M.Phil. in Mathematical Finance

Numerical Methods in Finance 2020

Tutorial 6

Problems

1. Consider the double integral

$$\int_{y_{\min}}^{y_{\max}} \int_{x_{\min}}^{x_{\max}} \sin xy \, dx \, dy.$$

Rewrite the integral in such a way that it can be evaluated as a Monte Carlo estimate using two dimensional uniform random variates. (Hint: either write down an expression for the density $w(x, y)$ for the current limits of integration, or perform a change of variables so that the limits of integration are 0 and 1 for both integrals, in which case $w(x, y) = \mathbb{I}_{[0,1)}(x)\mathbb{I}_{[0,1)}(y)$.)

Using

$$x_{\min} = 2, \qquad x_{\max} = 4, \qquad y_{\min} = 1, \qquad y_{\max} = 3,$$

write a Matlab program which uses crude Monte Carlo integration to plot estimates of the above integral as a function of sample size, for sample sizes $n = 1000, 2000, \ldots, 50000$. Use the Matlab function `integral2` to find a quadrature based estimate for the integral and, in conjunction with this estimate, plot upper and lower three-standard-deviation error bounds for the Monte Carlo estimates against sample size. For debugging purposes, initialize the seed at the beginning of your program with the command `rng(0);` and generate the uniform random numbers on $[0, 1)$ with the command `rand(2,n);` scaling them as appropriate.

2. Consider a geometric average rate asian call option with the following discounted payoff function

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

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, r is the risk-free rate and K is the strike price.

Write a Matlab program that uses the crude Monte Carlo method to plot estimates of the (risk-neutral) price for the option

$$\hat{c} = \frac{1}{n} \sum_{j=1}^{n} f^{\text{geom}}(\boldsymbol{Z}_j)$$

as a function of sample size, for sample sizes $n = 1000, 2000, \ldots, 50000$. Remember that each $S^{(j)}$ is a GBM process simulated at N intervals over $t \in [0, T]$. Use the following parameters:

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

Use the following closed-form formula to get the exact price the geometric average rate European call option

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

where Φ is the standard normal cumulative distribution function,

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

In conjunction with this price, plot upper and lower three-standard-deviation error bounds for the Monte Carlo estimates against sample size.

For debugging purposes, initialize the seed at the beginning of your program with the Matlab command `rng(0);` and, 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.