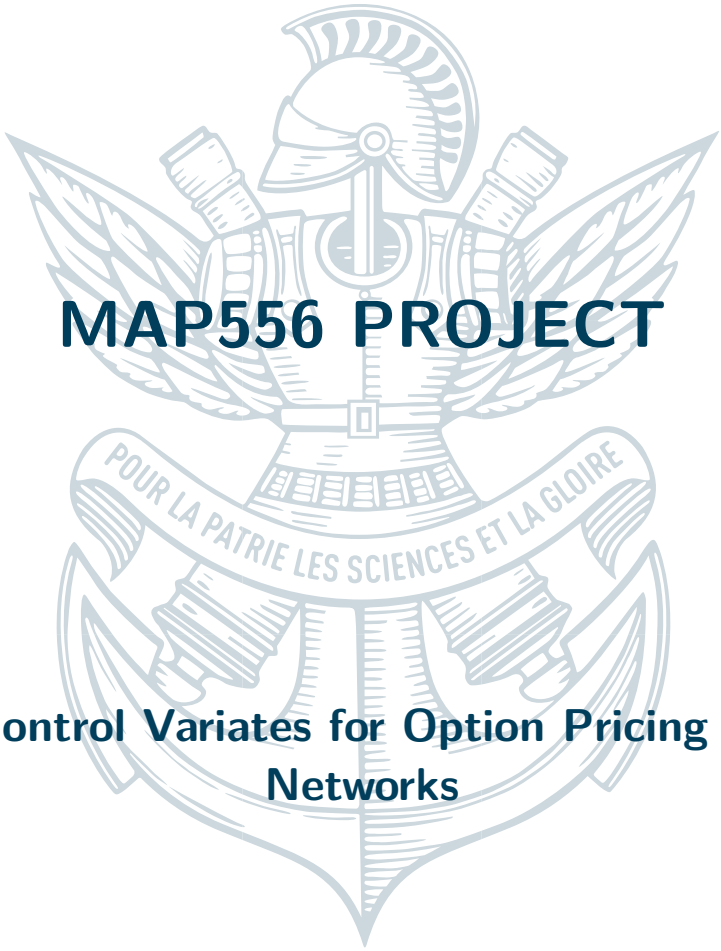


MAP556 PROJECT

Automatic Control Variates for Option Pricing using Neural Networks

8 décembre 2022

Franck SIGNE TALLA and Timothée SELOI

1 INTRODUCTION

Many pricing problems involve the calculation of a high-dimensional integral, which is usually estimated using Monte Carlo. The accuracy of a Monte Carlo estimator with M simulations is given by $\frac{\sigma}{\sqrt{M}}$. This convergence can be relatively slow depending on the variance σ^2 of the function to be integrated. To solve such a problem, variance reduction techniques such as control variates must be used.

In this report, we build on the work presented in the paper [1]. We study two approaches to improve Monte Carlo convergence using neural networks. The first approach is based on the fact that many high-dimensional financial problems have low effective dimensions [2]. By keeping only the necessary variables, we reduce the dimension of the input. The second approach is to construct an automatic control variate using neural networks. We learn the function to be integrated (which incorporates the scattering model plus the gain function) in order to construct a network that is highly correlated to it. The network that we use can be integrated analytically, which enables to use it as a control variate.

Finally, we show that the networks built resist a parameter change, which means that they can be trained once and still predict correctly prices when the parameters have changed.

2 PROBLEM DEFINITION, METHODOLOGY AND THEORETICAL ANALYSIS

2.1 Task Definition

Let N_{asset} be the number of underlying assets, which evolution follows either a Black and Scholes model, a local volatility model or a Heston model. The dynamic in the first two cases is given by :

$$\begin{cases} \forall i \in \{1, \dots, N_{asset}\}, & dS_t^i = S_t^i \left(rdt + \sigma_t^i(t, S_t^i) dB_t^i \right) \\ \forall i \neq j, & d(B_t^i B_t^j) = \rho dt \\ \forall i \in \{1, \dots, N_{asset}\}, & d(B_t^i B_t^i) = dt \end{cases}$$

where $\sigma_t^i(t, S_t^i) = \sigma$ in the Black and Scholes model, and $\sigma_t^i(t, S_t^i) = 0.6 \left(1.2 - e^{-0.1t - 0.001\left(S_t^i e^{rt} - S_0\right)^2} \right) e^{-0.05\sqrt{t}}$ in the local volatility model. In the Heston model, prices are given by :

$$\begin{cases} \forall i \in \{1, \dots, N_{asset}\}, & dS_t^i = S_t^i \left(rdt + \sqrt{\sigma_t^i} dB_t^i \right) \\ \forall i \in \{1, \dots, N_{asset}\}, & d\sigma_t^i = \kappa_i \left(a_i - \sigma_t^i \right) dt + \nu_i \sqrt{\sigma_t^i} d\hat{B}_t^i \end{cases}$$

where $(B, \hat{B}) \in \mathbb{R}^{2N_{asset}}$ is a brownian motion with covariance $\tilde{\Gamma} = \begin{pmatrix} \Gamma & \gamma\Gamma \\ \gamma\Gamma & \gamma^2\Gamma + (1-\gamma^2) I_{N_{asset}} \end{pmatrix}$ and Γ denotes the covariance matrix of B, while $\hat{B}$'s covariance matrix is the identity matrix $I_{N_{asset}}$.

The computation of an option price is often equivalent to the calculation of an expectation of the form $\mathbb{E}[f(Z)]$, where $Z := (Z_1, \dots, Z_N)$ is a random vector composed of N independent standard normal variables. N is proportional to N_{asset} and depends on the model and on N_{Euler}, the number of steps in the

Euler scheme. For example, with the Heston model, $N = 2 * N_{asset} * N_{Euler}$. $\mathbb{E}[f(Z)]$ may be approached by the Monte Carlo estimator $S_M = \frac{1}{M}\sum_{i=1}^{M} f(Z^{(i)})$, where $Z^{(i)}$ is a random sample of Z. Its variance is $Var(S_M) = Var\left(\frac{1}{M}\sum_{i=1}^{M} f(Z^{(i)})\right) = \frac{\sigma^2}{M}$ where $\sigma^2 := Var(f(Z))$.

In this report, we aim to find control variates that allow variance reduction.Therefore, our goal is to find a variable Y such that :

$$\begin{cases} Y = f(Z) - h(Z) + \mathbb{E}[h(Z)] \\ Var(Y) < Var(f(Z)) = \sigma^2 \end{cases}$$

$Var(Y) = Var(f(Z)) + Var(h(Z)) - 2Cov(f(Z), h(Z))$ so the second condition is ensured if :

$$Cov(f(Z), h(Z)) > \frac{1}{2}Var(h(Z))$$

As $\mathbb{E}[Y] = \mathbb{E}[f(Z)]$, it allows to use the control variate estimator : $S_{CV} = \frac{1}{M}\sum_{i=1}^{M} Y^{(i)}$, which variance is $Var(S_{CV}) = \frac{Var(Y)}{M} < Var(S_M)$. In the following, we will describe three methods to find such a variable by using neural networks.

2.2 Presentation of the Method

2.2.1 • First Approach : Network dimension reduction

First step We build a neural network that replicates the payoff function f. Recall that $Z \sim \mathcal{N}_N(0, I_N)$ is the input of the network. We look for n normal directions summarising the variance of f of the form $\tilde{Z}_i = \sum_{k=1}^{N} u_{ik} Z_k, \forall i \in \{1, \dots, N\}$ with the condition $\sum_{k=1}^{N} u_{ik}^2 = 1, \forall i \in \{1, \dots, N\}$. This ensures that $Var(\tilde{Z}_i) = 1, \forall i \in \{1, \dots, N\}$.

The network must verify the following criteria :

- The first hidden layer consists of n neurons. It has no activation function and no bias. Therefore, the output of this layer must have the form WZ.
- Additional hidden layers are added such that f is correctly approximated. We don't know in advance how many layers are needed but the network must be able to reconstitute the payoff.

Second step Once the network is trained, an optimal weight matrix for the first hidden layer is obtained. It is denoted by W. Then, we set $\tilde{Z}_i := \sum_{k=1}^{N} u_{ik} Z_k, \forall i \in \{1, \dots, N\}$ where $u_{ik} := \frac{w_{ik}}{\sqrt{\sum_{j=1}^{N} w_{ij}^2}}, \forall i \in \{1, \dots, N\}$ defines the weighted normalized matrix ensuring that $Var(\tilde{Z}_i) = 1$.

Third step $f(\mathbb{E}[Z|\tilde{Z}])$ is used as a control variate.

Since $f(\mathbb{E}[Z|\tilde{Z}])$ [1] is highly correlated to $f(Z)$, it will be an effective control variate. Furthermore, it only depends on the new variables $\tilde{Z}$ of small dimension, we can use a numerical integrator much faster and precise than Monte Carlo.

2.2.2 • Automatic control variate

Let $H : \mathbb{R}^N \to \mathbb{R}$ be a neural network that approximates the payoff function f given $Z \sim \mathcal{N}_N(0, I_N)$. Let $Y := f(Z) - H(Z) + \mathbb{E}[H(Z)]$. When the network will be trained, $H(Z)$ will be highly correlated to $f(Z)$ and we just need to compute $\mathbb{E}[H(Z)$ in order to use it as a control variate. The next two methods suggest two ideas for computing this expected value. The first one computes it numerically and the second one uses an analytic integration.

a) **Second approach : Using numerical integration**

First step The first step is the same as the one used in the first approach. Please refer to it.

1. For more detail about this method, please look at [1]

Second step Once the network is trained, an optimal weight matrix for the first hidden layer is obtained. It is denoted by W. $\tilde{H}$ represents the remaining layers (hidden layers and the output layer). Hence $\mathbb{E}[H(Z)] = \mathbb{E}[\tilde{H}(WZ)] = \mathbb{E}[\tilde{H}(\tilde{Z})]$. This allows to compute $\mathbb{E}[H(Z)]$.

b) **Third approach : Using analytical integration**

First step As before, we build a neural network that replicates the payoff function f, but this time, it has a simpler form, that enables the calculation of $\mathbb{E}[H(Z)]$ with an exact formula.

The network must verify the following criteria :

— The network only has one hidden layer and one output layer.

— The activation function of the hidden layer is the ReLu function and its bias is denoted by b_1. Therefore, the output of this layer must have the form $ReLu(W_1Z + b_1) = (W_1Z + b_1)^+$.

— The output layer is such that the network writes : $H(Z) = W_2(W_1Z + b_1)^+ + b_2$

Second step Once the network is trained, the explicit computation of $\mathbb{E}[H(Z)]$ is given by [2] :

$$\mathbb{E}[H(Z)] = W_2 \left(\frac{\sigma_i}{\sqrt{2\pi}} exp\left(-\frac{1}{2}\frac{\mu_i^2}{\sigma_i^2} \right) + \mu_i \left(1 - N\left(-\frac{\mu_i}{\sigma_i} \right) \right) \right)_{1 \le i \le n} + b_2 \tag{EHZ}$$

where n denotes the number of neurons in the hidden layer, $\mu_i = (b_1)_i$ and $\sigma_i^2 = \sum_{j=1}^N (W_1)_{ij}^2, \forall i \in \{1, \dots, n\}$.

2.3 Theoretical justification and guarantees

Since we found two mistakes in the demonstration of (EHZ) in [1], we correct them in the following. The details of the first approach are explained in the same article and for lack of space we prefer not to show these results again.

First, $\mathbb{E}[H(Z)] = W_2\mathbb{E}[(W_1 + b_1)^+] + b_2$ where $W_1Z + b_1 = \begin{bmatrix} \sum_{j=1}^N (W_1)_{1j}Z_j + (b_1)_1 \\ \vdots \\ \sum_{j=1}^N (W_1)_{nj}Z_j + (b_1)_n \end{bmatrix} = \begin{bmatrix} \sigma_1 Y_1 + \mu_1 \\ \vdots \\ \sigma_n Y_n + \mu_n \end{bmatrix}$,

with $\mu_i = (b_1)_i$ and $\sigma_i^2 = \sum_{j=1}^N (W_1)_{ij}^2$ and $Y \sim N(0,1)$.

Therefore, we obtain the next expression, which directly leads to (EHZ) :

$$\mathbb{E}\left[(\sigma_i Y_i + \mu_i)^+\right] = \sigma_i \int_{-\frac{\mu_i}{\sigma_i}}^{+\infty} y \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy + \mu_i \mathbb{P}\left(Y_i \ge -\frac{\mu_i}{\sigma_i} \right) = \frac{\sigma_i}{\sqrt{2\pi}} e^{-\frac{\mu_i^2}{2\sigma_i^2}} + \mu_i \left(1 - N(-\frac{\mu_i}{\sigma_i}) \right)$$

3
EXPERIMENTAL EVALUATION

3.1 Methodology

As mentioned earlier, our objective in this report is twofold : first, we want to find an estimator with lower variance than the Monte Carlo estimator, and second, we want our networks to be robust.

Therefore, the **two criteria** that we use to evaluate the methods are :

— Comparison of the prices given by the methods with Monte Carlo prices for the various price models (Black and Scholes, local volatility and Heston model). We also observe the variance of control variates estimators and compare it with the Monte Carlo estimator variance.

2. See §2.3. Theoretical justification and guarantees

— Robustness of the control variates : we test whether the network presented in the first approach (see §(2.2.1)) is able to resist changes in the variables of the payoff and the model. Note that the robustness is only tested with the first neural network on the arithmetic Asian option and on the AutoCall.

As in [1], we **test the methods with the following options** :
— a call on a basket with payoff $max(0, \sum_{i=1}^{N} w_i S_T^i - K)$;
— a put on a worst-of which payoff is $max(0, \min_{1\leq i\leq N} S_T^i$;
— a binary option on a basket which payoff is $G\mathbf{1}_{\sum_{i=1}^{N} w_i S_T^i \geq K}$;
— an arithmetic Asian option with payoff $max\left(0, \frac{1}{N_{Euler}} \sum_{k=1}^{N_{Euler}} \sum_{i=1}^{N} w_i S_{t_k}^i - K\right)$;
— an AutoCall, which can't be described here because of its complexity (see [1]). More precisely, we test the AutoCall price sensitivity with respect to several parameters (spot, strike, AutoCall barrier etc.).

In all approaches, **the network is trained with $N_{sample} > 10^5$ examples**, so that it always exceeds by far the number of features [3]. Each example is composed of N independent standard gaussians $Z = (Z_1, \dots, Z_N)$. The corresponding label is given by the discounted payoff $f(Z)$. An example of $f(Z)$ for the call on a basket in the Black and Scholes model is given in the following :

Example The covariance matrix of $(B_T^i)_{i\in\{1,\dots,N\}}$ is $T\Sigma$ where $\Sigma_{ij} = \mathbf{1}_{i=j} + \rho\mathbf{1}_{i\neq j}$. Let L be a square matrix such that $LL^T = \Sigma$. Thus, if $Z \sim \mathcal{N}_N(0, I_N)$, then $B_T := (B_T^1, \dots, B_T^N) \sim \mathcal{N}_N(0, T\Sigma)$. By noticing that $\sqrt{T}LZ \sim \mathcal{N}_N(0, TLL^T) = \mathcal{N}_N(0, T\Sigma)$ and by using the explicit expression of S_T^i in this case, we obtain the price of the call on a basket. Its payoff is $\mathbb{E}[f(Z)]$ and :

$$f(Z) = e^{-rT}\left(\sum_{i=1}^{N} w_i S_0 exp\left\{\left(r - \frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}(LZ)_i\right\} - K\right)^+$$

3.2 Results

With the Black and Scholes model, we use the following parameters for all $i, j \in \{1, \dots, N_{asset}\}$:

$$S_0^i = K = G = 100 \quad w_i = \frac{1}{10} \quad T = 1 \quad N_{asset} = 100 \quad \rho_{ij} = 0.75 * 1_{i\neq j} + 1_{i=j} \quad r = 0.02$$

Black and Scholes model results				
Option	Basket	Asian	Digit	Worst Of
Monte Carlo price	14.57	14.87	45.33	39.70
1^{st} approach price	14.37	14.50	45.30	39.10
1^{st} approach Var Ratio	383.70	422.9	12.32	5.6
Monte Carlo price	14.99	14.78	45.43	40.21
2^{nd} approach price	14.91	14.77	45.75	40.27
2^{nd} approach Var Ratio	378.87	1013.31	30.61	16.72
Monte Carlo price	14.98	14.89	45.41	40.24
3^{rd} approach price	14.95	14.85	45.82	40.22
3^{rd} approach Var Ratio	2463.6	692.95	46.61	25.81

With the local volatility model, we use the following parameters for all $i, j \in \{1, \dots, N_{asset}\}$:

$$S_0^i = K = G = 100 \quad w_i = \frac{1}{10} \quad T = 1 \quad N_{asset} = 10 \quad \rho_{ij} = 0.75 * 1_{i\neq j} + 1_{i=j} \quad N_{Euler} = 100 \quad r = 0.02$$

Local volatility model results				
Option	Basket	Asian	Digit	Worst Of
Monte Carlo price	6.17	4.03	45.82	13.25
1^{st} approach price	6.25	3.93	46.94	13.28
1^{st} approach Var Ratio	301.93	4.35	3.26	4.89

3. The maximum number of features is reached with the Heston model and it is equal to $N = 2 * N_{Heston} * N_{Euler} = 2000$

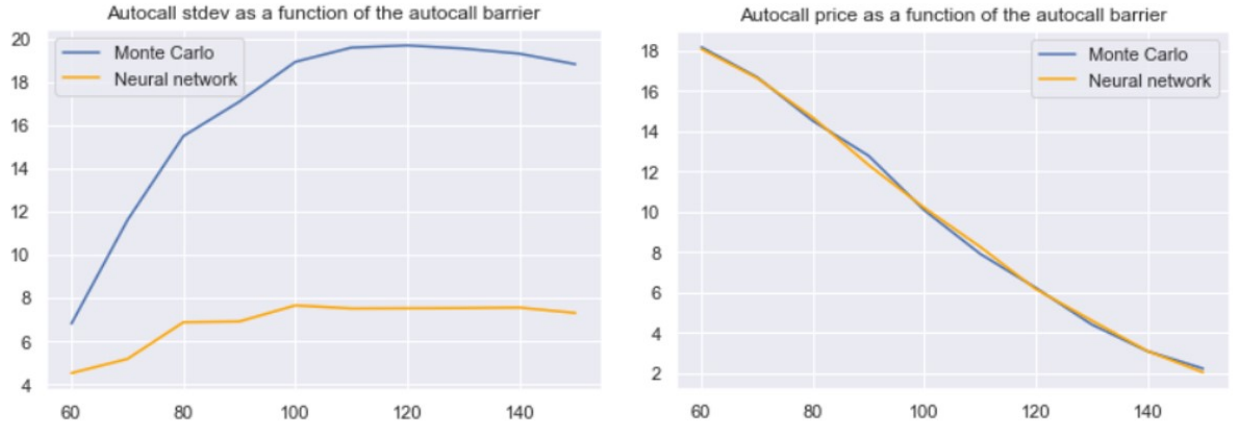
With the Heston model, we use the same parameters as before and for all $i, j \in \{1, \dots, N_{asset}\}$:

$$\rho_{ij} = 0.3 * 1_{i \neq j} + 1_{i=j} \quad \kappa_i = 2 \quad \gamma_i = -0.2 \quad \sigma_0^i = 0.04 \quad \nu_i = 0.01 \quad a_i = 0.04$$

Local volatility model results				
Option	Basket	Asian	Digit	Worst Of
Monte Carlo price	5.92	3.89	52.98	21.87
1^{st} approach price	6.04	3.74	52.25	21.70
1^{st} approach Var Ratio	155.31	327.47	6.62	2.01

The following graphs show the robustness of the first network when it comes to predicting the AutoCall price. **In the figures.pdf file, you will find a lot more graphs showing the robustness.**

FIGURE 1 – AutoCall price as a function of the barrier (right) and the corresponding standard deviation (left)



3.3 DISCUSSION

The previous results show, on the one hand, that the approaches do indeed recover Monte Carlo prices, and on the other hand, that the reduction in variance is significant in certain cases (mostly for the basket and Asian options). This confirms the results obtained in [1].

We have tested our methods under different models and shown through numerical examples that the control variables allow significant variance reduction. The first method we proposed has a major quality of resistance to parameter change. The other two methods are much more intuitive since we use a given network to approximate a gain function. They work better with non-linear functions and give very large speed gains. The last method which uses only one hidden layer is less efficient with complicated models but has the advantage of being very fast as the control expectation can be calculated analytically.

REFERENCES

[1] Jérôme LELONG, Zineb El Filali ECH-CHAFIQ et Adil REGHAI. "Automatic Control Variates for Option Pricing using Neural Networks". In : *Monte Carlo Methods and Applications, De Gruyter* (2021). DOI : https://hal.univ-grenoble-alpes.fr/hal-02891798.

[2] X. WANG et I. SLOAN. "Why are high-dimensional finance problems often of low effective dimension ?" In : *SIAM Journal on Scientific Computing* (2005).