Module 6: Monte Carlo question (Mike Giles)

The aim in the mini-assignment is to compute delta and vega for standard, Asian and down-and-out barrier calls using an adjoint (or "reverse") implementation of the pathwise sensitivity technique, as presented in one of the Module 6 lectures.

The code is to be validated by comparing to a standard ("forward") implementation of the pathwise sensitivity method, and also to "bumping" using very small changes in S_0 and σ.

(i) Use an Euler approximation to Geometric Brownian Motion, so that

$$S_{n+1} = S_n + r\,S_n\,\Delta t + \sigma\,S_n\,\Delta W_n.$$

with $S_0 = 100, r = 0.05, \sigma = 0.2$, maturity $T = 1$, and 20 timesteps.

The discounted payoff for a single path for a standard call with a strike of $K = 60$ is $\exp(-rT)\max(0, S_N - K)$.

Write code to compute delta and vega for a single in-the-money path, and validate the code by comparing the results to those obtained through bumping using $\Delta S_0 = 10^{-4}$ and $\Delta\sigma = 10^{-6}$.

Next, write code to compute delta and vega by the adjoint method presented in lectures. The agreement should be perfect to the level of machine accuracy.

(ii) Write a new code which does the same for an Asian call option in which the payoff is $\exp(-rT)\max(0, A_N - K)$, with the average A_N defined by the recursion

$$A_0 = 0, \quad A_{n+1} = A_n + S_n\,\Delta t.$$

Note that this is written in a way so that (S_n, A_n) can be viewed as a expanded state vector U_n, so that we have

$$U_{n+1} = f(U_n)$$

which puts things in the form presented in lectures.

Again implement the standard pathwise analysis and validate it using "bumping", and then develop the corresponding adjoint code.

(iii) The third test-case is a down-and-out barrier option, with barrier $B = 90$, for which the single path discounted payoff is

$$\exp(-rT)\,\max(0, S_N - K)\;P_N$$

where P_n is the (approximate) probability of not having crossed the barrier which is given by the following recursion which comes from the Monte Carlo lecture in Module 4:

$$P_0 = 1, \quad P_{n+1} = P_n \left\{ 1 - \exp\left(-\,\frac{2\,\max(0, S_n - B)\max(0, S_{n+1} - B)}{\sigma^2 S_n^2 \Delta t} \right) \right\}.$$

Carry out the same steps as for (ii). The details are more complicated / tedious, but the overall approach and structure of the code will be exactly the same.

Please hand in:

- A writeup of the maths involved in each of the three cases, perhaps about half a page for each. There is no need to reproduce any of the material from the lecture – all I'm interested in is what the mathematics from the lecture means in the context of these 3 applications.
- Your code, written in C, C++ or MATLAB – if you would like to do it in some other language please talk to me first.
- Evidence of your 3-way validation: bumping, standard pathwise, adjoint pathwise. The latter two should agree perfectly (to machine precision), while the first two will differ slightly.

(Note: I am not concerned at all here with the computational efficiency of your code. I am only concerned that you understand the mathematics and demonstrate that by obtaining the correct answers. These are model test cases – in practice, you would never use adjoints for these because we are not computing enough sensitivities for it to be the most efficient approach. The real benefits come in applications like interest rate products where there is a need to compute the sensitivity with respect to a large number of forward rates.)