Assignment

Base case is Black-Scholes with parameters:

$$T = 5, K = 1.025, S(0) = 1, r = 0.04, \mu = -0.03, \sigma = 0.2, n = 100, \theta = 0.5$$

The questions 1-4 can be answered using your kBlack::fdRunner().

Do not use winding or smoothing unless stated.

1/ Find the order of convergence for the implied volatility of a European call option as function of the time step length, Δt, for the three schemes $\theta = 0,1,0.5$. Use number of time steps $m = 10, 25, 50, 75, 100, 150, 200, 400, 800, 1600, 3200, 6400$ as data for this exercise. Consider your results.

2/ Assume that the early exercise premium, i.e. the difference between American and European option prices, converges at a rate of $O(\Delta t^a)$ and estimate a for the scheme $\theta = 0.5$. You can use the same set of time steps as in 1. Modify the code to return the early exercise boundary. Consider your findings.

3/ Fix the grid and consider the value of the digital call option. Change the strike over the values $K = 1.01, 1.02, \ldots, 1.15$ and draw the digital call option price as function of the strike. Redo the calculation with smoothing on. Why do we see these results?

4/ Increase the width of the grid (num std) and the number of spatial steps (num s) simultaneously so that $\Delta \ln S$ is kept constant. Let 'num std' = 1, 2, … , 10. Where is the sweet spot and why?

5/ Write the function kBlack::fdFwdRunner() that returns a full finite difference grid of European initial call prices. The result is a matrix with first dimension being all expiries and second dimension being all strikes.

Hint:

$$c(t_h, s_i) = E[e^{-\int_0^{t_h} r(u)du}(s(t_h) - s_i)^+] = \sum_{j=i}^{n-1}(s_j - s_i)\,p(t_h, s_j)$$

What setting of the finite difference solver guarantees $\delta_{ss}c \geq 0$?

For the case of $r = \mu = 0$ show numerically that the initial option prices satisfy

$$c(t_{h+1}) = [I + (1-\theta)\Delta t\bar{A}][I - \theta\Delta t\bar{A}]^{-1}c(t_h)$$

What is this useful for?

6/ Modify kBlack::fdRunner() to price a down-and-out call option by only modifying the payoff. Now consider the convergence as the number of time steps is varied as in 1.

If the error is $O(\Delta t^a)$, what is a? Can you modify the code so that we achieve a better convergence?