M.Phil. in Mathematical Finance

Numerical Methods in Finance 2020

Tutorial 8

Problems

1. Consider an arithmetic average rate call option with the following payoff

$$\max\left(\frac{1}{N+1}\sum_{i=0}^{N} S_i(\boldsymbol{Z}) - K, 0\right),$$

where $S_i(\boldsymbol{Z})$ is a GBM stock price process generated at times ih, with time interval $h = T/N$, using $\boldsymbol{Z} \sim \mathcal{N}_N(0, I)$, N is the number of averaging intervals and K is the strike price. Use the control variate technique to estimate the price of this option using the following parameters:

$$T = 1, \qquad S_0 = 100, \qquad \sigma = 45\%, \qquad r = 12\%, \qquad K = 50, \qquad N = 6.$$

As an effective control variate use the price of a geometric average rate call option, which has the following payoff function at maturity

$$\max\left(\left(\prod_{i=0}^{N} S_i(\boldsymbol{Z})\right)^{\frac{1}{N+1}} - K, 0\right),$$

and may be evaluated in closed-form as

$$c = \exp(-rT)\left[S_0 \exp\left(\underline{\mu} + \frac{1}{2}\underline{\sigma}^2\right)\Phi\left(\frac{\log\frac{S_0}{K} + \underline{\mu}}{\underline{\sigma}} + \underline{\sigma}\right) - K\Phi\left(\frac{\log\frac{S_0}{K} + \underline{\mu}}{\underline{\sigma}}\right)\right],$$

where Φ is the standard normal cumulative distribution function,

$$\underline{\mu} = \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2} \qquad \text{and} \qquad \underline{\sigma} = \sqrt{\frac{\sigma^2 T}{6}\left(\frac{T}{T+h} + 1\right)}.$$

Find an estimate for α using a sample of 1000 stock price paths, then, as a function of sample size, $n = 1000, 2000, \ldots, 50000$, plot raw Monte Carlo estimates for the geometric and arithmetic options, as well as the variance reduced estimate for the arithmetic option and a line indicating the closed form solution for the geometric option.

For debugging purposes, initialize the seed at the beginning of your program with the Matlab command `rng(0);` and generate the sample set for α-estimation using `randn(N,1000)` (Do this once before your main loop). In the loop that cycles through sample size, generate the normal random numbers with the command `randn(N,n)`. As far as possible vectorize your code.

Once you have working code, produce three-standard deviation bounds for the crude and control variate estimates of the arithmetic option. Plot these bounds around the best estimate for the control variate solution (i.e., the one for which you used a 50 000 sample size).

2. For a standard call option, with the parameters

$$T = 2, \qquad S_0 = 100, \qquad \sigma = 20\%, \qquad r = 10\%, \qquad K = 100,$$

compute and plot crude Monte Carlo estimates for the option price as a function of sample sizes in the range $n = 1000, 2000, \ldots, 50000$. Compute and plot the Black-Scholes price using

$$\text{BS} = S_0\Phi(d_1) - e^{-rT}K\Phi(d_2) \qquad \text{with} \qquad d_1 = \frac{\log(S_0/K) + \left(r + \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}} \qquad \text{and} \qquad d_2 = d_1 - \sigma\sqrt{T}.$$

Plot three-standard deviation bounds for the Monte Carlo estimates around this exact value. For debugging purposes, initialize the seed at the beginning of your program with the Matlab command `rng(0);`.

Now, using the first half of the normal random numbers (at each step in the loop that cycles through sample size), compute an antithetic Monte Carlo estimate for the option. To ensure an equitable comparison between the crude and antithetic estimates, plot these as a function of the full sample size for the crude Monte Carlo estimates (since the antithetic estimates consume roughly the same computational time). Plot three-standard-deviation bounds for the antithetic estimates. Note that the variance for the antithetic estimator is scaled by a factor of $n/2$ (not n).