N such that the computational time is about 15 seconds. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$ and $\rho = 0.75$.

Volatilities σ_1, σ_2	Method	N	Estimated price	Exact price	Relative error
0.10, 0.15	Uniform	250	2.0588	2.0584	0.0002045
	Region A, Level 2	201	2.0589		0.0002427
0.20, 0.30	Uniform	250	4.7122	4.7073	0.001034
	Region A, Level 2	201	4.7040		0.0007128
0.40, 0.60	Uniform	250	9.7627	9.7424	0.002086
	Region A, Level 2	201	9.7204		0.002258
0.80, 1.20	Uniform	250	19.4955	19.2751	0.01143
	Region A, Level 2	201	19.2642		0.0005649
1.60, 2.40	Uniform	250	36.1381	36.0371	0.002800
	Region A, Level 2	201	36.8665		0.004735
3.20, 4.80	Uniform	250	54.0660	56.7448	0.04721
	Region A, Level 2	201	53.3977		0.05899
6.40, 9.60	Uniform	250	32.9481	64.8269	0.4918
	Region A, Level 2	201	28.1844		0.5652

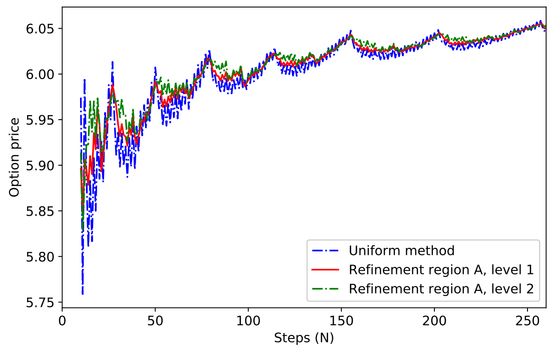
3.5.2 American two-asset correlation put option

Consider an American two-asset correlation put option that can be exercised at any time prior to maturity T with intrinsic value

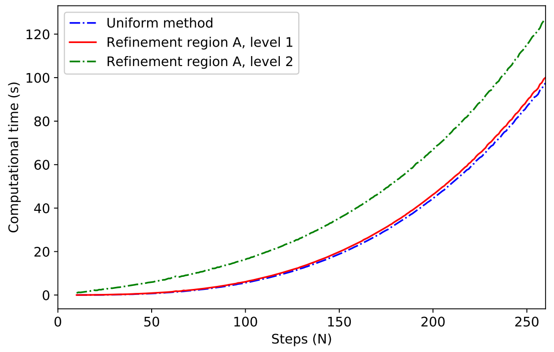
$$Y(S(t_n)) = \begin{cases} \max\{K_2 - S_2(t_n), 0\} & \text{if } S_1(t_n) < K_1, \\ 0 & \text{otherwise,} \end{cases} \tag{3.39}$$

where $0 < t_n \leq T$ are discrete time instances in the binomial model. There exists no closed-form solution to the price of this option. This means that it is not possible to use any exact reference price when pricing the option. However, its European counterpart can be priced exactly, as described in the section about examples of multi-asset options. Recall that an American version of an option should always be at least as expensive as the European version of it. This means that it is possible to get an idea of how well the pricing method of the American option works by checking this condition.

Figure 3.10 shows how the price of the American two-asset correlation put option varies depending on the number of steps N in the binomial model. Both the uniform binomial method and the binomial method with refinement region A is included in the figure. Refinement region A was chosen since results from the previous section indicated that it was a good choice for refinement. The same standard parameters as for the European two-asset correlation call option in the previous section are used. Thus the initial stock prices are $S_1(0) = 52$ and $S_2(0) = 65$, the strike prices are $K_1 = 50$ and $K_2 = 70$, the time of maturity is $T = 0.50$, the risk-free interest rate is $r = 0.10$, the correlation is $\rho = 0.75$, and the volatilities are $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.



(a) Option price.



(b) Computational time.

Figure 3.10: Option prices and computational times of the American two-asset correlation put option computed with the binomial model. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

The price in the binomial model for the American two-asset correlation put option seems to stabilise somewhere around 6.05 according to Figure 3.10. The exact price of the corresponding European option is approximately 3.9092. Hence the estimated price of the American option is higher than its European counterpart, as expected. How the American option price depends on the number of steps N is also summarised in Table 3.4.

Table 3.4: Estimated prices of an American two-asset correlation put option computed with the binomial model when the computational time is shorter than 15 seconds. Results are presented for the final N satisfying the time limit as well as for the 50 steps prior to the final step. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$, and $\sigma_2 = 0.30$.

Method	Final N	Estimated price for final N	Mean price of 50 last N	Standard deviation of 50 last N
Uniform	142	6.0160	6.0066	0.01119
Region A, Level 1	141	6.0187	6.0098	0.008987
Region A, Level 2	129	6.0189	6.0092	0.008992

It is not clear how well the refinement methods work in the American case since there is no exact price to compare the results to. The refinement methods have slightly smaller standard deviations than the uniform method according to Table 3.4. However, it seems like the binomial price is not oscillating around a horizontal line as in the previous European case but rather around a curve that is slightly increasing. This means that a smaller standard deviation does not necessarily imply a better estimate.

It is also interesting to observe how the American option price depends on the correlation ρ and the volatilities σ_1 and σ_2. Such results are shown in Tables 3.5 and 3.6 where the uniform method has been used for the estimates.

Table 3.5: Uniform option prices using the binomial model for an American two-asset correlation put option with different correlations ρ, and for N such that the computational time is about 15 seconds (i.e., $N = 142$). Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\sigma_1 = 0.20$, and $\sigma_2 = 0.30$

Correlation ρ	Estimated American price	Exact European price
-0.75	1.5857	0.2857
-0.50	2.5844	0.7474
-0.25	3.3908	1.2691
0.25	4.7801	2.4543
0.50	5.4136	3.1338
0.75	6.0160	3.9093

Table 3.6: Uniform option prices using the binomial model for an American two-asset correlation put option with different volatilities σ_1, σ_2, and for N such that the computational time is about 15 seconds (i.e., $N = 142$). Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$ and $\rho = 0.75$.

Volatilities σ_1, σ_2	Estimated American price	Exact European price
0.10, 0.15	2.7426	1.1028
0.20, 0.30	6.0160	3.9093
0.40, 0.60	11.1486	8.9953
0.80, 1.20	21.1980	18.3718
1.60, 2.40	38.6029	34.5981
3.20, 4.80	59.6698	55.0437
6.40, 9.60	68.6964	65.7421

Observe in Tables 3.5 and 3.6 that the American option is always more expensive than the European version with the same parameters. As expected, there is also a clear correlation between the American and European prices.

Another interesting property of the American option is its optimal exercise surface. Figure 3.11 shows what the projection onto the S_1S_2-plane of this surface looks like for a few different time instances. It is optimal to exercise the American two-asset correlation put option at time t if both $S_1(t)$ and $S_2(t)$ are inside the region bounded by the optimal exercise curves.

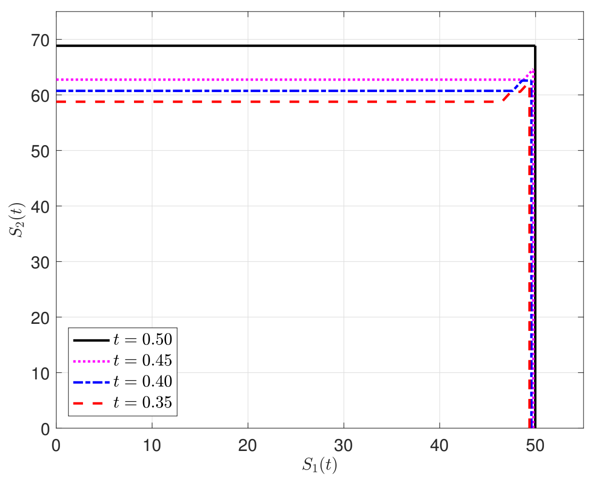


Figure 3.11: Projection of the optimal exercise surface for the American two-asset correlation put option computed with the binomial model. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

Figure 3.11 suggests that the right boundary corresponding to $S_1(t)$ remains almost unchanged for different values of t. However, the top boundary corresponding to $S_2(t)$ falls when t decreases. Moreover, the average vertical displacement is greater closer to the time of expiration. This observation is similar to what has be observed for the standard single asset American put option [12].

4 Monte Carlo Methods

The binomial tree used in the binomial pricing model gives a computational time that grows exponentially with the number of underlying assets. Thus when considering options with a larger number of underlying assets, another model is needed. The Monte Carlo method is a well-known approach in option pricing which is based on the idea to compute the average payoff for a large number of sample paths of the underlying assets.

4.1 Introduction to Monte Carlo simulations

Using the Monte Carlo method for estimating the fair price of the options, the same fundamental theory used in the previous method holds. Assume that the underlying assets are described by Representation 1. For a general non-American option with two underlying assets $S(t) = (S_1(t), S_2(t))$, the following algorithm is used for simulations in the Monte Carlo method:

1. *Create a partition* $0 = t_0 < t_1 < \cdots < t_N = T$ *of the interval* $[0,T]$.
2. *For a large number* n*, perform the following steps for each* $k = 1, \ldots, n$*:*
   (a) *Generate* $\widetilde{W}_i(t_j)$ *for* $i = 1, 2$ *and* $j = 1, \ldots, N$ *where* $\{\widetilde{W}_1(t)\}_{t\geq 0}$ *and* $\{\widetilde{W}_2(t)\}_{t\geq 0}$ *are independent Brownian motions.*
   (b) *Compute* $S_i(t_j)$ *as*

$$S_i(t_j) = S_i(0)\exp\left[\left(r - \frac{\sigma_{i1}^2 + \sigma_{i2}^2}{2}\right)t_j + \sigma_{i1}\widetilde{W}_1(t_j) + \sigma_{i2}\widetilde{W}_2(t_j)\right]. \quad (4.1)$$

   (c) *Compute the payoff* Y_k *at time* T.
   (d) *Compute the discounted payoff* V_k *as*

$$V_k = e^{-rT}Y_k. \quad (4.2)$$

3. *Compute the initial option price* $\widehat{V}_n$ *as*

$$\widehat{V}_n = (V_1 + \cdots + V_n)/n. \quad (4.3)$$

It is possible to simplify the algorithm for non-path-dependent options by not doing a partition of the interval $[0, T]$ and only generating the final value $\widetilde{W}_i(T)$ for $i = 1, 2$.

Furthermore, a confidence interval of the estimation is of great interest. Defining $\widehat{V}_n$ as in the algorithm above, then $\widehat{V}_n$ is an unbiased estimator for $n \geq 1$. Thus, a confidence interval is given by

$$\widehat{V}_n \pm z_{\delta/2}\frac{s_C}{\sqrt{n}} \tag{4.4}$$

where z_δ denotes the $1-\delta$ quantile of the standard normal distribution and s_C is the sample standard deviation defined by

$$s_C = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(V_i - \widehat{V}_n)^2}. \tag{4.5}$$

4.2 Variance reduction

One possible approach to increasing the precision of the estimator is to increase the number of samples, but this will heavily affect the computational time. Another approach is thus needed. Variance reduction techniques are used in order to increase the accuracy of an estimate without increasing the cost of simulation significantly.

4.2.1 Antithetic variates

Antithetic variates (AV) is a variance reduction technique where the variance is reduced by introducing negative dependence between pairs of replications. There are different ways to implement this method; one way is to use the simulated Brownian motions $\{\widetilde{W}_1(t)\}_{t\geq 0}$ and $\{\widetilde{W}_2(t)\}_{t\geq 0}$ above to define

$$\widetilde{W}_1'(t_j) = -\widetilde{W}_1(t_j), \quad j = 1,\dots,N, \tag{4.6}$$
$$\widetilde{W}_2'(t_j) = -\widetilde{W}_2(t_j), \quad j = 1,\dots,N. \tag{4.7}$$

Consider the i.i.d. pairs $(V_1, V_1'), (V_2, V_2'), \dots, (V_n, V_n')$ where V_k corresponds to the initial option price calculated from $\widetilde{W}_1(t_j)$ and $\widetilde{W}_2(t_j)$ for $j = 1,\dots,N$, whereas V_k' is the price obtained from $\widetilde{W}_1'(t_j)$ and $\widetilde{W}_2'(t_j)$ for $j = 1,\dots,N$. The antithetic variates estimator can be defined by [13]

$$\widehat{V}_{AV} = \frac{1}{n}\sum_{k=1}^{n}\left(\frac{V_k + V_k'}{2}\right). \tag{4.8}$$

The random variables V_k and V_k' will generally not be independent but they have the same distribution, so let V be a random variable with the common distribution. The central limit theorem can be applied to the independent observations

$$\left(\frac{V_1 + V_1'}{2}\right), \left(\frac{V_2 + V_2'}{2}\right), \dots, \left(\frac{V_n + V_n'}{2}\right) \tag{4.9}$$

and thus

$$\frac{\widehat{V}_{AV} - \mathbb{E}[V]}{\sigma_{AV}/\sqrt{n}} \to Z \quad \text{as } n \to \infty, \tag{4.10}$$

where Z is a standard normally distributed random variable and

$$\sigma_{AV}^2 = \mathrm{Var}\left[\frac{V_k + V_k'}{2}\right]. \tag{4.11}$$

The central limit theorem continues to hold even if the standard deviation σ_{AV} is replaced by the sample standard deviation s_{AV}, and thus an approximate confidence interval of $\widehat{V}_{AV}$ is given by

$$\widehat{V}_{AV} \pm z_{\delta/2}\frac{s_{AV}}{\sqrt{n}}, \tag{4.12}$$

where $z_{\delta/2}$ is the number such that the area to the right of it under the standard normal distribution function is equal to $\delta/2$.

The computational effort to compute $\widehat{V}_{AV}$ is approximately twice as high as for the regular Monte Carlo method without antithetic variates [13]. Therefore it is meaningful to use this method if

$$\mathrm{Var}\left[\widehat{V}_{AV}\right] < \mathrm{Var}\left[\frac{1}{2n}\sum_{k=1}^{2n} V_k\right]. \tag{4.13}$$

The inequality (4.13) holds if

$$\mathrm{Var}[V_k + V_k'] < 2\mathrm{Var}[V_k] \iff 2\mathrm{Var}[V_k] + 2\mathrm{Cov}[V_k, V_k'] < 2\mathrm{Var}[V_k], \tag{4.14}$$

i.e., if $\mathrm{Cov}[V_k, V_k'] < 0$. This condition will hold if negative dependence of the inputs implies a negative dependence of the initial option prices, which is typically the case.

4.2.2 Control variates

Another variance reduction technique known as the control variates (CV) method is based on the idea to use errors in estimates of a known quantity to reduce the error of an estimate of an unknown quantity. To illustrate this, suppose that $V_1, \dots, V_n$ are i.i.d. outputs from n replications of a Monte Carlo simulation and that the goal is to estimate $\mathbb{E}[V_k] \equiv \mathbb{E}[V]$, e.g., the discounted expected payoff of an option. For each replication, calculate another output U_k which has a known expectation $\mathbb{E}[U_k] \equiv \mathbb{E}[U]$. Suppose that the pairs (U_k, V_k), $k = 1, \dots, n$, are i.i.d. and define $V_k(b)$ for any fixed b as

$$V_k(b) = V_k - b(U_k - \mathbb{E}[U]). \tag{4.15}$$

The control variate estimator is given by the sample mean of all $V_k(b)$, i.e.,

$$\widehat{V}_{CV}(b) = \overline{V} - b(\overline{U} - \mathbb{E}[U]) = \frac{1}{n}\sum_{k=1}^{n}(V_k - b(U_k - \mathbb{E}[U])), \tag{4.16}$$

where it has been used that $\overline{U} = (U_1 + \dots + U_n)/n$ and $\overline{V} = (V_1 + \dots + V_n)/n$. Since $\overline{V}$ is an unbiased estimator of $\mathbb{E}[V]$, it holds that

$$\mathbb{E}[\widehat{V}_{CV}(b)] = \mathbb{E}[\overline{V} - b(\overline{U} - \mathbb{E}[U])] = \mathbb{E}[\overline{V}] = \mathbb{E}[V], \tag{4.17}$$

and thus (4.16) is an unbiased estimator of $\mathbb{E}[V]$. It is also easy to verify that (4.16) is a consistent estimator [13].

Assume that $\operatorname{Var}[U_k] \equiv \operatorname{Var}[U] = \sigma_U^2$ and $\operatorname{Var}[V_k] \equiv \operatorname{Var}[V] = \sigma_V^2$, and let ρ_{UV} be the correlation between U_k and V_k, then $V_k(b)$ has variance

$$\operatorname{Var}[V_k(b)] = \operatorname{Var}[V_k - b(U_k - E[U])] = \sigma_V^2 - 2b\sigma_U\sigma_V\rho_{UV} + b^2\sigma_U^2. \tag{4.18}$$

This means that $\widehat{V}_{CV}(b)$ has variance

$$\operatorname{Var}[\widehat{V}_{CV}(b)] = \frac{\sigma_V^2 - 2b\sigma_U\sigma_V\rho_{UV} + b^2\sigma_U^2}{n}. \tag{4.19}$$

Since $\operatorname{Var}[\overline{V}] = \sigma_V^2/n$, it follows from (4.19) that $\operatorname{Var}[\widehat{V}_{CV}(b)]$ is smaller than $\operatorname{Var}[\bar{V}]$ if $2b\sigma_U\sigma_V\rho_{UV} > b^2\sigma_U^2$. The choice of b that minimises (4.19) is given by

$$b^* = \frac{\sigma_V}{\sigma_U}\rho_{UV}, \tag{4.20}$$

which yields that the ratio between the variance of $\operatorname{Var}[\widehat{V}_{CV}(b^*)]$ and $\operatorname{Var}[\overline{V}]$ is

$$\frac{\operatorname{Var}[\overline{V} - b^*(\overline{U} - \mathbb{E}[U])]}{\operatorname{Var}[\overline{V}]} = 1 - \rho_{UV}^2. \tag{4.21}$$

Note that the relative advantage of using the optimal b in the control variate estimator is only dependent on the correlation coefficient ρ_{UV}. Moreover, it is of course desirable if ρ_{UV} is as high as possible.

In practise it is often difficult to find the optimal choice b^* since σ_Y and ρ_{UV} are often unknown. However, it is possible to use the population parameters to estimate b^* and thereby still obtain a control variate that can be very useful [13]. In that case, it follows from (4.20) that b^* can be estimated as

$$\widehat{b}_n = \frac{\sum_{k=1}^n (U_k - \overline{U})(V_k - \overline{V})}{\sum_{k=1}^n (U_k - \overline{U})^2}. \tag{4.22}$$

The estimate $\widehat{V}_{CV}(\widehat{b}_n)$ could be biased since it holds that

$$\mathbb{E}[\widehat{V}_{CV}(\widehat{b}_n)] - \mathbb{E}[V] = -\mathbb{E}[\widehat{b}_n(\overline{U} - \mathbb{E}[U])], \tag{4.23}$$

and the right hand side in (4.23) need not to be zero because $\widehat{b}_n$ and $\overline{U}$ are not independent. It is possible to eliminate this bias by dividing the replications into two parts and use the first part to estimate b^* and then use this estimate for the remaining replications. However, the biased given by (4.23) is of order $\mathcal{O}(1/n)$ whereas the the order of the standard error is $\mathcal{O}(1/\sqrt{n})$. This means that the bias is usually relatively small and it is not necessary to apply the elimination technique described above [13].

4.3 Control variate for the Asian option

Consider an Asian two-asset call option with payoff Y_c given by the continuous arithmetic average with equal weights, i.e.,

$$Y_c = \left(\frac{1}{T}\int_0^T \left(\frac{1}{2}S_1(t) + \frac{1}{2}S_2(t)\right) dt - K\right)^+. \tag{4.24}$$

By using a large N, this option can be approximated by the corresponding discrete arithmetic average, with payoff

$$Y_d = \left(\frac{1}{N+1} \sum_{j=0}^{N} \left(\frac{1}{2} S_1(t_j) + \frac{1}{2} S_2(t_j) \right) - K \right)^+ . \tag{4.25}$$

Assume that the stock prices follow geometric Brownian motions described by Representation 1 which in the risk-neutral world can be written as

$$S_1(t) = S_1(0) \exp \left(\left(r - \frac{\sigma_{11}^2 + \sigma_{12}^2}{2} \right) t + \sigma_{11} \widetilde{W}_1(t) + \sigma_{12} \widetilde{W}_2(t) \right), \tag{4.26}$$

$$S_2(t) = S_2(0) \exp \left(\left(r - \frac{\sigma_{21}^2 + \sigma_{22}^2}{2} \right) t + \sigma_{21} \widetilde{W}_1(t) + \sigma_{22} \widetilde{W}_2(t) \right). \tag{4.27}$$

It is easy to apply the standard Monte Carlo method to this setup and obtain an estimate of the initial price of an Asian option. Is is also straightforward to use the variance reduction technique based on antithetic variates. The usage of control variates requires more consideration. For the single-asset Asian option (2.46), Kemna and Vorst [14] suggested the usage of a geometric average Asian option as a control variate since this option can be priced by a closed formula. Inspired by their approach, we consider the continuous Asian two-asset geometric option corresponding to (4.24) with payoff Y_c' given by

$$Y_c' = \left(\exp \left(\frac{1}{T} \int_0^T \log \left(S_1(t)^{\frac{1}{2}} S_2(t)^{\frac{1}{2}} \right) dt \right) - K \right)^+ . \tag{4.28}$$

By using (4.26) and (4.27), we can rewrite this payoff as

$$\begin{aligned} Y_c' = &\left(\exp \left[\frac{1}{T} \int_0^T \frac{1}{2} \log \left\{ S_1(0) \exp \left(\left(r - \frac{\sigma_{11}^2 + \sigma_{12}^2}{2} \right) t + \sigma_{11} \widetilde{W}_1(t) + \sigma_{12} \widetilde{W}_2(t) \right) \right. \right. \right. \\ &\left. \left. \left. S_2(0) \exp \left(\left(r - \frac{\sigma_{21}^2 + \sigma_{22}^2}{2} \right) t + \sigma_{21} \widetilde{W}_1(t) + \sigma_{22} \widetilde{W}_2(t) \right) \right\} \right] dt - K \right)^+ \end{aligned} \tag{4.29}$$

$$\begin{aligned} = &\left(\sqrt{S_1(0) S_2(0)} \exp \left(\left(\frac{r}{2} - \frac{\sigma_{11}^2 + \sigma_{12}^2 + \sigma_{21}^2 + \sigma_{22}^2}{8} \right) T \right) \right. \\ &\left. \exp \left(\frac{\sigma_{11} + \sigma_{21}}{2T} \int_0^T \widetilde{W}_1(t) dt \right) \exp \left(\frac{\sigma_{12} + \sigma_{22}}{2T} \int_0^T \widetilde{W}_2(t) dt \right) - K \right)^+ . \end{aligned} \tag{4.30}$$

For any Brownian motion $\{W(t)\}_{t>0}$ it holds that

$$\int_0^T W(t) dt \sim \mathcal{N} \left(0, \frac{T^3}{3} \right), \tag{4.31}$$

and thus (4.30) is equal to

$$\left(\sqrt{S_1(0)S_2(0)}\exp\left(\left(\frac{r}{2}-\frac{\sigma_{11}^2+\sigma_{12}^2+\sigma_{21}^2+\sigma_{22}^2}{8}\right)T\right)\right.$$
$$\left.\exp\left(\frac{\sigma_{11}+\sigma_{21}}{2T}\sqrt{\frac{T^3}{3}}Z_1\right)\exp\left(\frac{\sigma_{12}+\sigma_{22}}{2T}\sqrt{\frac{T^3}{3}}Z_2\right)-K\right)^+, \tag{4.32}$$

where Z_1 and Z_2 are standard normal random variables. Since Z_1 and Z_2 are independent, it also follows that (4.32) is equal to

$$\left(\sqrt{S_1(0)S_2(0)}\exp\left\{\left(\frac{r}{2}-\frac{\sigma_{11}^2+\sigma_{12}^2+\sigma_{21}^2+\sigma_{22}^2}{8}\right)T\right.\right.$$
$$\left.\left.\frac{\sqrt{(\sigma_{11}+\sigma_{12})^2+(\sigma_{21}+\sigma_{22})^2}}{2\sqrt{3}}\sqrt{T}Z_3\right\}-K\right)^+, \tag{4.33}$$

where Z_3 is another standard normal random variable. Define $\overline{\sigma}$, and δ as

$$\overline{\sigma}=\frac{\sqrt{(\sigma_{11}+\sigma_{12})^2+(\sigma_{21}+\sigma_{22})^2}}{2\sqrt{3}}, \tag{4.34}$$

$$\delta=\frac{r}{2}-\frac{\bar{\sigma}^2}{2}+\frac{\sigma_{11}^2+\sigma_{12}^2+\sigma_{21}^2+\sigma_{22}^2}{8}. \tag{4.35}$$

Now (4.33) can be expressed as

$$\left(\sqrt{S_1(0)S_2(0)}\exp\left\{\left(r-\delta-\frac{\overline{\sigma}^2}{2}\right)T+\overline{\sigma}\widetilde{W}(T)\right\}-K\right)^+, \tag{4.36}$$

where $\widetilde{W}(T)$ is the value at time T of $\{\widetilde{W}(t)\}_{t>0}$ which is a Brownian motion under a risk-neutral probability measure. This means that it is possible to value the two-asset continuous geometric average Asian option by using the regular Black-Scholes formula for a single-asset European call with strike price K, maturity T, volatility $\overline{\sigma}$, initial value $\sqrt{S_1(0)S_2(0)}$, interest rate r, and continuous dividend yield δ. Hence it suitable to use the two-asset geometric average Asian option as a control variate for the two-asset arithmetic average Asian option.

The continuous two-asset geometric average Asian option (4.28) can be approximated by its discrete counterpart which is given by

$$Y_d'=\left(\left(\prod_{j=0}^{N}S_1(t_j)^{\frac{1}{2}}S_2(t_j)^{\frac{1}{2}}\right)^{\frac{1}{N+1}}-K\right)^+ \tag{4.37}$$

$$=\left(\exp\left(\frac{1}{N+1}\sum_{j=0}^{N}\log\left(S_1(t_j)^{\frac{1}{2}}S_2(t_j)^{\frac{1}{2}}\right)\right)-K\right)^+, \tag{4.38}$$

where it is assumed that N is a large number.

4.4 Results of the Monte Carlo method

This section shows results from Monte Carlo simulations of option prices. The European two-asset correlation call option is considered once more and also the Asian two-asset call option is priced. A comparison is made between the regular Monte Carlo method and different variance reduction techniques. Moreover, the number of replications, the correlations, and the volatilities are varied in order to study the impact on the price estimations. The Python code corresponding to this section is presented in Appendix E.

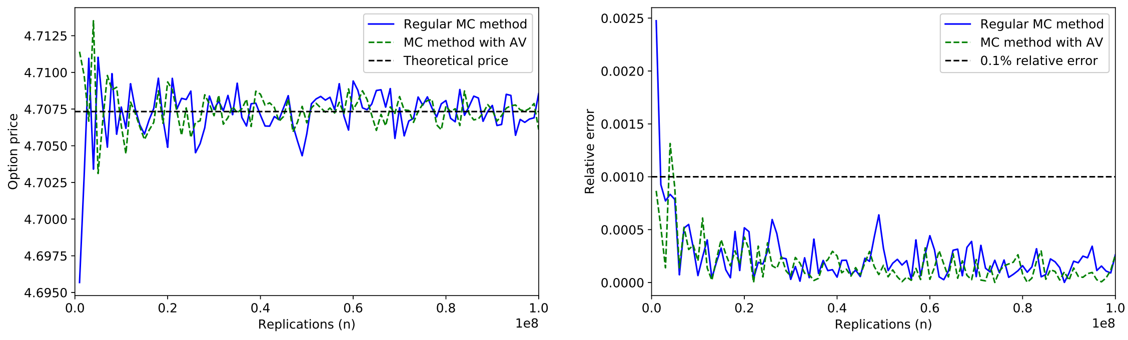
4.4.1 European two-asset correlation call option

To begin with, the regular Monte Carlo method and the Monte Carlo method with a variance reduction technique based on the antithetic variates (AV) is used to price the same European two-asset correlation call option as in the previous section. Thus let the initial stock prices be $S_1(0) = 52$ and $S_2(0) = 65$. Assume that the underlying assets pay no divided. Moreover, let the strike prices be $K_1 = 50$ and $K_2 = 70$, and set the risk-free rate to be $r = 0.10$. Assume also that the volatilities of the stocks, as described by Representation 2, are $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$, and let the correlation be $\rho = 0.75$. These volatilities and this correlation can be expressed in Representation 1 by, for example, the following volatility matrix:

$$\sigma = \begin{pmatrix} 0.20 & 0.00 \\ 0.225 & 0.1984313 \end{pmatrix} \tag{4.39}$$

This matrix is easily found by solving (2.36) and by letting $\sigma_{12} = 0$ in the solution.

By varying the number of replications n for the regular Monte Carlo Method as well as the Monte Carlo method with antithetic variates, the option price and the relative error can be observed as a function of the number of replications, see Figure 4.1.



(a) Option price. **(b)** Relative error.

Figure 4.1: Option prices and relative errors using the Monte Carlo method for the European two-asset correlation call option using different number of replications n. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

For the same option, the computational time for the different number of replications in the Monte Carlo method can be observed in Figure 4.2 below.

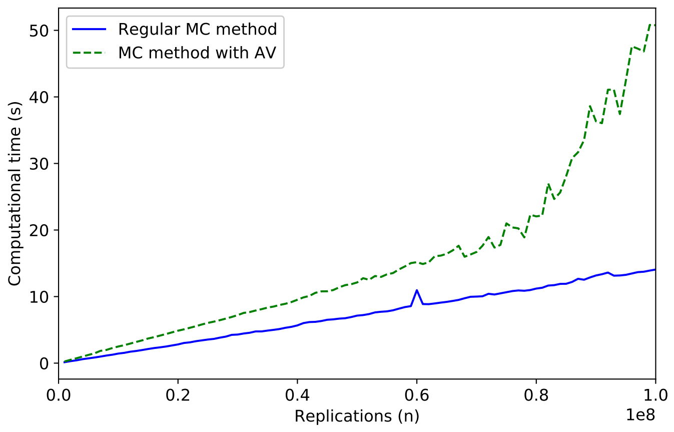


Figure 4.2: Computational times using the Monte Carlo method for the European two-asset correlation call option using different number of replications n. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

Choosing a fixed number of replications n it is easy to find confidence intervals for the estimates. Table 4.1 shows the estimates for three such fixed values of the number of replications n.

Table 4.1: Dependence on the number of replications n for Monte Carlo simulations of the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.

Replications n	Method	Estimated price	Confidence interval	Computational time	Relative error
1,000,000	Regular	4.7044	(4.6875, 4.7213)	0.1406 s	0.0006154
	AV	4.7120	(4.7020, 4.7220)	0.2500 s	0.0009995
10,000,000	Regular	4.7093	(4.7039, 4.7146)	1.4844 s	0.0004104
	AV	4.7085	(4.7053, 4.7117)	2.4844 s	0.0002469
100,000,000	Regular	4.7077	(4.7060, 4.7094)	15.1721 s	0.00007536
	AV	4.7075	(4.7065, 4.7085)	41.5943 s	0.00002645

As before, it is also interesting to see how the estimates depend on the correlation and the volatilities of the option. Tables 4.2 and 4.2 show how the price and confidence interval change for the two previously used Monte Carlo methods when the number of replications are $n = 1{,}000{,}000$.

Table 4.2: Dependence on the correlation ρ for Monte Carlo simulations of the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.10$, $\sigma_1 = 0.20$, $\sigma_2 = 0.30$. The number of replications is equal to 1,000,000.

Correlation ρ	Method	Estimated price	Confidence interval	Exact price	Relative error
-0.75	Regular	1.3047	(1.2972, 1.3122)	1.2981	0.005090
	AV	1.3029	(1.2979, 1.3079)		0.003731
-0.50	Regular	2.0880	(2.0772, 2.0987)	2.0913	0.001636
	AV	2.0897	(2.0827, 2.0967)		0.0008054
-0.25	Regular	2.8008	(2.7877, 2.8140)	2.7948	0.002157
	AV	2.7979	(2.7895, 2.8064)		0.001110
0.25	Regular	3.9619	(3.9459, 3.9779)	3.9599	0.0005048
	AV	3.9625	(3.9526, 3.9725)		0.0006597
0.50	Regular	4.4024	(4.3858, 4.4191)	4.4010	0.0003285
	AV	4.3981	(4.3880, 4.4082)		0.0006500
0.75	Regular	4.6849	(4.6681, 4.7017)	4.7073	0.004765
	AV	4.7042	(4.6942, 4.7143)		0.0006535

Table 4.3: Dependence on the volatilities σ_1 and σ_2 for Monte Carlo simulations of the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.10$ and $\rho = 0.75$. The number of replications is equal to 1,000,000.

Volatilities σ_1, σ_2	Method	Estimated price	Confidence interval	Exact price	Relative error
0.10, 0.15	Regular	2.0630	(2.0556, 2.0704)	2.0584	0.002234
	AV	2.0591	(2.0547, 2.0634)		0.0003261
0.20, 0.30	Regular	4.7052	(4.6883, 4.7221)	4.7073	0.0004586
	AV	4.7114	(4.7014, 4.7215)		0.0008730
0.40, 0.60	Regular	9.8033	(9.7633, 9.8433)	9.7424	0.006250
	AV	9.7574	(9.7326, 9.7822)		0.001540
0.80, 1.20	Regular	19.3039	(19.1951, 19.4127)	19.2751	0.001494
	AV	19.3148	(19.2426, 19.3871)		0.002059
1.60, 2.40	Regular	36.1671	(35.6585, 36.6756)	36.0371	0.003606
	AV	36.1320	(35.7812, 36.4829)		0.002634
3.20, 4.80	Regular	66.3985	(40.2512, 92.5459)	56.7448	0.1701
	AV	59.5473	(46.1413, 72.9532)		0.04939
6.40, 9.60	Regular	12.9838	(-4.2031, 30.1709)	64.8269	0.7997
	AV	52.4633	(-35.7854, 140.7121)		0.1907

4.4.2 Asian two-asset call option

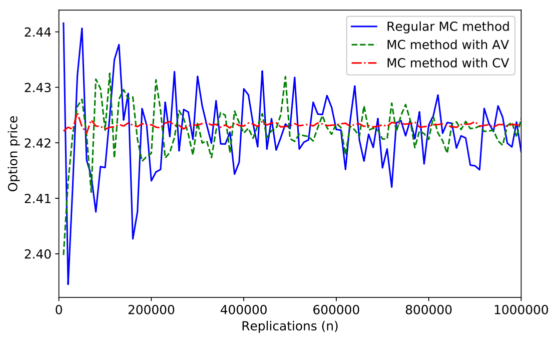
For pricing an Asian two-asset call option, three methods are used: The regular Monte Carlo method, the Monte Carlo method using antithetic variates (AV) and the Monte Carlo method using control variates (CV).

Let $N = 1000$ in (4.24) and (4.38), and suppose that $0 = t_0 < t_1 < \cdots < t_N = T$ are equally distanced in the interval $[0,T]$. Assume that the initial stock prices are

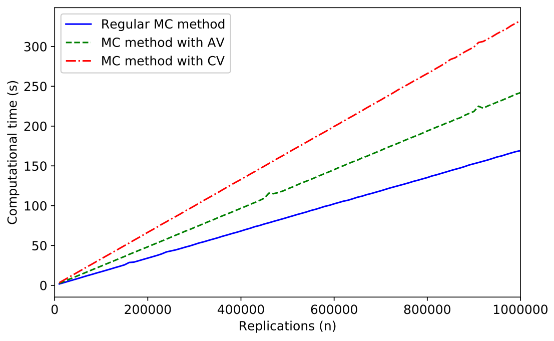
$S_1(0) = 52$ and $S_2(0) = 65$. Let the strike price be $K = 50$ and suppose the risk-free interest rate is $r = 0.10$. In Representation 2, assume that the volatilities of assets S_1 and S_2 are $\sigma_1 = 0.20$ respectively $\sigma_2 = 0.40$, and let the correlation be equal to $\rho = 0.50$. An approximate volatility matrix in Representation 1 for these parameters is given by

$$\sigma = \begin{pmatrix} 0.20 & 0.00 \\ 0.20 & 0.3464102 \end{pmatrix}. \tag{4.40}$$

Figure 4.3 shows how the price of this option and the computational time vary depending on the number of replications n in the Monte Carlo method.



(a) Option price.



(b) Computational time.

Figure 4.3: Option prices and computational times using the Monte Carlo method for an Asian two-asset call option using different number of replications n. Parameters: $S_1(0) = 51$, $S_2(0) = 48$, $K = 50$, $\alpha_1 = 0.50$, $\alpha_2 = 0.50$, $r = 0.10$, $\rho = 0.50$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.40$.

Table 4.4 summarises the statistics of Figure 4.3 for three different number of replications n, namely when $n = 10{,}000$, $n = 100{,}000$, and $n = 1{,}000{,}000$.

Table 4.4: Results of Monte Carlo simulations of the Asian two-asset call option. Parameters: $S_1(0) = 51$, $S_2(0) = 48$, $K = 50$, $\alpha_1 = 0.50$, $\alpha_2 = 0.50$, $r = 0.10$, $\rho = 0.50$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.40$.

Replications n	Method	Estimated price	Confidence interval	Computational time
10,000	Regular	2.4112	(2.3420, 2.4804)	1.7813 s
	AV	2.4342	(2.3980, 2.4704)	2.4844 s
	CV	2.4226	(2.4189, 2.4265)	3.4219 s
100,000	Regular	2.4009	(2.3790, 2.4227)	17.5784 s
	AV	2.4231	(2.4117, 2.4345)	24.8441 s
	CV	2.4236	(2.4224, 2.4247)	34.2973 s
1,000,000	Regular	2.4198	(2.4129, 2.4268)	175.2368 s
	AV	2.4221	(2.4186, 2.4258)	251.5273 s
	CV	2.4238	(2.4234, 2.4241)	343.6141 s

The results in Table 4.4 suggest that the control variates method outperforms the antithetic variates method as a variance reduction technique for the Asian two-asset call option since it gives rise to a smaller confidence interval. Table 4.5 and 4.6 show how the results change when the correlation and volatilities vary for $n = 100{,}000$.

Table 4.5: Dependence on the correlation ρ for Monte Carlo simulations of the Asian two-asset call option. Parameters: $S_1(0) = 51$, $S_2(0) = 48$, $K = 50$, $\alpha_1 = 0.50$, $\alpha_2 = 0.50$, $r = 0.10$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.40$. The number of replications is equal to 100,000.

Correlation ρ	Method	Estimated price	Confidence interval
-0.75	Regular	1.4512	(1.4385, 1.4638)
	AV	1.4410	(1.4347, 1.4474)
	CV	1.4458	(1.4436, 1.4480)
-0.50	Regular	1.6973	(1.6825, 1.7122)
	AV	1.6995	(1.6921, 1.7070)
	CV	1.7039	(1.7018, 1.7060)
-0.25	Regular	1.9170	(1.9001, 1.9339)
	AV	1.9122	(1.9037, 1.9208)
	CV	1.9157	(1.9138, 1.9176)
0.25	Regular	2.2838	(2.2634, 2.3043)
	AV	2.2662	(2.2559, 2.2766)
	CV	2.2697	(2.2683, 2.2710)
0.50	Regular	2.3969	(2.3750, 2.4187)
	AV	2.4091	(2.3979, 2.4204)
	CV	2.4238	(2.4227, 2.4250)
0.75	Regular	2.5749	(2.5513, 2.5984)
	AV	2.5695	(2.5573, 2.5818)
	CV	2.5676	(2.5667, 2.5686)

Table 4.6: Dependence on the volatilities σ_1, σ_2 for Monte Carlo simulations of the Asian two-asset call option. Parameters: $S_1(0) = 51$, $S_2(0) = 48$, $K = 50$, $\alpha_1 = 0.50$, $\alpha_2 = 0.50$, $r = 0.10$, and $\rho = 0.50$. The number of replications is equal to 100,000.

Volatilities σ_1, σ_2	Method	Estimated price	Confidence interval
0.10, 0.20	Regular	1.4233	(1.4120, 1.4345)
	AV	1.4251	(1.4201, 1.4302)
	CV	1.4229	(1.4226, 1.4232)
0.20, 0.40	Regular	2.4231	(2.4011, 2.4451)
	AV	2.4259	(2.4145, 2.4373)
	CV	2.4250	(2.4238, 2.4261)
0.40, 0.80	Regular	4.3753	(4.3294, 4.4212)
	AV	4.4090	(4.3826, 4.4354)
	CV	4.4428	(4.4378, 4.4479)
0.80, 1.60	Regular	8.4186	(8.3034, 8.5339)
	AV	8.4176	(8.3439, 8.4914)
	CV	8.4091	(8.3814, 8.4366)
1.60, 3.20	Regular	15.4854	(15.0528, 15.9180)
	AV	15.5121	(15.2052, 15.8189)
	CV	15.5198	(15.2767, 15.7628)
3.20, 6.40	Regular	25.0872	(21.1509, 29.0234)
	AV	24.0142	(21.7430, 26.2855)
	CV	25.4563	(22.5480, 28.3645)
6.40, 12.80	Regular	18.5260	(13.4732, 23.5788)
	AV	19.4838	(15.6469, 23.3208)
	CV	18.0951	(13.9332, 22.2569)

5 Partial Differential Equations and the Finite Element Method

Both the binomial model and the Monte Carlo method have shown to be rather inaccurate for large volatilities, thus a third pricing method based on solving partial differential equations (PDEs) is presented in this chapter. This method has been studied thoroughly for single-asset options since the introduction of the famous Black-Scholes formula in 1973 [15]. Multi-asset options are a bit more difficult to price by PDEs since they give rise to multi-dimensional equations with boundary conditions that can be hard to find. The finite element method (FEM) is a way to numerically solve PDEs. Hereinafter the finite element method applied to the pricing PDE will often be referred to as the PDE method.

5.1 The multi-dimensional Black-Scholes equation

As before, let $S(t) = (S_1(t), \dots, S_d(t))$ be the prices at time t of d underlying risky assets for a European multi-asset option. Moreover, assume that the risky assets satisfy the stochastic differential equations in Representation 1, which by dropping the t variable can be expressed in the shorter form

$$dS_i = S_i \mu_i dt + \sum_{j=1}^{d} S_i \sigma_{ij} dW_j, \quad i = 1, \dots, d, \tag{5.1}$$

where dW_j are independent one-dimensional Brownian motions, σ_{ij} corresponds to element i, j in the volatility matrix, and μ_i is the drift rate of asset S_i. Moreover, letting $V = V(S(t), t)$ be the price of a European multi-asset option the following theorem holds.

Theorem 5.1.1 (The multi-dimensional Black-Scholes equation)
The price function $V(S,t)$ of a European multi-asset option with time of maturity T satisfies

$$\frac{\partial V}{\partial t} + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij} S_i S_j \frac{\partial^2 V}{\partial S_i \partial S_j} + \sum_{i=1}^{d} (r - q_i) S_i \frac{\partial V}{\partial S_i} - rV = 0, \tag{5.2}$$

where r is the risk-free interest rate, q_i is the continuous dividend rate of asset S_i and $A = [a_{ij}]$ has entries given by

$$a_{ij} = \sum_{k=1}^{d} \sigma_{ik} \sigma_{jk} \quad i, j = 1, \dots, d. \tag{5.3}$$

Moreover, the terminal value is given by the payoff function Y, i.e.,

$$V(S,T) = Y(S). \tag{5.4}$$

Proof. Recall the discussion in Chapter 2 where it was argued that the value of an option should be equal to the value of a self-financing hedging portfolio. Assume that the shares Δ_i of asset S_i for $i = 1, \dots, d$ are chosen in such a way that the self-financing portfolio Π given by

$$\Pi(t) = V(S_1(t), \dots, S_d(t), t) - \sum_{i=1}^{d} \Delta_i S_i(t) \tag{5.5}$$

is risk-free in the interval $(t, t+dt)$. By Itô's formula for functions of several random variables, it holds that

$$\begin{aligned} d\Pi &= dV - \sum_{i=1}^{d} \Delta_i dS_i - \sum_{i=1}^{d} \Delta_i S_i q_i dS_i && (5.6)\\ &= \left(\frac{\partial V}{\partial t} + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij} S_i S_j \frac{\partial^2 V}{\partial S_i \partial S_j} \right) dt + \sum_{i=1}^{d} \frac{\partial V}{\partial S_i} dS_i \\ &\quad - \sum_{i=1}^{d} \Delta_i dS_i - \sum_{i=1}^{d} \Delta_i S_i q_i dt. && (5.7) \end{aligned}$$

In order for the portfolio to be risk-free, the terms involving dS_i in (5.7) should be zero. Thus the number of shares Δ_i must satisfy

$$\Delta_i = \frac{\partial V}{\partial S_i}. \tag{5.8}$$

Moreover, the portfolio should also earn the risk-free interest rate, meaning that

$$d\Pi = r\Pi dt = r \left(V - \sum_{i=1}^{d} \Delta_i S_i \right) dt. \tag{5.9}$$

Combining (5.7), (5.8), and (5.9), and eliminating dt, yields

$$\frac{\partial V}{\partial t} + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij} S_i S_j \frac{\partial^2 V}{\partial S_i \partial S_j} + \sum_{i=1}^{d} (r - q_i) S_i \frac{\partial V}{\partial S_i} - rV = 0, \tag{5.10}$$

which is the Black-Scholes equation for multi-asset options [4]. Moreover, V is equal to the payoff at the time of maturity, and thus

$$V(S_1, \dots, S_d, T) = Y(S_1, \dots, S_d). \tag{5.11}$$

□

Equation (5.2) is a multidimensional parabolic equation since A is a symmetrical non-negative matrix [4]. In order to determine the price of a European derivative, (5.2) should be solved in the domain $\Omega = \{0 \le S_i \le \infty, i = 1, \dots, d; 0 \le t \le T\}$.

By suitable transformations, (5.2) becomes a Cauchy problem with solution formula

$$V(S,t) = \left[\frac{1}{2\pi(T-t)}\right]^{\frac{d}{2}} \frac{e^{-r(T-t)}}{|\det A|^{\frac{1}{2}}} \int_0^\infty \cdots \int_0^\infty \frac{Y(\eta_1,\dots,\eta_d)}{\eta_1\dots\eta_d} \exp\left[-\frac{\boldsymbol{\alpha}^T A^{-1}\boldsymbol{\alpha}}{2(T-t)}\right] d\eta_1 \dots d\eta_d, \tag{5.12}$$

where Y is the payoff function, $T-t$ is the time left until maturity, $\boldsymbol{\alpha} = (\alpha_1,\dots,\alpha_d)^T$, and

$$\alpha_i = \log\frac{S_i}{\eta_i} + \left(r - q_i - \frac{\alpha_{ii}}{2}\right)(T-t), \quad i = 1,\dots,d. \tag{5.13}$$

A detailed proof of the solution formula (5.12) can be found in [4]. It should be remarked that this integral has singularities in the integrand and that the integral is very difficult to evaluate. Thus it is often more convenient to use the finite element method to find an approximate solution to (5.2) and (5.4).

5.2 The finite element method

The finite element method (FEM) is a widely applied method to find numerical approximations of partial differential equations on any domain. Solving a PDE with FEM requires specification of the weak formulation, then formulating the finite element problem and solving it using discrete methods. The results presented in this chapter are based on implementations using FEniCS and thus the general notations used in this section are similar to those used in FEniCS [16] [17].

A typical problem could be to find the solution to the PDE

$$\nabla^2 u = f \quad \text{in } \Omega, \tag{5.14}$$

$$u = u_0 \quad \text{on } \partial\Omega, \tag{5.15}$$

on the domain $\Omega \subset \mathbb{R}^2$. Starting with this kind of problem on a strong form gives a clear example on how to use FEM.

5.2.1 Weak formulation

As previously stated, the first step is to find the weak formulation. To do so, we multiply equation (5.14) with a test function, integrate over the given domain Ω, and then integrate by parts to reduce all second order derivatives to first order. For this, a space containing the test functions need to be defined. It is possible to choose the test space as a subspace of a Sobolev space. In order to define the Sobolev space, the following two definitions are needed.

Definition 7 ($L^2(\Omega)$ space)
Let Ω be an open subset of $\mathbb{R}^N$ with piecewise smooth boundary. Then the space $L^2(\Omega)$ is defined by

$$L^2(\Omega) = \left\{ v : \Omega \to \mathbb{R} \Big| \int_\Omega v^2 dx < \infty \right\}. \tag{5.16}$$

Definition 8 (Weak derivative)
Let $v \in L^2(\Omega)$. If the weak derivative of v exists, it is defined as a function $\partial^\alpha v \in L^2(\Omega)$ satisfying

$$\int_\Omega \partial^\alpha v \phi dx = (-1)^{|\alpha|} \int_\Omega v \partial^\alpha \phi dx \tag{5.17}$$

for all $\phi \in C_0^\infty(\Omega)$ where

$$\partial^\alpha \phi = \frac{\partial^{|\alpha|}}{\partial^{\alpha_1} x_1 \partial^{\alpha_2} x_2 \cdots \partial^{\alpha_n} x_n}. \tag{5.18}$$

Combining the above definitions it is possible to define the Sobolev space in a simple way as follows.

Definition 9 (Sobolev space $H^m(\Omega)$)
Let Ω be an open subset of $\mathbb{R}^N$ with piecewise smooth boundary. Let $m \geq 0$ be an integer. Then the Sobolev space $H^m(\Omega)$ is defined to be the set of all functions $v \in L^2(\Omega)$ such that $\partial^\alpha v \in L^2(\Omega)$ for all $|\alpha| \leq m$.

A corresponding norm to H^m can in turn be defined as

$$||v||_{H^m} = \left(\sum_{|\alpha| \leq m} \int_\Omega |\partial^\alpha v|^2 dx \right)^{1/2}. \tag{5.19}$$

As a test space, consider the following subspace of H^1, consisting of functions in H^1 that are zero at the boundary of the given domain.

Definition 10 (Test space)
Let Ω be an open subset of $\mathbb{R}^N$ with piecewise smooth boundary. Then the subspace of H^1 with functions that are zero on the boundary is defined by

$$H^1_{0,\partial\Omega}(\Omega) = \{v \in H^1(\Omega) : v = 0 \textit{ on } \partial\Omega \textit{ in the trace sense}\}. \tag{5.20}$$

This test space is from here on denoted H_0^1.

Similarly, a so called trial space can be defined for which the functions belonging to the space satisfy the given boundary conditions in the observed PDE.

Definition 11 (Trial space)
Let Ω be an open subset of $\mathbb{R}^N$ with piecewise smooth boundary. Then the subspace of H^1 functions that are u_0 on the boundary is defined by

$$H^1_{u_0,\partial\Omega}(\Omega) = \{v \in H^1(\Omega) : v = u_0 \textit{ on } \partial\Omega \textit{ in the trace sense}\}. \tag{5.21}$$

This trial space is from here on denoted $H^1_{u_0}$.

In order to formulate the weak form of the problem given by (5.15), the PDE is multiplied with a test function $v \in H_0^1$ and then integrated over the domain. This yields

$$\int_\Omega \nabla^2 u \cdot v dx = \int_\Omega f v dx \tag{5.22}$$

where the left hand term can be simplified using integration by parts and Green's formula. That is,

$$\int_\Omega \nabla^2 u \cdot v dx = \int_\Omega \nabla u \cdot \nabla v dx - \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds \tag{5.23}$$

where the second term of the right hand side is zero due to the definition of the test functions. Thus, the weak formulation is as follows: Find $u \in H^1_{u_0}$ such that

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx \tag{5.24}$$

holds for all $v \in H^1_0$.

5.2.2 Finite element problem

The second step is to define the finite element problem. That is, discretize the weak formulation and look for a solution in a discrete trial space. In mathematical terms, define $H^1_{0|h}$ and $H^1_{u_0|h}$ as finite dimensional subspaces of H^1_0 and $H^1_{u_0}$ respectively. Here h denotes the size of the partition in the discrete spaces. For the proposed example, the finite element problem is: Find $u_h \in H^1_{u_0|h}$ such that

$$\int_\Omega \nabla u_h \cdot \nabla v dx = \int_\Omega f v dx \tag{5.25}$$

holds for all $v \in H^1_{0|h}$.

5.2.3 Solving the finite element problem

In order to solve a PDE with FEM, one must construct the finite element function spaces such that combinations of the functions form a finite subset on the given domain Ω. Thus, the domain is split into smaller domains where local functions act and are added together in order to create a global function space satisfying the desired continuity and solution. This is done by using so called finite elements.

Definition 12 (Finite element)
The triple $(T, \mathcal{V}, \mathcal{L})$ is a finite element if

1. *T is a bounded closed subset of $\mathbb{R}^{\mathbb{N}}$ with nonempty interior and piecewise smooth boundary.*
2. *$\mathcal{V}$ is a finite dimensional function space on the domain T, of dimension n.*
3. *The set of degrees of freedom $\mathcal{L} = \{l_1, l_2, \dots, l_n\}$ is a basis for the dual space $\mathcal{V}'$, the space of bounded linear functionals on the space $\mathcal{V}$.*

Further, the following can be observed about the set $\mathcal{L}$.

Lemma 5.2.1 (Unisolvence)
$\mathcal{L}$ is a basis for the dual space $\mathcal{V}'$ if and only if $\mathcal{L}v = 0$ implies $v = 0$.

Definition 13 (Nodal basis)
The nodal basis $\{\phi\}_{i=1}^n$ for a finite element $(T, \mathcal{V}, \mathcal{L})$ is the unique basis satisfying $l_i(\phi_j) = \delta_{ij}$ for $j = 1, \dots, n$.

The nodal basis has the important property that if

$$u_h = \sum_{j=1}^n u_j \phi_j \tag{5.26}$$

then $l_i(u_h) = u_i$.

There are a few finite elements commonly used in FEM. For the purpose of this thesis, the so called Lagrange element (CG_2) is chosen.

Definition 14 (Lagrange element)
For $q = 1,2,\dots,$ the Lagrange element (CG_q) is defined by

$$\begin{aligned} &T \in \{\textit{interval, triangle, tetrahedron}\}, \\ &\mathcal{V} = \mathcal{P}_q(T) = \{\textit{polynomials on } T \textit{ of degree } \leq q\}, \\ &l_i(v) = v(x^i), \quad i = 1, \dots, n(q), \end{aligned} \tag{5.27}$$

where $\{x^i\}_{i=1}^{n(q)}$ is an enumeration of points in the domain T defined by

$$x = \begin{cases} i/q & 0 \leq i \leq q \qquad T \textit{ interval}, \\ (i/q, j/q), & 0 \leq i+j \leq q \qquad T \textit{ triangle}, \\ (i/q, j/q, k/q) & 0 \leq i+j+k \leq q \qquad T \textit{ tetrahedron}, \end{cases} \tag{5.28}$$

and the dimension $n(q)$ of the Lagrange finite element corresponds to the dimension of the degree q on T and

$$n(q) = \begin{cases} q+1 & T \textit{ interval}, \\ \frac{1}{2}(q+1)(q+2) & T \textit{ triangle}, \\ \frac{1}{6}(q+1)(q+2)(q+3) & T \textit{ tetrahedron}. \end{cases} \tag{5.29}$$

For the example in this section, it is now possible to make the anzats

$$u_h(x) = \sum_{j=1}^{N} U_j \phi_j(x) \tag{5.30}$$

where $\phi_j : \Omega \to \mathbb{R}$ for $j = 1,\dots,N$ is a basis for $H^1_{u_0|h}$. Insert (5.30) into the finite element problem (5.25) and take $v = \widehat{\phi}_i$ for $i = 1, \dots, N$. It yields that

$$\sum_{j=1}^{N} U_j \nabla \phi_j \cdot \nabla \widehat{\phi}_i dx = \int_{\Omega} f \widehat{\phi}_i dx \tag{5.31}$$

for $i = 1, \dots, N$, where $U \in \mathbb{R}^N$ is the vector to be computed.

For the implementations in this thesis, T is assumed to be a small triangular domain. That is, the domain Ω is split into a large number of triangles for which functions belonging to the space $\mathcal{V}(T)$ are defined for each triangle.

5.3 PDE and FEM for the European two-asset correlation call option

The PDE method can be used to price the European two-asset correlation call option. The PDE formulation is given in the following theorem.

Theorem 5.3.1
The price function $V(S,t)$ of a European two-asset correlation call option with maturity T and payoff $Y(S(T)) = (S_2(T) - K_2)^+$ if $S_1(T) > K_1$ and zero otherwise

satisfies the following initial value problem when the underlying assets pay no dividend:

$$
\begin{aligned}
-\frac{\partial V}{\partial t} + \frac{1}{2}\Bigg(a_{11}S_1^2\frac{\partial^2 V}{\partial S_1^2} + a_{12}S_1S_2\frac{\partial^2 V}{\partial S_1\partial S_2} + a_{21}S_1S_2\frac{\partial^2 V}{\partial S_2\partial S_1} \\
+a_{22}S_2^2\frac{\partial^2 V}{\partial S_2^2}\Bigg) + r\left(S_1\frac{\partial V}{\partial S_1} + S_2\frac{\partial V}{\partial S_2}\right) - rV = 0,
\end{aligned}
\tag{5.32a}
$$

$$
V(S,0) = Y(S), \tag{5.32b}
$$

and it has the following asymptotic behaviour

$$
V(S,t) \sim C(S_2, K_2, r, T-t, \sqrt{a_{22}}) \ \text{as}\ S_1 \to \infty, \tag{5.33a}
$$

$$
V(S,t) \sim S_2\Phi(y_1 + \rho\sqrt{a_{22}}\sqrt{T-t}) - K_2e^{-r(T-t)}\Phi(y_1) \ \text{as}\ S_2 \to \infty, \tag{5.33b}
$$

$$
\lim_{S_1\to 0} V(S,t) = \lim_{S_2\to 0} V(S,t) = 0, \tag{5.33c}
$$

where $C(S_i, K_i, r, T-t, \sqrt{a_{ii}})$ is the Black-Scholes price function of a standard European call option with underlying asset price S_i, strike price K_i, interest rate r, time until maturity $T-t$, and volatility $\sqrt{a_{ii}}$; Φ is the cumulative standard normal distribution; y_1 is given by (2.41) *with σ_1 and σ_2 replaced by $\sqrt{a_{11}}$ respectively $\sqrt{a_{22}}$; and ρ is given by* (2.44)*. That is, for volatilities σ_{11}, σ_{12}, σ_{21}, σ_{22} it holds*

$$
a_{11} = \sigma_{11}^2 + \sigma_{12}^2, \tag{5.34}
$$

$$
a_{22} = \sigma_{21}^2 + \sigma_{22}^2, \tag{5.35}
$$

$$
a_{12} = a_{21} = \sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}. \tag{5.36}
$$

Proof. Equation (5.32a) and (5.32b) follow immediately from Theorem 5.1.1 and by a change of variable $t \to T - t$.

When $S_1 \to \infty$ and K_1 is bounded the condition $S_1(T) > K_1$ will hold almost surely. Hence the European two-asset correlation call option essentially becomes identical to a European call option in that case and (5.33a) follows. The boundary condition (5.33b) can be justified by using the fact that there exists a closed formula for the European two-asset correlation call option which is easy to evaluate. Recalling (2.41)-(2.44), the price is given by

$$
V = S_2M\left(y_2 + \sqrt{a_{22}}\sqrt{T-t}, y_1 + \rho\sqrt{a_{22}}\sqrt{T-t}; \rho\right) - K_2e^{-r(T-t)}M(y_2,y_1;\rho) \tag{5.37}
$$

where

$$
\begin{aligned}
y_1 &= \frac{\log(S_1/K_1) + (r - a_{11}/2)(T-t)}{\sqrt{a_{11}}\sqrt{T-t}}, \\
y_2 &= \frac{\log(S_2/K_2) + (r - a_{22}/2)(T-t)}{\sqrt{a_{22}}\sqrt{T-t}},
\end{aligned}
\tag{5.38}
$$

$$
\rho = \frac{\frac{1}{2}(a_{12} + a_{21})}{\sqrt{a_{11}a_{22}}}), \tag{5.39}
$$

and

$$M(a,b;\rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \int_{-\infty}^{a} \int_{-\infty}^{b} \exp\left[-\frac{x^2 - 2\rho xy + y^2}{2(1-\rho^2)}\right] dxdy. \tag{5.40}$$

It is easily checked that

$$\begin{aligned} V &= S_2 M\left(y_2 + \sqrt{a_{22}}\sqrt{T-t}, y_1 + \rho\sqrt{a_{22}}\sqrt{T-t}; \rho\right) - K_2 e^{-r(T-t)} M(y_2, y_1; \rho) \\ &\sim \frac{S_2}{\sqrt{2\pi}} \int_{-\infty}^{y_1 + \rho\sqrt{a_{22}}\sqrt{T-t}} e^{-\frac{x^2}{2}} dx - K_2 e^{-r(T-t)} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{y_1} e^{-\frac{x^2}{2}} dx \quad \text{as } S_2 \to \infty. \end{aligned} \tag{5.41}$$

The right hand side of (5.41) can also be expressed as

$$S_2 \Phi\left(y_1 + \rho\sqrt{a_{22}}\sqrt{T-t}\right) - K_2 e^{-r(T-t)} \Phi(y_1) \tag{5.42}$$

where Φ is the cumulative standard normal distribution function given by

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{y^2}{2}} dy. \tag{5.43}$$

This justifies the boundary condition (5.33b).

The boundary conditions (5.33c) follow directly from the fact that the option will become worthless if either of the underlying assets approaches zero. $\square$

In order to solve (5.33) by using FEM it is necessary to discretize the problem. To begin with, the computational domain $(0,T) \times (0,\infty) \times (0,\infty)$ is limited to $(0,T) \times (0,Z_1) \times (0,Z_2) = (0,T) \times \Omega$ where $Z_1 >> K_1$ and $Z_2 >> K_2$. Moreover, the time derivative is approximated by the Euler backward method, i.e.,

$$\frac{\partial V}{\partial t}(S,t) = \frac{V(S,t) - V(S,t-\Delta t)}{\Delta t} + O(\Delta t), \tag{5.44}$$

$$V(S,t) = V(S,t-\Delta t) + \frac{\partial V}{\partial t}(S,t)\Delta t + O(\Delta t^2), \tag{5.45}$$

where Δt is the size of each time step. Substituting (5.32a) into (5.45) and using the notations $V^j = V(S,t)$ and $V^{j-1} = V(S,t-\Delta t)$ gives the iterative equation

$$\begin{aligned} V^{j-1} = V^j - \Bigg[\frac{1}{2}\Bigg(& a_{11} S_1^2 \frac{\partial^2 V^j}{\partial S_1^2} + a_{12} S_1 S_2 \frac{\partial^2 V^j}{\partial S_1 \partial S_2} + a_{21} S_1 S_2 \frac{\partial^2 V^j}{\partial S_2 \partial S_1} \\ &+ a_{22} S_2^2 \frac{\partial^2 V^j}{\partial S_2^2}\Bigg) + r\left(S_1 \frac{\partial V^j}{\partial S_1} + S_2 \frac{\partial V^j}{\partial S_2}\right) - rV^j\Bigg]\Delta t. \end{aligned} \tag{5.46}$$

The weak formulation is obtained by multiplying (5.46) by a test function $\phi \in H_0^1(\Omega)$ and integrating over Ω. By integration by parts and by using the notation $(f,g) =$

$\int_\Omega fgdS$, the weak formulation reads: find $V \in H^1_{u_0}$ such that

$$\begin{aligned}(V^j,\phi) + \Big\{ \frac{1}{2}a_{11}[2(S_1\partial_{S_1}V^j,\phi) + (S_1^2\partial_{S_1}V^j,\partial_{S_1}\phi)] + \frac{1}{2}a_{12}[(S_2\partial_{S_2}V^j,\phi) \\ +(S_1S_2\partial_{S_2}V^j,\partial_{S_1}\phi)] + \frac{1}{2}a_{21}[(S_1\partial_{S_1}V^j,\phi) + (S_1S_2\partial_{S_1}V^j,\partial_{S_2}\phi)] \\ +\frac{1}{2}a_{22}[2(S_2\partial_{S_2}V^j,\phi) + (S_2^2\partial_{S_2}V^j,\partial_{S_2}\phi)] - r[(S_1\partial_{S_1}V^j,\phi) + (S_2\partial_{S_2}V^j,\phi)] \\ +r(V^j,\phi)\Big\}\Delta t = (V^{j-1},\phi),\end{aligned} \tag{5.47}$$

holds $\forall\phi \in H^1_0$ and such that V satisfies $V^0(S) = Y(T)$ $\forall S \in \Omega$.

5.4 PDE and FEM for the European two-asset maximum call option

The maximum call option is a type of rainbow option that can also be priced by using the multi-dimensional Black-Scholes equation given by (5.2) and (5.4) together with suitable boundary conditions. Moreover, the price can be approximated by the finite element method.

Theorem 5.4.1
The price function $V(S,t)$ of a European two-asset maximum call option with maturity T and payoff $Y(S(T)) = max\{(S_1(T)-K_1)^+,(S_2(T)-K_2)^+\}$ satisfies the following initial value problem when the underlying assets pay no dividend:

$$\begin{aligned}-\frac{\partial V}{\partial t} + \frac{1}{2}\bigg(a_{11}S_1^2\frac{\partial^2 V}{\partial S_1^2} + a_{12}S_1S_2\frac{\partial^2 V}{\partial S_1\partial S_2} + a_{21}S_2S_1\frac{\partial^2 V}{\partial S_2\partial S_1} \\ +a_{22}S_2^2\frac{\partial^2 V}{\partial S_2^2}\bigg) + r\bigg(S_1\frac{\partial V}{\partial S_1} + S_2\frac{\partial V}{\partial S_2}\bigg) - rV = 0,\end{aligned} \tag{5.48a}$$

$$V(S,0) = Y(S), \tag{5.48b}$$

and it has the following asymptotic behaviour

$$\begin{aligned}V(S,t) \sim \max\Big\{S_1 - K_1e^{-r(T-t)}, S_2 - K_2e^{-r(T-t)}\Big\} \\ \text{as } S_1 \to \infty \text{ and/or } S_2 \to \infty,\end{aligned} \tag{5.48c}$$

$$\lim_{S_1\to 0} V(S,t) = C(S_2,K_2,r,T-t,\sqrt{a_{22}}), \tag{5.48d}$$

$$\lim_{S_2\to 0} V(S,t) = C(S_1,K_1,r,T-t,\sqrt{a_{11}}), \tag{5.48e}$$

where $C(S_i,K_i,r,T-t,\sqrt{a_{ii}})$ is the Black-Scholes price function of a standard European call option with underlying asset price S_i, strike price K_i, interest rate r, time until maturity $T-t$, and volatility $\sqrt{a_{ii}}$. For volatilities σ_{11}, σ_{12}, σ_{21}, σ_{22} it holds

$$a_{11} = \sigma_{11}^2 + \sigma_{12}^2, \tag{5.49}$$

$$a_{22} = \sigma_{21}^2 + \sigma_{22}^2, \tag{5.50}$$

$$a_{12} = a_{21} = \sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}. \tag{5.51}$$

Proof. Equation (5.48a) and (5.48b) follow immediately from Theorem 5.1.1 and by a change of variable $t \to T - t$.

The boundary conditions when $S_1 \to \infty$ and/or $S_2 \to \infty$ given by (5.48c) are straightforward. When $S_1 \to \infty$ and S_2 stays bounded the option can be valued as a call option on S_1 with strike K_1. It follows from the Black-Scholes formula for single assets that the price of such an option approaches $S_1 - K_1 e^{-r(T-t)}$ when $S_1 \to \infty$. The same reasoning holds for the boundary condition when $S_2 \to \infty$ and S_1 stays bounded, meaning that the option price approaches $S_2 - K_2 e^{-r(T-t)}$. When both $S_1 \to \infty$ and $S_2 \to \infty$, then the option price should be the maximum of the values derived above, i.e., $\max\left\{S_1 - K_1 e^{-r(T-t)}, S_2 - K_2 e^{-r(T-t)}\right\}$. Thus this becomes the general way to express the boundary condition when either of the underlying assets approaches infinity.

When $S_1 \to 0$ the European two-asset maximum call option essentially becomes a regular European call option on S_2 with strike K_2 and volatility $\sqrt{a_{22}}$. The volatility $\sqrt{a_{22}}$ of this option can easily be derived from the definition of the stochastic differential equations given by (5.1). Analogously, when $S_2 \to 0$ the European two-asset maximum call option can be regarded as a regular European call option on S_1 with strike K_1 and volatility $\sqrt{a_{11}}$. This completes the justification of the final boundary condition. $\square$

The weak formulation of (5.48) which is used to implement the finite element method reads as (5.47) but with the payoff corresponding to the European two-asset maximum call option.

5.5 Results of the PDE and FEM approach

In this section, PDEs and FEM are used to price the European two-asset correlation call option and the European two-asset maximum call option. As in the previous result sections, the dependence on the correlation and the volatilities is considered. Moreover, the impact of the size of the domain and the number of cells is also studied.

FEM has been implemented through the FEniCS project in Python. The corresponding code is presented in Appendix F.

5.5.1 European two-asset correlation call option

To price the same European two-asset correlation call option used in the previous methods, the strike prices are set to $K_1 = 50$ and $K_2 = 70$, and the risk-free rate is $r = 0.10$. Using the correlation $\rho = 0.75$ and volatilities $\sigma_1 = 0.20$, $\sigma_2 = 0.30$, the initial option price can be estimated to 4.7077 assuming initial stock prices $S_1(0) = 52$ and $S_2(0) = 65$. As before, it is assumed the underlying assets pay no dividends, and it can be shown that the theoretical option price is approximately 4.7073.

For the above estimate, a rectangle mesh with $nx = ny$ number of cells in each direction is used. This results in a total of $2 \cdot nx \cdot ny$ number of triangles and a total number of $(nx + 1)(ny + 1)$ vertices. For all possible initial values $S_1(0)$ and $S_2(0)$

the option price is shown in Figure 5.1 below.

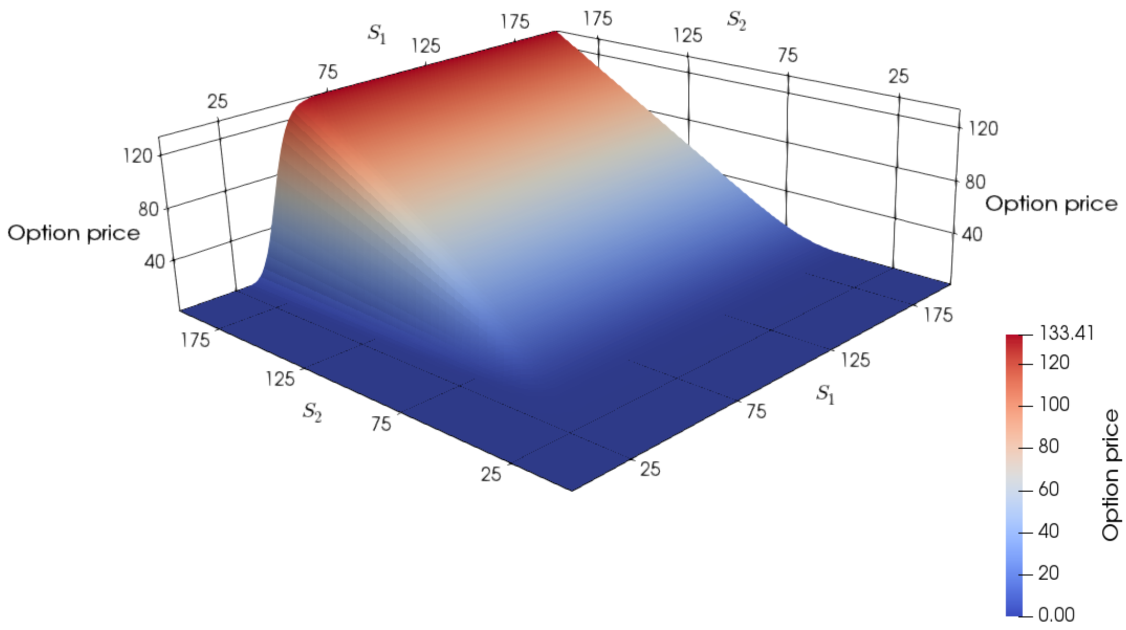
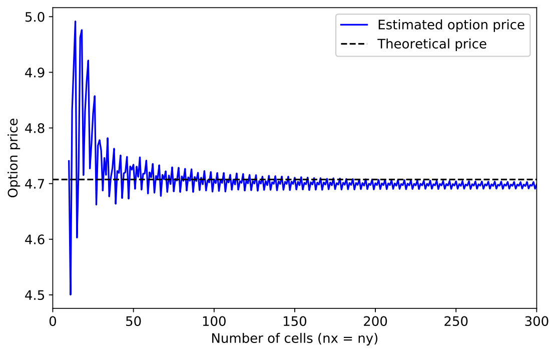
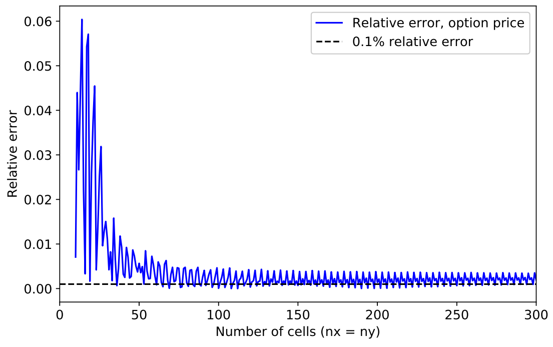


Figure 5.1: Option prices for the European two-asset correlation call option using the PDE method. Parameters: $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$. The domain is $(0.1, 200) \times (0.1, 200)$ with a mesh of size 100×100. The time step is 0.01.

For the previously mentioned parameters, it is of interest to see how the solution depends on the number of cells and the size of the domain to which $S_1(t)$ and $S_2(t)$ belong. The option price and the relative error is thus examined for different number of cells in the domain in Figure 5.2.



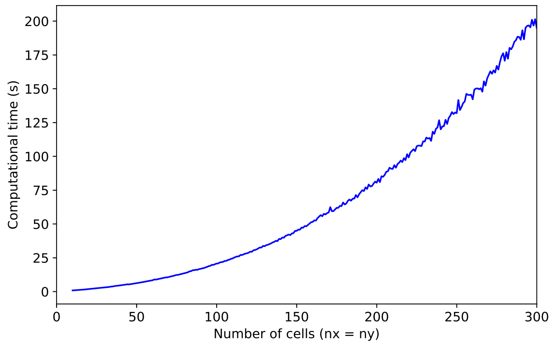
(a) Option price.



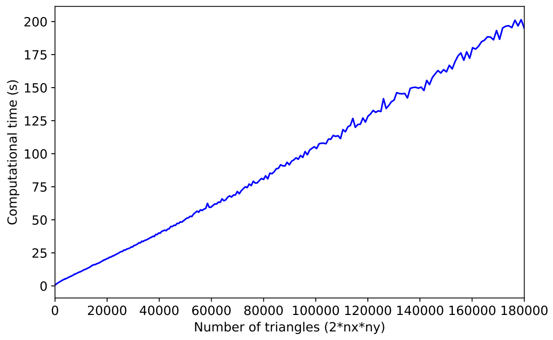
(b) Relative error.

Figure 5.2: Option prices and relative errors for the European two-asset correlation call option using different number of cells $nx = ny$ in the mesh in the PDE method. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.5$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$. The domain is $(0.1, 200) \times (0.1, 200)$ with a mesh of size 100×100. The time step is 0.01.

Similarly, it is also of interest to examine the computational time. Figure 5.3 below shows how the computational time grows by the number of cells in each direction of the mesh as well as by the number of triangles in the mesh.



(a) Number of cells in each direction.



(b) Number of triangles in the mesh.

Figure 5.3: Computational times for the European two-asset correlation call option using different number of cells $nx = ny$ in the mesh in the PDE method. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$.The domain is $(0.1, 200) \times (0.1, 200)$ with a mesh of size 100×100. The time step is 0.01.

In addition to this, Table 5.1 shows how the solution depends on the underlying domain and its partition. A comparison of three domains with different number cells in each direction of the mesh is shown in the same table.

Table 5.1: Dependence on the number of cells in the mesh and the size of the domain for PDE computations of the European two-asset correlation call option. Parameters: $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$. The tolerance is 10^{-10} and the time step is of size 0.01.

$nx = ny$	Domain	Estimated price	Relative error	Computational time
50	$(0.1, 200) \times (0.1, 200)$	4.7340	0.005670	6.7275 s
	$(0.1, 400) \times (0.1, 400)$	4.8254	0.02509	6.4884 s
	$(0.1, 800) \times (0.1, 800)$	4.8800	0.03668	6.5360 s
100	$(0.1, 200) \times (0.1, 200)$	4.7077	0.00007215	21.0625 s
	$(0.1, 400) \times (0.1, 400)$	4.7344	0.005756	21.5561 s
	$(0.1, 800) \times (0.1, 800)$	4.8251	0.02502	20.9703 s
150	$(0.1, 200) \times (0.1, 200)$	4.7107	0.0007091	45.4331 s
	$(0.1, 400) \times (0.1, 400)$	4.6853	0.004670	45.4727 s
	$(0.1, 800) \times (0.1, 800)$	4.7523	0.009555	45.3286 s
200	$(0.1, 200) \times (0.1, 200)$	4.7007	0.001410	82.6149 s
	$(0.1, 400) \times (0.1, 400)$	4.7073	0.000006656	80.7336 s
	$(0.1, 800) \times (0.1, 800)$	4.7346	0.005799	80.6539 s
250	$(0.1, 200) \times (0.1, 200)$	4.7042	0.0006586	133.1556 s
	$(0.1, 400) \times (0.1, 400)$	4.6993	0.001702	131.5817 s
	$(0.1, 800) \times (0.1, 800)$	4.6640	0.009214	131.9189s
300	$(0.1, 200) \times (0.1, 200)$	4.6983	0.001927	199.5349 s
	$(0.1, 400) \times (0.1, 400)$	4.7104	0.0006445	198.4216 s
	$(0.1, 800) \times (0.1, 800)$	4.6851	0.004729	198.2054 s

As for previous pricing methods, the dependence on the correlation and volatilities is observed for the finite element method as well. In Table 5.2 the correlation is examined, while in Table 5.3 the volatilities are examined.

Table 5.2: Dependence on the correlation ρ of the European two-asset correlation call option using the PDE method. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.1$, $\sigma_1 = 0.2$ and $\sigma_2 = 0.3$. The domain is set to $(0.1, 200) \times (0.1, 200)$ with mesh size 100×100. A tolerance of 10^{-10} is used and the step length is 0.01.

Correlation ρ	Estimated price	Exact price	Relative error
-0.75	1.3712	1.2981	0.05670
-0.50	2.1647	2.0914	0.03506
-0.25	2.8611	2.7948	0.02373
0.25	3.9989	3.9599	0.009828
0.50	4.4214	4.4010	0.004627
0.75	4.7077	4.7073	0.00007215

Table 5.3: Dependence on the volatilities σ_1, σ_2 for the PDE method of the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $r = 0.1$ and $\rho = 0.75$. The domain is set to $(0.1, 200) \times (0.1, 200)$ with mesh size 100×100. A tolerance of 10^{-10} is used and the step length is 0.01.

Volatilities σ_1, σ_2	Estimated price	Exact price	Relative error
0.10, 0.15	2.0607	2.0584	0.001104
0.20, 0.30	4.7077	4.7073	0.00007215
0.40, 0.60	9.7419	9.7424	0.00004966
0.80, 1.20	19.1997	19.2751	0.003913
1.60, 2.40	35.0139	36.0371	0.02839
3.20, 4.80	54.7507	56.7448	0.03514
6.40, 9.60	64.4333	64.8269	0.006072

5.5.2 European two-asset maximum call option

Using Theorem 5.4.1, the option price of a European two-asset maximum call option can be estimated by the finite element method. Figure 5.4 shows how the option price depends on the initial stock prices. Assuming the initial stock prices are $S_1(0) = 52$ and $S_2(0) = 65$, the initial price of the option can be estimated to 16.0553 in about 21 seconds.

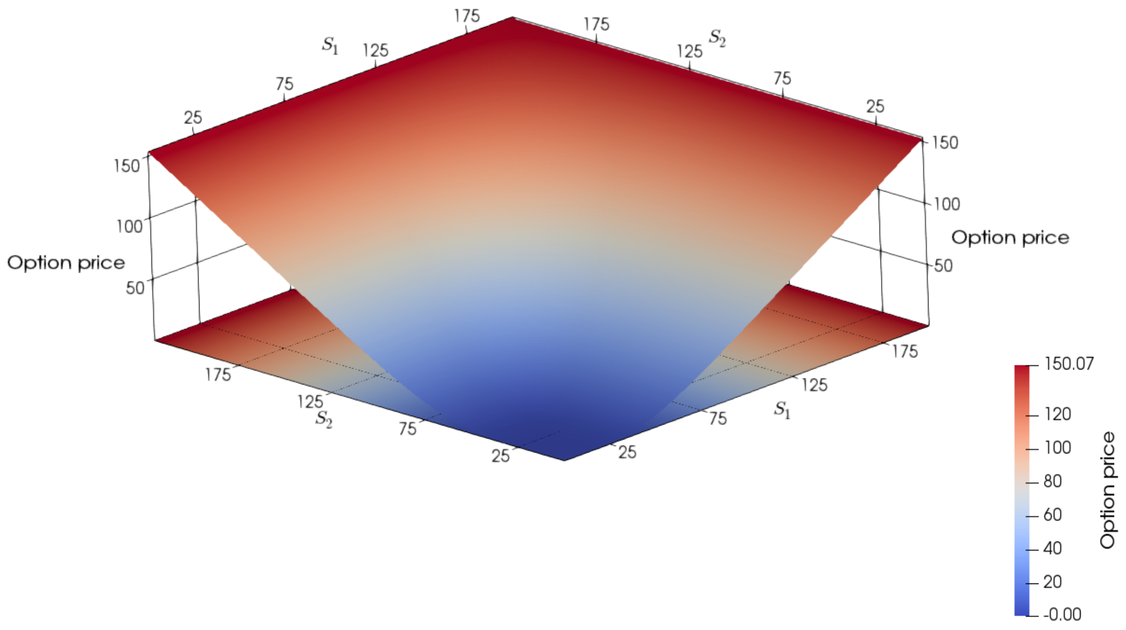


Figure 5.4: Option prices for the European two-asset maximum call option using the PDE method. Parameters: $K_1 = 50$, $K_2 = 70$, $T = 0.5$, $r = 0.1$, $\rho = 0.25$, $\sigma_1 = 0.60$, $\sigma_2 = 0.50$ and time step 0.01. The domain is set to $(0.1, 200) \times (0.1, 220)$ and a tolerance of 10^{-10} is used. Mesh size is 100×100.

Since the finite element method is dependent on the domain and its mesh, observing the dependence on the number of cells in each direction of the mesh is of great interest. Taking the previously used settings for the European two-asset maximum call option, the option price for different number of cells is observed in Figure 5.5.

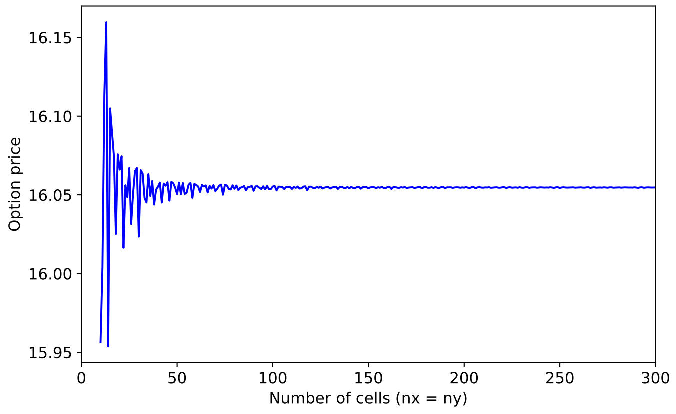


Figure 5.5: Option prices for the European two-asset maximum call option found by the PDE method using different number of cells $nx = ny$ in the mesh. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.5$, $r = 0.1$, $\rho = 0.25$, $\sigma_1 = 0.60$, $\sigma_2 = 0.50$ and time step 0.01. The domain is set to $(0.1, 200) \times (0.1, 220)$ and a tolerance of 10^{-10} is used.

It is not possible to use an exact reference price for the European maximum call option since this option cannot be priced by a closed formula. However, the estimates obtained by the PDE method can be compared to estimates from a long Monte Carlo simulation. This makes it possible to get an idea of how well the PDE method works. Moreover, the ”max-min parity” for the European maximum and minimum call option given by (2.52) can be used to check whether the result is accurate. The European minimum call option can be priced just like the European maximum call option but with a slight change of the boundary conditions. Both of these benchmark techniques are used in Tables 5.4 and 5.5 where the correlation ρ and the volatilities σ_1 and σ_2 vary.

Table 5.4: Dependence on the correlation ρ of the PDE method for the European two-asset maximum call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\sigma_1 = 0.60$ and $\sigma_2 = 0.50$. The domain is set to $(0.1, 200) \times (0.1, 220)$ with mesh size 100×100. A tolerance of 10^{-10} is used and the step length is 0.01. Long Monte Carlo simulations uses $n = 100{,}000{,}000$ and AV.

Correlation ρ	Method	Estimated price	Error "max-min" parity
-0.75	PDE	18.8468	0.002385
	MC with AV	18.8919	0.00001197
-0.50	PDE	18.3250	0.002385
	MC with AV	18.3674	0.00004377
-0.25	PDE	17.6935	00.002385
	MC with AV	17.7351	0.0000003274
0.25	PDE	16.0553	0.002385
	MC with AV	16.0913	0.00004187
0.50	PDE	14.9556	0.002386
	MC with AV	14.9901	0.00003023
0.75	PDE	13.4813	0.002386
	MC with AV	13.5112	0.00002032

Table 5.5: Dependence on the volatilities σ_1 and σ_2 of the PDE method of the European two-asset maximum call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$ and $\rho = 0.25$. The domain is set to $(0.1, 200) \times (0.1, 220)$ with mesh size 100×100. A tolerance of 10^{-10} is used and the step length is 0.01. Long Monte Carlo simulations uses $n = 100{,}000{,}000$ and AV.

Volatilities σ_1, σ_2	Method	Estimated price	Error "max-min" parity
0.15, 0.125	PDE	5.5446	0.0005520
	MC with AV	5.5467	0.000007845
0.30, 0.25	PDE	8.7477	0.001543
	MC with AV	8.7611	0.00001963
0.60, 0.50	PDE	16.0553	0.002385
	MC with AV	16.0919	0.00001038
1.20, 1.00	PDE	31.0655	0.002795
	MC with AV	31.4802	0.00003600
2.40, 2.00	PDE	55.5251	0.002869
	MC with AV	61.1464	0.0001712
4.80, 4.00	PDE	83.8711	0.001226
	MC with AV	100.2889	0.005022
9.60, 8.00	PDE	95.1856	0.00004707
	MC with AV	57.9031	0.5036

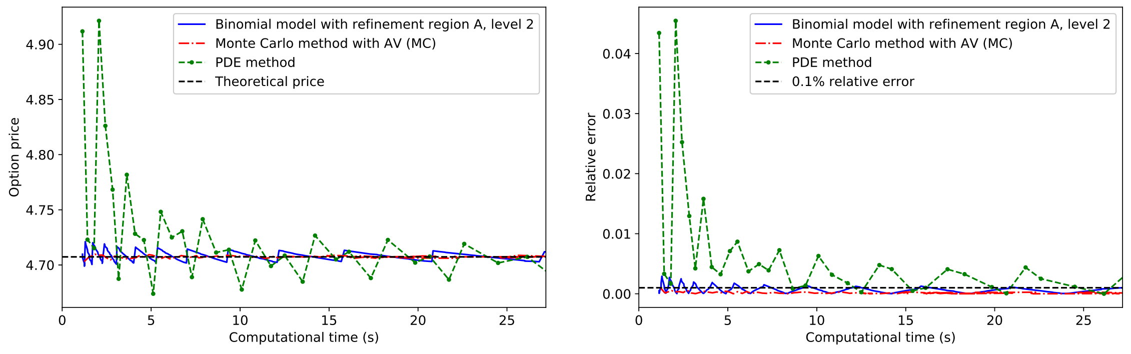
6
Conclusion

In this thesis, we have studied three different pricing methods for multi-asset options, namely the binomial pricing model, the Monte Carlo method, and the finite element method applied to the pricing PDE (the PDE method). Option prices, relative errors and computational times have been carefully studied as well as the dependence on the correlation and volatilities.

Below follows a comparison of the three previously presented methods based on the results presented earlier in the thesis. We also discuss the results and give suggestions for future research.

6.1 Comparison of the methods

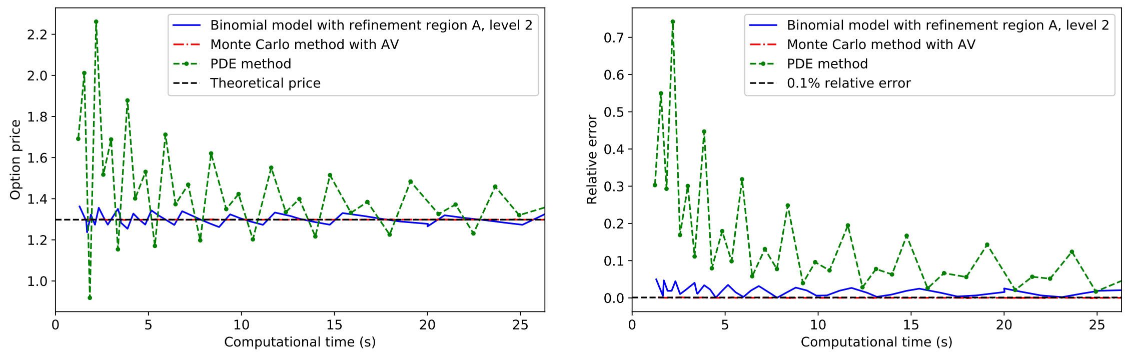
In order to compare the three pricing methods, it is beneficial to observe how the estimated option price depends on the computational time. Consider the standard European two-asset correlation call option used in the previous chapters with parameters $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$. The option price and the relative error for all three methods are shown in Figure 6.1 where the dependence on the computational times can be observed.



(a) Option price. **(b)** Relative error.

Figure 6.1: Option prices and relative errors using the different methods for the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$. Specifics for FEM: Time step 0.01, domain is set to $(0.1, 200) \times (0.1, 200)$ and a tolerance of 10^{-10} is used.

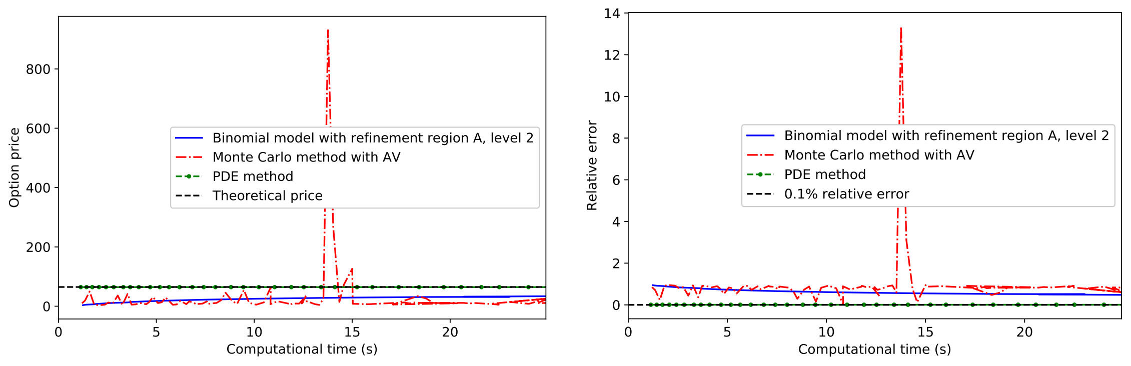
As in the previous chapters, it is also of interest to see how the results change with respect to the correlation. Such a comparison is shown in Figure 6.2. The same parameters as before is used but now the correlation is $\rho = -0.75$ instead of $\rho = 0.75$.



(a) Option price. **(b)** Relative error.

Figure 6.2: Option prices and relative errors using the different methods for the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = -0.75$, $\sigma_1 = 0.20$ and $\sigma_2 = 0.30$. Specifics for FEM: Time step 0.01, domain is set to $(0.1, 200) \times (0.1, 200)$ and a tolerance of 10^{-10} is used.

Further, previous results have indicated that the finite element method is favourable when observing higher values of the volatilities. This notion is verified in Figure 6.3 below where the volatilities are increased to $\sigma_1 = 6.40$, $\sigma_2 = 9.60$.



(a) Option price. **(b)** Relative error.

Figure 6.3: Option prices and relative errors using the different methods for the European two-asset correlation call option. Parameters: $S_1(0) = 52$, $S_2(0) = 65$, $K_1 = 50$, $K_2 = 70$, $T = 0.50$, $r = 0.10$, $\rho = 0.75$, $\sigma_1 = 6.40$ and $\sigma_2 = 9.60$. Specifics for FEM: Time step 0.01, domain is set to $(0.1, 200) \times (0.1, 200)$ and a tolerance of 10^{-10} is used.

6.2 Discussion

All of the pricing methods studied in this thesis have their respective advantages and disadvantages. We will now go through each of the methods and discuss what we have observed.

6.2.1 The binomial model

The binomial model has shown to be easy to implement for multi-asset options and its convergence rate is quite fast, which is illustrated in Figure 3.6. However, the major advantage of the binomial model is that it can easily be adapted to American options. None of the other pricing methods can handle American options that effortlessly.

It can also be concluded from Table 3.1 that the usage of refinement regions in the binomial model can improve the results significantly for the European two-asset correlation call option. It is unclear how well the refinement methods work for American options, but a slight decrease in the standard deviations of the estimates in Table 3.4 suggest that they could improve the result to some extent.

A disadvantage of the binomial model is that it performs badly for large volatilities and possibly also for some correlations. For example, the binomial model seems to underestimate the price of the European two-asset correlation call option when $\sigma_1 = 6.40$ and $\sigma_2 = 9.60$, as can be seen in Table 3.3. Another drawback of the binomial model is that the computational time seems to increase exponentially with the number of steps N, as illustrated in Figure 3.7.

6.2.2 The Monte Carlo method

The Monte Carlo method appears to be the fastest and most reliable out of the three methods when the volatilities are relatively small, as seen in Figure 6.1. It is also shown in Figures 4.2 and 4.3 that the computational time generally grows linearly with respect to the number of replications n, which makes the Monte Carlo method rather beneficial since the confidence intervals decrease as $1/\sqrt{n}$. The possibility to use the Monte Carlo method to price Asian options also basically makes it the only choice for such options. The binomial model cannot handle Asian options very well due to memory limitations and speed problems. Also it is very difficult to price multi-asset Asian options with the PDE method.

It has been shown that the usage of variance reduction techniques in the Monte Carlo method can significantly reduce the confidence intervals of the estimates. Especially the control variate method has shown to be very efficient for the Asian two-asset call option, as can be seen in Table 4.4. The second control variate method based on the antithetic variates had a much smaller impact on the confidence intervals and thus it was not as useful, especially considering the extra computational time that this method required.

A significant drawback of the Monte Carlo method is that it does not work well for large volatilities. The confidence intervals become very wide in that case and they can even include negative values (even though none of the price estimates are negative), as can be seen in Table 4.3. Another disadvantage is that the control variate method can eventually be difficult to implement since it is not always possible to find a suitable control variate.

6.2.3 The PDE method

The PDE method has resulted in the slowest convergence rate out of the three methods when the volatilities are low, as illustrated in Figure 6.1. However, when large volatilities are considered, this is the only method that appears to give a correct result. The other methods seem to underestimate the price in this case according to Figure 6.3.

Finding the optimal choice for the domain and the number of cells is not trivial for the PDE method since there is no guarantee that using a larger domain and more cells will give better results, as shown in Table 5.1. This suggests that it might be a good idea to compute estimated prices for a few different settings and see how the estimates vary. However, in general it seems like it is not beneficial to use a domain that is very large since the computational time increases linearly with the number of triangles in the domain, see Figure 5.3.

Compared to the other methods, the PDE method requires more adaption for each option because it is necessary to derive the boundary conditions by hand. Finding the correct boundary conditions and proving convergence can be troublesome, especially for more complicated options such as Asian options. It is also challenging to price American options with this method since it requires an analysis of the optimal exercise surface.

6.3 Future research

Throughout this thesis it has been assumed that the underlying assets follow a multidimensional geometric Brownian motion with constant coefficients. Future research could focus on implementing some stochastic volatility model where the drift rates and volatilities are random processes. This would make the computations more complicated but also more adapted to the real world.

More research is also needed about the convergence rate for the PDE method. It would be interesting to see if the finite element method could be improved in some way, perhaps by using an adaptive mesh. How to specify the boundary conditions for multi-asset options would also be an interesting area of research.

A
The Trinomial Model

Consider two stocks described by Representation 1 (2.20), i.e.,

$$S_1(t) = S_1(0)e^{\alpha_1 t+\sigma_{11}W_1(t)+\sigma_{12}W_2(t)}, \tag{A.1}$$
$$S_2(t) = S_2(0)e^{\alpha_2 t+\sigma_{21}W_1(t)+\sigma_{22}W_2(t)}, \tag{A.2}$$

where

$$\alpha_1 = \mu_1 - \frac{\sigma_{11}^2+\sigma_{12}^2}{2}, \tag{A.3}$$
$$\alpha_2 = \mu_2 - \frac{\sigma_{21}^2+\sigma_{22}^2}{2}. \tag{A.4}$$

Let $\{X_i^{(1)}\}_{i\in\mathbb{N}}$ and $\{X_i^{(2)}\}_{i\in\mathbb{N}}$ be two independent and identically distributed (iid) stochastic processes satisfying

$$X_i^{(j)} = \begin{cases} 1 & \text{with probability } p_u \\ 0 & \text{with probability } p_m \\ -1 & \text{with probability } p_d \end{cases} \tag{A.5}$$

for $j = 1,2$. The physical probabilities p_u, p_m, p_d are defined such that

$$\mathbb{E}\left[X_i^{(j)}\right] = 0 \implies p_u = p_d = p, \tag{A.6}$$

which means that

$$\operatorname{Var}\left[X_i^{(j)}\right] = \mathbb{E}\left[\left(X_i^{(j)}\right)^2\right] = p_u + p_d = 2p \tag{A.7}$$

for $1 = 1,\dots,N$ and $j = 1,2$. Let $t_0 = 0 < t_1 < \cdots < t_N = t$ and take $t_{i+1} - t_i = h$ with $N = t/h$. Define

$$\widetilde{S}_1(t_i) = \widetilde{S}_1(t_{i-1})e^{\widetilde{\alpha}_1+\widetilde{\sigma}_{11}X_i^{(1)}+\widetilde{\sigma}_{12}X_i^{(2)}}, \tag{A.8}$$
$$\widetilde{S}_2(t_i) = \widetilde{S}_2(t_{i-1})e^{\widetilde{\alpha}_2+\widetilde{\sigma}_{21}X_1^{(1)}+\widetilde{\sigma}_{22}X_i^{(2)}}, \tag{A.9}$$

which implies that

$$\widetilde{S}_1(t) = \widetilde{S}_1(0)e^{\widetilde{\alpha}_1 N+\widetilde{\sigma}_{11}M_N^{(1)}+\widetilde{\sigma}_{12}M_N^{(2)}}, \tag{A.10}$$
$$\widetilde{S}_2(t) = \widetilde{S}_2(0)e^{\widetilde{\alpha}_2 N+\widetilde{\sigma}_{21}M_N^{(1)}+\widetilde{\sigma}_{22}M_N^{(2)}}, \tag{A.11}$$

where

$$M_N^{(j)} = \sum_{i=0}^{N} X_i^{(j)}. \tag{A.12}$$

The goal is to specify the parameters $\widetilde{\alpha}_1$, $\widetilde{\alpha}_2$, $\widetilde{\sigma}_{11}$, $\widetilde{\sigma}_{12}$, $\widetilde{\sigma}_{21}$, and $\widetilde{\sigma}_{22}$ so that (A.10) and (A.11) converge to (A.1) respectively (A.2) when N goes to infinity. Observe that

$$\frac{1}{t}\mathbb{E}\left[\log \frac{S_1(t)}{S_1(0)}\right] = \alpha_1, \tag{A.13}$$

$$\frac{1}{t}\mathbb{E}\left[\log \frac{S_2(t)}{S_2(0)}\right] = \alpha_2, \tag{A.14}$$

and similarly

$$\frac{1}{t}\mathbb{E}\left[\log \frac{\widetilde{S}_1(t)}{\widetilde{S}_1(0)}\right] = \frac{\widetilde{\alpha}_1}{h}, \tag{A.15}$$

$$\frac{1}{t}\mathbb{E}\left[\log \frac{\widetilde{S}_2(t)}{\widetilde{S}_2(0)}\right] = \frac{\widetilde{\alpha}_2}{h}. \tag{A.16}$$

This means that

$$\widetilde{\alpha}_1 = h\alpha_1, \tag{A.17}$$

$$\widetilde{\alpha}_2 = h\alpha_2. \tag{A.18}$$

Furthermore, note that

$$\frac{1}{t}\mathrm{Var}\left[\left(\log \frac{S_1(t)}{S_1(0)}\right)\right] = \sigma_{11}^2 + \sigma_{12}^2, \tag{A.19}$$

$$\frac{1}{t}\mathrm{Var}\left[\left(\log \frac{S_2(t)}{S_2(0)}\right)\right] = \sigma_{21}^2 + \sigma_{22}^2, \tag{A.20}$$

and

$$\frac{1}{t}\mathrm{Cov}\left(\log \frac{S_1(t)}{S_1(0)}, \log \frac{S_2(t)}{S_2(0)}\right) = \sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}. \tag{A.21}$$

Similarly

$$\frac{1}{t}\mathrm{Var}\left[\left(\log \frac{\widetilde{S}_1(t)}{\widetilde{S}_1(0)}\right)\right] = \frac{1}{t}\left(\widetilde{\sigma}_{11}^2 \mathrm{Var}(M_N^{(1)}) + \widetilde{\sigma}_{12}^2 \mathrm{Var}(M_N^{(2)})\right) \tag{A.22}$$

$$= \frac{2p}{t}\left(\widetilde{\sigma}_{11}^2 N + \widetilde{\sigma}_{12}^2 N\right) \tag{A.23}$$

$$= \frac{2p}{h}\left(\widetilde{\sigma}_{11}^2 + \widetilde{\sigma}_{12}^2\right), \tag{A.24}$$

$$\frac{1}{t}\mathrm{Var}\left[\left(\log \frac{\widetilde{S}_2(t)}{\widetilde{S}_2(0)}\right)\right] = \frac{1}{t}\left(\widetilde{\sigma}_{21}^2 \mathrm{Var}(M_N^{(1)}) + \widetilde{\sigma}_{22}^2 \mathrm{Var}(M_N^{(2)})\right) \tag{A.25}$$

$$= \frac{2p}{t}\left(\widetilde{\sigma}_{21}^2 N + \widetilde{\sigma}_{22}^2 N\right) \tag{A.26}$$

$$= \frac{2p}{h}\left(\widetilde{\sigma}_{21}^2 + \widetilde{\sigma}_{22}^2\right). \tag{A.27}$$

and

$$\frac{1}{t}\mathrm{Cov}\left(\log\frac{\widetilde{S}_1(t)}{\widetilde{S}_1(0)}, \log\frac{\widetilde{S}_2(t)}{\widetilde{S}_2(0)}\right) = \frac{2p}{t}N(\widetilde{\sigma}_{11}\widetilde{\sigma}_{21} + \widetilde{\sigma}_{12}\widetilde{\sigma}_{22}) \tag{A.28}$$

$$= \frac{2p}{h}(\widetilde{\sigma}_{11}\widetilde{\sigma}_{21} + \widetilde{\sigma}_{12}\widetilde{\sigma}_{22}). \tag{A.29}$$

Setting the corresponding variances and covariances to be equal yields the following set of equations:

$$\sigma_{11}^2 + \sigma_{12}^2 = \frac{2p}{h}(\widetilde{\sigma}_{11}^2 + \widetilde{\sigma}_{12}^2), \tag{A.30}$$

$$\sigma_{21}^2 + \sigma_{22}^2 = \frac{2p}{h}(\widetilde{\sigma}_{21}^2 + \widetilde{\sigma}_{22}^2), \tag{A.31}$$

$$\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22} = \frac{2p}{h}(\widetilde{\sigma}_{11}\widetilde{\sigma}_{21} + \widetilde{\sigma}_{12}\widetilde{\sigma}_{22}), \tag{A.32}$$

which has a solution

$$\widetilde{\sigma}_{11} = \sqrt{\frac{h}{2p}}\sigma_{11}, \quad \widetilde{\sigma}_{12} = \sqrt{\frac{h}{2p}}\sigma_{12}, \tag{A.33}$$

$$\widetilde{\sigma}_{21} = \sqrt{\frac{h}{2p}}\sigma_{21}, \quad \widetilde{\sigma}_{22} = \sqrt{\frac{h}{2p}}\sigma_{22}. \tag{A.34}$$

Denote the risk-neutral probabilities in the trinomial model by q_u, q_m, and q_d. The following must hold:

$$q_u + q_m + q_d = 1. \tag{A.35}$$

Since the discounted stock prices are martingales under a risk-neutral measure, it holds that

$$\mathbb{E}\left[\frac{\widetilde{S}_1(t_k)}{\widetilde{S}_1(t_{k-1})}\middle| X_{k-1}^{(1)}, X_{k-1}^{(2)}\right] = e^r. \tag{A.36}$$

This yields

$$\begin{aligned} &e^{\widetilde{\sigma}_{11}+\widetilde{\sigma}_{12}}q_u^2 + e^{\widetilde{\sigma}_{11}}q_uq_m + e^{\widetilde{\sigma}_{11}-\widetilde{\sigma}_{12}}q_uq_d \\ &+ e^{\widetilde{\sigma}_{12}}q_mq_u + q_m^2 + e^{-\widetilde{\sigma}_{12}}q_mq_d \\ &+ e^{-\widetilde{\sigma}_{11}+\widetilde{\sigma}_{12}}q_dq_u + e^{-\widetilde{\sigma}_{11}}q_dq_m + e^{-\widetilde{\sigma}_{11}-\widetilde{\sigma}_{12}}q_d^2 = e^{r-\widetilde{\alpha}_1}. \end{aligned} \tag{A.37}$$

Similarly, for the second asset it holds that

$$\mathbb{E}\left[\frac{\widetilde{S}_1(t_k)}{\widetilde{S}_2(t_{k-1})}\middle| X_{k-1}^{(1)}, X_{k-1}^{(2)}\right] = e^r, \tag{A.38}$$

which yields

$$\begin{aligned} &e^{\widetilde{\sigma}_{21}+\widetilde{\sigma}_{22}}q_u^2 + e^{\widetilde{\sigma}_{21}}q_uq_m + e^{\widetilde{\sigma}_{21}-\widetilde{\sigma}_{22}}q_uq_d \\ &+ e^{\widetilde{\sigma}_{22}}q_mq_u + q_m^2 + e^{-\widetilde{\sigma}_{22}}q_mq_d \\ &+ e^{-\widetilde{\sigma}_{21}+\widetilde{\sigma}_{22}}q_dq_u + e^{-\widetilde{\sigma}_{21}}q_dq_m + e^{-\widetilde{\sigma}_{21}-\widetilde{\sigma}_{22}}q_d^2 = e^{r-\widetilde{\alpha}_2}. \end{aligned} \tag{A.39}$$

Inputting (A.37) and (A.39) together with the condition (A.33) into some mathematical software, such as Mathematica, yields that it is not always possible to find valid values of the parameters q_u, q_m and q_d. This means that a risk-neutral measure only exists for certain market parameters. Thus it follows from the two fundamental theorems of asset pricing that this trinomial market model is neither arbitrage-free nor complete [2].

B
Python Code: Black-Scholes Price of European Vanilla Options

```python
"""
Black_Scholes computes the Black-Scholes price of a vanilla European
call or put option.

Input parameters are cp = 1 for call and cp = -1 for put, initial stock
price S_0, strike price K, time until maturity t, volatility sigma,
risk-free interest rate r, and continuous dividend yield div.

The function returns the initial option price.
"""

import math
import scipy.stats as st

def Black_Scholes(cp, S_0, K, t, sigma, r, div):
        d1 = (math.log(S_0/K)+(r-div+0.5*math.pow(sigma,2))*t)/(sigma*math.sqrt(t))
        d2 = d1 - sigma*math.sqrt(t)

        option_price = (cp*S_0*math.exp(-div*t)*st.norm.cdf(cp*d1)) - (cp*K*math.exp(-r*t)*st.norm.cdf(cp*d2))
        return option_price
```

C

Python Code: Exact Price of a European Two-asset Correlation Call Option

```python
"""
exact_price_European_two_asset_corr_call computes the exact price
of an European two-asset correlation call option.

Input parameters are the initial asset prices S_0, volatilities sigma,
risk-free interest rate r, correlation rho, time until maturity T, and
strike prices K.

Returns the exact price as option_price.

Note: Uses a help-function bivariate_normal_pdf which returns the
probability denisity function (pdf) of the bivariate normal
distribution with parameters x, y and rho.
"""

import numpy as np
from scipy import integrate

def bivariate_normal_pdf(x, y, rho):
    return (1/(2*np.pi*np.sqrt(1-np.power(rho,2))))*np.exp(-1/(2*(1-np.power(rho,2)))*(np.power(x,2)-2*rho*x*y+np.power(y,2))))

def exact_price_European_two_asset_corr_call(S_0, sigma, r, rho, T, K):
    y_1 = (np.log(S_0[0]/K[0]) + T*(r-np.power(sigma[0], 2)/2))/(sigma[0]*np.sqrt(T))
    y_2 = (np.log(S_0[1]/K[1]) + T*(r-np.power(sigma[1], 2)/2))/(sigma[1]*np.sqrt(T))
    M_1 = integrate.nquad(bivariate_normal_pdf, [[-np.inf, y_1+rho*sigma[1]*np.sqrt(T)],[-np.inf, y_2+sigma[1]*np.sqrt(T)]], args=[rho])
    M_2 = integrate.nquad(bivariate_normal_pdf, [[-np.inf, y_1],[-np.inf,y_2]], args=([rho]))
    option_price = S_0[1]*M_1[0]-K[1]*np.exp(-r*T)*M_2[0]
    return option_price
```

D
Python Code: The Binomial Pricing Model

D.1 Log asset price

```python
"""
log_asset_price computes the log asset prices in the binomial
model at the time of maturity.

Input parameters are initial log asset price X_0, volatilities sigma,
number of intervals N and time until maturity T.

Returns a 3 dimensional matrix X_T of size (N+1)x(N+1)x2 with log
asset prices at time T.
"""

import numpy as np

def log_asset_price(X_0, sigma, N, T):
    dt = T/N
    h = np.sqrt(dt)*sigma
    X_T = np.zeros((N+1, N+1, 2))
    for column in range(0, N+1):
        for row in range(0, N+1):
            for element in range (0, 2):
                if element == 0:
                    X_T[row][column][element] = X_0[element] + h[element]*(-N+2*column)
                if element == 1:
                    X_T[row][column][element] = X_0[element] + h[element]*(N-2*row)
    return X_T
```

D.2 European two-asset correlation call option

D.2.1 Payoff

```
"""
payoff_European_two_asset_corr_call computes all possible payoffs of
a European two asset correlation option at a given time.

Input parameters are log asset prices log_prices, risk-free interest
rate r, volatilities sigma, strike prices K and time of which the
payoffs is seeked T.

Returns a matrix payoff_T of same size as the matrix log_prices with
all possible payoffs at time T.
"""

import numpy as np

def payoff_European_two_asset_corr_call(log_prices, r, sigma, K, T):
    [dim_row, dim_column, dim_element] = log_prices.shape
    payoff_T = np.zeros((dim_row, dim_column))
    for column in range(0, dim_column):
        for row in range(0, dim_row):
            X_1 = log_prices[row][column][0]
            S_1 = np.exp(X_1 +(r-np.power(sigma[0], 2)/2)*T)
            if S1 > K[0]:
                X_2 = log_prices[row][column][1]
                S_2 = np.exp(X_2+(r-np.power(sigma[1], 2)/2)*T)
                payoff_T[row][column] = max(S_2-K[1], 0)
    return payoff_T
```

D.2.2 Initial option price

```python
"""
initial_price_European_two_asset_corr_call computes the initial
option price of a European two-asset correlation call option.

Input parameters are the payoff at maturity option_value,
correlation rho, risk-free interest rate r, number of time
steps N, and time until maturity T.

Returns the initial option price in option_value[0][0].
"""

import numpy as np

def initial_price_European_two_asset_corr_call(option_value, rho, r, N, T):
    dt = T/N
    [dim_row, dim_column] = option_value.shape
    P_uu = P_dd = 1/4*(1+rho)
    P_ud = P_du = 1/4*(1-rho)
    while dim_row > 1:
        discounted_option_value = np.zeros((dim_row-1, dim_column-1))
        for row in range(0, dim_row-1):
            for column in range(0, dim_column-1):
                v_uu = option_value[row][column+1]
                v_dd = option_value[row+1][column]
                v_du = option_value[row][column]
                v_ud = option_value[row+1][column+1]
                discounted_option_value[row][column] = np.exp(-r*dt)*( v_uu*P_uu + v_ud*P_ud + v_du*P_du + v_dd*P_dd )
        return initial_price_European_two_asset_corr_call(discounted_option_value, rho, r, N, T)
    else :
        return option_value[0][0]
```

D.2.3 The binomial pricing model using a uniform mesh

```python
"""
binomial_uniform_European_two_asset_corr_call computes the intital
option price of a European two-asset correlation call option.
It uses the binomial model and a uniform mesh.

Input parameters are the initial stock prices S_0, volatilities sigma,
risk-free interest rate r, correlation rho, time of maturity T,
number of time steps N, and strike prices K.

The function returns the initial option price option_price_0.
"""

import numpy as np

def binomial_uniform_European_two_asset_corr_call(S_0, sigma, r, rho, T
    , N, K):
    log_price_T = log_asset_price(np.log(S_0), sigma, N, T)
    payoff_T = payoff_European_two_asset_corr_call(log_price_T, r,
    sigma, K, T)
    option_price_0 = initial_price_European_two_asset_corr_call(
    payoff_T, rho, r, N, T)
    return option_price_0
```

D.2.4 The binomial pricing model using refinement region

```
"""
binomial_refinement_European_two_asset_corr_call calculates the
initial option price of a European two-asset correlation option,
adding a finer mesh over the final time step T-dt.

Input parameters are initial stock prices S_0, volatilities sigma,
risk-free interest rate r, correlation rho, time until maturity T,
number of time steps N, strike prices K and refinement level RL.

Returns the initial option value option_value_0.
"""

import numpy as np

def binomial_refinement_European_two_asset_corr_call(S_0, sigma, r, rho
    , T, N, K, RL):
    dt = T/N
    h = np.sqrt(dt)*sigma

    # Compute log asset price at time T-dt
    X_T_minus_dt = log_asset_price(np.log(S_0), sigma, N-1, T-dt)

    # Find refinement region
    j_min = 0
    while X_T_minus_dt[0][j_min][0]+np.power(2,RL)*h[0] < np.log(K[0])
    -(r-np.power(sigma[0],2)/2)*T:
        j_min = j_min+1
        if j_min > N-1:
            break

    j_max = N-1
    while X_T_minus_dt[0][j_max][0]-np.power(2,RL)*h[0] >= np.log(K[0])
    -(r-np.power(sigma[0],2)/2)*T:
        j_max = j_max-1
        if j_max < 0:
            j_max = 0
            break

    i_max = N-1
    while X_T_minus_dt[i_max][0][1]+np.power(2,RL)*h[1] < np.log(K[1])
    -(r-np.power(sigma[1],2)/2)*T:
        i_max = i_max-1
        if i_max < 0:
            break

    i_min = 0
    while X_T_minus_dt[i_min][0][1]-np.power(2,RL)*h[1] >= np.log(K[1])
    -(r-np.power(sigma[1],2)/2)*T:
        i_min = i_min+1
        if i_min >= N:
            i_min = N
            break

```

```python
    # Compute option value in coarse mesh nodes and its complement
    option_value_T_minus_dt = np.zeros((N, N))
    for row in range(0, N):
        for column in range(0, N):
            # For refinement region A use condition:
            # (i_min <= row <= i_max and j_min <= column) or (row <= i_max and j_min <= column <= j_max):
            # For refinement region B use condition:
            # (i_min <= row <= i_max and j_min <= column <= j_max):
            # For refinement region C use condition:
            # (row <= i_max and j_min <= column):

            # Compute discounted option value using refinement region A:
            if (i_min <= row <= i_max and j_min <= column) or (row <= i_max and j_min <= column <= j_max):
                X_RF = log_asset_price(X_T_minus_dt[row][column], sigma, np.power(4,RL), dt)
                payoff_RF = payoff_European_two_asset_corr_call(X_RF, r, sigma, K, T)
                option_value_T_minus_dt[row][column] = initial_price_European_two_asset_corr_call(payoff_RF, rho, r, np.power(4,RL), dt)
            else:
                X_RF_comp = log_asset_price(X_T_minus_dt[row][column], sigma, 1, dt)
                payoff_RF_comp = payoff_European_two_asset_corr_call(X_RF_comp, r, sigma, K, T)
                option_value_T_minus_dt[row][column] = initial_price_European_two_asset_corr_call(payoff_RF_comp, rho, r, 1, dt)

    # Compute initial option price
    option_value_0 = initial_price_European_two_asset_corr_call(option_value_T_minus_dt, rho, r, N-1, T-dt)
    return option_value_0
```

D.3 American two-asset correlation put option

D.3.1 Payoff

```python
"""
payoff_American_two_asset_corr_put computes all possible payoffs of
an American two-asset correlation put option at a given time.

Input parameters are log asset prices log_prices, risk-free interest
rate r, volatilities sigma, time T, and strike prices K.

Returns a matrix payoff_T of the same size as the matrix log_prices
with all possible payoffs at time T.
"""

import numpy as np

def payoff_American_two_asset_corr_put(log_prices, r, sigma, K, T):
    [dim_row, dim_column, dim_element] = log_prices.shape
    payoff_T = np.zeros((dim_row, dim_column))
    for column in range(0, dim_column):
        for row in range(0, dim_row):
            X_1 = log_prices[row][column][0]
            S_1 = np.exp(X_1+(r-np.power(sigma[0], 2)/2)*T)
            if S_1 < K[0]:
                X_2 = log_prices[row][column][1]
                S_2 = np.exp(X_2+(r-np.power(sigma[1], 2)/2)*T)
                payoff_T[row][column] = max(K[1]-S_2,0)
    return payoff_T
```

D.3.2 Initial option price

```
"""
initial_price_American_two_asset_corr_put computes the initial
option price of an American two-asset correlation put option.

Input parameters are the payoff at maturity option_value, log asset
prices at the initial time X_0, correaltion rho, risk-free interest
rate r, number of time steps N, initial time t_0, time of maturity T,
strike price K, and volatility sigma.

Returns the option price at t_0 in option_value[0][0].
"""

import numpy as np

def initial_price_American_two_asset_corr_put(option_value, X_0, rho, r
    , N, t_0, T, K, sigma):
    dt = (T-t_0)/N
    [dim_row, dim_column] = option_value.shape
    P_uu = 1/4*(1+rho)
    P_dd = P_uu
    P_ud = 1/4*(1-rho)
    P_du = P_ud

    if dim_row > 1:
        log_asset_prices = log_asset_price(X_0,sigma,dim_row-2,(dim_row
    -2)*dt)
        discounted_option_value = np.zeros((dim_row-1, dim_column-1))
        for row in range(0, dim_row-1):
            for column in range(0, dim_column-1):
                v_uu = option_value[row][column+1]
                v_dd = option_value[row+1][column]
                v_du = option_value[row][column]
                v_ud = option_value[row+1][column+1]

                X_1 = log_asset_prices[row][column][0]
                S_1 = np.exp(X_1+(r-np.power(sigma[0], 2)/2)*(t_0+dt*(
    dim_row-2)))
                X_2 = log_asset_prices[row][column][1]
                S_2 = np.exp(X_2+(r-np.power(sigma[1], 2)/2)*(t_0+dt*(
    dim_row-2)))

                discounted_option_value[row][column] = max(np.exp(-r*dt
    )*(v_uu*P_uu + v_ud*P_ud + v_du*P_du + v_dd*P_dd), max(K[1]-S_2,0)
    *(S_1<K[0]))
        return initial_price_American_two_asset_corr_put(
    discounted_option_value,X_0, rho, r, N, t_0, T, K, sigma)
    else :
        return option_value[0][0]
```

D.3.3 The binomial pricing model using a uniform mesh

```
"""
binomial_uniform_American_two_asset_corr_put computes the initial
option price of an American two−asset correlation put option. It uses
the binomial model and a uniform mesh.

Input parameters are the initial stock prices S_0, volatilities sigma,
risk−free interest rate r, correlation rho, time of maturity T, number
of time steps N, and strike prices K.

The function returns the initial option price option_price_0.
"""

import numpy as np

def binomial_uniform_American_two_asset_corr_put(S_0, sigma, r, rho, T,
    N, K):
    log_price_T = log_asset_price(np.log(S_0), sigma, N, T)
    payoff_T =  payoff_American_two_asset_corr_put(log_price_T, r,
   sigma, K, T)
    option_price_0 = initial_price_American_two_asset_corr_put(payoff_T
    , np.log(S_0), rho, r, N, 0, T, K, sigma)
    return option_price_0
```

D.3.4 The binomial pricing model using refinement region

```python
"""
binomial_refinement_American_two_asset_corr_put calculates the
initial option price of an American two-asset correlation put
option, adding a finer mesh over the final time step T-dt.

Input parameters are initial stock prices S_0, volatilities sigma,
risk-free interest rate r, correlation rho, time until maturity T,
number of time steps N, strike prces K, and refinement level RL.

Returns the initial option value option_value_0
"""

import numpy as np

def binomial_refinement_American_two_asset_corr_put(S_0, sigma, r, rho,
     T, N, K, RL):
    dt = T/N
    h = np.sqrt(dt)*sigma

    # Compute log asset prices at time T-dt
    X_T_minus_dt = log_asset_price(np.log(S_0), sigma, N-1, T-dt)

    # Find refinement region
    j_min = 0
    while X_T_minus_dt[0][j_min][0]+np.power(2,RL)*h[0] < np.log(K[0])
   -(r-np.power(sigma[0],2)/2)*T:
        j_min = j_min+1
        if j_min > N-1:
            break

    j_max = N-1
    while X_T_minus_dt[0][j_max][0]-np.power(2,RL)*h[0] >= np.log(K[0])
   -(r-np.power(sigma[0],2)/2)*T:
        j_max = j_max-1
        if j_max < 0:
            j_max = 0
            break

    i_max = N-1
    while X_T_minus_dt[i_max][0][1]+np.power(2,RL)*h[1] < np.log(K[1])
   -(r-np.power(sigma[1],2)/2)*T:
        i_max = i_max-1
        if i_max < 0:
            break

    i_min = 0
    while X_T_minus_dt[i_min][0][1]-np.power(2,RL)*h[1] >= np.log(K[1])
   -(r-np.power(sigma[1],2)/2)*T:
        i_min = i_min+1
        if i_min >= N:
            i_min = N
            break

    # Compute option value in coarse mesh nodes and its complement
    option_value_T_minus_dt = np.zeros((N, N))
    for row in range(0, N):
        for column in range(0, N):
            if (i_min <= row <= i_max and column <= j_max) or (row >= i_min and j_min <= column <= j_max):
                X_RF = log_asset_price(X_T_minus_dt[row][column], sigma, np.power(4,RL), dt)
                payoff_Rf = payoff_American_two_asset_corr_put(X_RF, r, sigma, K, T)
                option_value_T_minus_dt[row][column] = initial_price_American_two_asset_corr_put(payoff_Rf, X_T_minus_dt[row][column], rho, r, np.power(4,RL), T-dt, T, K, sigma)
            else:
                X_RF_comp = log_asset_price(X_T_minus_dt[row][column], sigma, 1, dt)
                payoff_Rf_comp = payoff_American_two_asset_corr_put(X_RF_comp, r, sigma, K, T)
                option_value_T_minus_dt[row][column] = initial_price_American_two_asset_corr_put(payoff_Rf_comp, X_T_minus_dt[row][column], rho, r, 1, T-dt, T, K, sigma)
    # Compute initial option price
    option_value_0 = initial_price_American_two_asset_corr_put(option_value_T_minus_dt, np.log(S_0), rho, r, N-1, 0, T-dt, K, sigma)
    return option_value_0
```

E

Python Code: The Monte Carlo Method

E.1 European two-asset correlation call option

```
"""
MC_European_two_asset_corr_call computes the initial price of a
European two-asset correlation call option by using Monte Carlo
(MC) simulations.

The input parameters are the strike price K, the risk-free interest
rate r, the initial stock prices S_0, the volatility matrix sigma,
the time until maturity T, and the number of replications n.

The output is the estimated prices from the standard MC method and
the MC method with anithetic variates (AV). The function also returns
95% confidence intervals (CIs) corresponding to both of the price
estiamtes.
"""

import numpy as np
import scipy.stats as st
import math

def MC_European_two_asset_corr_call(K, r, S_0, sigma, T, n):

    # Generate value of Brownian motions at time of maturity T
    W_1_T = np.random.normal(0, np.sqrt(T), n)
    W_2_T = np.random.normal(0, np.sqrt(T), n)
    W_1_AV_T = -W_1_T
    W_2_AV_T = -W_2_T

    # Compute final stock prices
    S_1_T = S_0[0]*np.exp((r-(sigma[0,0]**2+sigma[0,1]**2)/2)*T+sigma
    [0,0]*W_1_T+sigma[0,1]*W_2_T)
    S_2_T = S_0[1]*np.exp((r-(sigma[1,0]**2+sigma[1,1]**2)/2)*T+sigma
    [1,0]*W_1_T+sigma[1,1]*W_2_T)

    S_1_AV_T = S_0[0]*np.exp((r-(sigma[0,0]**2+sigma[0,1]**2)/2)*T+
    sigma[0,0]*W_1_AV_T+sigma[0,1]*W_2_AV_T)
    S_2_AV_T = S_0[1]*np.exp((r-(sigma[1,0]**2+sigma[1,1]**2)/2)*T+
    sigma[1,0]*W_1_AV_T+sigma[1,1]*W_2_AV_T)

    # Compute discounted standard payoff
    Dpayoff_standard = np.exp(-r*T)*np.maximum(S_2_T-K[1],0)*(S_1_T>K
    [0])

    # Compute discounted payoff with antithetic variates
    Dpayoff_AV = np.exp(-r*T)*np.maximum(S_2_AV_T-K[1],0)*(S_1_AV_T>K
    [0])

    # Compute standard price and CI
    price_standard = np.mean(Dpayoff_standard)
    CI_standard = st.t.interval(0.95, len(Dpayoff_standard)-1, loc=
    price_standard , scale=st.sem(Dpayoff_standard))

    # Compute AV price and CI
    price_AV = np.mean((Dpayoff_standard+Dpayoff_AV)/2)
    CI_AV = st.t.interval(0.95, len(Dpayoff_AV)-1, loc=price_AV, scale=
    st.sem((Dpayoff_standard+Dpayoff_AV)/2))

    return [price_standard, price_AV, CI_standard, CI_AV]
```

E.2 Asian two-asset call option

```python
"""
MC_Asian_two_asset_call computes the initial price of an Asian
two-asset call option by using Monte Carlo (MC) simulations.

The input parameters are the strike price K, the risk-free interest
rate r, the initial stock prices S_0, the volatility matrix sigma,
the time until maturity T, the weights alpha_1 and alpha_2 of the
stocks, the number of partitions N, and the number of replications n.

The output is the estimated prices from the standard MC method, the
MC method with anithetic variates (AV), and the MC method with control
variates (CV). The function also returns 95% confidence intervals (CIs)
corresponding to all of the price estiamtes.
"""

import numpy as np
import scipy.stats as st
import math
import Black_Scholes as BS

def MC_Asian_two_asset_call(K, r, S_0, sigma, T, weights, N, n):

    payoff_standard = np.zeros(n)
    payoff_AV = np.zeros(n)
    payoff_geometric = np.zeros(n)

    dt = T/N

    for i in range (0, n):
        # Create numpy arrays for holding asset prices at different
    time
        # instances
        S_1_t = np.zeros(N+1)
        S_2_t = np.zeros(N+1)
        S_1_t_AV = np.zeros(N+1)
        S_2_t_AV = np.zeros(N+1)

        # Specify the initial asset prices
        S_1_t[0] = S_0[0]
        S_2_t[0] = S_0[1]
        S_1_t_AV[0] = S_0[0]
        S_2_t_AV[0] = S_0[1]

        # Generate steps in the Brownian motions
        dW_1 = np.zeros(N+1)
        dW_2 = np.zeros(N+1)

        dW_1[1:] = np.random.normal(0, np.sqrt(dt), N)
        dW_2[1:] = np.random.normal(0, np.sqrt(dt), N)

        # Generate steps in the Brownian motions used for the
    antithetic
        # variates method
        dW_1_AV = -dW_1
        dW_2_AV = -dW_2

        # Create arrays with the cumulative values of the Brownian motions
        W_1 = np.cumsum(dW_1)
        W_2 = np.cumsum(dW_2)
        W_1_AV = np.cumsum(dW_1_AV)
        W_2_AV = np.cumsum(dW_2_AV)

        t = np.linspace(0, T, num = N+1)

        # Compute asset prices at different time instances
        S_1_t = S_0[0]*np.exp((r-(sigma[0,0]**2+sigma[0,1]**2)/2)*t+sigma[0,0]*W_1+sigma[0,1]*W_2)
        S_2_t = S_0[1]*np.exp((r-(sigma[1,0]**2+sigma[1,1]**2)/2)*t+sigma[1,0]*W_1+sigma[1,1]*W_2)

        # Compute asset prices at different time instances for the
        # antithetic variates method
        S_1_t_AV = S_0[0]*np.exp((r-(sigma[0,0]**2+sigma[0,1]**2)/2)*t+sigma[0,0]*W_1_AV+sigma[0,1]*W_2_AV)
        S_2_t_AV = S_0[1]*np.exp((r-(sigma[1,0]**2+sigma[1,1]**2)/2)*t+sigma[1,0]*W_1_AV+sigma[1,1]*W_2_AV)

        # Assemble asset prices
        S = np.matrix((S_1_t, S_2_t))
        S_AV = np.matrix((S_1_t_AV, S_2_t_AV))

        # Multiply asset prices by weights
        S_weighted_standard = np.transpose(np.multiply(np.transpose(S), weights))
        S_weighted_AV = np.transpose(np.multiply(np.transpose(S_AV), weights))
        S_weighted_geometric = np.transpose(np.power(np.transpose(S), weights))

        payoff_standard[i] = max(1/(N+1)*np.sum(S_weighted_standard)-K,0)
        payoff_AV[i] = max(1/(N+1)*np.sum(S_weighted_AV)-K,0)
        payoff_geometric[i] = max(np.product(np.power(S_weighted_geometric,1/(N+1)))-K,0)

    # Compute standard price and CI
    Dpayoff_standard = np.exp(-r*T)*payoff_standard
    CI_standard = st.t.interval(0.95, len(Dpayoff_standard)-1, loc=np.mean(Dpayoff_standard), scale=st.sem(Dpayoff_standard))
    price_standard = np.mean(Dpayoff_standard)

    # Compute AV price and CI
    Dpayoff_AV = np.exp(-r*T)*payoff_AV
    CI_AV = st.t.interval(0.95, len(Dpayoff_AV)-1, loc=np.mean((Dpayoff_standard+Dpayoff_AV)/2), scale=st.sem((Dpayoff_standard+Dpayoff_AV)/2))
    price_AV = np.mean((Dpayoff_standard+Dpayoff_AV)/2)

    # With CV price and CI
    Dpayoff_geometric = np.exp(-r*T)*payoff_geometric
    b_estimate = np.sum((Dpayoff_geometric-np.mean(Dpayoff_geometric))*(Dpayoff_standard-np.mean(Dpayoff_standard)))/np.sum((Dpayoff_geometric-np.mean(Dpayoff_geometric))**2)

    S_0_geometric = np.sqrt(S_0[0]*S_0[1])
    sigma_squared_bar = ((sigma[0,0]+sigma[1,0])**2+(sigma[0,1]+sigma[1,1])**2)/12
    delta = r/2+(sigma[0,0]**2+sigma[0,1]**2+sigma[1,0]**2+sigma[1,1]**2)/8-sigma_squared_bar/2

    exact_geometric = BS.Black_Scholes(1, S_0_geometric, K, T, np.sqrt(sigma_squared_bar), r, delta)
    Dpayoff_CV = Dpayoff_standard-b_estimate*(Dpayoff_geometric-exact_geometric)
    price_CV = np.mean(Dpayoff_CV)
    CI_CV = st.t.interval(0.95, len(Dpayoff_CV)-1, loc=np.mean(Dpayoff_CV), scale=st.sem(Dpayoff_CV))

    return [price_standard, price_AV, price_CV, CI_standard, CI_AV, CI_CV]
```

F

FEniCS Code: The PDE Method

F.1 European two-asset correlation call option

```python
"""
PDE_European_two_asset_corr_call computes the initial price of a
European two-asset correlation call option using the finite element
method (FEM).

The input  parameters are the boundaries in domain, the strike price K,
the risk-free interest rate r, the size of the mesh mesh_size=[nx,ny]
where nx and ny is the number of cells in direction x and y
respectively, the initial stock prices S_0, volatility sigma, the time
until maturity T and step in time of size tau.

The output is the estimated price of the option.
"""

from fenics import *
import numpy as np
import Black_Scholes as BS
import scipy.stats as st

def PDE_European_two_asset_corr_call(domain, K, r, mesh_size, S_0,
    sigma, T, tau):
    # Define coefficients for volatility and correlation
    a_11 = sigma[0,0]*sigma[0,0] + sigma[0,1]*sigma[0,1]
    a_22 = sigma[1,0]*sigma[1,0] + sigma[1,1]*sigma[1,1]
    a_12 = sigma[0,0]*sigma[1,0] + sigma[0,1]*sigma[1,1]
    a_21 = sigma[1,0]*sigma[0,0] + sigma[1,1]*sigma[0,1]
    rho = 0.5*(a_12+a_21)/np.sqrt(a_11*a_22)

    # Set domain
    S_1_low = S_2_low = domain[0]
    S_1_high = S_2_high = domain[1]

    # Create mesh and define function space
    mesh = RectangleMesh(Point(S_1_low, S_2_low), Point(S_1_high,
    S_2_high), mesh_size[0], mesh_size[1], "right/left")
    V = FunctionSpace(mesh, 'Lagrange', 2)

    # Define boundary condition for when S_1 -> 0
    u_S_1_0 = Constant(0)

    def boundary_S_1_0(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[0], S_1_low, tol)

    bc_S_1_0 = DirichletBC(V, u_S_1_0, boundary_S_1_0)

    # Define boundary condition for when S_2 -> 0
    u_S_2_0 = Constant(0)

    def boundary_S_2_0(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[1], S_2_low, tol)

    bc_S_2_0 = DirichletBC(V, u_S_2_0, boundary_S_2_0)

    # Define boundary condition for when S_1 -> infty
    def boundary_S_1_infty(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[0], S_1_high, tol)

    class BoundaryValues_S_1_infty(Expression):
        def __init__(self, **kwargs):
            self.t = 0
        def eval(self, values, x):
            values[0] = BS.Black_Scholes(1, x[1], K[1], self.t, np.sqrt
(a_22), r, 0)

    u_S_1_infty = BoundaryValues_S_1_infty(degree = 2)
    bc_S_1_infty = DirichletBC(V, u_S_1_infty, boundary_S_1_infty)

    # Define boundary condition for when S_2 -> infty
    def boundary_S_2_infty(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[1], S_2_high, tol)

    class BoundaryValues_S_2_infty(Expression):
        def __init__(self, **kwargs):
            self.t = 0
        def eval(self, values, x):
            y_1 = ( np.log(x[0]/K[0])+(r-a_11/2)*self.t ) / np.sqrt(
a_11*self.t)
            values[0] = x[1]*st.norm.cdf(y_1+rho*np.sqrt(a_22*self.t))
- K[1]*np.exp(-r*self.t)*st.norm.cdf(y_1)

    u_S_2_infty = BoundaryValues_S_2_infty(degree = 2)
    bc_S_2_infty = DirichletBC(V, u_S_2_infty, boundary_S_2_infty)

    # Define whole boundary
    bcs = [bc_S_1_0, bc_S_2_0, bc_S_1_infty, bc_S_2_infty]

    # Define initial value
    u_0 = Expression('x[0] > K_1 ? fmax(x[1]-K_2,0): 0', degree = 2,
K_1 = K[0], K_2 = K[1], r = r, t = 0)
    u_1 = interpolate(u_0, V)

    # Define variational problem
    u = TrialFunction(V)
    v = TestFunction(V)

    el = V.ufl_element()

    exp1 = Expression('a_11*x[0]*tau', a_11 = a_11, tau = tau, element = el)
    exp2 = Expression('0.5*a_11*pow(x[0],2)*tau', a_11 = a_11, tau = tau, element = el)

    exp3 = Expression('0.5*a_12*x[1]*tau', a_12 = a_12, tau = tau, element = el)
    exp4 = Expression('0.5*a_12*x[0]*x[1]*tau', a_12 = a_12, tau = tau, element = el)

    exp5 = Expression('0.5*a_21*x[0]*tau', a_21 = a_21, tau = tau, element = el)
    exp6 = Expression('0.5*a_21*x[0]*x[1]*tau', a_21 = a_21, tau = tau, element = el)

    exp7 = Expression('a_22*x[1]*tau', a_22 = a_22, tau = tau, element = el)
    exp8 = Expression('0.5*a_22*pow(x[1],2)*tau', a_22 = a_22, tau = tau, element = el)

    exp9 = Expression('-r*x[0]*tau', r = r, tau = tau, element = el)
    exp10 = Expression('-r*x[1]*tau', r = r, tau = tau, element = el)

    exp11 = Expression('r*tau', r = r, tau = tau, element = el)

    a = u*v*dx + exp1*u.dx(0)*v*dx + exp2*u.dx(0)*v.dx(0)*dx + exp3*u.dx(1)*v*dx + exp4*u.dx(1)*v.dx(0)*dx +exp5*u.dx(0)*v*dx + exp6*u.dx(0)*v.dx(1)*dx + exp7*u.dx(1)*v*dx +exp8*u.dx(1)*v.dx(1)*dx +exp9*u.dx(0)*v*dx +exp10*u.dx(1)*v*dx + exp11*u*v*dx
    L = (u_1)*v*dx

    A = None
    b = None

    # Compute solution
    u = Function(V)
    N = int(T/tau)
    for i in range(1, N+1):
        t = i*tau

        # Update boundary condition
        u_S_1_infty.t = t
        u_S_2_infty.t = t

        # Compute solution
        A, b = assemble_system(a, L, bcs)
        solve(A, u.vector(), b)

        # Update previous solution
        u_1.assign(u)

    return u(S_0[0], S_0[1])
```

F.2 European two-asset maximum call option

```
"""
PDE_European_two_asset_max_call computes the initial price of a
European two-asset maximum call option using the finite element
method (FEM).

The input  parameters are the boundaries in domain, the strike price K,
the risk-free interest rate r, the size of the mesh mesh_size=[nx,ny]
where nx and ny is the number of cells in direction x and y
respectively, the initial stock prices S_0, volatility sigma, the time
until maturity T and step in time of size tau.

The output is the estimated price of the option.
"""

from fenics import *
import numpy as np
import Black_Scholes as BS

def PDE_European_two_asset_max_call(domain, K, r, mesh_size, S_0, sigma
    , T, tau):
    # Define coefficients for volatility
    a_11 = sigma[0,0]*sigma[0,0] + sigma[0,1]*sigma[0,1]
    a_22 = sigma[1,0]*sigma[1,0] + sigma[1,1]*sigma[1,1]
    a_12 = sigma[0,0]*sigma[1,0] + sigma[0,1]*sigma[1,1]
    a_21 = sigma[1,0]*sigma[0,0] + sigma[1,1]*sigma[0,1]

    # Set domain
    S_1_low = S_2_low = domain[0]
    S_1_high = domain[1]
    S_2_high = K[1] - K[0] + domain[1]

    # Create mesh over domain and define function space
    mesh = RectangleMesh(Point(S_1_low, S_2_low), Point(S_1_high,
    S_2_high), mesh_size[0], mesh_size[1], "right/left")
    V = FunctionSpace(mesh, 'Lagrange', 2)

    # Define boundary condition for when S_1 -> 0
    def boundary_S_1_0(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[0], S_1_low, tol)

    class BoundaryValues_S_1_0(Expression):
        def __init__(self, **kwargs):
            self.t = 0
        def eval(self, values, x):
            values[0] = max(BS.Black_Scholes(1, x[0], K[0], self.t, np.
    sqrt(a_11), r, 0), BS.Black_Scholes(1, x[1], K[1], self.t, np.sqrt(
    a_22), r, 0))

    u_S_1_0 = BoundaryValues_S_1_0(degree = 2)
    bc_S_1_0 = DirichletBC(V, u_S_1_0, boundary_S_1_0)

    # Define boundary condition for when S_2 -> 0
    def boundary_S_2_0(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[1], S_2_low, tol)

    class BoundaryValues_S_2_0(Expression):
        def __init__(self, **kwargs):
            self.t = 0
        def eval(self, values, x):
            values[0] = max(BS.Black_Scholes(1, x[0], K[0], self.t, np.
   sqrt(a_11), r, 0), BS.Black_Scholes(1, x[1], K[1], self.t, np.sqrt(
   a_22), r, 0))

    u_S_2_0 = BoundaryValues_S_2_0(degree = 2)
    bc_S_2_0 = DirichletBC(V, u_S_2_0, boundary_S_2_0)

    # Define boundary condition for when S_1 -> infty
    def boundary_S_1_infty(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[0], S_1_high, tol)

    class BoundaryValues_S_1_infty(Expression):
        def __init__(self, **kwargs):
            self.t = 0
        def eval(self, values, x):
            values[0] = max(BS.Black_Scholes(1, x[0], K[0], self.t, np.
   sqrt(a_11), r, 0), BS.Black_Scholes(1, x[1], K[1], self.t, np.sqrt(
   a_22), r, 0))

    u_S_1_infty = BoundaryValues_S_1_infty(degree = 2)
    bc_S_1_infty = DirichletBC(V, u_S_1_infty, boundary_S_1_infty)

    # Define boundary condition for when S_2 -> infty
    def boundary_S_2_infty(x, on_boundary):
        tol = 1E-10
        return on_boundary and near(x[1], S_2_high, tol)

    class BoundaryValues_S_2_infty(Expression):
        def __init__(self, **kwargs):
            self.t = 0
        def eval(self, values, x):
            values[0] = max(BS.Black_Scholes(1, x[0], K[0], self.t, np.
   sqrt(a_11), r, 0), BS.Black_Scholes(1, x[1], K[1], self.t, np.sqrt(
   a_22), r, 0))

    u_S_2_infty = BoundaryValues_S_2_infty(degree = 2)
    bc_S_2_infty = DirichletBC(V, u_S_2_infty, boundary_S_2_infty)

    # Define whole boundary
    bcs = [bc_S_1_0, bc_S_2_0, bc_S_1_infty, bc_S_2_infty]

    # Define initial value
    u_0 = Expression('fmax(fmax(x[0]-K_1,0), fmax(x[1]-K_2,0))', degree
   = 2, K_1 = K[0], K_2 = K[1])
    u_1 = interpolate(u_0, V)

    # Define variational problem
    u = TrialFunction(V)
    v = TestFunction(V)

    el = V.ufl_element()

    exp1 = Expression('a_11*x[0]*tau', a_11 = a_11, tau = tau, element = el)
    exp2 = Expression('0.5*a_11*pow(x[0],2)*tau', a_11 = a_11, tau = tau, element = el)

    exp3 = Expression('0.5*a_12*x[1]*tau', a_12 = a_12, tau = tau, element = el)
    exp4 = Expression('0.5*a_12*x[0]*x[1]*tau', a_12 = a_12, tau = tau, element = el)

    exp5 = Expression('0.5*a_21*x[0]*tau', a_21 = a_21, tau = tau, element = el)
    exp6 = Expression('0.5*a_21*x[0]*x[1]*tau', a_21 = a_21, tau = tau, element = el)

    exp7 = Expression('a_22*x[1]*tau', a_22 = a_22, tau = tau, element = el)
    exp8 = Expression('0.5*a_22*pow(x[1],2)*tau', a_22 = a_22, tau = tau, element = el)

    exp9 = Expression('-r*x[0]*tau', r = r, tau = tau, element = el)
    exp10 = Expression('-r*x[1]*tau', r = r, tau = tau, element = el)

    exp11 = Expression('r*tau', r = r, tau = tau, element = el)

    a = u*v*dx + exp1*u.dx(0)*v*dx + exp2*u.dx(0)*v.dx(0)*dx + exp3*u.dx(1)*v*dx + exp4*u.dx(1)*v.dx(0)*dx +exp5*u.dx(0)*v*dx + exp6*u.dx(0)*v.dx(1)*dx + exp7*u.dx(1)*v*dx +exp8*u.dx(1)*v.dx(1)*dx +exp9*u.dx(0)*v*dx +exp10*u.dx(1)*v*dx + exp11*u*v*dx
    L = (u_1)*v*dx

    A = None
    b = None

    # Compute solution
    u = Function(V)
    N = int(T/tau)
    for i in range(1, N+1):
        t = tau*i

        # Update boundary condition
        u_S_1_0.t = t
        u_S_2_0.t = t
        u_S_1_infty.t = t
        u_S_2_infty.t = t

        # Compute solution
        A, b = assemble_system(a, L, bcs)
        solve(A, u.vector(), b)

        # Update previous solution
        u_1.assign(u)

    return u(S_0[0], S_0[1])
```

Bibliography

[1] Calogero S. Stochastic calculus, Financial derivatives and PDEs. Lecture notes for the course TMA285 at Chalmers University of Technology; 2018.

[2] Shreve SE. Stochastic calculus for finance II: Continuous-time models. vol. 11. Springer Science & Business Media; 2004.

[3] Haug EG. The complete guide to option pricing formulas. vol. 2. McGraw-Hill New York; 2007.

[4] Jiang L. Mathematical modeling and methods of option pricing. World Scientific Publishing Company; 2005.

[5] Hozman J, Tichỳ T. DG method for numerical pricing of multi-asset Asian options—The case of options with floating strike. Applications of Mathematics. 2017;62(2):171–195.

[6] Moon KS, Kim WJ, Kim H. Adaptive lattice methods for multi-asset models. Computers & Mathematics with Applications. 2008;56(2):352–366.

[7] Boyle PP, Evnine J, Gibbs S. Numerical evaluation of multivariate contingent claims. The Review of Financial Studies. 1989;2(2):241–250.

[8] Figlewski S, Gao B. The adaptive mesh model: a new approach to efficient option pricing. Journal of Financial Economics. 1999;53(3):313–351.

[9] Diener F, Diener M. Asymptotics of the price oscillations of a European call option in a tree model. Mathematical finance. 2004;14(2):271–293.

[10] Kamrad B, Ritchken P. Multinomial approximating models for options with k state variables. Management science. 1991;37(12):1640–1652.

[11] Pascucci A. PDE and martingale methods in option pricing. Springer Science & Business Media; 2011.

[12] Kuske RA, Keller JB. Optimal exercise boundary for an American put option. Applied Mathematical Finance. 1998;5(2):107–116.

[13] Glasserman P. Monte Carlo methods in financial engineering. vol. 53. Springer Science & Business Media; 2013.

[14] Kemna AG, Vorst AC. A pricing method for options based on average asset values. Journal of Banking & Finance. 1990;14(1):113–129.

[15] Black F, Scholes M. The pricing of options and corporate liabilities. Journal of political economy. 1973;81(3):637–654.

[16] Langtangen HP, Logg A. Solving PDEs in Python: The FEniCS Tutorial I. Springer; 2016.

[17] Logg A, Mardal KA, Wells G. Automated solution of differential equations by the finite element method: The FEniCS book. vol. 84. Springer Science & Business Media; 2012.