NPDE for Option Pricing : Assignment #3

Due on

Instructor:Dr.Kopriva 12:20am

Jian Wang

Contents

Problem 1

[Summary]

1) CPU time for the ADI algorithms doubled when time meshes doubled and quadrupled when stock meshes quadrupled. The value of the option was 0.2459.

2) When the stock 1 price or stock 2 price was equal to 0 , we treated the boundary conditions as the one dimensional European put option problem and used the Black Scholes formula to calculated the option prices.

3) The main idea of the ADI method was to proceed in two stages, treating only one operator implicitly at each stage. For this rainbow option problem, the implementation of ADI algorithm was not easy and some papers showed that it was not as efficient as Operator splitting method.

[Statement of the problem]

In this assignment, we need to calculate the price of rainbow option with the price of the maximum of two risky assets. We try to compute the option for the parameters K = 100, T = 0.25, r = 0.15, σ_1 = 0.15, σ_2 =0.20, and ρ= -0.20. Our task is to report the value of the option based on those assumptions.

We choose the Alternating Direction Implicit (ADI) method to solve the linear system.

In this report, we also report the CPU time that is needed to get the solutions on a 128 X 128 point mesh. These times will be used to compare the methods among classmates. We also explain the choices that we have made and how they affect our solutions and computational efficiency.

[Description of The Mathematics]

PDE formula The BSM and boundary condition for the Rainbow European put option is:

$$V_t + \frac{1}{2}\sigma_1^2 S_1^2 V_{S_1S_1} + \frac{1}{2}\sigma_1^2 S_2^2 V_{S_2S_2} + rS_1 V_{S_1} + rS_2 V_{S_2} + \rho\sigma_1\sigma_2 S_1 S_2 V_{S_1S_2} - rv = 0$$

Where the payoff function is:

$$V(S_1, S_2, T) = max(K - max(S_1, S_2), 0)$$

[**Analytical solution of the Rainbow put option:**]
From the paper "R.M.Stulz. Options on the minimum or the maximum of two risky assets, Journal of Financial Economics, 10:161 - 185, February, 1982), we can find the exact solution of the Rainbow European

Problem 1 continued on next page...

put option.

In his paper:
F: represented the strike price.
V: represented one stock price.
H: represented the other stock price.
R: is the interest rate.
$\tau = T - t$: was the time remaining to the maturity.
σ_H: was the volatility of stock H.
σ_V: was the volatility of stock V.

The price of a put option on the maximum of two risky assets was:

$$PX(V, H, F, \tau) = e^{-R\tau}F - MX(V, H, 0, \tau) + MX(V, H, F, \tau)$$

Where $PX(V.H, F, \tau)$ is a European put option on the maximum of asset V and H with strike price F and time to maturity τ.
$MX(V, H, F, \tau) = C(V, F, \tau) + C(H, F, \tau) - M(V, H, F, \tau)$
Where, $C(A, F, \tau)$ was an European call option on asset A with strike price F.

$M(V, H, F, \tau) = HN_2(\gamma_1\sigma_H\sqrt{\tau}, (ln(V/H) - 0.5 * \sigma^2\tau/\sigma\sqrt{\tau}, (\rho_{VH}\sigma_V - \sigma_H)/\sigma) + VN_2(\gamma_2\sigma_V\sqrt{\tau}, (ln(H/V) - 0.5 * \sigma^2\tau/\sigma\sqrt{\tau}, (\rho_{VH}\sigma_H - \sigma_V)/\sigma) - Fe^{-r\tau}N_2(\gamma_1, \gamma_2, \rho_{VH})$

Where:
$\gamma_1 = (ln(H/F) + (R - 0.5 * \sigma_H^2)\tau)/(\sigma_H * \sqrt{\tau})$
$\gamma_2 = (ln(V/F) + (R - 0.5 * \sigma_V^2)\tau)/(\sigma_V * \sqrt{\tau})$
$\sigma^2 = \sigma_V^2 + \sigma_H^2 - 2\rho_{VH}\sigma_V\sigma_H$

[Description of Algorithm]

The algorithm to solve the rainbow put option was as follows:
$\frac{u_{ij}^{n+1} - u_{ij}^n}{\Delta\tau} = L_{ADI}^x u_{ij}^{n+1/2} + L_{ADI}^y u_{ij}^{n+1}$

step 1:
The ADI method equation can be rewritten as:

$$\alpha_i u_{i-1,j}^{n+1/2} + \beta_i u_{ij}^{n+1/2} + \gamma_i u_{i+1,j}^{n+1/2} = f_{ij}$$

Where:

$$\alpha_i = -\frac{(\sigma_1 x_i)^2}{4h^2}$$

$\beta_i = \frac{1}{\Delta\tau} + \frac{(\sigma_1 x_i)^2}{2h^2} + \frac{rx_j}{2h}\frac{r}{2}$

$f_{ij} = \frac{u_{ij}^n}{\Delta\tau} + \frac{1}{4}(\sigma_2 y_j)^2 \frac{u_{i,j+1}^n - 2u_{ij}^n + u_{i,j-1}^n}{h^2} + \frac{1}{2}\rho\sigma_1\sigma_2 x_i y_j \frac{u_{i+1,j+1}^n + u_{i-1,j-1}^n - u_{i-1,j+1}^n - u_{i+1,j-1}^n}{4h^2}$

So we can get the A_x as a tridiagonal matrix constructed from the above equation.

$$A_x = \begin{Bmatrix} 2\alpha_0 + \beta_0 & \gamma_0 - \alpha_0 & 0 & \cdots & 0 & 0 \\ \alpha_1 & \beta_1 & \gamma_1 & \cdots & 0 & 0 \\ 0 & \alpha_2 & \beta_w & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \beta_{N_x-1} & \gamma_{N_x-1} \\ 0 & 0 & 0 & \cdots & \alpha_{N_x} - \gamma_{N_x} & \beta_{N_x} + 2\gamma_{N_x} \end{Bmatrix}$$

The calculation algorithm was as follows:
1 : *for* $j = 0 : N_y$
2 : *for* $i = 0 : N_x$
3 : *Set* $\alpha_i, \beta_i, \gamma_i$ *and* $f_i j$
4 : *end*
5 : *Solve* $A_x u_{0:N_x,j}^{n+1/2} = f_{0:N_x,j}$ *by PSOR method*
6 : *end*

step2:

$\alpha_j u_{i,j-1}^{n+1} + \beta_j u_{ij}^{n+1} + \gamma_j u_{i,j+1}^{n+1} = g_{ij}$

Where:
$\alpha_j = -\frac{(\sigma_2 y_j)^2}{4h^2}$

$\beta_j = \frac{1}{\Delta\tau} + \frac{(\sigma_2 y_j)^2}{2h^2} + \frac{ry_j}{2h} + \frac{r}{2}$

$g_{ij} = \frac{u_{ij}^{n+1/2}}{\Delta\tau} + \frac{1}{4}(\sigma_1 x_i)^2 \frac{u_{i+1,j}^{n+1/2} - 2u_{ij}^{n+1/2} + u_{i-1,j}^{n+1/2}}{h^2} + \frac{1}{2} r x_i \frac{u_{i+1,j}^{n+1/2} - u_{i,j}^{n+1/2}}{h} + \frac{1}{2}\rho\sigma_1\sigma_2 x_i y_j \frac{u_{i+1,j+1}^{n+1/2} + u_{i-1,j-1}^{n+1/2} - u_{i-1,j+1}^{n+1/2} - u_{i+1,j-1}^{n+1/2}}{4h^2}$

So we can get the A_y as a tridiagonal matrix constructed from the above equation.

$$A_y = \begin{Bmatrix} 2\alpha_0 + \beta_0 & \gamma_0 - \alpha_0 & 0 & \cdots & 0 & 0 \\ \alpha_1 & \beta_1 & \gamma_1 & \cdots & 0 & 0 \\ 0 & \alpha_2 & \beta_w & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \beta_{N_y-1} & \gamma_{N_y-1} \\ 0 & 0 & 0 & \cdots & \alpha_{N_y} - \gamma_{N_y} & \beta_{N_y} + 2\gamma_{N_y} \end{Bmatrix}$$

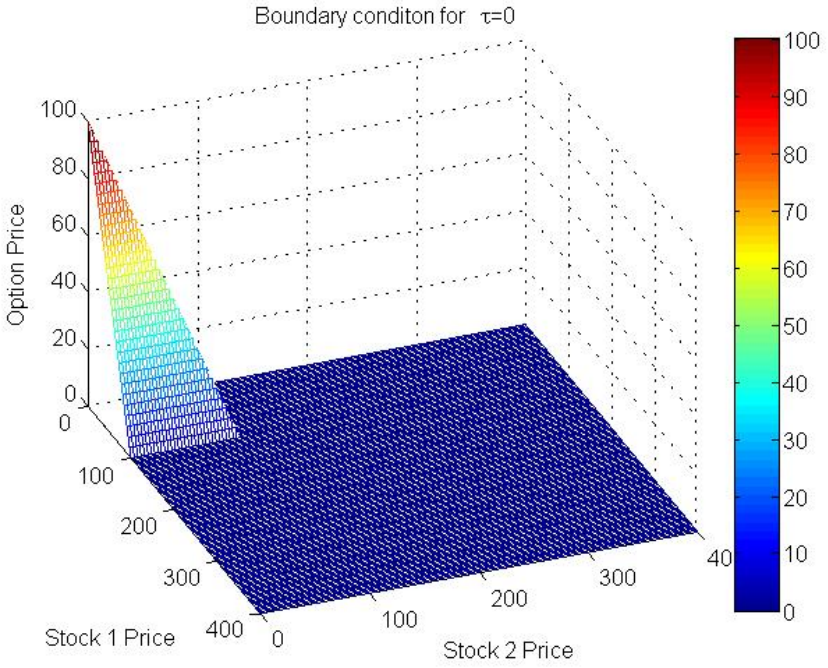
The calculation algorithm was as follows:
1 : *for* $i = 0 : N_x$

2 : *for* $j = 0 : N_y$
3 : *Set* $\alpha_j, \beta_j, \gamma_j$ *and* g_{ij}
4 : *end*
5 : *Solve* $A_y u^{n+1}_{i,0:N_y} = g_{i,0:N_y}$ *by PSOR method*
6 : *end*

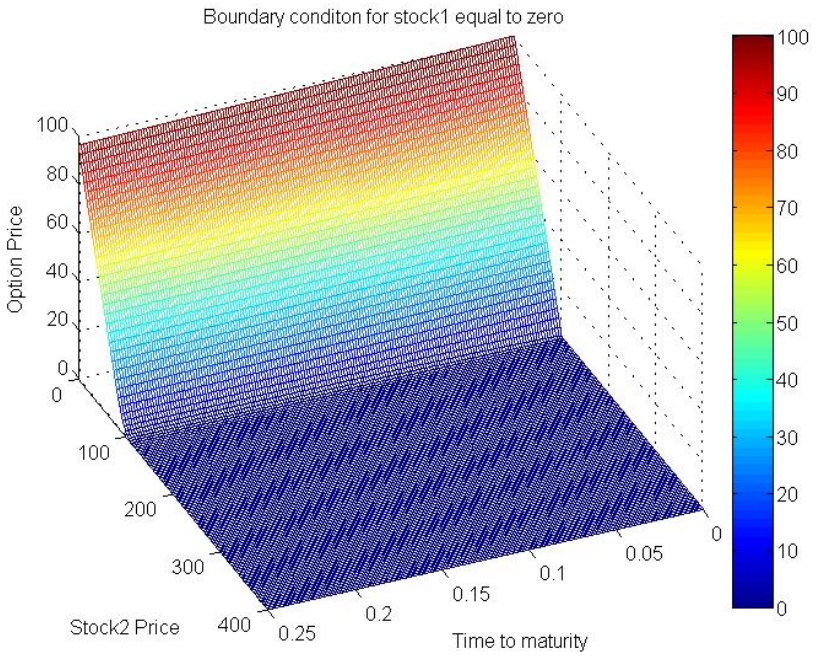
numerical results:

[Boundary conditions:]
We chose the following boundary conditions:

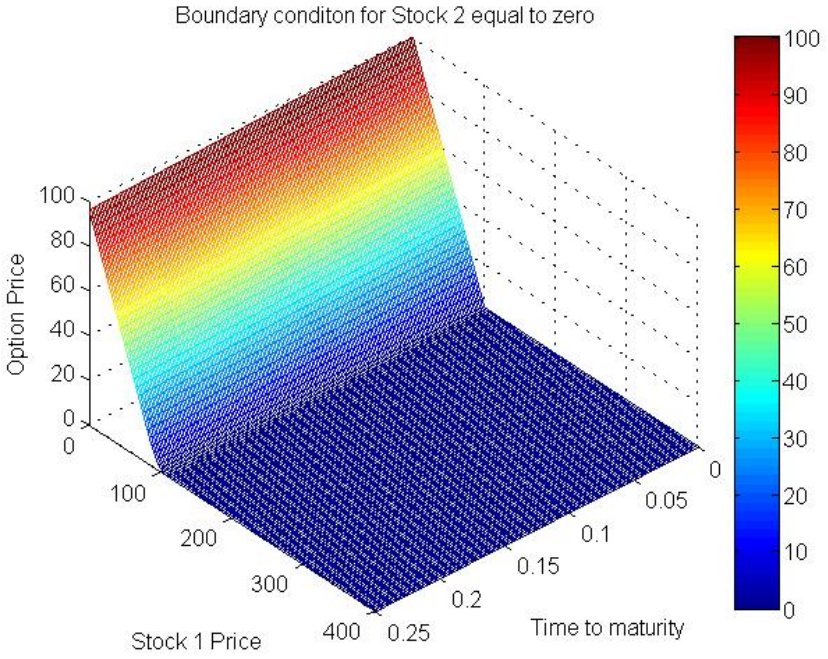
1)When Time to maturity was equal to 0 :
The boundary condition was the value payoff function based on S_1, S_2, and K.
$Payoff = max(K - max(S_1, S_2), 0)$
The following chart showed this boundary condition.



2) When stock price 1 was equal to zero.
The boundary condition was set to be the one dimensional European put option based on S_2, K,σ_2,r, and T. We used the Black Scholes formula to calculated the price of European put option.The following chart demonstrated this boundary condition.



3) When stock price 2 was equal to zero.
The boundary condition was set to be the one dimensional European put option based on S_1, K,σ_1,r, and T. The following chart demonstrated this boundary condition.



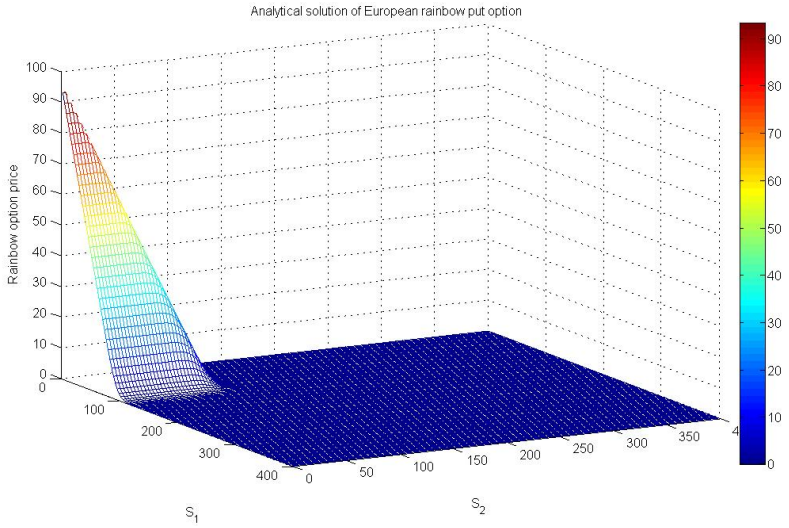
4) When stock price 1 or stock price 2 was equal to 4 times strike price.
The boundary condition here was set to be the 0;

Results and CPU times

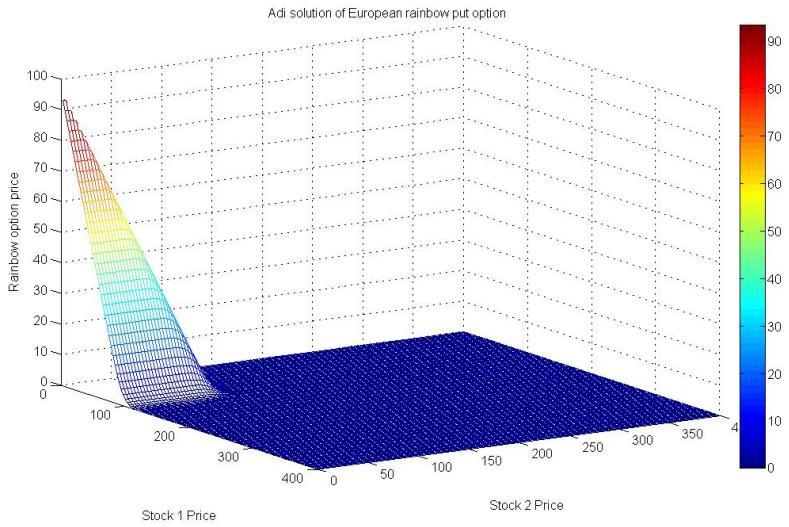
We set the stock meshes to be 128x128 and 256 x 256, and changed the time meshes from 40 to 640, doubled each time.
The following table listed the results of the CPU time.

CPU Time(S)		
Time meshes	Stock meshes(128x128)	Stock meshes(256x256)
40	3.26847	17.18241
80	5.603745	29.245583
160	10.716801	48.744667
320	20.84432	87.875921
640	45.651938	152.364333

The value of the option price was 0.2459 and the following two charts showed the analytical solutions and numerical solutions for the European rainbow put options based on different stock 1 and stock 2 prices.
Analytical solutions surfaces:



Numerical solutions(ADI) surfaces:


Adi solution of European rainbow put option
Rainbow option price
Stock 1 Price
Stock 2 Price
0
10
20
30
40
50
60
70
80
90
100
0
100
200
300
400
0
50
100
150
200
250
300
350