SORBONNE UNIVERSITY

NESTED MONTE CARLO

NUMERICAL PROBABILITY

Antoine Maio, Dimitri Sotnikov

May 4, 2024

CHAPTER 1

INTRODUCTION

The goal of our project was to study and implement five nested Monte Carlo methods proposed in the article Broadie, Du, and Moallemi 2011.

The problem we are facing is the estimation of the loss probability:

$$\alpha = \mathbb{P}(L \geq c)$$

for a given threshold $c \in \mathbb{R}$. Here, the loss L is a random variable the distribution of which depends on the realized scenario $\omega \in \Omega$, which is a random variable itself. Hence, two levels appear in the Monte Carlo simulation:

1. **Outer level**: simulate the scenarios $\omega_1, \ldots, \omega_n$.
2. **Inner level**: for each scenario ω_i, simulate the loss values $Z_{i,1}, \ldots, Z_{i,m_i}$ and estimate $\hat{L}(\omega_i)$:

$$\hat{L}(\omega_i) = \frac{1}{m_i} \sum_{j=1}^{m_i} Z_{i,j}.$$

3. Estimate the probability of the loss L exceeding c by

$$\hat{\alpha} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\hat{L}(\omega_i) \geq c}.$$

The following questions arises:

- *Is it possible to speed up a naive algorithm to make it numerically stable and convergent faster?*

- *Given a computation budget $k = \sum_{i=1}^{n} m_i$, how to choose optimally the values of n and $m_i,\ i = 1, \ldots, n$ to minimize the MSE of the estimator $\hat{\alpha}$?*
- *Do the new estimators converge?*
- *What can we say about their asymptotic behaviour?*

In section 2 we review the algorithms proposed by the authors and describe the motivation behind them. In section 3, we provide numerical results for two different examples (Gaussian loss and loss related to the selling of vanilla put option) and discuss the possibilities to introduce variance reduction. Section 4 is devoted to the discussion of the observed numerical results.

CHAPTER 2

METHODS

The following methods were proposed in Broadie, Du, and Moallemi 2011 and implemented within this project:

1. **Uniform sampling 1/3:2/3** (Algorithm 1 in the article): in this case $m_i = m$ does not depend on i, and we have $k = mn$. It was shown by authors that the optimal allocation is attained for $n \propto k^{2/3}$ and $m \propto k^{1/3}$. In this case, the optimal MSE decays as $k^{-2/3}$ which is slower than for standard Monte Carlo simulation. Moreover, the proportionality constant β is very difficult to calculate for real-world problems, so one takes simply $n = k^{2/3}$ and $m = k^{1/3}$.

2. **Optimal uniform sampling** (still Algorithm 1 in the article): For the considered example, the proportionality constant β mentioned in the description of the first method can be calculated explicitly and we take $n = \beta k^{2/3}$ and $m = \dfrac{1}{\beta} k^{1/3}$, with $\beta = \left(\frac{\alpha(1-\alpha)}{2\theta_c^2}\right)^{\frac{1}{3}}$, where

$$\theta_c = -\frac{1}{2}\frac{d}{dc}\left[f(c)\mathbb{E}[\sigma^2(\omega)|L(\omega) = c]\right].$$

   Here $\sigma^2(\omega)$ is the variance of the inner stage samples given the scenario ω. and $f(\cdot)$ denotes the density of the loss variable L. It is clear that evaluation of θ_c is impossible for any practically reasonable problem. Moreover, even if the optimal uniform and the uniform are asymptotically optimal, the choice of β is critical: in the paper, the authors show an order of magnitude of 10 on the MSE depending on the choice of this parameter, and we confirmed this result in 3.2.

3. **Sequential sampling** (Algorithm 2 in the article): The idea of the sequential algorithm is to generate more inner losses for scenarios ω such that $L(\omega)$ is close to c hence the accurate estimation of this value may change the final estimator involving $\mathbb{I}_{\hat{L}(\omega_i)\geq c}$. Thus, at each step one chooses a scenario ω_{i^*} such that the probability to change the value of $\mathbb{I}_{\hat{L}(\omega_i)\geq c}$ after the next loss simulation to be the highest. This is achieved by taking the scenario with the

smallest value of $\frac{m_i}{\sigma_i}|\hat{L}_i - c|$, where σ_i is the volatility of the inner loss given the scenario ω_i (assumed to be known). The parameters of the algorithm are m^0 - minimal value of m_i, $\bar{m}$ - average value of m_i such that $k = n\bar{m}$, n - number of outer scenarios simulations. For the value of n, we took the value obtained by the authors by minimization of the method's MSE. Thus, sequential method will serve as a benchmark other methods to be compared with. The figure 2.1 illustrated the approach described above. The goal is the estimation of α, or the area of the shaded region. There are two portfolio losses $\hat{L}_1$ and $\hat{L}_2$, which distributions are shown with a dashed line. As $\hat{\alpha} \triangleq \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}_{\{\hat{L}_i \geq c\}}$, it is clear that sampling of $\hat{L}_1$ is completely irrelevant, so we want to allocate more budget on m_2 rather than on m_1 in order to estimate more precisely the value of $L(\omega_2)$ that may change the value of indicator in the final estimator.

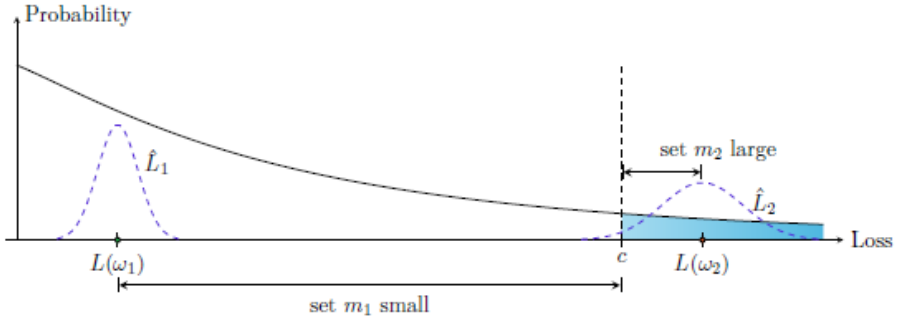


FIGURE 2.1
An illustration of the benefits of non-uniform sampling.

We also find it important to explain where the idea of minimization of $\frac{m_i}{\sigma_i}|\hat{L}_i - c|$ comes from. In fact, suppose that for each scenario ω_i we have already simulated m_i inner samples $\hat{Z}_{i,1},..,\hat{Z}_{i,m_i}$. The resulting loss for this scenario is then given by $\hat{L}_i \triangleq \frac{1}{m_i}\sum_{j=1}^{m_i} \hat{Z}_{i,j}$. Suppose that $\hat{L}_i \geq c$ and we wish to sample one more inner loss $\hat{Z}_{i,m_i+1}$. Then, the loss estimation $\hat{L}_i$ becomes $\hat{L}'_i \triangleq \frac{1}{m_i+1}\sum_{j=1}^{m_i+1} \hat{Z}_{i,j} = \frac{1}{m_i+1}\hat{Z}_{i,m_i+1} + \frac{m_i}{m_i+1}\hat{L}_i$. Since $\hat{\alpha} \triangleq \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}_{\{\hat{L}_i \geq c\}}$, new sample will change its value only if $\hat{L}'_i \leq c$. So, the probability of estimator's change is given by

$$P(\hat{L}_i < c) = P\left(\hat{Z}_{i,m_i+1} - L(\omega_i) < -m_i(\hat{L}_i - c) - (L(\omega_i) - c)\right) \tag{2.1}$$

$$\approx P\left(\hat{Z}_{i,m_i+1} - L(\omega_i) < -m_i|\hat{L}_i - c|\right) \leq \left(1 + \frac{m_i^2}{\sigma_i^2}|\hat{L}_i - c|^2\right)^{-1}. \tag{2.2}$$

This result is obtained by the authors under the assumption that $m_i \gg 1$. In this case, $-m_i(\hat{L}_i - c) - (L(\omega_i) - c) \approx -m_i|\hat{L}_i - c|$, and the application of the one-sided Chebyshev

Inequality gives the result. Hence, the minimization of $\frac{m_i}{\sigma_i}|\hat{L}_i - c|$ leads to the increase of the desired probability upper bound.

4. **Threshold-based method** (Algorithm 3 in the article): This algorithm simulates the inner losses for n outer scenarios one-by-one until the certain value γ of $\frac{m_i}{\sigma_i}|\hat{L}_i - c|$ is attained for each simulated scenario ω_i. This algorithm is not compared with others as it does not allow to control the computation budget k.

5. **Adaptive algorithm** (Algorithm 4 in the article): Neither sequential nor threshold-based method do not propose the way to choose n. This problem is addressed by the adaptive algorithm that starts with simulation of n^0 scenarios and increases this number according to the solution of an optimization problem at the beginning of each epoch of size τ_e. During each epoch, at least m^0 inner losses for each simulated scenario should be generated. Once it is done, the sequential algorithm is applied to already simulated scenarios until the number of inner losses simulations done within the epoch, attains τ_e. This algorithm finds a new optimal couple $(\bar{m}'(= \frac{1}{n}\sum_{i=1}^{n} m_i), n')$ given the current value $(\bar{m}, n)$ at the beginning of each epoch according to the following optimisation problem :

$$\begin{aligned} \text{minimize} \quad & \hat{B}^2 \left(\frac{\bar{m}}{\bar{m}'}\right)^4 + \hat{V}\left(\frac{n}{n'}\right) \\ \text{subject to} \quad & \bar{m}'n' = \bar{m}n + \tau_e, \\ & n \leq n' \leq n + \tau_e, \\ & \bar{m}' \geq 0, \end{aligned} \tag{2.3}$$

where $\hat{B}$ and $\hat{V}$ denote respectively the estimator of the bias and the variance. The multiplier $\left(\frac{\bar{m}}{\bar{m}'}\right)^4$ is justified by the $\bar{m}^{-4+\epsilon}$ asymptotic decreasing behaviour of the bias and the $\left(\frac{n}{n'}\right)$ term comes from the n^{-1} decrease of the variance. Hence, the functional being minimized, is a proxy to $B'^2 + V'$, i.e. the MSE of the final estimator. The first constraint $\bar{m}'n' = \bar{m}n + \tau_e$ ensures that the total number of inner samples equal τ_e. The authors define the bias estimator $\hat{B}$ as:

$$E[\hat{\alpha} - \alpha] \approx B \triangleq \hat{\alpha} - \bar{\alpha}$$

where $\bar{\alpha}$ is an approximation based on the central limit theorem. Namely, for large m_i, we suppose that $\hat{L}_i \sim \mathcal{N}\left(L(\omega_i), \frac{\sigma_i^2}{m_i}\right)$ and define

$$\bar{\alpha} \triangleq \frac{1}{n}\sum_{i=1}^{n} \Phi\left(\frac{m_i(\hat{L}_i - c)}{\sigma_i}\right)$$

So, the bias reads

$$\mathbb{E}[\hat{\alpha} - \alpha] = \frac{1}{n}\sum_{i=1}^{n}\left(P(\hat{L}_i \geq c) - \mathbb{I}_{\{L(\omega_i)\geq c\}}\right) \approx \frac{1}{n}\sum_{i=1}^{n}\left(\Phi\left(\frac{m_i(L(\omega_i) - c)}{\sigma_i}\right) - \mathbb{I}_{\{L(\omega_i)\geq c\}}\right)$$

By the similar heuristic, the variance estimator is defined as a variance of a Bernoulli-distributed random variable:

$$\text{Var}(\hat{\alpha}) \approx \hat{V} \triangleq \frac{\hat{\alpha}(1-\hat{\alpha})}{n}$$

The solution to (2.3) is explicit:

$$n' = \min\left(\max\left(\left(\frac{\hat{V}n}{4\hat{B}^2\bar{m}^4}(\bar{m}n+\tau_e)^4\right)^{1/5}, n\right), n+\tau_e\right), \quad \bar{m}' = \frac{mn+\tau_e}{n'}. \quad (2.4)$$

Below, the pseudo-code for each of the presented algorithms is provided.

6. **Adaptive algorithm with volatility estimation** (still Algorithm 4 in the article): In all the algorithms we considered, σ_i were supposed to be known explicitly, which is not always true in practice. In this algorithm, we estimate the volatility from the sample for simulated inner losses and use $\hat{\sigma}_i$ instead of exact value σ_i for each scenario.

Algorithm 1 Estimate the probability of a large loss using a uniform nested simulation.

procedure UNIFORM(m, n)
  for $i \leftarrow 1$ to n **do**
    generate scenario ω_i
    conditioned on scenario ω_i, generate i.i.d. samples $\hat{Z}_{i,1}, \ldots, \hat{Z}_{i,m}$ of portfolio losses
    compute an estimate of the loss in scenario ω_i, $\hat{L}_i \leftarrow \frac{1}{m}\sum_{j=1}^{m} \hat{Z}_{i,j}$
  end for
  compute an estimate of the probability of a large loss, $\hat{a} \leftarrow \frac{1}{n}\sum_{i=1}^{n} 1_{\{\hat{L}_i \geq c\}}$
  return $\hat{a}$
end procedure

Algorithm 2 Estimate the probability of a large loss using a sequential non-uniform nested simulation.

procedure SEQUENTIAL($m^0, \bar{m}, n$)
  for $i \leftarrow 1$ to n **do**
    generate scenario ω_i
    conditioned on scenario ω_i, generate i.i.d. samples $\hat{Z}_{i,1}, \ldots, \hat{Z}_{i,m^0}$ of portfolio losses
    $m_i \leftarrow m^0$
  end for
  while $\sum_{i=1}^{n} m_i < \bar{m}n$ **do**
    $i^* \leftarrow \arg\min_i m_i \left|\hat{L}_i - c\right| / \sigma_i$, where for each $1 \leq i \leq n$, $\hat{L}_i$ is the current estimate of the loss in scenario ω_i, $\hat{L}_i \leftarrow \frac{1}{m_i} \sum_{j=1}^{m_i} \hat{Z}_{i,j}$, and σ_i is the standard deviation of the distribution of losses in scenario ω_i
    generate one additional portfolio loss sample $\hat{Z}_{i^*,m_{i^*}+1}$ in scenario ω_{i^*}
    $m_{i^*} \leftarrow m_{i^*} + 1$
  end while
  compute an estimate of the probability of a large loss, $\hat{a} \leftarrow \frac{1}{n} \sum_{i=1}^{n} 1_{\{\hat{L}_i \geq c\}}$
  return $\hat{a}$
end procedure

Algorithm 3 Estimate the probability of a large loss using a threshold-based non-uniform nested simulation.

procedure THRESHOLD(γ, n)
  for $i \leftarrow 1$ to n **do**
    generate scenario ω_i
    set σ_i to be the standard deviation of the distribution of the losses in scenario ω_i
    $m_i \leftarrow 0$
    repeat
      generate one additional portfolio loss sample $\hat{Z}_{i,m_i+1}$ in scenario ω_i
      $m_i \leftarrow m_i + 1$
      compute an estimate of the loss in scenario ω_i, $\hat{L}_i \leftarrow \frac{1}{m_i} \sum_{j=1}^{m_i} \hat{Z}_{i,j}$
    until $\frac{m_i}{\sigma_i} \left|\hat{L}_i - c\right| \geq \gamma$
  end for
  compute an estimate of the probability of a large loss, $\hat{a} \leftarrow \frac{1}{n} \sum_{i=1}^{n} 1_{\{\hat{L}_i \geq c\}}$
  return $\hat{a}$
end procedure

Algorithm 4 Estimate the probability of a large loss using an adaptive non-uniform nested simulation.

procedure ADAPTIVE(m^0, n^0, τ_e, κ)
  generate scenarios $\omega_1, \omega_2, \ldots, \omega_{n^0}$
  $n \leftarrow n^0$
  for $i \leftarrow 1$ to n^0 **do**
    conditioned on scenario ω_i, generate i.i.d. samples $\hat{Z}_{i,1}, \ldots, \hat{Z}_{i,m^0}$ of portfolio losses
    $m_i \leftarrow m^0$
  end for
  for $\ell \leftarrow 1$ to $\lfloor \frac{n}{\tau_e} \rfloor$ **do**
    estimate the current bias and variance by $\hat{B}$ and $\hat{V}$
    determine a target number of scenarios by
    $n' \leftarrow \lfloor \min\left(\max\left(\left(\frac{\hat{V} n}{4 \hat{B}^2 \bar{m}^4} (\bar{m} n + \tau_e)^4 \right)^{1/5}, n \right), n + \tau_e \right) \rfloor$
  end for
  generate scenarios $\omega_{n+1}, \ldots, \omega_{n'}$, set $m_i \leftarrow 0$ for $i = n+1, \ldots, n'$
  $n \leftarrow n'$
  while $\sum_{i=1}^{n} m_i < \ell \tau_e$ **do**
    if $\min_i m_i < m^0$ **then**
      set $i^* \leftarrow \arg\min_i m_i$
    else
      set $i^* \leftarrow \arg\min_i m_i |\hat{L}_i - c| / \sigma_i$
    end if
    generate one additional portfolio loss sample $\hat{Z}_{i^*, m_{i^*}+1}$ in scenario ω_{i^*}
    $m_{i^*} \leftarrow m_{i^*} + 1$
  end while
  compute an estimate of the probability of a large loss, $\hat{a} \leftarrow \frac{1}{n} \sum_{i=1}^{n} 1_{\{\hat{L}_i \geq c\}}$
  return $\hat{a}$
end procedure

CHAPTER 3

NUMERICAL STUDY

3.1 GAUSSIAN EXAMPLE

We will illustrate the implementation of the algorithms on two examples considered in the article. In the first example, we suppose that scenario is a standard normal random variable

$$\omega \sim \mathcal{N}(0,1)$$

and the loss given scenario ω_i is described by

$$Z_{i,j} = -\omega_i + W_{i,j}, \quad W_{i,j} \sim \mathcal{N}(0, \sigma^2_{\text{outer}})$$

with $\sigma_{\text{outer}} = 5$.

For the numerical experiments we take the threshold level $c \approx 2.326347$ that corresponds to $\alpha = 0.01$.

3.2 PUT OPTION EXAMPLE

In this example, we consider a portfolio consisting of a put option sold at $t = 0$ at a price X_0. We are interested in the loss at time $t = \tau$, namely

$$L(\omega) = X_0 - \mathbb{E}[e^{-r(T-\tau)}(K - S_T(\omega, W))^+],$$

where

$$S_T(\omega, W) := S_\tau(\omega) e^{(r-\sigma^2/2)(T-\tau)+\sigma\sqrt{T-\tau}W}, \quad W \sim \mathcal{N}(0,1),$$

and the scenario describes the underlying price at $t = \tau$ simulated under the historical probability:

$$S_\tau(\omega) := S_0 e^{(\mu-\sigma^2/2)\tau+\sigma\sqrt{\tau}\omega}, \quad \omega \sim \mathcal{N}(0,1).$$

Hence, the inner loss sample takes the form

$$Z_{i,j} = X_0 - e^{-r(T-\tau)}(K - S_T(\omega_i, W_{i,j}))^+.$$

For this example, the computation of $\sigma(\omega_i)$ is reduced to the calculation of the second moment of $e^{-r(T-\tau)}(K - S_T(\omega_i, W_{i,j}))^+$

The following lemma allows us to calculate it explicitly.

Lemma 1 (Second moment of a European put) *Let $S_T = S_0 \exp\left((r - \frac{1}{2}\sigma^2)T + \sigma\sqrt{T}Z\right)$ denote the stock price at maturity under the Black-Scholes model, where $\phi(z)$ is the standard normal density function and $\Phi(x)$ is the cumulative distribution function. Then, the expected squared payoff of the European put is given by:*

$$\begin{aligned}\mathbb{E}\Big[\left(e^{-r(T-\tau)}(K - S_T)^+\right)^2\Big] = \\ e^{-2r(T-\tau)}K^2\Phi(-d_-) \\ - 2e^{-r(T-\tau)}KS_\tau\Phi(-d_+) \\ + e^{\sigma^2(T-\tau)}S_0^2\Phi(-d_- - 2\sigma\sqrt{T-\tau}),\end{aligned}$$

where d_+ and d_- are defined by

$$d_+ = \frac{\log\left(\frac{S_0}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}}, \quad d_- = d_+ - \sigma\sqrt{T}.$$

As these calculations and the formula for the variance were not presented in the article, we provide it here.

PROOF.

$$\begin{aligned}\mathbb{E}\left[\left((K - S_T)^+\right)^2\right] &= \int_{-\infty}^{-d_-}\left(K - S_0\exp\left((r - \frac{1}{2}\sigma^2)T + \sigma\sqrt{T}z\right)\right)^2\phi(z)\,dz = \\ &= \int_{-\infty}^{-d_-}\left(K^2 + S_0^2\exp\left(2(r - \frac{1}{2}\sigma^2)T + 2\sigma\sqrt{T}z\right) - \right. \\ &\quad \left. - 2KS_0\exp\left((r - \frac{1}{2}\sigma^2)T + \sigma\sqrt{T}z\right)\phi(z)\,dz = \right. \\ &= K^2\Phi(-d_-) - 2K\exp(rT)S_0\Phi(-d_-) + S_0^2\exp(2rT + \sigma^2T)\Phi(-d_- - 2\sigma\sqrt{T}).\end{aligned}$$

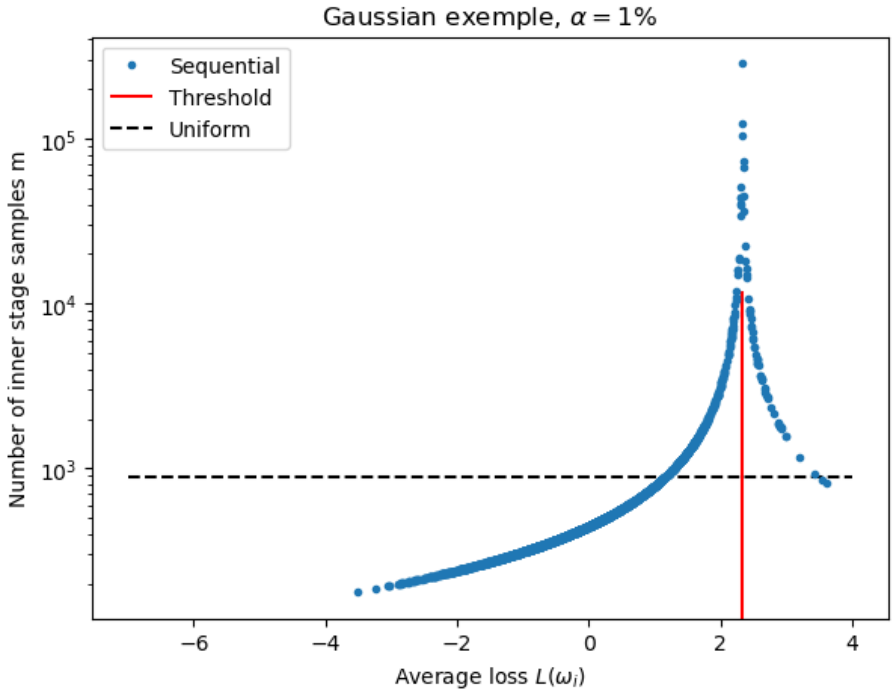


FIGURE 3.1
Distribution of the inner stage sample sizes for the Gaussian example.

3.3 NUMBER OF INNER STAGE SAMPLES

Here we illustrate the allocation of the inner losses sample sizes for the sequential method as a function of $L(\omega)$ for both examples we consider.

The obtained results correspond well to the plots provided in the article (fig. 6 (a), (b)) and demonstrate the impact of the indicator change probability maximization approach proposed in the sequential algorithm.

3.4 STATISTICAL ANALYSIS OF THE ALGORITHMS

As the running of each algorithm takes a lot of time, we needed several iterations to collect the data. Note also, that due to the extremely low speed of the sequential algorithm with the optimal parameters, we have less samples for it and the adaptive algorithm.

In this section, we compare the statistics such as variance, bias and MSE for each of the implemented methods. For the examples described in chapter 3, these statistics are provided in Table 3.1 for the Gaussian example and in Table 3.2 for the put option example. The empirical distributions of the

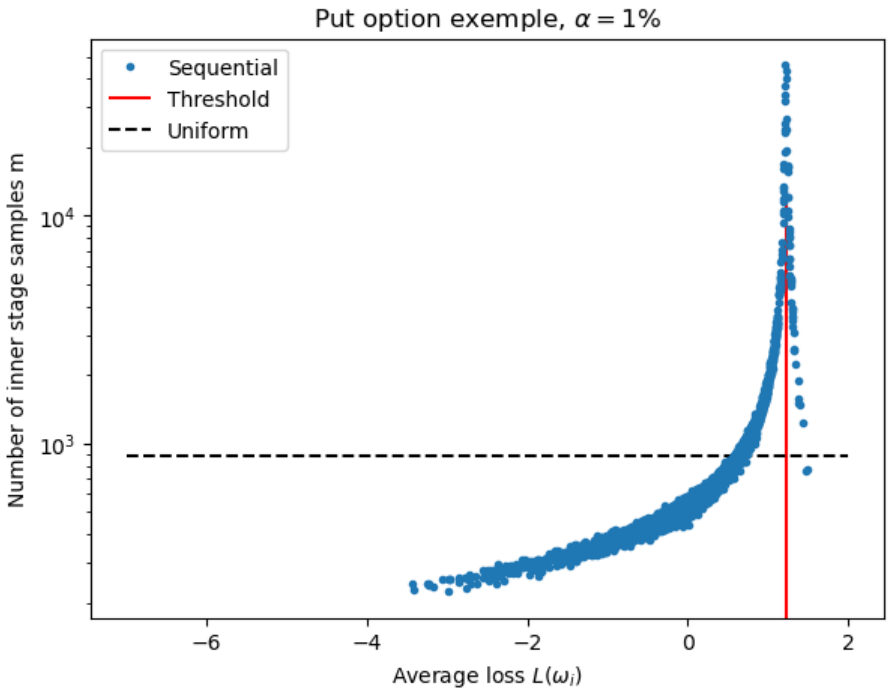


FIGURE 3.2
Distribution of the inner stage sample sizes for the put option example.

estimator $\hat{\alpha}$ are provided in fig. 3.3 for the first example and in fig. 3.4 for the second one.

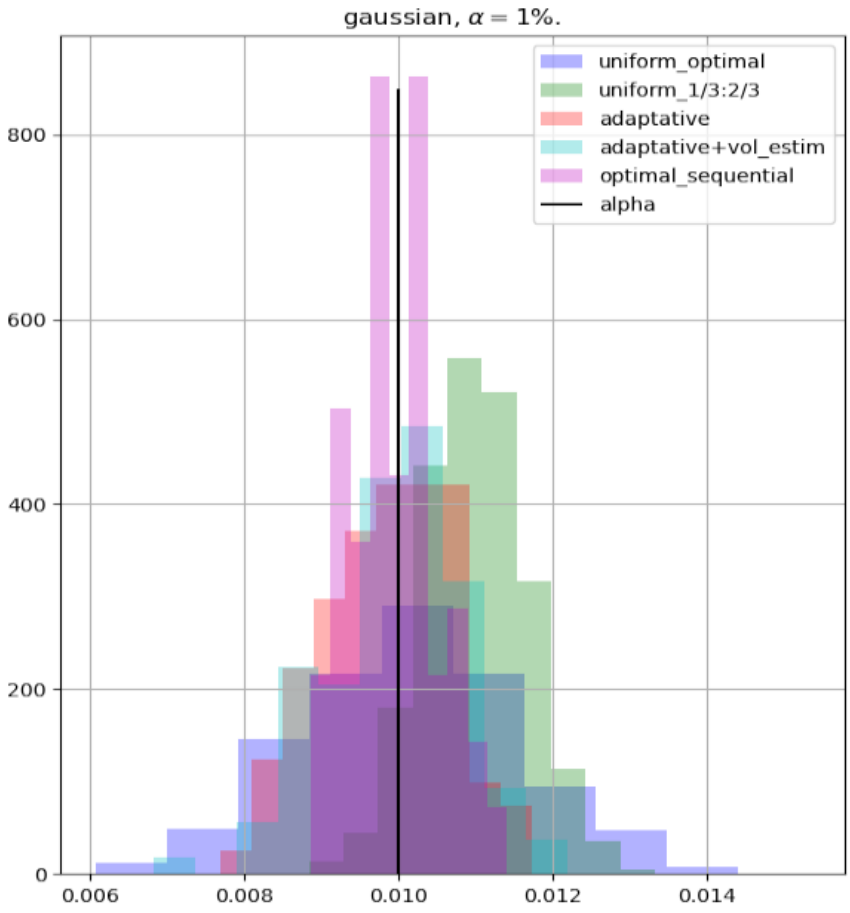


FIGURE 3.3
Empirical distribution of $\hat{\alpha}$ in the Gaussian example.

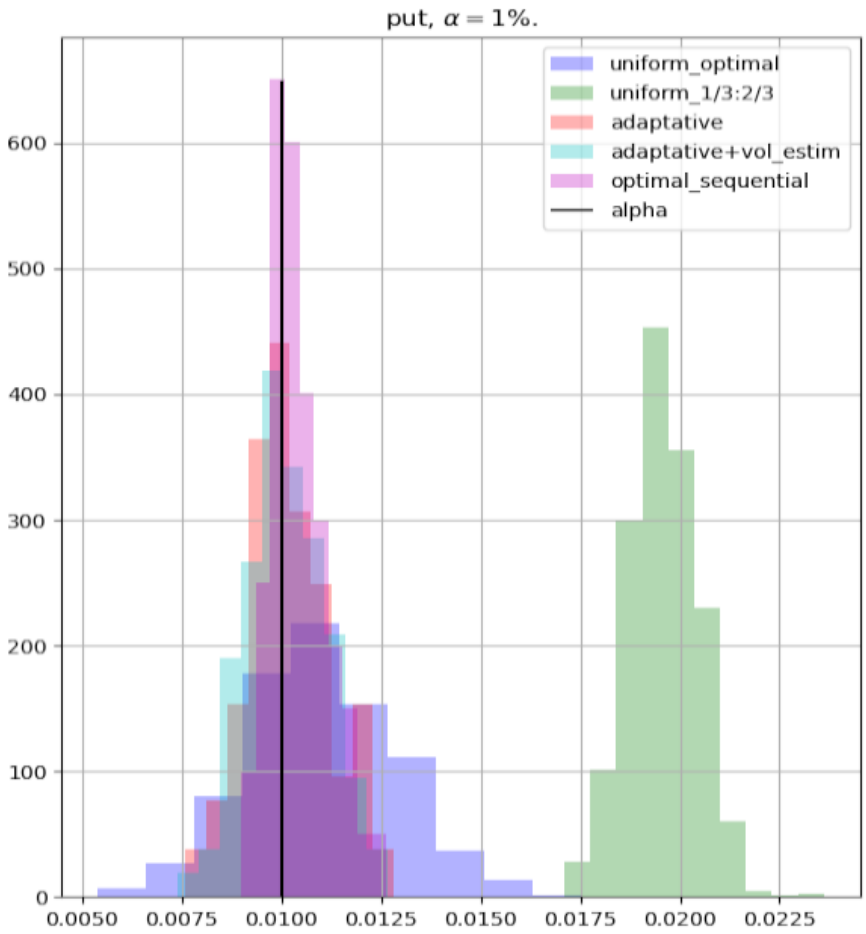


FIGURE 3.4
Empirical distribution of $\hat{\alpha}$ in the put option example.

	Bias squared	Variance	MSE
uniform_optimal	2.282883e-08	1.963741e-06	1.986569e-06
uniform_1/3:2/3	1.006678e-06	4.492207e-07	1.455899e-06
adaptative	1.686015e-08	6.616883e-07	6.785484e-07
adaptative+vol_estim	4.930910e-09	7.672919e-07	7.722228e-07
optimal_sequential	2.749557e-09	3.028519e-07	3.056015e-07

TABLE 3.1
Methods' statistics for the Gaussian example

	Bias squared	Variance	MSE
uniform_optimal	1.109091e-06	3.401587e-06	4.510678e-06
uniform_1/3:2/3	9.150180e-05	7.680689e-07	9.226987e-05
adaptative	3.770370e-08	1.102633e-06	1.140336e-06
adaptative+vol_estim	1.390654e-08	9.816640e-07	9.955705e-07
optimal_sequential	1.918015e-07	5.331615e-07	7.249630e-07

TABLE 3.2
Methods' statistics for the put option example

3.5 VARIANCE REDUCTION

The authors did not mention any variance reduction techniques. However, the proposed algorithms themselves can be considered as variance reduction methods (more precisely, MSE reduction). Therefore, it makes sense to compare them with the standard variance reduction techniques learnt in the course.

First, we tried to implement the antithetic method at the inner levels, but it did not result in any variance reduction. One of the possible explanations may be that the variance reduction at the inner level allows for a more precise estimation of $L(\omega_i)$, but does not impact significantly the final estimator.

In order to reduce its variance, we focused on importance sampling at the outer stage. Here is a theoretical overview following the logic of this technique: we suppose that scenario can be simulated as a function of a Gaussian random variable Z and we will change the measure in order to sample more scenarios in the region where the value of $\mathbb{1}_{L(\omega(Z))\geq c}$ will be positive (for example, where the average value of $L(\omega)$ will be close to c):

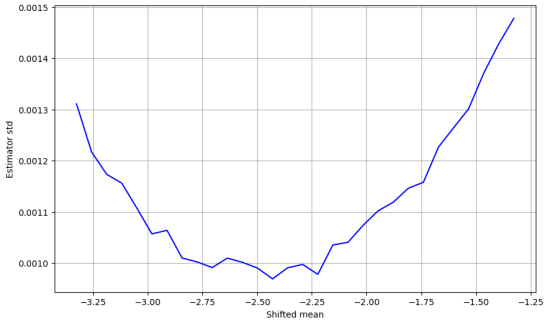
$$\mathbb{E}^p\left[\mathbb{1}_{L(\omega(Z))\geq c}\right] = \mathbb{E}^q\left[\frac{p(Z)}{q(Z)}\mathbb{1}_{L(\omega(Z))\geq c}\right] = \frac{1}{N}\sum_{i=1}^{N}\frac{p(Z_i)}{q(Z_i)}\mathbb{1}_{\hat{L}(\omega(Z_i))\geq c}. \tag{3.1}$$

Here we supposed that initially Z was distributed with density $p(\cdot)$ and after the change of measure its density became equal to $q(\cdot)$. In the normal case, where $p(\cdot)$ corresponds to $\mathcal{N}(0,1)$ and $q(\cdot)$ corresponds to $\mathcal{N}(\theta,1)$, one obtains

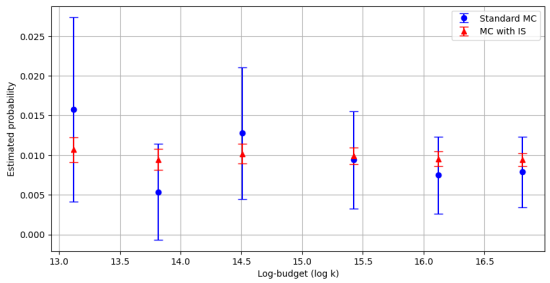
$$\frac{p(z)}{q(z)} = e^{\frac{\theta^2}{2}-z\theta}. \tag{3.2}$$

Note that we apply the importance sampling only at the outer simulation level in order to increase the number of samples with positive indicator value. The optimal value of θ can be determined

numerically in order to minimize the variance, see figures 3.5a, 3.6a. However, for the Gaussian case, a natural choice $\theta = -c$ will be close to the optimal one.

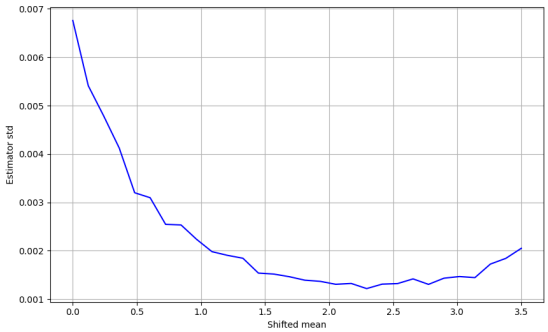


(A) Plot of the estimated standard deviation for a given mean shift based on the Importance Sampling method

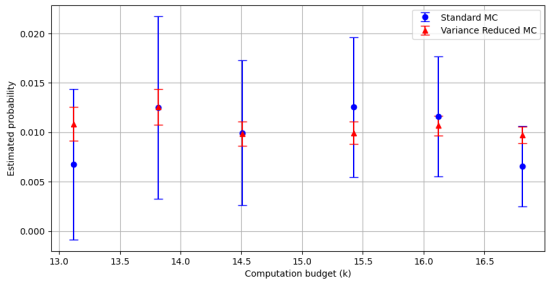


(B) Subplot comparing the final estimated probability and confidence intervals generated by Monte Carlo simulations for the Gaussian example.

FIGURE 3.5
Plots for the Gaussian example.



(A) Plot of the estimated standard deviation for a given mean shift based on the Importance Sampling method



(B) Subplot comparing the final estimated probability and confidence intervals generated by Monte Carlo simulations for the Put example.

FIGURE 3.6
Plots for the Put example.

	Bias squared	Variance	MSE
uniform_optimal_is	3.992918e-08	5.745139e-08	9.738057e-08
uniform_1/3:2/3_is	9.960860e-07	1.513412e-08	1.011220e-06

TABLE 3.3
Importance sampling statistics for the Gaussian example

	Bias squared	Variance	MSE
uniform_optimal_is	1.095291e-06	1.208782e-07	1.216169e-06
uniform_1/3:2/3_is	9.095110e-05	1.104268e-07	9.106153e-05

TABLE 3.4
Importance sampling statistics for the put option example

We see from the tables 3.3, 3.4 and Figures 3.5b, 3.6b that Importance Sampling introduces a variance reduction factor of approximately 10, compared to the results from tables 3.1 and 3.2. However, the large bias of 1/3:2/3 algorithm is the same as it could be expected.

3.6 CHOICE OF PROGRAMMING LANGUAGE

For the implementation of our project, Python was a natural choice for several reasons. On the one hand, Python offers an excellent library of tools for statistical analysis and visualization, making it much easier to explore and understand the academic concepts involved. However, it's important to recognize that C++ could have been a more powerful choice, especially for computationally-intensive operations such as Monte Carlo methods and random number generation, especially one-by-one simulation in the loop as it is needed for the sequential algorithm. Indeed, C++ is reputed to be much faster than Python, with performance figures that can be 10 to 100 times higher for computationally intensive tasks. Nevertheless, despite these potential advantages of C++, we preferred to focus on the ease and speed of implementation offered by Python. To compensate for Python's performance shortcomings, we used Numba, a JIT (Just-In-Time) compilation, that speeds up considerably the computations by compiling parts of the Python code into machine code on the fly. This approach enabled us to benefit from both Python's ease of use and a significant improvement in performance.

CHAPTER 4

DISCUSSION

Statistics obtained for the case $\alpha = 1\%$ for both problems are consistent with the ones provided by the authors of the article in Table 1 of Broadie, Du, and Moallemi 2011.

We see a very large bias that appears when using the uniform 1/3:2/3 method although its theoretical asymptotics is the same as for the optimal uniform algorithm. The error is even more dramatic for the put option example, possibly to the asymmetric and skewed loss distribution. We also note that optimal uniform algorithm demonstrates much more accurate results. This emphasizes the importance of the constant β in the allocation which is, as we already mentioned, typically not available.

The results above allow us to conclude that the adaptive method demonstrates a very good performance though it is not provided any information nor about optimal value of n, nor about the volatility σ. This stresses the robustness of this algorithm, on the contrary to the uniform algorithm exposed to the uncontrollable bias in the case of the wrong choice of the parameters or the sequential algorithm that also demonstrates a higher bias (and hence MSE) for non-optimal choice of the parameters.

However, it should be mentioned that all the sequential methods (all apart from the uniform) are much more time-consuming since the losses are simulated one-by-one rather than as arrays. Even with git-compilation, the execution times for sequantial and non-sequential methods differ by orders of magnitude. In order to make them comparable, one should either implement the methods in C/C++ or use languages such as cython.

Nevertheless, if comparing algorithms only in terms of budget, the simulation results provide strong evidence that sequential methods significantly outperform the uniform ones.

We have also found that variance reduction techniques, and in particular, importance sampling, allow us to decrease significantly the variance of the final estimator. However, it can not help to deal

with the bias caused by a bad choice of (n, m). It worth noting that integration of the importance sampling into the adaptive method will not be as straightforward as it was for the uniform algorithm since in deals with the variance estimation to allocate the available computational budget. However, it is still possible to combine these methods since the adapted algorithm relies already on many approximations and heuristics.

BIBLIOGRAPHY

Broadie, Mark, Yiping Du, and Ciamac C. Moallemi (2011). "Efficient Risk Estimation via Nested Sequential Simulation". In: *Management Science* 57.6, pp. 1172–1194. ISSN: 00251909, 15265501.