

Lüssem, Jens; Schumacher, Jürgen

**Working Paper**

## Simulation based option pricing

SFB 373 Discussion Paper, No. 2002,44

**Provided in Cooperation with:**

Collaborative Research Center 373: Quantification and Simulation of Economic Processes,  
Humboldt University Berlin

*Suggested Citation:* Lüssem, Jens; Schumacher, Jürgen (2002) : Simulation based option pricing, SFB 373 Discussion Paper, No. 2002,44, Humboldt University of Berlin, Interdisciplinary Research Project 373: Quantification and Simulation of Economic Processes, Berlin,  
<https://nbn-resolving.de/urn:nbn:de:kobv:11-10049021>

This Version is available at:

<https://hdl.handle.net/10419/65363>

**Standard-Nutzungsbedingungen:**

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

**Terms of use:**

*Documents in EconStor may be saved and copied for your personal and scholarly purposes.*

*You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.*

*If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.*

# Simulation based Option Pricing

Jens Lüssem<sup>1</sup> and Jürgen Schumacher<sup>2</sup>

## 1 Simulation techniques for option pricing

We introduce Monte Carlo techniques and Quasi Monte Carlo techniques for option pricing. First, we give an idea how to use simulation techniques to determine option prices, then - using the developed basic methods - we give examples how to price more complex i.e. exotic options even on more than one underlying. Finally we present a short guideline how to price exotic options with the proposed techniques.

First, we take a look at a European put on one underlying stock, a pricing problem which can be solved analytically e.g. by using the Black-Scholes formula. We start with this problem not only because it has become a kind of "standard problem" but also to have the possibility to compare the results of our approximation with an analytical solution. At the same time we look at the time-complexity of the used simulation technique. Next, we show how to price path dependent options with Monte Carlo methods. Afterwards, we show how to price a stock option on several underlyings. This implies that we have to solve a multi-dimensional simulation problem.

### 1.1 Introduction to simulation techniques

The idea behind randomized algorithms is that a random sample from a population (of input variables) is representative for the whole population. As a consequence, a randomized algorithm can be interpreted as a probability distribution on a set of deterministic algorithms.

We will see that there are three main advantages to randomized algorithms: 1. Performance: For many problems, it can be shown that randomized algorithms run faster than the best known deterministic algorithm. 2. Simplicity: Randomized algorithms are easier to describe and implement than comparable deterministic algorithms. 3. Flexibility: Randomized algorithms can be easily adapted.

In general one distinguishes two types of randomized algorithms. Las Vegas algorithms are randomized algorithms that always give correct results with only the variation from one run to another being its running time. Monte Carlo algorithms are randomized algorithms that may produce an incorrect solution for which one can bound the probability of occurrence. The quality of the solution can be seen as a random variable.

Within this chapter, we focus on Monte Carlo algorithms calculating the value of the following integral

$$\int_{[0,1]^d} f(x) dx \tag{1}$$

by evaluation of  $f(x)$  for independent uniform distributed random vectors  $X_1, X_2, \dots, X_n$ ,  $X_i \in [0, 1]^d$ .

---

<sup>1</sup>Landesbank Kiel

<sup>2</sup>University of Bonn, Department of Computer Science

The arithmetic mean of the values  $f(X_i)$  can be seen as a guess for the expected value of the random variable  $f(X_i)$  and therefore can be interpreted as an approximation for the value of the integral. According to the strong law of large numbers the estimator for the expected value (the arithmetic mean of the random function values) is converging to the expected value (the value of the integral) with an increasing sample size. The probability that the absolute error of the approximation result exceeds a fixed positive value  $\epsilon$  is limited and decreases to zero with an increasing sample size if the variance of  $f$  is finite.

## 1.2 Pricing path independent European options on one underlying

For the case of a European option on one underlying we have to approximate the following integral via Monte Carlo simulation:

$$e^{r(T-t)}\mathbb{E}[C_T(S_T)|S_t] = \int_0^\infty C_T(S_T)g(S_T|S_t, r, \sigma, T-t)dS_T \quad (2)$$

$$= \int_{[0,1)} C_T\{f(x, S_t, r, \sigma, T-t)\}dx \quad (3)$$

Where

$$g(S_T|S_t, r, \sigma, T-t) = \frac{\exp\left\{-\frac{(\log S_T - (\log S_t - (r - 0.5\sigma^2)(T-t)))^2}{2\sigma^2(T-t)}\right\}}{\sqrt{2\pi\sigma^2(T-t)}S_T}$$

is the risk neutral density function of the Black Scholes model with parameters:

- $S_T$  : price of the underlying at maturity
- $S_t$  : price of the underlying at time  $t$
- $r$  : risk free interest rate
- $\sigma$  : volatility of log returns of the underlying
- $T-t$  : time to maturity

$$S_T = f(x, S_t, r, \sigma, T-t) = S_t \exp\left\{\left(r - \frac{1}{2}\sigma^2\right)(T-t) + \sigma\sqrt{T-t}F^{-1}(x)\right\}$$

transforms the uniform distributed values  $x$  in  $g(S_T|S_t, r, \sigma, T-t)$  distributed underlying values  $S_T$ .  $F^{-1}(x)$  is the inverse of the cumulative normal distribution function and  $C_T(y)$  is the payoff function of the option.

The Monte Carlo simulation calculates the value of the integral in the following way:

1.  $n$  independent random underlying values  $S_T^1 \dots S_T^n$  are generated by computing  $f(x, S_t, r, \sigma, T-t)$  for a set of uniformly distributed pseudo random numbers  $X_1, \dots, X_n$ .
2. the option payoff  $C_T(S_T^i)$  is calculated for each  $S_T^i$ .
3. the value of the integral in (3) is then approximated by the arithmetic mean of the option payoffs:

$$\bar{C} = \frac{1}{n} \sum_{i=1}^n C_T(S_T^i)$$

We will now derive an estimate of the approximation error of the arithmetic mean. We assume that  $S_T^1 \dots S_T^n$  are independent random underlying samples of the  $g(S_T|S_t, r, \sigma, T-t)$  density. Using this assumption we can conclude that  $\bar{C}$  is a random variable with expected value

$$\mathbb{E}[\bar{C}] = e^{r(T-t)} C_t(S_t)$$

Additionally we have to assume that the variance of the option payoffs  $C_T(S_T)$  is given by:

$$\text{Var}[C_T(S_T)] = \int_{[0, \infty]} C_T(S_T)^2 g(S_T|S_t, r, \sigma, T-t) dS_T - \mathbb{E}[C_T(S_T)]^2 \quad (4)$$

exists. Then we get:

$$\text{Var}[\bar{C}] = \frac{1}{n^2} \sum_{i=1}^n \text{Var}[C_T(S_T^i)] = \frac{1}{n} \text{Var}[C_T(S_T)] \quad (5)$$

because of the independence of  $S_T^1, \dots, S_T^n$ .

The expected value of the random variable  $\bar{C}$  equals the value of the integral  $e^{r(T-t)} C_t(S_t)$  and its variance converges to zero with increasing  $n$ . The probability that the approximation error is greater than a fixed positive value decreases to 0 with an increasing number  $n$ . A first estimation of the error is given by the Chebychev inequality for  $\bar{C}$ ,

$$\mathbb{P}\left(|\bar{C} - e^{r(T-t)} C_t(S_t)| \geq a\right) \leq \frac{\frac{1}{n} \text{Var}[C_T(S_T)]}{a^2}$$

The bound given by this equation is rather imprecise since we do not make any assumptions on the distribution of the random variable. Only the expected value and the variance are used in the previous equation. According to the central limit theorem the distribution of  $\bar{C}$  converges to a normal distribution for  $n \rightarrow \infty$ . It follows that the difference between the approximation and the integral,  $\bar{C} - e^{r(T-t)} C_t(S_t)$  is approximately normally distributed with mean 0 and standard deviation

$$\sigma_{\bar{C}} = \sqrt{\frac{\text{Var}[C_T(S_T)]}{n}} \quad (7)$$

for large  $n$ . According to Boyle (1977) a value of  $n > 1000$  is sufficiently large in order to use the normal distribution for error estimation purposes.

We get the following equation if we assume that  $\bar{C} - e^{r(T-t)} C_t(S_t)$  is normal distributed:

$$\mathbb{P}\left(|\bar{C} - e^{r(T-t)} C_t(S_t)| \leq \epsilon\right) = \frac{1}{\sqrt{2\pi}} \int_{-\epsilon}^{\epsilon} \exp\left\{-\frac{u^2}{2\sigma_{\bar{C}}^2}\right\} du \quad (8)$$

If we choose  $k$  as a multiple of the standard deviation  $\sigma_{\bar{C}}$  of  $\bar{C}$ , then we get:

$$\begin{aligned} \mathbb{P}\left(|\bar{C} - e^{r(T-t)} C_t(S_t)| \leq k\sigma_{\bar{C}}\right) &= \mathbb{P}\left(\frac{|\bar{C} - e^{r(T-t)} C_t(S_t)|}{\sigma_{\bar{C}}} \leq k\right) \\ &= \frac{1}{\sqrt{2\pi}} \int_{-k}^k \exp\left\{-\frac{u^2}{2}\right\} du \\ &= p \end{aligned} \quad (9)$$

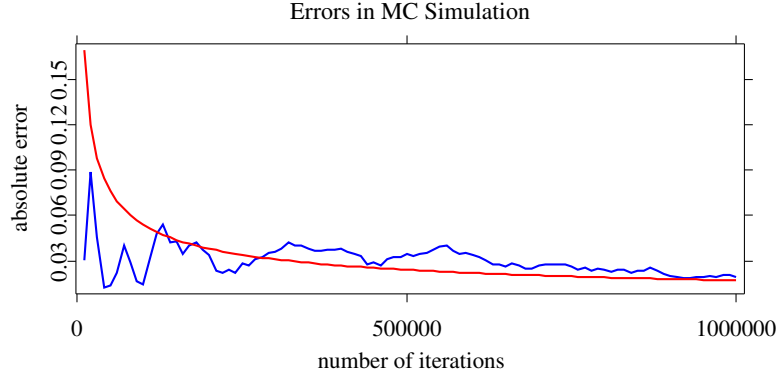


Figure 1: Absolute error of a European Call option price calculated by Monte Carlo simulations vs.  $n^{-1/2}$

Given a fixed probability level  $p$ , the error converges to zero with  $\mathcal{O}(1/\sqrt{n})$ . The error interval holds for  $k = 1, 2, 3$  with the respective probabilities  $p = 0.682, 0.955, 0.997$

The confidence intervals for a given probability level depend on the standard deviation of the payoff function  $C_T(S_T)$ :

$$\sigma_{C_T} = \sqrt{\text{Var}[C_T(S_T)]}. \quad (10)$$

In general, this standard deviation cannot be calculated with analytical methods. Therefore one calculates the empirical standard deviation  $\bar{\sigma}$  and uses it as a proxy for the error bounds:

$$\bar{\sigma} = \sqrt{\frac{1}{n-1} \sum_{k=1}^n \{C_T(S_T^i) - \bar{C}\}^2}. \quad (11)$$

Figure 1 shows the evolution of the absolute error of the price for a European call option calculated by Monte Carlo methods compared with the analytic solution. One can observe that the error tends to zero with  $\mathcal{O}(1/\sqrt{n})$ .

We would like to give some of the main properties of algorithms using Monte Carlo techniques. First from (9) it follows that the error bound tends to zero with  $\mathcal{O}(1/\sqrt{n})$  for a fixed probability level  $p$ . Second, the probability that a fixed error bound holds converges to 1 with  $\mathcal{O}(1/\sqrt{n})$ , Mavin H. Kalos (1986). Since these results hold independent of the dimension of the problem, which affects only the variance of the payoff function with respect to the Black-Scholes risk neutral density, the Monte Carlo method is especially well suited for the evaluation of option prices in multidimensional settings. Competing pricing methods e.g finite differences have exponential growing computational costs in the dimension of the problem. Another advantage of the Monte Carlo pricing method is the error estimate given by the empirical standard deviation which can be computed with a small additional effort.

The two most important drawbacks of the Monte Carlo simulation, mentioned in literature are its small convergence speed compared to other tech-

niques for options on few underlyings and the difficulties occurring for options with early exercise possibilities. For example, American options giving the investor the possibility to exercise the option at any time before and at maturity, are difficult to price. To evaluate an American option means to find an optimal exercise strategy which leads - using only basic Monte Carlo techniques - to a recursive algorithm with exponential time-complexity. But more advanced techniques using importance sampling methods show that Monte Carlo simulations can be applied to evaluate American contracts, Broadie (2000).

### 1.3 Pricing path dependent European options on one underlying

There are two categories of options. Path-independent options are options whose payoff depends only on the underlying prices at maturity. Path-dependent options are options whose payoff depends on underlying price outcomes  $S_{t_1}, \dots, S_{t_m}$  at several time points  $t_1 \leq \dots \leq t_m$  within the lifetime of the respective option.

Within the group of path-dependent options one can distinguish options with a payoff function depending on a continuously defined path variable and options with a payoff function depending on a fixed number of underlying values. The price of an option with many - usually equally spaced - exercise dates is often approximated by the price of an option with a continuously defined path variable and vice versa.

Examples for path-dependent options are barrier options, lookback options, and Asian options. The latter have a payoff function which is linked to the average value of the underlying on a specific set of dates during the life of the option. One distinguishes two basic forms of Asian options: options on the geometric mean (for which the price can be calculated with standard techniques) and options on the arithmetic mean (for which the price can not be determined using standard approaches). Asian options are frequently used in commodity markets. The volatility of the underlying prices of the commodities is usually very high so that prices for vanilla options are more expensive than for comparable Asian-style options.

### 1.4 Pricing options on multiple underlyings

In this section we show how to extend the Monte Carlo simulation technique to higher dimensions. The problem is not only that one has to deal with higher dimensional integrals, but also that one has to incorporate the underlying correlation structure between the considered securities. In our framework we need the covariance matrix of the log returns on an annual basis.

In general, a basket option is an option on several underlyings (a basket of underlyings). Basket options can be European-, American or even Asian-style options. Normally, the average of the underlying prices is taken to calculate the price of the basket option, but sometimes other functions are used.

The advantage of the usage of basket options instead of a series of one dimensional options is that the correlation between securities is taken into account. This may lead to better portfolio hedges. We will look at a basket option on five underlyings where the underlying price of the best security in the basket is taken to calculate the option price.

## 2 Quasi Monte Carlo (QMC) techniques for option pricing

### 2.1 Introduction to Quasi Monte Carlo techniques

QMC methods can be considered as an alternative to Monte Carlo simulation. Instead of (pseudo) random numbers, Quasi Monte Carlo algorithms use the elements of low discrepancy sequences to simulate underlying values.

The discrepancy of a set of points  $P \subset [0, 1]^s$  measures how evenly these points are distributed in the unit cube. The general measure of discrepancy is given by:

$$D_n(\mathcal{B}; P) := \sup_{B \in \mathcal{B}} \left| \frac{A(B; P)}{n} - \lambda_s(B) \right| \quad (12)$$

where  $A(B; P)$  is the number of points in  $P$  belonging to  $B$ ,  $\lambda_s(B)$  is the Lebesgue measure of the set  $B$ ,  $\mathcal{B}$  is a family of Lebesgue measurable subsets of  $[0, 1]^s$ , and  $n$  is the number of elements in  $P$ .

The discrepancy of a set is the largest difference between the number of points in a subset and the measure of the subset. If we define  $\mathcal{B}$  to be the family  $\mathcal{J}$  of subintervals  $\prod_{i=1}^s [0, u_i]$ , then we get a special measure, the star-discrepancy:

$$D_n^*(P) := D_n(\mathcal{J}; P) \quad (13)$$

### 2.2 Error bounds

For the star-discrepancy measure and reasonable assumption on the nature of the function that has to be integrated an upper bound on the error is given by the following theorem:

**THEOREM .1 (Koksma-Hlawka)** *If the function  $f$  is of finite variation  $V(f)$  in the sense of Hardy and Krause, then the following equation holds for all sets of points  $\{x_1, \dots, x_n\} \subset I^s = [0, 1]^s$*

$$\left| \frac{1}{n} \sum_{i=1}^n f(x_i) - \int_{I^s} f(u) du \right| \leq V(f) D_n^*(x_1, \dots, x_n) \quad (14)$$

A proof is given in Niederreiter (1992).

This means that the error is bounded from above by the product of the variation  $V(f)$ , which in our case is model and payoff dependent and the star-discrepancy of the sequence. The bound cannot be used for an automatic error estimation since the variation and the star-discrepancy cannot be computed easily. It has been shown though that sequences exist with a star-discrepancy of the order  $\mathcal{O}(n^{-1}(\ln n)^s)$ . All sequences with this asymptotic upper bound are called low-discrepancy sequences Niederreiter (1992). One particular low-discrepancy sequence is the Halton sequence.

### 2.3 Construction of the Halton sequence

We start with the construction of the one-dimensional Halton sequence within the interval  $[0, 1]$ . An element of this sequence is calculated by using the following equation:

$$x_i = \sum_{k=0}^{\infty} n_{k,i} p^{-k-1} \quad (15)$$

with  $i > 0$ ,  $p = 2$  and  $n_{k,i}$  determined by the following equation:

$$i = \sum_{k=0}^{\infty} n_{k,i} p^k; \quad 0 \leq n_{k,i} < p; \quad n_{k,i} \in \mathbb{N} \quad (16)$$

Note that with the above equation  $n_{k,i}$  is a function of  $i$  and takes values only in  $\{0; 1\}$ . To illustrate the algorithm we calculate the first three points.

$i = 1$ :  $n_{0,1} = 1$ ,  $n_{k,1} = 0$  for every  $k > 0$

$i = 2$ :  $n_{1,2} = 1$ ,  $n_{k,2} = 0$  for every  $k \neq 1$

$i = 3$ :  $n_{0,3} = n_{1,3} = 1$ ,  $n_{k,3} = 0$  for every  $k > 1$

Therefore we get the sequence  $1/2, 1/4, 3/4, 1/8, 5/8, \dots$ . The extension of this construction scheme to higher dimensions is straightforward. For every dimension  $j = 1, \dots, d$  we define  $x_i^j$  by

$$x_i^j = \sum_{k=0}^{\infty} n_{k,i}(j) p_j^{-k-1} \quad (17)$$

with  $p_j$  is the  $j$ th smallest prime number and  $n_{k,i}(j)$  is calculated as follows:

$$i = \sum_{k=0}^{\infty} n_{k,i}(j) p_j^k; \quad 0 \leq n_{k,i}(j) < p_j; \quad n_{k,i}(j) \in \mathbb{N} \quad \forall j \quad (18)$$

By using  $p_1 = 2$ ,  $p_2 = 3$  we get the following two-dimensional Halton sequence:  $(1/2; 1/3), (1/4; 2/3), \dots$ . In contrast to grid discretization schemes like  $i/n$   $i = 1, \dots, n$  low-discrepancy sequences fill the integration space in an incremental way avoiding the exponential growth of grid points of conventional schemes.

XploRe provides quantlets to generate pseudo random numbers and low discrepancy sequences. For the generation of the pseudo random numbers we use

```
erg = randomnumbers (seqnum,d,n)
generates n pseudo random vectors of dimension d
```

where `seqnum` is the number of the random generator according to Table 1, `d` is the dimension of the random vector and `n` the number of vectors generated.

The generation of low discrepancy sequences is provided by

```
erg = lowdiscrepancy (seqnum,d,n)
generates the first n low discrepancy sequence vectors of dimension
d
```



0	Park and Miller with Bays-Durham shuffle
1	L'Ecuyer with Bays-Durham shuffle
2	Knuth
3	generator from G. Marsaglia et al. Marsaglia (1993)
4	random number generator of your system
5	generator from ACM TOMS 17:98-111
6	multiply with carry gen. (Marsaglia) Marsaglia (1993)

Table 1: Random generator that can be used in XploRe

where `seqnum` is the number of the low discrepancy sequence according to Table 2.

0	Halton sequence
1	Sobol sequence
2	Faure sequence
3	Niederreiter sequence

Table 2: Low-discrepancy sequences available in XploRe, (Niederreiter, 1992) .

## 2.4 Experimental results

Figure 2 shows that two dimensional Halton points are much more equally spaced than pseudo random points. This leads to a smaller error at least for “smooth” functions.

The positive effect of using more evenly spread points for the simulation task is shown in Figure 3. The points of a low-discrepancy sequence are designed in order to fill the space evenly without any restrictions on the independence of sequence points where as the pseudo random points are designed to show no statistically significant deviation from the independence assumption. Because of the construction of the low discrepancy sequences one cannot calculate an empirical standard deviation of the estimator like for Monte Carlo methods and derive an error approximation for the estimation. One possible way out of this dilemma is the randomization of the low-discrepancy sequences using pseudo random numbers i.e. to shift the original quasi random numbers with pseudo random numbers Tuffin (1996). If  $x_1, \dots, x_n$  are scalar elements of a low-discrepancy sequence  $X$  then we can define a new low discrepancy sequence

$$W(\epsilon) = \{y_1, \dots, y_n\} \quad \text{with} \quad y_i = \begin{cases} x_i + \epsilon & x_i + \epsilon \leq 1 \\ (x_i + \epsilon) - 1 & \text{otherwise} \end{cases} \quad (19)$$

for a uniformly distributed value  $\epsilon$ . Then we can calculate an empirical standard deviation of the price estimates for different sequences  $W(\epsilon)$  for independent values  $\epsilon$  which can be used as a measure for the error. Experiments with payoff functions for European options show that this randomization technique reduces the convergence rate proportionally.

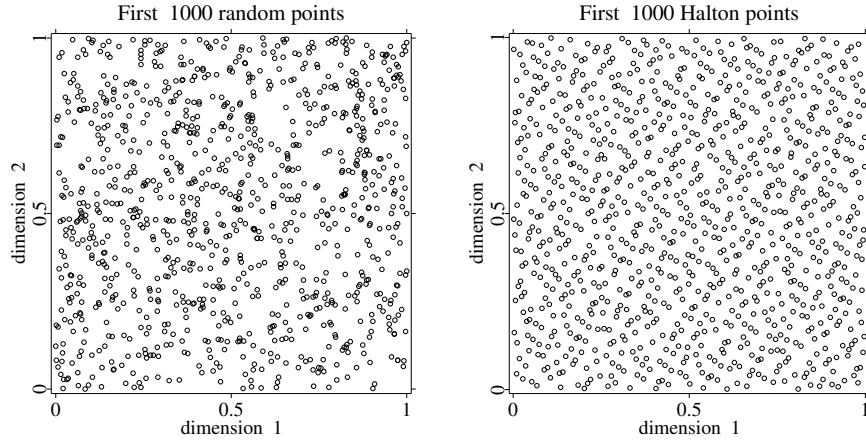


Figure 2: 1000 two-dimensional pseudo random points vs. 1000 Halton points  
 XFGS0PRandomNumbers.xpl, XFGS0PLowDiscrepancy.xpl

The advantage of the Quasi Monte Carlo simulation compared to the Monte Carlo simulation vanishes if the dimension increases. Especially the components with a high index number of the first elements in low-discrepancy sequences are not evenly distributed Niederreiter (1992). Figure 4 shows that the 49th and 50th component of the first 1000 Halton points are not evenly distributed. But the result for the first 10000 points of the sequence shows that the points become more evenly spread if the number of points increases.

However by using the Brownian Bridge path construction method we can limit the effect of the high dimensional components on a simulated underlying path and the corresponding path variable for the most common path dependent options, Morokoff (1996). This method start with an empty path with known start value and calculates at each step the underlying value for a time point with maximum time distance to all other time points with known underlying value until the whole path is computed. Experimental results show that we can still get a faster convergence of the QMC simulation for options up to 50 time points if we apply this path construction method.

### 3 Pricing options with simulation techniques - a guideline

In this section we would like to give a short guideline how to price exotic options with Monte Carlo and Quasi Monte Carlo simulation techniques within the framework described above. Furthermore we give some indications about the limits of these techniques.

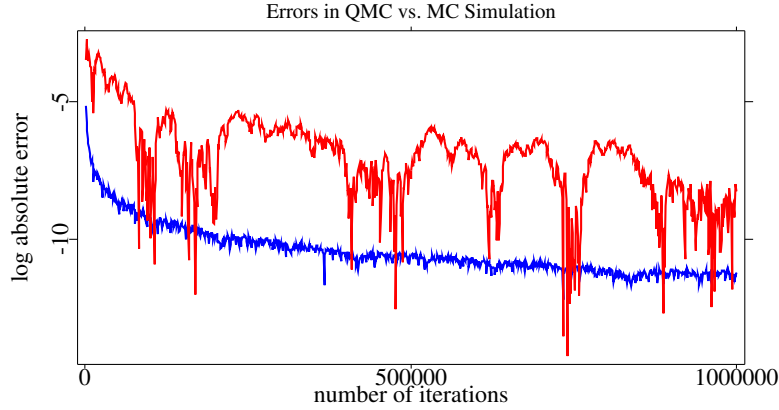


Figure 3: Absolute error of a random sequence and the Halton sequence for a put option

### 3.1 Construction of the payoff function

As a first step we have to define the payoff function corresponding to our option product. Within the methods defined in the quantlib `finance` we have to consider three different cases.

#### One underlying + path independent

In this case the payoff function is called by the pricing routine with the simulated underlying value at maturity as the single argument. It calculates the corresponding payoff and returns this value. We have defined the payoff function for a put option with strike price 100 as an example for a one dimensional payoff function.

#### Several underlying + path independent

For options whose payoff depends on the underlying values of several assets at maturity, we have to define a payoff function on the vector of the underlying values at maturity. An example for such an option is an exchange option that permits to swap a defined share with the best performing share in a basket. Its payoff function is given by:

$$C_T((S_T^1, \dots, S_T^5)) = \max\{0, \alpha_i(S_T^i - K^i) + 55 - S_T^3 | i = 1, \dots, 5\}$$

#### One underlying + path dependent

The third category of option types that are captured are path dependent options on one underlying. The payoff function of these options depends on the underlying values at several fixed time points during the lifetime of the option. Payoff functions for these contracts are called with a vector of underlying values whose  $i$ th element is the underlying value at the time  $t_i$  which has to be specified in the model.

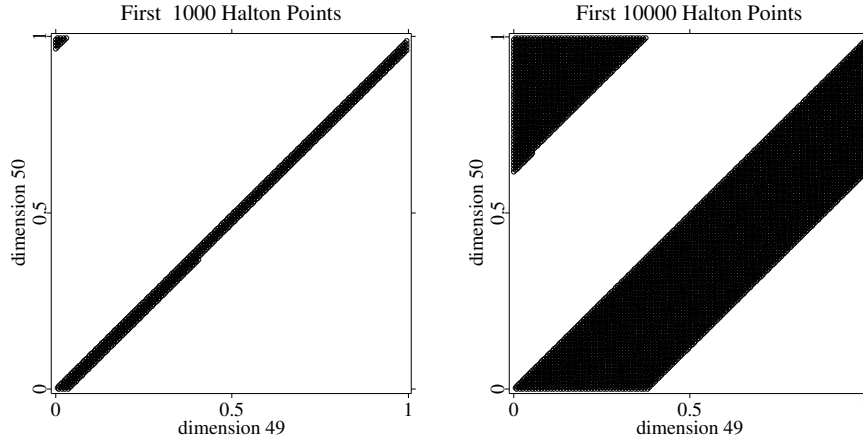


Figure 4: The first 1000 and 10000 Halton points of dimension 49 and 50  
[XFGS0PLowDiscrepancy.xpl](#)

### 3.2 Integration of the payoff function in the simulation framework

After defining the payoff function in XploRe we can start to calculate a price estimate with the help of the appropriate simulation routine. In the one dimensional case we just have to call

```
erg = BlackScholesPathIndependent1D (s0,r,vola,dt,opt,
                                     itr,gen)
    MC estimation of the option price for a path independent option.

erg = BlackScholesPathIndependent1DQMC (s0,r,vola,dt,opt,
                                       itr,gen)
    QMC estimation of the option price for a path independent
    option.
```

to get a price estimate and for the Monte Carlo case an empirical standard deviation with respect to a start price of  $s_0$ , a continuous risk free interest rate of  $r$ , a volatility  $vola$ , a time to maturity of  $dt$  years, the payoff function  $opt$ , sample size  $itr$  and the random/low-discrepancy generator with number  $gen$ . Table 1 shows the random number generators and table 2 the low-discrepancy generators that can be used. An application of these routines for a Put option can be found in [XFGS0P1DPut.xpl](#).

Pricing path-dependent options is only slightly more complicated. Here we have to define the vector of time points for which underlying prices have to be generated. This vector replaces the time to maturity used to price path independent options. Then we can apply one of the following methods to compute a price estimate for the path dependent option

```

erg = BlackScholesPathDependent1D (s0,r,vola,times,opt,
                                     itr,gen)
    MC estimation of the option price for path-dependent options.

erg = BlackScholesPathDependent1DQMC (s0,r,vola,times,opt,
                                       itr,gen)
    QMC estimation of the option price for path-dependent options,
    with:

```

with respect to the start price  $s_0$ , the continuous risk free interest rate  $r$ , the volatility  $vola$ , the time scheme  $times$ , the payoff function  $opt$ , sample size  $itr$  and the random/low-discrepancy generator with number  $gen$ , as given in Tables 1 and 2. Using the above quantlets, we calculate the price of an Asian call option in [XFGSOP1DAsian.xpl](#).

In the case of multidimensional options we have to define a start price vector and a covariance matrix instead of a single underlying price and volatility value. Then we can call one of the multi-dimensional simulation routines:

```

erg = BlackScholesPathIndependentMD (s0,r,vola,dt,opt
                                     ,itr,gen)
    MC estimation of the option price in the multidimensional Black
    Scholes model

erg = BlackScholesPathIndependentMDQMC (s0,r,vola,dt,opt
                                       ,itr,gen)
    QMC estimation of the option price in the multidimensional
    Black Scholes model

```

with respect to the  $m$  dimensional start price vector  $s_0$ , the continuous risk free interest rate  $r$ , the  $m \times m$  covariance matrix  $vola$ , the time to maturity  $dt$ , the payoff function  $opt$ , the number of iterations  $itr$  and the generator number  $gen$  according to the generators in Tables 1 and 2. Both quantlets are illustrated in [XFGSOPMD.xpl](#).

If in addition a dividend is paid during the time to maturity, we can use the following two quantlets to calculate the option prices.

```


erg = BlackScholesPathIndependentMDDiv (s0,r,div,vola
                                       ,dt,opt,itr,gen)
    MC estimation of the option price in the multidimensional Black
    Scholes model

erg = BlackScholesPathIndependentMDDivQMC (s0,r,div,vola
                                           ,dt,opt,itr,gen)
    QMC estimation of the option price in the multidimensional
    Black Scholes model

```

The additional argument  $div$  is a  $m$  dimensional vector of the continuously paid dividends. An application of these functions for our basket option is pro-

vided in

 XFGSOPMDDiv.xpl.

### 3.3 Restrictions for the payoff functions

Monte Carlo based option pricing methods are not applicable for all types of payoff functions. There is one theoretical, and some practical limitations for the method. Let us look at the theoretical limitation first.

In the derivation of the probabilistic error bounds we have to assume the existence of the payoff variance with respect to the risk neutral distribution. It follows that we are no longer able to derive the presented error bounds if this variance does not exist. However for most payoff functions occurring in practice and the Black Scholes model the difference between the payoff samples and the price can be bounded from above by a polynomial function in the difference between the underlying estimate and the start price for which the integral with respect to the risk neutral density exists. Consequently the variance of these payoff functions must be finite.

Much more important than the theoretical limitations are the practical limitations. In the first place Monte Carlo simulation relies on the quality of the pseudo random number generator used to generate the uniformly distributed samples. All generators used are widely tested, but it can't be guaranteed that the samples generated for a specific price estimation exhibit all assumed statistical properties. It is also important to know that all generators produce the same samples in a fixed length cycle. For example if we use the random number generator from Park and Miller with Bays-Durham shuffle, we will get the same samples after  $\approx 10^8$  method invocations.

Another possible error source is the transformation function which converts the uniformly distributed random numbers in normally distributed number. The approximation to the inverse of the normal distribution used in our case has a maximum absolute error of  $10^{-15}$  which is sufficiently good.

The most problematic cases for Monte Carlo based option pricing are options for which the probability of an occurrence of a strictly positive payoff is very small. Then we will get either price and variance estimates based on a few positive samples if we hit the payoff region or we get a zero payoff and variance if this improbable event does not occur. However in both cases we will get a very high relative error. More accurate results may be calculated by applying importance sampling to these options.

## References

- Bauer, H. (1991). *Wahrscheinlichkeitstheorie*, W. de Gruyter.
- Bosch, K. (1993). *Elementare Einführung in die Wahrscheinlichkeitsrechnung*, Vieweg.
- Boyle, P. P. (1977). Options: A monte carlo approach, *Journal of Financial Economics* 4: 323–338.
- Broadie, M., Glasserman, P., and Ha, Z. (2000) Pricing American Options by Simulation Using a Stochastic Mesh with Optimized Weights, *Probabilistic*

- Constrained Optimization: Methodology and Applications*, S. Uryasev ed., Kluwer.
- Marsaglia, George (1993 ) Monkey tests for random number generators, *Computers & Mathematics with Applications* **9**: 1-10.
- Niederreiter, H. (1992). *Random number generation and Quasi Monte Carlo methods*, 1 edn, Capital City Press, Montpellier Vermont.
- Joy, C.,Boyle, P., and Tan, K. S. (1996). Quasi monte carlo methods in numerical finance, *Management Science* **42**(6): 926–936.
- Tuffin, Bruno (1996). On the use of low discrepancy sequences in Monte Carlo Methods, *Technical Report IRISA - Institut de Recherche en Informatique et Systemes Aleatoires* **1060**.
- Morokoff, William J.,andCaflish, Russel E. (1996 ) Quasi-monte carlo simulation of random walks in finance, *In Monte Carlo and Quasi-Monte Carlo methods*, 340-352, University of Salzburg, Springer.
- Malvin H. Kalos (1986) *Monte Carlo Methods*, Wiley