Stacked Monte Carlo Methods for Option Pricing

Seminar paper

Felix Stubenvoll-Pretschuh

Matrikelnummer 01114103
e1114103@student.tuwien.ac.at

September 2020

Abstract

This seminar paper presents a Stacked Monte Carlo procedure and the algorithm's steps and discusses its application to the pricing of options in the Black-Scholes model. In this procedure different control variates for variance reduction are obtained via simple regression and fitting techniques and are ultimately combined ('stacked') to improve an existing standard Monte Carlo estimate. Its is shown that it offers great improvement in precision with regards to the existing estimate with very low additional computational cost.

Contents

1 Introduction **3**

2 Monte Carlo in Quantitative Finance **4**

3 Stacked Monte Carlo **5**
3.1 Variance reduction 5
3.2 Procedure 7
3.2.1 Model choice for control variate 7
3.2.2 Fitting the control variate 8
3.2.3 Estimating the control variate parameter 8
3.2.4 Integrating the control variate 9

4 Pricing Options in the Black-Scholes Model **10**
4.1 European call option 10
4.1.1 Dependency on model parameters 11
4.2 Asian call option 14

5 Conclusion **17**

1 Introduction

The valuation of options plays an important role in modern finance and there have been developed various methods to price such contracts. These methods include for example the Black-Scholes formula, arguably the most popular one, which enables us to obtain the value of European Call and Put Options in a closed form. In general however, the characteristics of many financial instruments do not allow for closed form pricing formulae thus, numerical methods are applied for solving the integrals and stochastic differential equation through which the instrument's price can often be expressed. As such, the Monte Carlo technique offers a simple and flexible way to option valuation problems, where an estimate is obtained via sampling from a certain underlying distribution. However, any numerical method is prone to approximation errors and instabilities, especially if the dimension of the problem increases. While the error from a Monte Carlo technique can in theory be rendered arbitrarily small if only the sample size is increased accordingly, this approach is only practical to a certain extend given the constraints imposed by computational resources and time. Thus different methods of error or variance reduction have been developed with the objective of achieving higher levels of stability of estimates with similar accuracy and computation time. As one of such, the Stacked Monte Carlo method has been derived for option pricing. In general, the term "stacking" refers to the idea of blending the output of different models. These models' predictions are fed as inputs into a second-level learning algorithm which aims at finding an optimal combination of predictors that should ultimately yield better results than any single one on its own. First introduced in [15] a stacked predictor can be constructed in various ways ranging from simple linear forms to more complex nonlinear combinations of inputs that require significant tuning and training time. Stacking was successfully applied to a wide range of different problems, see for example [4, 10, 11, 13, 16] .

The idea of the Stacked Monte Carlo procedure was introduced by Alonso, Tracey and Wolpert aiming at improving existing standard Monte Carlo estimates. With its original application in aeronautics the three authors developed the method to predict the fuel consumption of future aircrafts under various assumptions about technological advancement in the decade from 2020 to 2030 as well as to quantify super sonic boom volume in dependency of different pressure characteristics of airplanes [1, 14]. The method was then adopted by Jaquier, Malone and Oumgari in 2019 and applied to the pricing of options [9]. With a standard Monte Carlo (MC) sample as input the procedure applies cross-validation and regression techniques to obtain a number of control variates which then yield different MC estimates. These estimates are then stacked by taking the mean to generate the improved estimation. This seminar paper gives an introduction and overview of the procedure presented in [9]. The following section 2 briefly introduces Monte Carlo in general and its application in quantitative finance. Section 3 then covers the Stacked MC method and goes through the various steps carried out to end up with a stacked estimate. Finally, in section 4 the method's application to pricing options in the Black-Scholes model and its results are discussed.

2 Monte Carlo in Quantitative Finance

In this section the Monte Carlo technique for pricing an option is explained in a general setting. Assume we have an option on the underlying x with pricing function f and maturity T, i.e. $f(x)$ is the option's price given the corresponding value x of the underlying at maturity. x can be a single underlying or any number of underlyings following a certain distribution F_x with probability density function $p(x)$. Under no arbitrage arguments and with risk-neutral pricing the option's price today is its expected value at maturity discounted by the risk-free rate. As a function of x, the distribution of the option's maturity value can be obtained from the distribution of its underlying's terminal value. Thus, if we know the distribution of x at time T we can derive the option's expected value by integration. Without loss of generality we set the risk-free rate to zero, this gives us

$$\bar{f} = \mathbb{E}[f(x)] = \int_A f(x)p(x)\mathrm{d}x \tag{1}$$

as fair price for our option today, where A denotes the range of integration. If we now further know how to simulate the process governing the price of the underlying we are able to obtain a sample of trajectories and with it a sample of terminal values $(x_i)_{i=1:n}$ for x at time T which in turn gives us a sample of options prices by plugging them into f. A Monte Carlo estimate $\hat{f}$ for $\bar{f}$ is then given by

$$\hat{f}_n = \frac{1}{n}\sum_{i=1}^{n} f(x_i). \tag{2}$$

This is an unbiased estimate for $\bar{f}$ which can be seen by taking $\mathbb{E}[\hat{f}]$. If f is integrable over A, i.e. $\mathbb{E}[|f(x_i)|] < +\infty$ and since $f(x_i)$ iid, the strong law of large numbers applies and $\hat{f}_n \to \bar{f}$ for $n \to +\infty$. If f is square integrable over A then we set

$$\sigma_f^2 = \int_A (f(x_i) - \bar{f})^2 \mathrm{d}x. \tag{3}$$

and by the central limit theorem the error

$$\frac{\bar{f} - \hat{f}}{\sqrt{\sigma_f^2/n}} \tag{4}$$

tends to a standard normal distribution with increasing n. Since we are in a setting were $\bar{f}$ is not available in closed form, the parameter σ_f^2 is unknown too, but can be estimated by the sample variance of the Monte Carlo estimate

$$\hat{s}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(f(x_i) - \hat{f}). \tag{5}$$

So, in addition to the estimate $\hat{f}$ of $\bar{f}$ we also obtain a measure for its error and since $\bar{f} - \hat{f} \sim \mathcal{N}(0, \frac{\hat{s}^2}{\sqrt{n}})$ confidence limits on $\hat{f}$ can be obtained. This is not only applicable to the standard Monte Carlo estimator but also to the Stacked Monte Carlo estimator introduced in section 3.

The advantages of a MC method lies in its simplicity and flexibility. The procedure makes no assumptions about the distribution of the underlying so it is straightforward to change for example the behavior of underlying stock prices by simply sampling from a different stochastic process. In fact, the process can be chosen so as to exhibit mean-reversion or jump diffusion properties [6, 3]. However, it is not even necessary for the underlying's distribution to have an analytic form, one can simply use the empirical one to obtain the estimate. Additionally, every parameter can be assumed to follow a certain distribution, this makes the method applicable to stochastic volatility models such as the Heston Model [9, 8]. These properties make the Monte Carlo method applicable not only to pricing options but to a wide range of problems in quantitative finance and beyond.

3 Stacked Monte Carlo

This section introduces the Stacked Monte Carlo Method (StackedMC) and its procedure. StackedMC is a post-processing technique that aims at reducing the error of an existing Monte Carlo estimate. The idea is to learn a control variate with rather simple regression techniques from a Monte Carlo sample such that the learnt function is a good approximation of the original one.

3.1 Variance reduction

A central feature of the solution to a Monte Carlo problem (1) is its standard error, which has the form $\sigma_f/\sqrt{n}$ as shown in section 2. Only depending on the standard deviation of the function f and the number of samples n, the convergence rate is given by $O(n^{-1/2})$. Thus, cutting the error in half requires increasing the number of samples by four or, improving the precision one decimal point needs plus 100 samples. This rate is both a strength and a weakness of Monte Carlo methods. A weakness because in general this rate can not be improved, which makes Monte Carlo a not competitive method for calculating one-dimensional integral. Here, other methods promise better improvements with increasing n, take for example the simple trapeziodal rule as an estimate

$$\bar{f} \approx \frac{f(0) + f(1)}{2n} + \frac{1}{n}\sum_{i=1}^{n-1} f\left(\frac{i}{n}\right)$$

which a rate of $O(n^{-2})$ for a twice differentiable function f. At the same time Monte Carlo's strength lies in the fact that its convergence rate of $O(n^{-1/2})$ holds for all dimensions d. This is especially useful in estimating integrals in high dimensional spaces where the error of other deterministic numerical integration techniques becomes significantly larger. Take again the product trapeziodal rule in d dimensions which has an error of $O(n^{-2/d})$.

An alternative approach to increasing n in Monte Carlo methods is to reduce the constant σ_f in the error term which is known as variance reduction. Various techniques to obtain such a reduction have been developed. What they all have in common is that the original problem $f(x)$ gets changed and tweaked in such a way as to improve the precision of estimates from crude Monte Carlo methods. One of these techniques is called the Control Variate method. Its idea is to replace the original problem by simpler yet similar one, whose solution can be computed analytically or at least at low computational cost. Its solution is then used to improve the accuracy of more complex problem at hand.

The StackedMC procedure aims at learning such a control variate form a given Monte Carlo sample, such that the distribution of the learnt function approximates the one of the original one. Suppose the function $g(x)$, our control variate, is a reasonable good fit for $f(x)$. We can then rewrite the problem (1) with some constant α as

$$\begin{aligned}\bar{f} &= \int f(x)g(x)\mathrm{d}x + \alpha \int \left[g(x) - g(x)\right] \mathrm{d}x \\ &= \alpha \int g(x)p(x)\mathrm{d}x + \int \left[f(x) - \alpha g(x)\right] p(x)\mathrm{d}x \\ &= \alpha \bar{g} + \int \left[f(x) - \alpha g(x)\right] p(x)\mathrm{d}x\end{aligned} \tag{6}$$

Now, $\bar{g} = \int g(x)p(x)\mathrm{d}x$ can be computed analytically and since g is a good fit for f the term $f - \alpha g$ will have a smaller variance than the original problem for certain choice of α. For a sample $(x_i)_{i=1:n}$ we then obtain the MC estimate

$$\bar{f} \approx \alpha \bar{g} + \frac{1}{n} \sum_{i=1}^{n} \left[f(x_i) - g(x_i)\right] \tag{7}$$

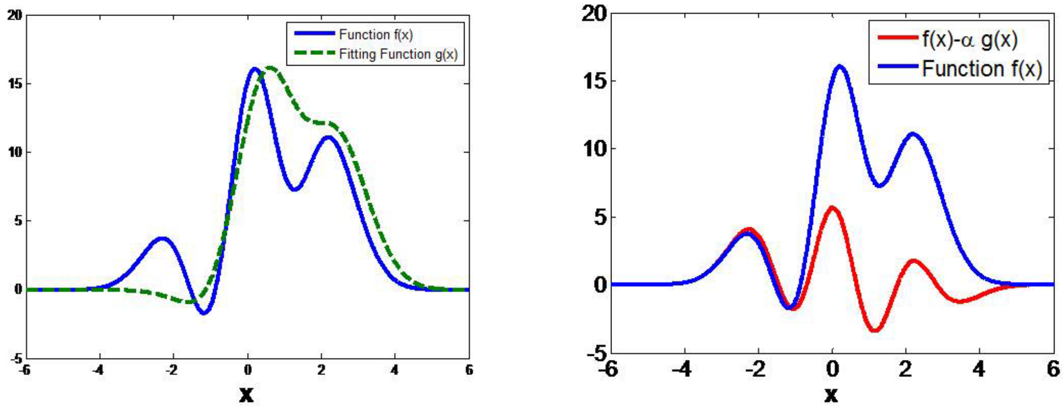
which has the variance

$$\sigma_{\bar{f}}^2 = \sigma_f^2 + \alpha^2 \sigma_g^2 - 2\alpha \sigma_{f,g}^2, \tag{8}$$

where $\sigma_{f,g}^2$ denotes the covariance of f and g. We now seek to minimize this term and by solving $\frac{\partial}{\partial \alpha} \sigma_{\bar{f}}^2 = 0$ we can show that this is achieved by the choosing

$$\alpha = \frac{\sigma_{f,g}}{\sigma_g} = \frac{\sigma_f}{\sigma_g} \rho_{f,g} \tag{9}$$

Plugging α into (8) we obtain $\sigma_{\bar{f}} = (1 - \rho_{f,g}^2)\sigma_f^2$, which implies that the condition $\rho_{f,g} \neq 0$ is sufficient to provide a reduction in variance. The effect of variance reduction is demonstrated in Figure 1.



(a) Example $f(x)$ and $g(x)$ (b) $f(x)$ and $f(x) - g(x)$

Figure 1: The original problem f and an approximation of it g. α was chosen as 0.85. We can see the term $f - \alpha g$ on the right side has smaller variance than f alone. If g is a good fit for f and the correlation between them is large, a high reduction is achieved.

3.2 Procedure

In the following sections the steps of the StackedMC's algorithm to obtain an estimate are described. In short, the following seven steps below are carried out:

1. Generate a Monte Carlo sample $(x_i)_{i=1:n}$ according to the distribution $p(x)$
2. Choose a model for the control variate. Fit the model at the sample points $f(x_i)$
3. Apply k-fold cross-validation to obtain k different estimates $\bar{f}_k$ for $\bar{f}$
4. **Stacking:** Obtain StackedMC estimate $\bar{f}_{SCM}$ as mean of the $\bar{f}_k$
5. Estimate the control variates optimal weight α
6. Integrate the control variate to obtain $\alpha\bar{g}$
7. Plug everything into (7) to obtain final estimate

3.2.1 Model choice for control variate

The first step is to pick a functional form for the control variate g. There are two conditions to choosing the model. First, g must be a reasonable, however not necessarily perfect) approximation of f, and second, the integral $\bar{g}$ must have a closed form solution or be at least costless to compute. An easy and convenient possibility is to fit a polynomial of order L, where the coefficients $\boldsymbol{c} = (c_k)_{k=0:L}$ are defined as the least square minimizer:

$$g(x) := c_0 + \sum_{k=1}^{L} c_k x^k \qquad \text{with} \qquad \mathbf{c} = \arg\min_{\mathbf{c}\in\mathbb{R}^L} \left\| f(x) - \sum_{k=0}^{L} c_k x^k \right\|^2 \tag{10}$$

The above choice can be used if one-dimensional problems, e.g. for a European call option. In the case of multi-dimensional problems, such as for basket options where $\boldsymbol{x} = (x_i)_{i=1:m}$, multivariate polynomials may be used:

$$g(\boldsymbol{x}) = \sum_{|\boldsymbol{\alpha} \leq L|} c_{\boldsymbol{\alpha}} \boldsymbol{x}^{\boldsymbol{\alpha}}, \qquad \boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_m) \in \mathbb{N}^m \tag{11}$$

Further, since call option payoffs can be discontinuous, a piecewise-polynomial function with zero value for parts of the domain may proof as a good choice, where $g(\boldsymbol{x}) = 0$ if $\boldsymbol{a}^T\boldsymbol{x} + \boldsymbol{a}_0 < 0$ and $g(x)$ is (11) otherwise for some truncation plane $\boldsymbol{a}$.

3.2.2 Fitting the control variate

Fitting algorithms such as the StackedMC method bear the risk of increasing the bias of the MC estimate. Moreover, when approximating functions there is always the danger of over-fitting. The first issue is avoided through the construction of the estimator (6). Since the expected value of the fit g is both added and subtracted, (6) remains an unbiased estimator of (1). The issue of over-fitting can be mitigated by using cross-validation. Here, the dataset N of samples of x is split into K disjoint sets (or folds) N_k. Each one of the folds is a training set once, the remaining $N - N_k$ sets are the training-set. From each training set a different fitting function g_k is obtained and its goodness of fit is assessed using a chosen metric. This is known as the out-of-sample error.

Following this procedure we get K fitting functions $\bar{g}_k$ of f for each training-set $N - N_k$, then plug them into (6) and use the respective test-set data $x_i^{(k)} \in N_k$ to get an estimates for the integral term $\bar{g}$ and for $\bar{f}$ as a whole with each one of the fitters. With $n_k := |N_k|$:

$$\bar{f}_k := \alpha \bar{g}_k + \frac{1}{n_k} \sum_{i=1}^{n_k} \left[f\left(x_i^{(k)}\right) - \alpha g_k \left(x_i^{(k)}\right)\right], \; k = 1...K \tag{12}$$

Standard procedure in cross-validation would asses the out-of-sample error for each fit and chose the best fit k^* as the one with the smallest error. This fit would then be used as single $g(x)$ to approximate $f(x)$, however, this again raises the issue of overfitting. Instead, at this point the technique of stacking is adopted and a final StackedMC estimate $\bar{f}_{SMC}$ of $\bar{f}$ is obtained by taking the mean

$$\bar{f}_{SMC} := \frac{1}{K} \sum_{k=1}^{K} \tilde{f}_k = \frac{1}{K} \sum_{j=1}^{K} \left[\alpha \bar{g}_k + \frac{1}{N_k} \sum_{i=1}^{N_k} f\left(x_i^{(k)}\right) - \alpha g_k \left(x_i^{(k)}\right)\right] \tag{13}$$

3.2.3 Estimating the control variate parameter

We can now use the predictions for the test-sets and the true values $f(x_i)$ to estimate α with the control variate formula (9) from above. The classical unbiased empirical estimates for mean, variance and covariance are used.

$$\hat{\mu}_g := \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{n_k} g_k\left(x_i^{(k)}\right) \qquad \hat{\mu}_f := \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

$$\hat{\sigma}_g^2 := \frac{1}{n-1}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\left[g_k\left(x_i^{(k)}\right) - \hat{\mu}_g\right]^2 \qquad \hat{\sigma}_f^2 := \frac{1}{n-1}\sum_{k=1}^{K}(f(x_i) - \hat{\mu}_f)^2$$

$$\hat{\sigma}_{f,g}^2 := \frac{1}{n-1}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\left[f\left(x_i^{(k)}\right) - \hat{\mu}_f\right]\left[g\left(x_i^{(k)}\right) - \hat{\mu}_g\right]$$

3.2.4 Integrating the control variate

This section covers the evaluation of the integral $\bar{g}$ if the model choice for the control variate is a polynomial, as discussed in section 3.2.1. If we look at the multivariate case, using multi-index notation $g(x)$ has the form:

$$\hat{g}(\boldsymbol{x}) = \int \sum_{|\boldsymbol{\alpha}|\leq L} c_{\boldsymbol{\alpha}} \boldsymbol{x}^{\boldsymbol{\alpha}} p(\boldsymbol{x}) d\boldsymbol{x} = \sum_{|\boldsymbol{\alpha}|\leq L} c_{\boldsymbol{\alpha}} \int \boldsymbol{x}^{\boldsymbol{\alpha}} p(\boldsymbol{x}) d\boldsymbol{x} = \sum_{|\boldsymbol{\alpha}|\leq L} c_{\boldsymbol{\alpha}} \mathbb{E}[\boldsymbol{x}^{\boldsymbol{\alpha}}] \tag{14}$$

with $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \mathbb{N}^m$ and random vector $\boldsymbol{x} = (x_i)_{i=1:m}$. The integration hence is reduced to the computation of moments. In the Black-Scholes model $\boldsymbol{x}$ is drawn from a one-dimensional normal distribution with zero mean, thus we obtain the integral easily by

$$\mathbb{E}[X^n] = \begin{cases} 0 & \text{, if n is odd} \\ \frac{n!\sigma^n}{2^{n/2}(n/2)!} & \text{, if n is even} \end{cases} \tag{15}$$

In the multidimensional case, as later for Asian Options, with a mutli-variate Gaussian distribution with covariance matrix Σ we are able to obtain the moments via the moment generating function

$$\mathbb{E}[e^{\boldsymbol{x}\boldsymbol{t}}] = e^{\frac{1}{2}\boldsymbol{t}^T\Sigma\boldsymbol{t}} \tag{16}$$

Finally, if the control variate has piecewise linear form with zero value for some truncated plane $\boldsymbol{a}$, with $\Omega_+ := \boldsymbol{x} : \boldsymbol{a}^T\boldsymbol{x} + \boldsymbol{a_0}$

$$g(\boldsymbol{x}) = \begin{cases} c_0 + \sum_j c_j x_j & \text{, if } \boldsymbol{x} \in \Omega_+ \\ 0 & \text{, else} \end{cases} \tag{17}$$

it is only necessary to compute the zeroth and first moment of the truncated Normal distribution in order to obtain $\bar{g}$, i.e.

$$\int_{\Omega_+} n(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \quad \text{and} \quad \int_{\Omega_+} x_j n(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}, \quad \text{for each } j = 1...m$$

with $n(.)$ denoting the density of the multivariate Gaussian distribution on the truncated plane Ω_+. This form for the control variate has some computational cost advantage and yields good results as shown in section 4. If the integrand is linear, as it is in our case above, the integral on a truncated domain can be computed in closed form as discussed in [12] and [9].

Given the value for the integral $\bar{g}$ as well as for α we can now plug both into equation (13) to obtain our final Stacked Monte Carlo estimate.

4 Pricing Options in the Black-Scholes Model

In the following section the established Stacked MC procedure is applied to price a European call option and a path depended Asian call option, both in the Black-Scholes model. The improvement against standard Monte Carlo estimates as well as the results' dependency on model parameters is assessed. The method can also be used in stochastic volatility models, for its application in the Heston Model see [9].

4.1 European call option

In order to apply the Stacked MC procedure we first need to define the pricing function f which we want to estimate as well as specify which distribution the random variable $X \sim F_X$, as input for f, follows. We consider a European call option who's payoff with strike price K is given by

$$(S_T - K)_+$$

whereas S_T denotes the stock price at time of maturity T. In the Black-Scholes Model the underlying stock price movements are modelled such that they follow a geometric Brownian Motion with constant drift r and volatility σ, i.e. S_t it satisfies the stochastic differential equation

$$\mathrm{d}S_t = rS_t\mathrm{d}t + \sigma S_t \mathrm{d}W_t, \tag{18}$$

for some one dimensional Brownian Motion W. The solution to this SDE can be derived as

$$S_t = S_0 e^{\left(r-\frac{\sigma^2}{2}\right)t+\sigma W_t}, \tag{19}$$

and from (18) and (19) it follows that $\log(S_t)$ is normally distributed (for details see [7], chapters 14-15) and thus the price of a call option at $t = 0$ is given by

$$e^{-rT}\mathbb{E}[(S_T - K)_+] = e^{-rT} \int_{\mathbb{R}} \left(S_0 e^{\left(r-\frac{\sigma^2}{2}\right)T+\sigma\sqrt{T}x} \right)_+ n(x)\mathrm{d}x \tag{20}$$

where the term in brakets in the integrand is our function $f(x)$ in (1) and $n(x)$ is the Gaussian density. By sampling x from a standard normal distribution we can apply the StackedMC algorithm. Let's assume a setting with the parameters:

$$(S_0, K, r, \sigma, T) = (100, 100, 0.05, 0.2, 1) \tag{21}$$

The exact option price given these parameters obtained via the Black-Scholes formula is $C_{BS} = 10.4506$. Sampling $N = 10^5$ realizations of a gaussian random variable, fitting a polynomial of order 4 for the function $g(.)$ at the sample points and using 2-fold Cross-validation in the StackedMC procedure yields the results presented in table 1.

MC			Stacked MC			Total	Improvement	
Price	KI	Time	Price	KI	Time	Time	Absolute	Ratio
10.4395	0.0913	0.57	10.4507	0.0061	0.30	0.87	0.0851	14.90

Table 1: StackedMC vs. standard MC for a standard European Call option

KI denotes the half width of the confidence interval for the obtained estimates calculated as

$$\mu \pm z_{0.95} \frac{\sigma_n}{\sqrt{n}}$$

where μ is the mean and $z_{0.95} = 1.96$ is the Gaussian quantile at the 95% level. σ_n is the sample standard deviation and calculated as such from obtained option prices $(f(x_i))_{i=1:n}$ in the standard Monte Carlo case and for the StackedMC one following (8), also using sample estimates as in section 3.2.3. Absolute improvement and improvement ratio are defined as $\mathrm{CI}_{\mathrm{MC}}$ - $\mathrm{CI}_{\mathrm{SMC}}$ and $\mathrm{CI}_{\mathrm{MC}}/\mathrm{CI}_{\mathrm{SMC}}$ respectively, time units are seconds. For better comparison the following results are scaled using the time to price a European Call option with $n = 10^5$ draws from table 1 (0.57 seconds) as 1 unit of time. The above results then read

MC	StackedMC	Total	Equivalent
1	0.526	1.526	222.08

Table 2: Run times scaled to 1 unit = 0.57 seconds

The last column in table 2 states an equivalent Monte Carlo time which is the time that would have been necessary to obtain a similar CI size as with StackedMC using the standard Monte Carlo alone. It is defined as the standard MC time multiplied by the squared improvement ratio. Assuming computation time increases linearly with increasing sample size and given the convergence rate $\mathcal{O}(n^{-1/2})$ for standard MC the equivalent time value means we would have to increase sample size by roughly 222 to reduce the error to the size of the StackedMC one, which constitutes a nearly 15-fold improvement to the standard approach. Opposed to this, the additional StackedMC procedure applied on top only takes up only about half of the standard MC runtime. The results in table 1 and 2 show that at least in this simple setting the discussed post processing technique imposes a significant advantage in accuracy and runtime.

4.1.1 Dependency on model parameters

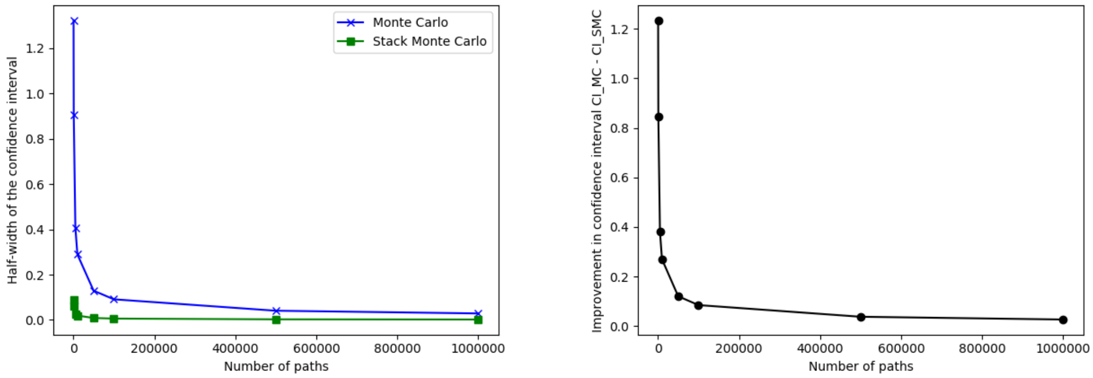
Every model or pricing procedure depends on its parameter settings, thus in this section we investigate how sample size, cross-validation and number of folds as well as choice of fitting model affect the StackedMC's performance.

Table 3 compares the results for the standard MC as well as the StackedMC for increasing sample size. Again, run time is scaled to 0.57 seconds as one unit of time.

In column one we see that MC estimate's variance is reduced as expected at a rate of $\mathcal{O}(n^{-1/2})$. Standard MC's run time scales linearly with increasing sample size so does equivalent time in the last column. While the ratio of improvement is roughly constant around 15-times the confidence interval's absolut reduction decreases as the number of draws grows, as seen in figure 2.

	MC		StackedMC		Improvement		Time	
Paths	CI	Time	CI	Time	Absolute	Ratio	Total	Equivalent
10^3	0.9059	0.01	0.0601	0.01	0.08	15.18	0.02	2.27
10^4	0.2895	0.10	0.0194	0.06	0.27	14.92	0.16	22.57
10^5	0.0913	1.00	0.0061	0.53	0.09	14.90	1.53	222.08
10^6	0.0288	10.15	0.0019	5.38	0.03	14.96	15.53	2271.72

Table 3: Sample size effect on variance reduction



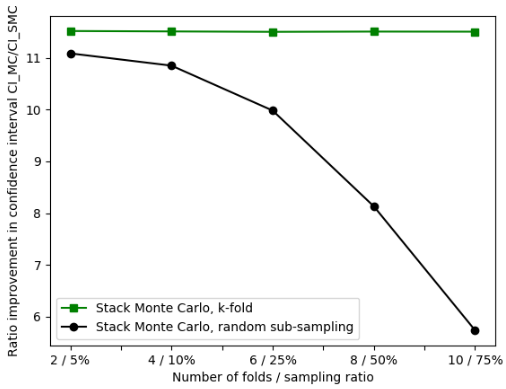
(a) Size of confidence interval

(b) Absolut confidence intervall reduction

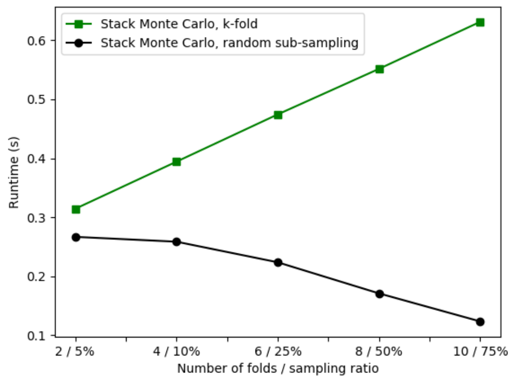
Figure 2: Sample size effect on variance reduction for MC and StackedMC

Figure 3 shows no significant improvement in variance reduction if the number of folds is increased, however, runtime grows rises linearly due to the additional time spent fitting more functions g_k. Compared to random sub-sampling where an increasing proportion of the sample is used as in-sample-set for fitting g and the rest as out-of-sample-set to calculate α, cross-validation in general offers greater variance reduction independent of the number of folds. Only at the smallest sample ratio of 5% the results of sub-sampling are comparable to the ones of cross validation.

Ultimately, different fitters for $g(.)$ are compared in figure 4. Polynomials of increasing order as well as a piecewise linear fitting function are used in the stacking procedure. The latter is obtained by filtering the zero-valued training data and fitting a linear function to the remaining points. Negative values of the fitter, i.e. values to the left of the intersection with the x-axis, are then set set to zero. For the polynomials the results show an increasing variance reduction up to order 8, especially from order 1 to 2 and with smaller increments thereafter. With an order higher than 8 the results worsen again, most likely due to overfitting of g to the training data. The piecewise linear function is at least as good as a polynomial of any order. Results of the two different fitters are most comparable for a polynomial of order 8, however a definite advantage of the partially linear one is the computational cost which is substantially lower.

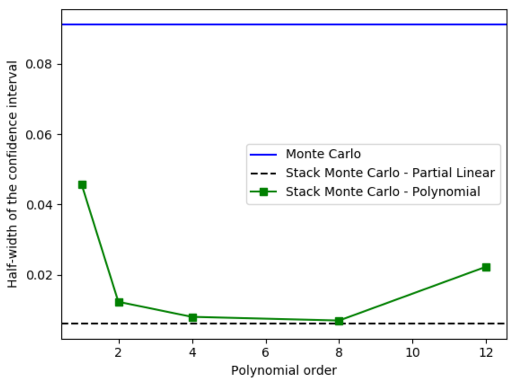


(a) Confidence interval reduction ratio

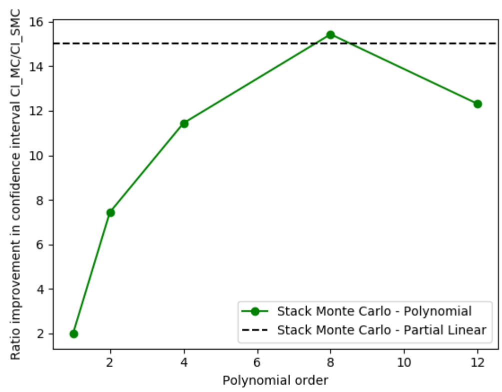


(b) Computation time of stacking procedure

Figure 3: Variance reduction by cross-validation – k-folds vs. random sub-sampling



(a) Size of confidence interval



(b) Confidence intervall reduction ratio

Figure 4: Variance reduction by choice of fitting model

4.2 Asian call option

In this section the StackedMC's performance for a path dependent option is examined. We consider an arithmetic Asian Option with maturity T and strike price K which's price is determined by some dates $(t_i)_{i=1:m}$ with $0 \le t_1 < t_2 < \ldots < t_m \le T$. Its payoff is given by

$$\Phi_T := \left(\frac{1}{m} \sum_{i=1}^{m} S_{t_i} - K \right)_+ .$$

Under no-arbitrage arguments with the risk-neutral measure the price of the option at $t = 0$ then reads as

$$e^{-rT} \mathbb{E}[\Phi_T].$$

To obtain the pricing function f to be estimated we use (19) and the independence of Gaussian increments in the Black-Scholes model to rewrite

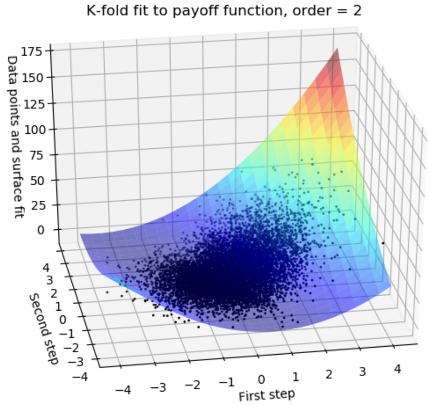
$$\begin{aligned} \sum_{i=1}^{m} S_{t_i} &= S_0 \sum_{i=1}^{m} \left(\frac{S_{t_i}}{S_{t_{i-1}}} \cdots \frac{S_{t_1}}{S_{t_0}} \right) = S_0 \sum_{i=1}^{m} \prod_{j=0}^{i} \frac{S_{t_j}}{S_{t_{j-1}}} \\ &= S_0 \sum_{i=1}^{m} \prod_{j=0}^{i} \frac{e^{\left(r - \frac{\sigma^2}{2}\right) t_j + \sigma W_{t_j}}}{e^{\left(r - \frac{\sigma^2}{2}\right) t_{j-1} + \sigma W_{t_{j-1}}}} = S_0 \sum_{i=1}^{m} \prod_{j=0}^{i} e^{\left(r - \frac{\sigma^2}{2}\right)(t_j - t_{j-1}) + \sigma (W_{t_j} - W_{t_{j-1}})} \qquad (22) \\ &= S_0 \sum_{i=1}^{m} \prod_{j=0}^{i} e^{\left(r - \frac{\sigma^2}{2}\right)(t_j - t_{j-1}) + \sigma \sqrt{t_j - t_{j-1}} X_j} =: h(X_1, \ldots, X_m) \end{aligned}$$

with $t_0 = 0$ and $(X_1, \ldots, X_m)$ is a Vektor of iid $\mathcal{N}(0, 1)$ random variables which correspond to the Brownian increments of the stock price. As Monte Carlo draw a matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ of iid Gaussian samples is simulated, where n again corresponds to the number of simulated stock price paths. Plugging the rows of $\boldsymbol{X}$ into function h from above we would obtain a standard MC estimate for the Asian Option's price as

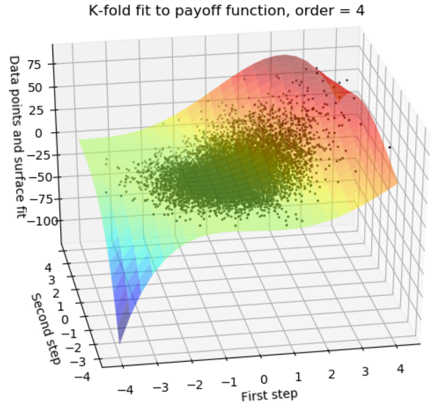
$$\bar{f} = \frac{1}{N} \sum_{i=1}^{n} \left(\frac{h(X_{i,1}, \ldots, X_{i,m}) - K}{m} \right)_+ , \qquad (23)$$

where the term after the sum is our pricing function, i.e. the function f in equation (1). Since each option price is now determined by m observations of the underlying stock price, the problem of fitting the control variate g has m dimensions. With the same parameters as in (21) polynomial surfaces of order 2 and 4 as well as a piecewise linear surface are fitted to the option prices obtained for each path of $\boldsymbol{X}$. The fitted surfaces are displayed in figure 5.

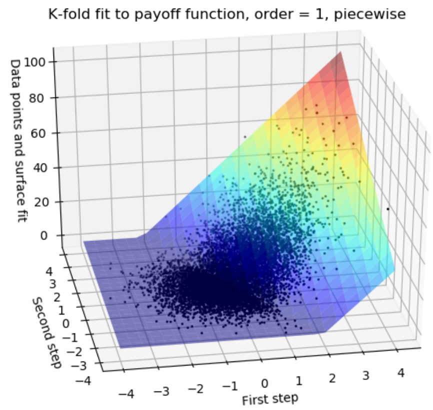
Tables 4, 5 and 6 present the results of the stacking procedure for an arithmetic Asian Option with increasing number of observation points t_i. What is most notable here is that while the time spent to compute the standard MC estimate only grows modestly with the number of observations, the runtime for fitting a polynomial surface to the data increases vastly. For $m = 50, 100$ a surface of order 4 was omitted due to its high computation time. This is where a piecewise linear fitter reveals its advantage offering a nearly 20-fold improvement in variance with only a fraction of additional computational cost.



(a) Polynomial order 2



(b) Polynomial order 4



(c) Piecewise linear

Figure 5: Variance reduction of Asian Option by fitter, M = 2

					Improvement	Time	
	Fit Model	Price	CI	Runtime	Ratio	Total	Equivalent
MC		8.1024	0.0698	21.36			
StackMC	Polynomial 2	8.1108	0.0097	1.23	7.18	22.59	1100.43
	Polynomial 4	8.1112	0.0064	1.54	10.97	22.90	2570.85
	Piecewise Linear	8.1117	0.0042	1.27	16.71	22.63	5963.16

Table 4: MC vs. StackedMC for an Asian Option with $m = 2$

					Improvement	Time	
	Fit Model	Price	CI	Runtime	Ratio	Total	Equivalent
MC		5.8532	0.0502	22.24			
StackMC	Polynomial 2	5.8582	0.0073	102.64	6.87	124.88	1049.08
	Piecewise Linear	5.8567	0.0025	3.10	19.88	25.34	8785.44

Table 5: MC vs. Stacked MC for an Asian Option with $m = 50$

					Improvement	Time	
	Fit Model	Price	CI	Runtime	Ratio	Total	Equivalent
MC		5.8209	0.0499	23.10			
StackMC	Polynomial 2	5.8100	0.0076	1149.01	6.60	1172.12	1005.32
	Piecewise Linear	5.8102	0.0025	5.59	19.84	28.69	9092.11

Table 6: MC vs. StackedMC for an Asian Option with $m = 100$

5 Conclusion

StackedMC is an easy to use an efficient post-processing technique to improve a standard Monte Carlo estimate. The procedure makes neither assumptions about the sampling process nor about the function f to be estimated. This implies that instead of a standard Monte Carlo draw a different sampling method can be used such as Importance Sampling or Markov Chain Monte Carlo and that it can be applied not only to smooth functions f but also to discontinuous or discrete valued ones. Furthermore, there are no assumptions made about g other than that it must predict out-of-sample values and that we want to be able to evaluate it analytically. This offers a wide spectrum of methods to find a good fit g.

Applied to the pricing of options in the Black-Scholes model the method shows notable advantages over the standard approach. In both considered cases the additional cost was negligible compared to the equivalent time needed for simple Monte Carlo to obtain similar precision. At the same time it offered significant variance reduction, measured as decrease in the confidence interval halfwidth, with a 15-fold improvement for the European option and an improvement of up to 20-fold for the Asian one. The choice of the fitting function g plays an important role in advantage of StackedMC. Here, a piecewise linear fitter proved to be a good choice both in terms of computational cost and precision which is especially visible with high dimensional fitting problems as is the case with the Asian option. While in this paper the procedure was only discussed in the Black-Scholes model it can also be applied to stochastic volatility models. For further results on this see [9].

References

[1] Juan Alonso, Brendan Tracey, and David Wolpert. Using supervised learning to improve monte carlo integral estimation. *AIAA Journal*, 51, 08 2011.

[2] Phelim P. Boyle. Options: A monte carlo approach. *Journal of Financial Economics*, 4(3):323 – 338, 1977.

[3] Bruno Casella and Gareth Roberts. Exact simulation of jump-diffusion processes with monte carlo applications. *Methodology and Computing in Applied Probability*, 13:449–473, 09 2011.

[4] Saso Džeroski and Bernard Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine Learning*, 54(3):255–273, 2004.

[5] Paul Glasserman. *Monte Carlo methods in financial engineering*. Applications of mathematics; 53. Springer, New York, NY [u.a.], 2003.

[6] Desmond Higham and Xuerong Mao. Convergence of monte carlo simulations involving the mean-reverting square root process. *J. Comput. Finance*, 8, 04 2005.

[7] John Hull. *Options, Futures and other Derivatives, Global Edition*. Pearson Education Limited, Harlow, EN, 9th edition, 2018.

[8] John C Hull and Alan White. The pricing of options on assets with stochastic volatilities. *Journal of Finance*, 42(2):281–300, 1987.

[9] Antoine Jacquier, Emma Malone, and Mugad Oumgari. Stacked monte carlo for option pricing. *SSRN Electronic Journal*, 2019.

[10] Yehuda Koren. The bellkor solution to the netflix grand prize. September 2009. `https://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf`.

[11] Georgios Sakkis, Ion Androutsopoulos, Georgios Paliouras, Vangelis Karkaletsis, Constantine D. Spyropoulos, and Panagiotis Stamatopoulos. Stacking classifiers for anti-spam filtering of e-mail. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, 2001.

[12] Jason J. Sharples and John C.V. Pezzey. Expectations of linear functions with respect to truncated multinormal distributions – with applications for uncertainty analysis in environmental modelling. *Environmental Modelling & Software*, 22(7):915 – 923, 2007.

[13] Kai Ming Ting and Ian H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10(1):271–289, May 1999.

[14] Brendan D. Tracey and David Hilton Wolpert. Reducing the error of monte carlo algorithms by learning control variates. *arXiv: Machine Learning*, 2016. Available at: https://arxiv.org/abs/1606.02261.

[15] David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241 – 259, 1992.

[16] Lu Xu, Jian-Hui Jiang, Yan-Ping Zhou, Hai-Long Wu, Guo-Li Shen, and Ru-Qin Yu. Mccv stacked regression for model combination and fast spectral interval selection in multivariate calibration. *Chemometrics and Intelligent Laboratory Systems*, 87(2):226 – 230, 2007.