DEGREE PROJECT IN MATHEMATICS,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2017

Combinatorial and price efficient optimization of the underlying assets in basket options

SARA ALEXIS

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES

Combinatorial and price efficient optimization of the underlying assets in basket options

SARA ALEXIS

Degree Projects in Optimization and Systems Theory (30 ECTS credits)
Degree Programme in in Industrial Engineering and Management (120 credits)
KTH Royal Institute of Technology year 2017
Supervisor at Nordea Markets: Kristofer Eriksson
Supervisor at KTH: Johan Karlsson
Examiner at KTH: Johan Karlsson

TRITA-MAT-E 2017:10
ISRN-KTH/MAT/E--17/10--SE

Royal Institute of Technology
School of Engineering Sciences
KTH SCI
SE-100 44 Stockholm, Sweden
URL: www.kth.se/sci

Abstract

The purpose of this thesis is to develop an optimization model that chooses the optimal and price efficient combination of underlying assets for a equally weighted basket option.

To obtain a price efficient combination of underlying assets a function that calculates the basket option price is needed, for further use in an optimization model. The closed-form basket option pricing is a great challenge, due to the lack of a distribution describing the augmented stochastic price process. Many types of approaches to price an basket option has been made. In this thesis, an analytical approximation of the basket option price has been used, where the analytical approximation aims to develop a method to describe the augmented price process. The approximation is done by moment matching, i.e. matching the first two moments of the real distribution of the basket option with an lognormal distribution. The obtained price function is adjusted and used as the objective function in the optimization model.

Furthermore, since the goal is to obtain en equally weighted basket option, the appropriate class of optimization models to use are binary optimization problems. This kind of optimization model is in general hard to solve - especially for increasing dimensions. Three different continuous relaxations of the binary problem has been applied in order to obtain continuous problems, that are easier to solve.

The results shows that the purpose of this thesis is fulfilled when formulating and solving the optimization problem - both as an binary and continuous nonlinear optimization model. Moreover, the results from a Monte Carlo simulation for correlated stochastic processes shows that the moment matching technique with a lognormal distribution is a good approximation for pricing a basket option.

Keywords: Option pricing, correlated stochastic processes, Basket option, moment matching, lognormal approximation, binary nonlinear optimization, continuous nonlinear optimization, Monte Carlo simulation, penalty methods

Kombinatorisk och priseffektiv optimering av antalet underliggande tillgångar i aktiekorgar

Sammanfattning

Syftet med detta examensarbete är att utveckla ett optimeringsverktyg som väljer den optimala och priseffektiva kombinationen av underliggande tillgångar för en likaviktad aktiekorg.

För att kunna hitta en priseffektiv kombination av underliggande tillgångar behöver man finna en passande funktion som bestämmer priset på en likaviktad aktiekorg. Prissättningen av dessa typer av optioner är en stor utmaning. Detta är på grund av bristen av en sannolikhetsfördelning som kan beskriva den utökade och korrelerade stokastiska prisprocess som uppstår för en aktiekorg. Många typer av prissättningar har undersökts och tillämpats. I detta arbete har en analytisk approximation använts för att kunna beskriva den underliggande pris processen approximativt. Uppskattningen görs genom att matcha de två första momenten av den verkliga fördelningen med motsvarande moment för en lognormal fördelning. Den erhållna prisfunktionen justeras och används som målfunktionen i optimeringsmodellen.

Binära ickelinjära optimeringsproblem är i allmänhet svåra att lösa - särskilt för ökande dimen-sioner av variabler. Tre olika kontinuerliga omformuleringar av det binära optimeringsproblemet har gjorts för att erhålla kontinuerliga problem som är lättare att lösa.

Resultaten visar att en optimal och priseffektiv kombination av underliggande aktier är möjlig att hitta genom att formulera ett optimeringsproblem - både som en binär och kontinuerlig ickelinjär optimeringsmodell. Dessutom visar resultaten från en Monte Carlo-simulering, i detta fall för korrelerade stokastiska processer, att moment matching metoden utförd med en lognormal fördelning är en god approximation för prissättningen av aktiekorgar.

Acknowledgements

I would like to thank both my supervisors assistant Professor **Johan Karlsson** at KTH and **Kristofer Eriksson** at Nordea Markets for their support, guidance and most importantly feedback during the master thesis. Insightful discussions, remarks and engagement helped me to develop and form my master thesis into the final product. I really want to thank Kristofer for valuable and helpful discussions about the financial industry, both practical and theoretical aspects.

I would also like to thank Fredrik Armerin from KTH for giving me great advice regarding the theme of basket option pricing. Finally, I want to recommend the optimization solvers from TOMLAB Optimization which I have used in this thesis. Especially for solving binary nonlinear optimization models, the KNITRO solver is strongly recommended.

Stockholm, February 2017

Sara Alexis

Contents

Chapter 1

Introduction

1.1 Standard option

An option is defined as a security which gives its holder, the buyer of the option, the right but not the obligation to buy or sell the underlying asset associated with the option. This has the possibility to be done for a specified price, called the strike price, on or before a given date and is guaranteed by the seller of the option. The seller of the option therefore has the obligation to sell respectively buy the underlying asset.

Furthermore, options are categorized into call and put options depending on the right the holder has: the call option gives the holder the right to buy the underlying security to the strike price, a put option gives the right to sell the underlying security to the strike price. In addition, different styles of options exists that are based on when the holder has the right to exercise the option; in other words, exercise the right to buy or sell the underlying assets for the strike price.

- An American option is where the holder of an option has the right to exercise his option at any time prior to its expiration date.
- An European option is where the option can only be exercised at the end of its life, at its maturity.
- The Bermudan option is a hybrid of the American and European option; this option can be exercised on predetermined dates.

The value of an option is derived in part from the value and characteristics of the underlying security. Thus, the moneyness of an option is an important factor to consider. The moneyness is defined as the relative position of the price of the underlying asset with respect to the strike price of the option, according to three levels of the relative position:

1. Out-of-money: either a call option where the asset price is greater than the strike price, or a put option where the asset price is less than the strike price.
2. At-the-money: an option in which the strike price equals the price of the underlying asset.
3. In-the-money: either a call option where the asset price is less than the strike price, or a put option where the asset price is greater than the strike price.

1.2 Basket option

A basket option differs from a basic option in the way that it gives the holder the right, but not the obligation, to buy or sell a *group* of underlying assets [16] for the strike price. The basket option has the same characteristics of a standard option, but the strike price is based on the weighted value of the prices of the several underlying assets. Hence, the value of a basket option depends on the weighted sum of the underlying assets and is dependent of the performance of them. The underlying assets of basket options, as well as simple options, can be of multiple types: stocks, indexes, currencies, commodities, credit spreads etc.

There are several advantages of using a basket option on several underlying assets, instead of using plain options of each asset as presented in [9] and [12], among many others. The basket put option is risk reducing in the way that it protects against price drops in all the underlying assets at the same time, in comparison with buying plain put options for each asset. However, for the basket call option this is not favorable since a drop in the underlying assets will cause an decrease in the basket call option value. Furthermore, the investor has the opportunity to hedge the risk exposure by only using one derivative instead of several ones. It is cheaper to utilize a basket option for a set of underlying assets, than the corresponding sum of individual options on each underlying asset. This is the case because the basket option benefits from the effect of correlation between the underlying assets, which the portfolio consisting of the individual options would not. Also, the transaction cost is lowered when a single option is purchased instead of several ones.

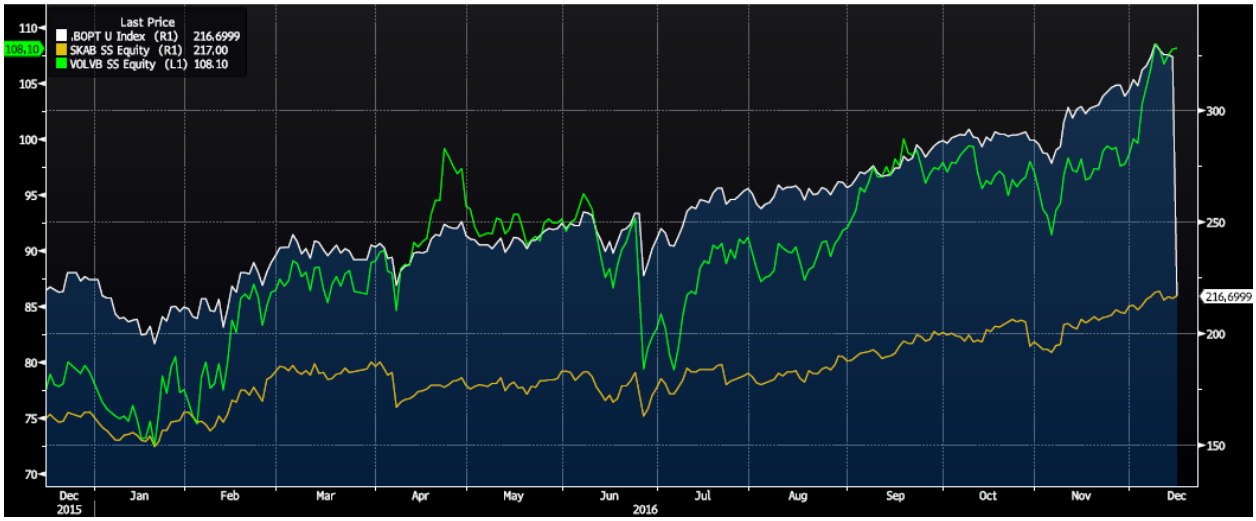


Figure 1.1: The price processes of an equally weighted basket option and its underlying stocks Volvo AB and Skanska AB

For pricing simple options on one underlying asset the Black & Scholes model have been widely used, which gives us a closed-form solution for the price of simple options. In addition, in reality the Black & Scholes model is often used to quote prices in terms of volatilities - in fact the options market often uses implied volatility, i.e. estimated volatility, to quote prices on options. Pricing models, calibrated for market quotes, are used for calculating prices when dealing with more complex options - such as the basket option.

In the case of a pricing basket option, a closed form solution for the Black & Scholes model does not exist. This is due to the lack of a distribution describing the augmented price process, for the group of underlying assets in a basket option. This is a feature that has made the closed-form basket option pricing a challenge and many types of approaches to price an basket option has been made. This could imply that there is no common way of pricing these products and therefore communicating prices with each other. The actors in the market price these products different and this will affect what price can be defined to be a cheap or expansive basket option. Also the pricing model is rather used as an tool when pricing the option, since there are several more factors taken into consideration when determining the price of an complex option. Hence, the price model is not and may necessary not be only the reason why the pricing of complex options, such as the basket option, is a complex matter.

1.3 Project definition

With necessary basics about options and especially basket options presented, the purpose and aim with this thesis can now be presented in order to formulate research questions. The questions will be operationalized with an methodology of defining and managing the necessary tasks.

1.3.1 Purpose and aim

The **purpose** of this thesis is to develop a tool that chooses the optimal and price efficient combination of underlying assets for a basket option. The idea is to construct an optimization model which can make this achievable.

Given a pricing model, the **aim** is to find the price efficient basket option based on finding the optimal subset of underlying assets. A price efficient basket option will in this thesis refer to the basket option with the lowest price. One should keep in mind that there is no common strategy among actors in the market when pricing a basket option. Hence finding a optimal subset of underlying assets, that yields the optimal and price efficient basket option, will be the optimal result specific for the chosen price model.

In addition, the **aim** is to incorporate the correlation between the underlying assets in the tool. Also, limitations on the number of assets belonging to different sectors, countries or/and other types of limitations on the assets characteristics will be taken into account. Since it is not possible to use an analytic method for pricing basket options and the goal is to find a price efficient basket option, the **aim** will also be to find an appropriate pricing method of a basket option.

1.3.2 Research questions and tasks

In order to operationalize the objectives, the following **research questions** are formulated:

1. Which basket option pricing model can be accurate and suitable to use as an objective function, in an optimization model?
2. How can we model and solve an optimization model that will return the most price efficient basket option?

The methodology of the thesis is constructed in order to answer and fulfill the research questions. Hence, the following **tasks** are formulated:

1. Investigate the basket option pricing and the challenges within it.
   (a) Study stochastic calculus for option pricing.
   (b) Research and study pricing methods for basket options.

   For evaluation: construct a model that use numerical simulation to calculate the basket option price and compare to the model specific results.
2. Set up an appropriate optimization model:
   (a) Adjust the chosen price function in order to fulfill the requirement of price efficiency.
   (b) Formulate necessary constraints in order to fulfill the presented aim of the tool.
   (c) Classify and study the optimization problem.
   (d) Choose appropriate optimization solver and do numerical simulations.
   (e) Evaluate the optimization model and the solving algorithm.

   For evaluation: A comparison of the results from the models will be made, by using an algorithm that finds the optimal combination of underlying assets with the lowest basket option price. This algorithm will find all possible and feasible combinations and chose the one with the lowest price function value. This will however not be computationally possible for a large number of combinations.

1.3.3 Delimitations

- The underlying assets that will be considered are only stocks. Furthermore, the basket option will be equally weighted.
- Only a limitation on the number of assets belonging to a certain sector will be applied.
- The focus in task 1 will not be to find a new pricing model or theoretically develop an existing pricing model.
- Only European options will be considered when performing the two tasks.
- Effects of currency movements will not be considered in the pricing model. Furthermore, all the underlying assets are handled in the same currency.
- There is no model calibration for market quotes.

1.4 Outline

Chapter 2: Study stochastic calculus and the Black & Scholes theory in one- and multidimensional case. This will be developed into studying basket option pricing. Eventually, a pricing method will be chosen in order to develop the thesis further to begin setting up the optimization model. In addition, the numerical simulation of the basket option will be described in this chapter.

Chapter 3: In order to utilize optimization theory for solving the problem, necessary optimization theory is defined and introduced.

The optimization models are derived and analyzed in the following ***Chapter 4***. The rest of the thesis is left for presenting the result and analysis in ***Chapter 5*** and for evaluating and stating conclusions in ***Chapter 6***.

Chapter 2

Stochastic calculus and Option pricing

2.1 Stochastic processes for stocks

Pricing a basket option requires to first understand the price process of a single underlying asset for an standard option. The theory can then be extended for the case of a basket option, which is characterized by an option with several underlying and correlated assets. A specific stochastic process that is used to model stock prices will be presented and derived for both cases, in order to study and analyze the Black & Scholes model - especially in a multidimensional environment suitable for a basket option. Eventually, the chapter will result in how a basket option can be priced.

2.1.1 Process for one stock

A stochastic process describes a variable whose value changes over time in an random and uncertain way. It can be simply described as an collection of random variables indexed by time [16]. Stochastic processes can be classified to be in discrete- or continuous time, meaning that variables can change at certain fixed points in time respectively at any time:

- Stochastic process in discrete-time denoted as:

$$\{S(t), t = 1, 2, 3...\} \tag{2.1}$$

- Stochastic process in continuous-time denoted as:

$$\{S(t), t \geq 0\} \tag{2.2}$$

Each of the processes can be classified further by being continuous- or discrete variable, i.e. the process $\{S(t)\}$ can take any value within a certain range or take only certain discrete values [16]. The general way of describing a stock's price process is by a continuous-variable, continuous-time stochastic process which will be the case in this thesis. Hence, the stochastic process for stock prices will be denoted as in (2.2).

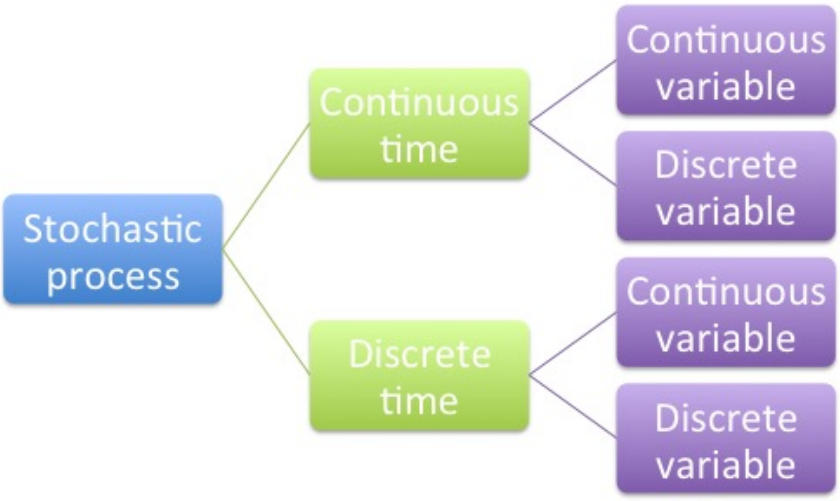


Figure 2.1: The components of a stochastic process

2.1.1.1 The Markov property

The first step towards deriving the stochastic process for stock prices is by defining the Markov property. It is assumed that stock prices follows a Markov process which has the following Markov property; the current value s_t of a variable S_t is the only relevant time point, for predicting the future value of the variable at $t+1$ [16]. In other words, past history and the development until the present value has emerged is irrelevant and does not effect when predicting future values:

$$p[S(t+1)|S(0) = s_0, S(1) = s_1, ..., S(t) = s_t] = p[S(t+1)|S(t) = s_t], \tag{2.3}$$

where p is the probability measure. The Markov property also implies that a probability distributions needs to be used, when obviously expressing uncertain predictions for future values. Hence, the probability distribution of the stock price at any future time is not affected by any path following the historical prices.

2.1.1.2 Wiener Processes

A particular type of Markov processes that is of main interest when dealing with stock prices is the Wiener process. According to [18], [1] this can be defined according to its following characteristics:

Definition 2.1.1. A Wiener process $\{S(t) : t \geq 0\}$, also called standard Brownian motion, is a stochastic process with the following properties:

1. S(0)=0 almost surely.
2. Non-overlapping increments are independent and stationary, i.e. S(T)-S(t) and S(U)-S(u) are independent random variables for all time points satisfying $t, s \geq 0$ and $0 \leq t < T \leq u < U$.
3. For all $0 \leq t < u$ the increment $S(u) - S(t)$ is a normal random variable with *zero mean* and *variance (u-t)*. I.e., $S(u) - S(t) \in N(0, u - t)$ for $u > t$.
4. With probability 1, the path $t \to S(t)$ is a continuous function on $[0, \infty)$.

Hence, *the Wiener process is a Gaussian process.* Furthermore, any process with the features of a Wiener process is characterized by the parameters:

- The mean change per unit time for a stochastic process denoted by μ, which is called the drift rate *(expected value at future time).*
- The variance per unit time *(variance of change in a time interval)*, known as the variance rate σ.
- The standard Wiener process $dW(t)$ which is defined as the Wiener process drift rate of zero and a variance rate of 1 (i.e. unit variance Wiener process).

Theorem 2.1.1. A standard Brownian motion, or Wiener process, $\{S(t) : t \geq 0\}$ is the solution of, i.e. satisfies, an stochastic differential equation (SDE) with constant drift and diffusion coefficients given by:

$$dS(t) = \mu dt + \sigma dW(t), \tag{2.4}$$

where

- the initial value of the process is defined as $S(0) = s_0$.
- μdt stands for the expected drift a per unit time.
- $\sigma dW(t)$ is the variability added to the path followed by S(t).

2.1.1.3 Itô Process

A type of generalized version of Wiener process is called the Itô process, where the parameters μ and σ are functions of the time t and the underlying variable $S(t)$ [1]:

$$dS(t) = \mu(S(t), t)dt + \sigma(S(t), t)dW(t). \tag{2.5}$$

This implies that the expected drift rate and variance rate of an Itô process can change over time. Also, this process is a Markov process since the change in $S(t)$ at time t depends on t and not on previous time points.

Itô's lemma

Itô's lemma is used to determine the derivative of a time-dependent function of a stochastic process. The main idea is derived from the chain rule of deriving in ordinary differential calculus, but in a stochastic setting [1].

Theorem 2.1.2. Let G(t) be an Ito drift-diffusion process which satisfies the stochastic differential equation:

$$dG(t) = \mu(G(t), t)dt + \sigma(G(t), t)dW(t). \tag{2.6}$$

If $f(g, t) \in C^2(R^2, R)$ then f(G(t),t)=f(G,t) is also an Ito drift-diffusion process with differential given by

$$\begin{aligned} d(f(G,t)) = \frac{\partial f}{\partial t}(G,t)dt + \mu(G,t)\frac{\partial f(G,t)}{\partial G}dt + \\ \frac{1}{2}\frac{\partial^2 f(G,t)}{\partial G^2}\sigma(G,t)^2 dt + \sigma(G,t)\frac{\partial f(G,t)}{\partial G}dW(t). \end{aligned} \tag{2.7}$$

In order to model the stock price distribution correctly, Itô's lemma is used to solve the stochastic differential equation representing Geometric Brownian motion, which is presented in the following section.

2.1.1.4 Geometric Brownian Motion

It is tempting to suggest that a stock price follows a standard Brownian motion; that is, having a constant expected drift rate and a constant variance rate. However, it does not capture the key aspect by assuming constant expected drift - expected return required by investors from a stock must be independent of the stock's price [16]. This assumption needs to be replaced with a constant expected return instead.

In addition, a standard Brownian motion is insufficient for asset price movements, but are a fundamental component in the construction of stochastic differential equations - which will eventually allow derivation of the famous Black-Scholes equation [16]. However, it has a non-zero probability of being negative. This is clearly not a property shared by real-world assets - stock prices cannot be less than zero.

As demonstrated in [11] among others, this is solved by first defining the following:

- $S(t)$ as the stock price at time t
- $\mu S(t)$ as expected drift in $S(t)$ where μ is a constant parameter, which is the expected rate of return on a stock

The expected increase in $S(t)$ under a short time interval dt is thus $\mu S(t)dt$:

$$\begin{aligned} dS(t) &= \mu S(t)dt \\ \frac{dS(t)}{S(t)} &= \mu dt. \end{aligned} \tag{2.8}$$

The solution to the differential equation (2.8) when integrating from t=0 to t=T is:

$$\begin{aligned} S(T) &= S(0)e^{\mu T} \\ S(0) &= s_0. \end{aligned} \tag{2.9}$$

Including uncertainty is relevant since this is the case in practice. Hence, we need to assume that the variability of return under the short time period δt is the same regardless of the stock price $S(t)$ - an investor is uncertain about the return regardless of what the price is. Hence, the standard deviation of the change of δt should be proportional to the stock price as follows:

$$dS(t) = \mu S(t)dt + \sigma S(t)dW \tag{2.10}$$

$$\frac{dS(t)}{S(t)} = \mu dt + \sigma dW. \tag{2.11}$$

The model of stock price behavior we have developed here is known as geometric Brownian motion. This is the most widely used model of stock price behavior. *In a risk-neutral world, μ equals the risk-free rate r when stocks do not pay dividends. If the asset S_t pays dividend, then $\mu = r - q$ follows.* We will continue by using μ.

Theorem 2.1.3. A Geometric Brownian motion is the solution of an SDE with linear drift and diffusion coefficients according to:

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t). \tag{2.12}$$

Note that the coefficients μ and σ, representing the drift and volatility of the asset, respectively, are both constant in this model. In more sophisticated models they can be made to be functions of $t, S(t)$ and other stochastic processes. The solution to (2.12) is given by:

$$\begin{aligned} S(t) &= S(0)e^{(\mu - \frac{\sigma^2}{2})t + \sigma W(t)} \\ S(0) &= s_0. \end{aligned} \tag{2.13}$$

In order to obtain the solution, Itô's lemma (presented in previous subsection) has been applied straightforward on the function $f(S(t)) = log(S(t))$ to the stochastic differential equation. Hence $S(t)$ is a log-normally distributed random variable, since the standard Brownian Motion $W(t)$ is normally distributed.

Lemma 2.1.4. At each fixed time point, the Geometric Brownian Motion $S(t)$ has a log-normal distribution with parameters

$$\begin{aligned} \mathbf{E}[S(t)] &= S(0)e^{\mu t} \\ \mathbf{Var}[S(t)] &= S(0)^2 e^{2\mu t}(e^{\sigma^2 t} - 1). \end{aligned} \tag{2.14}$$

Most economists prefer Geometric Brownian Motion as a model for market prices because it is everywhere positive (with probability 1), in contrast to Brownian Motion, including Brownian Motion with drift.

2.1.2 Multidimensional correlated process

Up to this point, one-dimensional and independent stochastic processes has been considered. Further follows the construction of multidimensional and correlated stochastic processes, as in [1]. In particular, we will proceed from the Geometric Brownian Motion and study it in a multidimensional setting, allowing the processes to be correlated with each other.

Consider n independent, i.e. uncorrelated, standard (unit variance) Wiener processes $W_1^*(t), .., W_n^*(t)$ and define a deterministic and constant matrix Δ:

$$\Delta = \begin{bmatrix} \Delta_{11} & \dots & \Delta_{1n} \\ \vdots & \ddots & \vdots \\ \Delta_{n1} & \dots & \Delta_{nn} \end{bmatrix} \tag{2.15}$$

where each row $\Delta_i = [\Delta_{i1}, .., \Delta_{in}]$ has unit length, i.e. $||\Delta_i|| = 1$. Now consider the n-dimensional process $W(t)$:

$$W(t) = \begin{bmatrix} W_1(t) \\ \vdots \\ W_n(t) \end{bmatrix} \tag{2.16}$$

where $W(t)$ is defined by:

$$W_i(t) = \sum_{j=1}^{n} \Delta_{ij} W_j^*(t), \quad i = 1, \dots, n. \tag{2.17}$$

Thus, with this construction each component $W_1(t), .., W_n(t)$ in $W(t)$ is separately a standard Wiener process. What has been done is building correlated Wiener processes $W(t)$ by taking combinations of uncorrelated Wiener processes $W^*(t)$. In the same manner:

$$dW_i(t) = \sum_{j=1}^{n} \Delta_{ij} dW_j^*(t), \quad i = 1, \dots, n. \tag{2.18}$$

Define the correlation matrix φ:

$$\varphi = \begin{bmatrix} \varphi_{11} & \dots & \varphi_{1n} \\ \vdots & \ddots & \vdots \\ \varphi_{n1} & \dots & \varphi_{nn} \end{bmatrix} \tag{2.19}$$

The correlation matrix φ of $W(t)$ is defined element-wise by [1]:

$$\begin{gathered} \mathbf{Cov}[dW_i(t), dW_j(t)] = \mathbf{Corr}[dW_i(t), dW_j(t)] * \mathbf{SD}[dW_i(t)] * \mathbf{SD}[dW_j(t)] \\ \text{i.e.} \quad \mathbf{Cov}[dW_i(t), dW_j(t)] = \varphi_{ij} dt \end{gathered} \tag{2.20}$$

where:

$$\begin{aligned}&\mathbf{SD}[dW_i(t)]\mathbf{SD}[dW_j(t)] = \sqrt{\mathbf{Var}[dW_i(t)]}\sqrt{\mathbf{Var}[dW_j(t)]}\\&= \sqrt{\mathbf{E}[d^2W_i(t)]}\sqrt{\mathbf{E}[d^2W_j(t)]} = \sqrt{dt}\sqrt{dt} = dt.\end{aligned} \tag{2.21}$$

To understand how (2.21) is derived, recall from **Definition 2.1.1.**3) that a Wiener process is a Gaussian process that has zero mean and a variance equal to the time increment. This implies:

$$\mathbf{Var}[dW_i(t)] = \mathbf{E}[d^2W_i(t)] - \mathbf{E}[dW_i(t)]^2 = \mathbf{E}[d^2W_i(t)]. \tag{2.22}$$

To demonstrate what (2.22) becomes, consider the following which are consequences of **Definition 2.1.1**. Define $\delta t = t - s$ and $\delta W(t) = W(t) - W(s)$, where $s < t$:

$$\begin{aligned}&\mathbf{E}[\delta W(t)] = 0\\&\mathbf{E}[\delta^2 W(t)] = \delta t\\&\mathbf{Var}[\delta W(t)] = \delta t\\&\mathbf{Var}[\delta^2 W(t)] = 2\delta t^2.\end{aligned} \tag{2.23}$$

We see from this that as δt tends to zero $\delta^2 W(t)$ will also tend to zero - the variance will approach zero much faster than the expected value. Thus, $\delta^2 W(t)$ will look "deterministic". This leads to believing that in the limit the following equality holds for an infinite small :

$$[dW_i(t)]^2 = dt. \tag{2.24}$$

Hence:

$$\mathbf{Var}[dW_i(t)] = \mathbf{E}[d^2W_i(t)] = dt. \tag{2.25}$$

Further motivation for this equality can be found in [1]. Then, the following can be obtained:

$$\begin{aligned}&\mathbf{Cov}[dW_i(t), dW_j(t)] = \mathbf{E}[dW_i(t)dW_j(t)] - \mathbf{E}[dW_i(t)]\mathbf{E}[dW_j(t)]\\&= \mathbf{E}[\sum_{k=1}^{n} \Delta_{ik} dW_k^*(t) \sum_{l=1}^{n} \Delta_{jl} dW_l^*(t)] = \sum_{kl} \Delta_{ik}\Delta_{jl}\mathbf{E}[dW_k^*(t)dW_l^*(t)]\\&= \sum_{k=1}^{n} \Delta_{ik}\Delta_{jk}dt = \Delta_i\Delta_j^\intercal dt.\end{aligned} \tag{2.26}$$

Hence, the following holds:

$$\varphi = \Delta\Delta^\intercal. \tag{2.27}$$

The n-dimensional correlated Geometric Brownian motion $S(t) = [S_1(t), .., S_n(t)]^\intercal$ can now be formulated. Each of the components $S_1(t), .., S_n(t)$ has a SDE $dS_i(t)$ of the form:

$$dS_i(t) = \mu_i S_i(t)dt + \sum_{j=1}^{n} \sigma_{ij} S_i(t)dW_i(t) \quad i = 1, .., n. \tag{2.28}$$

The SDE can also be written as:

$$dS_i(t) = \mu_i S_i(t)dt + \sigma_i S_i(t)dW_i(t) \quad i = 1, .., n. \tag{2.29}$$

Under a risk neutral measure Q the dynamics in (2.29) is rewritten. This implies that the stochastic process has to be transformed into its risk-neutral form by replacing μ by the risk-free interest rate r, for non-dividend paying assets - which will be the case through the subsection when referring to the risk neutral measure Q. Hence we have:

$$dS_i(t) = rS_i(t)dt + \sigma_i S_i(t)dW_i(t) \quad i = 1, .., n. \tag{2.30}$$

In the same manner as in subsection 2.1.1.3, utilizing a n-dimensional version of Itô's lemma, the explicit formula of the price processes (2.30) is:

$$S_i(t) = S_i(0)e^{(r-\frac{\sigma_i}{2}^2)t+\sigma_i W_i(t)} \quad i = 1, .., n. \tag{2.31}$$

By replacing μ with $(r - q_i)$ where q_i is the dividend yield, (2.30) and (2.31) can be adjusted for dividends. Furthermore, each of the random variables $S_i(t)$ in (2.31) are log-normally distributed under the risk neutral measure Q. This is due to the fact that the geometric Brownian motion is log-normally distributed at each fixed time point, as have been demonstrated for the case of one dimension (see section 2.1.1.4). In other words, the underlying asset's price at a fixed time follows a log normal distribution.

However, the problem of pricing basket option can now be identified. Define a basket option on n underlying assets as follows:

$$S_b(t) = \sum_{i=1}^{n} w_i S_i(t) \tag{2.32}$$

where w_i is the weight for asset i with price $S_i(t)$ according satisfying the dynamics given in (2.30). Furthermore, a basket option has a payoff that depends on the value of a basket of the underlying assets:

$$\begin{aligned} &\textbf{Call} \quad [\sum_{i=1}^{n} w_i S_i(T) - K)]^+ \\ &\textbf{Put} \quad [\sum_{i=1}^{n} K - w_i S_i(T)]^+. \end{aligned} \tag{2.33}$$

The price of an call respectively put option with payoff defined as in (2.33) can be computed by:

$$\begin{aligned} C(K,T) &= e^{-rT} E^Q[(\sum_{i=1}^{n} w_i S_i(T) - K)^+] \\ P(K,T) &= e^{-rT} E^Q[(K - \sum_{i=1}^{n} w_i S_i(T))^+] \end{aligned} \tag{2.34}$$

which is the price of an (European) basket option, under the risk neutral valuation Q. The distribution for $S_b(t)$ defined in (2.32) - which is a weighted sum of log normal variables S_i - is not log-normal again. There is no distribution that can characterize the random variable $S_b(t)$ [16]. Therefore a analytic solution or a closed-form solution does not exist for (2.34), due to lack of a distribution that can describe the random variable $S_b(t)$ under the risk neutral measure Q.

2.2 Black & Scholes model

The original paper by Black & Scholes [2] assumes that the price of the underlying asset is a stochastic process $\{S(t)\}$ which solves the stochastic differential equation

$$dS(t) = rS(t)dt + \sigma S(t)dW(t) \tag{2.35}$$

where μ denotes the continuously compounded expected return on the stock, σ denotes the volatility and $\{W(t)\}$ is a standard Brownian motion. In other words, $\{S(t)\}$ is a geometric Brownian motion.

Suppose there exists a vanilla European option with price C which is a function of the time t and the price of the asset S(t): C=C(S(t),t)=C(S,t). Itô's lemma is then applied on the function $C(S,t)$ to obtain the SDE:

$$dC = \frac{\partial C}{\partial t}dt + \frac{\partial C}{\partial S}(S,t)dS + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}(S,t)dS^2. \tag{2.36}$$

Substituting (2.35) into (2.36) yields:

$$dC = [\frac{\partial C}{\partial t}(S,t) + rS\frac{C}{S}(S,t) + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 C}{\partial S^2}(S,t)]dt + \sigma S\frac{\partial C}{\partial S}(S,t)dW_t. \tag{2.37}$$

In order to state that the Black & Scholes will grow at a risk free interest rate when all risk is eliminated, one need to determine how this portfolio changes in time.

Specifically, the infinitesimal change of a mixture of a call option and a quantity of assets is of interest. The quantity is denoted Δ and the infinitesimal change is derived by express equation (2.37) as $d(C + \Delta S)$.

$$d(C + \Delta S) = (\frac{\partial C}{\partial t}(S,t) + rS\frac{C}{S}(S,t) + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 C}{\partial S^2}(S,t) + \Delta rS)dt + \Delta S(\frac{\partial C}{\partial S} + \Delta)dW_t. \tag{2.38}$$

The choice of Δ is chosen such that the term associated with randomness is eliminated. Hence:

$$\Delta = -\frac{C}{S}(S,t). \tag{2.39}$$

This is called delta-hedging and provides a portfolio that is free of randomness. This is how one can apply the argument that it should grow at the risk free rate r, otherwise arbitrage opportunity would have existed. Hence the growth rate of the delta-hedged portfolio must be equal the continuously compounding risk free rate r. Hence, by this assumption and inserting (2.39) in the left-hand side of (2.38) one can now state the derived Black & Scholes equation:

$$\frac{\partial C}{\partial t}(S,t) + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 C}{\partial S^2}(S,t) = r(C - S\frac{\partial C}{\partial S}). \tag{2.40}$$

The Black-Scholes equation is a second-order linear partial differential equation (PDE). A unique solution exists when there is enough conditions. The payoff function $C(S(T),T) = max(S(T) - K, 0)$ at expiry T is a suited choice, where K is the strike price of the option and $S(T)$ the underlying assets price at T. Hence the Black-Scholes equation can now be solved.

The Black-Scholes formula calculates the price of European put and call options, it is consistent with the Black-Scholes equation. It can be obtained by solving the equation for the corresponding terminal and boundary conditions. Hence, the prices are

$$\begin{aligned} C(S(t),t) &= N(d_1)S(t) - N(d_2)Ke^{-r(T-t)} \\ P(S(t),t) &= N(-d_2)Ke^{-r(T-t)} - N(d_1)S(t). \end{aligned} \tag{2.41}$$

where $(T-t)$ is the time to maturity and $N(*)$ is the cumulative distribution function of the standard normal distribution.The cumulative distribution (cdf) function of any distribution is expressed as the probability that the variable take a value less than or equal to x. Specifically, for a normal distribution the cdf is:

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(x)dx = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = \Phi(\frac{x-\mu}{\sigma}). \tag{2.42}$$

Furthermore, the Black & Scholes parameters d_1 respectively d_2 are:

$$\begin{aligned} d_1 &= \frac{1}{\sigma\sqrt{T-t}}[ln(\frac{S_t}{K}) + (r + \frac{\sigma^2}{2})(T-t)] \\ d_2 &= d_1 - \sigma\sqrt{T-t}. \end{aligned} \tag{2.43}$$

An alternative formulation of the price functions is:

$$\begin{aligned} C(S(t),t) &= e^{-r(T-t)}(FN(d_1) - KN(d_2)) \\ P(S(t),t) &= e^{-r(T-t)}(KN(-d_2) - FN(d_1)) \end{aligned} \tag{2.44}$$

where $F = S(0)e^{r(T-t)}$ and adjusted Black & Scholes parameters:

$$\begin{aligned} d_1 &= \frac{1}{\sigma\sqrt{T-t}}[ln(\frac{F}{K}) + \frac{\sigma^2}{2})(T-t)] \\ d_2 &= d_1 - \sigma\sqrt{T-t}. \end{aligned} \tag{2.45}$$

The model can also be extended for instruments paying dividends D_i by calculating the forward price for each asset as

$$F = S(0)e^{rT} - \sum_{i=1}^{N} D_i e^{r(T-t_i)}. \tag{2.46}$$

2.2.1 Multidimensional Black-Scholes model

The classical Black-Scholes formula gives in closed form the price of a call or a put option on a single stock, when the latter is modeled as a geometric Brownian motion as has been shown in Chapter 2. As in the one dimensional case, the stocks are assumed to be modeled with multidimensional geometric Brownian dynamics which are correlated processes.

The underlying stocks each has price $S_i(t)$ which are given by the dynamics

$$dS_i(t) = rS_i(t)dt + \sigma_i S_i(t)dW_i(t) \tag{2.47}$$

$$\frac{dS_i(t)}{S_i(t)} = rdt + \sigma_i dW_i(t). \tag{2.48}$$

In the same manner as in Section (2.1) the partial differential equation known as the Black & Scholes equation can be derived for the multidimensional case where multidimensional Itô's formula is used, the derivation is omitted in this thesis and can be found in [4]. The multidimensional Black & Scholes equation is given by:

$$\frac{\partial \mathbf{C}}{\partial t} + \frac{1}{2}\sum_{i,j=1}^{d} \sigma_i \sigma_j \varphi_{ij} S_i S_j \frac{\partial^2 \mathbf{C}}{\partial S_i \partial S_j} + r\sum_{i=1}^{d} S_i \frac{\partial \mathbf{C}}{\partial S_i} - r\mathbf{C} = 0 \tag{2.49}$$

where the function $\mathbf{C}$ has a multidimensional vector of underlying assets.

Equation (2.49) is the Black & Scholes equation for pricing an option on several assets. The problem of pricing the above basket options in the Black & Scholes model is the following: the stock prices are modelled by geometric Brownian motions and are therefore log-normally distributed. As the sum of log-normally distributed random variables is not log-normal, it is not possible to derive an closed-form representation of the basket call and put prices. In addition, it is hard to solve it numerically, whereas a large number of assets will even make numerical solutions to PDE ineffective in terms of time and have slow convergence of the answers as several authors have been demonstrating, among others [7], [26], [13].

2.3 Approaches to price basket option

Obtaining a explicit analytic expression for $P(K,t)$ in (2.41) is not possible, due to not having an explicit analytic expression for the distribution of the weighted average $S_b(t)$. Thus, pricing basket options is not a trivial task. Financial practitioners have to resort to numerical integration, simulations, or approximations. In high dimensions, numerical integration and simulation methods may be too slow and inefficient for practical purposes. Many areas of computational finance require robust and accurate algorithms to price these options [7]. Several approaches have been proposed in the literature, the most usual has been:

- **Numerical methods**: the most mentioned numerical method in multi-asset option pricing in literature is Monte Carlo simulation. Yields a numerical estimate of the price and is often used as a benchmark to valuate other approaches.

- **Partial differential equations approach**: numerical suggestions. The price of an basket option can be determined as a solution of a partial differential equation (PDE), which is the Black & Scholes equation as mentioned in previous section.We have seen that this PDE cannot be easily solved analytically, especially when for high dimensions. Generally, different transforms can be used in order to solve PDE's.

- Calculating **upper and lower bounds** of the basket option price, putting it in relation to a numerical simulation in several cases to get a "sense" of the basket option. In addition, this approach aims more to find an optimal price or hedge when incorporated in optimization models (not price optimization by choosing appropriate underlyings, which is our objective).

- **Analytic approximations**: techniques consist of approximating the real distribution of the payoff by another that is more tractable - an known method here is called moment matching were the moments of the real distribution are matched with the moments for a suitable chosen distribution. Other ideas are based on conditioning the price process of the underlying assets.

The approach by utilizing *analytical approximations* has been chosen as the suitable method for deriving the price function for a basket option. The motivation for this have been that price functions derived from this approach can easier be adjusted, in order to decrease the complexity of the function by omit unnecessary terms. Also, this method is a suitable and optimization "friendly" price function, for further use in an optimization model. In addition, the moment matching method itself has the advantage providing a closed-form analytic formula, alike the model of the Black-Scholes formula [3]. It provides easiness when varying the dimension of the basket option, i.e. the amount of underlying assets which is important for this thesis.

Furthermore, numerical estimation of a basket option price, that is needed in order to evaluate the chosen price function method, will be gained by using *Monte Carlo simulation*. Since the focus in this thesis is on basket options with underlying assets that are correlated, a multidimensional Monte Carlo simulation that generates correlated paths is of main interest.

2.3.1 Numerical estimate by Monte Carlo simulation

The Monte Carlo simulation is conducted for each underlying asset in the basket option. Recall that the assets are correlated log-normal variables, hence the a Monte Carlo simulation that generates paths for correlated log-normal variables will be used. The generated paths are suitable to be used in the Monte-Carlo approach to pricing options on a basket of assets. Thus, the basket option price will be estimated by performing an numerical simulation for n-dimensional correlated Geometric Brownian motions, as in [15] and remillard2013statistical.

Recall that in equation (2.23) from Subsection 2.1.2, it was given that each asset $S_i(t)$ is assumed to follow the Geometric Brownian motion which can be formulated as:

$$dS_i(t) = rS_i(t)dt + \sum_{j=1}^{n} \sigma_{ij} S_i(t) dW_i(t) \quad i = 1, .., n. \tag{2.50}$$

By utilizing that the correlated Wiener processes $dW_i(t)$ are a combination of uncorrelated Wiener processes $dW_j^*(t)$ as formulated in (2.18), the following can be obtained:

$$dS_i(t) = rS_i(t)dt + \sum_{j=1}^{n} \Omega_{ij} S_i(t) d^*W_j(t) \quad i = 1, .., n. \tag{2.51}$$

where $\Omega_{ij} = \sigma_{ij}\Delta_{ij}$. Ω_{ij} is thus decomposed by applying the Cholesky decomposition, a LU factorization of the covariance matrix according to:

$$\sigma_{ij} = \varphi_{ij}\sigma_i\sigma_j = (\Omega\Omega^T)_{ij}. \tag{2.52}$$

The Cholesky matrix Ω transforms uncorrelated variables into variables whose variances and covariances are given by the covariance matrix. In particular, the Cholesky transformation maps the

standard normal variables into variables for the multivariate normal distribution.

The solution $S_i(t)$ for the GBM in (2.45) is:

$$S_i(t) = S_i(0)e^{(r-\frac{{\sigma_i}^2}{2})t+\sum_{j=1}^{n}\Omega_{ij}W_j(t)}. \tag{2.53}$$

Equation (2.53) is discretized by using discrete Euler Scheme, recall the property of the Wiener process where $W(t) - W(s) \sim N(0, t-s)$. Hence:

$$S_i(t) = S_i(0)e^{(r-\frac{\sigma_i^2}{2})t+\sum_{j=1}^{n}\Omega_{ij}Z_i\sqrt{t}} \tag{2.54}$$

where $Z_i \sim N(0,1)$. The path for each underlying asset is simulated according to (2.54). The formula allows to simulate a path for each individual underlying asset at every timestep t. In the case of dividend paying assets, the term r is replaced by $(r - q_i)$ in (2.54).

2.3.2 Price function by analytic approximation method

Analytic approximations focuses on dealing with the problem of finding an good approximation for the unknown distribution of $S_b(t)$, when pricing the basket option. All suggested models proceeds from the Black & Scholes model. The general idea is that the expected value and variance is at least matched with the real distribution, a technique known as matching moments. Further adjustments are also possible when moment matching with the real distribution.

The moment matching procedure is conducted as follows. Let us first define, as before:

$$S_b(t) = \sum_{i=1}^{N} w_i S_i(t). \tag{2.55}$$

The forward value of each underlying asset is given by:

$$\begin{aligned} F_i &= S_i(0)e^{rT} \\ F_i &= S_i(0)e^{rT} - \sum_{i=1}^{N} D_i e^{r(T-t_i)} \end{aligned} \tag{2.56}$$

for non dividend-paying respectively dividend-paying assets, where time t_i is the time point a dividend is paid out. Thus, the forward value of the basket can be formulated as:

$$F_b = \sum_{i=1}^{N} w_i F_i. \tag{2.57}$$

Since the variables $S_i(t)$ are lognormally distributed, then $Y_i = logS_i(t)$ are normal variables. Consider one lognormal variable S and define:

$$\frac{Y-\mu}{\sigma} = Z \sim N(0,1). \tag{2.58}$$

Thus, with this observation the k^{th} non-centered moments of a lognormal variable S with mean μ and variance σ^2 can computed as:

$$\mathbf{E}[S^k] = \mathbf{E}[e^{kY}] = \mathbf{E}[e^{k(\sigma Z+\mu)}] = e^{k\mu+(k\sigma)^2/2}\mathbf{E}[e^{(k\sigma)Z-(k\sigma)^2/2}] \tag{2.59}$$

$$\mathbf{E}[e^{(k\sigma)Z-(k\sigma)^2/2}] = \int_{-\infty}^{\infty} \frac{e^{-z^2/2+(k\sigma)z-k(\sigma)^2/2}}{\sqrt{2\pi}}dz = \int_{-\infty}^{\infty} \frac{e^{-(z-k\sigma)^2/2}}{\sqrt{2\pi}}dz = 1, \tag{2.60}$$

$$\mathbf{E}[S^k] = e^{k\mu+\frac{1}{2}k^2\sigma^2}. \tag{2.61}$$

From the moment of a lognormal variable, the moment of an asset $S(t)$ that follows the geometric Brownian motion in equation (2.18) can computed. Consider the first moment at $t = T$ and for the case of non-dividend paying stocks:

$$\mathbf{E}[S(T)] = \mathbf{E}[S(0)e^{(r-\sigma^2/2)*T+\sigma W(T)}] = S(0)e^{(r-\sigma^2/2)T}\mathbf{E}[e^{\sigma W(T)}]. \tag{2.62}$$

Recall **Definition 2.1.1.**3), then:

$$\mathbf{E}[e^{\sigma W(T)}] = e^{\sigma^2 T/2}. \tag{2.63}$$

Hence, the first moment is given by

$$\mathbf{E}[S(T)] = S(0)e^{rT}. \tag{2.64}$$

By using the fact that the product of a lognormal random variable is lognormal again, the second moment is in the samme manner given by:

$$\mathbf{E}[S(T)^2] = S(0)^2e^{(2r+\sigma^2)T}. \tag{2.65}$$

Thus, we can extend this to the case of the basket option $S_b(T)$ consisting of n correlated assets. The first moment M is easily derived by using (2.51):

$$M = \mathbf{E}[S_b(T)] = \mathbf{E}[\sum_{i=1}^{N} w_iS_i(t)] = \sum_{i}^{N} w_iF_i. \tag{2.66}$$

Since $S_b(T)$ is a sum of correlated variables, we need to know $\mathbf{E}[S_j(t)S_i(t)]$ in order to derive the second moment V^2 since it involves the product of correlated variables [19], [5], [20]. Consider the PDE:

$$d[S_i(t)S_j(t)] = S_i(t)dS_j(t) + S_j(t)dS_i(t) + dS_i(t)dS_j(t) \tag{2.67}$$

where $dS_i(t)$ respectively $dS_j(t)$ are given by (2.17). Taking the expectation on both the right hand and left hand side of (2.54) and recalling that $\mathbf{E}[dW(t)] = 0$:

$$d\mathbf{E}[S_i(t)S_j(t)] = (2r + \rho_{ij}\sigma_i\sigma_j)\mathbf{E}[S_i(t)S_j(t)]dt \tag{2.68}$$

$$\mathbf{E}[S_i(t)S_j(t)] = S_i(0)S_j(0)e^{(2r+\rho_{ij}\sigma_i\sigma_j)t}. \tag{2.69}$$

Hence, the second moment V^2 is given by:

$$V^2 = \mathbf{E}[S_b^2(T)] = \mathbf{E}[\sum_{i=1}^{N} w_iS_i(t)\sum_{j=1}^{N} w_jS_j(t)] = \sum_{i,j}^{N} w_iw_jF_iF_je^{\rho_{ij}\sigma_i\sigma_jT}. \tag{2.70}$$

The derivation is done in the same manner for the case of dividend-paying stocks. By using the second definition of the forward price F_i (2.43) we obtain the first two moments M and V^2, for an basket option with underlying stocks that pays dividends.

Finally, the moments of the basket option can be derived by matching the derived moments to the ones of a known distribution; the moment matching technique itself is arranged by matching the moments of a chosen distribution with the theoretical moments in (2.66). Finally, the distribution parameters can be derived and move further in the Black & Scholes setting to formulate the basket option price.

To conclude this section, analytical approximation demonstrates about considering the first two moments or more as input to the closed form solution approximation. The literature has so far used the moment matching technique and tested different given distributions to match the moments with:

- A log-normal random variable [28], [21], [17]
- A Inverse gamma random variable [10]
- Edgeworth expansion around the lognormal distribution [10], [12]
- A Johnson random variable [10]
- A log-extended-skew-normal random variable [29], [9], [24],

There is many papers on different distributions that has been tested, also several authors has tested the same. The author in this thesis has tried to find the most widely used, in addition there are certainty more papers than those mentioned that has tried an presented distribution. A good comparison of this can be done with [6] which is a recent paper on analytic approximations made this far in the literature.

2.3.2.1 Moment matching with a log normal variable

Matching the real distribution with more than two moments will not be considered in this thesis in order to keep the chosen price function as simple as possible. The suitable chosen pricing method is the moment matching method used in [17].

The distribution that will be used to approximate the basket option is the *log-normal distribution*. The idea is to match the first two moments of a log-normal variable e^X with mean M and variance $V^2 - M^2$, with the moments of the original distribution of the weighted sum of she stock prices $S_b(t)$. First, let us note that X is a normally distributed random variable with mean m and variance v^2 [17]:

$$\begin{aligned} m &= 2log(M) - 0.5log(V^2) \\ v^2 &= log(V^2) - 2log(M). \end{aligned} \tag{2.71}$$

The moments of the original distribution:

$$\begin{aligned} M &= \sum_{i=1}^{n} w_i F_i(T) \\ V^2 &= \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j F_i^T F_j^T e^{\sigma_i \sigma_j \rho_{ij} T}. \end{aligned} \tag{2.72}$$

Hence, the moment matching can be done by putting:

$$\begin{aligned} E(S_b(T)) &= E(e^X) = e^{m+0.5v^2} \\ E(S_b^2(T)) &= E(e^{2X}) = e^{2m+2v^2}. \end{aligned} \tag{2.73}$$

By solving the equation in (2.73) for the distribution parameters m and v, one can utilize the Black & Scholes formula to compute the price. Thus, the basket call option price is approximated by:

$$P_{Basket}(T) = e^{-rT} E[(S_b(T) - K)^+] = e^{-rT} M N(d_1) - K N(d_2) \tag{2.74}$$

$$\begin{aligned} d_1 &= \frac{m - ln(K) + v^2}{v} \\ d_2 &= d_1 - v. \end{aligned} \tag{2.75}$$

In the same manner, the formula for basket put option price can be approximated. The more detailed version of the derivation is omitted and can be found in [17] and [12].

Chapter 3

Optimization theory

3.1 Nonlinear optimization models

In order to go further with the modeling, optimization theory need to be defined and explained before we can set up and solve the optimization models of main interest in this thesis. What follows is an introduction to the optimization method used for the analysis of the proposed method and the algorithm behind chosen solvers, later used in the simulations.

The goal is to find the most price efficient basket option, by choosing the optimal combination of underlying stocks from a given set. In other words, the objective is to find the cheapest basket option by choosing the optimal combination of stocks. Thus, an pricing formula will be the objective function in the optimization model. We have seen in subsection 2.3.2 that the derived price function that will be used as the objective function is nonlinear. Hence, the optimization problem of interest is a nonlinear minimization problem.

Let $f(x)$ be the nonlinear function of several variables and hence a multidimensional, nonlinear minimization problem with a setup of inequality respectively equality constraints is formulated as follows:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & x \in \mathbb{N}. \end{aligned} \tag{3.1}$$

In general, a nonlinear optimization model is more difficult to solve than a linear optimization model, due to several reasons. Among others, the reason can be that different starting vectors may lead to different final solutions, which is related to the way solvers for nonlinear problems works [8]. Thus, it is also hard for an numerical solver to distinguish a local optimum from a global optimum, since numerical methods for solving nonlinear problems have limited information about the problem. For the solver, this is enough information to recognize when you are at an local point and not going further for an better solution, hence it is not easy to find a global optimum to an nonlinear problem.

3.1.1 Global and local optima

Global respectively local optimum to an optimization problem is generally defined as follows:

Definition 3.1.1. A point x^* is locally optimal if it is feasible and if there exists some $\epsilon > 0$ such that all feasible points x with $||x^* - x||_2 \leq \epsilon$ satisfy $f(x^*) \leq f(x)$.

Definition 3.1.2. A point x^* is globally optimal if it is feasible and if for all feasible points x satisfy $f(x^*) \leq f(x)$.

An important element of convex optimization which is presented in subsection 3.1.2 and derive their most utility, is that *for a convex optimization problem all locally optimal points are globally optimal*. The proof is omitted and can be found in [14].

3.1.2 Convex optimization

An optimization model can be classified as being a convex optimization model. The convexity property can make optimization in some sense "easier" than the general case. In order to understand the advantages associated with convex optimization problems, we define what an convex optimization problem is. A convex optimization model is given as an optimization problem of the form [14]:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & x \in \mathbb{N} \end{aligned} \tag{3.2}$$

where f and g are convex functions and h is an affine function.

3.1.2.1 Convex functions

Definition 3.1.3. A function $f : \mathbb{R}^n \Rightarrow \mathbb{R}$ is convex if its domain $D(f)$ is a convex set, and if for all $x, y \in D(f)$ and $0 \leq \theta \leq 1$

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y). \tag{3.3}$$

This definition shows that if we pick any two points on the graph of a convex function and draw a straight line between them, then the portion of the function between these two points will lie below this straight line. Furthermore, a function f is strictly convex if **Definition 3.1.3** holds with strict inequality for $x \neq y$ and $0 \leq \theta \leq 1$. In addition, f is concave if $-f$ is convex, and likewise f is strictly concave if $-f$ is strictly convex.

First order condition for convexity can then formulated according:
Suppose a function $f : \mathbb{R}^n \Rightarrow \mathbb{R}$ is differentiable. Then f is convex if and only if $D(f)$ is a convex set and for all $x, y \in D(f)$

$$f(y) \geq f(x) + \Delta_x f(x)^T (y - x). \tag{3.4}$$

The first order condition for convexity says that f is convex if and only if we take our function and draw a tangent line at any point. Every point on this line will lie below the corresponding point on f. Similar to the definition of convexity, f will be strictly convex if this holds with strict inequality. f will be concave if the inequality is reversed and strictly concave if the reverse inequality is strict.

The *second order condition for convexity* is formulated according to:
Suppose a function $f : \mathbb{R}^n \Rightarrow \mathbb{R}$ is twice differentiable. Then f is convex if and only if $D(f)$ is a convex set and its *Hessian* is positive semidefinite: i.e. any $x \in D(f)$:

$$\Delta_x^2 f(x) \succeq 0. \tag{3.5}$$

In one dimension, the condition is equivalent to that the second derivative $f''(x)$ always is non-negative. Furthermore, alike to both the definition and first order condition - f is strictly convex if its Hessian is positive definite, concave if the Hessian is negative semidefinite, and strictly concave if the Hessian is negative definite. The Hessian matrix $\mathbf{H}$ is a square matrix of second order partial derivatives of the function f. Suppose all second partial derivatives of f exists and are continuous over the domain of the function $D(f)$, then the Hessian matrix $\mathbf{H}$ of f is a nxn matrix defined component-wise as:

$$\mathbf{H}_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j} \tag{3.6}$$

The definiteness of the Hessian and nature of the stationary point which the Hessian is evaluated at, can be determined by the eigenvalues λ - which in this case is a $(nx1)$ of the Hessian - according to:

- If all eigenvalues of $\mathbf{H}$ are positive, the stationary point is a minimum point.
- If all eigenvalues of $\mathbf{H}$ are negative, the stationary point is a maximum point.
- If $\mathbf{H}$ has both positive and negative eigenvalues, the stationary point is a saddle point.

Lastly, a special case of convex functions will be defined. *Affine functions* are examples of convex functions of one variable. Let $f : \mathbb{R}^n \Rightarrow \mathbb{R}$. f is called affine if $f(x) = b^T x + c$ for some $b \in \mathbb{R}^n, c \in \mathbb{R}$. In this case the Hessian $\Delta_x^2 f(x) = 0 \ \forall x$. The zero matrix is both positive and negative semidefinite, hence f is both concave and convex.

3.1.2.2 Advantages of convex optimization

Convex optimization problems provides two advantages, in comparison to nonconvex problems. One main strength of these models is the guarantee of global optimality. In conjunction with a convex feasible region, a convex objective function (if minimizing) ensures that all local optima are global optima. If you get to a feasible solution and no neighboring solution is better, then the obtained solution is globally optimal. This allows local search algorithms to guarantee optimal solutions. Thus, without convexity a local search algorithm can converge to a suboptimal solution. Another nice advantage of convex optimization problems is that a convex feasible region makes it easier to ensure that you do not generate infeasible solutions, while searching for an optimum. If you have two feasible solutions, any solution within the line segment connecting them is feasible. This doesn't mean you can poke around at arbitrary distances in any arbitrary direction from a feasible solution, but it does facilitate local searches.

3.1.3 Optimality conditions

The Karush-Kuhn-Tucker (KKT) conditions are the *first order necessary conditions* for a solution in nonlinear programming to be optimal [14]. In general, many optimization algorithms can be interpreted as methods for numerically solving the KKT system of equations, since there is cases where a closed-form solution to the conditions cannot be derived. Such an algorithm will be presented and used in section (3.2).

Recall the minimization (primal) problem:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & x \in \mathbb{N} \end{aligned} \tag{3.7}$$

and define the Lagrangian as:

$$L(x, \lambda, \mu) = f(x) + \lambda^T h(x) + \mu^T g(x) \tag{3.8}$$

and define:

$$D(\mu, \lambda) = \min_{x \in \mathcal{R}^n} \; L(x, \mu, \lambda) \tag{3.9}$$

The dual problem is then:

$$\begin{aligned} \max_{\mu \in \mathcal{R}^m, \lambda \in \mathcal{R}^k} \quad & D(\mu, \lambda) \\ \text{s.t.} \quad & \mu \geq 0. \end{aligned} \tag{3.10}$$

Given primal feasible x and dual feasible (λ, u), the difference of the objective values in the primal and dual problem is defined as the duality gap:

$$\min_{x \in \mathcal{R}^n} \quad f(x) \quad - \max_{\mu \in \mathcal{R}^m, \lambda \in \mathcal{R}^k} \quad D(\mu, \lambda). \tag{3.11}$$

Suppose x^* and (λ^*, μ^*) are primal and dual solutions to their corresponding optimization problem, with zero duality gap. Then, (x^*) is local minimizer such that (λ^*, μ^*), called the Lagrangian multipliers, must satisfy the KKT conditions at (x^*, λ^*, μ^*):

1. $\Delta f(x^*) + \lambda^{*T} \Delta h(x^*) + \mu^{*T} \Delta g(x^*) = 0$ (stationarity)
2. $g(x^*) \leq 0$, $h(x^*) = 0$ (primal feasibility)
3. $\mu^* \geq 0$ (dual feasability)
4. $\mu^{*T} g(x^*) = 0.$ (complementary slackness)

Suppose x^* is a local minimum of (3.17), together with (λ^*, μ^*) satisfying the KKT conditions. If $d \neq 0$ is a feasible direction at x^* and the objective function $f(x)$ has zero slope in the direction

of d ($\Delta f(x^*) * d = 0$), then the curvature of the Lagrangian $L(x, \mu, \lambda)$ must be nonnegative in this direction:

$$d^T \Delta^2_{xx} L(x^*, \lambda^*, \mu^*) d \leq 0. \tag{3.12}$$

This condition is the *seconder order necessary condition* for x being a local minimizer. The directions for which $f(x)$ has zero slope is determined by using

$$\Delta f(x^*) * d = [\lambda^T \Delta h(x^*) + \mu^T \Delta g(x^*)] * d. \tag{3.13}$$

Since it is assumed that the KKT conditions holds in (x^*, μ^*, λ^*), the complementarity slackness and primal feasibility conditions hold. Thus, it follows that $(\Delta f(x^*) * d = 0)$ for a feasible direction d if and only if:

$$\begin{aligned} \Delta g(x^*) * d &= 0 \ \text{ when } \ \mu > 0 \\ \Delta g(x^*) * d &\geq 0 \ \text{ when } \ \mu = 0. \end{aligned} \tag{3.14}$$

Suppose at some feasible point x^* there are Lagrangian multipliers (μ^*, λ^*) so that the KKT conditions are satisfied. If the *seconder order sufficient condition* is satisfied x^* is a strict local minimizer:

$$d^T \Delta^2_{xx} L(x^*, \lambda^*, \mu^*) d > 0. \tag{3.15}$$

3.2 Binary optimization models

The class of nonlinear optimization models that will be of main interest in this thesis, is the binary optimization model. Binary, or Boolean, optimization models are classical combinatorial optimization problems [25]. In these problems, each variable x can take the value 0 or 1:

$$x_i \in \{0, 1\}, \ i = 1, \ldots, m. \tag{3.16}$$

Hence (3.17) is rewritten as a binary non-linear optimization problem with both equality and inequality constraints, with binary restriction on the variables to be optimized:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \{0, 1\} \ . \end{aligned} \tag{3.17}$$

The binary optimization problem is actually a type of integer programming problem. In general cases they are difficult to solve, requiring auxiliary search techniques - branch and bound and cutting plane methods for example [14]. As we will be dealing with an optimization model that is both binary and nonlinear, we understand from the argumentation in section 3.1 that this increases the complexity of solving such problems. Especially, the difficulty increases as the number of discrete variables increases [22].

3.2.1 Penalty method

Penalty methods aim to solve constrained optimization problems by a series of of unconstrained problems, ideally with solutions that will converge to the optimal solution of the original problem [14]. The idea follows by adding a *penalty function* in the objective function of the original problem - the idea by doing this is to replace an certain constraint. The penalty function consists of a penalty parameter multiplied by a measure of violation of the constraint; it has the property of prescribing a high cost for violation of the constraints.

The procedure follows by increasing the penalty parameter at each new iteration and using the solution from previous iteration as the initial guess. Solutions of these successive optimization problems will by this eventually converge to the solution of the original problem.

Barrier methods constitute an alternative class of algorithms for constrained optimization. These methods also add a penalty-like term to the objective function, but in this case the iterates are forced to remain interior to the feasible domain. The barrier is in place to bias the iterates to remain away from the boundary of the feasible region. Thus, the barrier function has the property of favoring points in the interior of the feasible region, over those near the boundary. As for the penalty method, at each new iteration the solution from previous iteration is used as the initial guess. Solutions of the the barrier problem will then eventually converge to the solution of the original constrained problem.

3.2.2 Relaxation of binary models

As argumented in [22], one of the challenging optimization problems is to determine the minimizer of a nonlinear programming problem that has binary variables. A difficulty is the work to solve such problems as the number of discrete variables increases. According to [22], a binary optimization problem can be transformed to a corresponding continuous nonlinear optimization problem. This is possible by adding the appropriate penalty term in the objective function (i.e. utilizing a penalty method) and relaxing the bounds the variables. Relaxing an constraint or bound means replacing it with an weaker constraint or bound on the variables. The idea is that the penalty term will force the variables to be 0 or 1 in order to compensate for the constraint that has been deleted. Hence, this will lead to that the solutions of this approximated, i.e. relaxed, optimization problem will eventually converge to the solution of the original problem.

With the suggested penalty method applied to the binary optimization problem, there is risk that local minimizers will exists at almost all feasible integer points. Therefore, it is not sufficient to introduce only the penalty term and hope to obtain the global minimizer by solving the resulting problem. It is highly likely that many undesired stationary points and local minimizers are being found in the process. It is therefore relevant to consider the so called logarithmic smoothing function.

Thus, the relaxation of the binary optimization model is extended by considering a logarithmic smoothing function, which is added in combination with the penalty term in the objective function. By adding a logarithmic smoothing function, smoothing of the original problem is applied. The original binary problem becomes replaced either by a single or a sequence of problems, whose

objective function is "smoother" - a function whose second or higher order derivatives are smaller in magnitude. The idea of smoothing is to add a strictly convex function to the original objective - hence obtaining a convex optimization problem (given that the feasible region is convex). Thus, for sufficiently large μ any local minimizer of the transformed optimization problem is also the unique global minimizer. The optimization problem will with this extension have fewer local minimum or even just one local minimum, obtaining a global optimization problem that is easier to solve.

The following relaxations of the binary optimization problem will be considered, whereas the two last suggestions are inspired by [22]:

1. Relaxation is applied to the binary constraint, for $\mathbf{1}$ being the $(nx1)$ vector of ones:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & 0 \leq x \leq \mathbf{1}. \end{aligned} \tag{3.18}$$

2. Relaxation is applied to the binary constraint and a penalty term, along with the penalty parameter $\mu > 0$ is added to the objective function:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) + \mu \sum_{j=1}^{N} x_j(1 - x_j) \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & 0 \leq x \leq \mathbf{1}. \end{aligned} \tag{3.19}$$

   For each iteration, the penalty term μ_k is increased. Since this is an optimization model based on the penalty method, the starting vector is the solution obtained in previous iteration.

3. An logarithmic smoothing function is added to the objective function, along with the relaxation and penalty term as in previous models:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) + \mu \sum_{j=1}^{N} x_j(1 - x_j) + \lambda \sum_{i=1}^{N} [-\ln(x_i) - \ln(1 - x_i)] \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & 0 \leq x \leq \mathbf{1} \end{aligned} \tag{3.20}$$

   The logarithmic smoothing function is given by $h(x) =$- $\ln x$ - $\ln(1 - x)$. The basic idea is to solve the problem for a decreasing sequence of $\lambda > 0$, starting with a large value and ending with one that may be close to zero. In addition, as before μ is an parameter that is increased in each iteration. Also, the starting vector is the solution obtained in previous iteration.

3.3 Algorithm for solving

Two suitable solvers has been tested and chosen for solving two different optimization problems for this thesis - a binary nonlinear problem and corresponding continuous problems. The algorithms that they are based on are:

3.3.1 Branch-and-bound method

The suitable and used algorithm for the binary optimization problem, from the solver KNITRO, is the a nonlinear branch and bound method. The branch-and-bound technique is an instance of the divide and conquer; a large problem is divided into smaller but few ones - this is the branch part. The algorithm is conquering by estimate how good a solution can be, obtained by each smaller problem. It may then be necessary to divide the problem further until reaching a problem that can be handled and solved - this is the bound part.

Not all nodes get expanded (i.e., their children generated). Rather, a carefully selected criterion determines which node to expand and when, and another criterion tells the algorithm when an optimal solution has been found. The detailed version of how the algorithm proceeds is found in the article written by Nemhauser and Wolsey (1989), found as an Chapter VI in the book [23].

The outline is as follows; B-B divides the problem into subproblems and tries to fathom each subproblem by the help of a relaxation. Relaxations of the subproblems have the following properties:

- All feasible solutions are also feasible in the relaxed problem.
- The optimal value of the relaxed problem is an upper bound of the optimal value of the original problem.
- There are cases when the optimal solution of the relaxed problem is also optimal in the original one

A subproblem is fathomed, meaning that no further branching is needed, in one of the following cases:

1. The optimal solution of the relaxed subproblem satisfies the constraints of the unrelaxed subproblem and its relaxed and non-relaxed objective function values are equal.
2. The infeasibility of the relaxed subproblem implies that the unrelaxed subproblem is infeasible as well.
3. The upper bound provided by the relaxed subproblem is less (in the case if alternative optimal solution are sought) or less or equal (if no alternative optimal solution is requested) than the objective function value of the best known feasible solution.

The algorithm can stop if all subsets (branches) are fathomed. Furthermore, the best bound rule is used for the bounding criterion. The best bound rule chooses the node with the smallest (or largest, in the case of a maximization problem) relaxed objective value. This strategy tends to reduce the number of nodes to be processed and can improve lower bounds quickly. Now, to the overall algorithm:

1. It selects a leaf of the branching tree, i.e. a subproblem not divided yet into further subproblems.

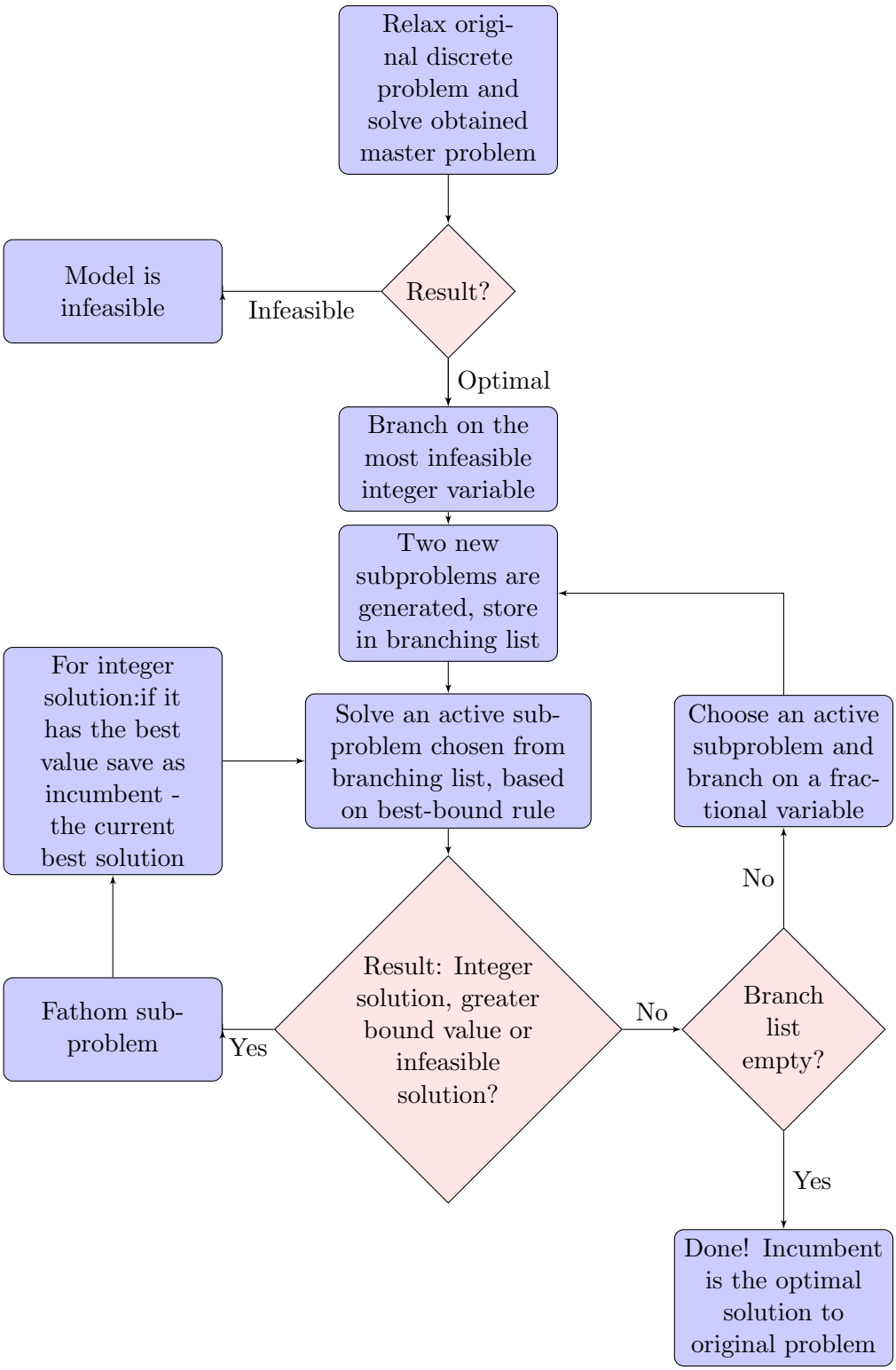
2. The subproblem is divided into further subproblems (branches) and their relaxations are defined.

3. Each new relaxed subproblem is solved and checked if it belongs to one of the above-mentioned cases. If so then it is fathomed and no further investigation is needed. If not then it must be stored for further branching.

4. If a new feasible solution is found which is better than the so far best one, then even stored branches having an upper bound less than the value of the new best feasible solution can be deleted without further investigation.

An summary of the algorithm is found in the next page, presented in a flow chart.

Since we have a nonlinear optimization problem, the sub-problems will be nonlinear as well. Those subproblems are continuous relaxations of the original MINLP problem, so an appropriate algorithm to solve these problems are one that solves nonlinear optimization problems. This will be the Interior/Direct algorithm, which is described in the next subsection.


Relax original discrete problem and solve obtained master problem
Result?
Infeasible
Model is infeasible
Optimal
Branch on the most infeasible integer variable
Two new subproblems are generated, store in branching list
Solve an active subproblem chosen from branching list, based on best-bound rule
For integer solution:if it has the best value save as incumbent - the current best solution
Result: Integer solution, greater bound value or infeasible solution?
Yes
Fathom subproblem
No
Branch list empty?
No
Choose an active subproblem and branch on a fractional variable
Yes
Done! Incumbent is the optimal solution to original problem

3.3.2 Interior-point method with direct step

The underlying algorithm used by KNITRO for continuous nonlinear optimization problems is interior-point method with the direct step. The interior point method always attempts to compute a new iterate, by solving the KKT system of equations using direct linear algebra - the direct step.

The approach to constrained minimization is to solve a sequence of approximate minimization problems. It replaces the nonlinear programming problem by a series of barrier subproblems, controlled by a barrier parameter. The barrier parameter is decreased for every subproblem, this process is repeated until the original problem has been solved to a desired accuracy. The specific interior-point method that is implemented in KNITRO and used in this thesis is presented in [27]

The algorithm proceeds as follows. The interior-point method solves a sequence of barrier minimization subproblems. First, let us define the original problem under consideration:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \\ & h(x) = 0, \\ & x \in \mathbb{N}. \end{aligned} \tag{3.21}$$

The interior method replaces (3.21) by a sequence of subproblems; for each μ bigger than 0, the approximate problem is then:

$$\begin{aligned} \min_{x \in \mathcal{R}^n} \quad & f(x) - \mu \ln(s) \\ \text{s.t.} \quad & g(x) + s = 0, \\ & h(x) = 0, \\ & x \in \mathbb{N}. \end{aligned} \tag{3.22}$$

The logarithmic term is called a barrier function and s is a vector of slack variables. We define $z = (x, s)$ and $\phi_\mu(z) = f(x) - \mu\, ln(s)$ and write the Lagrangian function for the subproblem as:

$$L(z, \lambda, \mu) = \phi_\mu(z) + \lambda_h^T h(x) + \lambda_g^T (g(x) + s) \tag{3.23}$$

where $\lambda = (\lambda_h, \lambda_g)$ are Lagrange multipliers. The first-order optimality conditions for the subproblem are the formulated as:

$$\begin{bmatrix} \Delta f(x) + A_h(x)^T \lambda_h + A_g(x)^T \lambda_g \\ S\Lambda_g e - \mu e \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{3.24}$$

$$\begin{aligned} & g(x) + s = 0 \\ & h(x) = 0 \\ & s \leq 0 \\ & \lambda_g \leq 0. \end{aligned} \tag{3.25}$$

To solve the approximate problem, the algorithm uses by default a direct step in z=(x,s) by applying Newtons method to the system given by he first-order optimality conditions.

This result in the primal-dual system given by:

$$\begin{bmatrix} W(z,\lambda;\mu) & A(x)^T \\ A(x) & 0 \end{bmatrix} = \begin{bmatrix} dz \\ d_\lambda \end{bmatrix} = - \begin{bmatrix} \Delta_z L(z,\lambda,\mu) \\ c(z) \end{bmatrix} \tag{3.26}$$

where

$$d_z = \begin{bmatrix} d_x \\ d_z \end{bmatrix}, d_\lambda = \begin{bmatrix} d_h \\ d_g \end{bmatrix}, c(z) = \begin{bmatrix} h(x) \\ g(x)+s \end{bmatrix}, A(x) = \begin{bmatrix} A_h(x) & 0 \\ A_g(x) & I \end{bmatrix}, W(z,\lambda;\mu) = \begin{bmatrix} \Delta^2_{xx} L(z,\lambda;\mu) & 0 \\ 0 & S^{-1}\Delta_g. \end{bmatrix} \tag{3.27}$$

The new iterate, also called Newton step, is given by

$$\begin{aligned} z* &= z + \sigma_z d_z \\ \lambda* &= \lambda + \sigma_\lambda d_\lambda \end{aligned} \tag{3.28}$$

where the parameters σ_z and σ_λ are steplengths computed in two stages. First, for $0 < \tau < 1$ the following is computed:

$$\begin{aligned} \sigma_z^{max} &= max\{\sigma \in (0,1] : s + \sigma d_s \geq (1-\tau)s\} \\ \sigma_\lambda^{max} &= max\{\sigma \in (0,1] : \lambda_g + \sigma d_g \geq (1-\tau)\lambda_g\}. \end{aligned} \tag{3.29}$$

Then, the algorithm uses a backtracking line search that compute the steplenghts to provide sufficient decrease in the merit function given by:

$$\begin{aligned} \sigma_z &\in (0, \sigma_z^{max}] \\ \sigma_\lambda &\in (0, \sigma_\lambda^{max}]. \end{aligned} \tag{3.30}$$

At each iteration the algorithm decreases a merit function

$$\phi_\mu(z) + v||c(z)||. \tag{3.31}$$

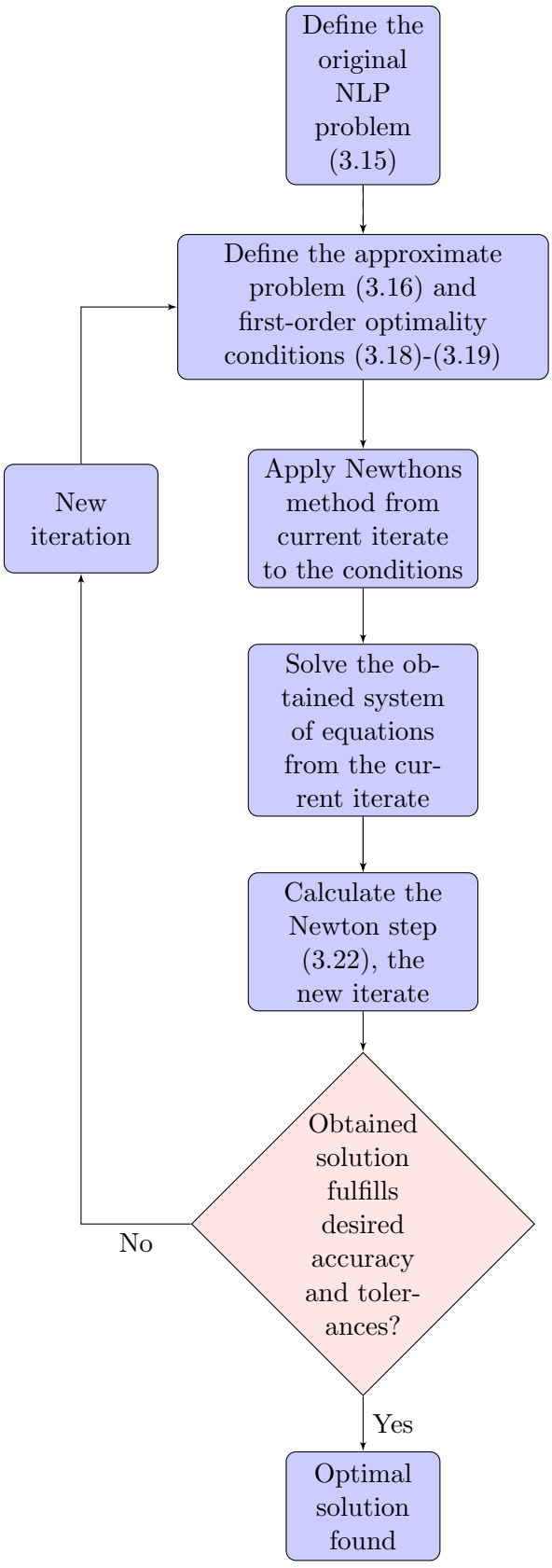
The parameter v may increase with iteration number in order to force the solution towards feasibility. If an attempted step does not decrease the merit function, the algorithm rejects the attempted step, and attempts a new step.

As μ decreases to zero which it must do, the minimum of f_{mu} should approach the minimum of f. μ is updated according to:

$$\mu_{k+1} = \frac{\mu_k}{100}. \tag{3.32}$$

The more detailed description of how the interior algorithm works in KNITRO is demonstrated by the paper [27], where the paper developed the new interior point algorithm that was specifically implemented in the KNITRO software package.

In the figure below follows an summarize of the algorithm in a flow chart.


Define the original NLP problem (3.15)
Define the approximate problem (3.16) and first-order optimality conditions (3.18)-(3.19)
Apply Newthons method from current iterate to the conditions
Solve the obtained system of equations from the current iterate
Calculate the Newton step (3.22), the new iterate
Obtained solution fulfills desired accuracy and tolerances?
No
New iteration
Yes
Optimal solution found

Chapter 4

The optimization model

4.1 Definition of objective function

We begin with deriving the ***objective function*** for the optimization model. The price function $P_{Basket}(T)$ derived in subsection 2.3.2 will be used in the optimization model as the objective function. ***The optimization variables will be the variables representing the weights*** $w \in \mathcal{R}^n$ ***in the price function*** **and will be modeled as binary variables**, since the focus is to find the optimal and price efficient combination of underlying assets for a basket option. In addition, the basket option consists of *stocks that pays dividends*. For the Monte Carlo simulation, the moment matching and therefore for the stochastic process that these two are based on, this implies that the formulas will be adjusted for stocks that pays dividends as have been presented in subsection 2.1.2.

Furthermore, recall the theory in subsection 2.1.2 where it was shown that the basket call respectively put option price are derived in the same manner, relying on the same mathematical theory. The only difference is the payoff function. Hence, it is not relevant to evaluate the optimization model for both a basket put and call option. In this thesis, only the *basket call option* will be considered.

Recall from subsection 2.1.2 given by the equations (2.66)-(2.70), that for $w \in \mathcal{R}^n$ the expected price $P_{Basket}(w)$ at the maturity time T for a European call option, derived by moment matching with the lognormal distribution, and the adjusted Black & Scholes parameters $d_1(w)$ and $d_2(w)$ are given by:

$$P_{Basket}(w) = e^{-rT}[M(w)N(d_1(w)) - KN(d_2(w))] \tag{4.1}$$

$$\begin{aligned} d_1(w) &= \frac{m(w) - \ln(K) + v^2(w)}{v(w)} \\ d_2(w) &= d_1(w) - v(w), \end{aligned} \tag{4.2}$$

where the first two moments for a lognormal variable, $m(w)$ and $v(w)^2$ respectively for the basket option, $M(w)$ and $V(w)^2$ are given by:

$$\begin{aligned} m(w) &= 2\log[M(w)] - 0.5\log[V^2(w)] \\ v^2(w) &= \log[V(w)^2] - 2\log[M(w)] \\ M(w) &= \sum_{i=1}^{n} w_i F_i \\ V^2(w) &= \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j F_i^T F_j^T e^{\sigma_i \sigma_j \rho_{ij} T} \end{aligned} \tag{4.3}$$

In addition, recall that the remaining parameters are given by:

- r is a given annual interest rate for the period $t = 0$ to $t = T$
- F_k is the forward price of asset k
- σ_k is the volatility for asset k - the standard deviation of the stock's returns
- ρ_{ij} is the constant correlation for asset i and asset j

In this thesis, the option into consideration is an *at-the-money option*. This implies that the strike price K becomes a function of the weights w in (4.1) and (4.2) according to:

$$K(w) = \sum_{i=1}^{n} w_i S_i(0), \tag{4.4}$$

where $S_i(0)$ is the spot price for asset i.

The price of the basket option given by (4.1) is dependent on the magnitude of the forward prices of the underlying assets, by observing the appearance of $M(w) = \sum_{i=1}^{n} w_i F_i$ in the price function. A solution to this is to standardize the basket option price with the nominal value of the basket option:

$$\%_{Basket} = \text{Basket option return} = \frac{\text{Option price } P_{Basket}(w)}{\text{Nominal value}}. \tag{4.5}$$

Since we are dealing with an at-the-money option, the nominal value is equal to the strike of the basket option which is the current price of the basket option, as have been demonstrated:

$$\text{Nominal value} = K(w) = \sum_{i=1}^{n} w_i S_i(0). \tag{4.6}$$

This can be generalized for other types of options. For a constant q which is the percentage strike price, define:

$$K(w) = \text{Nominal value} * q \tag{4.7}$$

where $q > 1$ for an out-of-money option and $q < 1$ for an in-the-money option. For an at-the-money option $q = 1$, which is the case in this thesis.

Hence, dividing (4.1) with (4.6) the price function $P_{Basket}(w)$ is adjusted for dependence on the magnitude of prices - as well as making the option price more sensitive to the number of underlying assets:

$$P_{Basket}(w) = \frac{e^{-rT}[M(w)N[d_1(w)] - K(w)N[d_2(w)]]}{K(w)}. \tag{4.8}$$

Moreover, the underlying assets can have different spot prices or returns. Since the objective is to obtain an equally-weighted basket option, we need to magnitude-adjust each weight variable w_i in $P(w)$ according to:

$$\frac{K(w^0)}{S_i(0)}\frac{w_i}{W} \tag{4.9}$$

where $K(w^0)$ is an initial value of the nominal price for the basket option, for an arbitrary chosen w^0. Observe that by dividing $\frac{w_i}{W}$, the weight variables w which are binary variables will be compatible with the price function. This is due to the assumption that the weights for the underlying assets in a basket option should sum up to 1. Obviously, each weight will be normalized with the desired number of stocks W for the price efficient basket option (this constraint will be presented in next section 4.2.)

Finally, ***the objective function*** is now correctly formulated as follows:

$$\begin{aligned}
P_{Basket}(w) &= \frac{e^{-rT}[M(w)N[d_1(w)] - K(w)N[d_2(w)]]}{\sum_{i=1}^{n}\frac{K(w^0)}{S_i(0)}\frac{w_i}{W}S_i(0)} \\
m(w) &= 2\log[M(w)] - 0.5\log[V^2(w)] \\
v^2(w) &= \log[V(w)^2] - 2\log[M(w)] \\
M(w) &= \sum_{i=1}^{n}\frac{K(w^0)}{S_i(0)}\frac{w_i}{W}F_i \\
V^2(w) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{K(w^0)}{S_i(0)}\frac{w_i}{W}\frac{K(w^0)}{S_j(0)}\frac{w_j}{W}F_i^T F_j^T e^{\sigma_i\sigma_j\rho_{ij}T} \\
d_1(w) &= \frac{m(w) - \ln(K) + v^2(w)}{v(w)} \\
d_2(w) &= d_1(w) - v(w).
\end{aligned} \tag{4.10}$$

4.2 Definition of constraints

The last part for the formulation of the optimization model is to define the ***constraints***. Since we are only interested in one share of each stock, the variables representing number of underlying assets in the basket option will be binary as previously mentioned:

$$w \in \{0,1\}^n \tag{4.11}$$

The key is to choose the best W number of stocks from a given set of underlying assets with n number of assets, which is formulated as

$$\sum_{i=1}^{n} w_i = W \tag{4.12}$$

Finally, in order to support a diversified combination of stocks in the basket option, there will be a numerical limitation Z on the number of stocks out of $m_j > Z$ available stocks that exists in section j. This is formulated as:

$$\sum_{i=1}^{m_j} w_i \leq Z \quad \forall j = 1,2,3,... \tag{4.13}$$

where this limitation holds for the weight w_i that belongs to section j.

4.3 Method and model setup

4.3.1 Binary nonlinear optimization problem

The optimization model can now be defined, where all the parameters and constants have been defined in previous section:

$$\begin{aligned} \min_{w \in \mathcal{R}^n} \quad & \frac{e^{-rT}[M(w)N[d_1(w)] - K(w)N[d_2(w)]]}{\sum_{i=1}^{n} \frac{K(w^0)}{S_i(0)} \frac{w_i}{W} S_i(0)} \\ \text{s.t.} \quad & w \in \{0,1\}^n \\ & \sum_{i=1}^{N} w_i = W \\ & \sum_{\forall i \in S_j} w_i \leq Z \end{aligned} \tag{4.14}$$

where:

$$
\begin{aligned}
m(w) &= 2\log[M(w)] - 0.5\log[V^2(w)] \\
v^2(w) &= \log[V(w)^2] - 2\log[M(w)] \\
M(w) &= \sum_{i=1}^{n} \frac{K(w^0)}{S_i(0)} \frac{w_i}{W} F_i \\
V^2(w) &= \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{K(w^0)}{S_i(0)} \frac{w_i}{W} \frac{K(w^0)}{S_j(0)} \frac{w_j}{W} F_i^T F_j^T e^{\sigma_i \sigma_j \rho_{ij} T} \\
d_1(w) &= \frac{m(w) - \ln(K) + v^2(w)}{v(w)} \\
d_2(w) &= d_1(w) - v(w).
\end{aligned}
\tag{4.15}
$$

This problem is a binary nonlinear optimization problem. It is a nonconvex optimization problem, since the objective function is not a convex function. The linear inequality and equality constraints are affine functions, hence the feasible region is convex. The binary nonlinear optimization problem will be solved with the Branch-and-bound method, described in subsection 3.3.1 provided by the solver KNITRO. The results from the optimization model will be compared with the results from an algorithm that finds the price efficient solution by searches through all possible combinations.

As presented and argumented in section 3.1 and section 3.2 for nonlinear respectively binary optimization models, this optimization model which is a combination of both can be hard to solve - especially as the number of binary variable increases. Among others, a global optimum can be hard to find and different starting vectors may lead to different final solutions [8]. Thus, it is suitable to consider how the the binary optimization model can be approximated by an continuous optimization model, which may be easier to solve.

4.3.2 Continuous nonlinear optimization models

Three different relaxations of the binary optimization model given in (4.14) along with the same parameters as in (4.15) will be considered, for which all the binary constraints $w_i \in [0, 1]$ is relaxed. All these optimization problems will be solved with the Interior point method presented in subsection 3.3.2, provided by the solver KNITRO. The main ideas for the different relaxed optimization problems have been presented and motivated more in detail in section 3.2. Also, the basic ideas that they are based on are presented in section 3.1.

Three different continuous optimization models will be considered which are based on three different relaxations of the binary optimization model. For $\mathbf{1}$ being the $(nx1)$ vector of ones, the following three continuous nonlinear optimization models will be solved:

1. Continuous NLP model 1, where the binary constraint has been relaxed:

$$\begin{aligned} \min_{w\in\mathcal{R}^n} \quad & P_{Basket}(w) = \frac{e^{-rT}[M(w)N[d_1(w)] - K(w)N[d_2(w)]]}{\sum_{i=1}^{n} \frac{K(w^0)}{S_i(0)} \frac{w_i}{W} S_i(0)} \\ \text{s.t.} \quad & 0 \leq w \leq \mathbf{1} \\ & \sum_{i=1}^{N} w_i = W \\ & \sum_{\forall i \in S_j} w_i \leq Z \end{aligned} \tag{4.16}$$

2. First, define

$$\Phi(w) = \frac{e^{-rT}[M(w)N[d_1(w)] - K(w)N[d_2(w)]]}{\sum_{i=1}^{n} \frac{K(w^0)}{S_i(0)} \frac{w_i}{W} S_i(0)} \tag{4.17}$$

   then for $\mu > 0$ the continuous NLP model 2 is defined as:

$$\begin{aligned} \min_{w\in\mathcal{R}^n} \quad & \Phi(w) + \mu \sum_{j=1}^{n} w_j(1 - w_j) \\ \text{s.t.} \quad & 0 \leq w \leq \mathbf{1} \\ & \sum_{i=1}^{N} w_i = W \\ & \sum_{\forall i \in S_j} w_i \leq Z \end{aligned} \tag{4.18}$$

   where the binary constraint has been relaxed and an penalty function has been added to the objective function. The penalty function consists of a penalty parameter multiplied by a measure of violation of the constraint. For each iteration, an increased value of μ is chosen and the starting vector is the obtained solution from previous iteration.

3. In order to avoid the difficulty of solving an optimization problem with several local optimum, which can be the case for continuous NLP 2, we extend (4.18) by introduce an logarithmic smoothing function in objective function. By adding a logarithmic smoothing function, smoothing of the original problem is applied.

   The idea of smoothing is to add a strictly convex function to the original objective - hence obtaining a convex optimization problem (given that the feasible region is convex). In this way, instead of having several local optimum, we can obtain an optimization problem have fewer local minimum - or even just one local minimum. This can make it easier to obtain a global optimization problem that is easier to solve.

Hence, for $\lambda > 0$ the continuous NLP model 3 is:

$$\begin{aligned} \min_{w \in \mathcal{R}^n} \quad & \Phi(w) + \mu \sum_{j=1}^{n} w_j(1 - w_j) + \lambda \sum_{i=1}^{n} [-\ln(w_i) - \ln(1 - w_i)] \\ \text{s.t.} \quad & 0 \leq w \leq \mathbf{1} \\ & \sum_{i=1}^{N} w_i = W \\ & \sum_{\forall i \in S_j} w_i \leq Z \end{aligned} \tag{4.19}$$

In each iteration, λ is decreased and μ is increased as in NLP 2. Furthermore, the starting vector is the solution obtained in previous iteration. In order to evaluate the logarithmic smoothing method the Hessian will be numerical simulated for various feasible w, to see whether the objective function in (4.19) is convex. The linear inequality and equality constraints are affine functions, hence the feasible region is convex. Thus, if the objective function is convex then the obtained optimal solution can be claimed to be globally optimal. The optimization problem NLP 3 is considered to be a convex optimization problem, if both the objective function and feasible set is convex.

Chapter 5

Results and analysis

This chapter will first present some input data and the association of the optimization variables to an stock of a company. Then the results and analysis for the optimization will be presented, which is done separately for the big respectively small dimension of the problem. Furthermore, in section 5.4 an sensitivity analysis has been done on the parameter W in order to analyze how the number of combinations will affect the results and the different optimizations models' performance. Last, results from an Monte Carlo simulation for correlated Geometric Brownian Motion is presented to see how the chosen pricing method of basket option performs.

5.1 Input data

Parameter	**Small dimension**	**Big dimension**
n	10	30
W	5	15
Z	2	4
j	1	2
m_j	4	[9, 10]
r	-0.05%	-0.05%
T	5	5

Table 5.1: Input data for the parameters in the optimization models

The input data for the forward prices, correlation matrix and spot prices for the underlying assets can be found in Appendix C. Furthermore, the stocks that will be used as underlying assets in the problem with a small $(w_1 - w_{10})$ respectively big dimension $(w_1 - w_{30})$ of variables are given in Table 5.2. The chosen stocks are the 30 most traded stocks in OMX Stockholm in 2016, often referred to as OMXS30.

Stock	**Weight**
H&M AB	w_1
Nordea Bank AB	w_2
Swedbank AB	w_3
Svenska Handelsbanken AB	w_4
Atlas Copco A	w_5
Assa Abloy AB	w_6
SEB AB	w_7
LM Ericsson AB	w_8
Svenska Cellulosa AB	w_9
Telia Co AB	w_{10}
Volvo AB	w_{11}
Investor A	w_{12}
Hexagon AB	w_{13}
Sandvik AB	w_{14}
Investor B	w_{15}
Atlas Copco B	w_{16}
ABB Ltd	w_{17}
Skanska AB	w_{18}
AstraZeneca PLC	w_{19}
Melker Schorling AB	w_{20}
Electrolux AB	w_{21}
Autoliv Inc	w_{22}
SKF AB	w_{23}
ICA Gruppen AB	w_{24}
Swedish Match AB	w_{25}
Alfa Laval AB	w_{26}
Securitas AB	w_{27}
Investment AB,Latour	w_{28}
Boliden AB	w_{29}
Kinnevik AB	w_{30}

Table 5.2: The stocks and their corresponding weight variable

5.2 Small dimensional basket option

The result for the algorithm that manually finds the optimal combination is presented in Table 5.3. This solution is the global optimal solution that we can refer and compare with the results from the optimization models, since it iterates through all possible combinations. In other words, this solution will be the *price efficient solution* which we will refer to in the remaining part of the subsection. The optimization models may give different optimal solutions, depending on the starting vector to the problem. Therefore this result is vital in order to avoid comparisons of several different optimal solutions obtained for various starting vectors.

The results combination is obtained by finding the solution to the problem by:

- Generating all possible unique permutations that does not have more than two stocks from each available sector.
- Calculating the basket option price for each combination.
- Find the combination yielding the lowest price.

w^*	
w_1	1.000
w_2	0.000
w_3	1.000
w_4	0.000
w_5	0.000
w_6	0.000
w_7	1.000
w_8	1.000
w_9	0.000
w_{10}	1.000
$P_{Basket}(w^*)$	10.640%

Table 5.3: The combination w that yields the minimum value of $P_{Basket}(w)$

Table 5.4 briefly presents the results obtained by the optimization models. This follows for the binary nonlinear optimization problem and the three different continuous versions of the binary problem - continuous NLP 1, NLP 2 respectively NLP 3. For simplicity, the objective function value for each optimization model is denoted by $f(w^*)$. As can be observed, the obtained optimal solutions w^* from all the models, except from NLP 1 (consider the rounded solution of w^* in NLP 1), are exactly the same. It is the same as the one obtained when finding the price efficient solution manually, given in Table 5.3. In addition, observe that the rounded solution of NLP 1 is infeasible since it exceeds the number of stocks that can only be 5:

$$w^* = \begin{bmatrix} 1.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 1.000 & 1.000 & 1.000 & 1.000 \end{bmatrix}^T. \tag{5.1}$$

Furthermore, from Table 5.4 another observation is that $f(w^*)$ is the same for the binary NLP and continuous NLP 2 - which is the price function value also obtained in Table 5.3. $f(w^*)$ is slightly larger for continuous NLP 3, but $P_{Basket}(w^*)$ is the same as for binary NLP and continuous NLP 2, since the obtained optimal solution w^* is the same. By this observation, together with the results which shows that NLP 2 and NLP 3 have the same optimal solution as the binary problem, we can conclude that the models NLP 2 and NLP 3 are good approximations to the binary problem.

w^*	**Binary NLP**	**Continuous NLP 1**	**Continuous NLP 2**	**Continuous NLP 3**
w_1	1.000	0.708	1.000	1.000
w_2	0.000	0.000	0.000	0.000
w_3	1.000	1.000	1.000	1.000
w_4	0.000	0.0221	0.000	0.000
w_5	0.000	0.178	0.000	0.000
w_6	0.000	0.000	0.000	0.000
w_7	1.000	0.506	1.000	1.000
w_8	1.000	1.000	1.000	1.000
w_9	0.000	0.586	0.000	0.000
w_{10}	1.000	1.000	1.000	1.000
$f(w^*)$	10.640 %	10.464 %	10.640 %	10.650 %
$P_{Basket}(w^*)$	10.640 %	10.464 %	10.640 %	10.640 %

Table 5.4: Optimal solution and corresponding objective value for the optimization models

Results from the continuous NLP 2 are shown in Table 5.5, 5.6 and 5.7. w^*, $f(w^*)$ and $P_{Basket}(w^*)$ are shown for each iteration for an increasing value of μ. The optimization was carried out for several starting vectors w^0, whereas the result was the same. Therefore we display the result for one w^0. The optimization is terminated when the obtained optimal solution have integer values - or close to integer values as possible, without increasing the the objective value $f(w^*)$. As shown in all the tables, integer values for thr weights in w were are obtained for $\mu = 0.01$, but the price efficient solution w^* is obtained at $\mu = 0.1$.

In Table 5.5 the starting vector w_0 is the unrounded solution that was obtained in continuous NLP 1. Furthermore, similar and very alike values for w^* and $f(wj)$ at each iteration, as in Table 5.5, are obtained with different starting vectors - carried out with the same values of increasing μ. In all of these cases, the optimal solution converges to the same solution w^* at $\mu^* = 0.01$ as in Table 5.5. This is however not the price efficient solution, therefore other values for μ needed to be tested for in order to see if it was possible to find the correct and price efficient solution. This is presented in Tables 5.6 and 5.7.

w^*	w^0	$\mu = 0.0001$	$\mu = 0.001$	$\mu = 0.01$
w_1	0.708	0.724	0.952	1.000
w_2	0.000	0.000	0.000	0.000
w_3	1.000	1.000	1.000	1.000
w_4	0.0221	0.005	0.000	0.000
w_5	0.178	0.167	0.000	0.000
w_6	0.000	0.000	0.000	0.000
w_7	0.506	0.519	0.523	0.000
w_8	1.000	0.997	1.000	1.000
w_9	0.586	0.588	0.524	1.000
w_{10}	1.000	1.000	1.000	1.000
$f(w^*)$		10.472 %	10.536 %	10.656 %
$P_{Basket}(w^*)$		10.464 %	10.481 %	10.656 %

Table 5.5: Optimal solution and objective value from Continuous NLP 2 for increasing μ

In Table 5.6 a different optimal solution was obtained at $\mu = 0.1$, than the one obtained in Table 5.5 for $\mu = 0.01$. The difference here is that μ is increased by a factor of 100 instead of 10 in the last iteration - the violation of the constrains is in this case punished more by increasing it with factor 100. The obtained solution is the correct, i.e. price efficient, solution with a lower objective function value $f(w^*)$ than for the one obtained in 5.5. The same value is obtained with the original price function, hence $P_{Basket}(w^*) = f(w^*)$. In Table 5.7 the same results are obtained in each iteration, when increasing μ with a factor of 100 starting from the initial iteration. Hence, increasing μ with 100 seems to be a the optimal strategy. As mentioned before, the objective function in continuous NLP 2 is a good approximation of the objective function in the binary NLP, since similar values (comparing $f(w^*)$ and $P_{Basket}(w^*)$) are obtained for several w observed in Table 5.5, 5.6 and 5.7.

w^*	w^0	$\mu = 0.0001$	$\mu = 0.001$	$\mu = 0.1$
w_1	0.708	0.724	0.952	1.000
w_2	0.000	0.000	0.000	0.000
w_3	1.000	1.000	1.000	1.000
w_4	0.0221	0.005	0.000	0.000
w_5	0.178	0.167	0.000	0.000
w_6	0.000	0.000	0.000	0.000
w_7	0.506	0.519	0.523	1.000
w_8	1.000	0.997	1.000	1.000
w_9	0.586	0.588	0.524	0.000
w_{10}	1.000	1.000	1.000	1.000
$f(w^*)$		10.472 %	10.536 %	10.640 %
$P_{Basket}(w^*)$		10.464 %	10.481 %	10.640 %

Table 5.6: Optimal solution and objective value from Continuous NLP 2 for increasing μ

w^*	w^0	$\mu = 0.00001$	$\mu = 0.001$	$\mu = 0.1$
w_1	0.708	0.711	0.952	1.000
w_2	0.000	0.000	0.000	0.000
w_3	1.000	1.000	1.000	1.000
w_4	0.0221	0.020	0.000	0.000
w_5	0.178	0.178	0.000	0.000
w_6	0.000	0.000	0.000	0.000
w_7	0.506	0.508	0.523	1.000
w_8	1.000	0.996	1.000	1.000
w_9	0.586	0.588	0.524	0.000
w_{10}	1.000	1.000	1.000	1.000
$f(w^*)$		10.464 %	10.536 %	10.640 %
$P_{Basket}(w^*)$		10.464 %	10.481 %	10.640 %

Table 5.7: Optimal solution and objective value from Continuous NLP 2 for increasing μ

Furthermore, the Hessian for the objective function in NLP 2 was numerical simulated and the eigenvalues for the Hessian and some analysis of the results can be found in Appendix A. In this case, all eigenvalues were negative for several numerical simulations - which is not weird since the penalty term is a concave function, for increasing μ this term becomes more dominant and hence the eigenvalues becomes more negative. Hence, the Hessian is a negative definite matrix.

The results from continuous NLP 3 are given in Table 5.8, 5.9 and 5.10 for increasing values of μ and decreasing values of λ. The same results are obtained, with different starting vectors. Hence the results are shown for one starting vector - which is the unrounded solution in NLP 1. However, testing with different values of decreasing λ and increasing μ to find the optimal solution yields different solution - by comparing Table 5.8, 5.9 and 5.10. The same solution, which is not price efficient, is found in 5.9 as the one found in Table 5.5 - it is a local optimal solution but not the global (price efficient).

Comparing the result in Table 5.8 with the results in 5.9 we see that a good intuition is to begin with at least $\mu = \lambda$ and not $\mu < \lambda$. In addition, a good intuition is also to start with the same initial value μ^0 as observed in NLP 2 - not with $\mu = 0.1$ right away. This strategy does not yield the price efficient solution, tested with several starting vectors as well - the results can be observed in 5.10. A good idea is to let μ increase at the same time as decreasing λ, instead of starting with the value for μ that has yielded integer values in NLP 2.

w^*	w^0	$\mu = \lambda = 0.0001$	$\mu = 0.001, \lambda = 10^{-5}$	$\mu = 0.01, \lambda = 10^{-6}$
w_1	0.708	0.669	0.900	1.000
w_2	0.000	0.009	0.001	0.000
w_3	1.000	0.984	0.999	1.000
w_4	0.0221	0.138	0.010	0.000
w_5	0.178	0.219	0.031	0.000
w_6	0.000	0.022	0.002	0.000
w_7	0.506	0.455	0.519	1.000
w_8	1.000	0.919	0.991	1.000
w_9	0.586	0.595	0.547	0.000
w_{10}	1.000	0.990	0.999	1.000
$f(w^*)$		10.804 %	10.588 %	10.650 %
$P_{Basket}(w^*)$		10.512 %	10.478 %	10.640 %

Table 5.8: Optimal solution and objective value from Continuous NLP 3 for decreasing λ and increasing μ

w^*	w^0	$\mu = 0.0001, \lambda = 0.1$	$\mu = 0.001, \lambda = 0.001$	$\mu = 0.01, \lambda = 10^{-5}$	$\mu = 0.1, \lambda = 10^{-6}$
w_1	0.708	0.501	0.564	0.999	1.000
w_2	0.000	0.481	0.071	0.000	0.000
w_3	1.000	0.509	0.879	0.999	1.000
w_4	0.0221	0.496	0.297	0.001	0.000
w_5	0.178	0.498	0.341	0.001	0.000
w_6	0.000	0.491	0.138	0.001	0.000
w_7	0.506	0.499	0.445	0.001	0.000
w_8	1.000	0.508	0.779	0.999	1.000
w_9	0.586	0.502	0.569	0.998	1.000
w_{10}	1.000	0.515	0.917	0.999	1.000
$f(w^*)$		151.216 %	12.874 %	10.736 %	10.669 %
$P_{Basket}(w^*)$		12.528 %	10.834 %	10.656 %	10.656 %

Table 5.9: Optimal solution and objective value from Continuous NLP 3 for decreasing λ and increasing μ

w^*	w^0	$\mu = 0.01, \lambda = 0.01$	$\mu = 0.01, \lambda = 0.0001$	$\mu = 0.01, \lambda = 10^{-6}$
w_1	0.708	0.509	0.988	1.000
w_2	0.000	0.309	0.005	0.000
w_3	1.000	0.611	0.994	1.000
w_4	0.0221	0.455	0.012	0.000
w_5	0.178	0.471	0.011	0.000
w_6	0.000	0.391	0.007	0.000
w_7	0.506	0.490	0.015	0.000
w_8	1.000	0.584	0.990	1.000
w_9	0.586	0.516	0.984	1.000
w_{10}	1.000	0.663	0.995	1.000
$f(w^*)$		28.571 %	11.226 %	10.666 %
$P_{Basket}(w^*)$		11.894 %	10.658 %	10.656 %

Table 5.10: Optimal solution and objective value from Continuous NLP 3 for decreasing λ (and increasing μ)

The Hessian was numerical simulated and the results and some analysis can be found in Appendix A. For several feasible w and for each simulation the results showed that the eigenvalues were all positive - hence the Hessian for the objective function is positive definite. This means that the logarithmic smoothing method succeeds in order to make the continuous NLP 3 an global optimization problem.

One can conclude that the optimal solution obtained by the binary NLP, continuous NLP 2 and NLP 3 are global optimal solutions, since the same result is obtained by the algorithm that seeks through all possible and feasible combinations. In addition, the objective function in NLP 3 is shown to be convex by numerical simulations of the Hessian which was shown to be positive definite.

5.3 Large dimensional basket option

Table 5.11 shows the results from the four optimization models. For simplicity, the objective function value for each optimization model is denoted by $f(w^*)$. As can be observed, all the optimization models yields the same optimal solution w^*; in NLP 1 and NLP 3 for slightly different objective function values. Note that the solution obtained in continuous NLP 1 is feasible when being rounded which can be observed in Table 5.11. As in the small dimensional model, the optimal objective value $f(w^*)$ was the same for binary NLP and continuous NLP 2. By this observation and together the results which shows that all continuous models have the same optimal solution as the binary optimization problem, we can conclude that the models NLP 1, NLP 2 and NLP 3 are good approximations to the binary problem. Of course, several starting vectors have been tested for all the different optimization models; the same results were obtained.

Overall, when comparing to the small dimensional problem all the models have yield the same optimal solution regarding the starting vector. In addition, the result was obtained much easier and straight forward without being forced to tries different strategies for (λ, μ) in NLP 3 and too many starting vectors - as for the small dimensional problem. Note that in this quite big dimensional problem, an comparison with an algorithm that searches all possible combinations was not possible.

w^*	Binary NLP	Continuous NLP 1	Continuous NLP 2	Continuous NLP 3
w_1	1.000	1.000	1.000	1.000
w_2	0.000	0.000	0.000	0.000
w_3	0.000	0.000	0.000	0.000
w_4	0.000	0.000	0.000	0.000
w_5	0.000	0.237	0.000	0.000
w_6	0.000	0.000	0.000	0.000
w_7	0.000	0.000	0.000	0.000
w_8	1.000	1.000	1.000	1.000
w_9	1.000	1.000	1.000	1.000
w_{10}	1.000	1.000	1.000	1.000
w_{11}	0.000	0.000	0.000	0.000
w_{12}	1.000	1.000	1.000	1.000
w_{13}	0.000	0.000	0.000	0.000
w_{14}	0.000	0.000	0.000	0.000
w_{15}	0.000	0.000	0.000	0.000
w_{16}	0.000	0.000	0.000	0.000
w_{17}	1.000	1.000	1.000	1.000
w_{18}	1.000	1.000	1.000	1.000
w_{19}	1.000	0.999	1.000	1.000
w_{20}	1.000	1.000	1.000	1.000
w_{21}	1.000	0.542	1.000	1.000
w_{22}	0.000	0.222	0.000	0.000
w_{23}	0.000	0.000	0.000	0.000
w_{24}	1.000	1.000	1.000	1.000
w_{25}	1.000	1.000	1.000	1.000
w_{26}	0.000	0.000	0.000	0.000
w_{27}	1.000	1.000	1.000	1.000
w_{28}	1.000	1.000	1.000	1.000
w_{29}	0.000	0.000	0.000	0.000
w_{30}	1 .000	1 .000	1.000	1.000
$f(w^*)$	6.532%	6.508%	6.532%	6.564 %
$P_{Basket}(w^*)$	6.532%	6.508%	6.532%	6.532 %
Solving time	10 s	10 s	10 s	18 s

Table 5.11: Optimal solution and corresponding objective value for the optimization models

Results from continuous NLP 2 are shown in Table 5.12, for iterations with increasing μ. The same result was obtained with several starting vectors. In both these examples optimal solution w^* converged to the same. In Table 5.12 the results are showed for the starting vector w^0 being the unrounded solution obtained in continuous NLP 1. Integer values were first obtained for $\mu = 0.001$ in all cases of starting vectors.

w^*	w^0	$\mu = 0.0001$	$\mu = 0.001$
w_1	1.000	1.000	1.000
w_2	0.000	0.000	0.000
w_3	0.000	0.000	0.000
w_4	0.000	0.000	0.000
w_5	0.237	0.217	0.000
w_6	0.000	0.000	0.000
w_7	0.000	0.000	0.000
w_8	1.000	1.000	1.000
w_9	1.000	1.000	1.000
w_{10}	1.000	1.000	1.000
w_{11}	0.000	0.000	0.000
w_{12}	1.000	1.000	1.000
w_{13}	0.000	0.000	0.000
w_{14}	0.000	0.000	0.000
w_{15}	0.000	0.000	0.000
w_{16}	0.000	0.000	0.000
w_{17}	1.000	1.000	1.000
w_{18}	1.000	1.000	1.000
w_{19}	0.999	1.000	1.000
w_{20}	1.000	1.000	1.000
w_{21}	0.542	0.574	1.000
w_{22}	0.222	0.210	0.000
w_{23}	0.000	0.000	0.000
w_{24}	1.000	1.000	1.000
w_{25}	1.000	1.000	1.000
w_{26}	0.000	0.000	0.000
w_{27}	1.000	1.000	1.000
w_{28}	1.000	1.000	1.000
w_{29}	0.000	0.000	0.000
w_{30}	1.000	1.000	1.000
$f(w^*)$		6.514 %	6.532 %
$P_{Basket}(w^*)$		6.509 %	6.532 %

Table 5.12: Optimal solution and objective value from Continuous NLP 2, for increasing μ)

Results from the continuous NLP 3 for several different starting vectors showed the same result, where the same results are obtained for every iteration. In Table 5.13 the results are shown for the unrounded solution from NLP 1 as the starting vector for increasing values of μ and decreasing values of λ. The results indicates us that is a good strategy can be to begin with $\mu^* = 0.001$ which yielded the integer values in NLP 2. Note that in the case of the small dimensional basket option, this opposite was recommended.

w^*	w^0	$\mu = 0.001, \lambda = 0.001$	$\mu = 0.01, \lambda = 10^{-4}$	$\mu = 0.01, \lambda = 10^{-6}$
w_1	1.000	0.709	0.991	1.000
w_2	0.000	0.108	0.006	0.000
w_3	0.000	0.308	0.008	0.000
w_4	0.000	0.173	0.007	0.000
w_5	0.237	0.394	0.011	0.000
w_6	0.000	0.284	0.009	0.000
w_7	0.000	0.200	0.007	0.000
w_8	1.000	0.804	0.992	1.000
w_9	1.000	0.725	0.991	1.000
w_{10}	1.000	0.844	0.993	1.000
w_{11}	0.000	0.156	0.008	0.000
w_{12}	1.000	0.807	0.992	1.000
w_{13}	0.000	0.209	0.008	0.000
w_{14}	0.000	0.177	0.008	0.000
w_{15}	0.000	0.179	0.007	0.000
w_{16}	0.000	0.277	0.010	0.000
w_{17}	1.000	0.600	0.991	1.000
w_{18}	1.000	0.907	0.995	1.000
w_{19}	0.999	0.701	0.990	1.000
w_{20}	1.000	0.542	0.989	1.000
w_{21}	0.542	0.572	0.989	1.000
w_{22}	0.222	0.518	0.011	0.000
w_{23}	0.000	0.298	0.010	0.000
w_{24}	1.000	0.915	0.995	1.000
w_{25}	1.000	0.862	0.993	1.000
w_{26}	0.000	0.217	0.009	0.000
w_{27}	1.000	0.689	0.991	1.000
w_{28}	1.000	0.827	0.992	1.000
w_{29}	0.000	0.141	0.007	0.000
w_{30}	1.000	0.855	0.993	1.000
$f(w^*)$		13.836 %	8.277 %	6.564 %
$P_{Basket}(w^*)$		7.896 %	6.585 %	6.532 %

Table 5.13: Optimal solution and objective value from Continuous NLP 3, for decreasing λ (and increasing μ)

For continuous NLP 3 the Hessian was numerical simulated for several combinations and positive eigenvalues were obtained. Hence, results shows that the Hessian is positive definite. In continuous NLP 2, the eigenvalues were negative - hence the results shows that Hessian is negative definite. Note that the same observations were made for the Hessians in NLP 2 and NLP 2 for the small dimensional basket option. The results and analysis for the Hessians can be found in Appendix B.

5.4 Varied W

To investigate further whether the continuous nonlinear problems are appropriate to use instead of the binary problem and also to have more result to evaluate all the optimization models for, W was changed to something smaller respectively bigger than its value given in 5.1. Especially in the case of an small dimensional problem, it is relevant to understand how the complexity of the combination affects the optimization. The results can be observed in Table 5.14:

W = 3					**W = 8**			
w^*	Binary NLP	Continuous NLP 1	Continuous NLP 2	Continuous NLP 3	Binary NLP	Continuous NLP 1	Continuous NLP 2	Continuous NLP 3
w_1	0.000	0.278	0.000	0.000	1.000	1.000	1.000	1.000
w_2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
w_3	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
w_4	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000
w_5	0.000	0.002	0.000	0.000	1.000	1.000	1.000	1.000
w_6	0.000	0.000	0.000	0.000	1.000	1.000	1.000	1.000
w_7	0.000	0.009	0.000	0.000	1.000	1.000	0.999	1.000
w_8	1.000	0.505	1.000	1.000	1.000	1.000	1.000	1.000
w_9	0.000	0.206	0.000	0.000	1.000	1.000	1.000	1.000
w_{10}	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$f(w^*)$	9.998 %	9.631 %	9.998 %	10.101 %	11.655 %	11.655 %	11.655 %	13.0178 %

Table 5.14: Sensitivity analysis for W, small dimensional problem

When searching for the price efficient combination w^* where $W = 8$ the optimization gave much better results than for $W = 5$, as well as it was "easier" to find the correct solution. By "easier" we mean how important the starting vector is in order to find correct solutions. Also, if one need strategies for (μ, λ) or if it is not crucial and the solutions is obtained straight forward. For $W = 3$ the same pattern was recognized and the continuous models gave better results comparing to $W = 5$ - with NLP 2 the correct solutions was easily obtained and NLP 3 also found the same solution.

One key observation is that the solution obtained in NLP 1 is an good starting vector here - if the solution is rounded it is actually equal the price efficient solution. As for the big dimensional problem in section 5.3 this was also the case, hence the same solution among the optimization models was obtained. So this tells us that the results from NLP 1 can be a factor determining if the further versions of the relaxations, NLP 2 and NLP 3, will be "easy" to use in order to find price efficient solutions.

The same analysis was done for the big dimensional model. The problem was tested for $W = 8$ and the result is given in Table 5.15. For values larger than $W = 19$ the problem optimization is not possible, since the optimization problem became infeasible in all cases - badly scaled problem could also be the reason according to the KNITRO algorithm manual. Several starting vectors were tested but it was not possible to solve the problem. The algorithm converged to an infeasible point from which it cannot further decrease the infeasibility measure.

Finally, the same observations were made for the Hessian in these cases, as has been analyzed in Appendix A and B and commented in previous section for both the large and small dimensional cases.

W = 8				
w^*	Binary NLP	Continuous NLP 1	Continuous NLP 2	Continuous NLP 3
w_1	0.000	0.000	0.000	0.000
w_2	0.000	0.000	0.000	0.000
w_3	0.000	0.000	0.000	0.000
w_4	0.000	0.000	0.000	0.000
w_5	0.000	0.000	0.000	0.000
w_6	0.000	0.000	0.000	0.000
w_7	0.000	0.000	0.000	0.000
w_8	0.000	0.022	0.000	0.000
w_9	0.000	0.000	0.000	0.000
w_{10}	1.000	0.978	1.000	1.000
w_{11}	0.000	0.000	0.000	0.000
w_{12}	1.000	1.000	1.000	1.000
w_{13}	0.000	0.000	0.000	0.000
w_{14}	0.000	0.000	0.000	0.000
w_{15}	0.000	0.000	0.000	0.000
w_{16}	0.000	0.000	0.000	0.000
w_{17}	0.000	0.000	0.000	0.000
w_{18}	1.000	1.000	1.000	1.000
w_{19}	0.000	0.000	0.000	0.000
w_{20}	1.000	1.000	1.000	1.000
w_{21}	0.000	0.000	0.000	0.000
w_{22}	0.000	0.000	0.000	0.000
w_{23}	0.000	0.000	0.000	0.000
w_{24}	1.000	1.000	1.000	1.000
w_{25}	1.000	1.000	1.000	1.000
w_{26}	0.000	0.000	0.000	0.000
w_{27}	0.000	0.000	0.000	0.000
w_{28}	1.000	1.000	1.000	1.000
w_{29}	0.000	0.000	0.000	0.000
w_{30}	1.000	1.000	1.000	1.000
$f(w^*)$	3.410 %	3.410 %	3.410 %	3.414 %

Table 5.15: Sensitivity analysis for W, big dimensional problem

5.5 Monte Carlo simulation of basket option price

Given the optimal combination of underlying assets we can simulate the basket option price. This is done simulating each stock price according to the Monte-Carlo simulation for multidimensional correlated stochastic processes. Then for each simulation of the set of stock prices the basket option price $P^{MC}_{Basket}(T)$ is calculated. This is done for the small and big dimensional basket option with the same input data as for the optimization.

Several Monte Carlo simulations were done for $P_{Basket}(T)$, which can be observed in Table 5.16. For the small dimensional basket option, the simulated price converges to values of $P^{MC}_{Basket} \approx 9.000\%$. Recall that the price calculated by moment matching formula $P^{MM}_{Basket} = 10.640\%$. Furthermore, for the larger problem the simulated price converges to a value of $P^{MC}_{Basket} = 23.353\%$ and $P_{Basket} \approx 5.600\%$ Hence, the moment matching method used in this thesis seem to be a good approximating pricing method for basket options.

P_{Basket}	**Monte Carlo (MC)**	**Moment Matching (MM)**
Small dimension	9.220 %	10.640 %
Big dimension	5.638 %	6.532 %

Table 5.16: Monte Carlo simulations of the basket option price, for n=10 and n=30

The basket option payoff $S_B(T) - K$ for each sample is illustrated in a histogram shown in Figure 5.1 and 5.2. Furthermore, the price function P_{Basket}, or the return of the basket option, over time for can be studied in Figure 5.5 and 5.6. The simulated price for each correlated stock can be observed in Figure 5.3 and 5.4.

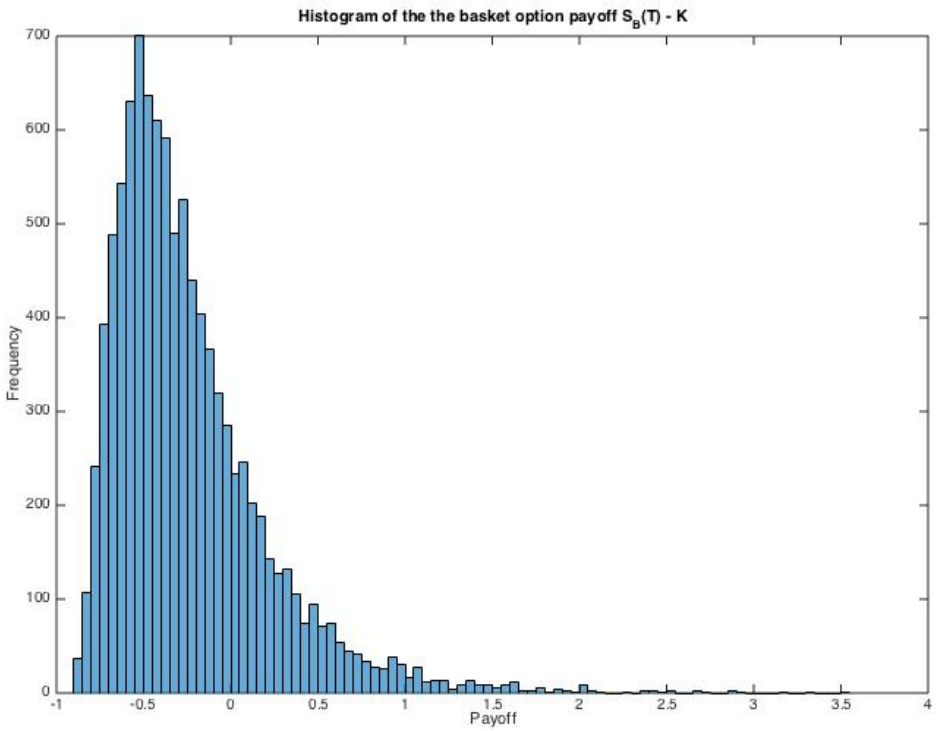


Figure 5.1: Simulation with 10000 samples and $P^{MC}_{Basket} = 9.135$ %, n=10

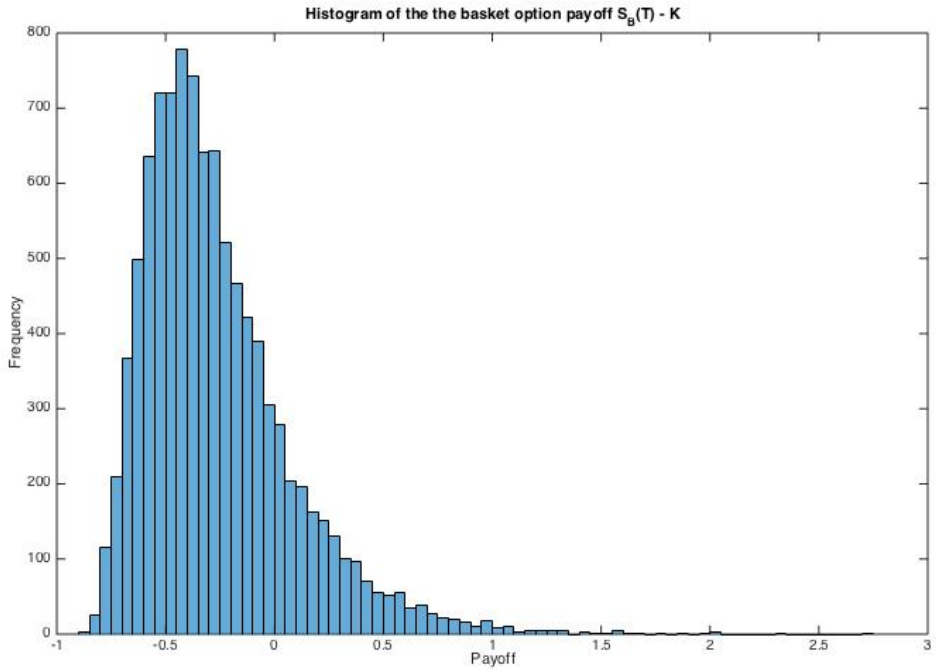


Figure 5.2: Simulation with 10000 samples and $P^{MC}_{Basket} = 5.435$ % , n=30

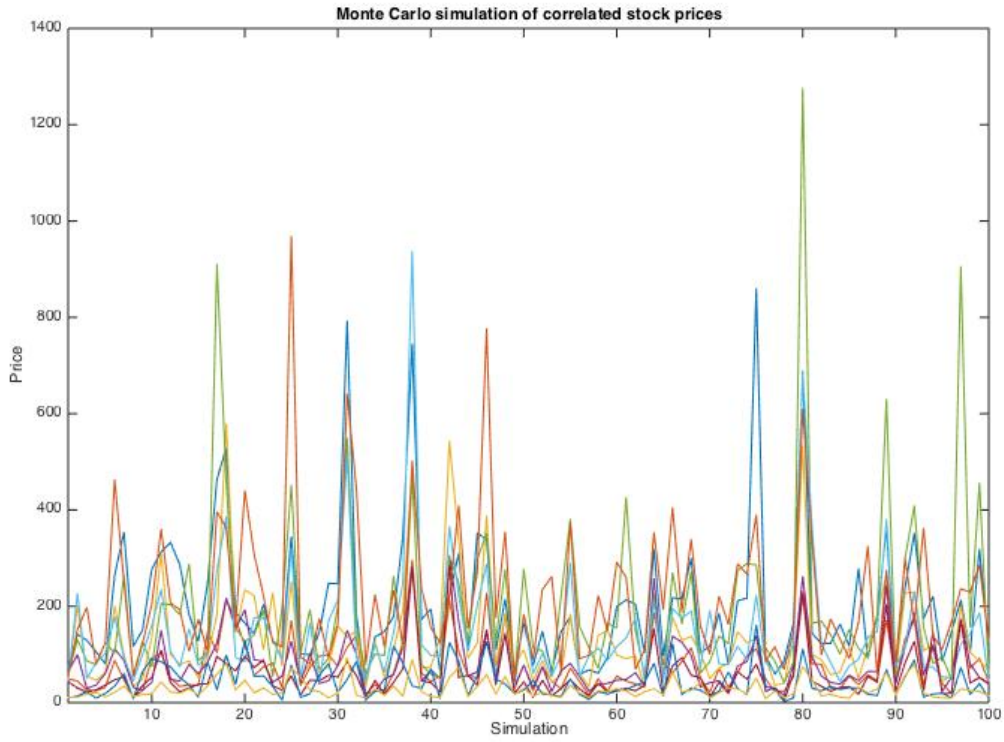


Figure 5.3: The simulated stock prices for the stocks given by the weights w_1 to w_{10}

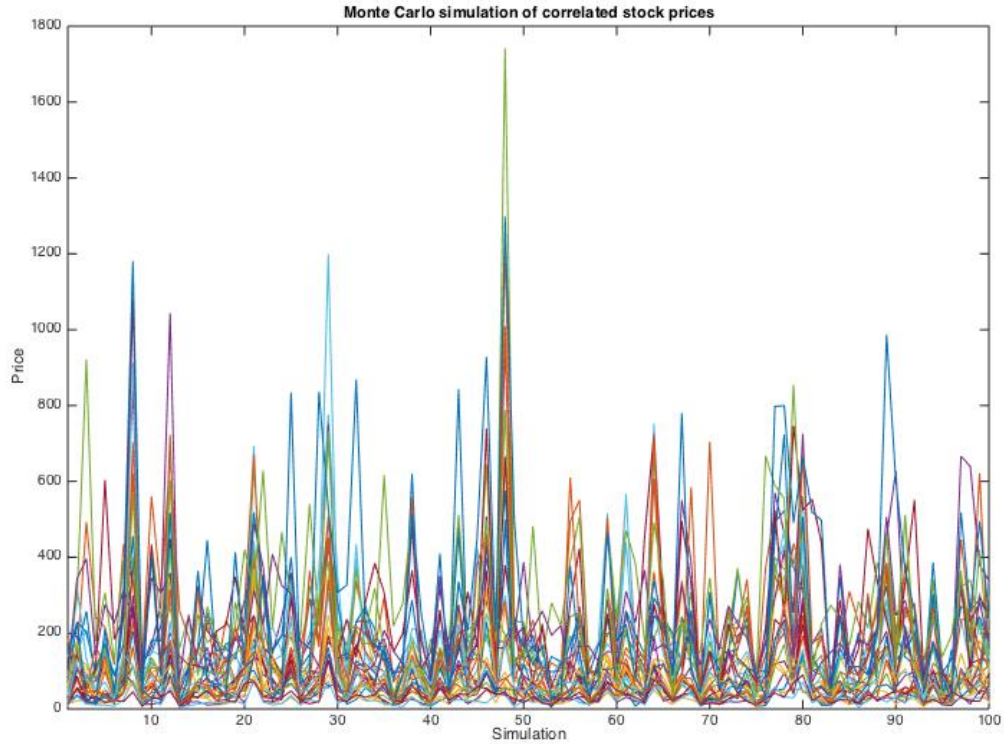


Figure 5.4: The simulated stock prices for the stocks given by the weights w_1 to w_{30}

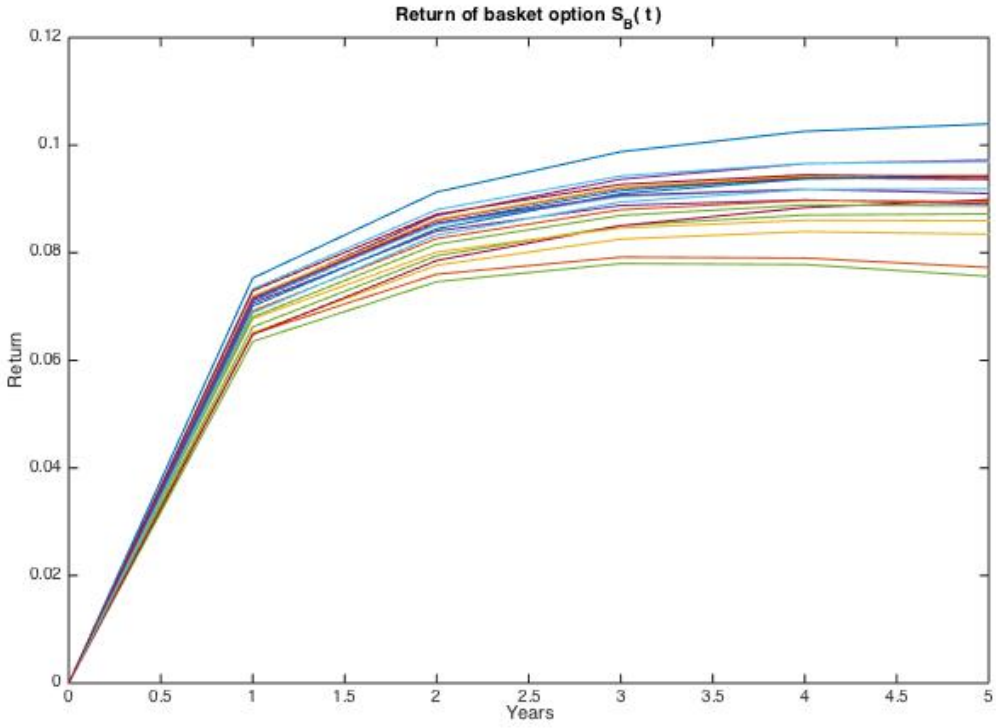


Figure 5.5: The return of small dimensional basket option from t=0 to t=T

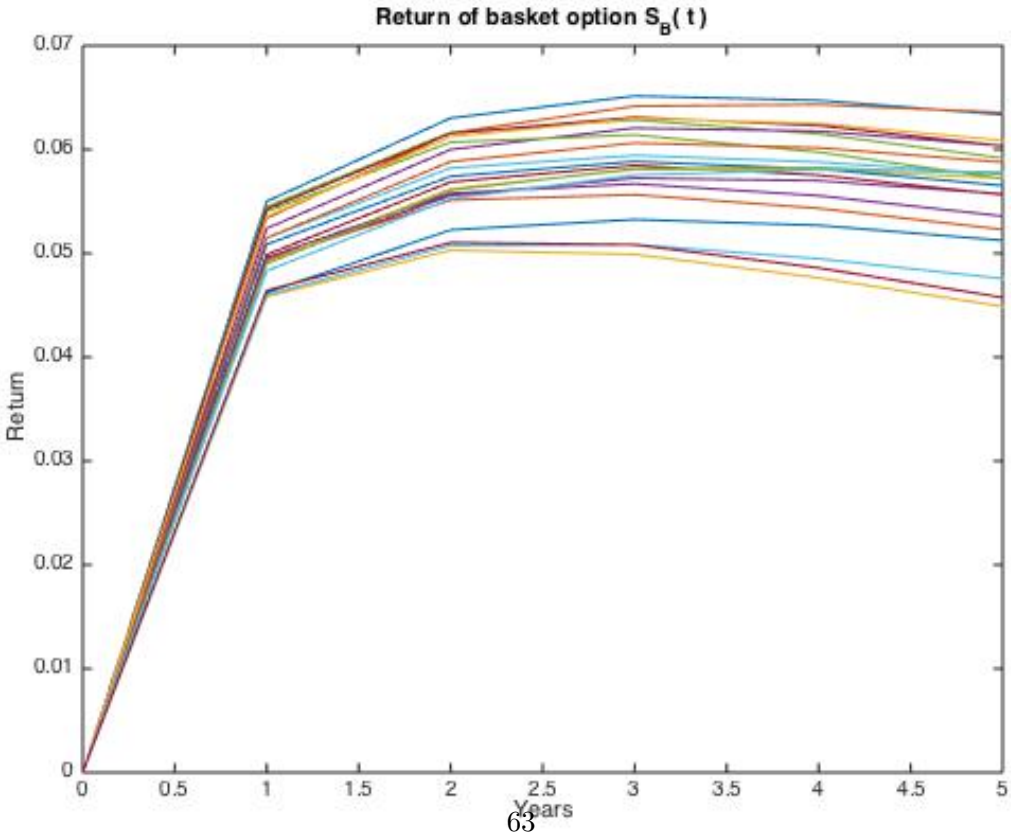


Figure 5.6: The return of large dimensional basket option from t=0 to t=T

Chapter 6

Evaluation and conclusion

6.1 Research question 1

Due to the lack of a distribution for the weighted average of lognormal variables, there is no analytical formula for pricing the basket option. Hence, choosing a pricing model will prompt that the needed input data will be model specific. This implies that there is need of a calibration of the pricing model, in order to use input data from other unknown sources.

For example, an important input data that is plugged into the pricing model is the volatility, which can be calculated from historic data or derived from a option's price - known as implied volatility. The implied volatility can for example be calculated from the Black & Scholes model, which is most likely the case in reality, and will obviously not have the same value if obtained from another price model. So using a volatility, calculated by one price model, into another pricing model will lead to that a different price is obtained. This would be a price that is not observed in the market, and for the writer of the option, this may imply that they will not receive the premium that is traded on the market. Hence, choosing a pricing model will prompt that the needed input data quoted from another pricing model should not be used if the price model is not calibrated before doing this.

Thus, a further research would be to conduct a calibration for the pricing model. This would make the model more usable by being able to utilize input data from other unknown sources. It is hard to know or it could be unknown how other actors' input parameters, such as the volatility, looks like and have been calculated. This is making the pricing of a basket option hard in order to be realistic for the trading market.

In addition, an advantage of the moment matching method was that the method provides a closed-form analytic formula that was very alike of the Black & Scholes model, which is observed by inspecting the chosen pricing formula. Thus, one may find it appropriate to use implied volatilities calculated from the Black & Scholes formula in this pricing model. In regards to the moment matching method made in this thesis, done for the two first moments with an lognormal variable, we can conclude that it is a good approximation method for pricing a basket option. Comparing the results with simulated prices from the Monte Carlo simulation shows this.

Another interesting topic for further research would be to improve the moment matching done in the thesis, by including the skewness and kurtosis. One issue here could be that the complexity of the price function increases, at the same as the computational effort also increases. Skewness and kurtosis are the the third respectively fourth moments, dervied in the same manner as in section 2.3:

$$\begin{aligned}\mathbf{E}[S_B^3(T)] = M_3 &= \sum_{i,j,k}^{N} w_i w_j w_k F_i F_j F_k e^{\rho_{ij}\sigma_i\sigma_j T+\rho_{ik}\sigma_i\sigma_k T+\rho_{jk}\sigma_j\sigma_k T} \\ \mathbf{E}[S_B^4(T)] = M_4 &= \sum_{i,j,k,l}^{N} w_i w_j w_k w_l F_i F_j F_k F_l e^{\rho_{ij}\sigma_i\sigma_j T+\rho_{ik}\sigma_i\sigma_k T+\rho_{jk}\sigma_j\sigma_k T+\rho_{jl}\sigma_j\sigma_l T+\rho_{kl}\sigma_k\sigma_l T}\end{aligned} \tag{6.1}$$

It would be interesting to see if the approximation would get better by including higher moments, especially since the results shows that the approximation becomes less good in bigger dimensions of the basket option. Furthermore other alternatives than the lognormal distribution can be used for the moment matching method, which has been mentioned in Chapter 3. Thus another further investigation would be to compare which distribution that, used for approximation of the weighted average, would perform better.

6.2 Research question 2

One conclusion is that the both the binary optimization model and the three different continuous NLP are suitable in order to find the optimal combination of underlying assets for a basket option. Furthermore, one of the reasons of considering the continuous nonlinear optimization problems is due to the fact that these problems are in general easier to solve. Hence it can be concluded that there are two suitable approaches for modeling this thesis optimization problem - as a binary and an continuous problem. Furthermore, the result also implies that both the underlying algorithms, branch-and-bound and interior point method with direct step, are both suitable when solving the optimizations problem of finding the price efficient combination of stocks.

In addition, the continuous problems could be an indicator of how accurate the solution of the binary nonlinear problem is and vice versa - since it is shown by the results that the models converge to the same optimal solution. Thus, the continuous models could be used for comparison of obtained result from the binary optimization model as well - since these types of problems are hard to solve, especially for increasing number of binary variables. It is also hard for the solvers to these kinds of problems to distinguish between local and global optimum [8], highlighting once again the importance of comparing and analyzing results obtained from both binary and nonlinear optimization problems.

In the small dimension, we had the possibility to compare the results from the optimization with the price efficient combination obtained from an algorithm that tests all possible combinations. Hence, for the small problem the obtained solutions from the optimization models are price efficient and also global optimal solutions. Furthermore, we saw that the sampled Hessian for continuous NLP 3 were positive definite - as the author in [22] claimed that the logarithmic smoothing would do for the transformed objective function. Since we have linear inequality and equality constraints, the

feasible region is convex as well. Based on this result, the solution obtained in continuous NLP 3 is shown to be a global optimal solution, once again, since the objective function and feasible region is convex.

The results showed us that the same optimal solutions were obtained by the different optimization problem in the large dimensional case. As for the small dimensional case, the numerical estimations of the Hessian for NLP 3 were shown to be positive definite. However, an comparison with the algorithm that search through all possible combinations was not possible due to the high number of combinations. One may conclude that the solutions may be price efficient as well. The sampled Hessian for the continuous NLP 3 were positive definite as well. With the same reasoning as for the small dimensional case, the optimal solution for continuous NLP 3 is shown to be global optimal to the problem, based on the obtained results for the Hessian.

Appendices

Appendix A

Small dimensional problem

A.1 Comments about the Hessians

For the binary optimization problem, all the possible and feasible combinations w were used in the iterations for sampling the Hessian $H(w)$ and investigate if it is positive definite. The combinations are the same that is generated by the algorithm that finds the optimal combination, based on the minimum basket option price P_{Basket}. For all the samples the Hessian was not positive definite, since the eigenvalues Λ were both negative and positive - more specifically the first eigenvalue was negative and the rest were positive. As an numerical example, the eigenvalues for w was

w	Λ
1.000	-0.01064
0.000	0.002053
1.000	0.002412
0.000	0.002732
0.000	0.003408
0.000	0.004356
1.000	0.006104
1.000	0.006804
0.000	0.009861
1.000	0.01201

Table A.1: An example of the eigenvalues for sampled Hessian for the bianary NLP

For continuous NLP 1, several (feasible) combinations were tested and none of the samples of the Hessian were positive definite. In addition, sampling was also done with elements in w that was non-integers and the same patterns were obtained - the magnitude of the eigenvalues were not remarkably different either. The same pattern was recognized as for the binary problem, the majority of the eigenvalues were positive. An example illustrating this can be observed in A.2 for the optimal solution obtained in continuous NLP 1.

w	Λ
0.708	-0.01182
0.000	0.002161
1.000	0.002452
0.022	0.002815
0.178	0.003584
0.000	0.004601
0.506	0.006114
1.000	0.008049
0.586	0.009974
1.000	0.01149

Table A.2: An example of the eigenvalues for sampled Hessian for the continuous NLP 1

In continuous NLP 2 the Hessian was samples for several combinations, but also for every increasing μ. Sampling was also done with elements in w that was non-integers and the results showed no difference from the case with integer values - the same observation was made. For a small μ the same pattern for the eigenvalues were obtained as for binary NLP and continuous NLP 1. But as μ increased, the number of negative eigenvalues also increased - for $\mu > 0.001$ all the eigenvalues were negative. An example of this can be observed in Table A.3, when sampling the Hessian for two values of μ for the given w - which the optimal solution vector.

w	Λ when $\mu = 0.0001$	Λ when $\mu = 0.01$
1.000	-0.01196	-0.03064
0.000	0.001961	-0.01795
1.000	0.002257	-0.01759
0.000	0.002615	-0.01727
0.000	0.003381	-0.01659
0.000	0.004405	-0.01564
1.000	0.005905	-0.01390
1.000	0.007865	-0.01320
0.000	0.009786	-0.01014
1.000	0.01131	-0.007994

Table A.3: An example of the eigenvalues for sampled Hessian for the continuous NLP 2

When sampling the Hessian for various combinations for the continuous NLP 3, the eigenvalues were positive and larger in magnitude comparing to the other optimization models. Decreasing or increasing λ did not affect the Hessian positive definiteness. The only different observation was that in some cases when sampling the Hessian the first eigenvalue became approximately 0. In Table A.4 the eigenvalues Λ for different values of (λ, μ) are shown. The input vector for calculating $H(w)$ were the optimal solution obtained for that given value of (λ, μ). It is interesting to see what happens to the Hessian at each iteration since the optimal solution for one problem is the starting vector for the next iteration of a decreasing λ. The vector w given in Table A.4 is thus the integer

solution obtained in the end of the optimization process, actually when $\mu = 0.01, \lambda = 10^{-6}$.

w	Λ for $\mu = 0.01, \lambda = 10^{-6}$	Λ for $\mu = 0.01, \lambda = 1$	Λ for $\mu = 1, \lambda = 1$	Λ for $\mu = 10^{-6}, \lambda = 10^{-6}$
1.000	45.8035	7.9679	5.9880	-0.002399
0.000	51.9616	7.9822	6.002214	0.003211
1.000	90.07532	7.9823	6.002398	0.003656
0.000	95.5524	7.9827	6.002801	0.006321
0.000	110.8444	7.9835	6.003529	0.008507
0.000	128.2440	7.9843	6.004499	0.028350
0.000	183.7840	7.9860	6.006011	17.3423
1.000	229.3387	7.9881	6.008154	37.4185
1.000	378.4986	7.9891	6.009266	92.5438
1.000	507.8464	7.9984	6.01853	125.1759

Table A.4: An example of the eigenvalues for sampled Hessian for the continuous NLP 3

Appendix B

Big dimensional model

B.1 Comments about the Hessians

For all the optimization models, the same tests were done to see what characteristics the Hessian has. Since it is a large number of different combinations it was of course not impossible to test this for all possible combinations w. The same observations was made for the binary NLP and continuous NLP 1 as for the small dimensional case - all eigenvalues except the first one were positive, whereas the first eigenvalue did not have large magnitude. The results for this can be viewed in the two first tables (from left to right) presented in Table B.1.

For several feasible combinations, the Hessian for NLP 2 were not positive definite which can be seen by the values for the eigenvalues in the third table (from left to right) in Table B.1. As we can see, for bigger values of μ there are a mix of positive and negative eigenvalues- but it is a majority of negative values. Obviously, when μ is very small the Hessian has the same characteristics as for NLP 1.

When sampling the Hessian for NLP 3, it was shown to be positive definite, an example of the sampling can be seen in Table B.2. In addition, when sampling for feasible w the Hessian remained positive definite.

w	Λ
1.000	-0.008215
0.000	0.000039
0.000	0.000045
0.000	0.000156
0.000	0.000216
0.000	0.000247
0.000	0.000273
1.000	0.000305
1.000	0.000319
1.000	0.000329
0.000	0.000351
1.000	0.000380
0.000	0.000397
0.000	0.000457
0.000	0.000484
0.000	0.000532
1.000	0.000622
1.000	0.000657
1.000	0.000697
1.000	0.000778
1.000	0.000851
0.000	0.001016
0.000	0.001043
1.000	0.001261
1.000	0.001622
0.000	0.001792
1.000	0.001939
1.000	0.002662
0.000	0.004103
1.000	0.006160

w	Λ
1.000	-0.008548
0.000	0.000039
0.000	0.000045
0.000	0.000156
0.236	0.000216
0.000	0.000247
0.000	0.000273
1.000	0.000305
1.000	0.000319
1.000	0.000329
0.000	0.000350
1.000	0.000380
0.000	0.000397
0.000	0.000457
0.000	0.000484
0.000	0.000532
1.000	0.000621
1.000	0.000657
1.000	0.000698
1.000	0.000778
0.542	0.000851
0.222	0.001014
0.000	0.001042
1.000	0.001261
1.000	0.001620
0.000	0.001758
1.000	0.001958
1.000	0.002605
0.000	0.004042
1.000	0.006150

w	Λ for $\mu = 0.001$	Λ for $\mu = 0.000001$
1.000	-0.010215	-0.008550
0.000	-0.001961	0.000037
0.000	-0.001955	0.000043
0.000	-0.001844	0.000154
0.000	-0.001784	0.000214
0.000	-0.001753	0.000245
0.000	-0.001727	0.000271
1.000	-0.001695	0.000303
1.000	-0.001681	0.000317
1.000	-0.001671	0.000327
0.000	-0.001649	0.000348
1.000	-0.001620	0.000378
0.000	-0.001603	0.000395
0.000	-0.001543	0.000455
0.000	-0.001516	0.000482
0.000	-0.001468	0.000530
1.000	-0.001378	0.000619
1.000	-0.001343	0.000655
1.000	-0.001303	0.000696
1.000	-0.001222	0.000776
1.000	-0.001149	0.000849
0.000	-0.000984	0.001012
0.000	-0.000957	0.001040
1.000	-0.000739	0.001259
1.000	-0.000378	0.001618
0.000	-0.000208	0.001757
1.000	-0.000061	0.001956
1.000	0.000662	0.002603
0.000	0.002103	0.004040
1.000	0.004160	0.006148

Table B.1: Examples of sampled eigenvalues for the sampled Hessian for the binary NLP, continuous NLP 1 respectively continuous NLP 2.

w	Λ for $\mu = 0.001\lambda = 10^{-5}$	Λ for $\mu = 0.001\lambda = 100$	Λ for $\mu = 10^{-7}\lambda = 10^{-5}$
1.000	9.9901	100000100.1904	9.9921
0.000	9.9980	100000100.1983	10.0000
0.000	9.9981	100000100.1984	10.0001
0.000	9.9982	100000100.1985	10.0002
0.000	9.9982	100000100.1985	10.0002
0.000	9.9982	100000100.1985	10.0002
0.000	9.9983	100000100.1986	10.0003
1.000	9.9983	100000100.1986	10.0003
1.000	9.9983	100000100.1986	10.0003
1.000	9.9983	100000100.1986	10.0003
0.000	9.9983	100000100.1986	10.0003
1.000	9.9984	100000100.1987	10.0004
0.000	9.9984	100000100.1987	10.0004
0.000	9.9984	100000100.1987	10.0004
0.000	9.9985	100000100.1988	10.0005
0.000	9.9986	100000100.1989	10.0006
1.000	9.9986	100000100.1989	10.0006
1.000	9.9986	100000100.1989	10.0006
1.000	9.9987	100000100.1990	10.0007
1.000	9.9988	100000100.1990	10.0008
1.000	9.9989	100000100.1992	10.0009
0.000	9.9989	100000100.1992	10.0009
0.000	9.9991	100000100.1994	10.0011
1.000	9.9995	100000100.1997	10.0015
1.000	9.9995	100000100.1998	10.0015
0.000	9.9997	100000100.2000	10.0017
1.000	10.0002	100000100.2005	10.0022
1.000	10.0014	100000100.2017	10.0034
0.000	10.0060	100000100.2063	10.0080
1.000	999.9998	10000000100.0198	1000.0018

Table B.2: An example of the eigenvalues for sampled Hessian for the continuous NLP 3

Appendix C

Input data

C.1 Small dimensional model

265.9000	229.6700	0.2735
85.1500	82.7300	0.2992
197.9000	152.2700	0.2803
114.3000	95.7700	0.3056
247.1000	201.0500	0.3292
175.5000	164.7000	0.2850
84.2000	65.4200	0.3329
60.5000	46.7100	0.3478
269.3000	245.6500	0.2666
38.8400	29.7600	0.2551

Table C.1: Spotprice S_0, Forward price F and Volatility σ for $w_1 - w_{10}$

1.0000	0.6358	0.5772	0.6084	0.6922	0.6608	0.6102	0.5078	0.6132	0.6695
0.6358	1.0000	0.7593	0.7932	0.5958	0.6020	0.7997	0.5209	0.5021	0.5880
0.5772	0.7593	1.0000	0.7570	0.6089	0.6225	0.7822	0.5269	0.5081	0.6494
0.6084	0.7932	0.7570	1.0000	0.6597	0.6379	0.8284	0.4597	0.5036	0.6235
0.6922	0.5958	0.6089	0.6597	1.0000	0.6756	0.6573	0.4973	0.4942	0.6626
0.6608	0.6020	0.6225	0.6379	0.6756	1.0000	0.5759	0.4657	0.5875	0.6394
0.6102	0.7997	0.7822	0.8284	0.6573	0.5759	1.0000	0.5204	0.4798	0.6420
0.5078	0.5209	0.5269	0.4597	0.4973	0.4657	0.5204	1.0000	0.4897	0.4968
0.6132	0.5021	0.5081	0.5036	0.4942	0.5875	0.4798	0.4897	1.0000	0.5777
0.6695	0.5880	0.6494	0.6235	0.6626	0.6394	0.6420	0.4968	0.5777	1.0000

Table C.2: Correlation matrix ρ for the ten stocks given by $w_1 - w_{10}$

C.2 Large dimensional model

265.9000	229.6700	0.2670
85.1500	82.7300	0.2975
197.9000	152.2700	0.2778
114.3000	95.7700	0.3002
247.1000	201.0500	0.3270
175.5000	164.7000	0.2794
84.2000	65.4200	0.3203
60.5000	46.7100	0.3398
269.3000	245.6500	0.2645
38.8400	29.7600	0.2591
266.9000	255.2600	0.3215
86.1500	49.3600	0.2545
198.9000	196.8300	0.2976
115.3000	105.8700	0.3327
248.1000	210.9100	0.2493
176.5000	154.1600	0.3119
85.2000	82.2800	0.2121
61.5000	33.9000	0.2939
270.3000	262.8200	0.2620
39.8400	28.4000	0.2772
267.9000	243.4500	0.3409
87.1500	82.5500	0.3171
199.9000	179.2700	0.3429
116.3000	79.4300	0.2682
249.1000	219.2100	0.2236
177.5000	161.5100	0.3274
86.2000	73.1800	0.2658
62.5000	37.6500	0.2610
271.3000	258.7300	0.4478
130.5000	64.3500	0.3004

Table C.3: Spotprice S_0, Forward price F and Volatility σ for $w_1 - w_{30}$

Table C.4: Correlation matrix ρ for the 30 stocks given by $w_1 - w_{30}$

1.0000	0.6358	0.5772	0.6084	0.6922	0.6608	0.6102	0.5078	0.6132	0.6695	0.6904	0.7783	0.6726	0.6232	0.7620	0.6951	0.6145	0.5837	0.5428	0.6531	0.4214	0.5417	0.5239	0.2527	0.4248	0.6228	0.5720	0.5966	0.4539	0.6876
0.6358	1.0000	0.7593	0.7932	0.5958	0.6020	0.7997	0.5209	0.5021	0.5880	0.6482	0.7675	0.6119	0.5953	0.7507	0.5995	0.6420	0.5673	0.5181	0.5544	0.4724	0.5092	0.5248	0.2367	0.4239	0.5797	0.5869	0.5756	0.5225	0.6452
0.5772	0.7593	1.0000	0.7570	0.6089	0.6225	0.7822	0.5269	0.5081	0.6494	0.6726	0.7683	0.5866	0.6406	0.7488	0.5996	0.6740	0.5861	0.4715	0.5986	0.4899	0.4815	0.5591	0.3407	0.4260	0.5635	0.6270	0.5749	0.5200	0.6480
0.6084	0.7932	0.7570	1.0000	0.6597	0.6379	0.8284	0.4597	0.5036	0.6235	0.6576	0.7593	0.5862	0.6628	0.7492	0.6549	0.6519	0.5951	0.4433	0.5413	0.4985	0.4816	0.5783	0.2308	0.3876	0.6375	0.6108	0.5781	0.5358	0.6469
0.6922	0.5958	0.6089	0.6597	1.0000	0.6756	0.6573	0.4973	0.4942	0.6626	0.8239	0.7525	0.6899	0.8552	0.7361	0.9807	0.7154	0.6081	0.3959	0.6493	0.5008	0.5336	0.7695	0.3332	0.3578	0.7711	0.6464	0.5827	0.6391	0.7161
0.6608	0.6020	0.6225	0.6379	0.6756	1.0000	0.5759	0.4657	0.5875	0.6394	0.6205	0.7597	0.6730	0.6471	0.7481	0.6824	0.6606	0.6257	0.5460	0.6634	0.4649	0.4765	0.5489	0.3353	0.5159	0.6376	0.6701	0.6098	0.5226	0.6858
0.6102	0.7997	0.7822	0.8284	0.6573	0.5759	1.0000	0.5204	0.4798	0.6420	0.7125	0.7867	0.6046	0.6954	0.7672	0.6526	0.6708	0.6132	0.3987	0.5631	0.5070	0.4917	0.5956	0.2455	0.3627	0.6286	0.6120	0.5546	0.6072	0.6691
0.5078	0.5209	0.5269	0.4597	0.4973	0.4657	0.5204	1.0000	0.4897	0.4968	0.5212	0.5896	0.4873	0.5282	0.5822	0.4998	0.4908	0.4702	0.4000	0.4742	0.3504	0.3670	0.4083	0.3356	0.3939	0.5250	0.5332	0.5059	0.4277	0.4835
0.6132	0.5021	0.5081	0.5036	0.4942	0.5875	0.4798	0.4897	1.0000	0.5777	0.4587	0.6162	0.5544	0.4545	0.6160	0.5033	0.4998	0.5185	0.5703	0.5144	0.3749	0.3705	0.4091	0.3762	0.4666	0.5096	0.5370	0.4570	0.3257	0.5023
0.6695	0.5880	0.6494	0.6235	0.6626	0.6394	0.6420	0.4968	0.5777	1.0000	0.6775	0.7162	0.6446	0.6471	0.6851	0.6597	0.6402	0.6124	0.4944	0.5922	0.5087	0.3937	0.5326	0.3441	0.4745	0.6272	0.6322	0.5273	0.4538	0.6657
0.6904	0.6482	0.6726	0.6576	0.8239	0.6205	0.7125	0.5212	0.4587	0.6775	1.0000	0.7802	0.6928	0.8108	0.7522	0.8148	0.7213	0.6885	0.3556	0.6521	0.5347	0.5634	0.7331	0.2883	0.3779	0.7096	0.6273	0.5662	0.6254	0.7259
0.7783	0.7675	0.7683	0.7593	0.7525	0.7597	0.7867	0.5896	0.6162	0.7162	0.7802	1.0000	0.7385	0.7374	0.9781	0.7628	0.7355	0.6882	0.5778	0.6855	0.5250	0.5745	0.6062	0.3578	0.5084	0.7167	0.6839	0.6729	0.5834	0.8377
0.6726	0.6119	0.5866	0.5862	0.6899	0.6730	0.6046	0.4873	0.5544	0.6446	0.6928	0.7385	1.0000	0.6865	0.7287	0.6886	0.6531	0.6183	0.4348	0.7398	0.4405	0.5019	0.5972	0.3375	0.3768	0.6539	0.6405	0.5928	0.5251	0.6680
0.6232	0.5953	0.6406	0.6628	0.8552	0.6471	0.6954	0.5282	0.4545	0.6471	0.8108	0.7374	0.6865	1.0000	0.7084	0.8344	0.7781	0.6040	0.3827	0.6344	0.4843	0.4984	0.7736	0.3202	0.4018	0.7711	0.6639	0.5736	0.6564	0.7000
0.7620	0.7507	0.7488	0.7492	0.7361	0.7481	0.7672	0.5822	0.6160	0.6851	0.7522	0.9781	0.7287	0.7084	1.0000	0.7451	0.7010	0.6712	0.5728	0.6628	0.5012	0.5399	0.5817	0.3508	0.4903	0.6997	0.6794	0.6517	0.5689	0.8166
0.6951	0.5995	0.5996	0.6549	0.9807	0.6824	0.6526	0.4998	0.5033	0.6597	0.8148	0.7628	0.6886	0.8344	0.7451	1.0000	0.6907	0.6083	0.4025	0.6417	0.5050	0.5263	0.7516	0.3608	0.3700	0.7692	0.6471	0.5724	0.6332	0.7200
0.6145	0.6420	0.6740	0.6519	0.7154	0.6606	0.6708	0.4908	0.4998	0.6402	0.7213	0.7355	0.6531	0.7781	0.7010	0.6907	1.0000	0.5791	0.4643	0.5753	0.5090	0.5201	0.6203	0.3240	0.4108	0.7044	0.6469	0.5560	0.6100	0.6638
0.5837	0.5673	0.5861	0.5951	0.6081	0.6257	0.6132	0.4702	0.5185	0.6124	0.6885	0.6882	0.6183	0.6040	0.6712	0.6083	0.5791	1.0000	0.4149	0.6387	0.4714	0.5024	0.5647	0.2950	0.4282	0.5993	0.6105	0.5500	0.4854	0.6151
0.5428	0.5181	0.4715	0.4433	0.3959	0.5460	0.3987	0.4000	0.5703	0.4944	0.3556	0.5778	0.4348	0.3827	0.5728	0.4025	0.4643	0.4149	1.0000	0.4168	0.3090	0.3058	0.2842	0.2222	0.4769	0.3969	0.5271	0.4463	0.2902	0.4782
0.6531	0.5544	0.5986	0.5413	0.6493	0.6634	0.5631	0.4742	0.5144	0.5922	0.6521	0.6855	0.7398	0.6344	0.6628	0.6417	0.5753	0.6387	0.4168	1.0000	0.4287	0.5322	0.5595	0.3045	0.3725	0.6000	0.6179	0.6428	0.4553	0.6545
0.4214	0.4724	0.4899	0.4985	0.5008	0.4649	0.5070	0.3504	0.3749	0.5087	0.5347	0.5250	0.4405	0.4843	0.5012	0.5050	0.5090	0.4714	0.3090	0.4287	1.0000	0.4245	0.4449	0.2830	0.2881	0.4820	0.4213	0.3909	0.4752	0.5006
0.5417	0.5092	0.4815	0.4816	0.5336	0.4765	0.4917	0.3670	0.3705	0.3937	0.5634	0.5745	0.5019	0.4984	0.5399	0.5263	0.5201	0.5024	0.3058	0.5322	0.4245	1.0000	0.4561	0.1542	0.2022	0.5339	0.4587	0.5629	0.4023	0.5471
0.5239	0.5248	0.5591	0.5783	0.7695	0.5489	0.5956	0.4083	0.4091	0.5326	0.7331	0.6062	0.5972	0.7736	0.5817	0.7516	0.6203	0.5647	0.2842	0.5595	0.4449	0.4561	1.0000	0.2853	0.2631	0.6340	0.5600	0.5029	0.5689	0.5576
0.2527	0.2367	0.3407	0.2308	0.3332	0.3353	0.2455	0.3356	0.3762	0.3441	0.2883	0.3578	0.3375	0.3202	0.3508	0.3608	0.3240	0.2950	0.2222	0.3045	0.2830	0.1542	0.2853	1.0000	0.3561	0.3067	0.3934	0.2432	0.2228	0.3195
0.4248	0.4239	0.4260	0.3876	0.3578	0.5159	0.3627	0.3939	0.4666	0.4745	0.3779	0.5084	0.3768	0.4018	0.4903	0.3700	0.4108	0.4282	0.4769	0.3725	0.2881	0.2022	0.2631	0.3561	1.0000	0.3647	0.5452	0.3577	0.1878	0.4268
0.6228	0.5797	0.5635	0.6375	0.7711	0.6376	0.6286	0.5250	0.5096	0.6272	0.7096	0.7167	0.6539	0.7711	0.6997	0.7692	0.7044	0.5993	0.3969	0.6000	0.4820	0.5339	0.6340	0.3067	0.3647	1.0000	0.6207	0.5573	0.5980	0.6824
0.5720	0.5869	0.6270	0.6108	0.6464	0.6701	0.6120	0.5332	0.5370	0.6322	0.6273	0.6839	0.6405	0.6639	0.6794	0.6471	0.6469	0.6105	0.5271	0.6179	0.4213	0.4587	0.5600	0.3934	0.5452	0.6207	1.0000	0.5906	0.5129	0.6201
0.5966	0.5756	0.5749	0.5781	0.5827	0.6098	0.5546	0.5059	0.4570	0.5273	0.5662	0.6729	0.5928	0.5736	0.6517	0.5724	0.5560	0.5500	0.4463	0.6428	0.3909	0.5629	0.5029	0.2432	0.3577	0.5573	0.5906	1.0000	0.4312	0.6351
0.4539	0.5225	0.5200	0.5358	0.6391	0.5226	0.6072	0.4277	0.3257	0.4538	0.6254	0.5834	0.5251	0.6564	0.5689	0.6332	0.6100	0.4854	0.2902	0.4553	0.4752	0.4023	0.5689	0.2228	0.1878	0.5980	0.5129	0.4312	1.0000	0.5899
0.6876	0.6452	0.6480	0.6469	0.7161	0.6858	0.6691	0.4835	0.5023	0.6657	0.7259	0.8377	0.6680	0.7000	0.8166	0.7200	0.6638	0.6151	0.4782	0.6545	0.5006	0.5471	0.5576	0.3195	0.4268	0.6824	0.6201	0.6351	0.5899	1.0000

Bibliography

[1] Tomas Björk. *Arbitrage theory in continuous time*. Oxford university press, 2009.

[2] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *The journal of political economy*, pages 637–654, 1973.

[3] Svetlana Borovkova. Basket options and implied correlations: a closed form approach. In *CFC conference, London*, 2007.

[4] Thomas P Branson and Yang Ho Choi. Option pricing on multiple assets. *Acta Applicandae Mathematica*, 94(2):137–162, 2006.

[5] Mercurio F. Rapisarda F. Brigo, D. and R. Scotti. Approximated moment-matching dynamics for basket-options simulation. *Quantitative Finance*, (4):1–16.

[6] Ruggero Caldana, Gianluca Fusai, Alessandro Gnoatto, and Martino Grasselli. General closed-form basket option pricing bounds. *Quantitative Finance*, 16(4):535–554, 2016.

[7] René Carmona and Valdo Durrleman. Generalizing the black-scholes formula to multivariate contingent claims. *Journal of computational finance*, 9(2):43, 2005.

[8] John W Chinneck. Practical optimization: a gentle introduction. *Systems and Computer Engineering), Carleton University, Ottawa. http://www. sce. carleton. ca/faculty/chinneck/po. html*, 2006.

[9] G Deelstra, Jan Liinev, and Michèle Vanmaele. Pricing of arithmetic basket and asian basket options by conditioning. In *20th International Meeting of French Finance Association (AFFI) and 7th International Congress of Insurance: Mathematics & Economics (IME)*, 2003.

[10] Georges Dionne and HEC Montréal. *Heterogeneous basket options pricing using analytical approximations*. Montréal: HEC Montréal, Centre de recherche en e-finance, 2006.

[11] Steven R Dunbar. Stochastic processes and advanced mathematical finance.

[12] Isabel Ehrlich. Pricing basket options with smile. 2012.

[13] Erik Ekedahl, Eric Hansander, and Erik Lehto. Dimension reduction for the black-scholes equation. *Department of Information Technology, Uppsala University*, 2007.

[14] Igor Griva, Stephen G Nash, and Ariela Sofer. *Linear and nonlinear optimization*. Siam, 2009.

[15] Martin Haugh. The monte carlo framework, examples from finance and generating correlated random variables. *Course Notes*, 2004.

[16] John C Hull. *Options, futures, and other derivatives*. Pearson Education India, 2006.

[17] Nengjiu Ju. Pricing asian and basket options via taylor expansion. *Journal of Computational Finance*, 5(3):79–103, 2002.

[18] Timo Koski. Lecture notes: Probability and random processes at KTH.

[19] Edmond Levy. Pricing european average rate currency options. *Journal of International Money and Finance*, 11(5):474–491, 1992.

[20] Daniël Linders and Wim Schoutens. Basket option pricing and implied correlation in a one-factor lévy model. In *Innovations in Derivatives Markets*, pages 335–367. Springer, 2016.

[21] Walter Mudzimbabwe. *Pricing methods for Asian options*. PhD thesis, PHD thesis, University of the Western Cape, 2009.

[22] Walter Murray and Kien-Ming Ng. An algorithm for nonlinear optimization problems with binary variables. *Computational Optimization and Applications*, 47(2):257–288, 2010.

[23] George L Nemhauser and Laurence A Wolsey. Chapter vi integer programming. *Handbooks in Operations Research and Management Science*, 1:447–527, 1989.

[24] Tommaso Paletta, Arturo Leccadito, and Radu Tunaru. Pricing and hedging basket options with exact moment matching. *Available at SSRN 2368316*, 2014.

[25] Pablo A. Parrilo. Algebraic techniques and semidefinite optimization, lecture 3. 2006.

[26] Olivier Pironneau and Yves Achdou. Partial differential equations for option pricing. *Handbook of Numerical Analysis*, 15:369–495, 2009.

[27] Richard A Waltz, José Luis Morales, Jorge Nocedal, and Dominique Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical programming*, 107(3):391–408, 2006.

[28] Paul Wilmott. *The Best of Wilmott 2, Chapter 16*. John Wiley & Sons, 2006.

[29] Jinke Zhou and Xiaolu Wang. Accurate closed-form approximation for pricing asian and basket options. *Applied stochastic models in business and industry*, 24(4):343–358, 2008.

TRITA -MAT-E 2017:10
ISRN -KTH/MAT/E--17/10--SE