Discretization Methods for Option Pricing

Black-Scholes Formula, Monte Carlo Method, Euler and Milstein Scheme for Numerical Solution of SDE

Hang Miao

1 Introduction

Black-Scholes-Merton model, since its inception in 1973, becomes the epitome of modern finance theory and still regarded as one of the best ways for pricing an options contract in the last few decade. Due to its simplicity and mild conditions, various methods for pricing path-dependent exotic option developed later could still be applied to pricing this earlier option model.

This paper aims to illustrate from scratch of six different approaches to price European style call and put option with a focus on four numerical discretizaiton methods: Euler Spot, Euler Log Spot, Milstein Spot and Milstein Log Spot. Euler Spot discretize the stochastic differential equation (SDE) of the underlying asset process using Euler scheme. Euler Log Spot discretizaiton method discretize the stochastic differential equation of the Log underlying asset process using Euler scheme. Milstein Spot method discretize the stochastic differential equation of the underlying asset process using Milstein scheme. Milstein Log Spot method discretize the stochastic differential equation of the Log underlying asset process using Milstein scheme. When the underlying asset process follows geometric Brownian motion, which is the case in this paper, Milstein Log Spot coincide with Euler Log Spot. These numerical discretizaiton methods approaches are wildely used and could be easily extended to price more complex and exotic options.

In order to check on the accuracy of these discretization methods, I also implemented two additional option pricing methods: Black-Scholes analytical formula and one-step Monte Carlo simulation. The first one use the original Black-Scholes formula but with a derivation using no-arbitrage pricing thoery and martingale theory proposed by Harrison and Kreps (1979) and Harrison and Pliska (1981). Closed-form formula for derivatives with other payoff functions (non-path dependent) could be easily derived using this approach. The other based on Monte Carlo techniques first proposed by Boyle (1977) with closed-form solution of SDE instead of numerical approximation of SDE using Euler or Milstein schemes in the previous four cases.

Each approach is implemented by C++ with zero dependency except its standard library. Prior to C++ 11, standard library had no Normal_distribution class that could draw random samples or compute cumulative density function. To address this issue, Box-Muler algorithm is applied to generate random samples from normal distribution; Hasting method is applied to approximate CDF of normal distribution for computational efficiency. The Black-Scholes formula and one-step Monte Carlo simulation method is also benchmarked by QuantLib using python interface.

The rest of paper is structured as follows: Section 2 lay out the foundation for the general derivative pricing formula under risk neutral measure. Section 3 discuss six different ways to pricing option: 3.1 derive the Black-Scholes formula using martingale theory; 3.2-3.5 use Monte Carlo simulation to approximate the expectation under risk neutral measure; 3.2 use the closed-form solution of geometric Brownian motion to form a one time step MC simulation; 3.3 use the Euler scheme to approximate the entire paths of numerical solution for the underlying SDE of spot process; 3.4 approximate the SDE of log spot process; 3.5 and 3.6 are 3.3 and 3.4 replaced Euler scheme by the Milstein scheme respectively. In practice, if the payoff function for the derivatives is not path dependent (like European Option in this paper), then there is no need to simulate the whole path like what we did in section 3.3-3.6. While If the payoff function is path dependent (like those continuously or discretely monitored Asian options) then we need to simulate the whole path. Simulate the entire path is the general method but computational inefficient for derivatives with non-path dependent payoff functions. Section 4 set up the initial parameters and gives the testing result which benchmarked by QuantLib. The error diagnostic and the convergence of MC method is provided in the end.

2 Model Setting

Suppose the underlying stock price follows Geometric Brownian with continuous dividend payment d.

$$\mathrm{d}S_t = (\alpha - d)S_t\mathrm{d}t \ + \ \sigma S_t \mathrm{d}W_t$$

The market money account risk free return is r. Let the market price of risk $\theta = \frac{\alpha - r}{\sigma}$, and apply Girsanov's Theorem to change the physical measure to risk neutral measure under which $\mathrm{d}\widetilde{W}_t = \theta \mathrm{d}t + \mathrm{d}W_t$ is a Brownian Motion. The Stock price under risk neutral measure is

$$\mathrm{d}S_t = (r - d)S_t\mathrm{d}t \ + \ \sigma S_t \mathrm{d}\widetilde{W}_t$$

The discounted portfolio of holding stock and money account is a martingale under risk neutral measure. Given any derivatives which payoff is $V(T)$ where $V(T)$ is $\mathcal{F}_T$-measurable. Applying martingale representation Theorem, we can find an adapted process $\Gamma(t)$ such that the short position of this derivative can be hedged. Under the no arbitrage condition, the price of the derivative should be the same as the value of the portfolio hedge the short position. Therefore,

$$D_t V_t = D_t X_t = \widetilde{\mathbb{E}}[D_T X_T | \mathcal{F}_t] = \widetilde{\mathbb{E}}[D_T V_T | \mathcal{F}_t]$$

Since r is constant, $D_t = \mathrm{e}^{-rt}$ is a deterministic independent of $\mathcal{F}_t$. $V_T = (S_T - K)^+$ for European Call Option. $V_T = (K - S_T)^+$ for European Put Option. So we have

$$C_t = \mathrm{e}^{-r(T-t)}\widetilde{\mathbb{E}}[(S_T - K)^+ | \mathcal{F}_t]$$

$$P_t = \mathrm{e}^{-r(T-t)}\widetilde{\mathbb{E}}[(K - S_T)^+ | \mathcal{F}_t]$$

3 Pricing the European Option

3.1 Black-Scholes Formula

The closed form solution for the stock price SDE under risk neutral measure

$$\mathrm{d}S_t = (r-d)S_t\mathrm{d}t \ + \ \sigma S_t \mathrm{d}\widetilde{W}_t$$

is

$$S_t = S_0 \mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\widetilde{W}_t} = S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}}, \quad \text{where } \mathbf{Z} \sim \text{Normal}(0,1)$$

Substitute into the risk neutral pricing formula above, we have

$$C_0 = \mathrm{e}^{-rt}\widetilde{\mathbb{E}}[(S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}} - K)^+]$$

$$= \mathrm{e}^{-rt}\int_{-\infty}^{d_-}(S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}z} - K)\frac{1}{2\pi}\mathrm{e}^{-\frac{z^2}{2}}\mathrm{d}z$$

The calculation of the integral lead us to the formula for the call option

$$C_{0,t} = \mathrm{e}^{-rt}[FN(d_+) - KN(d_-)]$$

where $F = S_0\mathrm{e}^{(r-d)t}$, $d_\pm = \frac{1}{\sigma\sqrt{t}}\left[\log\frac{S_0}{K} + (r-d\pm\frac{1}{2}\sigma^2)t\right]$ By the same argument the formula for European Put option is

$$P_{0,t} = \mathrm{e}^{-rt}[-FN(-d_+) + KN(-d_-)]$$

plug into all the given parameters and use the "CumulativeNormal(double x)" function in the "Normals.h" to compute $N(\pm d_\pm)$ we have the option price calculated via explicit formula written in file "BSMformula.h" and "BSMformula.cpp".

3.2 Monte Carlo Simulation with Closed-Form SDE Solution

From the Black-Scholes model assumption, underlying spot SDE follows Geometric Brownian Motion, which has the following closed-form solution:

$$S_t = S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\widetilde{W}_t} = S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}}, \quad \text{where } \mathbf{Z} \sim \text{Normal}(0,1)$$

Plug into the general risk neutral formula for option we have:

$$C_0 = \mathrm{e}^{-rt}\widetilde{\mathbb{E}}[(S_t-K)^+] = \mathrm{e}^{-rt}\widetilde{\mathbb{E}}[(S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}} - K)^+]$$

$$P_0 = \mathrm{e}^{-rt}\widetilde{\mathbb{E}}[(K-S_t)^+] = \mathrm{e}^{-rt}\widetilde{\mathbb{E}}[(K - S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}})^+]$$

According to the strong law of large number, the average of a sequence of random variable converge almost surly to the expected value. So we can estimate $\widetilde{\mathbb{E}}[(S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}} - K)^+]$ by simulate a large number of $S_0\mathrm{e}^{(r-d-\frac{1}{2}\sigma^2)t+\sigma\sqrt{t}\mathbf{Z}} - K)^+$ then take the average. This is also called one time step Monte Carlo simulation. $\mathbf{Z}$ is generated by "GetOneGaussianByBoxMuller()" from "random1.h" (If C++ 11 or later, normal_distribution should be used, since the Marsaglia polar method is faster than Box Muller). The price is calculated in the file "simpleMonteCarlo.h" and "simpleMonteCarlo.cpp".

3.3 Monte Carlo Simulation with Euler Scheme Numerical Solution for SDE of Spot Process

The spot SDE under risk neutral measure is

$$\mathrm{d}S_t = (r-d)S_t\mathrm{d}t \ + \ \sigma S_t \mathrm{d}\widetilde{W}_t$$

The Euler discretization scheme for the approximation of the above spot SDE is

$$S_{t+1} = S_t + (r-d)S_t h + \sigma S_t \sqrt{h}\mathbf{Z}, \ t = 0, \cdots 252$$

where $S_0 = 100, \ h = \frac{1}{252}, \ \mathbf{Z} \sim \text{Normal}(0,1)$ Use the Stock price at the last step S_{252} in place of S_T for the formula for Vanilla Option

$$C_0 = \mathrm{e}^{-r(T-0)}\widetilde{\mathbb{E}}[(S_T - K)^+ | \mathcal{F}_0] = \mathrm{e}^{-rT}\widetilde{\mathbb{E}}[(S_{252} - K)^+]$$

Use the Monte Carlo simulation to approximate $\widetilde{\mathbb{E}}[(S_{252}-K)^+]$ i.e obtain 10000 S_{252} by generating 10000 path of stock using Euler Scheme. Then compute the average of $(S_{252} - K)^+$. Same argument is used for put option. The option price is calculated in the file "EulerSchemeMC.h" and "EulerSchemeMC.cpp"

3.4 Monte Carlo Simulation with Euler Scheme Numerical Solution for SDE of Log Spot Process

Applying Ito's formula, the Log spot SDE under risk neutral measure is

$$\begin{aligned}
\mathrm{d}\log S_t &= \frac{1}{S_t}\mathrm{d}S_t - \frac{1}{2}\frac{1}{S_t^2}\mathrm{d}\langle S_t^c, S_t^c\rangle \\
&= (r-d)\mathrm{d}t \ + \ \sigma \mathrm{d}\widetilde{W}_t - \frac{1}{2}\frac{1}{S_t^2}\sigma^2 S_t^2 \mathrm{d}t \\
&= (r - d - \frac{1}{2}\sigma^2)\mathrm{d}t \ + \ \sigma \mathrm{d}\widetilde{W}_t
\end{aligned}$$

The Euler Scheme for the approximation of SDE for log spot is

$$\log S_{t+1} = \log S_t + (r - d - \frac{1}{2}\sigma^2)h + \sigma\sqrt{h}\mathbf{Z}, \ t = 0, \cdots 252$$

where $\log S_0 = \log(100), \ h = \frac{1}{252}, \ \mathbf{Z} \sim \text{Normal}(0,1)$ Use the exponential of the log Stock price at the last step $\exp(\log S_{252})$ in place of S_T for the formula for Vanilla Option

$$C_0 = \mathrm{e}^{-r(T-0)}\widetilde{\mathbb{E}}[(S_T - K)^+ | \mathcal{F}_0] = \mathrm{e}^{-rT}\widetilde{\mathbb{E}}[(\exp(\log S_{252}) - K)^+]$$

Use the Monte Carlo simulation to approximate $\widetilde{\mathbb{E}}[(\exp(\log S_{252}) - K)^+]$ i.e obtain 10000 $\log S_{252}$ by generating 10000 path of stock using Euler Scheme. Then take the exponential for each of them. Compute the average of $(\exp(\log S_{252}) - K)^+$. Same argument is used for put option. The option price is calculated in the file "logEulerSchemeMC.h" and "logEulerSchemeMC.cpp"

3.5 Monte Carlo Simulation with Milstein Scheme Numerical Solution for SDE of Spot Process

The spot SDE under risk neutral measure is

$$\begin{aligned} \mathrm{d}S_t &= (r-d)S_t\mathrm{d}t \ + \ \sigma S_t \mathrm{d}\widetilde{W}_t \\ &\equiv \alpha(S_t)\mathrm{d}t \ + \ \beta(S_t)\mathrm{d}\widetilde{W}_t \end{aligned}$$

where $\alpha(S_t) = (r-d)S_t$ and $\beta(S_t) = \sigma S_t$.
The Milstein Scheme for the approximation of SDE for spot is

$$\begin{aligned} S_{t+1} &= S_t + \alpha(S_t)h + \beta(S_t)\Delta\widetilde{W}_t + \frac{1}{2}\beta(S_t)\frac{\partial\beta(S_t)}{\partial S_t}\Big((\Delta\widetilde{W}_t)^2 - h\Big) \\ &= S_t + (r-d)S_t h + \sigma S_t\sqrt{h}\mathbf{Z} + \frac{1}{2}\sigma^2 S_t h[\mathbf{Z}^2 - 1], \ t = 0, \cdots 252 \end{aligned}$$

where $S_0 = 100, \ h = \frac{1}{252}, \ \mathbf{Z} \sim \text{Normal}(0,1)$ Use the Stock price at the last step S_{252} in place of S_T for the formula for Vanilla Option

$$C_0 = \mathrm{e}^{-r(T-0)}\widetilde{\mathbb{E}}[(S_T - K)^+|\mathcal{F}_0] = \mathrm{e}^{-rT}\widetilde{\mathbb{E}}[(S_{252} - K)^+]$$

Use the Monte Carlo simulation to approximate $\widetilde{\mathbb{E}}[(S_{252}-K)^+]$ i.e obtain 10000 S_{252} by generating 10000 path of stock using Milstein Scheme. Then compute the average of $(S_{252} - K)^+$. Same argument is used for put option. The option price is calculated in the file "MilsteinSchemeMC.h" and ""MilsteinSchemeMC.cpp"

3.6 Monte Carlo Simulation with Milstein Scheme Numerical Solution for SDE of Log Spot Process

Milstein scheme coincide with Euler scheme for log spot process in section 3.4. Applying Ito's formula, the Log spot SDE under risk neutral measure is

$$\begin{aligned} \mathrm{d}\log S_t &= \frac{1}{S_t}\mathrm{d}S_t - \frac{1}{2}\frac{1}{S_t^2}\mathrm{d}\langle S_t^c, S_t^c\rangle \\ &= (r - d - \frac{1}{2}\sigma^2)\mathrm{d}t \ + \ \sigma\mathrm{d}\widetilde{W}_t \\ &\equiv \alpha(S_t)\mathrm{d}t \ + \ \beta(S_t)\mathrm{d}\widetilde{W}_t \end{aligned}$$

where $\alpha(S_t) = r - d - \frac{1}{2}\sigma^2$ and $\beta(S_t) = \sigma$.
The Milstein Scheme for the approximation of SDE for log spot is

$$\begin{aligned} \log S_{t+1} &= S_t + \alpha(S_t)h + \beta(S_t)\Delta\widetilde{W}_t + \frac{1}{2}\beta(S_t)\frac{\partial\beta(S_t)}{\partial S_t}\Big((\Delta\widetilde{W}_t)^2 - h\Big) \\ &= \log S_t + (r - d - \frac{1}{2}\sigma^2)h + \sigma\sqrt{h}\mathbf{Z}, \ t = 0, \cdots 252 \end{aligned}$$

Which is exactly same as the Euler scheme for Log spot. The rest of the details follows in section 3.4.

4 Program Testing Result

4.1 Output Result

By using the above initial parameters, the testing program price the option using Black-Scholes formula, Monte Carlo method with Closed-Form solution of SDE, MC with Euler approximation of Spot SDE, Log Spot SDE, and MC with Milstein scheme Spot and Log Spot SDE. The associated call and put price are as follow

Table 1: Option Pricing Results. Spot price $S_0 = 100$, risk free interest rate $r = 0.05$, dividend rate $d = 0.02$, annualized volatility $\sigma = 0.3$, strike $K = 110$ expire in one year $T = 1$, number of observation dates steps $= 252$, number of simulated path nPaths $= 10,000$.

	Black-Scholes	Closed-Form	Euler Spot	Euler Log Spot	Milstein Spot	Milstein Log Spot
Call	9.05705	9.14619	9.12482	9.02136	9.09699	9.02136
Put	15.6724	15.6426	15.6205	15.5473	15.6371	15.5473

4.2 Benchmark Result

QuantLib is used to benchmark the above result with the same initial parameter in section 4.1. The Black-Scholes Formula gives the same result. One step MC and multistep gives very close result. Only one step MC and Euler spot is implemented since QuantLib doesn't support Log spot process. QuantLib Milstein scheme (in 'ExtendedBlackScholesProcess') currently lies in experimental directory which not belong to the stable version.

Table 2: Benchmark with QuantLib. Spot price $S_0 = 100$, risk free interest rate $r = 0.05$, dividend rate $d = 0.02$, annualized volatility $\sigma = 0.3$, strike $K = 110$ expire in one year $T = 1$, number of observation dates steps $= 252$, number of simulated path nPaths $= 10,000$.

	Black-Scholes	Closed-Form SDE	Euler Spot
Call	9.057062	9.071492	9.024224
Put	15.672431	15.633974	15.714965

4.3 Convergence Diagnostics

Given the Number of path parameter (Npaths), by running in different seeds we can calculate the Variance of the option price. Then we could compute the relationship between Error and Npaths as follows

$$\text{Error} = \sqrt{\frac{\text{Variance}}{nPaths}}.$$

In program, NPaths is choosed as a Fibonacci sequence starting from $n_1 = 50$, $n_2 = 150$ to $n_{12} = 76800$. Given each NPaths, seed changed 50 times using "srand" to produce the Error and mean under each NPaths. The final result is in the table attached below. The relationship of option Error and the NPaths is also ploted as follow. We can see that as nPaths increases, the error decreases due the law of large number.

Table 3: Mean of Call Option

NPaths	Closed-Form	EulerSpot	EulerLogSpot	MilsteinSpot	MilsteinLogSpot
50	12.4278	13.0146	7.71552	9.39581	7.71552
150	7.54791	9.23363	8.25299	8.92651	8.25299
300	9.68846	9.01617	8.49866	9.27559	8.49866
600	9.31149	9.08261	8.70454	8.99807	8.70454
1200	9.07024	8.94621	9.0425	9.11973	9.0425
2400	9.00549	8.98091	9.18603	8.97183	9.18603
4800	9.00299	9.07668	9.09096	9.00039	9.09096
9600	9.06168	9.08271	9.11163	9.10058	9.11163
19200	9.04057	9.07641	9.06883	9.04919	9.06883
38400	9.0928	9.0562	9.05618	9.05822	9.05618
76800	9.06445	9.04656	9.06396	9.03481	9.06396

Table 4: Error of Call Option

NPaths	Closed-Form	EulerSpot	EulerLogSpot	MilsteinSpot	MilsteinLogSpot
50	0.48198	0.562477	0.190098	0.171951	0.190098
150	0.152892	0.0963186	0.0673836	0.0850447	0.0673836
300	0.0545372	0.0477321	0.0323567	0.048569	0.0323567
600	0.0297463	0.0178786	0.0272778	0.0435449	0.0272778
1200	0.0120253	0.0193152	0.0122964	0.0148295	0.0122964
2400	0.00640163	0.00651427	0.00813375	0.00715684	0.00813375
4800	0.00359834	0.00319437	0.00378622	0.00420864	0.00378622
9600	0.00240123	0.00170416	0.00186497	0.00188511	0.00186497
19200	0.000835645	0.00107828	0.000916756	0.000849735	0.000916756
38400	0.000464014	0.000451622	0.000435038	0.000415189	0.000435038
76800	0.000288675	0.000266927	0.000226643	0.00024831	0.000226643

Table 5: Mean of Put Option

NPaths	Closed-Form	EulerSpot	EulerLogSpot	MilsteinSpot	MilsteinLogSpot
50	17.2901	16.2067	16.2685	13.8681	16.2685
150	14.718	14.5519	15.0634	16.3866	15.0634
300	15.4324	16.5315	16.0065	15.196	16.0065
600	15.6692	15.8932	16.527	15.5531	16.527
1200	15.6897	15.6545	15.6339	15.5757	15.6339
2400	15.6721	15.722	15.6459	15.6643	15.6459
4800	15.7118	15.652	15.6448	15.7561	15.6448
9600	15.6619	15.6805	15.6769	15.6872	15.6769
19200	15.6571	15.6676	15.7009	15.6974	15.7009
38400	15.6632	15.681	15.6743	15.669	15.6743
76800	15.6744	15.6789	15.6749	15.6925	15.6749

Table 6: Error of Put Option

NPaths	Closed-Form	EulerSpot	EulerLogSpot	MilsteinSpot	MilsteinLogSpot
50	0.253936	0.0818879	0.143639	0.359711	0.143639
150	0.0905469	0.0954826	0.0722399	0.115937	0.0722399
300	0.0314607	0.0547219	0.0213896	0.100056	0.0213896
600	0.0123601	0.0210026	0.0419954	0.0336913	0.0419954
1200	0.011188	0.0133212	0.0141825	0.00560861	0.0141825
2400	0.00577265	0.00849157	0.00572551	0.00767946	0.00572551
4800	0.0027956	0.00398886	0.00380689	0.00426739	0.00380689
9600	0.00150727	0.00167457	0.00175146	0.00180954	0.00175146
19200	0.000719603	0.000768757	0.000812548	0.000917328	0.000812548
38400	0.000505321	0.000477189	0.000370818	0.000454732	0.000370818
76800	0.00018432	0.000247371	0.000207585	0.000227719	0.000207585

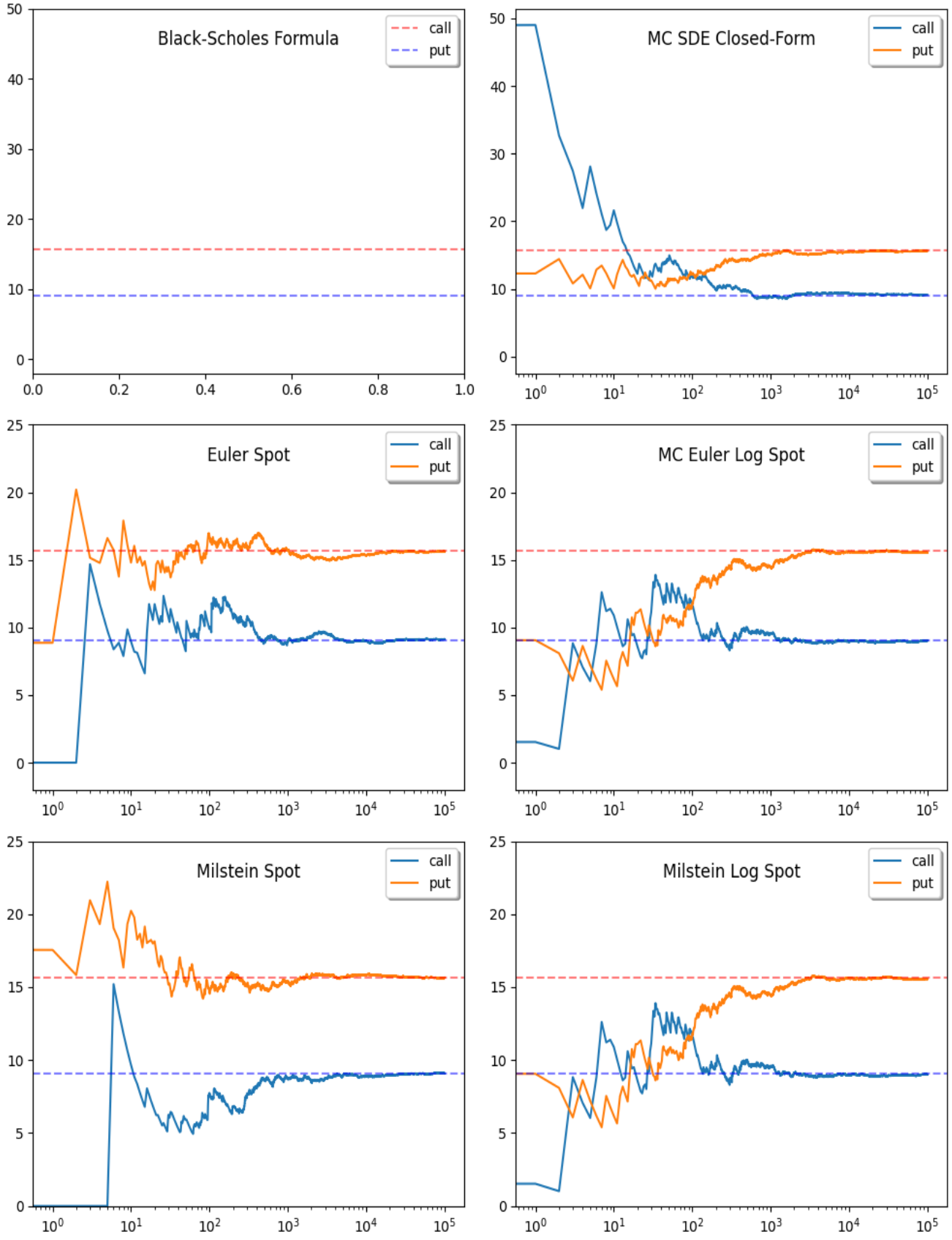

Figure 1: Convergence Diagnostics for Mean

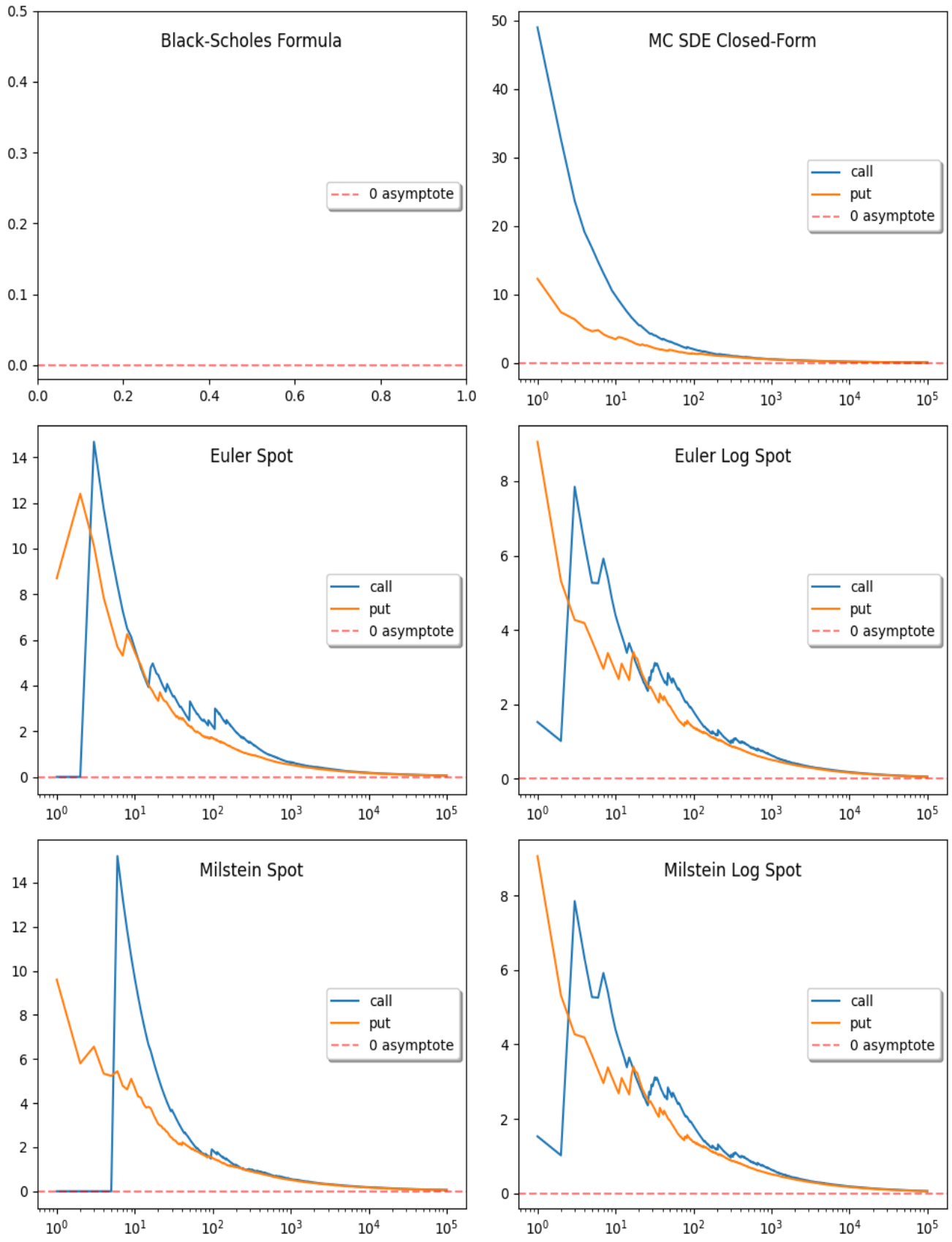

Figure 2: Convergence Diagnostics for Standard Deviation

References

Boyle, P., Broadie, M., and Glasserman, P. (1997). Monte carlo methods for security pricing. Journal of economic dynamics and control, 21(8-9):1267–1321.

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

Broadie, M. and Glasserman, P. (1996). Estimating security price derivatives using simulation. Management science, 42(2):269–285.

Curran, M. (1994). Valuing asian and portfolio options by conditioning on the geometric mean price. Management science, 40(12):1705–1711.

Fishman, G. S. and Moore, III, L. R. (1986). An exhaustive analysis of multiplicative congruential random number generators with modulus 2ˆ31-1. SIAM Journal on Scientific and Statistical Computing, 7(1):24–45.

Fusai, G. and Kyriakou, I. (2016). General optimized lower and upper bounds for discrete and continuous arithmetic asian options. Mathematics of Operations Research, 41(2):531–559.

Geman, H. and Yor, M. (1993). Bessel processes, asian options, and perpetuities. Mathematical finance, 3(4):349–375.

Glasserman, P. (2004). Monte Carlo methods in financial engineering, volume 53. Springer.

Harrison, J. and Kreps, D. M. (1979). Martingales and arbitrage in multiperiod securities markets. Journal of Economic Theory, 20(3):381–408.

Harrison, J. and Pliska, S. R. (1981). Martingales and stochastic integrals in the theory of continuous trading. Stochastic Processes and their Applications, 11(3):215–260.

Horvath, A. and Medvegyev, P. (2016). Pricing asian options: a comparison of numerical and simulation approaches twenty years later. Journal of Mathematical Finance, 6:810–841.

Joshi, M. S. and Joshi, M. S. (2004). C++ design patterns and derivatives pricing, volume 1. Cambridge University Press.

Kemna, A. G. and Vorst, A. C. (1990). A pricing method for options based on average asset values. Journal of Banking & Finance, 14(1):113–129.

L'ecuyer, P. (1988). Efficient and portable combined random number generators. Communications of the ACM, 31(6):742–751.

Lehmer, D. H. (1951). Mathematical methods in large-scale computing units. Annu. Comput. Lab. Harvard Univ., 26:141–146.

Park, S. K. and Miller, K. W. (1988). Random number generators: good ones are hard to find. Communications of the ACM, 31(10):1192–1201.

Rubinstein, M. and Reiner, E. (1991). Breaking down the barriers. 4(8):28–35.

Shreve, S. E. et al. (2004). Stochastic calculus for finance II: Continuous-time models, volume 11. Springer.

Thompson, G. W. P. (1999). Fast narrow bounds on the value of asian options. Technical report.

Turnbull, S. M. and Wakeman, L. M. (1991). A quick algorithm for pricing european average options. Journal of financial and quantitative analysis, 26(3):377–389.