QF4102 Financial Modelling and Computation Assignment 3

G01 Wang Zexin, Chen Penghao

November 19, 2017

1 Transformed Black-Scholes PDE model

Consider the **transformed** Black-Scholes PDE model:

$$\begin{cases} \frac{\partial u}{\partial t} + \frac{\sigma^2}{2}\frac{\partial^2 u}{\partial x^2} + (r - q - \frac{\sigma^2}{2})\frac{\partial u}{\partial x} - ru = 0, \quad x \in (-\infty, \infty), t \in [0, T) \\ u(x, T) = \varphi(x), \end{cases}$$

1.1 Derivation of fully implicit scheme

Evaluate partial derivatives at (x_n^i, t_n) where $t_n = n\Delta t, x_n^i = i\Delta x, n \in [0, \frac{T}{\Delta t}), i \in [-x_{max}, x_{max}], I_{max} = \frac{x_{max}}{\Delta x}$

$$\text{Use the forward time finite difference formulae : } \left.\frac{\partial u}{\partial t}\right|_{(x_n^i, t_n)} = \frac{u_{n+1}^i - u_n^i}{\Delta t} + O(\Delta t)$$

$$\text{Use the centred space finite difference formulae : } \left.\frac{\partial u}{\partial x}\right|_{(x_n^i, t_n)} = \frac{u_n^{i+1} - u_n^{i-1}}{2\Delta x} + O[(\Delta x)^2]$$

$$\text{Use the centred space finite difference formulae : } \left.\frac{\partial^2 u}{\partial x^2}\right|_{(x_n^i, t_n)} = \frac{u_n^{i+1} - 2u_n^i + u_n^{i-1}}{(\Delta x)^2} + O[(\Delta x)^2]$$

The finite difference equation is hence :

$$\frac{u_{n+1}^i - u_n^i}{\Delta t} + O(\Delta t) + \frac{\sigma^2}{2}\frac{u_n^{i+1} - 2u_n^i + u_n^{i-1}}{(\Delta x)^2} + (r - q - \frac{\sigma^2}{2})\frac{u_n^{i+1} - u_n^{i-1}}{2\Delta x} + O[(\Delta x)^2] - ru_n^i = 0$$

$$\frac{u_{n+1}^i - u_n^i}{\Delta t} + O(\Delta t) + O[(\Delta x)^2] = -\frac{\sigma^2}{2}\frac{u_n^{i+1} - 2u_n^i + u_n^{i-1}}{(\Delta x)^2} - (r - q - \frac{\sigma^2}{2})\frac{u_n^{i+1} - u_n^{i-1}}{2\Delta x} + ru_n^i$$

$$\frac{U_{n+1}^i - U_n^i}{\Delta t} = rU_n^i - \frac{\sigma^2}{2}\frac{U_n^{i+1} - 2U_n^i + U_n^{i-1}}{(\Delta x)^2} - (r - q - \frac{\sigma^2}{2})\frac{U_n^{i+1} - U_n^{i-1}}{2\Delta x}$$

$$U_{n+1}^i = U_n^i + \Delta t[rU_n^i - \frac{\sigma^2}{2}\frac{U_n^{i+1} - 2U_n^i + U_n^{i-1}}{(\Delta x)^2} - (r - q - \frac{\sigma^2}{2})\frac{U_n^{i+1} - U_n^{i-1}}{2\Delta x}]$$

$$U_{n+1}^i = U_n^i(1 + r\Delta t) - \frac{\Delta t}{2(\Delta x)^2}[\sigma^2(U_n^{i+1} - 2U_n^i + U_n^{i-1}) + \Delta x(r - q - \frac{\sigma^2}{2})(U_n^{i+1} - U_n^{i-1})]$$

$$U_{n+1}^i = U_n^{i-1}[\frac{\Delta t(r - q - \frac{\sigma^2}{2})}{2\Delta x} - \frac{\sigma^2\Delta t}{2(\Delta x)^2}] + U_n^i[1 + r\Delta t + \frac{\sigma^2\Delta t}{(\Delta x)^2}] + U_n^{i+1}[-\frac{\Delta t(r - q - \frac{\sigma^2}{2})}{2\Delta x} - \frac{\sigma^2\Delta t}{2(\Delta x)^2}]$$

$$U_{n+1}^i = aU_n^{i-1} + bU_n^i + cU_n^{i+1}, \forall I_{min} + 1 \le i \le I_{max} - 1$$

$$\text{where } a = \gamma - \frac{\alpha}{2}, b = \beta + \alpha, c = -\gamma - \frac{\alpha}{2}, \alpha = \frac{\sigma^2\Delta t}{(\Delta x)^2}, \beta = 1 + r\Delta t, \gamma = \frac{\Delta t(r - q - \frac{\sigma^2}{2})}{2\Delta x}$$

The boundary conditions are as follows:

$$U_n^{I_{max}} = e^{-q(T - n\Delta t)}\exp(I_{max}\Delta x) - e^{-r(T - n\Delta t)}X, \text{ when the underlying value is very large at} \exp(I_{max}\Delta x)$$

$$U_n^{I_{min}} = 0, \text{ when the underlying value is very small at} \exp(I_{min}\Delta x)$$

With the values of $U_n^{I_{min}}$ and $U_n^{I_{max}}$ specified, we can express the FDE into matrix form.

$$\begin{bmatrix} b & c & \dots & \dots & \dots & \dots & \dots \\ a & b & c & \dots & \dots & \dots & \dots \\ \dots & a & b & c & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & a & b & c & \dots \\ \dots & \dots & \dots & \dots & a & b & c \\ \dots & \dots & \dots & \dots & \dots & a & b \end{bmatrix} \begin{bmatrix} U_n^{I_{min}+1} \\ U_n^{I_{min}+2} \\ U_n^{I_{min}+3} \\ \dots \\ U_n^{I_{max}-3} \\ U_n^{I_{max}-2} \\ U_n^{I_{max}-1} \end{bmatrix} = \begin{bmatrix} U_{n+1}^{I_{min}+1} \\ U_{n+1}^{I_{min}+2} \\ U_{n+1}^{I_{min}+3} \\ \dots \\ U_{n+1}^{I_{max}-3} \\ U_{n+1}^{I_{max}-2} \\ U_{n+1}^{I_{max}-1} \end{bmatrix} + \begin{bmatrix} -aU_n^{I_{min}} \\ 0 \\ \vdots \\ \vdots \\ 0 \\ -cU_n^{I_{max}} \end{bmatrix}$$

More concisely, we can name the tridiagonal matrix A and the right hand side vector F to express the FDE in this form: $AU_n = U_{n+1} + F \rightarrow U_n = A^{-1}(U_{n+1} + F)$

1.2 Finite Difference Scheme Algorithm on fully implicit scheme

Data: S_0, X, r, T, σ, I, N, x_{max}

Result: c_{IDS}, Option Premium

$\Delta t = \dfrac{T}{N}$, $\Delta x = \dfrac{x_{max}}{I}$;

$\alpha = \frac{\Delta t(r-q-\frac{\sigma^2}{2})}{2\Delta x}$;

$\beta = 1 + r\Delta t$;

$\gamma = \frac{\sigma^2 \Delta t}{2(\Delta x)^2}$;

$a = \alpha - \gamma$;

$b = \beta + \alpha$;

$c = -\alpha - \gamma$;

for $i = -I+1, -I+2, \dots, I-2, I-1$ **do**

  $U_N^i = max(\exp(i\Delta x) - X, 0)$;

end

Generate a tridiagonal matrix A of dimension $(2I-1) * (2I-1)$,

with $A_{i,i} = b \,\forall i = 1, 2, \dots 2I-1$,$A_{i,i-1} = b \,\forall i = 2, \dots 2I-1$, $A_{i,i+1} = b \,\forall i = 1, 2, \dots 2I-2$.

for $j = N-1, N-2, \dots, 0$ **do**

  Generate a vector F of length $(2I-1)$, with $F_{2I-1} = c\exp(-r(T - j\Delta t))(S_{max} - X)$, $F_i = 0$ otherwise;

  $U_j = A^{-1}(U_{j+1} + F)$;

end

$i_0 = round\left(\dfrac{\ln S_0}{\Delta x}\right)$;

$c_{\text{IDS}} = U_0^{i_0}$;

For the European vanilla call option with strike price \$5, time to maturity of 1 year, current underlier price of \$5.25, volatility of 30%, risk free rate of 3%, dividend yield of 10%. Using a grid with values of x in the truncated domain $[-5, 5]$, with $N = 1500$ and I taking values from 100 to 1500 with increments of 100, the option value estimates are obtained as tabulated below:

I	N	Option price
100	1500	0.522776022894615
200	1500	0.523228505305835
300	1500	0.523105617704949
400	1500	0.522656345235669
500	1500	0.522797471560081
600	1500	0.522898874252548
700	1500	0.522950382492270
800	1500	0.522963742476891
900	1500	0.522959159142429
1000	1500	0.522890228302100
1100	1500	0.522914330062009
1200	1500	0.522936922834744
1300	1500	0.522947345908465
1400	1500	0.522949937081634
1500	1500	0.522947446794060

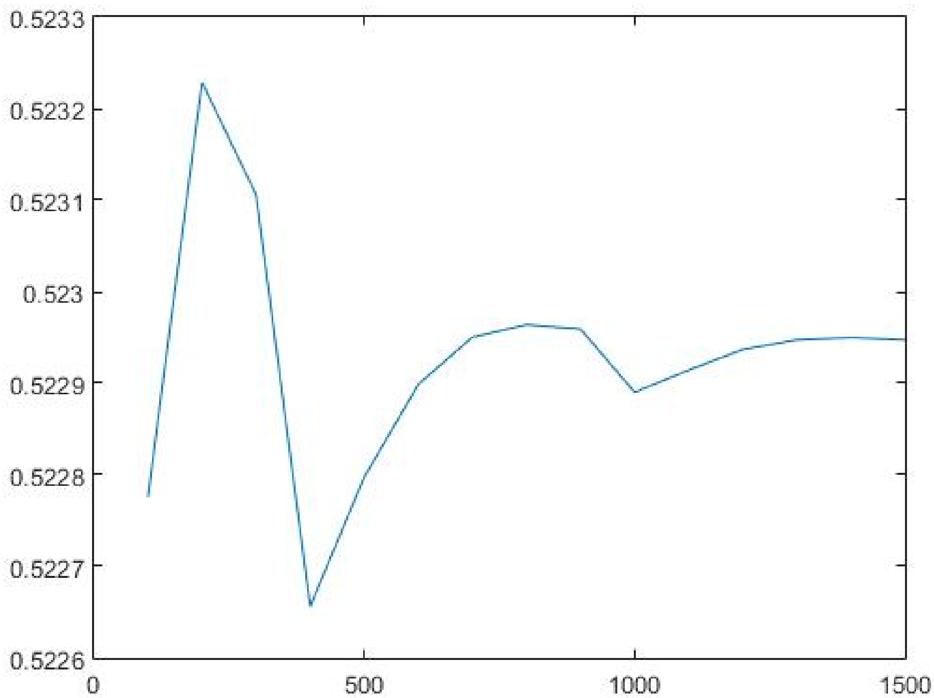

Figure 1: European vanilla call option value estimates against I with increments of 100

Since the value obtained at $I = 100$ is already quite close to the true value, it may be difficult to observe the convergence in the diagram of I going from 100 to 1500 with increments of 100 (figure 1). Hence, we plotted the diagram of I going from 100 to 1500 with increments of 20 in order to investigate further.

From figure 2, it is obvious that the option value converges to the true value at around \$0.52294 as the fluctations become smaller and smaller as I increases.

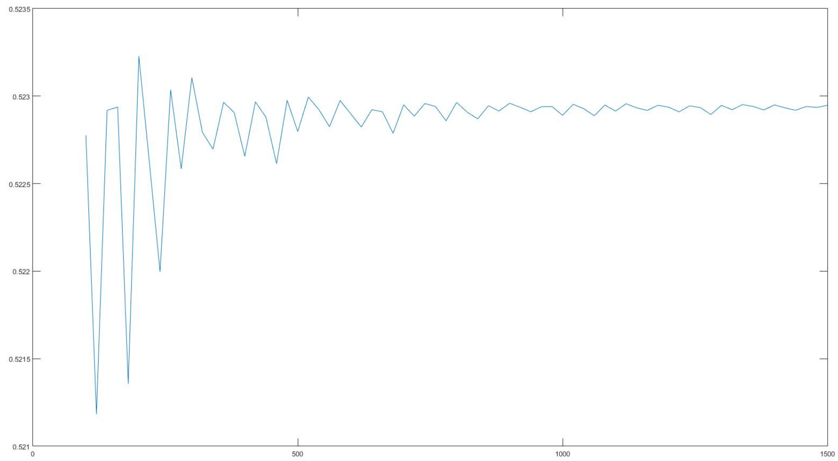

Figure 2: European vanilla call option value estimates against I with increments of 20

1.3 American vanilla call option using PSOR

For the American vanilla call option with same parameters as the above European vanilla call, we can use the projected SOR algorithm for the calculation of U_n from V_{n+1} with parameters $\epsilon = 1.0 * 10^{-6}, \omega = 1.3$.

I	N	Option price
100	1500	0.582849189993941
200	1500	0.583505358758593
300	1500	0.583457275858546
400	1500	0.582944640173874
500	1500	0.583166191075223
600	1500	0.583290840686203
700	1500	0.583345916640081
800	1500	0.583358092446549
900	1500	0.583351410333823
1000	1500	0.583260621906101
1100	1500	0.583286109957217
1200	1500	0.583301049237271
1300	1500	0.583302179614822
1400	1500	0.583294042928569
1500	1500	0.583280020736888

We have plotted the values estimated with I going from 100 to 1500 with increments of 100 in figure 3.

Similar to the previous section, we plotted another diagram of I going from 100 to 1500 with increments of 25 in order to investigate further as figure 4.

From figure 4, it is obvious that the option value converges to the true value at around \$0.5834 as the fluctations become smaller and smaller as I increases.

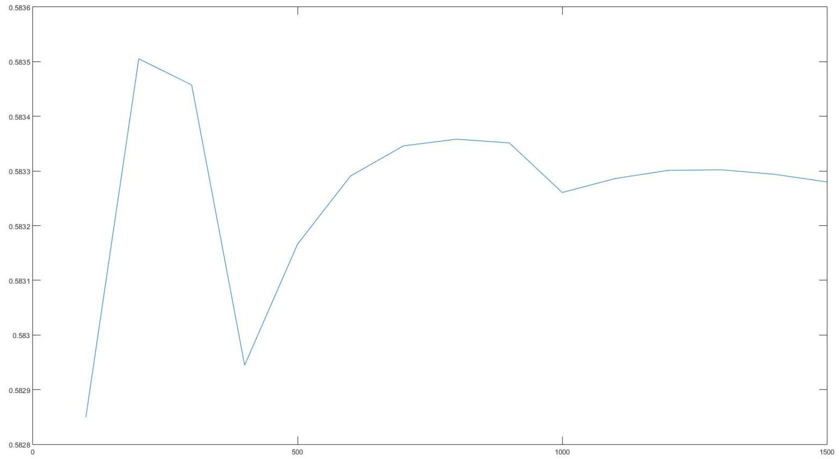

Figure 3: American vanilla call option value estimates against I with increments of 100

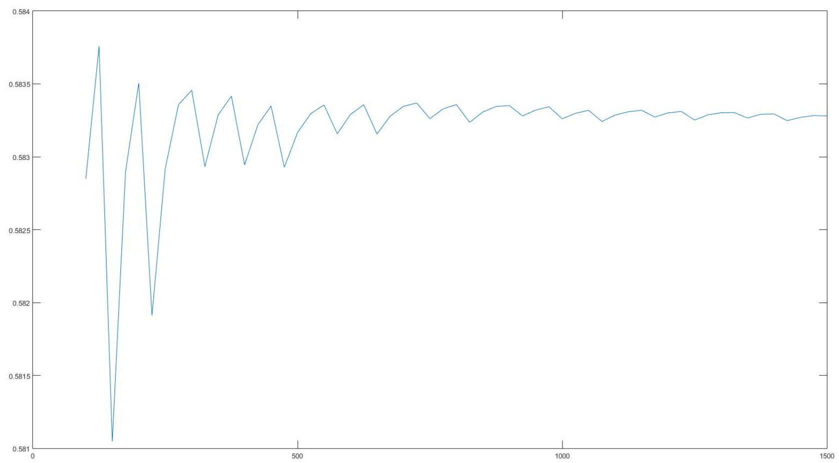

Figure 4: American vanilla call option value estimates against I with increments of 25

2 Valuation of digital call option

2.1 Algo