Prof. Dr. Karsten Urban
Mazen Ali

Institute for Numerical Mathematics
Ulm University
Summer Term 2018

Numerical Finance – Sheet 8

(due 18.06.2018)

Exercise 1: Black Scholes PDE and Finite Differences

Consider the Black-Scholes problem (non-dividend paying) with the corresponding backward PDE-problem

$$\begin{cases} V_t(S,t) + \frac{\sigma^2}{2} S^2 V_{SS}(S,t) + rSV_S(S,t) - rV(S,t) = 0, \quad \forall S > 0, 0 \le t \le T, \\ V(S,T) = g(S), \quad \forall S > 0. \end{cases}$$

a) Use the transformations

$$S = Ke^x, \quad t = T - \frac{\tau}{\frac{1}{2}\sigma^2}, \quad q = \frac{2r}{\sigma^2}, \quad v(\tau, x) := V\left(Ke^x, T - \frac{2\tau}{\sigma^2}\right)$$
$$y(x,\tau) = \frac{1}{K} \exp\left(\tfrac{1}{2}(q-1)x + \left(\tfrac{1}{4}(q-1)^2 + q\right)\tau\right) v(\tau, x),$$

to show that the above problem is equivalent to the heat equation with initial condition:

$$\begin{cases} y_\tau(x,\tau) - y_{xx}(x,\tau) = 0, & x \in \mathbb{R}, 0 \le \tau \le \frac{1}{2}\sigma^2 T, \\ y(x,0) = \exp(\frac{x}{2}(q-1)) \frac{1}{K} g(K\exp(x)), & x \in \mathbb{R}. \end{cases}$$

b) Derive the initial conditions of the transformed Black-Scholes equation when

- g is a European call, and
- g is a European put

c) Argue why the boundary conditions

$$y(x,\tau) = r_1(x,\tau),\ x \to -\infty \quad \text{and} \quad y(x,\tau) = r_2(x,\tau),\ x \to +\infty$$

with

- $r_1(x,\tau) = 0$, $r_2(x,\tau) = \exp(\frac{1}{2}(q+1)x + \frac{1}{4}(q+1)^2\tau)$ for a call option, and
- $r_1(x,\tau) = \exp(\frac{1}{2}(q-1)x + \frac{1}{4}(q-1)^2\tau)$, $r_2(x,\tau) = 0$ for a put option

are reasonable. For simplicity, you might assume $q \ge 1$ for the call option and $q \le 1$ for the put option.

Programming Exercise 1: FD for Black Scholes – Explicit Euler (14 Points)

Consider an European Put option with $\sigma = 0.4$, $r = 0.04$, $T = 1$, $K = 12$. Implement the explicit Euler method to price this option.

a) Plot the surface of all option prices for $S_0 \in [0, 20]$, using a discretization of $N = 100$ points in space and $M = 400$ points in time.

b) For a fixed spatial discretization of $N = 2^9$ and time discretizations $M \in [2^4, 2^{12}]$, compute the error of your approximation for $t = 0$, $S_0 = K$. What do you observe?

Hint: Make sure that the strike K corresponds to a discretization point.

Programming Exercise 2: FD for Black Scholes – Crank-Nicolson (24 + 8 Points)

Consider an European Put option with $\sigma = 0.4$, $r = 0.04$, $T = 1$, $K = 12$. Implement the Crank-Nicolson method to price this option. In order to do so, you'll find some material for the solution of the arising equation systems on the homepage. You will have to

- take a look at the files *densematrix.h* and *cg.h*, where you find an implementation of a (dense) matrix class and a cg function that is templated on arbitrary matrix and vector types,
- implement the missing operators in the header *operators.h* that provides the necessary functionality for vector-vector and matrix-vector operations.

a) Plot the surface of all option prices for $S_0 \in [0, 20]$, using a discretization of $N = 100$ points in space and $M = 400$ points in time.

b) For a fixed spatial discretization of $N = 2^9$ and time discretizations $M \in [2^4, 2^{12}]$, compute the error of your approximation for $t = 0$, $S_0 = K$. What do you observe?

Hint: Make sure that the strike K corresponds to a discretization point.

c)* Replace the dense matrix class by a sparse matrix class, that stores the values in a coordinate storage scheme, i.e. it saves only the row and column indizes as well as the value for each non-zero entry of the matrix. Of course, you also have to overload the matrix-vector multiplication for this matrix class. Does the use of this class improve your computation times?

Comment: Spend some time on the operators as well as the matrix classes, since we will reuse them on future exercise sheets.