# Computing Greeks with Multilevel Monte Carlo Methods using Importance Sampling

Supervisor - Dr Lukas Szpruch
Candidate Number - 605148




Dissertation for MSc Mathematical & Computational Finance
Trinity Term 2012

June 22, 2012

**Abstract**

This paper presents a new efficient way to reduce the variance of an estimator of popular payoffs and greeks encounter in financial mathematics. The idea is to apply Importance Sampling with the Multilevel Monte Carlo recently introduced by M.B. Giles. So far, Importance Sampling was proved successful in combination with standard Monte Carlo method. We will show efficiency of our approach on the estimation of financial derivatives prices and then on the estimation of Greeks (i.e. sensitivities of the payoffs with regards to the model parameters). We will perform our analysis in the Black & Scholes' framework. This study is then aimed to experiment and compare the impact of Importance Sampling on Multilevel Monte Carlo variance.

*key words:* Importance Sampling, Multilevel Monte Carlo, Monte Carlo, Milstein scheme, Variance Reduction method, Greeks, Likelihood Ratio Method, Pathwise Sensitivity Method.

## Acknowledgements:

*I would like to thank especially Dr Lukas Szpruch for the help he has given throughout this project and for the interesting meeting we held together. I would also like to thank Dr Gyurko for organizing the MSC of Mathematical and Computational Finance. I am honored to be part of Oxford University and more particularly of this MSc and to have been in contact with very competent and interesting individuals.*

# Introduction:

In this paper, we apply a new, simple and efficient way to reduce the variance of an estimator using Importance Sampling on both Monte Carlo and Multilevel Monte Carlo methods. To begin with, let us recall that Importance Sampling is a method to estimate the expected value of a random variable by changing the probability measure under consideration. Let X be a random variable, p(x) its probability distribution and $\widetilde{p}(x)$ its probability distribution under $\mathbb{Q}$, then:

$$\mathbb{E}[X] = \int x.p(x)dx = \int x.\frac{p(x)}{\widetilde{p}(x)}.\widetilde{p}(x)dx = \mathbb{E}^{\mathbb{Q}}[X.R(X)]. \tag{1}$$

R(.) is called the Radon-Nidodym derivative and is defined as:

$$R(X) = \frac{p(x)}{\widetilde{p}(x)}.$$

Although, Importance Sampling conceptually is a simple technique, 'in practice' it is not obvious how to find a measure $\widetilde{p}(x)$ that gives us a better estimator for the problem under consideration. Therefore, we developed a simple technique designed to deal with the simulation of rare events. First we will demonstrate effectiveness of our approach using standard Monte Carlo and later we will improve our estimator even further by combining it with Multilevel Monte Carlo. To the best of our knowledge, this approach has not been tested before.

Essentially, this thesis will compare the variance of the estimator for four different regimes:

1. Monte Carlo without Importance Sampling; which will be called 'MC off';
2. Monte Carlo with Importance Sampling; which will be called 'MC on';
3. Multilevel Monte Carlo without Importance Sampling; which will be called 'MLMC off';
4. Multilevel Monte Carlo with Importance Sampling; which will be called 'MLMC on';

We will also compute the Computational Cost of these methods in order to give the reader a rigorous and complete study of the developed method.

Thus, the goal is to find out which of the four previously mentioned estimators is the most effective. The graphic representation of our study is represented in figure 1.

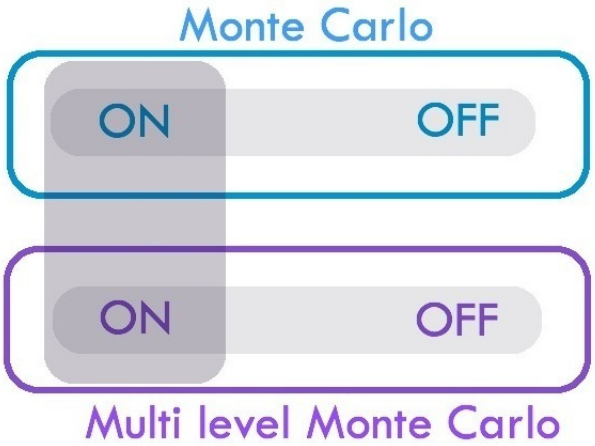



**Figure1:** Structure of our thesis. The grey clouds mean that we need to understand which methods have the biggest variance reduction impact on the estimator. So we need to compare the variance reduction impact of MC 'on' vs MC 'off'; MC 'on' vs MLMC 'on'; MLMC 'on' vs MLMC 'off'. We do not need to have a look at MLMC 'off' vs MC 'off' as literature already gives us a answer to that. Multilevel Monte Carlo is more efficient than Standard Monte Carlo.

Note that there is only three grey clouds since we already know the variance reduction superiority of 'MLMC off' over 'MC off'. We will therefore only focus on the three remaining clouds.This is the core of our work: trying to identify these relationships.

The thesis is structured as follows:

In the first part, we will present basic results of Monte Carlo simulation.

In the second part, we will develop the Importance Sampling for the simulation of rare events.

In the third part, we are going to test our method for the evaluation of the price of the derivatives. We will therefore have a first idea of the different relationships between the four approximation techniques we mentioned earlier on.

The fourth part is designed to analyse the Computational Costs of the four techniques.

In the fifth part of this thesis, we will extend our study to the simulation of Greeks. We will consider two types of estimators: Likelihood Ratio Method and Pathwise Sensitivity method.

Throughout this study, we will focus on European and Digital Call options as this will correspond to the smooth and non-smooth payoff.

# Part I
# General Results

In this part, we will present basic facts that we need to perform our study.

## I.1 Geometric Brownian Motion

Throughout the paper, we assume that the price process, $(S_t,\, t \in [0,T])$, follows a Geometric Brownian Motion, that is:

$$dS_t = \mu S_t dt + \sigma S_t dW_t,$$

where: $\mu$ is the drift (expected return value under the physic measure $\mathbb{P}$), $\sigma$ is the volatility, $W_t$ is the Brownian Motion.

Under the Risk Neutral Measure $\mathbb{Q}$, the above equation reads:

$$dS_t = rS_t dt + \sigma S_t dW_t^{\mathbb{Q}}, \tag{2}$$

where r is the constant risk-less discount factor. By Ito's lemma we have:

$$S_t = S_0 \exp\left((r - \frac{\sigma^2}{2})t + \sigma W_t^{\mathbb{Q}}\right),$$

where $S_0$ is the initial condition.

## I.2 European and Digital Call

First, the discounted payoff $P_{call}$ of a European Call option with strike K, interest rate r and time to maturity T has the following form:

$$P_{call} = \exp(-r.T).\max\left(S_T - K;\, 0\right). \tag{3}$$

Let us recall the price of European Call under Black and Scholes hypothesis is given by:

$$\begin{gathered} Price_{call} = S_0 N(d_1) - Ke^{-rT} N(d_2) \\ d_1 = \frac{\ln(S_0/K) + (r + \frac{\sigma^2}{2})T}{\sigma\sqrt{T}} \\ d_2 = d_1 - \sigma\sqrt{T} \end{gathered} \tag{4}$$

where $N(.)$ is the standard normal probability density function.

Also, the discounted payoff $P_{digital}$ of a Digital Call option with strike K, interest rate r and time to maturity T is given by:

$$P_{digital} = \exp(-r.T).\mathbb{I}_{\{S_T - K \geq 0\}}. \tag{5}$$

The value of the price of this derivative is under Black and Scholes hypothesis:

$$\begin{gathered} Price_{digital} = e^{-rT} N(d_2) \\ d_1 = \frac{\ln(S_0/K) + (r + \frac{\sigma^2}{2})T}{\sigma\sqrt{T}} \\ d_2 = d_1 - \sigma\sqrt{T} \end{gathered} \tag{6}$$

### I.1.3 Approximation techniques

Here in this paper we will focus on two approximation methods for the price process $S_t$. The Euler-Maruyama discretisation for equation (2) is given by:

$$S_{(n+1)\delta t} = S_{n\delta t} \left(1 + r.\delta T + \sigma \delta W_n\right), \tag{7}$$

where: N is the number of steps, $\delta T = \frac{T}{N}$, $\delta W_n = W_{(n+1)\delta T} - W_{n\delta T}$ and $n \in \{0, ...., N-1\}$.

The second approximation we use is the Milstein scheme. The Milstein scheme for equation (2) is given by:

$$S_{(n+1)\delta t} = S_{n\delta t} \left(1 + r.\delta t + \sigma \delta W_n + \frac{\sigma^2}{2}\left((\delta W_n)^2 - \delta T\right)\right), \tag{8}$$

where: N is the number of steps, $\delta T = \frac{T}{N}$, $\delta W_n = W_{(n+1)\delta T} - W_{n\delta T}$ and $n \in \{0, ...., N-1\}$.

The Milstein approximation gives a higher strong rate of convergence than Euler-Maruyama scheme. From Multilevel Monte Carlo perspective Milstein scheme gives optimal behaviour of variance and therefore this is our scheme of choice.

## I.2 Monte Carlo methods

Classic Monte Carlo methods are the standard and easiest way to approximate expected values. This quantity is particularly interesting in Mathematical Finance as, under risk neutral measure assumptions, pricing is equal to the discounted expected value of the payoff. For instance, in the case of European pricing where: $\bar{f}(S_t)$ is the discounted payoff (which is a function of the underlying $S_t$), $P(\overline{f})$ is the price of the derivatives with the discounted payoff $\overline{f}$, $\hat{P}_N(\overline{f})$ is the approximation of the price P with N simulated paths, $S_t$ is the underlying with $t \in [0, T]$, T is the maturity, then we have:

$$P(\overline{f}) = \mathbb{E}^{\mathbb{Q}}\left[\overline{f}(S_T)\right] \simeq \overline{P}_N(\overline{f}) = \frac{1}{N}\sum_{i=1}^{N} \bar{f}(S_T^i). \tag{9}$$

The advantages of this method are:

- simplicity and flexibility;
- possibility to implement it with parallelism to speed it up;
- can be easily generalised to multi dimensional problem;

Its weaknesses are:

- not as efficient as finite differences in very low dimension;
- not very efficient with options with optimal exercise time.

## I.3 Multilevel Monte Carlo methods

Recently, M.B. Giles introduced Multilevel Monte Carlo method that significantly improved Monte Carlo simulation. In its most general form, multilevel Monte Carlo (MLMC) simulation uses a number of levels of resolution, $l = 0, 1, ..., L,$, with $l = 0$ being the coarsest, and $l = L$ being the finest. In the context of a SDE simulation, level 0 may have just one timestep for the whole time interval [0; T], whereas level L might have $2^L$ uniform timesteps $\Delta t_l = 2^{-L}T$.

If P denotes the payoff (or other output functional of interest), and $P_l$ denotes its approximation on level l, then the expected value $\mathbb{E}[P_L]$ on the nest level is equal to the expected value $\mathbb{E}[P_0]$ on the coarsest level plus a sum of corrections which give the difference in expectation between simulations on successive levels. That is:

$$\mathbb{E}\left[\hat{P}_L\right] = \mathbb{E}\left[\hat{P}_0\right] + \sum_{l=1}^{L} \mathbb{E}\left[\hat{P}_l^f - \hat{P}_{l-1}^c\right]. \tag{10}$$

Using equation (1) to combine Importance Sampling with MLMC we obtain:

$$\mathbb{E}^{\mathbb{Q}}\left[\hat{P}_L R_L\right] = \mathbb{E}^{\mathbb{Q}}\left[\hat{P}_0 R_0\right] + \sum_{l=1}^{L} \mathbb{E}^{\mathbb{Q}}\left[\hat{P^f}_l R_l^f - \hat{P^c}_{l-1} R_{l-1}^c\right]. \tag{11}$$

Notice that in order to not violate this telescopic sum, we need to change the measure in a consistent way throughout the levels. That is following condition got to hold:

$$\mathbb{E}^{\mathbb{Q}}\left[\hat{P}^{f}{}_{l}R_{l}^{f}\right] = \mathbb{E}^{\mathbb{Q}}\left[\hat{P}^{c}{}_{l}R_{l}^{c}\right] \ for\, l = 0, 1, ..., L. \tag{12}$$

In the next section, we will develop the method that allows to hold condition (12).

# Part II
# Importance Sampling Methods

Importance Sampling can be very useful if we want to approximate rare events. Let us recall that Importance Sampling is used to evaluate the expected value of a random variable by changing the probability measure.

Let us consider the following example. If we want to approximate $\mathbb{P}[Z \geq 4]$ with Z being a standard normally distributed random variable ($Z \sim \mathcal{N}(0,1)$). As a standard normally distributed random variable has 99% to remain between $\pm 3$ standard deviation: it is a rare event. Thus, using Importance Sampling and changing the probability measure so that, under $\mathbb{Q}$ , $Z \sim \mathcal{N}(0,4)$ for instance will make the evalution of $\mathbb{P}[Z \geq 4]$ much more efficient.

If we want to consider a real financial situation, we can evoke insurance contracts to protect a certain client from a rare event that could cause significant damage. For instance, in the case of commodities companies that want insurance from tanker crashes, problems in the transport etc (which are rare events), they could protect themselves using a suitably designed digital option. This kind of insurance contracts may protect companies from large rise/fall of market values of some assets. This kind of contracts can be for example cash-or-nothing digital option with very high strike. That is in the case when an asset reaches a certain barrier, the contract will pay a large amount of money. Computing the price of such contracts can be very challenging as we need to accurately estimate very rare events.

## II.1 Our first approach

In the case of rare events, we want to use Multilevel Monte Carlo and combine it with Importance Sampling so that we don't need to simulate a large number of paths. This will reduce the Computational Cost which is the main advantage of Importance Sampling. However, as we explained in the previous section, we need to develop such a change of measure so that condition (12) holds.

Let us first consider simulation of Brownian Motion. In order to develop our method, we will use the following properties of Brownian Motion.

1. There is no scale dependance of the Brownian Motion. We have that $V_t = \frac{1}{\sqrt{c}} W_{ct}$ where $W_t$ is a Brownian Motion, for every c $> 0$, is another Brownian Motion.

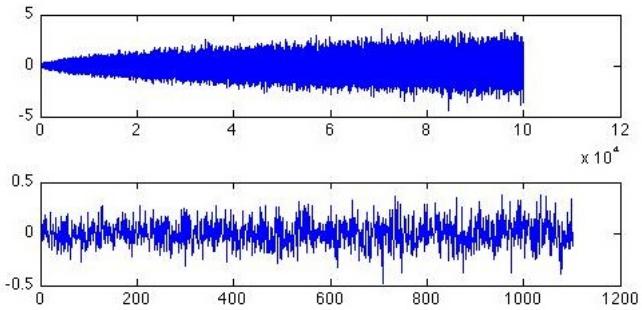



**Figure2:** Basic demonstration of time rescaled brownian motion. What we did is to rescale time to see that we keep similarities.

2. Let us use as a lemma the law of Iterated Logarithm:

*Law of Iterated Logarithm applied to Brownian Motion:*

*Suppose we have $W_t$ a Brownian Motion, then we have the following result:*

$$\limsup_{t\to+\infty} \frac{|W_t|}{\sqrt{2t\log\log t}} = 1 \; almost\, surely. \tag{13}$$

From equation (13), we obtain two opposite functions that act as a limit envelope of the Brownian Motion. This will allow us to gather all the paths in a restricted segment.

As we want to use this envelope over our Geometric Brownian Motion, an important remark is that if we add a linear term, we can have two opposite functions that englobes the move of a drifted Brownian Motion.

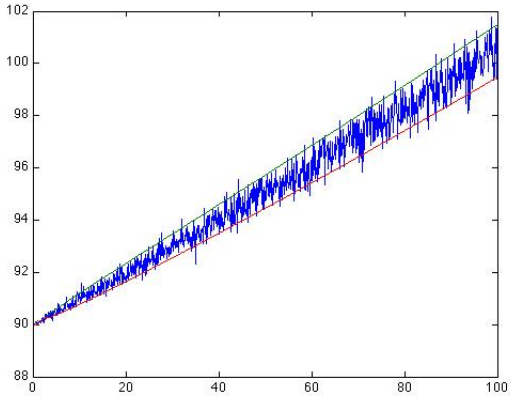



**Figure3:** Here, $S_t$ follows equation (2), it is a Geometric Brownian Motion. We then take $r = 0.05$, $\sigma = 0.2$, $S_0 = 90$ and T = 100.

These functions (envelope 1 and 2 respectively $E_1$ and $E_2$) are then given by:

$$E_{1/2}(t) = S_0 \exp\left(\mu t \pm \sigma\sqrt{2t\log\log t}\right).$$

Now, we want to apply this on a wide range of financial products. If we want to consider 'short maturity' (small T) products for instance, we can apply the 'scaling' arguments (property 1.) so that we can consider an analoguous situation where T is big enough to use the envelope functions.

The main idea of our study is to ensure that all the simulated paths will terminate near the strike. If we denote $K$ as the strike of the European or Digital Call option, we need to specify a segment, $[K - \delta K; K + \delta K]$ (with $\delta > 0$), where we will gather all the paths.

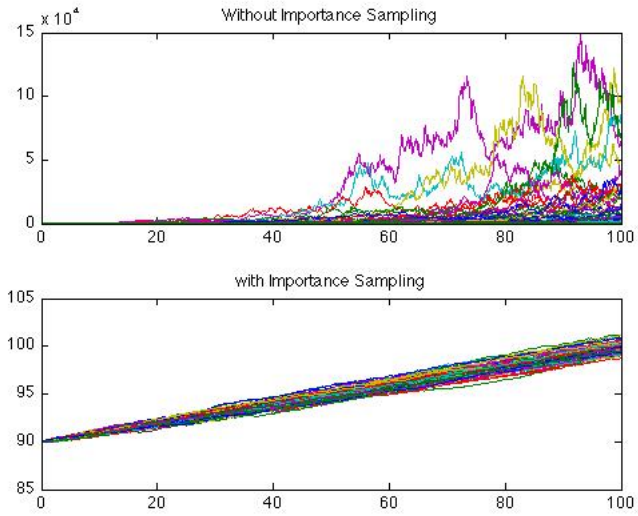



**Figure4:** Use of Importance Sampling and envelope functions to gather paths in a restricted area. $S_t$ follows equation (2), it is a Geometric Brownian Motion. Here, for both graphs: $S_0 = 90$, $r = 0.05$, $\sigma = 0.2$, K=100 and $\delta K$=1.

The figure 4 shows, due to this method, by using only few paths, we could get a very good approximation of the price of a European or Digital option by means of Monte Carlo method.

In order to use this method, we need to compute the new drift $\widetilde{\mu}$ and volatility $\widetilde{\sigma}$ of the asset under the new probability measure $\mathbb{Q}$. This is straightforward as we want two conditions at the maturity. We want the lower part of the envelope to finish at $\log(K - \delta K)$ and the upper part to finish at $\log(K + \delta K)$. In order to find $\widetilde{\mu}$ and $\widetilde{\sigma}$, we need to solve the following system:

$$\begin{cases} S_0 \exp(\widetilde{\mu}T + \tilde{\sigma}E_1(T)) = K + \delta K \\ S_0 \exp(\widetilde{\mu}T + \tilde{\sigma}E_2(T)) = K - \delta K \end{cases},$$

$$\begin{pmatrix} T & \sqrt{2T\log\log T} \\ T & -\sqrt{2T\log\log T} \end{pmatrix} \begin{pmatrix} \widetilde{\mu} \\ \widetilde{\sigma} \end{pmatrix} = \begin{pmatrix} \log\left(\frac{K+\delta K}{S_0}\right) \\ \log\left(\frac{K-\delta K}{S_0}\right) \end{pmatrix}. \tag{14}$$

Therefore, in the case of this change of measure, the Radon-Nykodim transformation associated will be:

$$\mathbb{P}\left[S_T \geq K\right] = \mathbb{E}\left[1_{S_T \geq K}\right] = \mathbb{E}^{\mathbb{Q}}\left[1_{S_T \geq K} R_{\widetilde{\mu},\widetilde{\sigma}}\right] = \int 1_{S_T \geq K} R_{\widetilde{\mu},\widetilde{\sigma}}(x)\tilde{p}(x,\mu,\sigma)dx,$$

$$\tilde{p}(x,\mu,\sigma) = \frac{1}{\sqrt{2\pi\tilde{\sigma}^2}} \exp\left(-\frac{(x-\tilde{\mu})^2}{2\tilde{\sigma}^2}\right).$$

Thus we have:

$$R_{\widetilde{\mu},\widetilde{\sigma}} = \frac{\tilde{\sigma}}{\sigma} \exp\left(-\frac{(x-\tilde{\mu})^2}{(x-\mu)^2}\frac{\sigma^2}{\tilde{\sigma}^2}\right).$$

Where we consider $S_t$ that follows equation (2) and

$$x = \log\frac{S_T}{S_0}.$$

Hence, we are considering the ratio of two log-normal distributions as we set ourselves with a Geometric Brownian Motion.

## II.2 Limit and new approach

### II.2.1 Singular Measures

When we started to experiment this idea of change of measure, we noticed that the results were not satisfactory. By decreasing the time-steps in Milstein scheme (10), we started constructing two singular measures. Refer a read to [2] for more details.

In the case of $S_0 = 10$, $K = 200$, $T = 10$, $\sigma = 0.20$, $r = 0.05$, by solving (14), we obtained: (with $\mu = r - \frac{\sigma^2}{2}$)

| $\mu$ | $\widetilde{\mu}$ | $\sigma$ | $\widetilde{\sigma}$ |
|---|---|---|---|
| 0.03 | 0.29954 | 0.2 | 0.0000612 |

**Comment:** $\mu$ and $\sigma$ are the parameters of the underlying following a Geometric Brownian motion under $\mathbb{P}$, $\widetilde{\mu}$ and $\widetilde{\sigma}$ are the parameters under $\mathbb{Q}$. We see that $\widetilde{\sigma}$ is decreasing extremely as we are changing measure.

Here, $\widetilde{\mu}$ and $\widetilde{\sigma}$ are the new parameters of the lognormal distribution under $\mathbb{Q}$. In this case, we cannot use Importance Sampling to change both $\mu$ and $\sigma$ as we would like to. The reason is $\widetilde{\sigma}$ tends to be too small and makes the Radon Nikodym derivative explodes.

### II.2.2 New approach

As we have seen previously, we cannot use Multilevel or even Monte Carlo with this type of change of measure in the case of very rare events. What we will do now is to focus on changing only the drift in order for the stochastic process $(S_.)$ to be close to the strike at T.

So, now we allow ourselves to only focus on a change of drift to increase the probability of paths landing near of the strike. By using the Geometric Brownian Motion assumptions:

$$S_T = S_0 \exp\left(\mu T + \sigma B_T\right).$$

And we want:

$$S_T \simeq K,$$

which is equivalent to:

$$\tilde{\mu} T + \sigma W_T^{\mathbb{Q}} \simeq \log\left(\frac{K}{S_0}\right).$$

We translate this condition by saying that on average we want:

$$\mathbb{E}^{\mathbb{Q}}\left[\tilde{\mu} T + \sigma W_T^{\mathbb{Q}}\right] = \log\left(\frac{K}{S_0}\right),$$

Thus:

$$\mathbb{E}\left[\tilde{\mu} T + \sigma W_T^{\mathbb{Q}}\right] = \mathbb{E}\left[\tilde{\mu} T\right] + \mathbb{E}\left[\sigma W_T^{\mathbb{Q}}\right] = \mathbb{E}\left[\tilde{\mu} T\right] = \log\left(\frac{K}{S_0}\right)$$

Hence:

$$\tilde{\mu} = \frac{1}{T} \log \frac{K}{S_0} \tag{15}$$

This is the $\tilde{\mu}$ we will have for our Geometric Brownian Motion in the new probability space after using Importance Sampling. Figure 5 shows that by changing only the drift, we still obtain fairly satisfactory results.

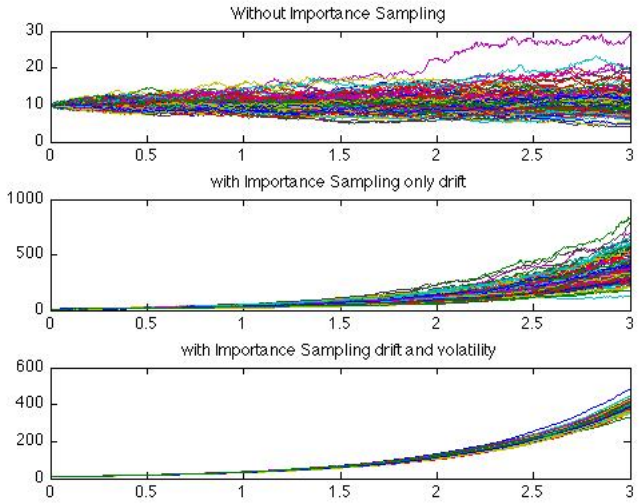



**Figure5:** Generation of 100 paths in three cases, without Importance Sampling, with Importance Sampling on only the drift, with Importance Sampling on both drift and volatility. $S_t$ follows a Geometric Brownian Motion, equation (2). Parameters are: $T = 3$, $S_0 = 10$, $K = 400$, $r = 0.05$ and $\sigma = 0.20$. No discretisation of the asset.

If we now consider the average absolute distance of the final value of the paths to the strike: $|S_T - K|$, we observe this:

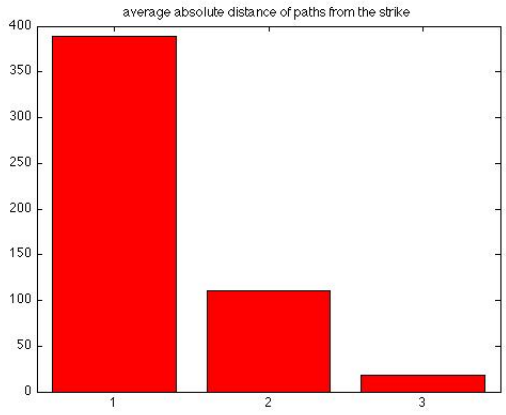


**Figure6:** Evolution of the average absolute distance of the paths at maturity to the strike: $|S_T - K|$. ($K = 400$, $T = 3$, $S_0 = 10$, r=0.05, $\sigma = 0.2$ and $S_t$ follows equation (2)). Cases are: 1. Without Importance Sampling 2. Change of drift 3. Change of drift and volatility. No discretisation of the asset.

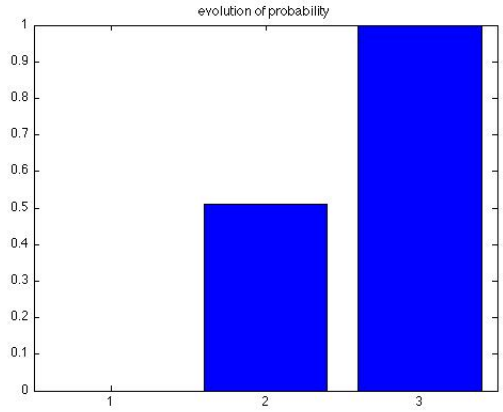



**Figure7:** Evolution of the probability of the payoff to be at maturity within the range: $[K - \delta K; K + \delta K]$ with the same parameters as set previously. Cases are the same as in Figure 5 and 6. No discretisation of the asset.

Figure 6 and 7 confirms that changing only the drift is indeed a good candidate for Importance Sampling. This is our new approach in order to perform our Monte Carlo estimation and Multilevel Monte Carlo estimation.

# Part III
# Comparison of Variance Reduction Methods

In this section, we will analyse the impact of the Importance Sampling method we explained on the previous section on Standard and Multilevel Monte Carlo. As we mentioned in the Introduction, we will focus on European and Digital Call. Those payoffs are defined in section I.2. First, the variance for standard Monte Carlo method is:

$$\mathbb{V}(\bar{P}(\bar{f})) = \frac{N}{N-1}\left(\bar{P}_N(\bar{f}^2) - \left(\bar{P}_N(\bar{f})\right)^2\right).$$

In order to derive the estimator for the variance of Multilevel Monte Carlo, let us give more details on MLMC simulation.

## III.1 Multilevel Monte Carlo

As we mentioned in the Introduction, our paper introduces a new and very useful way to use Importance Sampling with Multilevel Monte Carlo. Previously, we stated the basic definitions that will be recurrent throughout our studies. Here, we are going to explain what is Multilevel Monte Carlo.

Let us recall that the Multilevel Monte Carlo estimator has the form:

$$\begin{aligned} &\mathbb{E}\left[\hat{P}_L\right] = \mathbb{E}\left[\hat{P}_0\right] + \sum_{l=1}^{L} \mathbb{E}\left[\hat{P}_l^f - \hat{P}_{l-1}^c\right] \\ \textit{with} \quad &\mathbb{E}\left[\hat{P}_l^f - \hat{P}_{l-1}^c\right] \simeq Y_l = \frac{1}{N_l}\sum_{i=1}^{N_l}\left(\left(\hat{P}_l^f\right)^{(i)} - \left(\hat{P}_{l-1}^c\right)^{(i)}\right) \end{aligned}, \tag{16}$$

where $\hat{P}_l^f$ is the fine approximation ($2^l$ steps for the discretisation) and $\hat{P}_{l-1}^c$ is the coarse approximation ($2^{l-1}$ steps for the discretisation).

The variance of this method is:

$$\mathbb{V}\left[\hat{P}_L\right] = \sum_{l=0}^{L} \frac{1}{N_l}\mathbb{V}\left[Y_l\right] \tag{17}$$

In order to implement the MLMC estimator, we need to find optimal parameters for L and $N_l$ for $l \in [0, L]$. M.B. Giles gave in [1] a full detailed explanation, we will only name the results.

Let us start with a quick analysis. In this case, we will make some extra-assumptions. We will consider a Euler-Maruyama discretisation -equation (7)-, with Lipschitz payoff function and a underlying that satisfies equation (2). In this case, there is a $\mathcal{O}(\frac{T}{M^L})$ strong convergence. Hence, as $l \to \infty$, we have:

$$\begin{array}{c} \mathbb{E}\left[\hat{P}_l - P\right] = \mathcal{O}(\frac{T}{M^l}) \\ \mathbb{V}(\hat{P}_l - P) = \mathcal{O}(\frac{T}{M^l}) \end{array}.$$

Thus, as we want to set the MSE to be a $\mathcal{O}(\epsilon^2)$ with $\epsilon > 0$, that is:

$$MSE = \mathbb{V}(\hat{P}(f,S)) + \left(\mathbb{E}\left[\hat{P}_l\right] - \mathbb{E}\left[P\right]\right)^2 = \mathcal{O}(\epsilon^2),$$

hence:

$$L = ceiling(-\frac{\log \epsilon}{\log M}) + \mathcal{O}(1) \Longrightarrow \begin{array}{c} \mathbb{E}\left[\hat{P}_l - P\right] = \mathcal{O}(\epsilon) \\ \mathbb{V}(\hat{P}_l - P) = \mathcal{O}(\epsilon) \end{array}.$$

Finally, by using equation (17), we have:

$$N_l = \mathcal{O}\left(\frac{LT}{\epsilon^2 M^l}\right) \Longrightarrow \mathbb{V}(\hat{P}(f,S)) = \mathcal{O}(\epsilon^2)$$

It can be shown that the optimal $N_l$ is:

$$N_l = \frac{2}{\epsilon^2}\sqrt{\mathbb{V}\left(Y_l\right)\frac{T}{M^l}}\left(\sum_{l=0}^{L}\sqrt{\mathbb{V}\left(Y_l\right)\frac{M^l}{T}}\right).$$

We will refer to M.B. Giles paper [1] for more details. A full Matlab code of this method can be found in Annex 2.

## III.2 Monte Carlo 'On' vs Monte Carlo 'Off'

So, as we have said before we will first use a Monte Carlo method to estimate the price of different payoffs: European and Digital call. During this study, we used a Milstein appoximation. Also, we have the following notation:

- $error = \mathbb{E}\left[|f(S_T) - V|\right]$, where V is the actual payoff value;
- $MSE = \mathbb{E}\left[\left|(f(S_T) - V)^2\right|\right]$, standard mean squared error;
- $expected\,value = \mathbb{E}\left[f(S_T)\right]$, expected value of the estimation;
- $variance = \mathbb{V}\left[f(S_T)\right]$, variance of the estimator.

### III.2.1 European Call

Here, we are focusing on European Call. It is defined in Section I.2 and the payoff is given in equation (3). Also, we need the Black & Scholes value of the European Call. Closed Formula of the price is given in equation (4). In the table below we have the main results for standard Monte Carlo with ('on') and without ('off') Importance Sampling in a standard set of parameters

| Value | Expected Value off | Expected Value on | Error on | MSE on | Variance on |
|---|---|---|---|---|---|
| 1.91 e-04 | 0 | 1.89 e-04 | 2.54 e-05 | 1.03 e-09 | 1.02 e-09 |

**Comment:** Those results have been obtained with $S_0 = 20$, $r = 0.05$, $\sigma = 0.2$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

As you can see, the event is too rare to be considered by the standard Monte Carlo method. However, using Importance Sampling we can approximate the payoff very efficiently.

Let us consider the following payoffs with and without Importance Sampling:

$$\hat{P}_N = \frac{1}{N} \sum_i \exp(-rT) \max\left(S_T^{(i)} - K; 0\right).$$

$$\hat{P}_N^{IS} = \frac{1}{N} \sum_i \exp(-rT) \max\left(S_T^{(i)} - K; 0\right) R_{\tilde{\mu}}\left(S_T^{(i)}\right),$$

where $S_T$ follows a Geometric Brownian Motion with a new drift $\tilde{\mu}$ under $\mathbb{Q}$. The Figure 8 shows the evolution of variance for different initial conditions $S_0$ (the further from the strike, the rare the event).

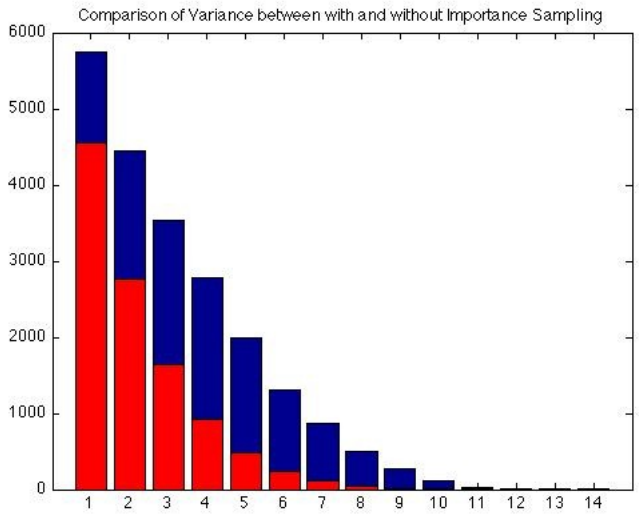



**Figure8:** Evolution of the variance in the case of Importance Sampling (in grey) and without Importance Sampling (in black). The samples are done with various value of $S_0$. These values are first near to the strike and then we consider rarer events. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

### III.2.2 Digital European Call

Here, we will focus on Digital European Call Option. We defined this option in Section I.2. We also gave the closed formula of the payoff in equation (5) and the price in equation (6). Let us recall that the estimators we are using in this case for Standard Monte Carlo are:

$$\hat{P}_N = \frac{1}{N} \sum_i \exp(-rT) \mathbb{I}_{S_T^{(i)} \geq K}.$$

$$\hat{P}_N^{IS} = \frac{1}{N} \sum_i \exp(-rT) \mathbb{I}_{S_T^{(i)} \geq K} R_{\tilde{\mu}} \left( S_T^{(i)} \right),$$

where $S_T$ follows a Geometric Brownian Motion with a new drift $\tilde{\mu}$ under $\mathbb{Q}$. Figure 9 shows the behaviour of variance for different initial conditions.

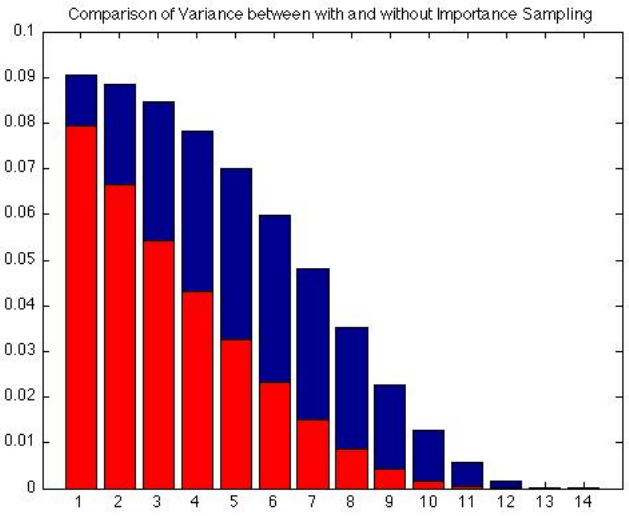



**Figure9:** Evolution of the variance in the case of Importance Sampling (grey) and without Importance Sampling (black). The samples are done with various value of $S_0$. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

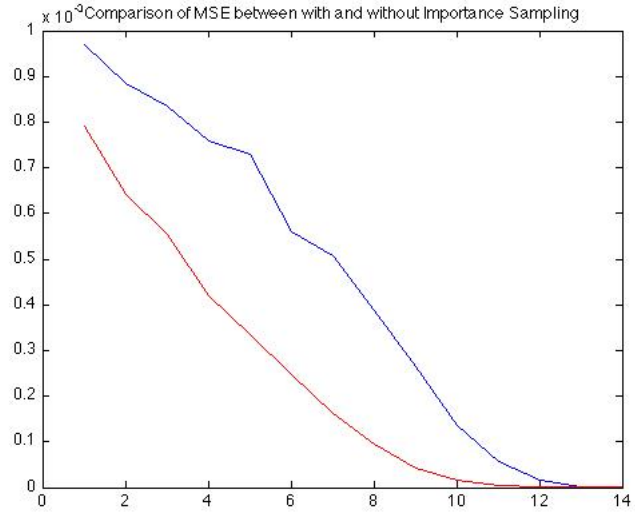


**Figure10:** Evolution of the Mean Squared Error between on (grey) and off (black) Importance Sampling Monte Carlo. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

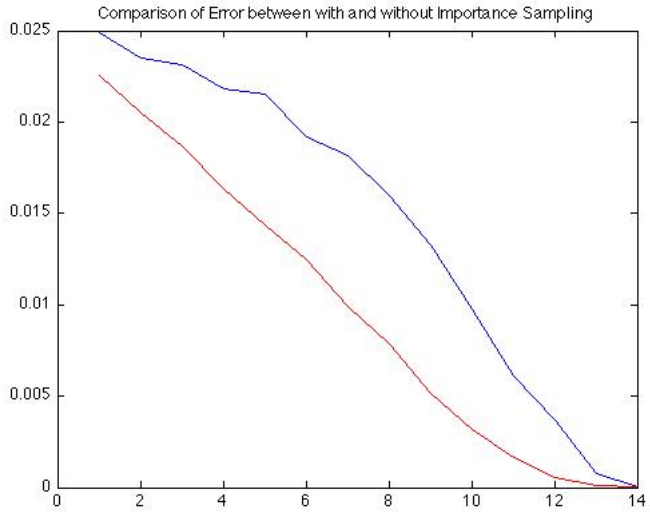



**Figure11:** Evolution of the Error between on (grey) and off (black) importance sampling Monte Carlo. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

In figures 10 and 11, we analysed the mean-squared-error and the error for different initial conditions. When we consider very high distance between $S_0$ and $K$ (value n°13-14), the 'off' Monte Carlo (i.e. the one without Importance Sampling) always returns 0 because it cannot handle the approximation of a very small option. It seems from the three previous curves that in the case of rare events, the Monte Carlo 'on' and 'off' are similar, but it is not the case: the 'off' Monte Carlo is unable to give an estimation. This can be seen on figure 12 when we consider extreme rare event.

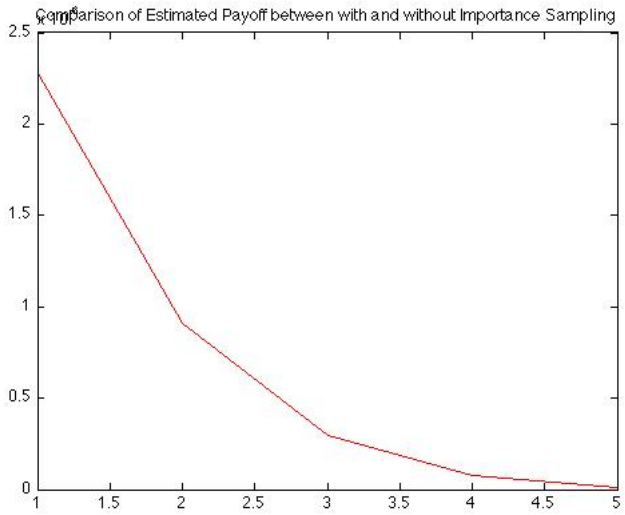



**Figure12:** Expected value of the Monte Carlo estimator on (grey) and off (black). $S_0 = 20$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2. THere is only the grey curve as the black sticks to 0.

## III.3 Multilevel Monte Carlo 'On' vs Multilevel Monte Carlo 'Off'

Here, we will focus on Multilevel Monte Carlo as we introduced it in section I.3. We used equation (17) to estimate the variance.

### III.3.1 European Call

In this section, we are going to analyse the impact of Importance Sampling in terms of variance reduction on Multilevel Monte Carlo. We gather in table 13 the results for both the value estimation and the variance reduction:

| $S_0$ | Call Value | MLMC 'off' | MLMC 'on' |
|---|---|---|---|
| 140 | 42.501205 | 42.601439 | 42.497173 |
| 130 | 35.710638 | 35.679134 | 35.721340 |
| 120 | 29.293691 | 29.293661 | 29.293558 |
| 110 | 23.379729 | 23.346370 | 23.378433 |
| 100 | 18.031898 | 17.983233 | 18.033380 |
| 90 | 13.314569 | 13.311271 | 13.306934 |
| 80 | 9.2888101 | 9.2998682 | 9.2777760 |
| 70 | 6.0049588 | 6.0156556 | 6.0048039 |
| 60 | 3.4912938 | 3.5025771 | 3.4895390 |
| 50 | 1.7381873 | 1.60528535 | 1.7175196 |
| 40 | 0.67901464 | 0.15622010 | 0.6788842 |
| 30 | 0.17453156 | 0.10751252 | 0.1726515 |
| 20 | 0.0191756 | 0 | 0.0183061 |
| 10 | 1.907e-4 | 0 | 1.891e-4 |

**Table13:** First Column: Values of $S_0$. Second Colum: Value of the Call with a Geometric Brownian Motion approximation. Third Column: Value of the Call with Multilevel Monte Carlo without Importance Sampling ('off'). Fourth Column: Value of the Call with Multilevel Monte Carlo with Importance Sampling ('on'). Initial Parameters are: $T = 10$, $K = 200$, $r = 0.05$ and $\sigma = 0.20$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

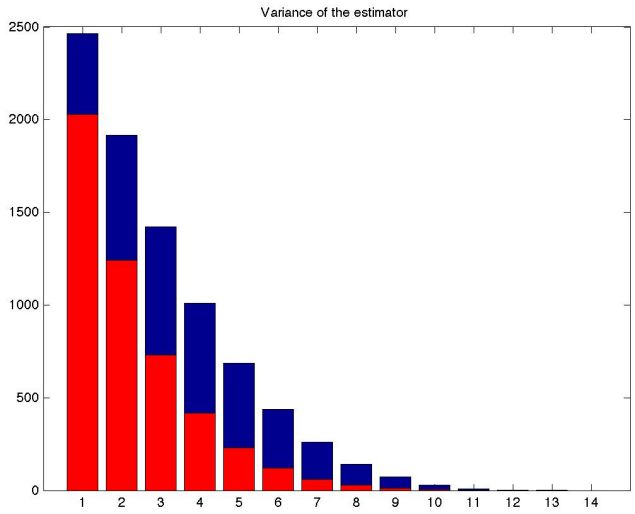



**Figure14:** Evolution of the variance in the case of Importance Sampling (grey) and without Importance Sampling (black). The samples are done with various value of $S_0$. These values are first near to the strike and then we consider rarer events. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Initial Parameters are: $T = 10$, $K = 200$, $r = 0.05$ and $\sigma = 0.20$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

Figure 14 is similar to figure 9, Importance Sampling has the same effect on Multilevel Monte Carlo as it had on Monte Carlo. It significantly reduces the variance, as expected.

### III.3.2 Digital Call Option

Now we are considering the variance of Multilevel Monte Carlo with and without Importance Sampling in the case of a discontinuous payoff. Table 15 confirms superiority of our approach with regards to the standard Monte Carlo approach.

| $S_0$ | Value Digital | Value 'off' | Value 'on' |
|---|---|---|---|
| 50 | 0.0260422582 | 0.01802998 | 0.02554849756 |
| 47.5 | 0.0218559316 | 0.01162486 | 0.02104766286 |
| 45 | 0.0180569457 | 0 | 0.01791993715 |
| 42.5 | 0.0146536465 | 0.00332660 | 0.01473198523 |
| 40 | 0.0116498190 | 0 | 0.01010447113 |
| 37.5 | 0.0090439562 | 0 | 0.00899062316 |
| 35 | 0.0068285708 | 0 | 0.00671805078 |
| 32.5 | 0.00498960889 | 0 | 0.00489204506 |
| 30 | 0.00350604889 | 0 | 0.003778892797 |
| 27.5 | 0.00234979023 | 0 | 0.000812082244 |
| 25 | 0.00148595954 | 0 | 0.001414652725 |
| 22.5 | 0.00087377589 | 0 | 0.000845950248 |
| 20 | 0.00046811110 | 0 | 0.000222961193 |
| 17.5 | 0.00022183502 | 0 | 0.000149321820 |
| 15 | 0.00008891900 | 0 | 0.000085688117 |
| 12.5 | 0.00002804750 | 0 | 0.000022811351 |
| 10 | 0.00000613533 | 0 | 0.000059812973 |
| 7.5 | 0.00000072517 | 0 | 0.000005269010 |
| 5 | 0.00000002547 | 0 | 0.000000062523 |

**Table15:** First Column: Values of $S_0$. Second Column: Value of the Digital Call with a Geometric Brownian Motion approximation. Third Column: Value of the Digital Call with Multilevel Monte Carlo without Importance Sampling ('off'). Fourth Column: Value of the Digital Call with Multilevel Monte Carlo with Importance Sampling ('on'). Initial Parameters are: $T = 10$, $K = 200$, $r = 0.05$ and $\sigma = 0.20$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

Table 15 shows that MLMC 'off' estimator is unable to give an approximation of the value whereas the MLMC 'on' can.

Here is the variance of the Multilevel Monte Carlo on estimator. We will not display the variance of MLMC 'off' as it does not even bring any estimation.

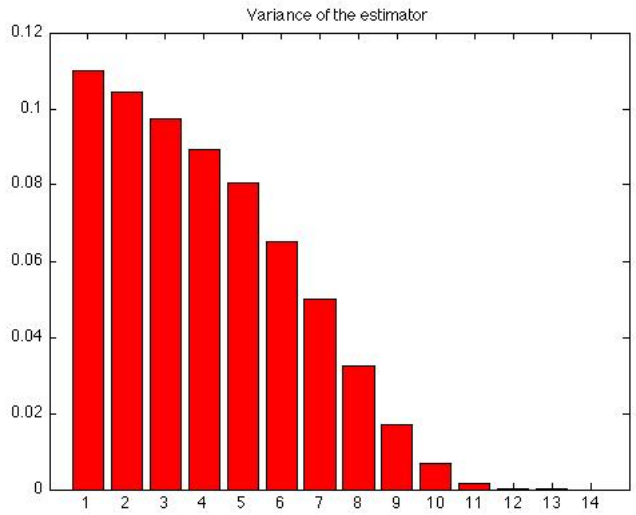



**Figure16:** Evolution of the variance in the case of Importance Sampling (grey) and without Importance Sampling (black). The samples are done with various value of $S_0$. These values are first near to the strike and then we consider rarer events. Initial Parameters are: $T = 10$, $K = 200$, $r = 0.05$ and $\sigma = 0.20$. Each rows are for values of $S_0 = \{50; 47,5; 45; 42,5; 40; 37,5; 35; 32,5; 30; 27,5; 25; 22,5; 20; 17,5; 15; 12,5; 10; 7,5; 5\}$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

We compared the difference between MLMC 'on' and MLMC 'off', and the outcome is quite clear: MLMC 'on' clearly outperforms the variance reduction results.

## III.4 Monte Carlo 'On' vs Multilevel Monte Carlo 'On' - European Call

From previous sections we have seen that results for European and Digital Call were similar, so we will focus only on European Call. From figure 17 we see that MLMC 'on' clearly outperforms MC 'on'.

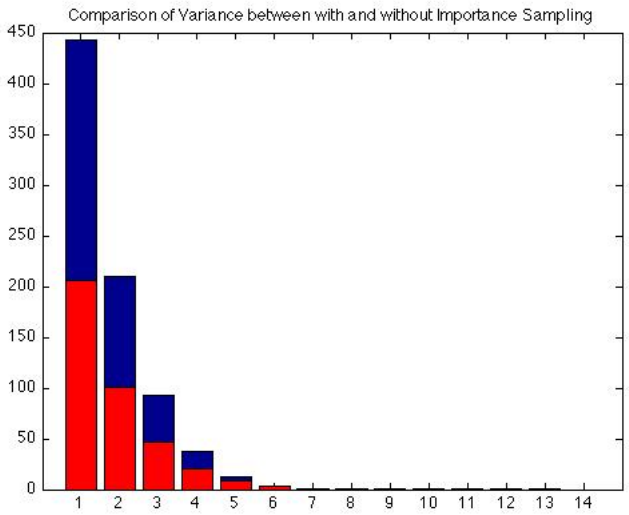



**Figure17:** Comparison of MLMC 'on' (grey) and MC 'on' (black). The samples are done with various value of $S_0$. These values are first near to the strike and then we consider rare events. Initial Parameters are: $T = 10$, $K = 200$, $r = 0.05$ and $\sigma = 0.20$. Each rows are for values of $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

# Part IV
# Computational Cost: MC 'on-off' vs MLMC 'on-off'

In this section, we will analyse the Computational Cost for the four previously introduced methods. We will fix the Mean Squared Error at a certain value and see the difference of Computational Cost between different approaches.

## IV.1 Theoretical Computational Cost

As we specified earlier on, we are mainly interested to put some boundaries on the Mean Squared Error. Let us remind that:

$$MSE = \mathbb{E}\left[\left(\hat{Y} - \mathbb{E}\left[Y\right]\right)^2\right],$$

$$MSE = \mathbb{E}\left[\left(\hat{Y} - \mathbb{E}\left[\hat{Y}\right]\right)^2\right] + \mathbb{E}\left[\left(\mathbb{E}\left[\hat{Y}\right] - \mathbb{E}\left[Y\right]\right)^2\right].$$

As it is described in M.B. Giles work, for instance in [8], our goal is to be able to fix this MSE for both standard Monte Carlo and Multilevel Monte Carlo. More precisely, we want to set $MSE = \mathcal{O}\left(\varepsilon^2\right)$. Let us see how we can do this:

1. For Standard Monte Carlo, we have:

$$MSE = \mathcal{O}\left(\frac{1}{N_{paths}}\right) + \mathcal{O}\left(\Delta t^2\right).$$

Thus, we need to have:

$$N_{paths} = \mathcal{O}\left(\varepsilon^{-2}\right),$$

$$\Delta t = \mathcal{O}\left(\varepsilon^{-1}\right).$$

And, in this case, as we roughly have:

$$Computational\,cost_{Std\,MC} = N_{paths}.Complexity_{1path} \simeq N_{paths}.\Delta t = \mathcal{O}\left(\varepsilon^{-3}\right).$$

2. For Multilevel Monte Carlo, we have in paper [8] the following theorem:

***Theorem*** *Let $P$ denote a functional of the solution of a stochastic differential equation, and let $P_l$ denote the corresponding level $l$ numerical approximation. If there exist independent estimators $Y_l$ based on $N_l$ Monte Carlo samples, and positive constants $\alpha, \beta, \gamma, c_1, c_2, c_3$ such that $\alpha \geq \frac{1}{2}\min(\beta, \gamma)$ and:*

**i)** $|\mathbb{E}[P_l - P]| \leq c_1 2^{-\alpha l}$

**ii)** $\mathbb{E}[Y_l] = \begin{cases} \mathbb{E}[Y_0], & l = 0 \\ \mathbb{E}[P_l - P_{l-1}], & l > 0 \end{cases}$

**iii)** $\mathbb{V}[Y_l] \leq c_2 N_l^{-1} 2^{-\beta l}$

**iv)** $C_l \leq c_3 N_l 2^{\gamma l}$ where $C_l$ is the computation complexity of $Y_l$

*then, there exists a positive constant $c_4$such that for any $\varepsilon < e^{-1}$ there are values $L$ and $N_l$ for which the multilevel estimator*

$$Y = \sum_{l=0}^{L} Y_l \ ,$$

*has a mean squared error with bound*

$$MSE < \varepsilon^2,$$

with a computational complexity C with bound

$$C \leq \begin{cases} c_4 \varepsilon^{-2} & \beta > \gamma \\ c_4 \varepsilon^{-2} (\log \varepsilon)^2 & \beta = \gamma \\ c_4 \varepsilon^{-2-(\gamma-\beta)/\alpha} & 0 < \beta < \gamma \end{cases} .$$

This theoretical study shows us the logic of this section: we fix the MSE to be at a certain range, and we see how the Computational Cost evolves for the four different techniques (MC 'off', MC 'on', MLMC 'off', MLMC 'on'). This is a complementary study of the one we did before where we analysed the variance reduction impact.

## IV.2 Comparison between the methods

In this section we are going to show the results of the evolution of computation cost in each of the following case:

1. MLMC 'off' vs. MC 'off';
2. MLMC 'on' vs MC 'on';
3. MC 'off' vs MC 'off";
4. MLMC 'off' vs MLMC 'on'.

But, as you can imagine, we need to specify the type of payoff we are going to use. As we want to first start with simple payoff, we are going to stick to the two we used before:

1. European Call;
2. European Digital Call.

### IV.2.1 European Call

Let us remind the discounted payoff of the European Call and its Monte Carlo estimator are given by:

$$P_{call} = \exp(-rT)\max(S_T - K; 0),$$

$$\hat{P}_N = \frac{1}{N}\sum_i \exp(-rT)\max\left(S_T^{(i)} - K; 0\right).$$

Thus, figure 18 now presents the comparison of the Computational Cost for all the methods.

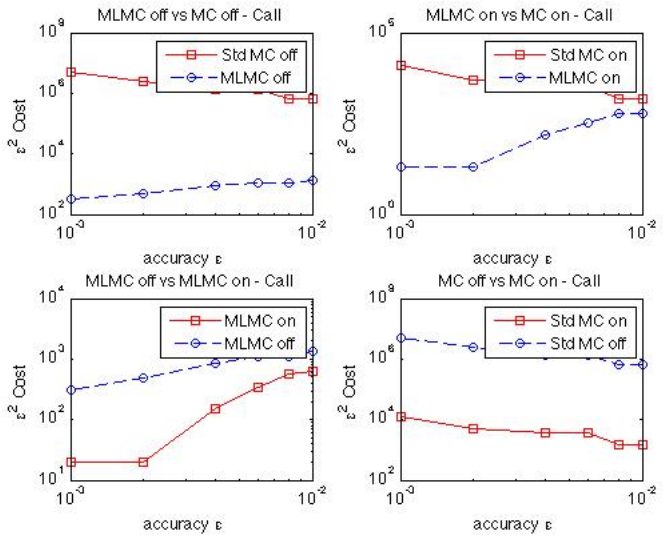



**Figure18:** 1) MLMC 'off' vs MC 'off' 2) MLMC 'on' vs MC 'on' 3) MLMC 'off' vs MLMC 'on' 4) MC 'off' vs MC 'on'. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 100$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

So let us get through these graphs of Figure 18:

1. In the first one, we can see that first, the MLMC 'off"s $\varepsilon^2.Computational\,Cost$ is roughly a constant function of the accuracy $\varepsilon$ (otherwise there would be some $\log\varepsilon$ term). This is consistent with our theoretical expectation as we observe the $\mathcal{O}\left(\varepsilon^2\right)$ behavior. If we have a look at the shape of the standard MC 'off"s $\varepsilon^2.Computational\,Cost$ we see a decreasing linear function of the accuracy with slope -1. As the previous graphs are in a log log scale, we see that we obtain the theoretical $\mathcal{O}(\varepsilon^{-3})$ behavior. As a result, we can observe how MLMC diminishes the Computational Cost in comparison to MC;

2. In the second one (top right), we can have the same analysis as we did previously. As we could imagine, changing the measure a.k.a using Importance Sampling, will not affect the behaviour of the Computational Cost. Hence, as we can observe, we hold a roughly constant MLMC $\varepsilon^2.Computational\,Cost$. Note that it is roughly constant since a $\log\varepsilon$ term can appear. Thus, in this case, we will get the slightly increasing term and have a positive slope;

3. In figure 3 (below left), we are comparing the MLMC 'on' with the MLMC 'off' $\varepsilon^2.Computational\,Cost$. As we can see, both of the curves roughly

keeps the same 'shape': it shows that we keep a $\mathcal{O}(\varepsilon^{-2})$ and the supplementary $\log \varepsilon$ term. Also, we can see that the 'on' option clearly diminishes the computational cost very significantly;

4. In the last figure, we do the same comparison as in point 3. and we come to the same conclusion: we keep the shape of the $\varepsilon^2.Computational\,Cost$ in $\mathcal{O}(\varepsilon^{-3})$ and we reduce significantly this cost with the Importance Sampling 'on'.

### IV.2.2 European Digital Call

Similarly, let us remind the discounted payoff of the Digital Call and its Monte Carlo estimator are given by:

$$P_{digital} = \exp(-rT)\,\mathbb{I}_{S_T \geq K},$$

$$\hat{P}_N = \frac{1}{N} \sum_i \exp(-rT)\,\mathbb{I}_{S_T^{(i)} \geq K}.$$

Thus, figure 19 presents the comparison of the Computational Cost of the Methods.

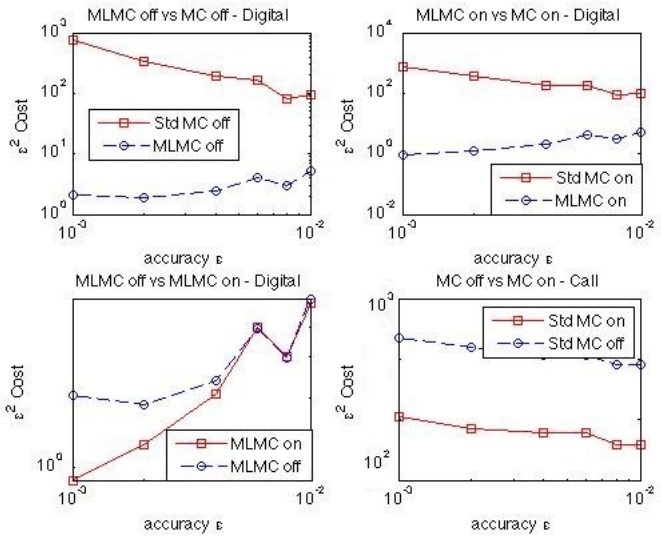



**Figure19:** Same comparison as in the case of European Call but for European Digital Call. 1) MLMC 'off' vs MC 'off' 2) MLMC 'on' vs MC

'on' 3) MLMC 'off' vs MLMC 'on' 4) MC 'off' vs MC 'on'. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 100$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

The general comment we can make is that we have the expected results.

# Part V
# Computation of Greeks

This section presents the last step of our studies. We will apply the Importance Sampling method to estimate Greeks. We will focus on the vega and delta of European Call. First, let us start with a basic presentation of the different methods used in order to compute greeks. We will also present their advantages and drawbacks. Greeks are an essential tool in risk analysis for financial derivatives. We will here present several methods used to compute them. We will also indicate their advantages and disadvantages.

## V.1 Finite Difference Method

Let us start with the simplest and more intuitive method. We denote $\alpha(\theta) = \mathbb{E}[P(\theta)]$, then finite difference approximations is used to compute the different greeks by the following:

$$\hat{\Delta}_\theta = \frac{\alpha(\theta+h) - \alpha(\theta-h)}{2h} = \frac{d\alpha}{d\theta}(\theta) + O(h^2)$$

Let us now focus on a discontinuous payoff. We will consider for instance a digital call that we introduced in section I.2. It is quite straightfoward to see from payoff equation -equation (5)- that we have:

$$\mathbb{E}[\hat{\Delta}_\theta] = O(1)$$

$$\mathbb{V}[\hat{\Delta}_\theta] = O(\frac{1}{h})$$

Thus, we have the following problem in the case of a discontinuous payoff:

- small h gives a large variance;
- large h gives a large finite difference discretisation error;

Hence, even though this is a very easy/popular approach, it has some weaknesses such as:

- biased estimator;
- difficulties if discontinuous payoff;

- expensive computation (double simulation);
- machine roundoff error in case of small h.

## V.2 Pathwise Sensitivity approach

The Pathwise Sensitivity method can be computed under some sufficient conditions such as:

$$\left\{ \begin{array}{c} |f(x) - f(y)| \leq K_f \, |x - y| \\ |S_T(\theta, \omega) - S_T(\theta_0, \omega)| \leq |\theta - \theta_0| \, M(\omega) \\ \mathbb{E}[M] < \infty \end{array} \right\} \Longrightarrow \frac{d}{d\theta} \mathbb{E}[f(S_T)] \Big|_{\theta=\theta_0} = \mathbb{E}\left[ f^{'}(S_T) \, \frac{\partial S_T(\theta, \omega)}{\partial \theta} \Big|_{\theta=\theta_0} \right].$$

Thus, if we use a standard Monte Carlo estimator, as long as the payoff remains differentiable with regards to the asset, we see that we have a working method.

The problem of this method is when we have a discontinuous payoff because standard Monte Carlo approach could lead to incorrect approximation. Payoff smoothing methods are then used in order to approximate greeks with Pathwise Sensitivity approach. Also, we can consider the payoff as a sum of differentiable payoffs and then use the linearity of the method in order to compute the derivative.

Thus we can sum up the advantages of the Pathwise sensitivity approach:

- unlike Likelihood Ratio Method method, Pathwise sensitivity approach does not blow up in variance;
- Pathwise sensitivity can be seen as a limit of finite difference methods;
- This method can easily handle various approximation methods of the SDE: Milstein scheme for instance;
- Payoff smoothing methods can improve the estimator;

Now, the drawbacks of this method are:

- There is a need of a differentiable payoff with regards to the asset;
- Changing the payoff into a sum of differentiable payoffs can trigger some problems (call-spread for instance).

## V.3 Likelihood Ratio Method

The main advantage of this method is that it can be applied for non-smooth payoffs. The idea is to apply the derivative operator on the distribution, not on the payoff itself. In fact, under sufficient conditions such as:

$$\left\{ \begin{array}{c} \mathbb{E}\left[|f(x)|^q\right] < \infty \\ \left| \frac{\partial \log p(x,\theta)}{\partial \theta} \frac{p(x,\theta)}{p(x,\theta_0)} \right| \leq M(x) \\ \mathbb{E}[|M(x)|^r] < \infty \end{array} \right\} \Longrightarrow \frac{d}{d\theta}\mathbb{E}[f(x)]\Big|_{\theta=\theta_0} = \mathbb{E}\left[ f(x)\ \frac{\partial \log p(x,\theta)}{\partial \theta}\Big|_{\theta=\theta_0} \right].$$

Of course, these assumptions work in the case of a differentiable distribution. Let us compute this with a Geometric Brownian Motion with Euler-Maruyama discretisation method -equation (7)-:

$$\log \hat{p}_n = -\log \hat{S}_{(n-1)\Delta t} - \log \sigma - \frac{1}{2}\log\left(2\pi\Delta t\right) - \frac{1}{2}\frac{\left(\hat{S}_{n\Delta t} - \hat{S}_{(n-1)\Delta t}(1+r\Delta t)\right)^2}{\sigma^2 \hat{S}^2_{(n-1)\Delta t}\Delta t}.$$

Thus, if we want to compute vega, which is detailed in section I.1, we have for a Geometric Brownian Motion (section I.1.3), with $Z_n \sim \mathcal{N}(0,1)$ (this is associated to compute $\delta W_n$), we have:

$$\mathbb{V}\left[\frac{\partial}{\partial\sigma}\mathbb{E}\left[f(\hat{S}_T)\right]\right] = \mathbb{V}\left[\left(\sum_n \frac{Z_n^2 - 1}{\sigma}\right) f(\hat{S}_T)\right] = O(\frac{1}{\Delta t}).$$

(note this time that $f$ is the payoff without any discount factor).

This, is a great drawback of Likelihood Ratio Method method, as it tends to explode in terms of variance. Simply we can sum up the advantages of Likelihood Ratio Method:

- can be computed with every payoff as long as they have a smooth variance, and finite variance;
- easily computable with Euler Maruyama methods;

The disadvantages of this method are:

- the $\mathrm{O}(\frac{1}{\Delta t})$ blows up the variance;
- if we consider for instance a Milstein Scheme, the distribution cannot be easily computed;
- the variance is generally higher than Path-wise sensitivity.

*NB: In this paper we only studied the computation of Greeks with Likelihood Ration Method and Pathwise Sensitivity method.*

## V.4 Vega of European Call

We will compare two methods of approximation of the greeks: Likelihood Ratio Method and Pathwise Sensitivity method in the case of the MC and the MLMC approximation. Here, we are computing vega.

The vega of a portfolio is the sensitivity of its value from the volatility $\sigma$ of the underlying. The formula is:

$$\nu = \frac{\partial \Pi}{\partial \sigma}. \tag{18}$$

Under Black & Scholes assumption, when we consider an asset as a Geometric Brownian Motion, we then have under Black adn Scholes:

$$\begin{aligned} \nu_{call} &= S_0\sqrt{T}N'(d_1) \\ d_1 &= \frac{\ln(S_0/K)+(r+\frac{\sigma^2}{2})T}{\sigma\sqrt{T}} \end{aligned} \tag{19}$$

### V.4.1 Monte Carlo with Likelihood Ratio Method

The estimator we are using is:

$$\widetilde{\nu}_{call} = \frac{1}{N} \sum_i \exp(-rT) \max\left(S_T^{(i)} - K; 0\right) \left(\frac{(W_T^2 - T)}{\sigma T} - W_T\right). \tag{20}$$

Figure 20 then shows the comparison between MC 'on' and MC 'off' in the case of the computation of the vega a European Call with LRM.

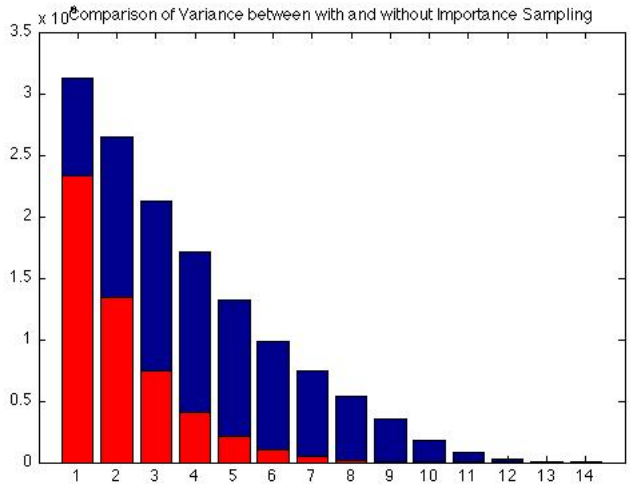



**Figure20:** MC approximation of Vega using Likelihood Ratio Method. grey: MC 'on', black: MC 'off'. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

Figure 20 shows that Importance Sampling still improves significantly the computation of vega in the case of a Standard Monte Carlo method.

### V.4.2 Monte Carlo with Pathwise Sensitivity

The estimator we are using is given by:

$$\widetilde{\nu}_{call} = \frac{1}{N} \sum_i \exp(-rT) \frac{1}{2} (1 + sign(S_T - K)) S_T(W_T - \sigma T). \tag{21}$$

Figure 21 then shows the comparison between MC 'on' and MC 'off' in the case of the computation of the vega a European Call.

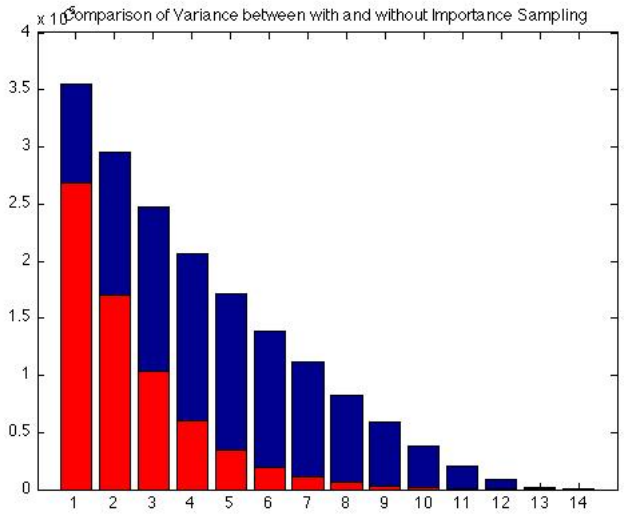



**Figure21:** MC approximation of Vega using Pathwise sensitivity. grey: MC 'on', black: MC 'off'. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

### V.4.3 MLMC with Likelihood Ratio Method

Figure 22 shows the evolution of the $\varepsilon^2.Computational\,Cost$ for the four different methods.

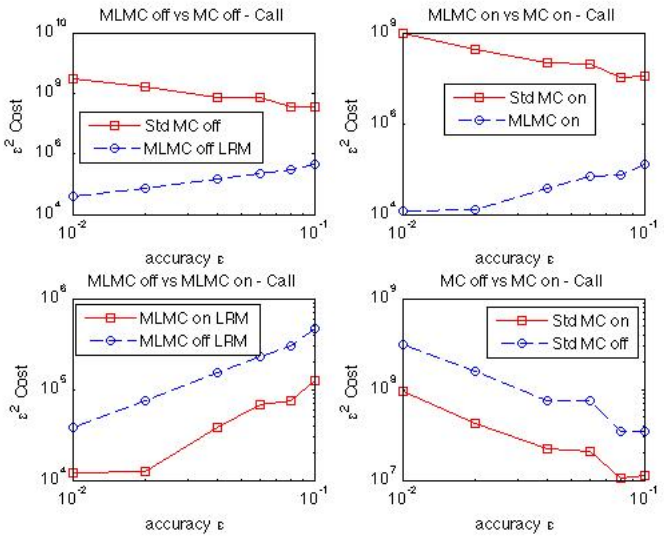



**Figure22:** Computational Cost comparison LRM of vega of a European Call. 1) MLMC 'off' vs MC 'off'; 2) MLMC 'on vs MC 'on'; 3) MLMC 'off' vs MLMC 'on'; 4) MC 'off' vs MC 'on'. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 40$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

### V.4.4 MLMC with Pathwise Sensitivity

Figure 23 shows the Computational Cost comparison in the case of Pathwise Sensitivity.

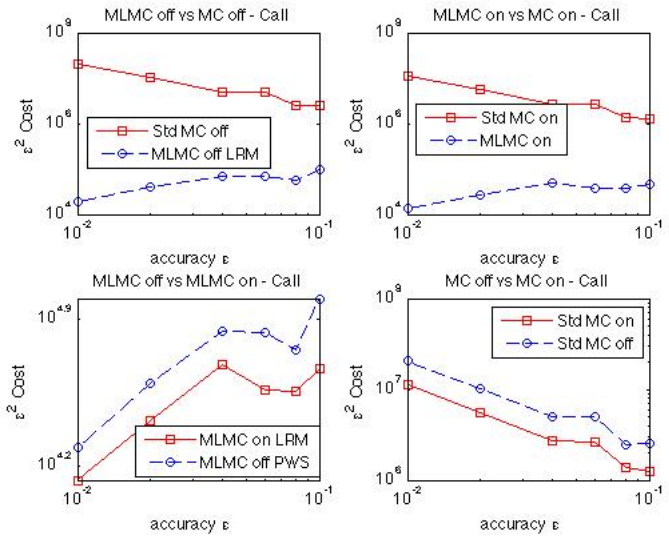



**Figure23:** Computational Cost comparison. 1) MLMC 'off' vs MC 'off'; 2) MLMC 'on vs MC 'on'; 3) MLMC 'off' vs MLMC 'on'; 4) MC 'off' vs MC 'on'. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 40$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

Let us give a quick general analysis of the different graphs we have. In the top graphs, we observe that standard Monte Carlo method has a *Computational Cost* in $\mathcal{O}(\varepsilon^{-3})$. This is the reason why we obtain this decreasing linear behavior. Once again, for MLMC, we obtain a roughly constant that shows a $\mathcal{O}(\varepsilon^{-2})$ *Computational Cost*.

Now, if we have a look to below graphs, we see first that using Importance Sampling will not affect the shape of the *Computational Cost* since we can see how 'parrallel' are the two curves in these two figures. Also, we can denote that the 'on' curves, both in Standard Monte Carlo and in the Multilevel Monte Carlo which indicates Importance Sampling clearly diminishes this *Computational Cost*.

### V.4.5 MLMC Likelihood Ratio Method vs MLMC Pathwise Sensitivity

Figure 24 shows a comparison between Pathwise sensitivity and Likelihood Ratio Method:

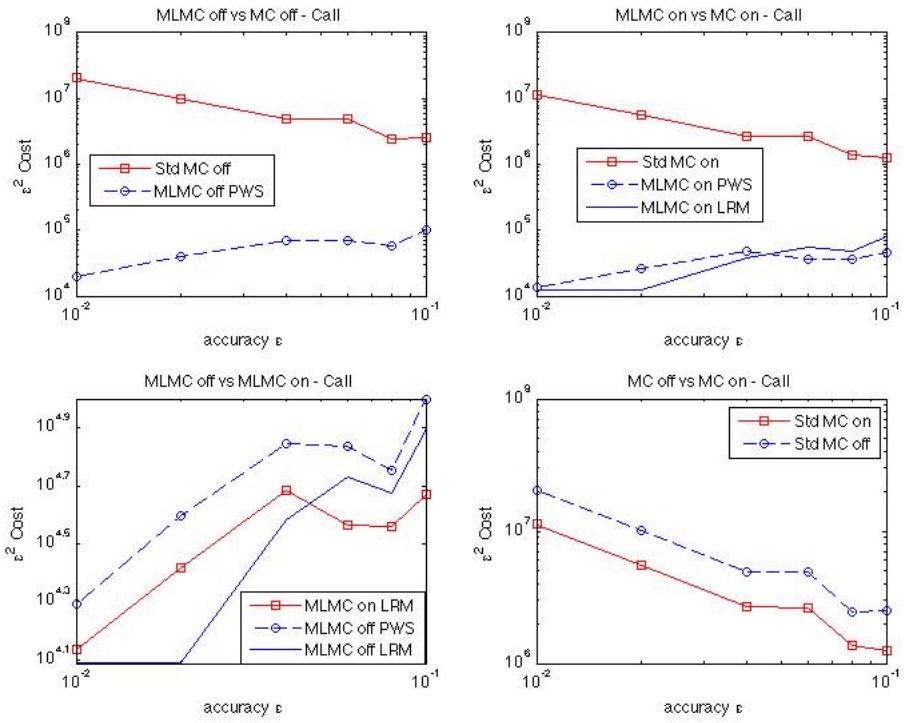



**Figure24:** Comparison between Likelihood Ratio Method and Pathwise sensitivity - vega European Call. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 40$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

Figure 24 shows that the two methods are equivalent and both outperfoms the MLMC and MC 'off'.

## V.5 Delta of European Call

We had a look at a continuous payoff in term of $\sigma$ so that the derivative is easily computable. Now let us focus on discontinuous Greeks. For instance, let us focus on the delta of a Standard European Call.

First, the delta of a portfolio is the sensitivity of the value of the portfolio from its starting value. The formula where $P$ is the value of the portfolio and S the starting value is:

$$\Delta = \frac{\partial P}{\partial S}. \tag{22}$$

Under Black & Scholes assumption, the delta of a European Call is:

$$\begin{aligned} \Delta_{call} &= N(d_1) \\ d_1 &= \frac{\ln(S_0/K)+(r+\frac{\sigma^2}{2})T}{\sigma\sqrt{T}} \end{aligned} \tag{23}$$

### V.5.1 Monte Carlo with Likelihood Ratio Method

The estimator we are using is:

$$\widetilde{\Delta}_{call} = \frac{1}{N} \sum_i \exp(-rT) \max\left(S_T^{(i)} - K; 0\right) \left(\frac{W_t}{S_0 \sigma T}\right). \tag{24}$$

Figure 25 then shows the comprison between MC 'on' and MC 'off' in the case of the computation of the delta a European Call.

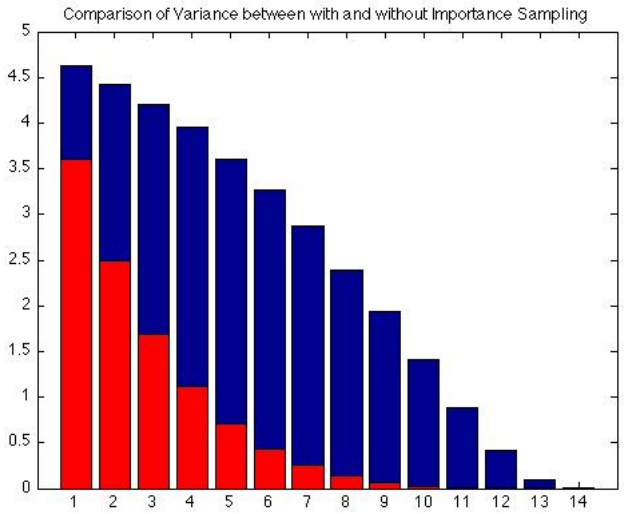


**Figure25:** MC approximation of Delta using Likelihood Ratio Method. grey: MC 'on', black: MC 'off'. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$ and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

We can observe a clear improvement of the variance.

### V.5.1 Monte Carlo with Pathwise Sensitivity

The estimator we are using is:

$$\widetilde{\Delta}_{call} = \frac{1}{N} \sum_i \exp(-rT) \frac{1}{2} (1 + sign(S_T - K)) \frac{S_T}{S_0}. \tag{25}$$

Figure 25 then shows the comprison between MC 'on' and MC 'off' in the case of the computation of the delta a European Call.

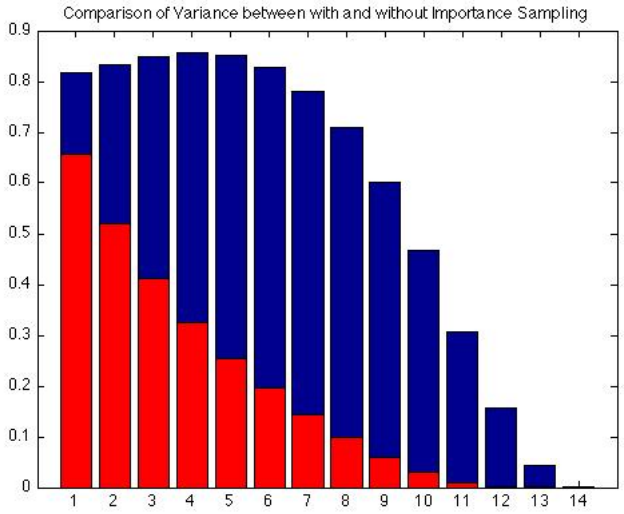



**Figure26:** MC approximation of Delta using Pathwise sensitivity. grey: MC 'on', black: MC 'off'. $S_0 = \{140; 130; 120; 110; 100; 90; 80; 70; 60; 50; 40; 30; 20; 10\}$. Those results have been obtained with $r = 0.05$, $\sigma = 0.2$, $K = 200$

and $T = 10$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Digital Call is described in I.2.

### V.5.3 MLMC with Likelihood Ratio Method

Figure 27 shows the comparison of the four Monte Carlo techniques in the case of Likelihood Ratio Method.

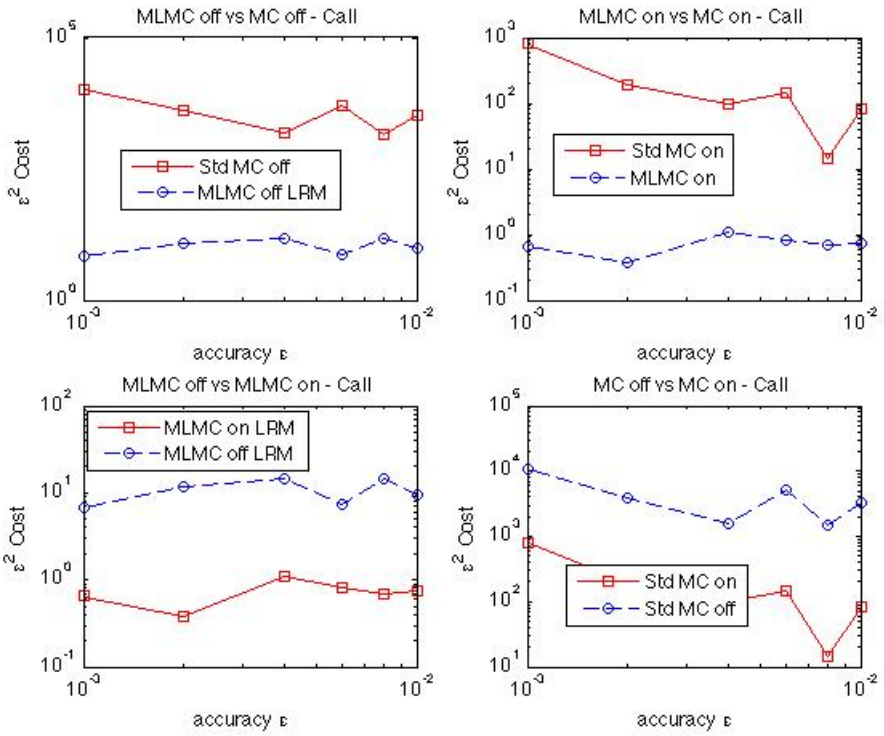



**Figure27:** Likelihood Ratio Method: Computational Cost of Delta approximation: comparison. 1) MLMC 'off' vs MC 'off'; 2) MLMC 'on vs MC 'on'; 3) MLMC 'off' vs MLMC 'on'; 4) MC 'off' vs MC 'on'. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 40$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

The results are consistent with our studies, the Importance Sampling works with LRM when we estimate delta.

### V.5.4 MLMC with Pathwise Sensitivity

Figure 28 shows the comparison of the four Monte Carlo techniques with Pathwise Sensitivity.

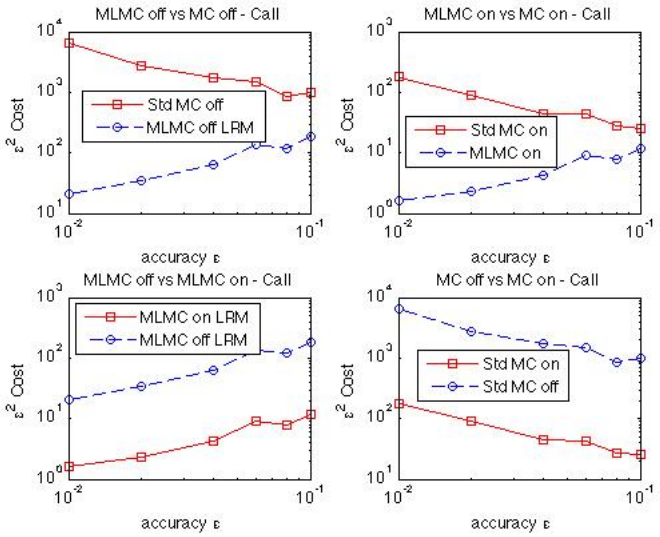



**Figure28:** Pathwise sensitivity: Computational Cost of Delta approximation: comparison. 1) MLMC 'off' vs MC 'off'; 2) MLMC 'on vs MC 'on'; 3) MLMC 'off' vs MLMC 'on'; 4) MC 'off' vs MC 'on'. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 40$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

### V.5.5 MLMC with Likelihood Ratio Method vs MLMC Pathwise Sensitivity

Figure 29 shows the comparison between Pathwise Sensitivity and Likelihood Ratio Method.

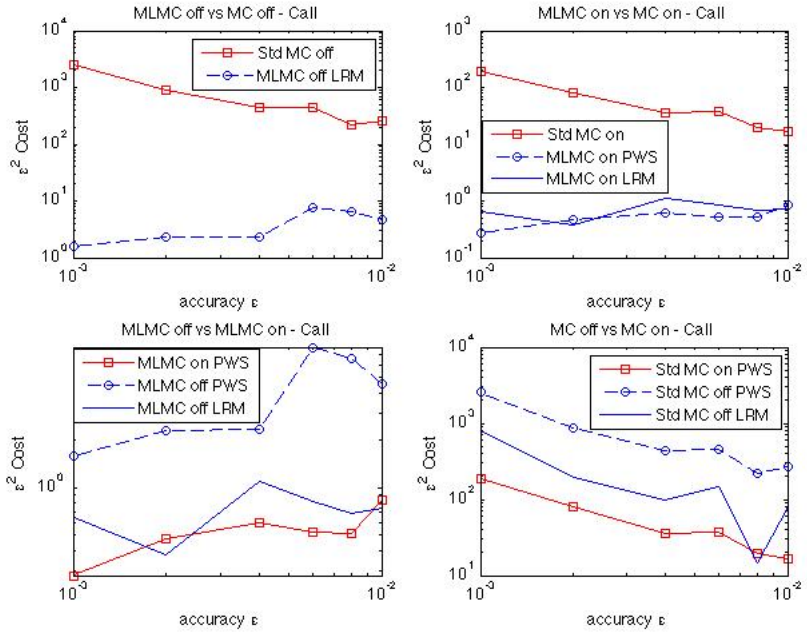



**Figure29:** Comparison between Likelihood Ratio Method and Pathwise sensitivity. Delta - European Call. We have the parameters: $T = 3$, $K = 100$, $r = 0.05$ and $\sigma = 0.20$, $S_0 = 40$ and $\varepsilon = [0.001, 0.002, 0.004, 0.006, 0.008, 0.01]$. $S_t$ follows a Geometric Brownian Motion. We used a Milstein approximation for the underlying -equation (8)-. Call is described in I.2.

# Conclusion:

Let us sum up the studies we have done in this thesis. We combined Multilevel Monte Carlo method with Importance Sampling.

In order to do that, we developed an appropriate change of measure that does not violate the telescopic sum of the MLMC. We tested this change of measure on standard Monte Carlo simulation. The obtained result were very promising.

In the case of rare events simulation, the Monte Carlo estimator has significantly smaller variance. We used that same change of measure with MLMC to further reduce the variance and therefore decrease the computation complexity of our estimator. We tested this idea on pricing European and Digital Call as well as Greeks for European Calls.

Our studies clearly demostrate that MLMC method combined with Importance Sampling outperforms standard approach of simulating financial derivatives for rare events. This might have profound consequences in many branches of financial engineering such as risk analysis.

# References


[1] M. B. Giles, 2008, 'Multilevel Monte Carlo path simulation', Operation Research, 56(3):607-617

[2] P. Glasserman, 2004, 'Monte Carlo Methods in Financial Engineering', Springer, New York

[3] M.B. Giles, 2009, 'Vibrato Monte Carlo sensitivities' in 'Monte Carlo and Quasi-Monte Carlo Methods', Springer, New York

[4] S. Burgos, M.B. Giles, 2011, 'Computing Greeks using multilevel path simulation', Technical report, in L. Plaskota and H. Wozniakowski, editors, Monte Carlo and Quasi-Monte Carlo Methods 2010. Springer-Verlag, 2012.

[5] M. B. Giles, 2006, 'Improved multilevel Monte Carlo convergence using the Milstein Scheme', Tecnical Report, Oxford University Computing Laboratory, Parks Road, Oxford, U.K.

[6] P. L'Ecuyer, 2004, 'Quasi-Monte Carlo methods in Finance', in R.G Ingalls, M.D. Rossetti, J.S. Smith and B.A. Peters, editors, Proceedings of the 2004 Winter Simulation Conference, pages 1645-1655. IEEE Press, 2004.

[7] S. Asmussen & P. Glynn, 2007, 'Stochastic Simulation', in Springer, New York, Volume 57, 2007, DOI: 10.1007/978-0-387-69033-9

[8] L. Szpruch & M. B. Giles, 2012, 'Multilevel Monte Carlo methods for Application in Finance', Tecnical report

[9] S. Heinrich, 2001, 'Multilevel Monte Carlo Methods', volume 2179 of Lecture Notes in Computer Science, pages 58-67. Springer-Verlag, New-York.

# Annexes:

## Annex 1: Monte Carlo Simulation with an without Importance Sampling

First Part:

```
clear all;
clc

S0 = 140: -10: 10;
T = 10;
K = 200;
r = 0.05;
sigma = 0.2;

expected_value = zeros(1,length(S0));
variance_on  = zeros(1,length(S0));
variance_off = zeros(1,length(S0));
error_on = zeros(1,length(S0));
error_off = zeros(1,length(S0));
mse_on = zeros(1,length(S0));
mse_off = zeros(1,length(S0));
expected_value_on = zeros(1,length(S0));
expected_value_off = zeros(1,length(S0));

for j = 1:length(S0);
    j
    value_off = mc_payoff(r, sigma, S0(j), T, K, 10^6, ...
                          'call_european', 'off', 'vega');
    value_on = mc_payoff(r, sigma, S0(j), T, K, 10^6, ...
                         'call_european', 'on', 'vega');
    value = european_call_value(r,sigma,T,S0(j),K,'vega');
    expected_value(j) = value;
    error_off(j) =  error_off(j) + abs(value_off(1) - value);
    error_on(j) = error_on(j) + abs(value_on(1) - value);
    expected_value_off(j) =  value_off(1);
    expected_value_on(j) = value_on(1);
    mse_off(j) = mse_off(j) + (value_off(1) - value)^2;
    mse_on(j) = mse_on(j) + (value_on(1) - value)^2;
    variance_off(j) = variance_off(j) + value_off(2);
    variance_on(j) = variance_on(j) + value_on(2);
end;
```

```
bar( variance_off)
hold on
bar( variance_on, 'r')
title('Comparison of Variance between with and without Importance Sampling');

%{
subplot(221);
bar( variance_off)
hold on bar( variance_on, 'r')
title('Comparison of Variance between with and without ...
Importance Sampling');

subplot(222);
plot(error_off)
hold on
plot( error_on, 'r')
title('Comparison of Error between with and without ...
Importance Sampling');

subplot(223);
plot( mse_off)
hold on
plot( mse_on, 'r')
title('Comparison of MSE between with and without ...
Importance Sampling');

subplot(224);
plot(expected_value_off)
hold on
plot(expected_value_on, 'r')
plot( value, '--g')
title('Comparison of Estimated Payoff between with ...
and without Importance Sampling');
%}
```

Second Part:

```
function [ value ] = mc_payoff( r, sigma, S0, T, K, N_simu, ...
                                payoff, opt, opt_g )
%UNTITLED5 Summary of this function goes here
%   Detailed explanation goes here

payoff = inline([payoff, '( K, S, S0, T, opt_g, r)']);
N_pas = 10;

switch opt
    case 'on'
        Dt = T/N_pas;
        S_T = S0*ones(1,N_simu);
        plotage = ones(N_pas, N_simu);
        %S_T_1 = S0*ones(1,N_simu);
        delta_K = 5;
        mu = r - sigma^2/2;

        A=[ T sqrt(2*T*log(log(T))); T -sqrt(2*T*log(log(T))) ];
        Y = A^(-1)*[log((K+delta_K)/S0) log((K-delta_K)/S0)]';

        mu_2 = (1/T)*log(K/S0); %Y(1);
        sigma_2 = sigma; %Y(2);

        %mu_3 = Y(1);
        %sigma_3 = Y(2);

        if strcmp(opt_g, 'value') == 1
          for i = 1:N_pas;
            DW = sqrt(Dt).*randn(1, N_simu);
            S_T = S_T.*(1 + mu_2*Dt + sigma_2*DW + ...
                  (sigma_2^2/2)*(DW).^2 );
            plotage(i,:) = S_T;
            %S_T_1 = S_T_1.*(1 + mu_3*Dt + ...
            sigma_3*DW + (sigma_3^2/2)*(DW).^2 );
          end;
        else
          DW = sqrt(T).*randn(1, N_simu);
          S_T = S_T.*exp( mu_2*T + sigma_2*DW );
        end;

        X = log(S_T/S0);
        R = (sigma_2/sigma)*exp( - 0.5*((X-mu*T)/(sigma*sqrt(T))).^2 ...
            + 0.5*((X-mu_2*T)/(sigma_2*sqrt(T))).^2 );
```

```
        %{

        X_1 = log(S_T_1/S0);
        R01 = exp( (1/(2*(sigma*sqrt(T))^2))* ...
              ( -(X_1-mu).^2+(X_1-mu_3).^2 ) );

        R12 = (sigma_3/sigma)*exp( -0.5*(1/(sigma*sqrt(T))^2 ...
              - 1/(sigma_3*sqrt(T))^2)*(X_1-mu_2).^2 );
        R02 = 0.5 *( (R01+R12).^2 - R01.^2 -R12.^2 )
        a = (sigma_3/sigma)*exp( - 0.5*((X-mu*T)/(sigma*sqrt(T))).^2
...
        + 0.5*((X-mu_3*T)/(sigma_3*sqrt(T))).^2 )
        P_1 =  R02.*payoff(S_T_1, K, r, T);
        value_1 = sum(P_1)/N_simu

        %}

        P = R.*payoff(S_T, K, r, T, opt_g, S0);
        value = zeros(1,2);
        value(1) = sum(P)/N_simu;
        value(2) = sum(P.^2)/N_simu - (value(1))^2;


        %{
        figure();
        for i = 1:N_simu
            hold all;
            plot(plotage(:,i))
        end
        %}

    case 'off'
      mu = r-sigma^2/2;
      Dt = T/N_pas;
      S_T = S0*ones(1,N_simu);

      if strcmp(opt_g, 'value') == 1
        for i = 1:N_pas;
          DW = sqrt(Dt).*randn(1, N_simu);
          S_T = S_T.*(1 + mu*Dt + sigma*DW + sigma^2/2*(DW).^2 );
        end;
      else
        DW = sqrt(T).*randn(1, N_simu);
        S_T = S_T.*exp( mu*T + sigma*DW );
      end;
```

```
        P = payoff(S_T, K, r, T, opt_g, S0);

        value = zeros(1, 2);
        value(1) = sum(P)/N_simu;
        value(2) = sum(P.^2)/N_simu - (value(1))^2;

end
end
```

## Annex 2: MLMC code

First Part:

```
clear all
format 'long';
clc

%Initial Value
S0 = 100;
T = 3;
K = 100;
r = 0.05;
sigma = 0.20;
M = 2;
eps = [ 0.001 0.002 0.004 0.006 0.008 0.01 ];

mlmc_on = zeros(1, length(eps));
mlmc_off = zeros(1, length(eps));
mc_on = zeros(1, length(eps));
mc_off = zeros(1, length(eps));

value_call = zeros(1, length(S0));
value_on = zeros(1, length(S0));
variance_on = zeros(1, length(S0));
value_off = zeros(1, length(S0));
variance_off = zeros(1, length(S0));

for i = 1:length(eps)
    i
    value_call(i) = european_call_value(r,sigma,T,S0,K,'vega');
    off = mlmc_payoff_eps(r,sigma,S0,T,K,eps(i),M, ...
          'call_european', 'off', 'vega');
    value_off(i) = off(1);
    variance_off(i) = off(2);
    mlmc_off(i) = off(3);
    mc_off(i) = off(4);
    on  = mlmc_payoff_eps(r,sigma,S0,T,K,eps(i),M, ...
          'call_european', 'on', 'vega');
    value_on(i) = on(1);
    variance_on(i) = on(2);
    mlmc_on(i) = on(3);
    mc_on(i) = on(4);
end;
```

```
subplot(221)
loglog(eps,eps.^2.*mc_off(:),'rsq-', eps,eps.^2.*mlmc_off(:),'bO--')
xlabel('accuracy \epsilon');
ylabel('\epsilon^2 Cost');
legend('Std MC off','MLMC off',1)
title('MLMC off vs MC off - Call')

subplot(222)
loglog(eps,eps.^2.*mc_on(:),'rsq-', eps,eps.^2.*mlmc_on(:),'bO--')
xlabel('accuracy \epsilon');
ylabel('\epsilon^2 Cost');
legend('Std MC on','MLMC on', 1)
title('MLMC on vs MC on - Call')

subplot(223)
loglog(eps,eps.^2.*mlmc_on(:),'rsq-', eps,eps.^2.*mlmc_off(:),'bO--')
xlabel('accuracy \epsilon');
ylabel('\epsilon^2 Cost');
legend('MLMC on Likelihood Ratio Method','MLMC off Likelihood Ratio Method',1)
title('MLMC off vs MLMC on - Call')

subplot(224)
loglog(eps,eps.^2.*mc_on(:),'rsq-', eps,eps.^2.*mc_off(:),'bO--')
xlabel('accuracy \epsilon');
ylabel('\epsilon^2 Cost');
legend('Std MC on','Std MC off',1)
title('MC off vs MC on - Call')

%{
figure()
plot(value_call)
hold all
plot(value_off, 'g')
hold all
plot(value_on, 'r')
title('Value of the option')

figure()
bar(variance_off)
hold all;
bar(variance_on, 'r')
title('Variance of the estimator')
%}
```

Second Part:

```
function [ Y_l ] = mlmc_level_payoff(r, sigma, S0, T, K, N, M, ...
                   payoff, l, opt, r_2, sigma_2, opt_g)
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here

payoff = inline([payoff, '( K, S, S0, T, opt_g, r)']);

Y_l = ones(1,4);

switch opt
   case 'on'
     mu = r - sigma^2/2;
     mu_2 = r_2 - sigma_2^2/2;

     if l==0
       %je génère les paths jusqu'à maturité
       DW_0 = sqrt(T).*randn(1, N);
       DD_0 = 1 + (mu_2*T).*ones(1,N)+sigma.*DW_0+(sigma^2/2).*(DW_0.^2);
       S_T = S0.*DD_0;

       %je fais le changement de variable
       X = log(S_T/S0);
       R = (sigma_2/sigma)*exp( - 0.5*((X-mu*T)/(sigma*sqrt(T))).^2 ...
            + 0.5*((X-mu_2*T)/(sigma_2*sqrt(T))).^2 );

       P = R.*payoff( S_T, K, r, T, opt_g, S0);
       P_f = P;

     else
       Dt = T/M^l;
       S_f = S0;
       S_c = S0;

       for n=1:M^(l-1)
         DW_f = sqrt(Dt).*randn(M, N);
         DW_c = DW_f(1, 1:1:end)+DW_f(2, 1:1:end);
         DD_f = 1+(mu_2*Dt).*ones(2,N)+sigma_2.*DW_f+ ...
                (sigma_2^2/2).*(DW_f.^2);
         DD_c = 1+(mu_2*2*Dt).*ones(1,N)+sigma_2.*DW_c+ ...
                (sigma_2^2/2).*(DW_c.^2);

         S_f = S_f.*prod(DD_f(:,1:1:end));
         S_c = S_c.*DD_c;
```

```
    end;

    X_1 = log( S_f/S0 );
    R_1 = (sigma_2/sigma)*exp(-0.5*((X_1-mu*T)/(sigma*sqrt(T))).^2...
          +0.5*((X_1-mu_2*T)/(sigma_2*sqrt(T))).^2 );

    X_2 = log( S_c/S0 );
    R_2 = (sigma_2/sigma)*exp(-0.5*((X_2-mu*T)/(sigma*sqrt(T))).^2...
          +0.5*((X_2-mu_2*T)/(sigma_2*sqrt(T))).^2 );

    P_f = payoff( S_f, K, r, T, opt_g, S0);
    P_c = payoff( S_c, K, r, T, opt_g, S0);

    P = R_1.*P_f - R_2.*P_c;
  end

  %mlmc
  Y_l(1) = sum(P);
  Y_l(2) = sum(P.^2);

  %mc
  Y_l(3) = sum(P_f);
  Y_l(4) = sum(P_f.^2);

case 'off'
  if l==0
    DW_0 = sqrt(T).*randn(1, N);
    DD_0 = 1+((r-sigma^2/2)*T).*ones(1,N)+sigma.*DW_0...
           +(sigma^2/2).*(DW_0.^2);

    P = payoff( S0.*DD_0, K, r, T, opt_g, S0);
    sum(P);
    P_f = P;
  else
    Dt = T/M^l;
    S_f = S0;
    S_c = S0;

    for n=1:M^(l-1)
      DW_f = sqrt(Dt).*randn(M, N);
      DW_c = DW_f(1, 1:1:end)+DW_f(2, 1:1:end);

      DD_f = 1+((r-sigma^2/2)*Dt).*ones(2,N)...
             +sigma.*DW_f+(sigma^2/2).*(DW_f.^2);
      DD_c = 1+((r-sigma^2/2)*2*Dt).*ones(1,N)...
             +sigma.*DW_c+(sigma^2/2).*(DW_c.^2);
```

```
            S_f = S_f.*prod(DD_f(:,1:1:end));
            S_c = S_c.*DD_c;
        end;

        P_f = payoff( S_f, K, r, T, opt_g, S0);
        P_c = payoff( S_c, K, r, T, opt_g, S0);

        P = P_f - P_c;
    end

    %mlmc
    Y_l(1) = sum(P);
    Y_l(2) = sum(P.^2);

    %mc
    Y_l(3) = sum(P_f);
    Y_l(4) = sum(P_f.^2);

end

end
```

Third Part:

```
function [ value ] = mlmc_payoff_eps(r, sigma, S0, T, K, eps, M,...
                      payoff, opt, opt_g)
%UNTITLED5 Summary of this function goes here
%   Detailed explanation goes here

L_max = ceil(-log(eps)/log(M));

switch opt
    case 'on'
      value = zeros(1, 4);
      delta_K = 5;

      A=[ T sqrt(2*T*log(log(T))); T -sqrt(2*T*log(log(T))) ];
      Y = A^(-1)*[log((K+delta_K)/S0) log((K-delta_K)/S0)]';

      mu_2 = Y(1);
      sigma_2 = sigma; %Y(2);
      r_2 = mu_2+(sigma_2^2)/2;

      'on'
      Nl = Nl_estimator_payoff(r, sigma, S0, T, K, M, eps, ...
           payoff, opt, r_2, sigma_2, opt_g);
      mlmc_cost = (1+1/M)*sum(Nl.*M.^(0:L_max));

      pivot = 0;
      variance = 0;

      for l=1:L_max+1
        val = mlmc_level_payoff(r, sigma, S0, T, K, Nl(l), M, ...
              payoff, l-1, opt, r_2, sigma_2, opt_g);
        x = val(1)/Nl(l);
        pivot = pivot + x;
        variance = variance + (val(2)/Nl(l) - x^2);

        if l == L_max+1
          var_mc = (val(4)/Nl(l) - (val(3)/Nl(l))^2);
        end
      end;
      mc_cost = sum( (2*var_mc/eps^2).*M.^(0:L_max) );

      value(1) = pivot;
      value(2) = variance;
      value(3) = mlmc_cost;
```

```
        value(4) = mc_cost;

    case 'off'
        value = zeros(1, 4);

        'off'
        Nl = Nl_estimator_payoff(r, sigma, S0, T, K, M, eps,...
             payoff, opt, 0, 0, opt_g);
        mlmc_cost = (1+1/M)*sum(Nl.*M.^(0:L_max));

        pivot = 0;
        variance = 0;

        for l=1:L_max+1
          val = mlmc_level_payoff(r, sigma, S0, T, K, Nl(l), M,...
                payoff, l-1, opt, 0, 0, opt_g);
          x = val(1)/Nl(l);
          pivot = pivot + x;
          variance = variance + (val(2)/Nl(l) - x^2);
          if l == L_max+1
            var_mc = (val(4)/Nl(l) - (val(3)/Nl(l))^2);
          end
        end;

        mc_cost = sum( (2*var_mc/eps^2).*M.^(0:L_max) );

        value(1) = pivot;
        value(2) = variance;
        value(3) = mlmc_cost;
        value(4) = mc_cost;
end

end
```