Barrier Options Pricing Under Local Volatility - User Instructions

Matthew Knowles

January 202

1 Inputting Data

Each script takes a set of inputs that it requires to compute the price of options. It is esiest to define the parameters at the top of the "main.m" script, where a set of sample values is already given. The first set of data is specific to the underlying stock and the option thereof. This includes things like the strike and barrier price, as well risk-free rate and dividend yield. There are also the simmulation parameters N, M, Nmc and T that must be inputted.

2 Pricing Options

There are two ways to price options using this project. The easiest way is to simply run the "main.m" script. This will use the data inputted by the user at the top and call each of the functions to price an option using the three FDM schemes, and a monte-carlo simmulation. All prices are then outputted to the console window.

Should you wish to price an option using just one method, simply call that method, passing in the input data as appropriate. Alternatively, comment out the ones you don't wish to use in the "main.m" file.

3 Brief Mathematical Background

Since for the up-and-out call if $S_j > B$, the value is 0, and if $S_j < K$ the value is zero, we used Neumann boundary conditions for this project.

3.1 Explicit Scheme

As mentioned, for the Explicit FDM scheme, Neumann boundary conditions were used. For this reason, the equation used for updating from step k to $k+1$ is

$$V_{K+1} = AE_k V_k + \begin{pmatrix} L_1(2V_1 - V_2) \\ 0 \\ \vdots \\ 0 \\ U_N(2V_N - V_{N-1}) \end{pmatrix}$$

The AE_k matrix is calculated in matlab by using the `diag` function to create the 3 main diagonals as individual NxN matrices which are then summed to give AE_k.

3.2 Implicit Scheme

The implicit scheme is slightly more involved, using Neumann boundary conditions gives us the following relationship:

$$V_k = AI_{k+1}V_{k+1}.$$

Since V_k and AI_{k+1} are known, and the fact AI_{k+1} is tridiagonal, we can simply employ the tridiagonal solving algorithm to calculate V_{k+1}.

3.3 Crank Nicholson

For the CN scheme, we used the original Dirichlet boundary conditions due to it being relatively easy to calculate. We have the AE_k matrix calculated in the same way as before. We then use MATLAB's "eye(N)" function to obtain the term $AE_k + I$. We then calculate the boundary to get the term:

$$V_{RHS} = (AE_k + I)V_k + \begin{pmatrix} L_{2,k}V_{1,k} - \hat{L} - 2, k + 1V_{1,k+1} \\ 0 \\ \vdots \\ 0 \\ U_{N,k}V_{N+1,k} - \hat{U}_{N,k+1}V_{N+1,k+1} \end{pmatrix}.$$

This term is on the right hand side of the CN equation, with $(AI_k + I)V_{k+1}$ on the left hand side. Now the V_{RHS} term is a column vector much like V_{k+1}, and $(AI_k + I)$ is a known tridiagonal matrix, so we again apply the solving algorithm to obtain V_{k+1}.