Prof. Dr. Karsten Urban
Mazen Ali

Institute for Numerical Mathematics
Ulm University
Summer Term 2018

Numerical Finance – Sheet 5

(due 28.05.2018)

Exercise 1: Binomial Method

a) Consider the value $V(t,S)$ of an option in the Black-Scholes model. $\Delta(t,S) := \frac{\partial V(t,S)}{\partial S}$ is the so called *delta* hedge parameter. Show that one can approximate the *delta* by the hedge parameter

$$\Delta_{Binom}(t,S) := \frac{V^{(u)} - V^{(d)}}{uS - dS} \text{ with } V^{(u)} := V(t+\Delta t, uS),\ V^{(d)} := V(t+\Delta t, dS)$$

of the binomial model where for fixed S the error $|\Delta(S,t) - \Delta_{Binom}(S,t)|$ is of order $\mathcal{O}(\Delta t)$.

b) Show that $V_{0,M} \to V(0,S_0)$ by proving

(i) If we denote by $\mathbb{E}[V_T]$ the expectation of the payoff in the binomial model at maturity, show that

$$\mathbb{E}[V_T] = \sum_{j=\alpha}^{M} \binom{M}{j} p^j (1-p)^{M-j} (S_0 u^j d^{M-j} - K),$$

where

$$\alpha = \left\lceil -\frac{\log(\frac{S_0}{K}) + M\log(d)}{\log(u) - \log(d)} \right\rceil.$$

(ii) Show that for $\widetilde{p} := pue^{-r\Delta t}$

$$V_{0,M} = e^{-rT}\mathbb{E}[V_T] = S_0 B_{M,\widetilde{p}}(\alpha) - e^{-rT} K B_{M,p}(\alpha)$$

where $B_{M,p}(j) := \mathbb{P}[X \geq j]$ for a (M,p)-binomially distributed random variable X.

(iii) Use the central limit theorem and (ii) to show that

$$\lim_{M\to\infty} V_{0,M} = S_0 F(d_1) + e^{-rT} K F(d_2)$$

where

$$d_1 = \frac{\log(\frac{S_0}{K}) + (r + \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}, \quad d_2 = \frac{\log(\frac{S_0}{K}) + (r - \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}.$$

c) Derive complexity and storage estimates for the binomial method with respect to the number of time steps M.

Hint:
Using power series expansions for e^x and $\sqrt{1-x}$, one can show that

$$u = 1 + \sigma\sqrt{\Delta t} + \frac{1}{2}\sigma^2\Delta t + \mathcal{O}((\Delta t)^{3/2}) = e^{\sigma\sqrt{\Delta t}} + \mathcal{O}((\Delta t)^{3/2}),$$
$$d = 1 - \sigma\sqrt{\Delta t} + \frac{1}{2}\sigma^2\Delta t + \mathcal{O}((\Delta t)^{3/2}) = e^{-\sigma\sqrt{\Delta t}} + \mathcal{O}((\Delta t)^{3/2}).$$

Programming Exercise 1: Implied Volatilities (8 Points)

An option price $V = V(0, S_0)$ depends on the parameters S_0, K, T, r and σ. Of these, only σ is not observable at the market. However, one can use the observed option price V_{traded} to infer the so-called *implied volatility* σ by solving $V(\sigma) = V_{traded}$ for σ. Usually, one then finds that σ is *not* constant, as assumed in the Black-Scholes model. For this reason, one often looks at the *implied volatility surface*, i.e. the implied volatility as a function of strike and maturity: $\sigma = \sigma(K, T)$.

On November 12th, 2012, Google Inc. (GOOG) stocks were traded at 665.50\$. At the same time, European Call options on Google were priced as in Table 1. The 1-year US Treasury Bills rate was $r = 0.17\%$.

	Maturity T			
Strike K	0.0972	0.3333	0.5833	1.1667
450	248.50	233.30	221.00	227.50
550	118.40	125.99	124.50	143.00
615	52.60	66.15	95.20	136.80
650	28.63	47.10	70.18	89.03
700	6.52	25.00	37.50	64.00
800	0.49	4.83	12.66	33.30
850	0.25	1.70	10.10	19.60

Table 1: Call option prices on Google

Use this data to construct an empirical volatility surface for the Google Call option by computing the implied volatility at the different data points. In order to do so,

- use Newton's method to find the implied volatilities, where you
- compute the option prices $V(\sigma)$ using a binomial tree,
- and the derivative $\frac{\partial}{\partial\sigma}V(\sigma)$ w.r.t σ using central differences, i.e. $\frac{\partial}{\partial\sigma}V(\sigma) \approx \frac{V(\sigma+h)-V(\sigma-h)}{2h}$.

Plot the empirical volatilities $\sigma(K, T)$ in a 3D-Plot. What do you see?

Hints:

- Of course, the Newton method is very sensitive w.r.t. initial values. Using $\sigma^{(0)} = 0.7$ should work for all given data points. (As an alternative to Newton, *Brent's method* is often used in practice.)
- On the homepage you find the data points in a text file, where the first line contains the strike values, the second the maturities and the rest of the file the data for V_{traded} as in Table 1.
- Gnuplot can plot 3D data given in the following format (note the blank line between different x-values):

```
x1 y1 z(x1,y1)
...
x1 yn z(x1,yn)

x2 y1 z(x2,y1)
...
x2 yn z(x2,yn)
```

  Use `splot ''data.txt'' w lp`. (*w lp* stands for *with linespoints*).

Programming Exercise 2: Simulating Wiener Processes (8 Points)

There are several methods to generate paths of a Wiener process. We will consider here the two most common ones.

a) **Random Walk**:
Making use of the fact that the increments of a Wiener process W are independently distributed with $W_t - W_s \sim N(0, t-s)$ for all $0 \le s < t \le T$, we can simulate paths in the following way. Let $0 = t_0 < t_1 < \cdots < t_N = T$, $Z_1, \ldots, Z_N \sim N(0,1)$. Then define

$$\begin{aligned} W_0 &= 0, \\ W_{t_{i+1}} &= W_{t_i} + \sqrt{t_{i+1} - t_i}\, Z_{i-1} \qquad \text{for } i = 1, \ldots, N. \end{aligned}$$

(Note that this corresponds to the Euler-Maruyama method for the PDE $dX_t = dW_t$, $X_0 = 0$.)

b) **Brownian Bridge**:
One can also start by generating the endpoint W_T of the process, and then fill in the other values using the distribution of W_t conditional on the already generated points. This procedure is known as Brownian Bridge. More precisely, consider $N = 2^n$ and $t_i - t_{i-1} = \frac{T}{N}$ for all $i = 1, \ldots, N$. One then generates first W_T, then $W_{\frac{T}{2}}$ based on W_0 and W_T, then $W_{\frac{T}{4}}$, $W_{\frac{3T}{4}}$ based on W_0, $W_{\frac{T}{2}}$ and W_T, etc.

Suppose one has already calculated the $(j-1)$th refinement $W_0, W_{\frac{T}{2^{j-1}}}, W_{\frac{2T}{2^{j-1}}}, \ldots, W_{\frac{2^{j-1}T}{2^{j-1}}}$. Due to the independence of the increments, the value of W_s depends only on the two nearest values $W_{s_k} := W_{\frac{kT}{2^{j-1}}} < s < W_{\frac{(k+1)T}{2^{j-1}}} =: W_{s_{k+1}}$. In our construction, we always consider $s = \frac{1}{2}\left(\frac{kT}{2^{j-1}} + \frac{(k+1)T}{2^{j-1}}\right)$, i.e., the midpoint between two already calculated time points. In that case, the conditional distribution of W_s is given by

$$(W_s \mid W_{s_k} = w_{s_k}, W_{s_{k+1}} = w_{s_{k+1}}) \sim \mathcal{N}\left(\frac{w_{s_k} + w_{s_{k+1}}}{2}, \frac{T}{2^{j+1}}\right).$$

This yields the following construction, using $Z_1, \ldots, Z_N \sim \mathcal{N}(0,1)$:

$$W^0(0) = 0, W^0(T) = \sqrt{T} Z_1,$$

$$W^1(t) = \begin{cases} \frac{1}{2}\big(W^0(0) + W^0(T)\big) + \sqrt{\frac{T}{2^2}}\, Z_2, & t = \frac{1}{2}, \\ W^0(t), & t = 0, T, \end{cases}$$

$$W^j(t) = \begin{cases} \frac{1}{2}\left(W^{j-1}\left(\frac{2kT}{2^j}\right) + W^{j-1}\left(\frac{(2k+2)T}{2^j}\right)\right) + \sqrt{\frac{T}{2^{j+1}}}\, Z_., & t = \frac{(2k+1)T}{2^j}, \quad k = 0, \ldots, 2^{j-1} - 1, \\ W^{j-1}\left(\frac{kT}{2^{n-1}}\right), & t = \frac{2kT}{2^j}, \quad k = 0, \ldots, 2^{j-1}, \end{cases}$$

for $j = 2, \ldots, n$. (Note that the second line in $W^j(t)$ corresponds just to the values of W that have already been calculated).

Generate and plot paths of $\{W(t), 0 \le t \le 1\}$ using both methods for different N (for example $N = 4, 8, 32, 512, 1024$).

Hint: You can use the standard library `<random>` to generate random numbers according to the normal distribution (better than the random number generator we implemented in class).