Honours Project
Pricing of Himalaya Options

Obeid Mahomed
omahomed@cam.wits.ac.za
0417991J

Programme in Advanced Mathematics of Finance
School of Computational and Applied Mathematics
University of the Witwatersrand, Johannesburg

Abstract. This project examines the pricing of a specific type of Mountain Range option, viz. Himalaya options. A Himalaya option is a type of structured product which may be classified as a multi-asset path-dependent European option. The complex nature of the basic Himalayan structure infers that closed-form valuation is not possible. We examine three variants of the basic Himalayan structure and present various numerical techniques for valuing the different structures. In particular, we examine Monte Carlo and Quasi-Monte Carlo Methods for numerical integration. Finally, we present an approximation algorithm for the valuation of a deterministic variant of the Himalaya option.

Contents

Chapter 1

Introduction

1.1 Himalaya Options

Mountain Range options were introduced as a new generation of exotic equity options in 1998 by Société Générale. These derivatives, although classified as being multi-asset European options in nature, differ significantly from traditional European basket options. Mountain Range options amalgamate the path-dependency of classic barrier options with the multi-dimensionality of traditional basket options to produce an entirely new dimension to the field of derivative pricing and risk management.

The first major issue when dealing with Mountain Range options is that their pricing structure is primarily dependent on the correlation between the stocks underlying the option. In practice, historical correlations are often used as estimates for future periods, thus significant error can be propagated through the pricing methodology if markets are volatile or historical data is inconsistent. Thus, significant attention is required in this area to ensure acceptable results from pricing models. The second issue, which is of primary concern in practice, is the difficulty associated with identifying the volatility smile associated with such products. This is particularly significant in the case of Himalaya options as the volatility smile becomes distorted when stocks are removed from the basket at the specified measurement dates. For a more detailed discussion on this issue, refer to [21].

We now introduce the Himalaya Option, a specific type of Mountain Range option, whose basic structure may be described as a call on the sum of the best performers of a basket of stocks over a particular time horizon. At initiation a set of measurement dates are specified, the number of measurement dates being equal to the number of underlying assets. The unique feature of this option is the withdrawal of the best performer from the basket after each pre-specified measurement date,

having secured that performance for the elapsed period, until only one stock remains.

In this project, we shall focus primarily on Himalaya options written on baskets consisting of a maximum of five underlying assets. Following the variations to the Himalaya option presented in [18], we will now present the three variants of the Himalaya option that will be considered for analysis.

1.2 Variants of the Himalaya Option

Consider the general case of a Himalaya option written on n underlying stocks. Suppose the time horizon under consideration is $[0, T]$, where time 0 represents the valuation date and time T represents the maturity date of the contract. Now, we partition the time horizon $[0, T]$ as follows :

$$[t_0, t_1]\,,\ [t_1, t_2]\,,\ \ldots\ ,\ [t_{n-1}, t_n]$$

where $t_0 = 0$ and $t_n = T$, all time measured in years. Here the times $t_1, t_2, \ldots, t_n$ represent measurement dates during the life of the contract.

We now introduce further notation :

- Let $S(i, j)$ denote the price of share i at time t_j.
- Let $P(i, j) = \frac{S(i,j)}{S(i,0)}$ denote the performance of share i over the period $[t_0, t_j]$.
- Let $p(i, j) = \frac{S(i,j)}{S(i,j-1)}$ denote the performance of share i over the period $[t_{j-1}, t_j]$.

where $i, j \in \{1, 2, \ldots, n\}$. Furthermore,

- Let $C(j, n)$[1] denote the accumulation factor over the period $[t_j, t_n]$.

where $j \in \{0, 1, 2, \ldots, n-1\}$. Also,

- Let $\{I_j\}_{j=1}^{n}$ denote a sequence of index sets, such that I_j contains the indices of the stocks that are still *alive* in the contract over the period $[t_{j-1}, t_j]$.
- Let M denote a function such that $M : I_j \mapsto I_j$, where $j \in \{1, 2, \ldots, n\}$. More explicitly,
$$M(I_j) = \sum_{i \in I_j} \mathbb{I}\{\max_{k \in I_j}\{P(k, j)\} = P(i, j)\}.$$

[1] $C(j, n) = \exp(r(0; t_j, t_n)(t_n - t_j))$, where $r(0; t_j, t_n)$ is the NACC forward rate which applies over the period $[t_j, t_n]$. Of course, here we are assuming that we have bootstrapped the risk-free NACC yield curve, where $r(0, t_i)$ denotes the relevant risk-free NACC rate which applies over the period $[t_0, t_i]$, where $i \in \{1, 2, \ldots, n\}$.

and analogously, we define the function m such that $m : I_j \mapsto I_j$, where $j \in \{1, 2, \ldots, n\}$, and[2]

$$m(I_j) = \sum_{i \in I_j} \mathbb{I}\{ \max_{k \in I_j}\{p(k,j)\} = p(i,j) \}.$$

We will denote the value of a Himalaya option at some time $t \in [0, T]$ by :

$$H_{\text{type}}^{\text{perf. type}}(t; 0, T, A) \ ,$$

where A represents the nominal value, time 0 the initiation and time T the maturity of the contract, and

$$\text{type} = \begin{cases} \text{GF or DGF} \ , & \text{Globally-floored Himalaya option,} \\ \text{LF or DLF} \ , & \text{Locally-floored Himalaya option,} \\ \text{M or DM} \ , & \text{Multiplicative Himalaya option.} \end{cases}$$

and

$$\text{perf. type} = \begin{cases} 1 \ , & \text{if the performance measure } P(i,j) \text{ is used,} \\ 2 \ , & \text{if the performance measure } p(i,j) \text{ is used.} \end{cases}$$

Thus, we consider the Globally-floored (GF), Locally-floored (LF) and Multiplicative (M) variants of the Himalaya option. All of these exhibit the basic structure mentioned earlier, viz. at each measurement date t_j the performance of the shares that are still *alive* in the contract, represented by the index set I_j, are measured according to the specified type, i.e. 1 or 2, and thereafter the best performer is withdrawn from the contract. The sequence of index sets $\{I_j\}_{j=1}^n$, together with the functions M and m enable us to record which stocks are *alive* in each of the measurement periods of the above mentioned structure. We can see this by the following :

$$\begin{aligned} &I_1 = \{1, 2, \ldots, n\} \ , \\ &I_j = I_{j-1} \setminus \{M(I_{j-1})\}, \text{ if perf. type} = 1 \ , \\ &I_j = I_{j-1} \setminus \{m(I_{j-1})\}, \text{ if perf. type} = 2 \ , \end{aligned}$$

where $j \in \{2, 3, \ldots, n\}$. In addition to this basic structure, we consider a deterministic structure of each of the three variants, which we denote by DGF, LGF and DM respectively. In this structure the buyer of the option specifies the sequence of index sets $\{I_j\}_{j=1}^n$ *a priori*, i.e. the buyer specifies at initiation which stock is to

[2] $\mathbb{I}(\omega)$ represents the indicator function of the event ω, i.e. $\mathbb{I}(\omega) = 1$, if ω occurs and is 0 otherwise.

be removed from the contract at each measurement date. A financial justification for the consideration of such a structure is that the end-user of such derivatives, are primarily asset managers whom have long term views regarding market movements and generally also believe that in a *bull market* a good performing stock will continue to *run*.[3] Thus, in this scenario the basic Himalayan structure may not be an attractive investment to the asset manager, whereas the deterministic structure may enhance the performance of his portfolio by allowing him the freedom to incorporate his views of the market into the derivative contract.

Using the notation developed above, we may represent the terminal payoffs of the three variants of the Himalaya option as follows :

Globally-floored Himalaya option

$$H^1_{\mathrm{GF/DGF}}(T;0,T,A) = A\max\left\{\sum_{j=1}^{n}\left(P(M(I_j),j)-1\right)C(j,n)\,,\,0\right\} \tag{1.1}$$

$$H^2_{\mathrm{GF/DGF}}(T;0,T,A) = A\max\left\{\sum_{j=1}^{n}\left(p(m(I_j),j)-1\right)C(j,n)\,,\,0\right\} \tag{1.2}$$

Locally-floored Himalaya option

$$H^1_{\mathrm{LF/DLF}}(T;0,T,A) = A\sum_{j=1}^{n}\max\left\{\left(P(M(I_j),j)-1\right)\,,\,0\right\}C(j,n) \tag{1.3}$$

$$H^2_{\mathrm{LF/DLF}}(T;0,T,A) = A\sum_{j=1}^{n}\max\left\{\left(p(m(I_j),j)-1\right)\,,\,0\right\}C(j,n) \tag{1.4}$$

Multiplicative Himalaya option

$$H^2_{\mathrm{M/DM}}(T;0,T,A) = A\max\left\{\prod_{j=1}^{n}p(m(I_j),j)-1,0\right\} \tag{1.5}$$

Recall that for the deterministic structure the sequence of index sets $\{I_j\}_{j=1}^n$ is specified at initiation of the contract by the buyer of the option.

[3] This means that the good performance of the stock price should, in general, continue into the future.

1.3 Structure of the Project

The main objective of this project is to analyze various numerical techniques for estimating the value of Himalaya options. Under a simplistic market model, described in the next chapter, we first apply the principle of risk-neutral valuation to represent the pricing problem as a multi-dimensional integral. Due to the complex nature of the payoffs associated with Himalaya options, only the deterministic-locally floored Himalaya option yields a closed-form solution, however this solution requires the ability to calculate multi-variate normal probabilities. All of the other himalayan structures do not yield closed-form solutions, therefore we have to resort to numerical techniques for estimating the value of the variants of the Himalaya option.

The first technique that we consider is the classical method of crude Monte Carlo integration. Thereafter, we seek to reduce the variance associated with the crude Monte Carlo estimator by analyzing some popular variance reduction techniques, viz. control variates, antithetic variates, moment matching and Latin hypercube sampling. We also discuss the application of these techniques to the problem of pricing Himalaya options.

The next numerical mechanism we consider is that of quasi Monte Carlo integration. We briefly discuss the main concepts underlying the method of quasi-Monte Carlo integration with particular emphasis on analyzing the error associated with a quasi-Monte Carlo estimator based on a particular low discrepancy sequence. Our study of quasi-Monte Carlo techniques also constitutes the analysis of the generation of certain low discrepancy sequences, using both open and closed rules. In particular, we consider the construction of Halton sequences, Faure Sequences, Sobol' sequences, Niederreiter sequences and Hammersley sets. Thereafter, we discuss the application of the method to the pricing of Himalaya options.

Finally, we present an approximation algorithm for valuing the deterministic locally-floored Himalaya option. This algorithm is based on the algorithm derived by Boyle and Tse, [3], for valuing options on the maximum of several assets. The final part of the project will provide a brief discussion on the numerical results obtained.

Chapter 2

Pricing Himalaya Options

In this section we shall present the problem of pricing the different variants of the Himalaya option. We shall discuss the assumptions and framework under which the pricing problem will be approached, and thereafter present the general problem of pricing a Himalaya option in the form of a multi-dimensional integral.

2.1 Assumptions

We shall consider pricing under the traditional Black-Scholes framework, as presented in [1]. In particular, though, we would like to emphasize certain characteristics concerning the risk-free and risky assets, which represent the tradable instruments in the market. Consider the time horizon $[0, T]$, representing the *life* of the option, then we have the following :

- A constant risk-free NACC[1] rate, r, which applies over the period $[0, T]$. We assume that r is the rate governing borrowing and lending over the specified period.
- Each stock has a constant continuous dividend yield, which applies over the period $[0, T]$. Let q_i denote the constant continuous dividend yield associated with the i^{th} stock, $S(i, \cdot)$, over the specified period.
- Each stock has an associated volatility which is assumed to be constant over the period $[0, T]$. Let σ_i denote the volatility associated with the i^{th} stock, $S(i, \cdot)$, over the specified period.

The last assumption above is of course the most problematic, as volatility is certainly not a constant. As noted in [21] and [18], the difficulty inherent in identifying

[1] Previously, when defining the terminal payoffs of the variants of the Himalaya option, we assumed the existence of a term structure for interest rates. This is, indeed, a more realistic interpretation and can be incorporated into the pricing models which follow.

the *volatility smile* or *skew* associated with this family of options severely hampers the choice of suitable pricing methodologies. This difficulty leads one to stochastic volatility models, and thereby the realm of incomplete markets. However, as discussed in [18] and [19], the Himalaya structure can be priced and hedged in an advanced Heston framework as well as under the Hull-White stochastic volatility model. For further information on these models refer to [11] and [12], respectively.

This project will focus on pricing the himalayan structures under the simplified market model specified above. Even within such a simplified framework, the analysis of these himalayan structures entails significant difficulty, leading to simulation and approximation algorithms for estimates of prices of the options.

We suppose that the market contains n dividend paying stocks, and we let $S(i,t)$ denote the price of the i^{th} stock at some time $t \in [0,T]$, a slight modification of earlier notation. We assume that the behavior of the price of each stock may be modelled as a stochastic process following *geometric Brownian motion*. Thus, we have the family of stock price processes determined by the *stochastic differential equations* :

$$dS(i,t) = \mu_i S(i,t)dt + \sigma_i S(i,t)dW(i,t) \tag{2.1}$$

where μ_i is the *drift rate* and σ_i is the *volatility* of the i^{th} stock, for $1 \leq i \leq n$ and $\{\mathbf{W}_t\}_{t\geq 0} = \{\,(W(1,t), W(2,t), \ldots, W(n,t))\,\}_{t\geq 0}$ is an n-dimensional Brownian motion under the market measure $\mathbb{P}$.

2.2 The Pricing Problem as a Multi-dimensional Integral

In modern financial mathematics, we model security prices as stochastic processes to reflect future uncertainty. The assumption of no arbitrage in the market then implies that market prices of securities, suitably normalized, are *martingales*. This martingale property further implies that we may write the current price of a security as an expectation of its future cash flows under a probability measure known as the *equivalent martingale measure* or the *risk-neutral measure*, which we shall denote by $\mathbb{Q}$. Furthermore, the assumption of a complete market implies that such a measure is unique. For a more detailed discussion on *martingale pricing theory* refer to [7].

Consider a Himalaya option written on n dividend paying stocks with initiation at time 0, thus the stock prices at time 0 are the basis for performance measurement,

and expiry of the contract at time T. We specify measurement dates $t_1, t_2, \ldots, t_n$ during the life of the the contract $[0, T]$, such that $t_1 < t_2 < \ldots < t_n = T$. Now, the Himalaya structure may be characterized as a path-dependent European option, thus the terminal payoff depends depends on the stock prices realized at each measurement time point. Under the martingale pricing framework, we now have the family of stock prices specified by the following *stochastic differential equations* :

$$dS(i,t) = (r - q_i)S(i,t)dt + \sigma_i S(i,t)dZ(i,t) \ , \tag{2.2}$$

where $(r - q_i)$ is the *risk-neutral drift rate* and σ_i is the *volatility* of the i^{th} stock, for $1 \leq i \leq n$ and $\{\mathbf{Z}_t\}_{t\geq 0} = \{ \, (\, Z(1,t), Z(2,t), \ldots, Z(n,t) \,) \, \}_{t\geq 0}$ is an n-dimensional Brownian motion under the risk-neutral measure $\mathbb{Q}$. Furthermore at each time t, $\mathbf{Z}_t$ has an n-dimensional normal distribution with mean vector $\mathbf{0}$ and variance-covariance matrix Σ_t, i.e. $\mathbf{Z}_t \sim N_n(\mathbf{0}, \Sigma_t)$, where

$$\Sigma_t = t \, P = \{\sigma_{ij}^t\} = \rho_{ij} \, t \ ,$$

and

$$P = \{p_{ij}\} = \rho_{ij} \ ,$$

is the correlation matrix, i.e. ρ_{ij} is the correlation between the instantaneous returns of stock i and stock j. The solution to the above family of stochastic differential equations, (2.2), at some time t_j given that we are presently at time t, $t_j > t$, is given by :

$$S(i,t_j) = S(i,t) \exp\left((r - q_i - \frac{1}{2}\sigma_i^2)(t_j - t) + \sigma_i (Z(i,t_j) - Z(i,t)) \right) \ , \tag{2.3}$$

for $1 \geq i \geq n$. Let $\mathbf{S}_{t_j} = (\, S(1,t_j), S(2,t_j), \ldots, S(n,t_j) \,)'$ denote the vector of stock prices at time t_j, $j \in \{1, 2, \ldots, n\}$. Now, using earlier notation, let $H(t; 0, T, A)$ be a generic function denoting the value of a Himalayan structure at any time $t \in [0, T]$, then we may re-write the terminal payoff of such a structure as :

$$H(T; 0, T, A) = h(\, \mathbf{S}_{t_1}, \mathbf{S}_{t_2}, \ldots, \mathbf{S}_{t_n} \,) \ ,$$

for some function $h : \mathbb{R}_+^{n^2} \mapsto \mathbb{R}$, representing one of the Himalaya payoffs introduced earlier. Using *martingale pricing theory*, assuming that we are at initiation of the contract, the current price of the derivative security is the discounted expected value of its payoff under the measure $\mathbb{Q}$, given by :

$$\begin{aligned} & H(0; 0, T, A) \\ &= \exp(-rT) \, \mathbb{E}_0^{\mathbb{Q}}[\, h(\, \mathbf{S}_{t_1}, \mathbf{S}_{t_2}, \ldots, \mathbf{S}_{t_n} \,) \,] \\ &= \exp(-rT) \int_{\mathbb{R}_+^{n^2}} h(\, \mathbf{S}_{t_1}, \mathbf{S}_{t_2}, \ldots, \mathbf{S}_{t_n} \,) \, f(\, \mathbf{S}_{t_1}, \mathbf{S}_{t_2}, \ldots, \mathbf{S}_{t_n} \,) \, d\mathbf{S}_{t_1} d\mathbf{S}_{t_2} \ldots d\mathbf{S}_{t_n}, \end{aligned} \tag{2.4}$$

where $f(\mathbf{S}_{t_1}, \mathbf{S}_{t_2}, \ldots, \mathbf{S}_{t_n})$ is the probability density function of an n^2-dimensional log-normal distribution.

Thus, the problem of valuing any of the variants of the Himalaya option, written on n assets, leads to an integral in dimension n^2. The degree of complexity of the integral is determined by the nature of the payoff function, h, coupled with the number of assets underlying the structure. In the case of the three variants of the Himalaya option that we consider, the basic himalayan structures yield no closed-form solutions, while the deterministic himalayan structures do in some cases yield closed-form solutions. However, it should be noted that these formulas require the ability to evaluate multivariate cumulative normal probabilities. The calculation of such cumulative multivariate normal probabilities leads almost always to estimation via numerical integration, although approximation algorithms do exist for the univariate, bivariate and trivariate cases. We will consider the valuation of the Himalaya options primarily via numerical integration, in particular, using Monte Carlo and quasi-Monte Carlo methods. In later chapters, however, we will present an approximation algorithm for valuing options on the maximum of several assets, the option as presented by Johnson in [14], with the ultimate goal of being able to present an algorithm for valuing the deterministic locally-floored Himalaya options.

Another important point that needs to be mentioned, at this early stage, is that we will consider valuation of the Himalaya options in the first measurement period only, i.e. we will find $H(t; 0, T, A)$ for $t \in [0, T]$. Due to the unique nature of the payoffs associated with Himalaya options, as time passes components of the payoff become deterministic coupled with a reduction in the number of shares underlying the option. Thus, at any time during the life of the contract, other than the first measurement period, the valuation of the option entails both a deterministic and stochastic component. The deterministic component is easily dealt with, while the stochastic component is identical to the initial structure of the option, however there will now be an appropriate reduction to the dimensions of the initial problem. Hence, we conclude that valuation of the option during the first measurement period is the key issue to deal with, and overcoming this problem will then facilitate valuation at any time during the life of the contract.

Chapter 3

Monte Carlo Integration

3.1 Introduction

As we have seen, the valuation of Himalaya options lies in our ability to evaluate multidimensional integrals. The general form of such an integral is given by (2.4) and as mentioned earlier, the complex structure of most payoffs associated with Himalaya options imply that closed-form solutions are not possible, while less complex structures which may be decomposed into *simpler* options, such as the locally-floored Himalaya option, still require estimation of cumulative multivariate normal probabilities. In view of such difficulties, we have to resort to numerical methods for estimating multi-dimensional integrals, in the process, thereby obtaining estimates for the value of the options.

A Himalaya option written on n assets results in an integral problem in dimension n^2. Thus, even when $n = 3$ the resulting problem requires numerical integration of an integral in nine dimensions. Classical multiple quadrature rules, for a review see [22], for numerical integration are very effective in low dimensions, however in higher dimensions these rules are computationally intensive and therefore impractical, traditionally referred to as the *curse of dimensionality*. In such high dimensional situations, a first resort is Monte Carlo integration. Consequently, our first attempt at valuing Himalaya options leads us to numerical integration via crude Monte Carlo integration.

3.2 Crude Monte Carlo Integration

In this section, we shall briefly review the method of crude Monte Carlo integration. Suppose that $\mathbf{X} \in \mathbb{R}^d$ is a real-valued random vector defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$, where $\Omega \subseteq \mathbb{R}^d$, $\mathcal{F}$ is a σ-field on Ω and $\mathbb{P}$ is the probability measure

defined on this space. Suppose that h is any function[1] such that $h : \mathbb{R}^d \mapsto \mathbb{R}$. We consider the general problem of evaluating an integral of the form

$$I_\Omega(h) = \mathbb{E}[\, h(\mathbf{X})\,] = \int_\Omega h(\mathbf{x})\, f(\mathbf{x})\, d\mathbf{x}\ , \tag{3.1}$$

where f is the probability density function of the random vector $\mathbf{X}$. Suppose now that $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ is a random sample of size N from the distribution of $\mathbf{X}$. Then, an unbiased estimator of the integral in (3.1) is given by

$$\hat{I}_{\Omega,N}(h) = \frac{1}{N} \sum_{i=1}^{N} h(\mathbf{x}_i)\ ,$$

and the strong law of large numbers ensures that the estimator $\hat{I}_{\Omega,N}(h)$ converges (*almost surely*) to the exact value $I_\Omega(h)$, as $N \to \infty$. Also, by the central limit theorem (for large N), we have that

$$\hat{I}_{\Omega,N}(h) \sim N\left(I_\Omega(h), \frac{\mathbb{V}\mathrm{ar}[\, h(\mathbf{X})\,]}{N}\right)\ ,$$

i.e. $\hat{I}_{\Omega,N}(h)$ is normally distributed with mean $I_\Omega(h)$ and variance $\frac{1}{N}\mathbb{V}\mathrm{ar}[\, h(\mathbf{X})\,]$. The variance of the estimator $\hat{I}_{\Omega,N}(h)$ may be estimated using the sample variance, which we define by

$$\widehat{\mathbb{V}\mathrm{ar}}_N[\, h(\mathbf{X})\,] = \frac{1}{N-1} \sum_{i=1}^{N} (\, h(\mathbf{x}_i) - \hat{I}_{\Omega,N}(h)\,)^2\ .$$

We may use the above sample variance to obtain a confidence interval for $\hat{I}_{\Omega,N}(h)$, which may then be used to obtain an error estimate for $|I_\Omega(h) - \hat{I}_{\Omega,N}(h)|$.

From the variance of the Monte Carlo estimator we can conclude that the Monte Carlo convergence rate is $O\left(\frac{1}{\sqrt{N}}\right)$, in a probabilistic sense. This convergence rate is considered to be too slow, however the simplicity of the technique coupled with the fact that it is independent of dimension makes crude Monte Carlo integration an effective tool that may be applied in a broad spectrum of real-world problems. It should also be noted, though, that techniques do exist to improve the rate of convergence of the crude Monte Carlo method and these are referred to as variance reduction techniques which will be presented in §3.3. We shall now use this technique to value the three variants of the Himalaya option presented in §1.2. The next section will provide general algorithms detailing the steps required to estimate the values of Himalaya options using the crude Monte Carlo technique.

[1] To ensure convergence, we do require that h satisfy the weak condition of almost sure continuity

3.2.1 Application to the Pricing Problem

We now consider the application of the crude Monte Carlo technique to the problem of pricing Himalaya options. As stated in §2.2, the general problem involves the evaluation of a multi-dimensional integral, the general form of such an integral is given by (2.4). At this preliminary stage, we can already see that in order to value any of the variants of the Himalaya option, in the context of Monte Carlo integration, requires the ability to simulate random variates from a multi-dimensional log-normal distribution. This seems a fairly daunting task, however upon further consideration one realizes that the problem actually requires the ability to generate correlated stock price paths over the life of the option.[2]

Since we have chosen geometric Brownian motion as the process governing the evolution of stock prices, we may generate correlated stock paths using the family of equations given by (2.3). Furthermore, equation (2.3) infers that this requires the ability to generate random vectors from the multivariate normal distribution, $N_n(\mathbf{0}, P)$, where P, defined in section §2.2, is the correlation matrix which is equivalent to the variance-covariance matrix in this case. The generation of such random vectors is achieved easily using the *Cholesky decomposition*[3] of the matrix P, provided that one has access to an efficient standard normal random number generator.

We now consider the pricing of the three variants of the Himalaya option, written on n stocks, assuming that all of the required pricing parameters have been specified. In what follows, we present a general algorithm for valuing Himalaya options using a crude Monte Carlo estimator based on a sample of size N. This algorithms applies to both the basic and deterministic himalayan structures as well as each of the three variants, viz. the globally-floored, locally-floored and multiplicative Himalaya options.

Algorithm 3.1. *Consider the valuation of the option at any time $t \in [t_0, t_1]$. Firstly, obtain the Cholesky decomposition of the correlation matrix, $P = CC'$. Now, at each measurement time $t_i \in \{t_1, t_2, \ldots, t_n\}$, repeat the following steps:*

1. *Generate N random vectors from the $N_n(\mathbf{0}, I)$ distribution, which we denote by the $n \times N$ matrix Z. Set $X = CZ$, thereby transforming each random vector,*

[2] This approach is actually equivalent to simulating the multi-dimensional log-normal distribution given in (2.4).

[3] The Cholesky decomposition is a matrix factorization technique for symmetric positive definite matrices, which enables us to write the $n \times n$ matrix P as $P = CC'$, where the matrix C is an $n \times n$ lower triangular matrix.

such that each column of X represents a random vector from the $N_n(\mathbf{0}, P)$ distribution.

2. *Using the $n \times N$ stock price matrix at the previous time point together with the matrix of random numbers, X, in equation (2.3), generate a new $n \times N$ stock price matrix.*

3. *Measure the performance of the stocks in each of the N columns of the stock price matrix, according to the specified performance type.*

4. *Record the return of the best performer in each of the N columns of the stock price matrix, according to the particular variant being considered, and add the return obtained from each column to the respective payoff being realized.*

5. *Remove the best performer from each of the N columns of the stock price matrix, which can be achieved in a computer program by replacing those particular components of the stock price matrix with degenerate variables.*

Note that, the i^{th} column of the stock price matrix represents the i^{th} simulated path followed by the n underlying stocks. Finally, we obtain the crude Monte Carlo estimate by discounting the average of the N recorded payoffs. Also, for the deterministic Himalayan structure we alter step 5, since we specify at initiation which stock is to be removed from the structure at each measurement date.

In the next section, we review certain numerical results that were obtained using the crude Monte Carlo method to estimate the value of certain variants of the Himalaya option.

3.2.2 Numerical Results

Rather than present results for all of the Himalaya options presented in §1.2, we have selected the following options:

1. Globally-floored Himalaya option with $p(\cdot, \cdot)$ as the designated performance measure. We denote this option by GF2.

2. Deterministic Locally-floored Himalaya option with $P(\cdot, \cdot)$ as the designated performance measure. We denote this option by DLF1.

We deal with the case of three underlying non-dividend paying stocks, and consider valuation of the options under four different scenarios. Further, we assume that the options have a term of $[0, T]$, where time 0 is the initiation and time T is the expiry of the contract, with measurement times $\{t_i\}_{i=1}^{3}$, such that $t_1 < t_2 < t_3 = T$.

We consider valuation at time 0 and have the following pricing parameters which remain the same in all four scenarios:

- $A = 1$ and $S(i, 0) = 100$, for each $i \in \{1, 2, 3\}$.
- $r = 10\%$, $T = \frac{3}{2}$ and $t_i = \frac{i}{2}$, for each $i \in \{1, 2, 3\}$.
- For the deterministic structures, we specify $I_1 = \{1, 2, 3\}$, $I_2 = \{2, 3\}$ and $I_3 = \{3\}$.

Therefore, the parameters that are varied are the volatilities and correlations of the underlying stocks. These are varied as follows:

Scenario One

- $\sigma_i = 20\%$, for each $i \in \{1, 2, 3\}$.
- $\rho_{ij} = 0.9$, for all $i, j \in \{1, 2, 3\}$, such that $i \neq j$.

Scenario Two

- $\sigma_i = 20\%$, for each $i \in \{1, 2, 3\}$.
- $\rho_{12} = -0.4$, $\rho_{13} = 0.6$ and $\rho_{23} = -0.2$.

Scenario Three

- $\sigma_i = 20\%$, for each $i \in \{1, 2, 3\}$.
- $\rho_{ij} = -0.4$, for all $i, j \in \{1, 2, 3\}$, such that $i \neq j$.

Scenario Four

- $\sigma_1 = 20\%$, $\sigma_2 = 25\%$ and $\sigma_3 = 30\%$.
- $\rho_{ij} = 0.9$, for all $i, j \in \{1, 2, 3\}$, such that $i \neq j$.

This setup will be maintained throughout the project, and all numerical results will be presented in the context of the above scenarios. Thus, we use this framework as a basis for comparing the different techniques that are applied to the pricing problem.

Results for the GF2 Himalaya Option

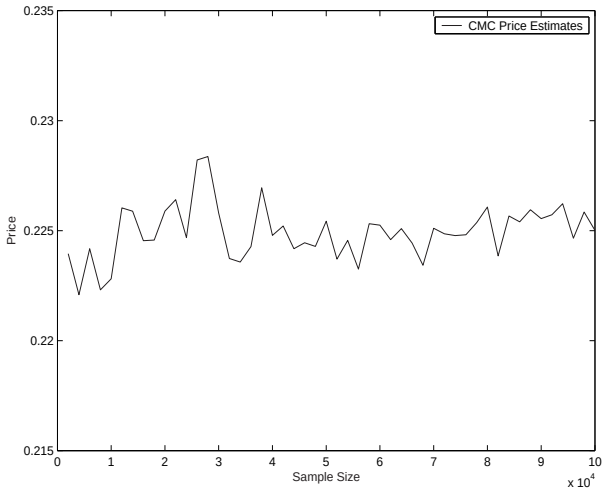

Fig. 3.1: *Plot of crude Monte Carlo price estimates for the GF2 Himalaya option vs. sample size, under Scenario One.*

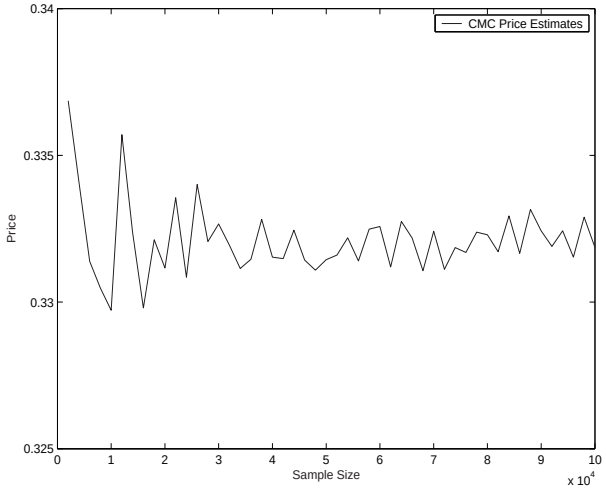

Fig. 3.2: *Plot of crude Monte Carlo price estimates for the GF2 Himalaya option vs. sample size, under Scenario Two.*

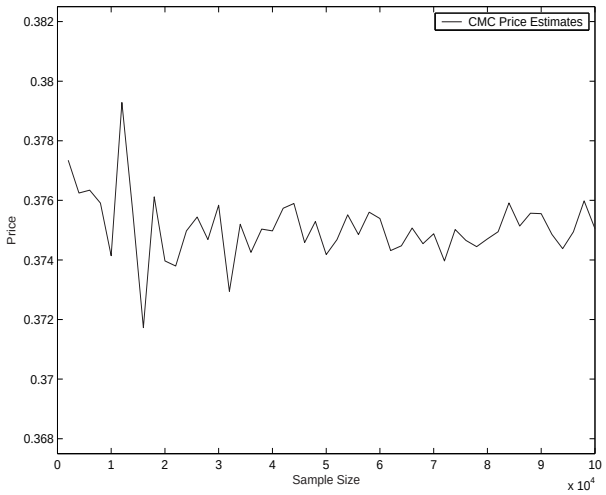

Fig. 3.3: *Plot of crude Monte Carlo price estimates for the GF2 Himalaya option vs. sample size, under Scenario Three.*

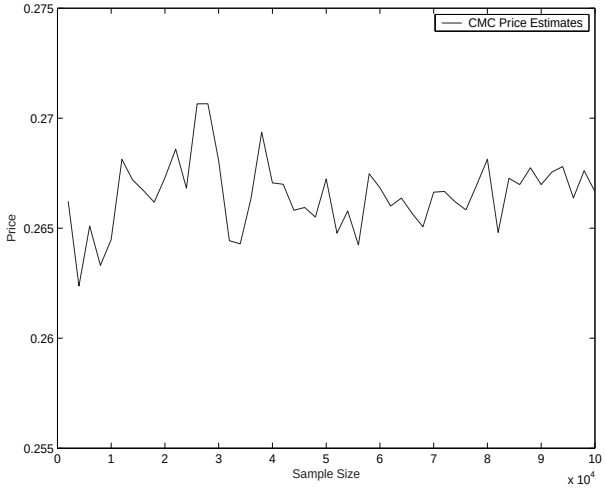

Fig. 3.4: *Plot of crude Monte Carlo price estimates for the GF2 Himalaya option vs. sample size, under Scenario Four.*

Results for the DLF1 Himalaya Option

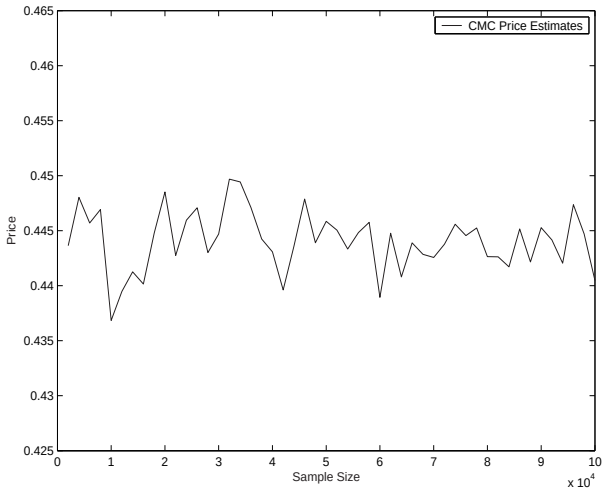

Fig. 3.5: *Plot of crude Monte Carlo price estimates for the DLF1 Himalaya option vs. sample size, under Scenario One.*

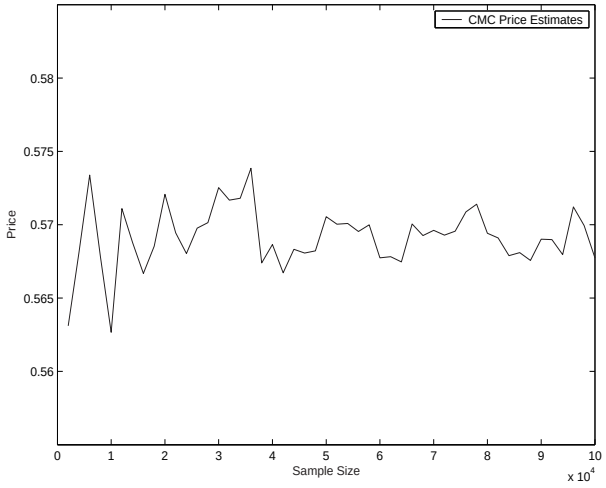

Fig. 3.6: *Plot of crude Monte Carlo price estimates for the DLF1 Himalaya option vs. sample size, under Scenario Two.*

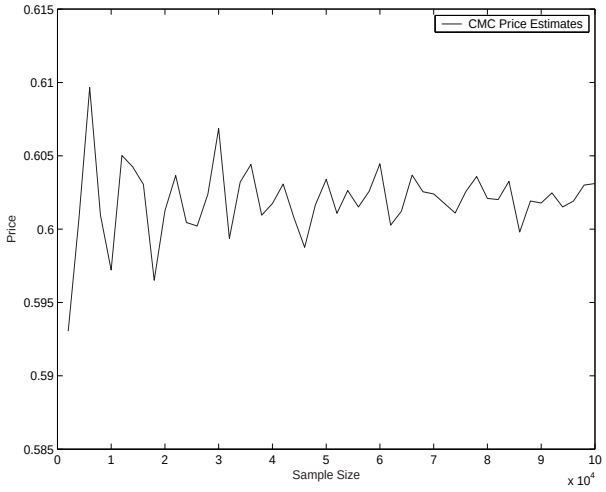

Fig. 3.7: *Plot of crude Monte Carlo price estimates for the DLF1 Himalaya option vs. sample size, under Scenario Three.*

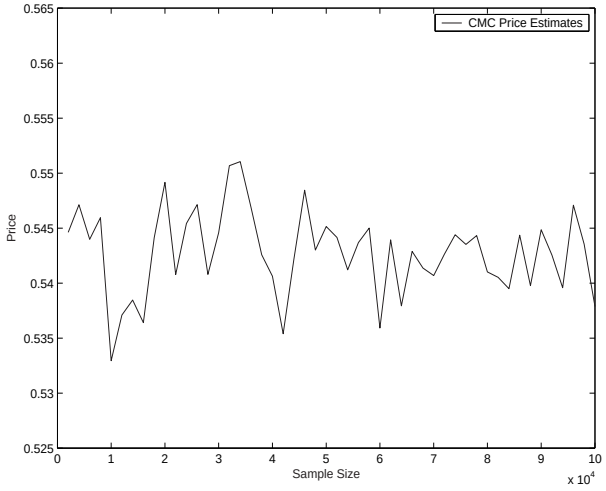

Fig. 3.8: *Plot of crude Monte Carlo price estimates for the DLF1 Himalaya option vs. sample size, under Scenario Four.*

3.3 Variance Reduction Techniques

The crude Monte Carlo estimator $\hat{I}_{\Omega,N}(h)$ can be improved by increasing the sample size N, however, this approach may be computationally intensive. This is evidenced by observing that improving the crude Monte Carlo estimate by a factor of ten requires a hundred-fold increase in the sample size. An alternative approach exploits the fact that

$$\mathbb{V}\mathrm{ar}[\,\hat{I}_{\Omega,N}(h)\,] = \frac{\mathbb{V}\mathrm{ar}[\,h(\mathbf{X})\,]}{N}\ ,$$

and thereby achieves an improved estimator by concentrating on reducing the magnitude of $\mathbb{V}\mathrm{ar}[\,h(\mathbf{X})\,]$. Such techniques are aptly referred to as variance reduction techniques and can be segmented into two broad categories. The first category of such techniques exploit tractable features of a model to appropriately adjust simulation output while the second category focusses on reducing the variability of inputs into a simulation.

As a final introductory note, it should be noted that the most effective use of variance reduction techniques is attained by identifying and exploiting particular, unique features of a problem. This point will become apparent when we apply the method of control variates to the Himalaya option pricing problem. The following sections will briefly outline some techniques that may be used to reduce the variance associated with a crude Monte Carlo estimator.

3.3.1 Antithetic Variates

The basic idea behind the method of antithetic variates is to attempt reducing the variance of the Monte Carlo estimate by producing negative dependence between pairs of simulation outputs. The method of antithetic variates takes on various forms, however we have chosen to analyze the method in the same context as presented in [2].

We now consider an integral of the form

$$I_{[0,1)^d}(g) = \mathbb{E}[\,g(\mathbf{Y})\,] = \int_{[0,1)^d} g(\mathbf{y})\,d\mathbf{y}\ , \tag{3.2}$$

where $\mathbf{Y}$ is uniformly distributed over the region $[0,1)^d$, i.e. $\mathbf{Y} \sim U[0,1)^d$. The integrals that are of concern to us are of the form of (3.1), however it is possible to transform such integrals into integrals over the d-dimensional unit cube. Consider a random sample $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$ of size N from the distribution of $\mathbf{Y}$, then we may form the Monte Carlo estimator of the integral in (3.2) as

$$\hat{I}_{[0,1)^d,N}(g) = \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{y}_i)\ .$$

Now, we use the fact that if $\mathbf{Y} \sim U[0,1)^d$ then $\mathbf{1}_d - \mathbf{Y}$ is also uniformly distributed over the region $[0,1)^d$, where $\mathbf{1}_d$ denotes the d-dimensional vector of ones. This observation forms the basis of the method of antithetic variates, as we now have a method of inducing negative dependence between pairs of simulation inputs. This is achieved, very simply, by transforming our original sample to produce the antithetic sample given by $\mathbf{1}_d - \mathbf{y}_1, \mathbf{1}_d - \mathbf{y}_2, \ldots, \mathbf{1}_d - \mathbf{y}_N$. Consequently, we may now define a new unbiased estimator of (3.2) based on the antithetic sample, as

$$I^{\text{as}}_{[0,1)^d,N}(g) = \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{1}_d - \mathbf{y}_i) \; .$$

Note that the variance of the above estimator is identical to that of the crude Monte Carlo estimator. The antithetic Monte Carlo estimator of (3.2) is defined by

$$I^{\text{av}}_{[0,1)^d,N}(g) = \frac{1}{2}\left(I^{\text{as}}_{[0,1)^d,N}(g) + \hat{I}_{[0,1)^d,N}(g) \right) \; .$$

Essentially, we now have an estimator based on a sample size of $2N$ but unlike the traditional crude Monte Carlo estimator where we are ensured an improved estimate by doubling the sample size, the antithetic estimator is based on two samples of size N which exhibit dependence thus we are not ensured that we have attained an improved estimate. To verify that we have indeed obtained a better estimator we must consider the following observations.

First, we consider the variance of the antithetic estimator, which yields :

$$\begin{aligned} & \mathbb{V}\text{ar}\left[I^{\text{av}}_{[0,1)^d,N}(g) \right] \\ &= \mathbb{V}\text{ar}\left[\frac{1}{2}\left(I^{\text{as}}_{[0,1)^d,N}(g) + \hat{I}_{[0,1)^d,N}(g) \right) \right] \\ &= \frac{1}{2}\mathbb{V}\text{ar}\left[\hat{I}_{[0,1)^d,N}(g) \right] + \frac{1}{2}\mathbb{V}\text{ar}\left[\hat{I}_{[0,1)^d,N}(g) \right] \rho\left[\hat{I}_{[0,1)^d,N}(g), I^{\text{as}}_{[0,1)^d,N}(g) \right] \; , \end{aligned} \tag{3.3}$$

where we have used the fact that $\mathbb{V}\text{ar}\left[I^{\text{as}}_{[0,1)^d,N}(g) \right] = \mathbb{V}\text{ar}\left[\hat{I}_{[0,1)^d,N}(g) \right]$, and where $\rho[\,\mathbf{A},\mathbf{B}\,]$ denotes the correlation between two random variables $\mathbf{A}$ and $\mathbf{B}$. From equation (3.3), it follows that a reduction in variance will be obtained if and only if $\hat{I}_{[0,1)^d,N}(g)$ and $I^{\text{as}}_{[0,1)^d,N}(g)$ are not perfectly positively correlated. However, we also have assurance that the resulting antithetic estimator will never be worse than our original crude Monte Carlo estimator.

A second observation to be considered, relates directly to the generation of random variates involved in the Monte Carlo simulation, which in this scenario is $U[0,1)^d$ variates. If the random variates involved in a Monte Carlo simulation can be generated efficiently such that the reduction in computation time obtained by doubling

the sample size via antithetic variates rather than doubling the sample size by generating independent random variates is negligible, then the use of antithetic variates will only be viable if

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left[\, I^{\mathrm{av}}_{[0,1)^d,N}(g)\,\right] &\leq \mathbb{V}\mathrm{ar}\left[\, \hat{I}_{[0,1)^d,2N}(g)\,\right] \\ &= \frac{1}{2}\mathbb{V}\mathrm{ar}\left[\, \hat{I}_{[0,1)^d,N}(g)\,\right].\end{aligned}$$

Therefore, from equation (3.3) we require the correlation between the crude Monte Carlo estimator, $\hat{I}_{[0,1)^d,N}$, and the antithetic sample estimator, $I^{\mathrm{as}}_{[0,1)^d,N}$, to be less than or equal to zero. For a more detailed discussion on antithetic variates, refer to [9] and [2].

3.3.2 Control Variates

The method of control variates seeks to reduce the variance associated with a particular crude Monte Carlo integral estimate by exploiting known information about the error associated with estimates of another known integral (with some similarity to the original integral). To present the technique, we again consider the estimation of the integral in (3.1), viz.

$$I_\Omega(h) = \int_\Omega h(\mathbf{x})\, f(\mathbf{x})\, d\mathbf{x} = \mathbb{E}[\, h(\mathbf{X})\,]\,.$$

Suppose that $k : \mathbb{R}^d \mapsto \mathbb{R}$ is any other function such that

$$I_\Omega(k) = \int_\Omega k(\mathbf{x})\, f(\mathbf{x})\, d\mathbf{x} = \mathbb{E}[\, k(\mathbf{X})\,]\,, \tag{3.4}$$

is a known quantity. Consider a random sample $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ of size N from the distribution of $\mathbf{X}$. Then we may produce crude Monte Carlo estimates of both the integrals, denoted by $\hat{I}_{\Omega,N}(h)$ and $\hat{I}_{\Omega,N}(k)$ respectively. Now, based on this random sample we can compare the Monte Carlo estimate of the integral in (3.4), denoted by $\hat{I}_{\Omega,N}(k)$, to the actual value of the integral, denoted by $I_\Omega(k)$, and there by obtain an estimate of the error or bias present in the random sample. This estimate of the error present in the random sample can then be utilized to appropriately adjust the Monte Carlo estimate of the integral in (3.1), which is a rough description of the method of control variates. More formally, the control variate Monte Carlo estimator is given by

$$I^{\mathrm{cv}}_{\Omega,N}(h) = \hat{I}_{\Omega,N}(h) - \beta\left(\, \hat{I}_{\Omega,N}(k) - I_\Omega(k)\,\right)\,, \tag{3.5}$$

where β is a fixed constant, also note that $I^{\mathrm{cv}}_{\Omega,N}(h)$ is an unbiased estimator of $I_\Omega(h)$.

The variance of the control variate estimator is given by

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left[\, I^{\mathrm{cv}}_{\Omega,N}(h)\,\right] &= \mathbb{V}\mathrm{ar}\left[\,\hat{I}_{\Omega,N}(h)\,\right] + \beta^2\mathbb{V}\mathrm{ar}\left[\,\hat{I}_{\Omega,N}(k)\,\right] - 2\beta\,\mathbb{C}\mathrm{ov}\left[\,\hat{I}_{\Omega,N}(h),\hat{I}_{\Omega,N}(k)\,\right] \\ &= \frac{1}{N}\left(\,\mathbb{V}\mathrm{ar}\left[\,h(\mathbf{X})\,\right] + \beta^2\mathbb{V}\mathrm{ar}\left[\,k(\mathbf{X})\,\right] - 2\beta\,\mathbb{C}\mathrm{ov}\left[\,h(\mathbf{X}),k(\mathbf{X})\,\right]\,\right).\end{aligned}$$

The optimal value of the parameter β is defined to be that value which minimizes the variance of the control variate estimator, which we denote by $\beta^\star$. From the above expression, it follows that the optimal value for β is given by

$$\beta^\star = \frac{\mathbb{C}\mathrm{ov}\left[\,h(\mathbf{X}),k(\mathbf{X})\,\right]}{\mathbb{V}\mathrm{ar}\left[\,h(\mathbf{X})\,\right]}\,. \tag{3.6}$$

If we substitute this optimal value of β into the expression for the variance of the control variate estimator, $I^{\mathrm{cv}}_{\Omega,N}(h)$, and simplify and then divide the resulting expression by the variance of the crude Monte Carlo estimator, $\hat{I}_{\Omega,N}(h)$, we obtain the following expression for the ratio of the variance of the optimal control variate estimator to that of the variance of the crude Monte Carlo estimator

$$\begin{aligned}\frac{\mathbb{V}\mathrm{ar}\left[\, I^{\mathrm{cv}}_{\Omega,N}(h)\,\right]}{\mathbb{V}\mathrm{ar}\left[\,\hat{I}_{\Omega,N}(h)\,\right]} &= 1 - \left(\,\rho\left[\,\hat{I}_{\Omega,N}(h),\hat{I}_{\Omega,N}(k)\,\right]\,\right)^2 \\ &= 1 - \left(\,\rho[\,h(\mathbf{X}),k(\mathbf{X})\,]\,\right)^2\,. \qquad (3.7)\end{aligned}$$

The above expression provides us with useful information pertaining to both the choice of a suitable control variate as well as the performance of a chosen control variate.

In particular, we may assert that when the optimal value for β is used, the performance of a particular control variate is directly dependent on the magnitude of the correlation between the random variable $h(\mathbf{X})$ and the control random variable $k(\mathbf{X})$. Clearly, if these random variables are independent then the use of the control variate estimator is worthless. The expression (3.7) thus strongly suggests that an effective control variate estimator would require a high degree of correlation between the control random variable and the random variable of interest. We shall exploit this particular observation in our search for effective control variate estimators for the values of Himalaya options.

The above observations are all very pleasing, however it should be noted that these theoretical results hold exactly if and only if we can obtain the exact value for $\beta^\star$. In practice, the statistics that are used to evaluate $\beta^\star$ are not known, however we can extract most of the benefit of a particular control variate estimator by estimating

the optimal value for β using sample statistics. The sample estimate for $\beta^\star$ based on a random sample of size N_1, which we denote by $\hat{\beta}^\star_{N_1}$, is given by

$$\hat{\beta}^\star_{N_1} = \frac{\sum_{i=1}^{N_1} \Big(h(\mathbf{x}_i) - \hat{I}_{\Omega,N_1}(h) \Big) \Big(k(\mathbf{x}_i) - \hat{I}_{\Omega,N_1}(k) \Big)}{\sum_{i=1}^{N_1} \Big(h(\mathbf{x}_i) - \hat{I}_{\Omega,N_1}(h) \Big)^2} \,. \tag{3.8}$$

Replacing the true value $\beta^\star$ by the sample estimate $\hat{\beta}^\star_{N_1}$ does have certain repercussions. The sample estimate introduces bias into the estimation, however as noted in [10] this problem is easily solved by estimating the optimal value for β using a subsample N_1 of the total sample N, and then performing the actual simulation using the the remaining sample, of size $N - N_1$. For a more comprehensive discussion on the method of control variates refer to [10].

3.3.3 Moment Matching

The method of moment matching focusses on altering sample mean values produced in a Monte Carlo simulation to match the corresponding population means. Essentially then, this method adjusts finite sample means to coincide with population means that would be attained if an infinitely large sample could be generated. In derivative pricing, such a technique has an intuitive justification. A derivative security derives its value from underlying assets therefore, as stated in [10], one could argue that accurate pricing of the derivative cannot be attained without accurate pricing of the underlying securities. The technique we concentrate on, in the context of derivative pricing, is ensuring that the sample means of asset prices produced during Monte Carlo simulations match their population means under the risk-neutral valuation framework.

The method we consider is presented in [10] and is referred to as moment matching through path adjustments. Recall the notation formulated in §1.2 as well as the assumptions of §2.1. Consider a European derivative security written on n assets, with initiation at time 0 and maturity at time T. Then, under the risk-neutral measure, $\mathbb{Q}$, we have that $\mathbb{E}^{\mathbb{Q}}[\, S(i,T)\,] = e^{((r-q_i)T)} S(i,0)$, for each $i \in \{1, 2, \ldots, n\}$. Suppose we implement crude Monte Carlo integration to value the option, based on a random sample of size N. Thus, for each asset i we generate a random sample of size N at time T, denoted by $S_1(i,T), S_2(i,T), \ldots, S_N(i,T)$. We define the sample mean for the i^{th} random sample by

$$\overline{S(i,T)} = \frac{1}{N} \sum_{j=1}^{N} S_j(i,T) \,.$$

In general, for finite sample sizes the above sample means will not match the corresponding population means which means that the underlying securities have been mispriced, thereby implicitly asserting that the derivative contract has been mispriced. To ensure correspondence between sample and population means we can apply one of the following transformations to each of the n random samples of stock prices generated at time T, i.e. we can apply either

$$\tilde{S}_j(i,T) = S_j(i,T)\frac{e^{((r-q_i)T)}S(i,0)}{\overline{S(i,T)}} \ , \tag{3.9}$$

or

$$\tilde{S}_j(i,T) = S_j(i,T) + e^{((r-q_i)T)}S(i,0) - \overline{S(i,T)} \ , \tag{3.10}$$

for each $j \in \{1,2,\ldots,N\}$. The transformed random samples can now be used to value the derivative security. The additive transformation, (3.10), preserves the martingale property, however it may produce negative stock prices. For this reason, the multiplicative transformation, (3.9), is the preferred choice as it ensures positivity.

On a final note, transforming the original simulated asset prices in the above fashion alters their distribution, i.e. the resulting stock prices will not, in general, preserve the distribution of the original sample. This introduces bias into the outputs of the Monte Carlo simulation, however it is stated in [10] that this bias vanishes as the sample size N is increased, typically at a rate of $O\left(\frac{1}{N}\right)$.

3.3.4 Latin Hypercube Sampling

Latin hypercube sampling is an extension of one-dimensional stratified sampling techniques to multiple dimensions. Stratified sampling is a mechanism of sampling from a probability distribution by initially dividing the domain of the probability distribution into disjoint regions, called *strata*, and thereafter sampling from the probability distribution in a manner such that the empirical probabilities obtained from the sample coincide with the theoretical probabilities, for each *stratum*. Whereas in moment matching techniques we ensure that the moments of the sample distribution coincide with that of the population distribution, here we ensure that the sampled distribution is consistent with the population distribution, in terms of probability, in disjoint regions throughout the entire domain of the population probability distribution.

We again consider estimating an integral of the form of (3.2), viz.

$$I_{[0,1)^d}(g) = \mathbb{E}[\, g(\mathbf{Y})\,] = \int_{[0,1)^d} g(\mathbf{y})\, d\mathbf{y} \ ,$$

where $\mathbf{Y}$ is uniformly distributed over the region $[0,1)^d$, i.e. $\mathbf{Y} \sim U[0,1)^d$. Therefore crude Monte Carlo estimation would require the generation of random variates from the $U[0,1)^d$ distribution. Suppose that we wanted to apply the classical method of stratified sampling to generate a sample from the d-dimensional hypercube $[0,1)^d$. This would entail partitioning each coordinate axis into N strata, say, thereby producing N^d strata for the hypercube $[0,1)^d$. Consequently, to ensure that each stratum is sampled we require a sample size of at least N^d. An estimator based on this sample would require N^d evaluations of the integrand. Clearly then, for problems in high dimensions this method is computationally expensive and would lead to inefficiency.

Latin hypercube sampling avoids the inefficiencies of the classical stratification method by firstly stratifying each of the d one-dimensional marginal distributions using N equiprobable disjoint regions of the domain, where N denotes the size of the sample under consideration. Thereafter a suitable permutation of coordinates is employed to obtain a sample which ensures that the one-dimensional marginal distributions are stratified. This method was first introduced in [16]. We will now present an algorithm, which appears in [10], for generating Latin hypercube samples of size N from the $U[0,1)^d$ distribution.

Algorithm 3.2.

1. *Generate an $N \times d$ matrix $\mathbf{V}$ of independent $U[0,1)$ random variates.*
2. *Generate d independent random permutations $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_d$ of the set of integers $\{1, 2, \ldots, N\}$.*
3. *Set $\mathbf{S}(i,j) = N^{-1}(\,\mathbf{a}_j(i) - \mathbf{V}(i,j)\,)$, for $i = 1, 2, \ldots, N$; $j = 1, 2, \ldots, d$.*

Now, the i^{th} row of the matrix $\mathbf{S}$ represents a point in the d-dimensional hypercube, for each $i \in \{1, 2, \ldots, N\}$. Once we have drawn a Latin hypercube sample from the d-dimensional hypercube, we may construct a Latin Hypercube sample from any other distribution using the inverse transform method.

The Latin hypercube sample estimator of the integral in (3.2), is given by

$$I^{\text{lhs}}_{[0,1)^d,N}(h) = \frac{1}{N}\sum_{i=1}^{N} g(\,\mathbf{S}_{(i)}\,)\,, \tag{3.11}$$

where $\mathbf{S}_{(i)}$ denotes the i^{th} row of the matrix $\mathbf{P}$. Furthermore, it is stated in [10] that for $N \geq 2$ and any square-integrable function g, the variance of the Latin hypercube

sample estimator is bounded above by $\frac{\mathbb{V}\mathrm{ar}[\,g(\mathbf{Y})\,]}{N-1}$, where $Y \sim U[0,1)^d$. This means that the variance of the Latin hypercube sample estimator based on a sample of size N is smaller than the variance of the crude Monte Carlo estimator based on a sample of size $N-1$.

3.3.5 Application to the Pricing Problem

In this section, we shall briefly discuss how each of the variance reduction techniques can be applied to reduce the variance associated with the crude Monte Carlo estimator of the value of a Himalaya option. The first technique that we consider is that of antithetic variates.

Antithetic Variates

We discussed the method of antithetic variates in the context of estimating an integral whose domain of integration was the d-dimensional unit cube. Consequently, the estimation of the integral using crude Monte Carlo integration required the generation of a random sample from the $U[0,1)^d$ distribution. Using the inverse transform method, however, we can generalize the results obtained in §3.3.1 to any distribution.
Now, the crude Monte Carlo estimator for the value of the option requires the generation of random vectors from the $N_n(\mathbf{0}, I)$ distribution. Consider the random sample $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N$ from the $N_n(\mathbf{0}, I)$ distribution, then generalizing the results of §3.3.1 we have that the corresponding antithetic sample is given by $-\mathbf{z}_1, -\mathbf{z}_2, \ldots, -\mathbf{z}_N$.

Therefore, the method of antithetic variates may be implemented using a modified version of algorithm 3.1. This modification is achieved by replacing step one by:

1. *Generate N random vectors from the $N_n(\mathbf{0}, I)$ distribution, which we denote by the $n \times N$ matrix Z. Now, supplement this matrix with the antithetic sample and thereby produce the $n \times 2N$ matrix $Z_A = [Z, -Z]$. Set $X = CZ_A$, thereby transforming each random vector, such that each column of X represents a random vector from the $N_n(\mathbf{0}, P)$ distribution.*

Besides the above modification, the rest of the algorithm remains the same, except that we have essentially doubled the sample size. As a result, the resulting estimator is the antithetic Monte Carlo estimator.

Control Variates

Finding a control variate for any of the variants of the Himalaya option is a non-trivial task. Here we propose methods to find appropriate control variates for each of the three variants of the Himalaya option. The controls that we eventually utilize are simple portfolios consisting of either vanilla European call options or forward starting European call options. Each of these constituent options may be regarded as a call option on the simple return of the underlying stock, i.e. a call option written on the performance of a single underlying stock with a strike price of one. Hence, we can easily calculate the value of such portfolios of options analytically, using the principle of risk-neutral valuation.

Consider the structure of the globally- and locally-floored Himalaya options, with $P(\cdot,\cdot)$ as the designated performance measure. The terminal payoffs of these two options are given by

$$H^1_{\mathrm{GF/DGF}}(T;0,T,A) = A\max\left\{\sum_{j=1}^{n}\left(P(M(I_j),j)-1\right)C(j,n)\,,\,0\right\}\,,$$

$$H^1_{\mathrm{LF/DLF}}(T;0,T,A) = A\sum_{j=1}^{n}\max\left\{\left(P(M(I_j),j)-1\right)\,,\,0\right\}C(j,n)\,.$$

From the terminal payoffs given above, it is clear that if we could choose a portfolio of n vanilla European call options, such that the j^{th} call option is

- initiated at time 0 and expires at measurement time t_j, and
- is written on the performance of stock $M(I_j)$ with a strike price of one, and
- we reinvest the payoff at time t_j at the risk-free rate for maturity at time $t_n = T$,

then the terminal payoff provided by this portfolio will be perfectly correlated with the terminal payoff of the locally-floored Himalaya option, and highly positively correlated with the payoff of the globally-floored Himalaya option. Therefore, by the results presented in §3.3.2, such a portfolio of options will provide an adequate control variate for the globally- and locally-floored Himalaya options.

However, the sequence $\{M(I_j)\}_{j=1}^n$ is actually a family of random variables, where the value of $M(I_j)$ is known at time t_j. Hence the above formulation is not very useful, however, it does form a basis from which we can develop a method for finding an appropriate control. Note that, in this setting of n underlying assets, there are

$n!$ possible portfolios of options that we can choose from. We shall now present an algorithm which finds the appropriate portfolio, together with the sample estimate of the optimal coefficient $\beta^{\star}_{N_1}$.

Algorithm 3.3. *Based on a sample of size N_1, do the following:*

1. *Generate N_1 possible payoffs of the Himalaya option, under consideration.*
2. *Generate N_1 possible payoffs of each of the possible $n!$ portfolios of options.*
3. *Calculate sample correlation coefficients between the payoff of the Himalaya option and the payoff of each of the $n!$ possible portfolios.*
4. *Select the portfolio with the largest absolute value for the sample correlation coefficient.*
5. *Calculate the sample estimate $\hat{\beta}^{\star}_{N_1}$, based on the outcomes of this simulation.*

This approach, exploits the result implied by equation (3.7). Equation (3.7) ensures a smaller variance for the control variate Monte Carlo estimator if the control variable is correlated with the variable of interest. The larger the strength of the correlation, the more effective the control variate.

Similarly, for the globally- and locally-floored Himalaya options, with $p(\cdot,\cdot)$ as the designated performance measure. The terminal payoffs of these two options are given by equations (1.2) and (1.4), respectively. From these terminal payoffs, it is evident that if we could choose a portfolio of n forward starting European call options[4], such that the j^{th} call option is

- initiated at time 0, has a forward start date at time t_{j-1} and expiry at measurement time t_j, and
- is written on the performance of stock $m(I_j)$ with a strike price of one, and
- we reinvest the payoff at time t_j at the risk-free rate for maturity at time $t_n = T$,

then, again, the terminal payoff provided by this portfolio will be perfectly correlated with the terminal payoff of the locally-floored Himalaya option, and highly positively correlated with the payoff of the globally-floored Himalaya option. Once again there are $n!$ possible portfolios to choose from and we implement algorithm

[4] Of course, the first option will be a vanilla European Call option, written on the performance of stock $m(I_j)$, with initiation at time 0 and expiry at time t_1.

3.3, in order to find the appropriate control.

Finally, for the multiplicative Himalaya option we consider, as a control variate, a single vanilla European Call option written on the performance of one of the underlying stocks with initiation at time 0 and expiry at time $t_n = T$. Thus, there are n possible options to choose from and we implement a modified version of algorithm 3.3, to find an appropriate control variate.

On a final note, the analytic valuation of the options chosen as control variates is straightforward, therefore we will not present these formulae. Also, once the appropriate control variate has been selected it is simple to include the pricing of the control, via Monte Carlo, in our general algorithm given in §3.2.1. Once this is done, the control variate Monte Carlo estimator is calculated using equation (3.5), where β is replaced with $\hat{\beta}^{\star}_{N_1}$.

Moment Matching

We apply the method of moment matching to the problem of pricing Himalaya options via crude Monte Carlo integration by transforming each of the simulated stock price paths using equation (3.9). We only consider the multiplicative transform since it ensures positivity of security prices.

Therefore, the implementation of this method may be achieved easily by appropriately modifying algorithm 3.1. This modification entails replacing step two of algorithm 3.1. with

2. *Using the $n \times N$ stock price matrix at the previous time point together with the matrix of random numbers, X, in equation (2.3), generate a new $n \times N$ stock price matrix. Calculate the sample moments for each of the n stocks, by finding the average of each of the n rows of the stock price matrix. Then, apply equation (3.9) row-wise to the components of the stock price matrix.*

The rest of the algorithm remains unchanged and we refer to the resulting estimator as the moment matching Monte Carlo estimator for the value of the Himalaya option.

Latin Hypercube Sampling

In order to obtain a Latin hypercube sample estimator for the price of a Himalaya option, written on n assets, we need to generate a Latin Hypercube sample from the $N_n(\mathbf{0}, I)$ distribution. In §3.3.4 we gave an algorithm for generating a Latin hypercube sample from the $U[0,1)^d$ distribution. Therefore, we may obtain a Latin hypercube sample of size N from the $N_n(\mathbf{0}, I)$ distribution by generating a Latin hypercube sample of size N from the $U[0,1)^n$ distribution and thereafter apply the inverse cumulative normal function to each of the elements of this sample. Thus, an appropriate modification of algorithm 3.1 will facilitate the production of an estimator based on a Latin hypercube sample.

The algorithm is obtained by replacing the first step of algorithm 3.1 with

1. Generate a Latin hypercube sample of size N from the $U[0,1)^n$ distribution. Apply the inverse cumulative normal function to each of the elements of this sample, to obtain a Latin hypercube sample of size N from the $N_n(\mathbf{0}, I)$ distribution. Denote this sample by the $n \times N$ matrix Z_L. Set $X = CZ_L$, thereby transforming the sample, such that each column of X represents a random vector from the $N_n(\mathbf{0}, P)$ distribution.

The other components of the algorithm remains the same and the estimate produced is the Latin hypercube sample estimate.

3.3.6 Numerical Results

In this section, we implement the variance reduction techniques discussed above, in an attempt to reduce the variance of the price estimates produced by the crude Monte Carlo estimator in §3.2.2. Again, we consider the valuation of the GF2 and DLF1 Himalaya options under the four scenarios presented in §3.2.2. We obtain estimates for the value of the options using each of the variance reduction techniques presented above, and plot these results against the results obtained in §3.2.2. We do this for each of the four scenarios, and the following results were obtained.

Scenario One - GF2 Himalaya option

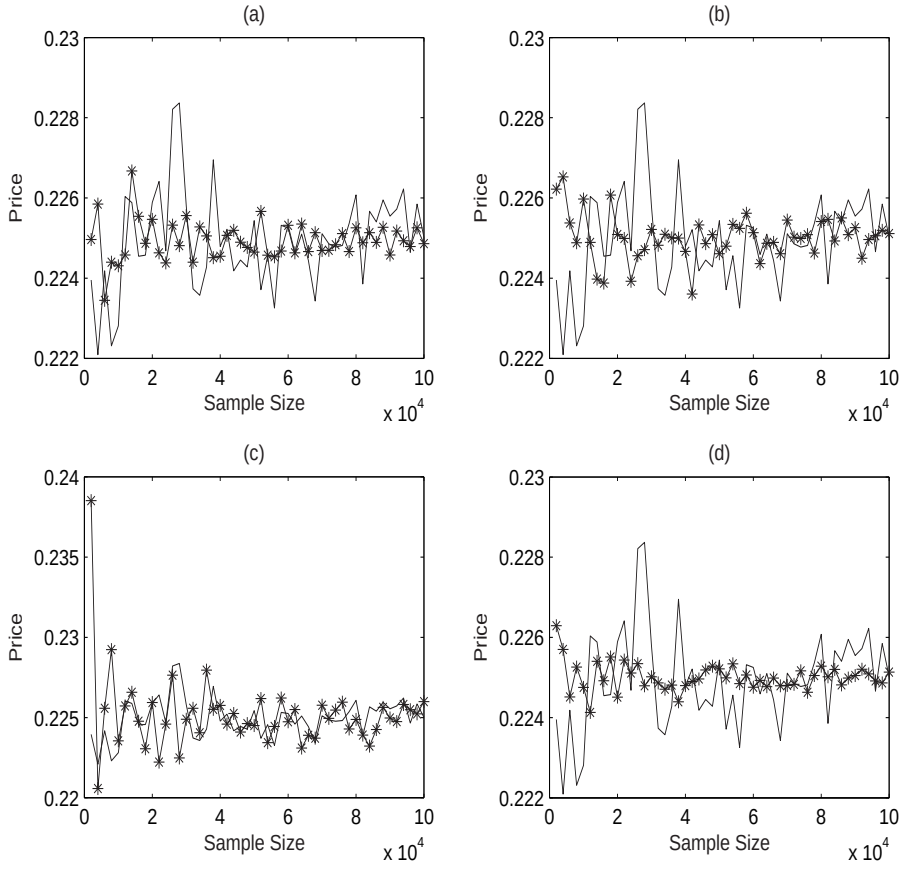

Fig. 3.9: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario Two - GF2 Himalaya option

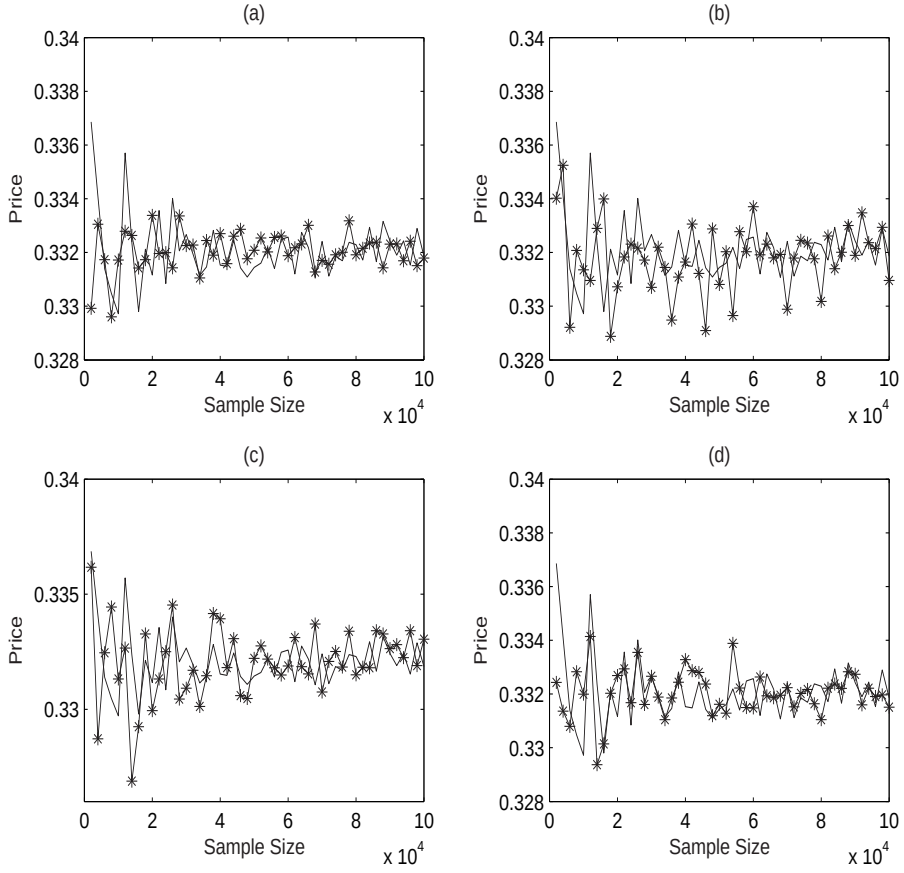

Fig. 3.10: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario Three - GF2 Himalaya option

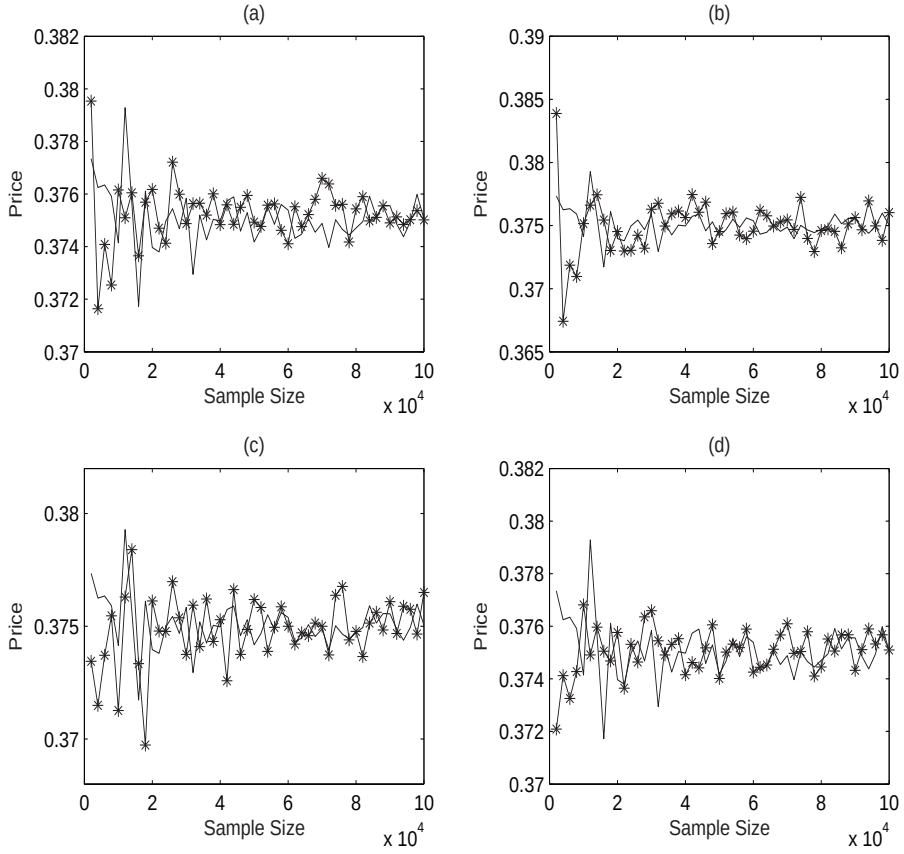

Fig. 3.11: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario Four - GF2 Himalaya option

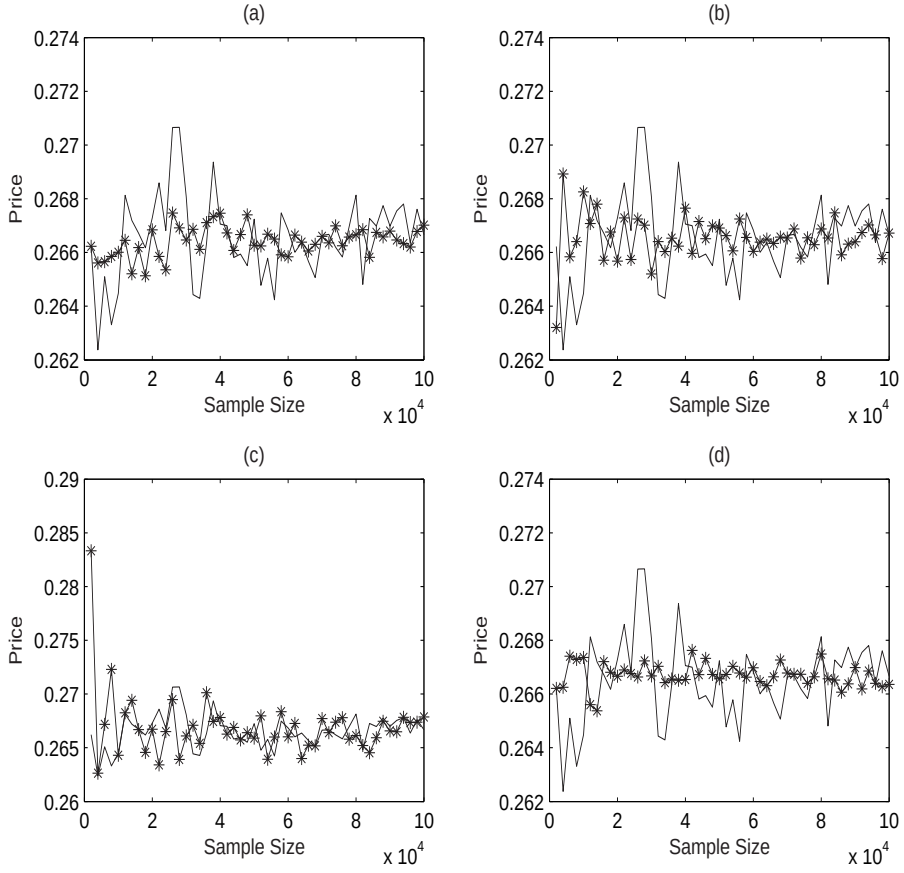

Fig. 3.12: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario One - DLF1 Himalaya option

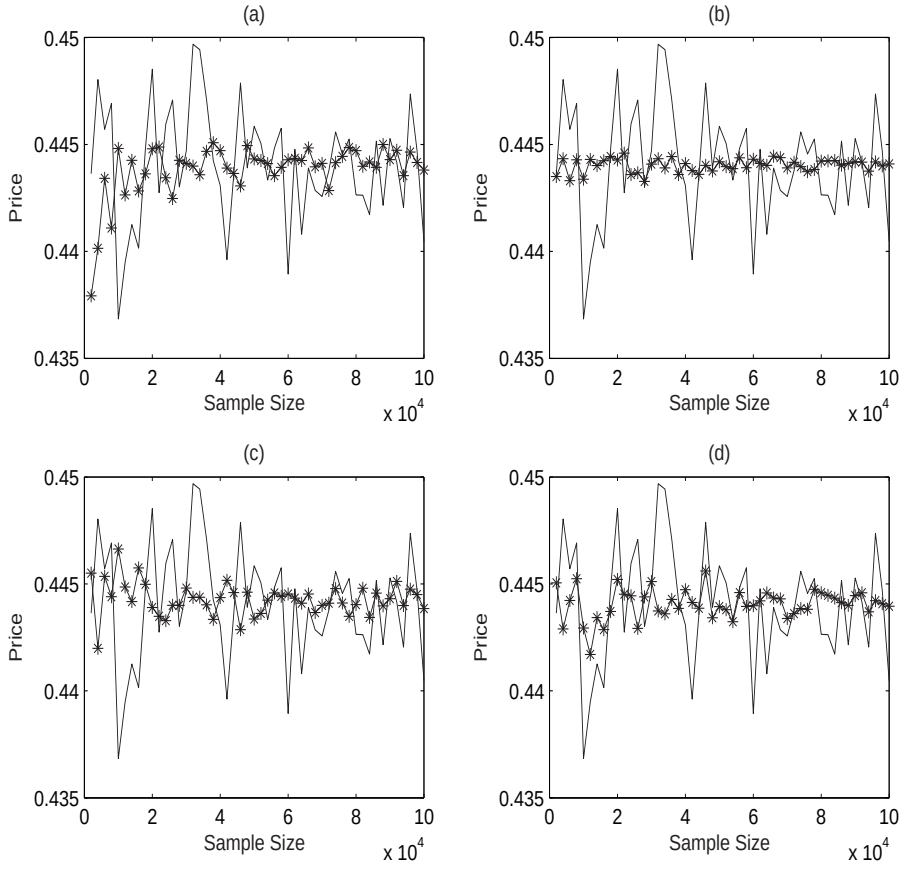

Fig. 3.13: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario Two - DLF1 Himalaya option

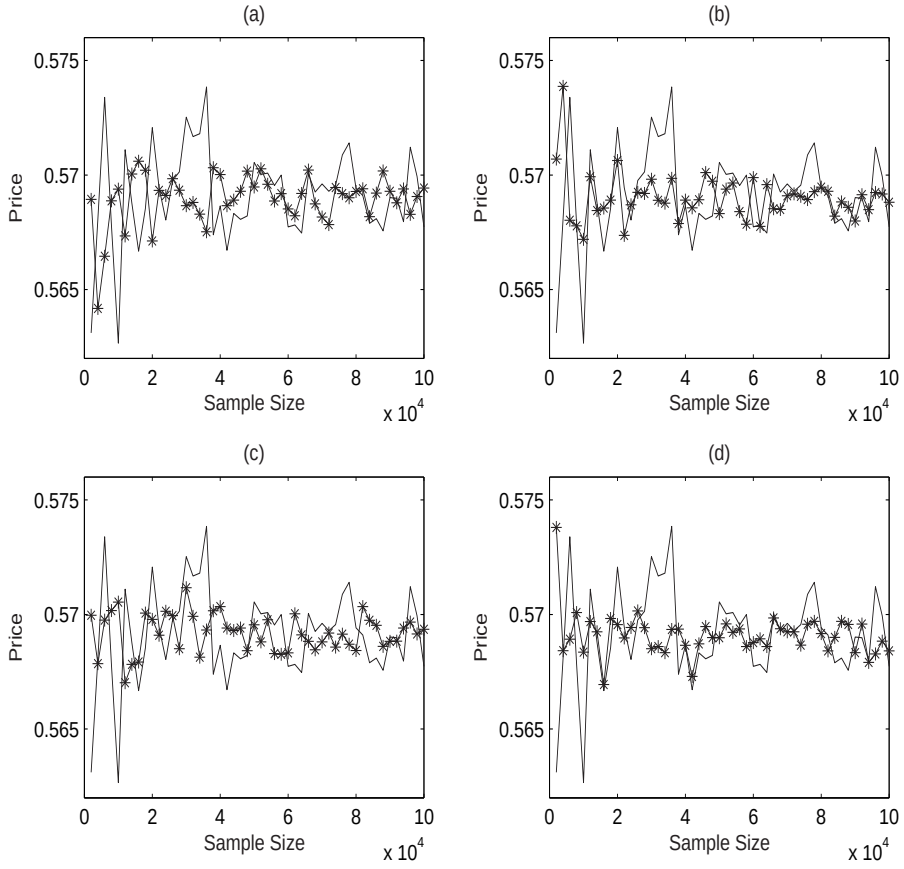

Fig. 3.14: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario Three - DLF1 Himalaya option

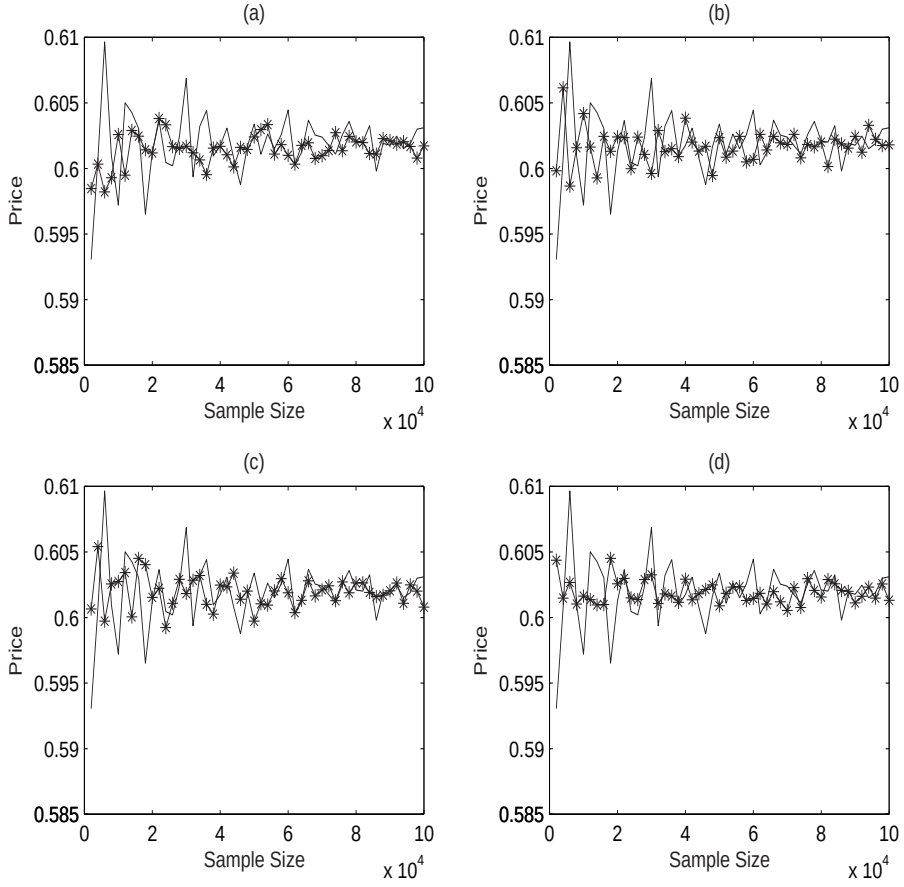

Fig. 3.15: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario Four - DLF1 Himalaya option

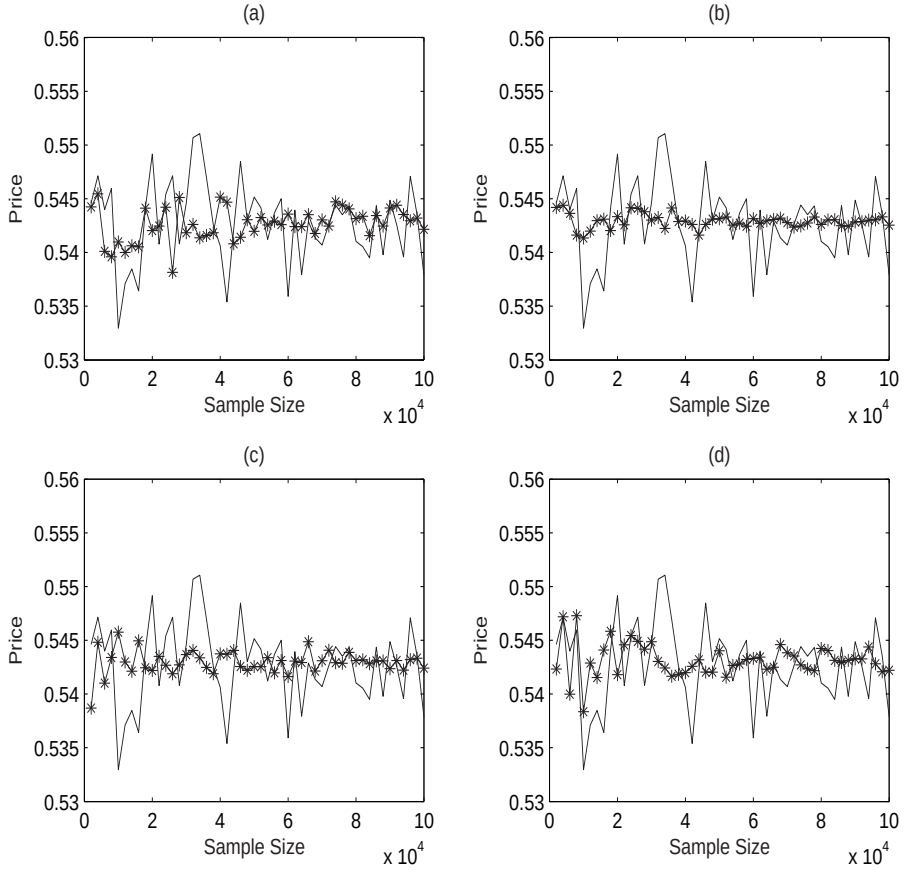

Fig. 3.16: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Chapter 4

Quasi-Monte Carlo Integration

4.1 Introduction

Monte Carlo methods are widely employed in a broad range of real-world problems and applications. The major drawback associated with classical Monte Carlo integration as a technique for integral estimation is the slow convergence rate of the Monte Carlo estimator. This slow convergence rate means that time consuming computations have to be made in order to obtain results with an acceptable level of error. As a direct consequence, a great deal of research has been devoted to the area of quasi-Monte Carlo methods for integral estimation. The popularity of this method is based on the faster convergence rate, which can be close to $O\left(\frac{1}{N}\right)$ under certain conditions. This deterministic alternative uses *low discrepancy sequences* rather than pseudo-random numbers. As a result, whereas classical Monte Carlo methods have their theoretical basis in probability theory, quasi-Monte Carlo methods draw their theoretical results from the field of abstract algebra.

Low discrepancy sequences form the crux of the quasi-Monte Carlo technique. Unlike pseudo-random numbers, such sequences are generated using deterministic rules with the general goal of filling the d-dimensional hypercube $[0,1)^d$ in an optimally regular manner. These deterministic rules are divided into two categories, viz. open rules and closed rules. We will provide definitions for both these rules when we consider the generation of low discrepancy sequences.

As mentioned above, low discrepancy sequences are used to deterministically generate points in the d-dimensional hypercube $[0,1)^d$. As a result, the quasi-Monte Carlo technique focusses on estimating integrals of the form

$$I_{[0,1)^d}(g) = \int_{[0,1)^d} g(\mathbf{y})\, d\mathbf{y}\ . \tag{4.1}$$

Unfortunately, this general form does not coincide exactly with our general problem of estimating integrals of the form of (3.1). However, as mentioned in §3.3, we may employ suitable transformations to transform integrals of the form of (3.1) to integrals over the d-dimensional unit cube. The final issue concerns the quasi-Monte Carlo estimator of the integral (4.1). As mentioned earlier, the theoretical basis of the quasi-Monte Carlo technique lies in the field of abstract algebra, therefore the strong law of large numbers is no longer at our disposal. However, a remarkable result known as *Weyl's criterion*[1] enables us to define the quasi-Monte Carlo estimator of the integral (4.1), based on the low discrepancy sequence $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N \in [0,1)^d$, as

$$I^{\text{qmc}}_{[0,1)^d,N}(g) = \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{y}_i)\ . \tag{4.2}$$

Finally, based on the recent popularity of the application of quasi-Monte Carlo techniques to derivative pricing, which began with the results obtained in [20], we will apply the quasi-Monte Carlo technique to the problem of pricing Himalaya options.

4.2 Error Analysis

In this section we consider the problem of estimating the error associated with a certain quasi-Monte Carlo estimator. Notice that, with Monte Carlo integration we can rely on confidence intervals for probabilistic estimates of the error associated with a certain crude Monte Carlo estimator, here however we cannot resort to probabilistic methods. In order to assess the quality of a certain quasi-Monte Carlo estimator, and thereby the effectiveness of the underlying low discrepancy sequence, we examine the rate of convergence of the quantity

$$\left| I^{\text{qmc}}_{[0,1)^d,N}(g) - I_{[0,1)^d}(g) \right| \quad , \quad \text{as } N \to \infty.$$

In order to facilitate further analysis we present certain definitions and theorems previously published in [17].

Definition 4.1. Suppose that $S = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ is a sequence of points in $[0,1)^d$. Let $\mathcal{L}$ be the Lebesgue measure on $[0,1)^d$, and let $\mathcal{B}$ be a non-empty set of $\mathcal{L}$-measurable subsets of $[0,1)^d$. Then we define a general notion of discrepancy for the sequence S as

$$D_N(\mathcal{B}, S) = \sup_{B\in\mathcal{B}} \left| \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\{\mathbf{y}_i \in B\} - \mathcal{L}(B) \right| .$$

[1] Weyl, Hermann, *Über die Gleichverteilung von Zahlen mod. Eins*, Math. Ann. **77**(1916), 313-352.

By specialization of the set $\mathcal{B}$, we can obtain two important definitions of discrepancy:
The discrepancy of the sequence S, denoted by $D_N(S)$, is defined for the set $\mathcal{B}$ consisting all subsets of $[0,1)^d$ of the form $\prod_{j=1}^{N}[s_j, t_j)$.
The star discrepancy of the sequence S, denoted by $D_N^*(S)$, is defined for the set $\mathcal{B}$ consisting all subsets of $[0,1)^d$ of the form $\prod_{j=1}^{N}[0, t_j)$.

Both these measures provide the worst approximation error obtained by the sequence S when used to estimate the volume of a hypercube contained in $[0,1)^d$. Whereas the discrepancy measure uses all hypercubes contained in $[0,1)^d$, the star discrepancy measure considers all hypercubes with a base point at the origin. Consequently, both provide a measure by which we can assess the quality in the uniformity of the distribution of the sequence S over the d-dimensional hypercube, $[0,1)^d$. Accordingly, we have the following theorem:

Theorem 4.2. *If $S = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ is a sequence of points in $[0,1)^d$, the following three propositions are equivalent:*

1. *S is uniformly distributed in $[0,1)^d$.*
2. $\lim_{N\to\infty} D_N(S) = 0$.
3. $\lim_{N\to\infty} D_N^*(S) = 0$.

The star discrepancy measure is computationally easier to deal with than the discrepancy measure, hence it is the preferred measure when assessing the uniformity of a particular sequence. Low discrepancy sequences are defined as those sequences whose discrepancies approach zero asymptotically.

Assume that the function g in (4.1) is d-times continuously differentiable on the domain $[0,1)^d$, i.e. $g \in C^d([0,1)^d)$, then we may analyze the error inherent in a quasi-Monte Carlo estimator of the integral (4.1) using the following results. First, we consider the variation associated with the function g.

Definition 4.3. The variation of a function g on $[0,1)^d$, in the sense of *Vitali* is defined by

$$V^{(d)}(g) = \int_{[0,1)^d} \left| \frac{\partial^d g}{\partial y_1 \ldots \partial y_d} \right| dy_1 \ldots dy_d \, .$$

Suppose that $1 \le k \le d$ and $1 \le i_1 < i_2 < \ldots < i_k \le d$, and let $V^{(k)}(g_{i_1,\ldots,i_k})$ be the Vitali variation of the function g restricted to the d-dimensional subset $\{(y_1, y_2, \ldots, y_d) \in [0,1)^d : y_i = 0 \text{ for } i \notin \{i_1, i_2, \ldots, i_k\}\}$.

Definition 4.4. The variation of a function g on $[0,1)^d$, in the sense of *Hardy and Krause* is defined by

$$V(g) = \sum_{k=1}^{d} \sum_{1 \le i_1 < i_2 < \ldots < i_k \le d} V^{(k)}(g_{i_1,\ldots,i_k}) \ .$$

Finally, we can relate the error associated with a quasi-Monte Carlo estimator to the discrepancy of the underlying quasi-random sequence using the following inequality.

Theorem 4.5. *For any sequence* $S = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ *in* $[0,1)^d$*, we have*

$$\left| I^{qmc}_{[0,1)^d,N}(g) - I_{[0,1)^d}(g) \right| \le V(g) D^*_N(S) \ .$$

This inequality is known as the Koksma-Hlawka Inequality.

This result gives us a strict upper bound for the error associated with a quasi-Monte Carlo estimator based on a particular sequence. Thus for any given function g our choice of quasi-random sequence will be based on the discrepancy measure of the sequence. However, if the variation of the the function g, in the sense of Hardy-Krause, is very small then the correct choice of quasi-random sequence becomes less relevant. Unfortunately, the computation of Hardy-Krause variation is, in general, very complex. Therefore, much research has been devoted to the search for low discrepancy sequences. For a more detailed discussion, including proofs of the theory presented in this section refer to [17].

4.3 Low Discrepancy Sequences

In this section, we shall consider the construction of certain low discrepancy sequences. More specifically, we consider sequences generated in the d-dimensional unit cube $[0,1)^d$, using both open and closed rules and also provide construction algorithms.

4.3.1 Open Rules

The defining property of a sequence of quasi-random numbers generated using open rules is that once we have generated a particular sequence, we can increase the sample size without having to regenerate an entirely new sequence. This is due to the fact that a sequence of length N is a subset of a sequence of length $N+1$. That is, an open rule will insert points into an existing sequence in a manner aimed at increasing the uniformity of the entire sequence. This feature is particularly advantageous if one considers efficiency of the generator as well as the amount of computation required to obtain a higher degree of accuracy when estimating integrals. We consider van

der Corput sequences, Halton sequences, Faure sequences, Sobol' sequences and Niederreiter sequences.[2]

4.3.2 Van der Corput Sequences

Van der Corput sequences refer to one-dimensional low discrepancy sequences. These sequences are generated using the representation of the set of integers in terms of an integer base b and thereafter applying the *base b radical inverse function* to this unique representation.
Every positive integer m can be represented uniquely in base b as

$$m = \sum_{i=0}^{\infty} a_i b^i \ , \tag{4.3}$$

where each $a_i \in \mathbb{Z}_b$.[3] Now, the base b radical inverse function maps each positive integer m to a point in the unit interval $[0, 1)$. We define this function by

$$\varphi_b(m) = \sum_{i=0}^{\infty} \frac{a_i}{b^{i+1}} \ .$$

Therefore, a van der Corput sequence in base b of size N, which we denote by $y_0, y_1, \ldots, y_{N-1}$, is obtained by setting $y_j = \varphi_b(j)$ for each $j \in \{0, 1, \ldots, N-1\}$. In [17], Niederreiter shows that the star discrepancy of the first N elements of any van der Corput sequence, denoted by the set $S = \{y_0, y_1, \ldots, y_{N-1}\}$, is given by

$$D_N^*(S) = O\left(\frac{\log N}{N}\right) \ ,$$

with an implicit constant depending on the base b.

4.3.3 Halton Sequences

Halton sequences may be characterized as multi-dimensional van der Corput sequences, since a Halton sequence in d-dimensions has as its j^{th} coordinate a van der Corput sequence in base b_j for each $j \in \{1, 2, \ldots, d\}$. Also, the bases $b_1, b_2, \ldots, b_d$ are chosen to be pairwise relatively prime integers greater than one. This eliminates correlation between coordinates and therefore ensures that the sequence fills up the the d-dimensional unit cube uniformly.

[2] More specifically, van der Corput sequences, Faure sequences, Sobol' sequences and Niederreiter sequences are examples of (t, s)-sequences. For more information on (t, s)-sequences refer to [17].

[3] Note that, $\mathbb{Z}_b$ represents the finite field of order b, i.e. this field consists of the integers $\{0, 1, \ldots, b-1\}$. Also the operations of addition and multiplication are closed, associative and commutative.

Therefore, a d-dimensional Halton sequence of size N, denoted by $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1}$, is obtained by setting

$$\mathbf{y}_j = (\varphi_{b_1}(j), \varphi_{b_2}(j), \ldots, \varphi_{b_d}(j)) ,$$

for each $j \in \{0, 1, \ldots, N-1\}$, with pairwise relatively prime integer bases $b_1, b_2, \ldots, b_d$ greater than one. The star discrepancy of the first N elements of the above Halton sequence, denoted by the set $S = \{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1}\}$, satisfies

$$D_N^*(S) \leq C_d(b_1, \ldots, b_d) \frac{(\log N)^d}{N} + O\left(\frac{(\log N)^{d-1}}{N} \right) .$$

Note that the coefficient $C_d(b_1, \ldots, b_d)$ is independent of N but does depend on the dimension d. Furthermore, this upper bound for the star discrepancy is minimized by taking the bases $b_1, b_2, \ldots, b_d$ to be the first d primes. These results are presented in [10] and [17]. In [17], Niederreiter also explains that d-dimensional Halton sequences, generated using the first d primes, exhibit very poor uniformity when d is large, i.e. in high dimensions, which is a direct consequence of the behavior of the van der Corput sequence with a large base. For a more comprehensive discussion refer to [10] and [17].

Kocis and Whiten, [15], introduced the concept of leaped Halton sequences as a possible solution to the poor uniformity characteristics exhibited by Halton sequences in high dimensions. Whereas a Halton sequence of size N is generated using the first N elements of the set $\mathbb{N}_0$[4], a leaped Halton sequence is generated using the first N elements of the set $\ell\,\mathbb{N}_0$, for any integer $\ell \geq 2$, called the leap parameter.

Thus, a d-dimensional leaped Halton sequence of length N, which we denote by $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1}$, is obtained by setting

$$\mathbf{y}_j = (\varphi_{b_1}(j\ell), \varphi_{b_2}(j\ell), \ldots, \varphi_{b_d}(j\ell)) ,$$

for each $j \in \{0, 1, \ldots, N-1\}$, where $b_1, b_2, \ldots, b_d$ are as defined above and ℓ is any integer greater than one. Kocis and Whiten further suggest that ℓ should be relatively prime to the bases $b_1, b_2, \ldots, b_d$, however, even with such a condition the quality of the resulting sequence is highly sensitive to the choice of leap parameter.

[4] $\mathbb{N}_0 = \{ 0, 1, 2, \ldots \}$.

4.3.4 Faure Sequences

In [8], Faure introduced an alternative extension of one-dimensional van der Corput sequences to higher dimensions to that proposed by Halton. Whereas Halton sequences utilize van der Corput sequences with large bases in high dimensions, Faure sequences use a single base b for all dimensions where b is chosen to be the smallest prime number greater or equal to the specified dimension d.

Consider a d-dimensional Faure sequence of length N, therefore, as mentioned above, the base b is chosen to be the smallest prime number greater or equal to d. We provide an algorithm to generate such a sequence based on the matrix-vector construction given in [10].
First we must define for each dimension $i \in \{1, 2, \ldots, d\}$, a generator matrix $\mathbf{C}^{(i)}$. These matrices have the property

$$\mathbf{C}^{(i)} = \left(\mathbf{C}^{(1)}\right)^i ,$$

for each $i \in \{1, 2, \ldots, d\}$. Also, each generator matrix $\mathbf{C}^{(i)}$ has dimensions $r \times r$, where $r = 1 + \lfloor \log(N-1)/\log(b) \rfloor$. The (m, n)-th entry of the matrix $\mathbf{C}^{(1)}$ is given by

$$\mathbf{C}^{(1)}(m, n) = (n-1)!/(m-1)!(n-m)! ,$$

if $n \geq m$ and zero otherwise. Thus, we have completely specified all the required generator matrices. We now have the following algorithm.

Algorithm 4.6. *For each $j \in \{0, 1, \ldots, N-1\}$:*

1. *We find the base-b expansion of j, using (4.3), and represent the coefficients of this expansion as the $r \times 1$ vector $\mathbf{a} = (a_0, a_1, \ldots, a_{r-1})'$.*

2. *For each coordinate $i \in \{1, 2, \ldots, d\}$, we define an $r \times 1$ vector $\mathbf{x}^{(i)}$ as*

$$\mathbf{x}^{(i)} = \mathbf{C}^{(i-1)}\mathbf{a} \mod b . \tag{4.4}$$

3. *Then, set*

$$\mathbf{y}_j = \left(\sum_{\ell=1}^{r} x_\ell^{(1)} b^{-\ell} , \sum_{\ell=1}^{r} x_\ell^{(2)} b^{-\ell} , \ldots , \sum_{\ell=1}^{r} x_\ell^{(d)} b^{-\ell} \right)' .$$

Finally, the sequence $S = \{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1}\}$ represents the d-dimensional Faure sequence. Further, Tezuka and Tokuyama, [24], introduce the notion of *generalized*

Faure sequences by generalizing the construction of the generator matrices introduced above. We also consider these sequences, which are generated by replacing equation (4.4) in the above algorithm with

$$\mathbf{x}^{(i)} = \left(\mathbf{C}^{(i-1)}\right)' \mathbf{C}^{(i-1)}\mathbf{a} \mod b\,. \tag{4.5}$$

For further details, refer to [24].

In [8], Faure shows that the star discrepancy of the sequence S is given by

$$D_N^*(S) \leq F_d \frac{(\log N)^d}{N} + O\left(\frac{(\log N)^{d-1}}{N}\right)\,,$$

where F_d depends on d only. This is also presented in [10], and confirms that Faure sequences are low discrepancy sequences.

4.3.5 Sobol' Sequences

Sobol' sequences are also multi-dimensional generalizations of the basic van der Corput sequence, however these sequences are generated by permuting segments of the van der Corput sequence in base 2, only. Each coordinate of a multi-dimensional Sobol' sequence has its own generator matrix which is constructed using a primitive polynomial over $\mathbb{Z}_2$, however the basic construction for each coordinate is the same. A primitive polynomial over $\mathbb{Z}_2$ is a polynomial of the form

$$p(x) = x^q + c_1 x^{q-1} + \ldots + c_{q-1}x + 1\,, \tag{4.6}$$

where the coefficients are elements of $\mathbb{Z}_2$, and q denotes the degree of the polynomial. We use a different primitive polynomial for each coordinate, which are used to generate *direction numbers* which completely specify the generator matrix. If we use a primitive polynomial of degree q then we require q initial direction numbers $m_1, m_2, \ldots, m_q$, where each m_ℓ must be an odd integer between 0 and 2^ℓ. We shall now present an algorithm for generating a d-dimensional Sobol' sequence of length N, based on the algorithm given in [4].

Algorithm 4.7. *For each $i \in \{1, 2, \ldots, d\}$, repeat the following steps.*

1. *Select a primitive polynomial over $\mathbb{Z}_2$ of degree q, say.*
2. *Select odd integers $m_1, m_2, \ldots, m_q$ satisfying $1 \leq m_\ell < 2^\ell$, for each ℓ.*
3. *For each $\ell \in \{q+1, q+2, \ldots, r\}$, set*

$$m_\ell = \left(2c_1 m_{\ell-1} \oplus 2^2 c_2 m_{\ell-2} \oplus \ldots \oplus 2^{q-1} c_{q-1} m_{\ell-q+1} \oplus 2^q m_{\ell-q} \oplus m_{\ell-q}\right)/2^\ell\,,$$

where $\oplus$ is the bit-wise exclusive-or operation, and $r = 1 + \lfloor \log(N-1)/\log(2) \rfloor$.

4. *For each $j \in \{0, 1, \ldots, N-1\}$, find the binary expansion of j, using (4.3), and set*

$$y_j^{(i)} = a_0 m_1 \oplus a_1 m_2 \oplus \ldots a_{r-1} m_r \ .$$

The sequence $S = \{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1}\}$, defines the d-dimensional Sobol' sequence of length N, where

$$\mathbf{y}_j = (y_j^{(1)}, y_j^{(2)}, \ldots, y_j^{(d)})$$

for each j. [4] as well as [10] provide a more comprehensive construction.

The most vital component in the construction of a d-dimensional Sobol' sequence is the choice of the initial direction numbers for each dimension of the sequence. It is essential that the sequence of initial direction numbers for each dimension be chosen differently, else a large number of the initial components of each of the d one-dimensional dimensional sequences will be identical. For further information, Glasserman, [10], provides a good discussion on the choice of initial direction numbers.

4.3.6 Niederreiter Sequences

The best possible upper bound for the discrepancy of a d-dimensional sequence S of length $N \geq 2$ is believed to be

$$D_N(S) \leq C_d (\log N)^d + O\left((\log N)^{d-1}\right) \ .$$

This is stated in [5] without proof, where it is also claimed that Niederreiter sequences produce the smallest values of C_d currently known. Niederreiter sequences are examples of (t, s)-sequences generated in any base $b \geq 2$. Further information is given in [5], and algorithms are provided to generate such sequences when the base b is a prime of the power of a prime, however special attention is attributed to the case where $b = 2$. On the basis of numerical investigation using several multi-dimensional test integrals, Bratley, Fox and Niederreiter [5] recommend the Niederreiter sequence in base 2, based on the criteria of computational efficiency as well as numerical accuracy. For this reason, we consider the construction of the Niederreiter sequence in base 2, and provide an algorithm for generating a d-dimensional version of this sequence based on an algorithm given in [5].

Now, we consider the following algorithm for generating a d-dimensional Niederreiter sequence in base 2 of length N. All arithmetic is performed over the finite field $\mathbb{Z}_2$.

Algorithm 4.8. *For each $i \in \{1, 2, \ldots, d\}$, repeat the following steps.*

1. *Select a primitive polynomial, $p(x)$, over $\mathbb{Z}_2$ of degree e, say. Set $n = 0$, $q = -1$ and $u = e$.*

2. *Set $n = n + 1$. If $u = e$ go to step 3 else, go to step 4.*

3. *Calculate*
$$p(x)^{q+1} = x^m - b_{m-1}x^{m-1} - \ldots - b_0 \, ,$$
where $m = e(q + 1)$. Also set $u = 0$, $q = q + 1$, $v_k = 0$ for $0 \leq k \leq m - 2$, $v_{m-1} = 1$, and
$$v_k = \sum_{\ell=1}^{m} b_{m-\ell} \, v_{k-\ell} \, ,$$
for $m \leq k \leq max + e - 2$, where $max = 1 + \lfloor \log(N - 1)/\log(2) \rfloor$.

4. *Set $c_{nr} = v_{r+u}$, for $0 \leq r \leq max - 1$, also set $u = u + 1$. If $n < max$ go to step 2, else stop.*

5. *For each $j \in \{0, 1, \ldots, N-1\}$, find the binary expansion of j, using (4.3), and set*
$$y_j^{(i)} = \sum_{n=1}^{max} y_{j,n} \, 2^{-k} \, ,$$
where
$$y_{j,n} = \left(\sum_{r=0}^{max-1} c_{n,r} \, a_r \right) \in \mathbb{Z}_2 \, ,$$
for $1 \leq n \leq max$.

Therefore, the sequence $S = \{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N-1}\}$, represents the d-dimensional Niederreiter sequence of length N, where

$$\mathbf{y}_j = (\, y_j^{(1)}, y_j^{(2)}, \ldots, y_j^{(d)} \,)$$

for each j. In [5], it is stated that for an arbitrary d-dimensional sequence, one should choose a sequence of primitive polynomials that are distinct and pairwise co-prime with degrees as small as possible. This is achieved by choosing the first d polynomials of the sequence of all primitive polynomials over $\mathbb{Z}_2$ sorted in order of increasing degree.

On a final note, it should be noted that we are not assured that the good asymptotic behavior of Niederreiter sequences will lead to improved quasi-Monte Carlo

estimates. In practical applications we are not able to exploit the asymptotic behavior of these sequences, therefore the performance of the corresponding quasi-Monte Carlo estimator is highly dependent on the nature of the integral being estimated.

4.3.7 Closed Rules

Sequences generated using closed rules are far more rigid than those generated using open rules. Closed rules aim to minimize the discrepancy associated with a finite sequence, whereas open rules produce sequences that minimize discrepancy asymptotically. Thus, in general, we no longer have the convenient fact that a sequence of size N is a subsequence of a sequence of size $N+1$. This has serious repercussions when one considers practical applications of sequences generated using closed rules. The fact that one has to regenerate the sequence each time we wish to obtain a larger sample size severely hampers computational efficiency, however if the construction algorithm is fairly efficient, then these sequences can be highly advantageous. Here we will only consider one sequence, called Hammersley sets.

4.3.8 Hammersley Sets

A d-dimensional Hammersley set of size N, for any $N \geq 1$, denoted by S_N, is constructed from a $(d-1)$-dimensional Halton sequence. To illustrate this, suppose that S represents a $(d-1)$-dimensional Halton sequence of infinite length genereated from the bases $b_1, b_2, \ldots, b_{d-1}$. Then, for each $N \geq 1$, we have the Hammersley set defined by

$$S_N = \left\{ \left(\frac{j}{N}, \varphi_{b_1}(j), \ldots, \varphi_{b_{d-1}}(j) \right) : 0 \leq j \leq N-1 \right\} .$$

If $b_1, b_2, \ldots, b_{d-1}$ are the first $d-1$ primes, then we have the following upper bound for the star discrepancy of the Hammersley set S_N:

$$D_N^*(S_N) \leq C_{d-1}(b_1, \ldots, b_{d-1}) \frac{(\log N)^{d-1}}{N} + O\left(\frac{(\log N)^{d-2}}{N} \right) ,$$

where $C_{d-1}(b_1, \ldots, b_{d-1})$ depends on $(d-1)$ but is independent of d and N. This fact, together with the reduced powers of $\log N$, in the above expression, confirms that this closed rule analogue of traditional Halton sequences have better convergence properties. The above result is taken from [17]. One disadvantage is that these sequences are also plagued by poor uniformity characteristics in high dimensions, which is again a direct consequence of the poor behavior of the van der Corput generator with large base b.

4.4 Application to the Pricing Problem

Here we consider the application of the quasi-Monte Carlo technique to the problem of pricing Himalaya options. Recall that quasi-Monte Carlo integration is based on deterministic sequences whereas crude Monte Carlo integration is based on pseudo-random sequences. This is the central difference between the two methods, and is the only difference between the applications of the two techniques to the pricing problem.

A d-dimensional low discrepancy sequence aims to fill the unit cube $[0,1)^d$ in an optimally regular fashion. Consider a d-dimensional low discrepancy sequence, if we apply the inverse cumulative normal function to each of the points constituting this sequence then we obtain a quasi-random sample from the $N_d(\mathbf{0}, I)$ distribution. These normal random numbers may then be used to generate a d-dimensional standard Brownian motion.

We now consider the pricing of the three variants of the Himalaya option, written on n stocks, assuming that all of the required pricing parameters have been specified. In view of the above discussion, instead of deriving an entirely new algorithm which produces a quasi-Monte Carlo estimate based on a sample of size N, we simply modify the first step of algorithm 3.1 in the following manner:

1. *Generate an n-dimensional low discrepancy sequence of size N. Apply the inverse cumulative normal function to each of the elements of this sequence, to obtain a quasi-random sample of size N from the $N_n(\mathbf{0}, I)$ distribution. Denote this sample by the $n \times N$ matrix Z_Q. Set $X = CZ_Q$, thereby transforming the sample, such that each column of X represents a random vector from the $N_n(\mathbf{0}, P)$ distribution.*

The estimate obtained from this modified algorithm is the quasi-Monte Carlo estimate of the value of the Himalaya option based on a sample of size N.

4.5 Numerical Results

Here we apply the modified algorithm, stated above, to obtain quasi-Monte Carlo price estimates for both the GF2 and DLF1 Himalaya options, under the scenarios presented in §3.2.2. As before, we plot the quasi-Monte Carlo price estimates for each low discrepancy sequence separately together with the corresponding crude Monte Carlo price estimates. We obtain the following results:

Scenario One - GF2 Himalaya option

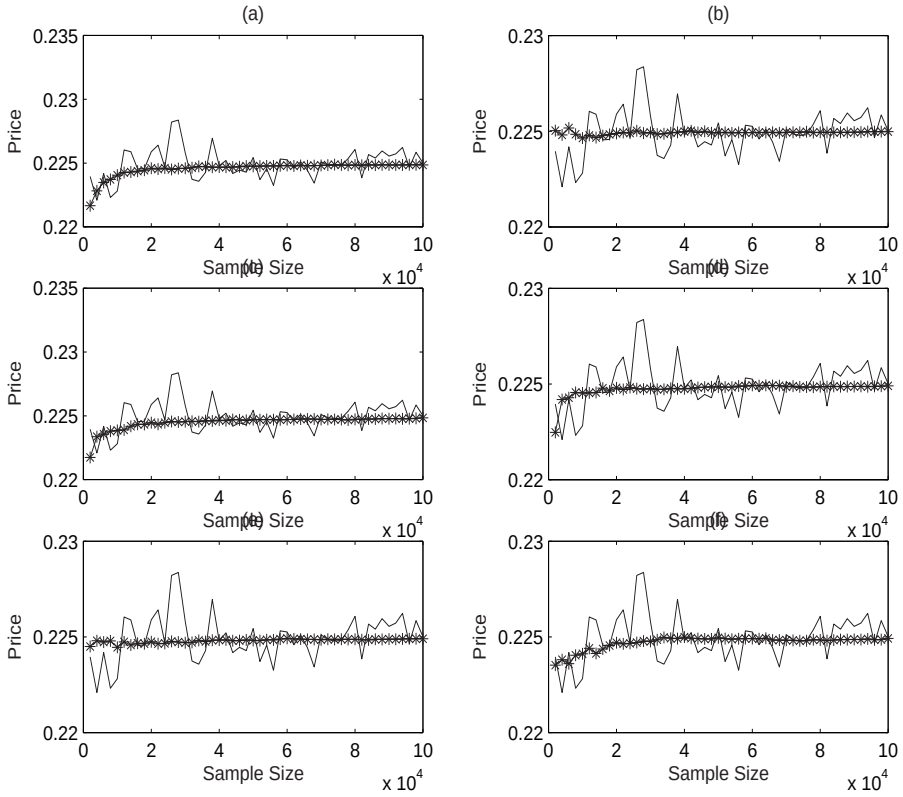

Fig. 4.1: *(a) Crude Monte Carlo price estimates vs. Halton quasi-Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Leaped Halton Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Faure quasi-Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Generalized Faure quasi-Monte Carlo price estimates; (e) Crude Monte Carlo price estimates vs. Sobol' quasi-Monte Carlo price estimates; (f) Crude Monte Carlo price estimates vs. Niederreiter quasi-Monte Carlo price estimates.*

Scenario Two - GF2 Himalaya option

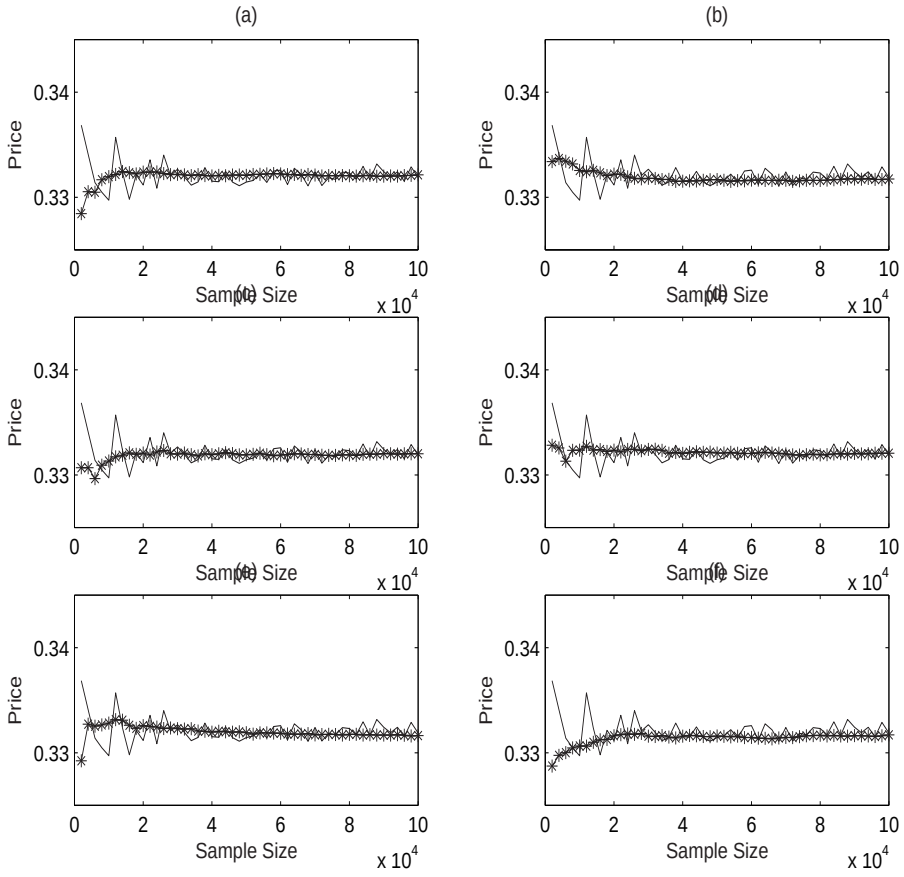

Fig. 4.2: *(a) Crude Monte Carlo price estimates vs. Halton quasi-Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Leaped Halton Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Faure quasi-Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Generalized Faure quasi-Monte Carlo price estimates; (e) Crude Monte Carlo price estimates vs. Sobol' quasi-Monte Carlo price estimates; (f) Crude Monte Carlo price estimates vs. Niederreiter quasi-Monte Carlo price estimates.*

Scenario Three - GF2 Himalaya option

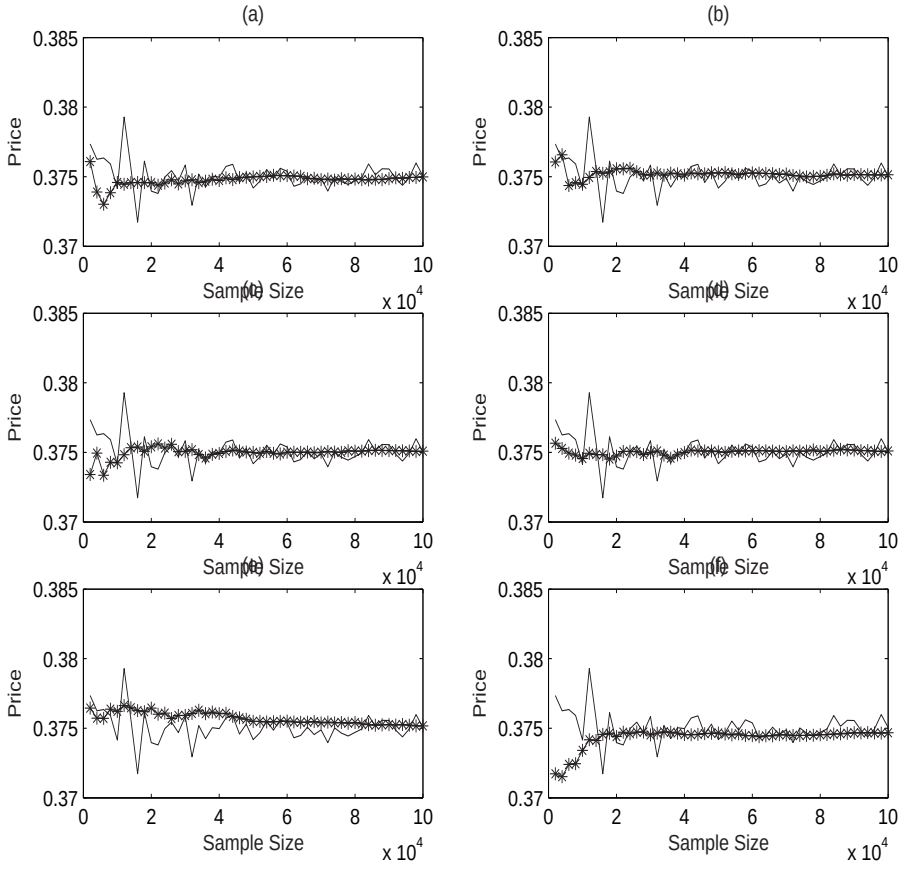

Fig. 4.3: *(a) Crude Monte Carlo price estimates vs. Halton quasi-Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Leaped Halton Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Faure quasi-Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Generalized Faure quasi-Monte Carlo price estimates; (e) Crude Monte Carlo price estimates vs. Sobol' quasi-Monte Carlo price estimates; (f) Crude Monte Carlo price estimates vs. Niederreiter quasi-Monte Carlo price estimates.*

Scenario Four - GF2 Himalaya option

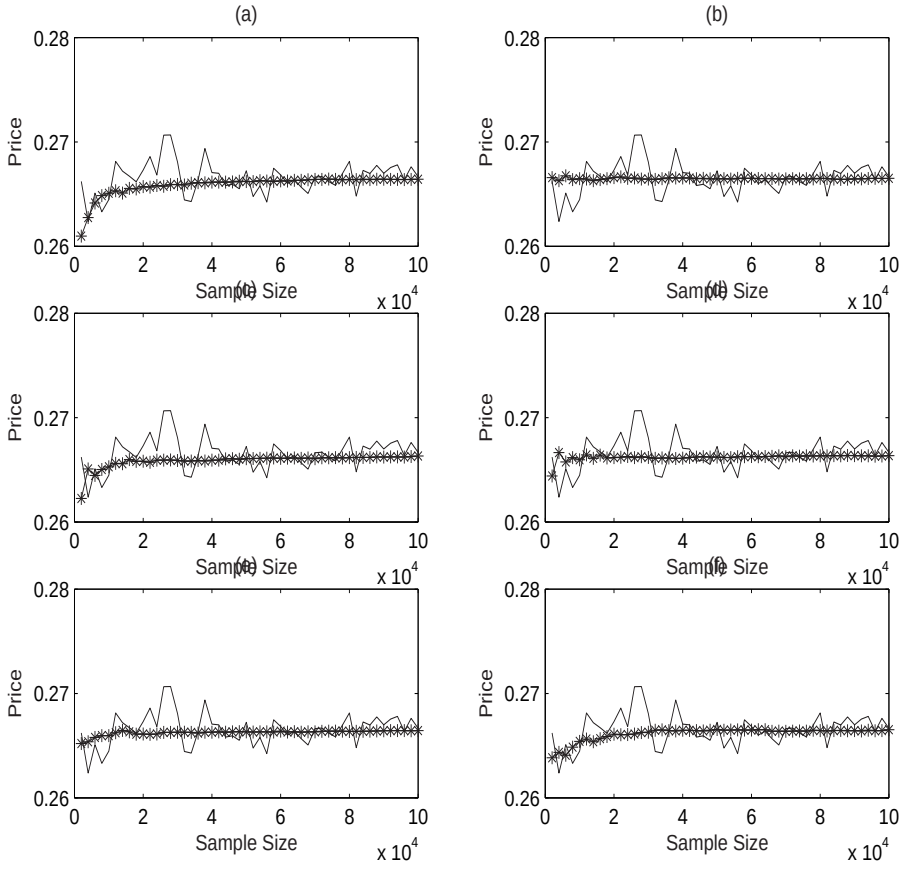

Fig. 4.4: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Scenario One - DLF1 Himalaya option

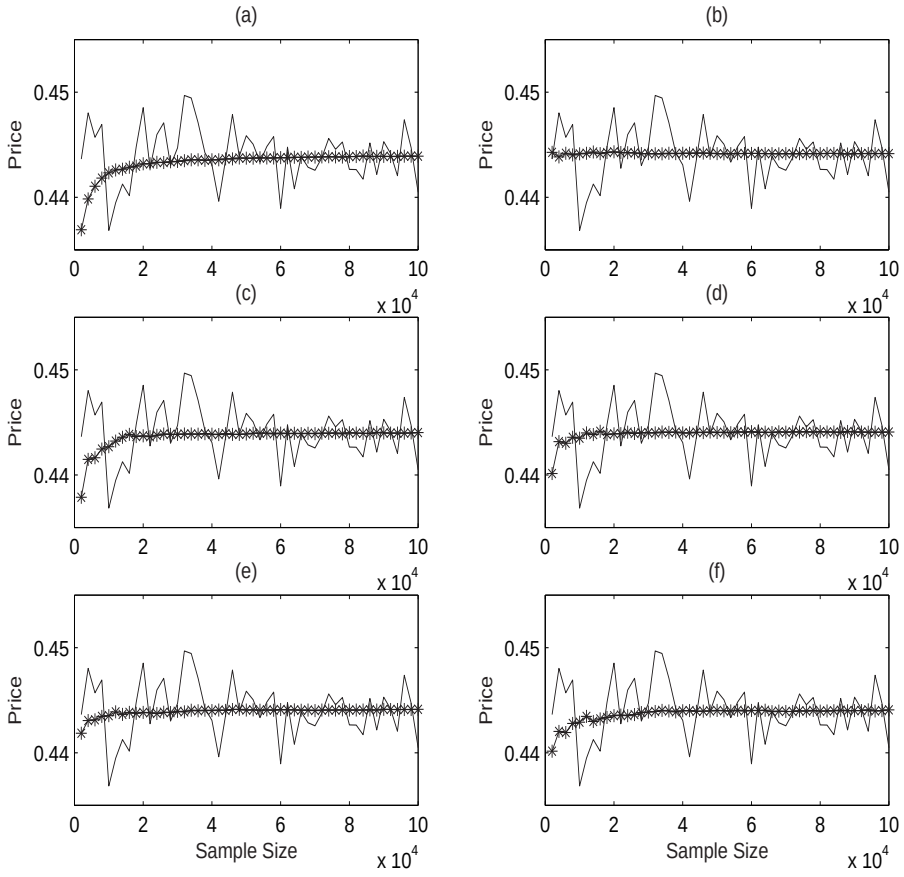

Fig. 4.5: *(a) Crude Monte Carlo price estimates vs. Halton quasi-Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Leaped Halton Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Faure quasi-Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Generalized Faure quasi-Monte Carlo price estimates; (e) Crude Monte Carlo price estimates vs. Sobol' quasi-Monte Carlo price estimates; (f) Crude Monte Carlo price estimates vs. Niederreiter quasi-Monte Carlo price estimates.*

Scenario Two - DLF1 Himalaya option

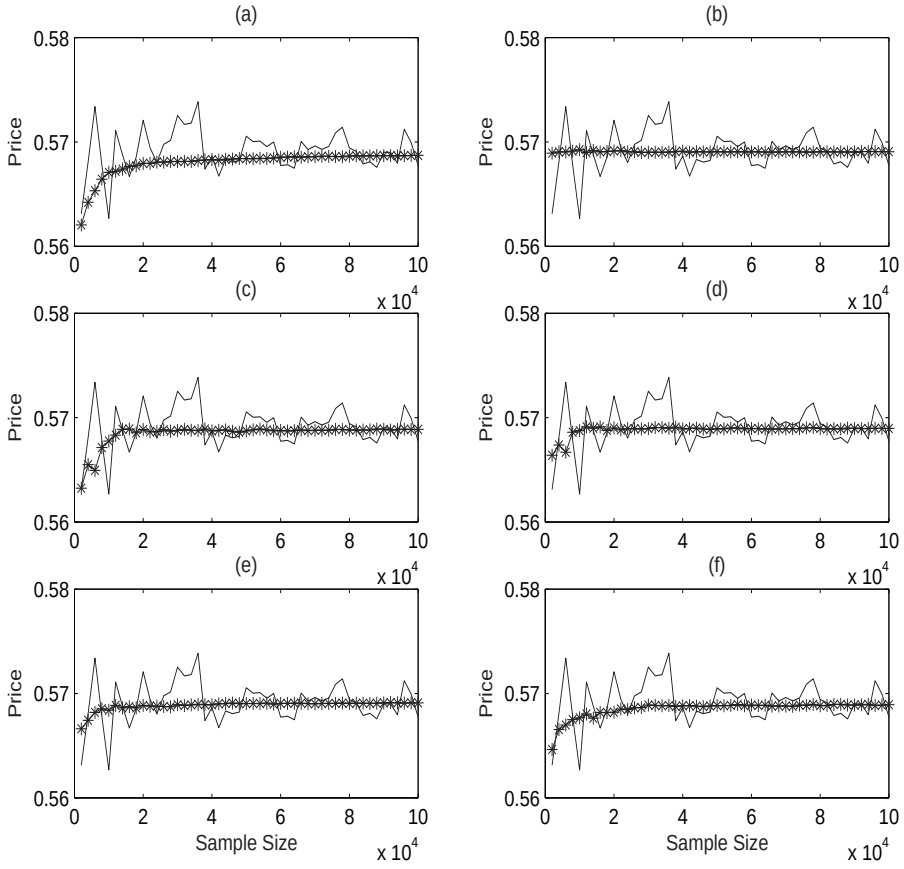

Fig. 4.6: *(a) Crude Monte Carlo price estimates vs. Halton quasi-Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Leaped Halton Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Faure quasi-Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Generalized Faure quasi-Monte Carlo price estimates; (e) Crude Monte Carlo price estimates vs. Sobol' quasi-Monte Carlo price estimates; (f) Crude Monte Carlo price estimates vs. Niederreiter quasi-Monte Carlo price estimates.*

Scenario Three - DLF1 Himalaya option

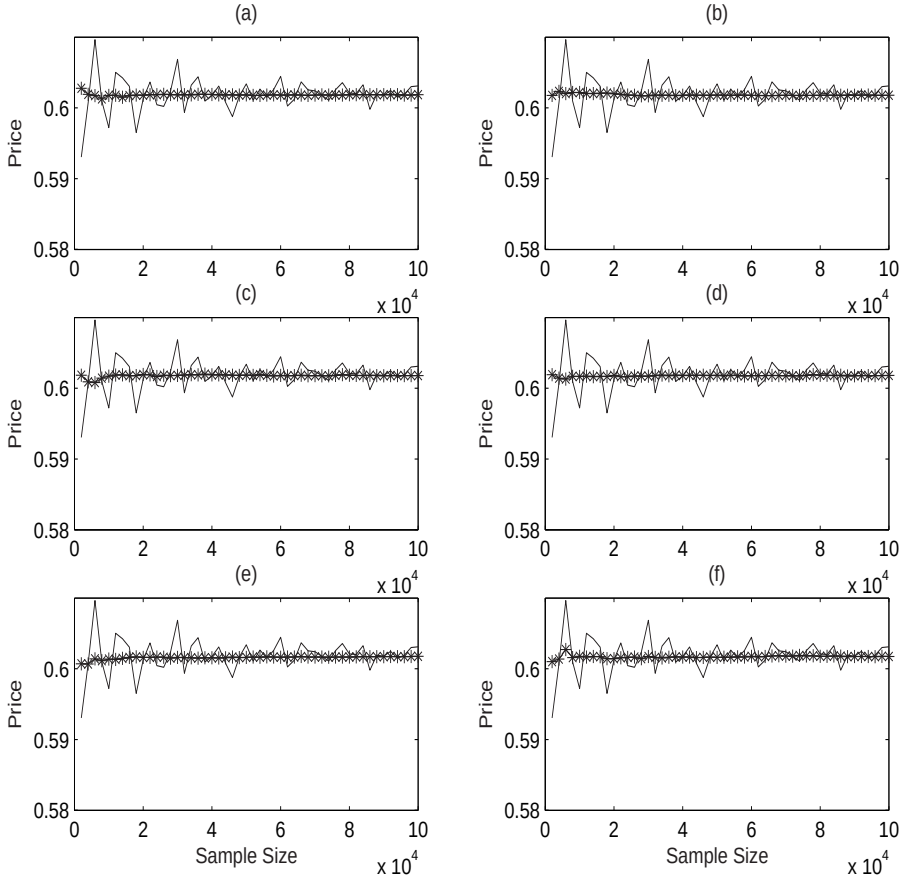

Fig. 4.7: *(a) Crude Monte Carlo price estimates vs. Halton quasi-Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Leaped Halton Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Faure quasi-Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Generalized Faure quasi-Monte Carlo price estimates; (e) Crude Monte Carlo price estimates vs. Sobol' quasi-Monte Carlo price estimates; (f) Crude Monte Carlo price estimates vs. Niederreiter quasi-Monte Carlo price estimates.*

Scenario Four - DLF1 Himalaya option

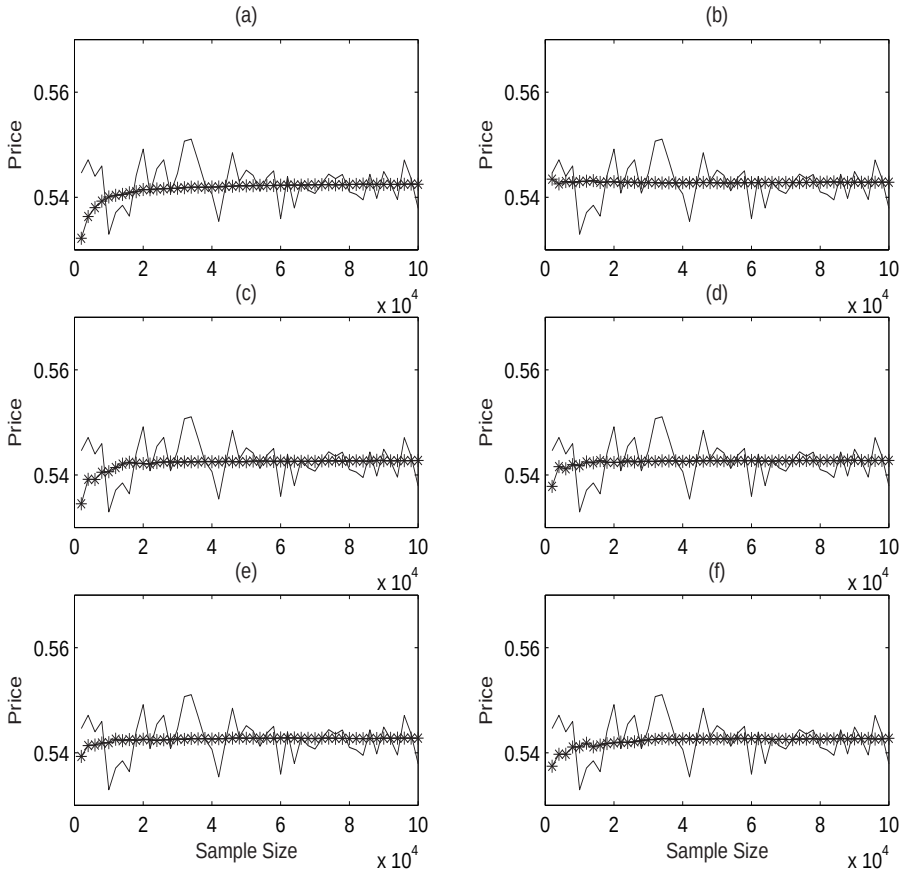

Fig. 4.8: *(a) Crude Monte Carlo price estimates vs. Antithetic Monte Carlo price estimates; (b) Crude Monte Carlo price estimates vs. Control variate Monte Carlo price estimates; (c) Crude Monte Carlo price estimates vs. Moment Matching Monte Carlo price estimates; (d) Crude Monte Carlo price estimates vs. Latin Hypercube sample price estimates.*

Chapter 5

An Algorithm for Valuing DLF Himalaya Options

5.1 Introduction

In this chapter we consider the valuation of the deterministic locally-floored (DLF) Himalaya option, written on n assets. We recall that the deterministic himalayan structure requires the buyer of the option to specify the sequence of index sets, $\{I_j\}_{j=1}^n$, at initiation of the contract. Also, using the notation from §1.2, we recall that the terminal payoffs of the deterministic locally-floored structures are given by

$$H^1_{\mathrm{DLF}}(T;0,T,A) = A\sum_{j=1}^{n} \max\left\{(P(M(I_j),j)-1)\ ,\ 0\right\} C(j,n) \tag{5.1}$$

$$H^2_{\mathrm{DLF}}(T;0,T,A) = A\sum_{j=1}^{n} \max\left\{(p(m(I_j),j)-1)\ ,\ 0\right\} C(j,n)\ . \tag{5.2}$$

We refer to the DLF Himalaya option with performance type 1 as the DLF1 Himalaya option, and analogously we refer to the option with performance type 2 as the DLF2 Himalaya option. Upon closer inspection, one realizes that these options are merely finite sums of call options on the maximum of the returns on several assets. Therefore, with minor modification to the results of Stulz, [23], and Johnson, [14], we can derive closed-form solutions for the above deterministic locally-floored himalayan structures. However, these closed-form solutions require the ability to evaluate n-variate cumulative normal probabilities. Although approximation algorithms do exist for the evaluation of univariate, bivariate and trivariate cumulative normal probabilities, one has to resort to numerical integration for n-variate cumulative normal probabilities when $n > 3$.

In order to avoid the estimation of multivariate cumulative normal probabilities via numerical integration, we consider an approximation algorithm for valuing call

options on the maximum of several assets. The algorithm that we consider was presented by Boyle and Tse, in [3]. In what follows, we modify the approximation algorithm to apply to the problem of valuing call options on the maximum of the returns on several assets and thereby obtain an approximation algorithm for valuing the DLF Himalaya options.

5.2 Call on the Maximum Return of n Assets

In order to attain an approximation algorithm for the valuation of DLF1 Himalaya options, we first consider the problem of deriving an appropriate algorithm for valuing a European call option on the maximum of the returns on n assets.

Consider an European call option on the maximum of the returns on n stocks, with initiation at time 0 and expiry at time T. Let I denote an index set which contains the indices of the n stocks underlying the option. Without any loss of generality, we assume that $I = \{1, 2, \ldots, n\}$, in this case. Further, let $V(t; 0, T, A, I)$ denote the value of the option at some time $t \in [0, T]$, where A represents the nominal value. Hence, using the notation developed in §1.2 and §2.1, we can write the terminal payoff of this option as

$$\begin{aligned} V(T; 0, T, A, I) &= A \max\{P(M(I), T) - 1\,,\, 0\} \\ &= A \max\left\{ \max_{i \in I} \left\{ \frac{S(i,T)}{S(i,0)} \right\} - 1\,,\, 0 \right\}\,. \end{aligned} \tag{5.3}$$

Now, under the assumptions given in §2.1 together with the principle of risk-neutral valuation, the value of the option at some time $t \in [0, T]$ is given by

$$\begin{aligned} V(t; 0, T, A, I) &= \exp[-r(T-t)]\, \mathbb{E}_t^{\mathbb{Q}} \left[A \max\left\{ \max_{i \in I} \left\{ \frac{S(i,T)}{S(i,0)} \right\} - 1\,,\, 0 \right\} \right] \\ &= \exp[-r(T-t)]\, \mathbb{E}_t^{\mathbb{Q}} \left[A \max\left\{ \max_{i \in I} \{ \exp[Y_i] \} - 1\,,\, 0 \right\} \right]\,, \end{aligned} \tag{5.4}$$

where we define the family of random variables $\{Y_i\}_{i \in I}$, such that each Y_i is normally distributed with mean

$$\log\left(\frac{S(i,t)}{S(i,0)} \right) + (r - q_i - \frac{1}{2}\sigma_i^2)(T-t)\,, \tag{5.5}$$

and variance

$$\sigma_i^2 (T-t)\,. \tag{5.6}$$

Also, the correlation between the random variables Y_i and Y_j is equal to ρ_{ij}, i.e. the correlation between the instantaneous returns of stock i and stock j.

Now that we have represented the value of the option in the form given by equation (5.4), we are in a position to apply the method of Boyle and Tse to approximate the value of the option.

5.3 Application of Boyle and Tse's Algorithm

In this section we will describe the algorithm of Boyle and Tse applied to the approximation of

$$V(t;0,T,A,I) = \exp[-r(T-t)]\,\mathbb{E}_t^{\mathbb{Q}}\left[A\max\left\{\max_{i\in I}\{\exp[Y_i]\}-1\,,\,0\right\}\right]\,.$$

The method of Boyle and Tse examines the problem of pricing options on the maximum of several assets in terms of order statistics. As a result, the most vital component of the algorithm is the approximation of the first four moments of the maximum of a set of random variables that have a multivariate normal distribution. This approximation is achieved using an algorithm derived by Clark, [6], and the description of this algorithm shall be the starting point of our analysis.

Clark's Algorithm

Suppose that $\mathbf{Y} = (Y_1, Y_2, Y_3)' \sim N_3(\underline{\mu}, \Sigma)$, i.e. $\mathbf{Y}$ has a trivariate normal distribution, where

$$\underline{\mu} = (\mu_1, \mu_2, \mu_3)'\,,$$

and

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{pmatrix}\,.$$

Note that, $\sigma_{ij} = \rho_{ij}\,\sigma_i\,\sigma_j$ for all $i \neq j$, where ρ_{ij} denotes the correlation coefficient between Y_i and Y_j.
Let $X = \max\{Y_1, Y_2\}$, then Clark derived exact expressions for the first four moments of X about zero which are given by

$$\begin{aligned}
\mathbb{E}[X] &= \mu_1\Phi(h) + \mu_2\Phi(-h) + \sigma\phi(h)\,,\\
\mathbb{E}[X^2] &= \left(\mu_1^2+\sigma_1^2\right)\Phi(h) + \left(\mu_2^2+\sigma_2^2\right)\Phi(-h) + (\mu_1+\mu_2)\sigma\phi(h)\,,\\
\mathbb{E}[X^3] &= \left(\mu_1^3+3\mu_1\sigma_1^2\right)\Phi(h) + \left(\mu_2^3+3\mu_2\sigma_2^2\right)\Phi(-h)\\
&\quad + [\,(\mu_1^2+\mu_1\mu_2+\mu_2^2)\sigma + \sigma^{-1}(2\sigma_1^4+\sigma_1^2\sigma_2^2\\
&\quad + 2\sigma_2^4 - 2\sigma_1^3\sigma_2\rho_{12} - 2\sigma_1\sigma_2^3\rho_{12} - \sigma_1^2\sigma_2^2\rho_{12}^2)\,]\,\phi(h)\,,
\end{aligned}$$

$$\begin{aligned}\mathbb{E}[X^4] = &\left(\mu_1^4 + 6\mu_1^2\sigma_1^2 + 3\sigma_1^4\right)\Phi(h) + \left(\mu_2^4 + 6\mu_2^2\sigma_2^2 + 3\sigma_2^4\right)\Phi(-h) \\ &[\,\left(\mu_1^3 + \mu_1^2\mu_2 + \mu_1\mu_2^2 + \mu_2^3\right)\sigma - 3h\left(\sigma_1^4 - \sigma_2^4\right) \\ &+ 4\mu_1\sigma_1^3\left(3\sigma^{-1}(\sigma_1 - \rho_{12}\sigma_2) - (\sigma_1 - \rho_{12}\sigma_2)^3\sigma^{-3}\right) \\ &+ 4\mu_2\sigma_2^3\left(3\sigma^{-1}(\sigma_2 - \rho_{12}\sigma_1) - (\sigma_2 - \rho\sigma_{12})^3\sigma^{-3}\right)\,]\,\phi(h)\ ,\end{aligned}$$

where

- $\sigma^2 = \sigma_1^2 - 2\sigma_1\sigma_2\rho_{12} + \sigma_2^2$,
- $h = \sigma^{-1}(\mu_1 - \mu_2)$,
- $\Phi(\cdot)$ denotes the univariate cumulative standard normal distribution function and
- $\phi(\cdot)$ denotes the univariate standard normal density function.

Also, the correlation coefficient between X and Y_3 is given by

$$\rho_{X3} = \frac{(\,\sigma_1\rho_{13}\Phi(h) + \sigma_2\rho_{23}\Phi(-h)\,)}{\sqrt{\mathbb{V}\mathrm{ar}[X]}}\ .$$

These exact results derived by Clark can now be utilized to approximate the first four moments of the maximum of a finite set of jointly distributed normal random variables. To describe this approximation procedure, we suppose that $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)' \sim N_n(\underline{\mu}, \Sigma)$, then we apply Clark's algorithm as follows.

Algorithm 5.1. *Set $X = Y_1$. For $i = 2, 3, \ldots, n$,*

1. *Assume that $X, Y_i, Y_{i+1}, \ldots, Y_n$ has a multivariate normal distribution, and set $X = \max\{X, Y_i\}$.*
2. *If $i < n$, then:*

   (a) Apply Clark's algorithm to obtain the mean and the variance of X.
   (b) Calculate the correlation coefficients of X with the remaining $(n - i)$ random variables.

   elseif $i = n$, then:

   (a) Apply Clark's algorithm to obtain the first four moments of X.

The assumption in step one is not accurate, however it is stated in [3] that previous numerical research has concluded that the method is fairly accurate over a broad

range of parameterizations. Now that we have this approximation technique at our disposal, we return to the analysis of equation (5.4).

Note that, we can rewrite equation (5.4) into the following form

$$\begin{aligned} V(t;0,T,A,I) &= A\exp[-r(T-t)]\,\mathbb{E}_t^{\mathbb{Q}}\left[-1+\max\left\{\max_{i\in I}\{\exp[Y_i]\}\ ,\ 1\right\}\right] \\ &= A\exp[-r(T-t)]\,\mathbb{E}_t^{\mathbb{Q}}\left[-1+\max\left\{\exp[M],\,1\right\}\right] \\ &= -A\exp[-r(T-t)]+A\exp[-r(T-t)]\,\mathbb{E}_t^{\mathbb{Q}}\left[\max\left\{\exp[M],\,1\right\}\right]\ , \end{aligned}$$

where $M=\max_{i\in I}\{Y_i\}$. Therefore, we concentrate on the approximation of the expectation in the final expression above. Following the method of Boyle an Tse, we calculate the mean and variance of the random variable M, using algorithm 5.1., and we denote these by μ_M and σ_M^2 respectively. Next, we use the standardization transformation

$$Z=\frac{M-\mu_M}{\sigma_M}\ ,$$

and rewrite the expectation under consideration as

$$\begin{aligned} \mathbb{E}_t^{\mathbb{Q}}\left[\max\left\{\exp[M],\,1\right\}\right] &= \mathbb{E}_t^{\mathbb{Q}}\left[\max\left\{\exp[\mu_M+\sigma_M Z],\,1\right\}\right] \\ &= \mathbb{E}_t^{\mathbb{Q}}\left[\exp[\mu_M+\sigma_M Z^*]\right]\ , \end{aligned} \tag{5.7}$$

where $Z^*=Z\,\mathbb{I}\{Z\geq K\}+K\,\mathbb{I}\{Z<K\}$, and $K=-\frac{\mu_M}{\sigma_M}$.

Now, equation (5.7) may be rewritten as

$$\mathbb{E}_t^{\mathbb{Q}}\left[\exp[\mu_M+\sigma_M Z^*]\right]=\exp[\mu_M+\sigma_M\mu^*]\mathbb{E}_t^{\mathbb{Q}}\left[\exp[\sigma_M(Z^*-\mu^*]\right]\ ,$$

where $\mu^*=\mathbb{E}_t^{\mathbb{Q}}[Z^*]$. Even further, using a fourth order Taylor series expansion and letting $\mu_i^*=\mathbb{E}_t^{\mathbb{Q}}[(Z^*-\mu^*)^i]$ for $i=2,3,4$, we have that

$$\mathbb{E}_t^{\mathbb{Q}}\left[\exp[\mu_M+\sigma_M Z^*]\right]\approx\exp[\mu_M+\sigma_M\mu^*]\left[1+\frac{\mu_2^*\sigma_M^2}{2!}+\frac{\mu_3^*\sigma_M^3}{3!}+\frac{\mu_4^*\sigma_M^4}{4!}\right]\ .$$

Clearly then, in order to obtain the approximation we need to evaluate the moments of the distribution of Z^*. Boyle and Tse propose the following method for computing the moments of Z^*. Firstly, we approximate the density function, $f(\cdot)$, of Z using a *Gram-Charlier expansion*, which yields the following

$$f(z)=\phi(z)\left(1+\frac{1}{3!}\left[\mathbb{E}_t^{\mathbb{Q}}[Z^3](z^3-3z)\right]+\frac{1}{4!}\left[(\mathbb{E}_t^{\mathbb{Q}}[Z^4]-3)(z^4-6z^2+3)\right]\right)\ .$$

Then, as discussed by Boyle and Tse, the moments of Z^* can be evaluated analytically using the probability density function above.

We now give the analytic formulae for the moments of Z^* which are obtained using the probability density function, $f(z)$, of the random variable Z, again these results are taken from [3]. Boyle and Tse give the following analytic formulae for the calculation of the moments of Z^*:

$$\mathbb{E}_t^{\mathbb{Q}}[(Z^*)^i] = H_{i0} + \frac{1}{3!}\left[\mathbb{E}_t^{\mathbb{Q}}[Z^3](H_{i3} - 3H_{i1})\right] + \frac{1}{4!}\left[(\mathbb{E}_t^{\mathbb{Q}}[Z^4] - 3)(H_{i4} - 6H_{i2} + 3H_{i0})\right] ,$$

for $i = 1, 2, 3, 4$ where

$$H_{ij} = K^i J_j + L_{i+j}$$

for $i = 1, 2, 3, 4$; $j = 0, 1, 2, 3, 4$, while J_i and L_i are given by

$$\begin{aligned}
L_0 &= 1 - \Phi(K) , \\
L_1 &= \phi(K) , \\
L_{i+1} &= iL_{i-1} + K^i\phi(K) , \quad \text{for } i = 1, 2, \ldots, 7 , \\
J_0 &= \Phi(K) , \\
J_1 &= -L_1 , \\
J_2 &= 1 - L_2 , \\
J_3 &= -L_3 , \\
J_4 &= 3 - L_4 .
\end{aligned}$$

On a final note, the moments of the random variable Z are obtained from the moments of the random variable M, which are in turn obtained via algorithm 5.1.

Therefore an approximation to the value of the call on the maximum returns of n assets, at some time $t \in [0, T]$, is given by

$$A\exp[-r(T-t)]\left(-1 + \exp[\mu_M + \sigma_M\mu^*]\left[1 + \frac{\mu_2^*\sigma_M^2}{2!} + \frac{\mu_3^*\sigma_M^3}{3!} + \frac{\mu_4^*\sigma_M^4}{4!}\right]\right) ,$$

which we shall denote by $V_{\text{bt}}(t; 0, T, A, I)$.

5.4 Valuation of the DLF1 Himalaya Option

First we consider the valuation of the DLF1 Himalaya option, written on n assets, where $P(\cdot, \cdot)$ is the designated performance measure. The term of the contract is $[0, T]$, where $\{t_i\}_{i=1}^n$ denote the n measurement dates, such that $0 = t_0 < t_1 < \ldots < t_n = T$. The terminal payoff of this structure is given by (5.1). Consequently, by the principle of risk-neutral valuation, the value of the option at some time $t \in [t_0, t_1]$ is given by

$$H_{\text{DLF}}^1(t; 0, T, A) = A\sum_{j=1}^{n} \exp[-r(t_j - t)]\,\mathbb{E}_t^{\mathbb{Q}}\left[\max\left\{P(M(I_j), j) - 1\,,\, 0\right\}\right] . \tag{5.8}$$

Considering the results of §5.2, we may rewrite equation (5.8) as the sum of call options on the maximum returns of several assets. Thus, we have that

$$H^1_{\mathrm{DLF}}(t;0,T,A) = \sum_{j=1}^{n} V(t;t_0,t_j,A,I_j)\ . \tag{5.9}$$

Note that the last index set, I_n, contains just one element and thus $V(t;t_0,t_n,A,I_n)$ is just the value of a vanilla European call option. However, the underlying of the vanilla call is the performance of the share indexed by I_n and the strike price is one. Using the Black-Scholes pricing formula for vanilla European call options, we may write $V(t;t_0,t_n,A,I_n)$ as

$$V(t;t_0,t_n,A,I_n) = A\left[\frac{S(I_n,t)}{S(I_n,0)}\exp[-q_{I_n}(T-t)]\,\Phi(d_1) - \exp[-r(T-t)]\,\Phi(d_2)\right]\ ,$$

where

$$d_{1,2} = \frac{\log\left(\frac{S(I_n,t)}{S(I_n,0)}\right) + (r - q_{I_n} \pm \frac{1}{2}\,\sigma^2_{I_n})(T-t)}{\sigma_{I_n}\sqrt{T-t}}\ .$$

Using the approximation method developed in §5.3 together with the above result, we have the following approximation for the value of the Himalaya option:

$$H^1_{\mathrm{DLF}}(t;0,T,A) \approx \sum_{j=1}^{n-1} V_{\mathrm{bt}}(t;t_0,t_j,A,I_j) + V(t;t_0,t_n,A,I_n)\ . \tag{5.10}$$

5.5 Forward Starting Call on the Maximum Return of n Assets

An approximation algorithm for the valuation of DLF2 Himalaya options, requires an appropriate approximation for the value of a forward starting European call option on the maximum of the returns on n assets.

Consider a forward starting European call option on the maximum of the returns on n stocks, with initiation at time 0, forward start at time s and expiry at time T. Let I denote an index set which contains the indices of the n stocks underlying the option. Without any loss of generality, we assume that $I = \{1,2,\ldots,n\}$, in this case. Further, let $V^f(t;s,T,A,I)$ denote the value of the option at some time $t \in [0,T]$, where A represents the nominal value. Now, using the notation developed in §1.2 and §2.1, we can write the terminal payoff of this option as

$$\begin{aligned} V^f(T;s,T,A,I) &= A\max\{p(m(I),T) - 1\ ,\ 0\} \\ &= A\max\left\{\max_{i\in I}\left\{\frac{S(i,T)}{S(i,s)}\right\} - 1\ ,\ 0\right\}\ . \end{aligned} \tag{5.11}$$

Hence, under the assumptions presented in §2.1 together with the principle of risk-neutral valuation, the value of the option at some time $t \in [0, s]$ is given by

$$\begin{aligned}
&V^f(t; s, T, A, I) \\
&= \exp[-r(T-t)]\, \mathbb{E}_t^{\mathbb{Q}} \left[A \max \left\{ \max_{i \in I} \left\{ \frac{S(i,T)}{S(i,s)} \right\} - 1\,,\, 0 \right\} \right] \\
&= \exp[-r(s-t)]\, \mathbb{E}_t^{\mathbb{Q}} \left[\exp[-r(T-s)]\, \mathbb{E}_s^{\mathbb{Q}} \left[A \max \left\{ \max_{i \in I} \left\{ \frac{S(i,T)}{S(i,s)} \right\} - 1\,,\, 0 \right\} \right] \right] \\
&= \exp[-r(s-t)]\, V(s; s, T, A, I)\ ,
\end{aligned}$$

where $V(s; s, T, A, I)$ denotes the value at time s of a call on the maximum of the returns on n assets with initiation at time s and expiry at time T. This result follows through the application of the tower property of conditional expectations.

Therefore, we can approximate the value of the forward starting call using the method of Boyle and Tse, given in §5.3. Suppose that $t \in [0, s]$, then we have the following approximate value for the forward starting call on the maximum of the returns on n assets:

$$V^f(t; s, T, A, I) \approx \exp[-r(s-t)]\, V_{\mathrm{bt}}(s; s, T, A, I)\ . \tag{5.12}$$

This result facilitates the derivation of an approximation algorithm for the value of the DLF2 Himalaya option, which is discussed in the next section.

5.6 Valuation of the DLF2 Himalaya Option

Now, we consider the valuation of the DLF2 Himalaya option, written on n assets, where $p(\cdot, \cdot)$ is the designated performance measure. Without any loss of generality, we assume that the term and measurement dates are as before. The terminal payoff associated with this structure is given by (5.2). Then, by the principle of risk-neutral valuation, the value of the option at some time $t \in [t_0, t_1]$ is given by

$$H^2_{\mathrm{DLF}}(t; 0, T, A) = A \sum_{j=1}^{n} \exp[-r(t_j - t)]\, \mathbb{E}_t^{\mathbb{Q}} \left[\max \left\{ p(m(I_j), j) - 1\,,\, 0 \right\} \right]\ . \tag{5.13}$$

Due to the results of §5.5, we may rewrite equation (5.13) as the sum of forward starting call options on the maximum returns of several assets. Thus, we have that

$$H^2_{\mathrm{DLF}}(t; 0, T, A) = \sum_{j=1}^{n} V^f(t; t_{j-1}, t_j, A, I_j)\ . \tag{5.14}$$

First, we observe that $V^f(t; t_0, t_1, A, I_1) = V(t; t_0, t_1, A, I_1)$. Also, the last index set, I_n, contains just one element and thus $V^f(t; t_{n-1}, t_n, A, I_n)$ is just the value of

a forward starting European call option. However, the underlying of the forward starting call is the performance of the share indexed by I_n and the strike price is one. Using the principal of risk-neutral valuation, we find that the value of such an option is given by

$$\begin{aligned} V^f(t;t_{n-1},t_n,A,I_n) =& A\,\exp[-r(t_{n-1}-t)-q_{I_n}(t_n-t_{n-1})]\,\Phi(d_1^f) \\ & - A\,\exp[-r(t_n-t)]\,\Phi(d_2^f)\;, \end{aligned}$$

where

$$d_{1,2}^f = \frac{(r-q_{I_n}\pm\frac{1}{2}\,\sigma_{I_n}^2)(t_n-t_{n-1})}{\sigma_{I_n}\sqrt{t_n-t_{n-1}}}\;.$$

Finally, using the results of §5.5 together with the observations made above, we have the following approximation for the value of the DLF2 Himalaya option:

$$\begin{aligned} H_{\mathrm{DLF}}^2(t;0,T,A) \;\approx\; & V_{\mathrm{bt}}(t;t_0,t_1,A,I_1)+V^f(t;t_{n-1},t_n,A,I_n) \\ & +\sum_{j=2}^{n-1}\exp[-r(t_{j-1}-t)]V_{\mathrm{bt}}(t_{j-1};t_{j-1},t_j,A,I_j)\;. \end{aligned} \tag{5.15}$$

5.7 Numerical Results

In this section, we apply the approximation algorithm developed in §5.4 to obtain price estimates for the DLF1 Himalaya option introduced earlier. We present price estimates for the the DLF1 Himalaya option under each of the four scenarios considered previously, together with corresponding price estimates based on the Monte Carlo methodologies discussed earlier. Note that, all of the price estimates obtained using Monte Carlo techniques are based on a sample size of 100 000. We also give the percentage differences between the price estimate obtained using the approximation algorithm and each of the price estimates obtained using Monte Carlo techniques. We obtain the following results:

Method	Price Estimate	% Difference
Approximation	0.444153	-
Crude Monte Carlo	0.440469	0.84
Antithetic Variates	0.443805	0.08
Control Variates	0.444089	0.01
Moment Matching	0.443849	0.07
Latin Hypercube	0.443955	0.04
Halton	0.443904	0.06
Leaped Halton	0.444154	0.00
Faure	0.444029	0.03
Generalized Faure	0.444084	0.02
Sobol'	0.444123	0.00
Niederreiter	0.444071	0.02

Tab. 5.1: Scenario One - Price Estimates for the DLF1 Himalaya Option.

Method	Price Estimate	% Difference
Approximation	0.572470	-
Crude Monte Carlo	0.567734	0.84
Antithetic Variates	0.569429	0.53
Control Variates	0.568810	0.64
Moment Matching	0.569356	0.55
Latin Hypercube	0.568408	0.71
Halton	0.568716	0.66
Leaped Halton	0.569057	0.60
Faure	0.568872	0.63
Generalized Faure	0.568964	0.62
Sobol'	0.569089	0.59
Niederreiter	0.568917	0.62

Tab. 5.2: Scenario Two - Price Estimates for the DLF1 Himalaya Option.

Method	Price Estimate	% Difference
Approximation	0.602004	-
Crude Monte Carlo	0.603111	-0.18
Antithetic Variates	0.601724	0.05
Control Variates	0.601798	0.03
Moment Matching	0.600801	0.20
Latin Hypercube	0.601309	0.12
Halton	0.601856	0.02
Leaped Halton	0.601799	0.03
Faure	0.601780	0.04
Generalized Faure	0.601814	0.03
Sobol'	0.601759	0.04
Niederreiter	0.601739	0.04

Tab. 5.3: Scenario Three - Price Estimates for the DLF1 Himalaya Option.

Method	Price Estimate	% Difference
Approximation	0.543362	-
Crude Monte Carlo	0.537883	1.02
Antithetic Variates	0.542150	0.22
Control Variates	0.542550	0.15
Moment Matching	0.542463	0.17
Latin Hypercube	0.542000	0.25
Halton	0.542480	0.16
Leaped Halton	0.542856	0.09
Faure	0.542753	0.11
Generalized Faure	0.542749	0.11
Sobol'	0.542790	0.11
Niederreiter	0.542738	0.11

Tab. 5.4: Scenario Four - Price Estimates for the DLF1 Himalaya Option.

Chapter 6

Conclusion

We have considered the pricing of Himalaya options, in particular, the valuation of the basic- and deterministic himalayan structures together with several variants thereof. The multi-asset path-dependent nature of such options provide a new dimension to the fields of risk management and derivative pricing. Even within the simplified Black-Scholes market model, closed-form solutions to the pricing problem are rendered obsolete as such calculations require the ability to evaluate cumulative multivariate normal probabilities in high dimensions. As a result, we have had to resort to numerical techniques for estimating integrals to obtain estimates for the prices of the variants of the Himalaya option. We considered classical crude Monte Carlo integration, variance reduction techniques, quasi-Monte Carlo integration as well as an approximation algorithm for valuing DLF Himalaya options.

First we considered the crude Monte Carlo technique for valuing the Himalaya options. The simplicity of this approach is the appealing factor, however numerical results that were obtained were unsatisfactory. The quality of the crude Monte Carlo estimates are highly dependent on the himalayan structure as well as the structures of the volatility and correlation of the underlying basket of stocks. Furthermore, even when the underlying correlation and volatility structures are favorable, the convergence rate of the crude Monte Carlo estimator is poor. Consequently, we considered techniques focussed on reducing the variance inherent in the crude Monte Carlo estimator. Again, the effectiveness of these techniques are highly dependent on the nature of the underlying correlation structure. A highly positively correlated basket of stocks stabilizes the himalayan structures and the control variate techniques work exceptionally well in this instance. The control variate technique, in particular, produces outstanding results when the underlying basket of stocks are highly positively correlated.

The need for faster convergence of the Monte Carlo estimator lead us naturally to

the quasi-Monte Carlo technique for numerical integration. We considered six different low discrepancy sequences and obtained quasi-Monte Carlo estimates to the pricing problem. Unlike the crude- and variance reduced Monte Carlo estimates, the quasi-Monte Carlo estimates appear to be unaffected by the underlying correlation and volatility structures. Further, all the quasi-Monte Carlo estimators exhibit excellent convergence rates. Therefore we are inclined to believe that quasi-Monte Carlo is the most effective numerical technique for valuing Himalaya options.

Finally, we considered an approximation algorithm for valuing the DLF Himalaya options. Numerical results suggest that this technique is also highly dependent on the underlying volatility and correlation structures. However, when the volatilities of the assets underlying the option are roughly equal and the underlying correlation is either strictly positive or negative, the algorithm provides a satisfactory estimate of the option price. In all other instances, the estimated values seem to marginally over-value the DLF option. The main advantage of the approximation algorithm, though, is the ease of implementation and the efficiency of the calculation when compared to the computationally intensive Monte Carlo techniques.

Appendix A

Primitive Polynomials and Direction Numbers

As discussed earlier, the generation of Sobol' and Niederreiter sequences requires primitive polynomials, $p(x)$, over $\mathbb{Z}_2$ while Sobol' sequences also require initial direction numbers. Here we present a table containing primitive polynomials arranged in order of increasing degree as well as initial direction numbers for Sobol' sequences.

The first column (d) lists the dimensions, the second column (q) contains the degree of the polynomials and columns four to ten contain the initial direction numbers. The third column (p) contains the primitive polynomials encoded as integers. The binary expansion of these integers represent the coefficients of the primitive polynomial. For example, 111 is the binary expansion of 7, which encodes the primitive polynomial $p(x) = x^2 + x + 1$.

The primitive polynomials and direction numbers shown here are given by Jäckel, in [13]. The initial direction numbers are found using a randomization technique formulated by Jäckel. For more information on this technique refer to [13]. Note that the table provides polynomials and direction numbers for up to twenty five dimensions only, since we only consider the valuation of Himalaya options written on n assets, where $n \leq 5$.

d	q	p	m_1	m_2	m_3	m_4	m_5	m_6	m_7
1	1	3	1						
2	2	7	1	1					
3	3	11	1	3	7				
4	3	13	1	1	5				
5	4	19	1	3	1	1			
6	4	25	1	1	3	7			
7	5	37	1	3	3	9	9		
8	5	59	1	3	7	7	21		
9	5	47	1	1	5	11	27		
10	5	61	1	1	7	3	29		
11	5	55	1	3	7	13	3		
12	5	41	1	3	5	1	15		
13	6	67	1	3	1	9	23	37	
14	6	97	1	1	3	13	11	7	
15	6	91	1	1	3	5	19	33	
16	6	109	1	3	7	13	25	5	
17	6	103	1	1	1	3	13	39	
18	6	115	1	1	5	11	7	11	
19	6	131	1	3	1	7	3	23	79
20	7	193	1	3	1	15	17	63	13
21	7	137	1	3	3	3	25	17	115
22	7	145	1	3	7	9	31	29	17
23	7	143	1	1	3	15	29	15	41
24	7	241	1	3	1	9	5	21	119
25	7	157	1	1	5	5	1	27	33

Tab. A.1: Primitive polynomials and direction numbers.

Bibliography

[1] F. Black and M. Scholes, *The pricing of options and corporate liabilities*, Journal of Political Economy **81** (1973), 637–659.

[2] P.P. Boyle, M. Broadie, and P. Glasserman, *Numerical valuation of high dimensional multivariate european securities*, Journal of Economic Dynamics and Control **21** (1997), 1267–1321.

[3] P.P. Boyle and Y.K. Tse, *An algorithm for computing values of options on the maximum or minimum of several assets*, Journal of Financial and Quantitative Analysis **25** (1990), 215–227.

[4] P. Bratley and B.L. Fox, *Algorithm 659: Implementing sobol's quasirandom sequence generator*, ACM Transactions on Mathematical Software **14** (1988), 88–100.

[5] P. Bratley, B.L. Fox, and H. Niederreiter, *Implementation and tests of low-discrepancy sequences*, ACM Transactions on Modeling and Computer Simulation **2** (1992), 195–213.

[6] C. Clark, *The greatest of a finite set of random variables*, Operations Research **9** (1961), 145–162.

[7] D. Duffie, *Dynamic asset pricing theory*, 2nd ed., Princeton University Press, 1996.

[8] H. Faure, *Discrépance de suites associées à un système de numération (en dimension s)*, Acta Arith. **41** (1982), 419–427.

[9] G.S. Fishman and B.D. Huang, *Antithetic variates revisited*, Communications of the ACM **26** (1983), 964–971.

[10] P. Glasserman, *Monte carlo methods in financial engineering*, Springer-Verlag, 2003.

[11] S.L. Heston, *A closed form solution for options with stochastic volatility with applications to bond and currency options*, Review of Financial Studies **6** (1993), no. 2, 327–343.

[12] J. Hull and A. White, *The pricing of options on assets with stochastic volatilities*, Journal of Finance **42** (1987), 281–300.

[13] P. Jäckel, *Monte carlo methods in finance*, J. Wiley and Sons Ltd., 2002.

[14] H. Johnson, *Options on the maximum or the minimum of several assets*, Journal of Financial and Quantitative Analysis **22** (1987), 277–283.

[15] L. Kocis and W.J. Whiten, *Computational investigations of low-discrepancy sequences*, ACM Transactions on Mathematical Software **23** (1997), 266–294.

[16] M.D. McKay, W.J. Conover, and R.J. Beckman, *A comparison of three methods for selecting values of input variables in the analysis of output from computer code*, Technometrics **21** (1979), 239–245.

[17] H. Niederreiter, *Random number generation and quasi-monte carlo methods*, SIAM, Philadelphia (1992).

[18] M. Overhaus, *Himalaya options*, Risk **15** (2002), no. 3, 101–104.

[19] M. Overhaus, A. Ferraris, T. Knudsen, R. Millward, L. Nguyen-Ngoc, and G. Schindlmayr, *Equity derivatives : Theory and applications*, Wiley, 2002.

[20] S.H. Paskov and J.F. Traub, *Faster valuation of financial derivatives*, Journal of Portfolio Management **22** (1995), 113–120.

[21] R. Quessette, *New products, new risks*, Risk **15** (2002), no. 3, 97–100.

[22] I.H. Sloan and S. Joe, *Lattice methods for multiple integration*, Oxford University Press, 1994.

[23] R.M. Stulz, *Options on the minimum or the maximum of two risky assets*, Journal of Financial Economics **10** (1982), 161–185.

[24] S. Tezuka and T. Tokuyama, *A note on polynomial arithmetic analogue of halton sequences*, ACM Transactions on Modeling and Computer Simulation **4** (1994), 279–284.