Finite Difference Schemes for Heston Model

Sensen Lin

Supervisor:

Dr. Christoph Reisinger

06/2008

A Graduate Project

submitted to the

Examiners, M.Sc. of Mathematical and Computational Finance

& Examination School

The University of Oxford,

United Kingdom

in partial fulfilment
of the requirements of the

M.Sc. of Mathematical and Computational Finance

by

Sensen Lin

lin.sensen@gmail.com

Acknowledgements

Great appreciation goes to Dr. Christoph Reisinger, my college advisor who chose this topic for me and instructed me while I was doing this work. I would also like to thank my class of MSc, who inspired me on various topics in financial mathematics.

ABSTRACT

Stochastic volatility is an interesting area in financial mathematics. Parabolic partial differential equations with mixed differentiation terms are the focus of numerical solution of Heston model. This document covers the numerical methods to Heston model.

Chapter 1 is an introduction to the problem and my main interest.

Chapter 2 is an overview of Heston model and its closed-form solution. The closed-form solution is a benchmark to test the numerical methods

Chapter 3 talks about the explicit scheme which is a straightforward method in solving Heston model. The result and restriction of this model are illustrated.

Chapter 4 discusses the ADI method dealing with special equations like Heston PDE. The details of this method are covered and comparison between schemes is given.

CONTENTS

1. INTRODUCTION

The Black-Scholes model concerns with the option pricing problems and has achieved great success, especially in stock option pricing. It also has practical and theoretical value toward hedging strategies by calculating the Greeks analytically and numerically. However, it has bias due to the variability of parameters which the original Black-Scholes model assumes to be constant. The stochastic properties of interest rate r and volatility σ often lead to more complicated approaches because analytic solutions are usually absent.

The stochastic volatility was first introduced to traditional model in the 1980s, when Hull and White [1] and others generalized the Black-Scholes model. Heston model was later presented in 1993 [2] which offered a practical method toward stochastic volatility. Moreover, a closed-form solution exists and the calculations for the Greeks are more straightforward.

Heston model is a nice benchmark in testing numerical schemes dealing with parabolic partial differential equations. The explicit scheme is a quick approach while it requires small time steps to retain the stability. This requires large number of time steps and it is not economic in computation. The alternating direction implicit (ADI) schemes are good alternative methods. Generally, ADI refers to any method that reduces multi-dimensional problem into several one-dimensional steps. Various ADI methods have made it possible to remove the restriction on time steps and to improve the accuracy and efficiency of work. The ADI schemes are good tools in dealing with parabolic partial differential equations like in stochastic volatility problem. [8]-[11] provided several ADI methods which can solve the PDEs with mixed-derivatives terms. What's the performance of ADI methods on the Greeks is of great interest and the numerical results will be given as a basis for analysis.

2. THE HESTON MODEL AND THE EXACT SOLUTION

The Heston model concerns with cases where volatility is stochastic. Assume that the spot index follows

$$dS = \mu S dt + \sqrt{v(t)} S dz_1(t) \tag{2.1}$$

where $z_1(t)$ is a Brownian motion as usual. In Heston model, the volatility follows

$$dv(t)=\kappa\left[\eta-v(t)\right]dt+\sigma\sqrt{v(t)}dz_2(t) \tag{2.2}$$

where $z_2(t)$ is another Brownian motion. Here v(t) is a mean-reversion with κ the reversion rate and η the reversion level. The correlation between the two Brownian motion is ρ. And the price U of contingent claims obeys the partial differential equation

$$\begin{aligned}&\frac{1}{2}vS^2\frac{\partial^2U}{\partial S^2}+\rho\sigma vS\frac{\partial^2U}{\partial S\partial v}+\frac{1}{2}\sigma^2v\frac{\partial^2U}{\partial v^2}+rS\frac{\partial U}{\partial S}\\&+\left[\kappa\left(\eta-v(t)\right)-\lambda(S,v,t)\right]\frac{\partial U}{\partial v}-rU+\frac{\partial U}{\partial t}=0\end{aligned} \tag{2.3}$$

The price of volatility risk $\lambda(S,v,t)$ is independent of particular asset. It can be obtained theoretically from any asset depending on volatility.

Take the European call option as an example. Assume the strike price to be K and expiring time T. The price is considered in the rectangular area of $[0,\infty]\times[0,\infty]$ and on time horizon $[0,T]$. The option price obeys equation (2.3) with boundary

$$\begin{aligned}&U(s,v,T)=\max(0,s-K),\\&U(0,v,t)=0,\\&\frac{\partial U}{\partial s}(\infty,v,t)=1,\\&rs\frac{\partial U}{\partial s}(s,0,t)+\kappa\eta\frac{\partial U}{\partial v}(s,0,t)-rU(s,0,t)+\frac{\partial U}{\partial t}(s,0,t)=0,\\&U(s,\infty,t)=s.\end{aligned} \tag{2.4}$$

Then the price is

$$U(s,v,t)=sP_1-Ke^{-r(T-t)}P_2 \tag{2.5}$$

in which

$$P_j\left(x,v,T;\ln K\right)=\frac{1}{2}+\frac{1}{\pi}\int_0^{\infty}\mathrm{Re}\left[\frac{e^{-i\phi\ln K}f_j\left(x,v,T;\phi\right)}{i\phi}\right]d\phi \tag{2.6}$$

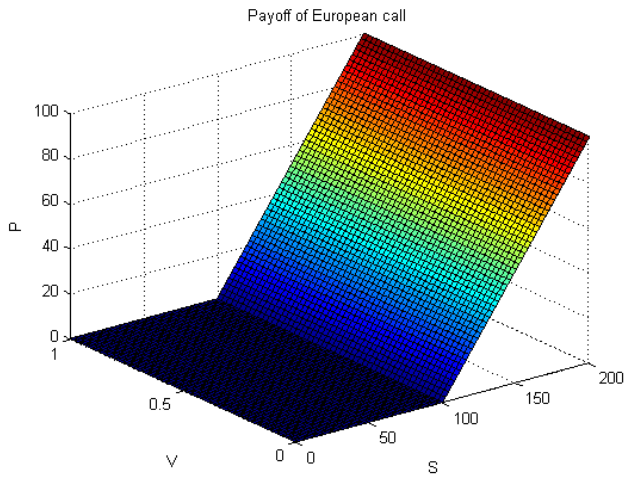
The characteristic functions $f_j\left(x,v,T;\phi\right)$ can be found in [2]. Numerical integration is available because the fast decay of the integrand.

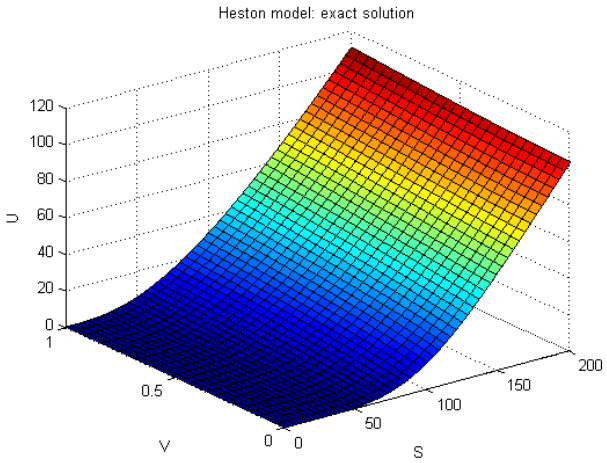
Another method is the Green's function, which is more useful way in computing the Greeks. The price is

$$U=\frac{1}{2\pi}e^{-r(T-t)}\int_{ic-\infty}^{ic+\infty}e^{-i\omega x}\tilde{W}\left(\omega,v,0\right)G\left(\omega,v,T-t\right)d\omega \tag{2.7}$$

where $x=\log S+r\left(T-t\right)$. $\tilde{W}\left(\omega,v,T-t\right)$ is the Fourier transformation of $W\left(\omega,v,T-t\right)=Ve^{r(T-t)}$ and $G\left(\omega,v,T-t\right)$ is the Green's function. More details are in the Appendix and are referred to [4]. Then the exact solution is used to calibrate the finite difference schemes.

Follow the specific parameter quantities chosen in [3]. The correlation factor $\rho=0.8$, interest rate $r=0.03$, reversion level $\eta=0.2$, reversion rate $\kappa=2$, and the volatility factor of volatility $\sigma=0.3$. The maturity time is $T=1$ and strike $K=100$. So the payoff and the price computed are as following:





3. EXPLICIT SCHEME

Here I start to implement the finite difference schemes. I confine the area of computation to a rectangular of $[0,S]\times[0,V]$ which is a bit larger than the area I really concern in order to eliminate the boundary effect. The discretization contains *I+1* nodes in *s* direction and *J+1* nodes in *v* direction. Use the central difference for the first-order differentiation. So all partial differentiations could be stated as following

$$
\begin{aligned}
(u_s)_{i,j} &\approx \frac{u_{i+1,j}-u_{i-1,j}}{2\Delta s},\\
(u_v)_{i,j} &\approx \frac{u_{i,j+1}-u_{i,j-1}}{2\Delta v},\\
(u_{ss})_{i,j} &\approx \frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{(\Delta s)^2},\\
(u_{vv})_{i,j} &\approx \frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{(\Delta v)^2},\\
(u_{sv})_{i,j} &\approx \frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{4\Delta s\Delta v}.
\end{aligned}
\tag{3.1}
$$

Hence we can reconstruct the current price problem into a linear transform series along time horizon by introducing the initial condition and the boundary condition. That is,

$$
U'(t)=GU(t)+R \tag{3.2}
$$

where G is the transformation matrix and R is the residue vector. Although the price $U(t)$ is actually a function of two variables s and v, an alternative way is treat it as a vector which includes values corresponding all grids in s and v direction. Given that there are I and J divisions along the two directions, the dimension of this vector will be $(I+1)(J+1)$.

We substitute the finite difference forms, together with the forward difference on time horizon, into the Heston PDE:

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}=\left[\left(s_i\right)^2 v_j \frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{2\left(\Delta s\right)^2}+\rho\sigma s_i v_j \frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{4\Delta s\Delta v}\right.$$
$$\left.+\sigma^2 v_j \frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{2\left(\Delta v\right)^2}+rs_i \frac{u_{i+1,j}-u_{i-1,j}}{2\Delta s}+\kappa\left(\eta-v_j\right)\frac{u_{i,j+1}-u_{i,j-1}}{2\Delta v}-ru_{i,j}\right]^n \tag{3.3}$$

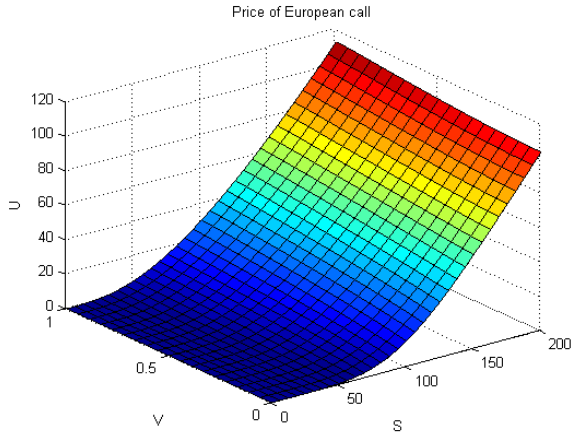
and so we get the explicit finite difference formula directly

$$u_{i,j}^{n+1}=A_{i,j}^{n}u_{i,j}^{n}+C_{i,j}^{n}u_{i-1,j}^{n}+D_{i,j}^{n}u_{i+1,j}^{n}+E_{i,j}^{n}u_{i,j-1}^{n}+F_{i,j}^{n}u_{i,j+1}^{n}$$
$$+B_{i,j}^{n}\left(u_{i-1,j-1}^{n}-u_{i-1,j+1}^{n}-u_{i+1,j-1}^{n}+u_{i+1,j+1}^{n}\right) \tag{3.4}$$

where

$$A_{i,j}^{n}=-i^2 v_j\Delta t-\frac{\sigma^2 j\Delta t}{\Delta v}+1-r\Delta t$$
$$C_{i,j}^{n}=\left(\frac{i^2 v_j}{2}-\frac{ri}{2}\right)\Delta t$$
$$D_{i,j}^{n}=\left(\frac{i^2 v_j}{2}+\frac{ri}{2}\right)\Delta t$$
$$E_{i,j}^{n}=\left(\frac{\sigma^2 j}{2\Delta v}-\frac{\kappa\left(\eta-v_j\right)}{2\Delta v}\right)\Delta t$$
$$F_{i,j}^{n}=\left(\frac{\sigma^2 j}{2\Delta v}+\frac{\kappa\left(\eta-v_j\right)}{2\Delta v}\right)\Delta t$$
$$B_{i,j}^{n}=\frac{\rho\sigma ij}{4}\Delta t \tag{3.5}$$

The implementation is straightforward, except that the boundary conditions need to be dealt specifically. Without loss of generality, suppose λ=0. With the same parameter of the above section, the graph of price landscape is



Then we can compare the explicit scheme with the exact solution to check the stability requirement of this scheme. Note that

$$A_{i,j}^n + C_{i,j}^n + D_{i,j}^n + E_{i,j}^n + F_{i,j}^n < 1 \tag{3.6}$$

So to make sure each of the five terms on the right side is above zero, it's necessary that for $0 \le i \le I$ and $0 \le j \le J$,

$$\Delta t \le \frac{1}{i^2 v_j + \frac{\sigma^2 v_j}{\Delta v} + r} = \frac{1}{i^2 v_j + \frac{J\sigma^2 v_j}{V} + r} \tag{3.7}$$

and

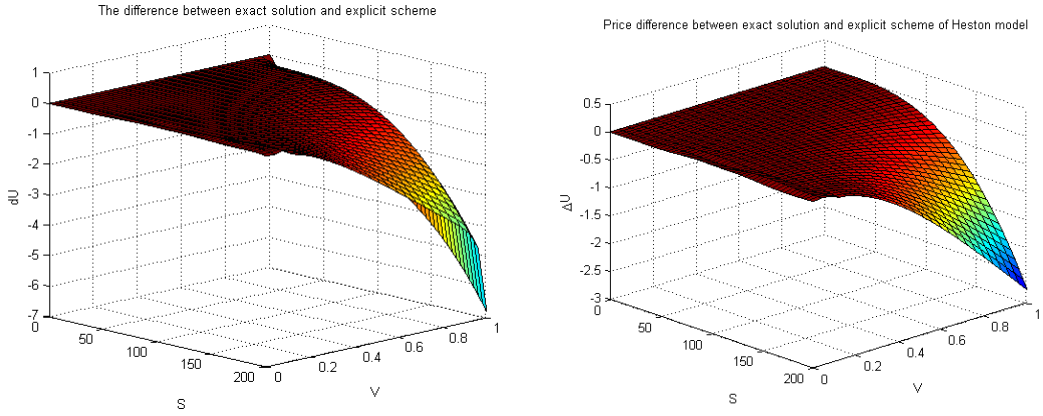
$$\begin{aligned} i &\ge \frac{r}{v_j} \\ j &\ge \frac{\kappa\left(\eta - v_j\right)}{\sigma^2} \end{aligned} \tag{3.8}$$

It's equivalent to say that

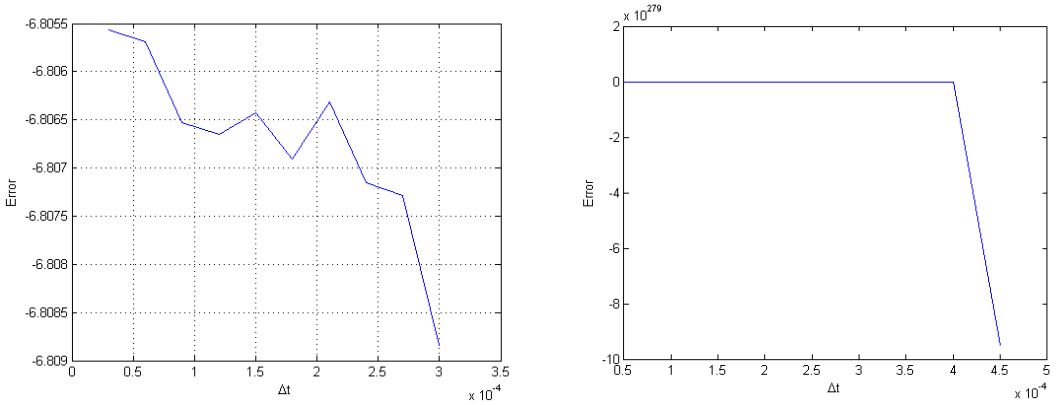
$$\Delta t \le \frac{1}{I^2 V + J\sigma^2 + r} \tag{3.9}$$

In the last implementation, we have $\Delta t = 1/3000$ and $\Delta t = 1/4000$, and $I = 40$, $J = 20$, $V = 1$, $\sigma = 0.3$, $r = 0.03$. So the right hand side is $1/1604.83 \ge \Delta t$, and the

requirement is met. The error landscape with different time steps is ($\Delta t = 1/3000$ on the left and $\Delta t = 1/4000$ on the right).



This plot is consistent with the above criterion set for Δt , i.e., the larger the price and volatility, the more likely stability would be demised. For the very node of (S,V), the error is likely to evolve as Δt increases like following.



Graph: the error of explicit scheme with the exact solution as time step length goes larger.

There is a sharp increase of error when Δt exceeds 4×10^{-4}, i.e. $1/2500$. There is also a slight increase of error before this point, showing that smaller time steps indeed increase the accuracy.

The computation of the Greeks plays an important role in trading strategies. So it's of great interest to examine the ability of the explicit scheme in computing the Greeks. The exact solutions provide the control sample needed in checking the explicit scheme. Central difference schemes are used in computing Greeks, as shown in equations (3.1). The outcome of the computation of the Greeks, including the value and difference between explicit scheme and exact solution is given in the Appendix 3.

The outcome shows that the overall result of explicit scheme is very closed to the exact solution. But the error becomes larger as Δt becomes larger. This is another evidence to show that the explicit scheme is conditionally stable. Furthermore, the relative error of Δ is smaller than Γ or Vega.

4. THE ADI SCHEME

4.1 BASIC IDEAS

Now we turned to implicit scheme. Here, large number of iterations are needed in each time steps, thus decrease the efficiency of computation. ADI scheme is a nice method to increase the efficiency in multi-dimensional problems while refrain the restriction on time steps. I start from the basic idea of the ADI scheme and implement it.

Similar to the explicit scheme, theoretically, the fully implicit scheme is based on the finite difference type of the Heston PDE:

$$\frac{u_{i,j}^{n}-u_{i,j}^{n-1}}{\Delta t}=\left[\left(s_i\right)^2 v_j\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{2\left(\Delta s\right)^2}+\rho\sigma s_i v_j\frac{u_{i+1,j+1}+u_{i-1,j-1}-u_{i-1,j+1}-u_{i+1,j-1}}{4\Delta s\Delta v}\right.$$
$$\left.+\sigma^2 v_j\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{2\left(\Delta v\right)^2}+rs_i\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta s}+\kappa\left(\eta-v_j\right)\frac{u_{i,j+1}-u_{i,j-1}}{2\Delta v}-ru_{i,j}\right]^{n} \quad (4.1)$$

Rewrite it as

$$
\begin{aligned}
&a_{i,j}^{n}u_{i,j}^{n}+c_{i,j}^{n}u_{i-1,j}^{n}+d_{i,j}^{n}u_{i+1,j}^{n}+e_{i,j}^{n}u_{i,j-1}^{n}+f_{i,j}^{n}u_{i,j+1}^{n}\\
&+b_{i,j}^{n}\left(u_{i-1,j-1}^{n}-u_{i-1,j+1}^{n}-u_{i+1,j-1}^{n}+u_{i+1,j+1}^{n}\right)=u_{i,j}^{n-1}
\end{aligned}
\tag{4.2}
$$

where

$$
\begin{aligned}
a_{i,j}^{n}&=1+i^{2}v_{j}\Delta t+\frac{\sigma^{2}j\Delta t}{\Delta v}+r\Delta t\\
c_{i,j}^{n}&=-\left(\frac{i^{2}v_{j}}{2}-\frac{ri}{2}\right)\Delta t\\
d_{i,j}^{n}&=-\left(\frac{i^{2}v_{j}}{2}+\frac{ri}{2}\right)\Delta t\\
e_{i,j}^{n}&=-\left(\frac{\sigma^{2}j}{2\Delta v}-\frac{\kappa\left(\eta-v_{j}\right)}{2\Delta v}\right)\Delta t\\
f_{i,j}^{n}&=-\left(\frac{\sigma^{2}j}{2\Delta v}+\frac{\kappa\left(\eta-v_{j}\right)}{2\Delta v}\right)\Delta t\\
b_{i,j}^{n}&=-\frac{\rho\sigma ij}{4}\Delta t
\end{aligned}
\tag{4.3}
$$

Thus, we can get the θ-scheme as the weighted average of the fully implicit scheme and explicit scheme.

$$
\begin{aligned}
&a_{i,j}^{n+1}u_{i,j}^{n+1}+c_{i,j}^{n+1}u_{i-1,j}^{n+1}+d_{i,j}^{n+1}u_{i+1,j}^{n+1}+e_{i,j}^{n+1}u_{i,j-1}^{n+1}+f_{i,j}^{n+1}u_{i,j+1}^{n+1}\\
&\quad+b_{i,j}^{n+1}\left(u_{i-1,j-1}^{n+1}-u_{i-1,j+1}^{n+1}-u_{i+1,j-1}^{n+1}+u_{i+1,j+1}^{n+1}\right)\\
&=A_{i,j}^{n}u_{i,j}^{n}+C_{i,j}^{n}u_{i-1,j}^{n}+D_{i,j}^{n}u_{i+1,j}^{n}+E_{i,j}^{n}u_{i,j-1}^{n}+F_{i,j}^{n}u_{i,j+1}^{n}\\
&\quad+B_{i,j}^{n}\left(u_{i-1,j-1}^{n}-u_{i-1,j+1}^{n}-u_{i+1,j-1}^{n}+u_{i+1,j+1}^{n}\right)
\end{aligned}
\tag{4.4}
$$

The coefficients are thus modified to reflect the implicit property θ. Because of the difficulty in dealing with the mixed derivatives term implicitly, we set $b_{i,j}^{n+1}=0$. Then we categorize the derivatives into three classes as in [5]. The θ-scheme is

$$
\left(1-\theta A_{1}-\theta A_{2}\right)u^{n+1}=\left[1+A_{0}+\left(1-\theta\right)A_{1}+\left(1-\theta\right)A_{2}\right]u^{n}+O\left(\Delta t^{3}\right)
\tag{4.5}
$$

A_0 corresponds to the mixed derivative term, A_1 the spatial derivatives in *s* direction and A_2 the spatial derivatives in *v* direction. The ru term is split into A_1 and A_2.

$$
\begin{aligned}
A_0 &= \frac{1}{4}\rho\sigma v S R_{sv}\delta_{sv} \\
A_1 &= \frac{1}{2}vS^2 R_{s2}\delta_{ss} + \frac{1}{2}rSR_s\delta_s - \frac{1}{2}r\Delta t \\
A_2 &= \frac{1}{2}\sigma^2 v R_{v2}\delta_{vv} + \frac{1}{2}\left[\kappa\left(\eta - v(t)\right) - \lambda\left(S,v,t\right)\right]R_v\delta_v - \frac{1}{2}r\Delta t
\end{aligned}
\tag{4.6}
$$

in which $R_s = \dfrac{\Delta t}{\Delta S}$, $R_v = \dfrac{\Delta t}{\Delta v}$, $R_{s2} = \dfrac{\Delta t}{\Delta S^2}$, $R_{v2} = \dfrac{\Delta t}{\Delta v^2}$, $R_{sv} = \dfrac{\Delta t}{\Delta S \Delta v}$

$$
\begin{aligned}
&\delta_s u_{i,j} = u_{i+1,j} - u_{i-1,j} \\
&\delta_{ss} u_{i,j} = u_{i+1,j} - 2u_{i,j} + u_{i-1,j} \qquad \text{etc.} \\
&\delta_{sv} u_{i,j} = u_{i+1,j+1} + u_{i-1,j-1} - u_{i-1,j+1} - u_{i+1,j-1}
\end{aligned}
\tag{4.7}
$$

Add $\theta^2 A_1 A_2 u^{n+1}$ to both sides of (4.5), we have

$$
\begin{aligned}
\left(1 - \theta A_1 - \theta A_2 + \theta^2 A_1 A_2\right)u^{n+1} = &\left[1 + A_0 + (1-\theta)A_1 + (1-\theta)A_2 + \theta^2 A_1 A_2\right]u^n \\
&+ \theta^2 A_1 A_2\left(u^{n+1} - u^n\right) + O\left(\Delta t^3\right)
\end{aligned}
\tag{4.8}
$$

As $A_1 A_2 \sim O\left(\Delta t^2\right)$ and $u^{n+1} - u^n \sim O\left(\Delta t\right)$, we can merge the term $\theta^2 A_1 A_2\left(u^{n+1} - u^n\right)$ into the error term. Equation (4.8) is equivalent to

$$
Au^{n+1} = \left(A + B\right)u^n + O\left(\Delta t^3\right) \tag{4.9}
$$

where

$$
A = \left(1 - \theta A_1\right)\left(1 - \theta A_2\right) \tag{4.10}
$$

and

$$
B = A_0 + A_1 + A_2 \tag{4.11}
$$

From this, we can get the alternative-direction implicit schemes. The simplest one among them is Douglas-Rachford method (DR method) as in demonstrated in [8,9]. Start from(4.9), ignoring the error term, we can write

$$
\left(1 - \theta A_1\right)\left(1 - \theta A_2\right)u^{n+1} = \left[1 + A_0 + (1-\theta)A_1\right]u^n - \left(1 - \theta A_1\right)\theta A_2 u^n \tag{4.12}
$$

The DR method has steps

$$\begin{aligned}(1-\theta A_1)Y &= \left[1+A_0+(1-\theta)A_1+A_2\right]u^n \\ (1-\theta A_2)u^{n+1} &= Y-\theta A_2 u^n\end{aligned} \tag{4.13}$$

When θ is 0, it becomes back to explicit scheme we've talked about in Chapter 3. More methods stem from this and have more steps to achieve more stable and accurate results. For example, the Craig and Sneyd [5] aimed to improve the stability and accuracy of the DR method, so they proposed a method which is unconditional stable for Heston PDE. The Craig-Sneyd method (CS method)is

$$\begin{aligned}(1-\theta A_1)Y_1 &= \left[1+A_0+(1-\theta)A_1+A_2\right]u^n \\ (1-\theta A_2)Y_2 &= Y_1-\theta A_2 u^n \\ (1-\theta A_1)Y_3 &= (1-\theta A_1)Y_1+\zeta A_0\left(Y_2-u^n\right) \\ (1-\theta A_2)u^{n+1} &= Y_3-\theta A_2 u^n\end{aligned} \tag{4.14}$$

where ζ is a positive weighting parameter.

Moreover, the Hundsdorfer and Verwer [10,11] proposed HV method:

$$\begin{aligned}(1-\theta A_1)Y_1 &= \left[1+A_0+(1-\theta)A_1+A_2\right]u^n \\ (1-\theta A_2)Y_2 &= Y_1-\theta A_2 u^n \\ (1-\theta A_1)Y_3 &= (1-\theta A_1)Y_1+\mu\left[A_0+(1-\theta)A_1\right]\left(Y_2-u^n\right) \\ (1-\theta A_2)u^{n+1} &= Y_3-\theta A_2 u^n\end{aligned} \tag{4.15}$$

The additional steps are added as the correction stages for stability. When the mixed derivatives term disappears, the CS method will be equivalent to the DR method but the HV method will not. Given any θ, the order of consistency of DR method is one. The order will be two if and only if $\theta=\zeta=\frac{1}{2}$ in CS method and $\mu=\frac{1}{2}$ in HV method [3,5,8,10].

4.2 BOUNDARY CONDITIONS

It's important to treat the boundary value carefully to maintain the accuracy of the results. The intermediate value of these methods are artificial and don't necessarily copy the real boundary conditions. Suppose the interior domain is

$$R_h = \{(i\Delta s, j\Delta v) : i = 1,...,I-1; j = 1,...,J-1\} \tag{4.16}$$

And boundary domain is $\partial R_h = {}_s\partial R_h + {}_v\partial R_h + {}_{sv}\partial R_h$, in which

$$\begin{aligned} {}_s\partial R_h &= \{(i\Delta s, j\Delta v) : i = 0, I; j = 1,..., J-1\} \\ {}_v\partial R_h &= \{(i\Delta s, j\Delta v) : i = 1,...,I-1; j = 0, J\} \\ {}_{sv}\partial R_h &= \{(i\Delta s, j\Delta v) : i = 0, I; j = 0, J\} \end{aligned} \tag{4.17}$$

From the first equation of (4.13), we can see that part of intermediate state Y_1 is required to obtain the result. So $Y_{1ij} : (i\Delta s, j\Delta v) \in {}_s\partial R_h$ are to be considered. For DR method, the intermediate boundary condition comes from the second equation of (4.13).

$$Y_b = (1 - \theta A_2) u_b^{n+1} + \theta A_2 u_b^n \tag{4.18}$$

If the boundary is time-independent, like the Dirichlet boundary condition $U(0, v, t) = 0$ in (2.4), the intermediate boundary is the same to the original boundary, according to (4.18). Otherwise, if the boundary is time-depedent, like Neumann condition $\frac{\partial U}{\partial s}(\infty, v, t) = 1$ in (2.4), Y_b should be calculated specifically at each time step. Use the second-order approximation on points at ${}_s\partial R_h$

$$\frac{\partial u}{\partial s}(I\Delta s, j\Delta v) \approx \frac{u_{I+1,j} - u_{I-1,j}}{2\Delta s} \tag{4.19}$$

Combine this and equation (4.1), and eliminate the value $u_{I+1,j}$, we have

$$\begin{aligned} u_{I,j}^{n+1} &= u_{I,j}^n - \Delta t \bullet r u_{I,j}^n + \Delta t \frac{I^2 j\Delta v}{2}\left(2u_{I-1,j}^n + 2\Delta s - 2u_{I,j}^n\right) \\ &+ \Delta t \frac{\sigma^2 j}{2\Delta v}\left(u_{I,j+1}^n - 2u_{I,j}^n + u_{I,j-1}^n\right) + rS\Delta t + \Delta t \frac{\kappa(\eta - j\Delta v)}{2\Delta v}\left(u_{I,j+1}^n - u_{I,j-1}^n\right) \end{aligned} \tag{4.20}$$

Thus we can get the intermediate boundary condition on ${}_s\partial R_h$ through (4.18). Note that this condition only works when computing the inside elements of *Y* and it has

nothing to do in computing u^{n+1}. So it is not necessary to leave the intermediate boundary condition as it is and I let all boundary elements of Y be zero.

For points at ${}_v\partial R_h$, I use the explicit scheme and use the second-order one-sided approximation for the derivatives:

$$u_v(0)=\frac{4u(\Delta v)-3u(0)-u(2\Delta v)}{2\Delta v}+O(\Delta v^2) \tag{4.21}$$

For CS method and HV method, the intermediate boundary condition is similar. I let Y_1 and Y_3 satisfy the same intermediate boundary condition as Y, while Y_2 the boundary condition as u^{n+1}.

4.3 IMPLEMENTATION

The DR method is the basis for the CS method and HV method and the functions used in the former one can fulfil the task of other two methods.

The idea of ADI scheme is to treat different dimensions separately, so we can avoid the mess of dealing with differentiation along different directions at the same time. In practise, the state vector at each time step is actually a matrix with x direction the volatility grids and y direction the index grids.

For example, on the right hand side of scheme (4.13), we deal with only the s direction, like

$$\begin{array}{c} \\ s \end{array}\begin{array}{c} \begin{array}{cccccc} & & v & & & \\ 0 & 1 & \cdots & & J & \end{array} \\ \begin{bmatrix} & & * & & \\ & \cdots & \vdots & \cdots & \\ & & * & & \end{bmatrix} \end{array} \begin{array}{l} \\ 0 \\ \vdots \\ I \end{array} \tag{4.22}$$

Each iteration deals with each column in the $(I+1)(J+1)$ vector. The transform matrix of A_1 is a $(I+1)\times(I+1)$ square matrix

$$\begin{pmatrix} -\frac{s_0^2 v_j}{(\Delta s)^2}-\frac{r}{2} & 0 & 0 & \cdots & 0 \\ \frac{s_1^2 v_j}{2(\Delta s)^2}-\frac{r s_1}{2\Delta s} & -\frac{s_1^2 v_j}{(\Delta s)^2}-\frac{r}{2} & \frac{s_1^2 v_j}{2(\Delta s)^2}+\frac{r s_1}{2\Delta s} & \cdots & 0 \\ 0 & \frac{s_2^2 v_j}{2(\Delta s)^2}-\frac{r s_2}{2\Delta s} & -\frac{s_2^2 v_j}{(\Delta s)^2}-\frac{r}{2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{s_I^2 v_j}{(\Delta s)^2}-\frac{r}{2} \end{pmatrix} \tag{4.23}$$

We reduce it to a $I \times I$ square matrix which is just the inside elements. We also change the inhomogeneous term to reflect the boundary of u^n.

In the meantime, same approach applies to the *v* direction. The transition matrix is

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & -\frac{\lambda^2 v_1}{(\Delta v)^2}-\frac{r}{2} & \frac{\lambda^2 v_1}{2(\Delta v)^2}+\frac{\kappa(\eta-v_1)}{2\Delta v} & \cdots & 0 \\ 0 & \frac{\lambda^2 v_2}{2(\Delta v)^2}-\frac{\kappa(\eta-v_2)}{2\Delta v} & -\frac{\lambda^2 v_2}{(\Delta v)^2}-\frac{r}{2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{\lambda^2 v_J}{(\Delta v)^2}-\frac{r}{2} \end{pmatrix} \tag{4.24}$$

And the state vector needs modified to reflect the boundary condition too.

Moreover, the transition matrix A_0 can be got via a linear transformation

$$\begin{aligned} \tilde{u}_{n-1}(i,j) = {} & u_{n-1}(i+1,j+1)+u_{n-1}(i-1,j-1) \\ & -u_{n-1}(i-1,j+1)-u_{n-1}(i+1,j-1) \end{aligned} \tag{4.25}$$

Then $A_0 u_{n-1} = \tilde{A}_0 \bullet \tilde{u}_{n-1}$ in which

$$\tilde{A}_0 = \frac{\rho\lambda}{4\Delta s \Delta v}\left[s_i v_j \right]_{(I+1)\times(J+1)}.$$

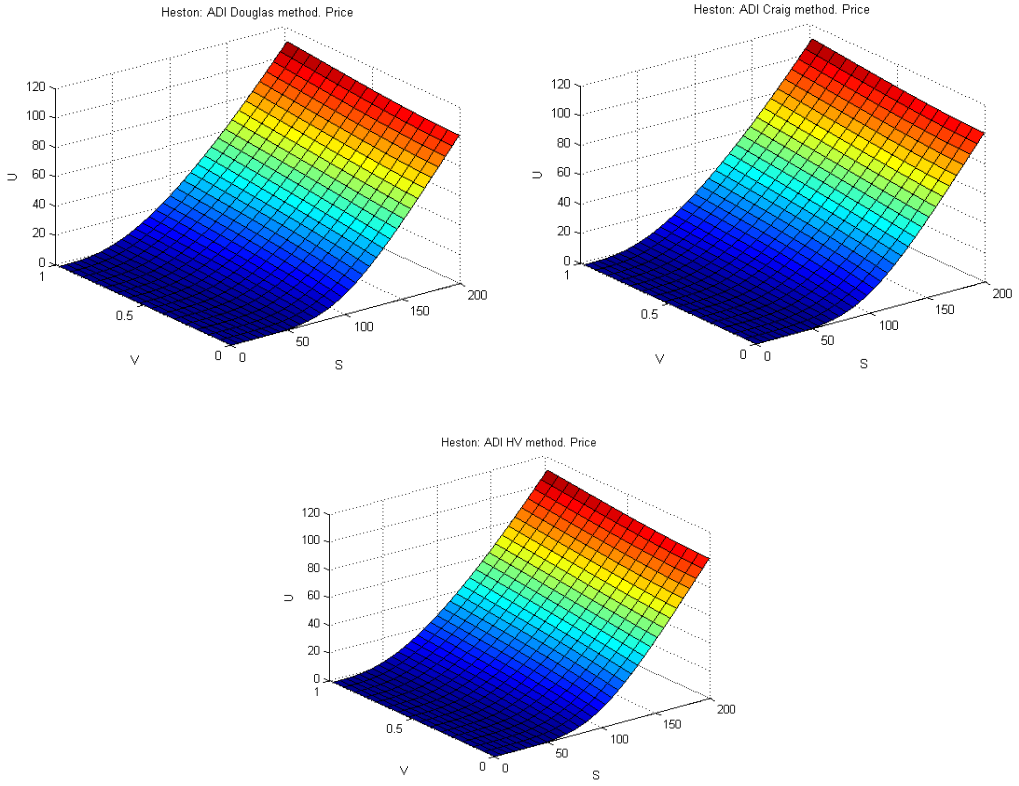
The dot means element-wise multiplication.

I use the Thomas algorithm to solve the left hand sides of the scheme because of the tridiagonal matrices.

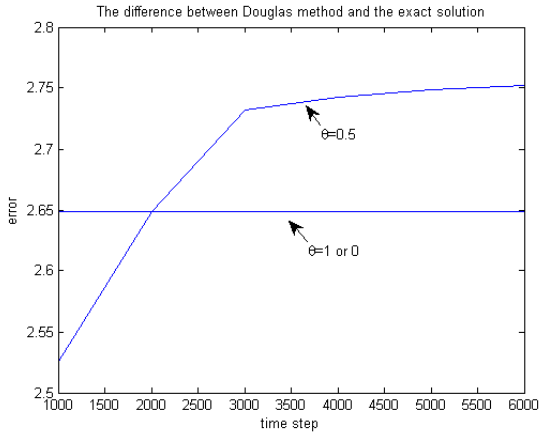
4.4 RESULT

We use the same parameters as in explicit scheme.

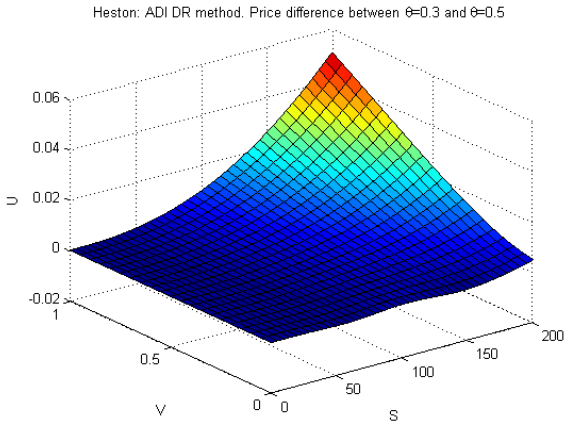
Use the Crank-Nicolson scheme, i.e. $\theta = 0.5$ and choose $\zeta = 0.5$ and $\mu = 0.5$ in order to achieve the order of consistency two. The result of the ADI schemes are as follows:



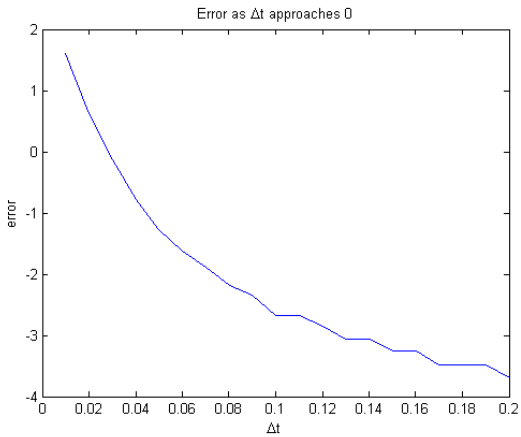
Now we are to check the affect of time step on the schemes. Below are the error at the point (S,V) between the Douglas scheme and the exact solution.



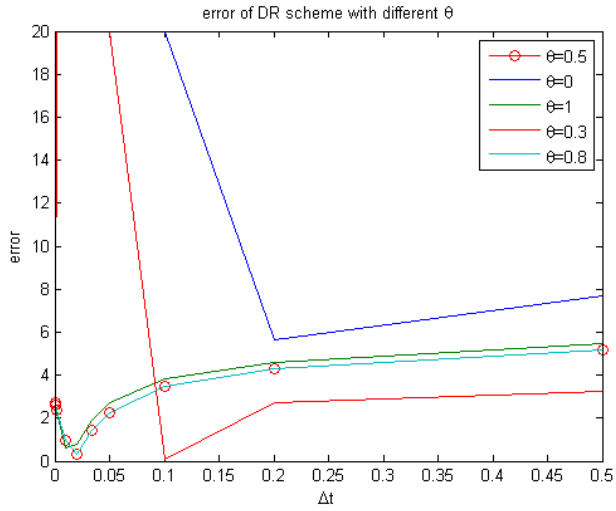
This plot shows that the ADI scheme can deal with computation of large Δt, which is not the case in the fully explicit scheme. Now let the number of time steps be fixed at 1000, the following graph shows the difference between the case θ=0.3 and θ=0.5.



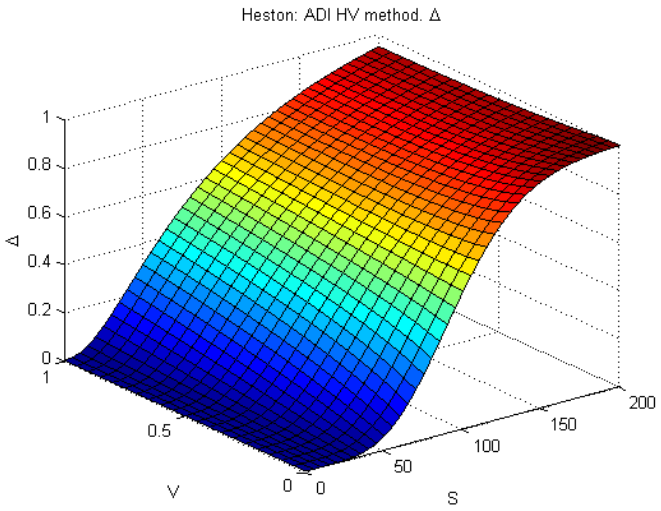
Now use $\mu = 0.3$ for HV Scheme (because the consistency of this scheme is irrelevant with μ), all others unchanged. The relations between error and Δt of the three ADI schemes are the same in the following area:



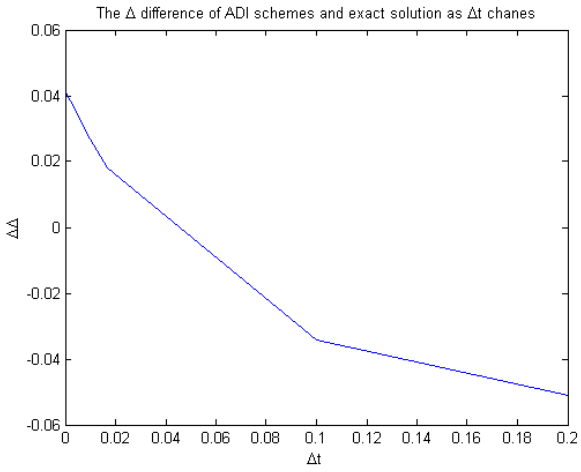
Then change θ, the pattern is different:



Next the work will be on the stability of computation of the Greeks via ADI schemes. The Delta computed by the three methods is the same.

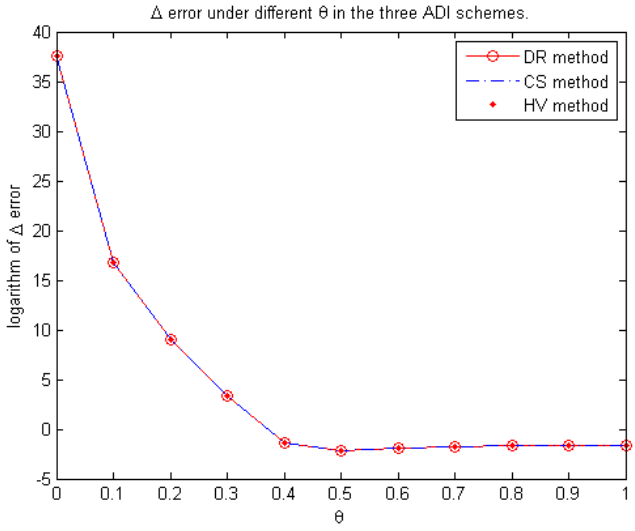


So first look at the stability of Δ as the time step length gets smaller. The three ADI schemes have the same results here when using the same parameter as before.



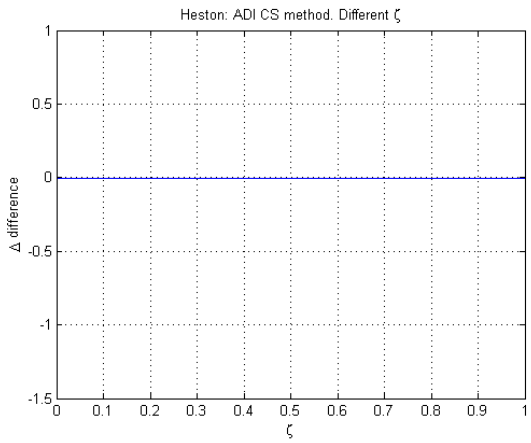
The results of the above two graph show that the difference is minimized around Δt=0.04, that is when the number of time steps is about 25. So from we use this parameter thereafter.

Then look at the stability of Δ when θ changes.

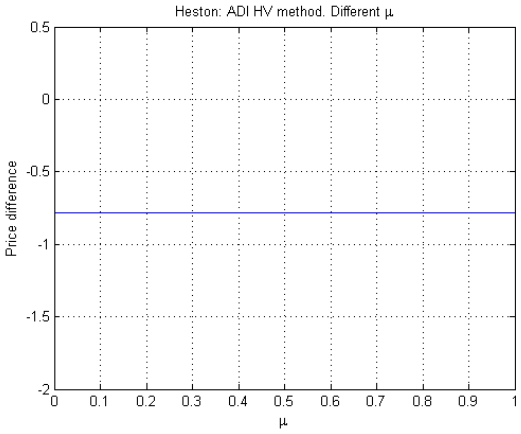


It shows that the error of Δ will diverge when θ approaches 0 from above. This is an indication of better accuracy of implicit scheme than explicit scheme on the computation of Δ. However, there is no evidence to show any difference in Δ computation between the three ADI schemes.

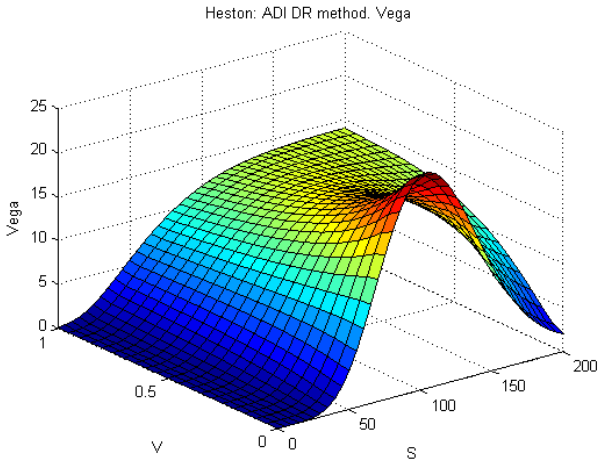
Also, the performance of CS method has no difference between different value of ζ.



Also the HV method is tested for different μ.



The Vega is also tested. The Vega computed via the DR method is



This is similar to the result I've got in Chapter 3.

5. CONCLUSION

From the result of my modelling, I've shown that the ADI schemes are reliable when we need the length of every time step Δt be long enough. The error between ADI scheme and exact solution is smaller than the explicit scheme.

The results have shown that the ADI scheme is more accurate when it is more 'implicit'. The parameter test shows that it's not very sensitive to the parameter of CS scheme. Work has been done on the stability of the ADI schemes and further work is needed to cover the efficiency of the numerical schemes.

REFERENCE

[1] J. C. Hull and A. White, The pricing of options on assets with stochastic volatility, *Journal of Finance*, 42 (1987) 281-300.

[2] S. Heston, A closed-form solution for options with stochastic volatility. *The Review of Financial Studies* (1993).

[3] K. in't Hout, ADI scheme in the numerical solution of the Heston PDE. *Proceedings to Numerical Analysis and Applied Mathematics* (2007),

[4] W. T. Shaw, Stochastic volatility, models of Heston type. Working paper.

[5] I. Craig and A. Sneyd, An alternating-direction implicit scheme for parabolic equations with mixed derivatives. *Comput. Math. Applic* (1988).

[6] M. Giles and R. Carter: Convergence analysis of Crank-Nicolson and Rannacher time-marching. *Journal of Computational Finance* (2006).

[7] J. C. Strikwerda, *Finite difference schemes and partial differential equations*. Wadsworth & Brooks/Cole, Pacific Grove, California (1989)

[8] J. Douglas and H. H. Rachford, On the numerical solution of heat conduction problems in two and three space variables, *Trans. Amer. Math. Soc.* 82 (1956) 421-439.

[9] D. W. Peaceman and H. H. Rachford, The numerical solution of parabolic and elliptic differential equations, *J. Soc. Ind. Appl. Math.* 3 (1955) 28-41.

[10] W. Hundsdorfer, Accuracy and stability of splitting with Stabilizing Corrections, *Applied Numerical Mathematics* 42 (2002) 213-233.

[11] J. G. Verwer, E. J. Spee, J. G. Blom and W. Hundsdorfer, A second order Rosenbrock method applied to photochemical dispersion problems, *SIAM J. Sci. Comput.* 20 (1999) 1456-1480.

[12] G. Fairweather and A. R. Mitchell, A new computational procedure for ADI methods. *SIAM Journal on Numerical Analysis,* 4(2) (June 1967) 163-170.

[13] N. Moodley, The Heston model: a practical approach with Matlab code. Graduation thesis. 2005.

APPENDIX

1. HESTON'S APPROACH FOR EXACT SOLUTION

Restate the Heston PDE:

$$\begin{aligned}&\frac{1}{2}vS^2\frac{\partial^2 U}{\partial S^2}+\rho\sigma vS\frac{\partial^2 U}{\partial S\partial v}+\frac{1}{2}\sigma^2 v\frac{\partial^2 U}{\partial v^2}+rS\frac{\partial U}{\partial S}\\&+\left[\kappa\left(\eta-v(t)\right)-\lambda(S,v,t)\right]\frac{\partial U}{\partial v}-rU+\frac{\partial U}{\partial t}=0\end{aligned}\tag{A1.1}$$

And the boundary conditions:

$$\begin{aligned}&U(s,v,T)=\max(0,s-K),\\&U(0,v,t)=0,\\&\frac{\partial U}{\partial s}(\infty,v,t)=1,\\&rs\frac{\partial U}{\partial s}(s,0,t)+\kappa\eta\frac{\partial U}{\partial v}(s,0,t)-rU(s,0,t)+\frac{\partial U}{\partial t}(s,0,t)=0,\\&U(s,\infty,t)=s.\end{aligned}\tag{A1.2}$$

Suppose that the solution to the Heston PDE is like the form of Black-Scholes model

$$U(s,v,t)=sP_1-KB(t,T)P_2\tag{A1.3}$$

where $B(t,T)$ represents the risk-free discounting asset, P_1 and P_2 are what we are going to find. Make the transform $x=\ln s$ and substitute the formula (A1.3) into the PDE (A1.1). We'll get

$$\begin{aligned}&\frac{1}{2}v\frac{\partial^2 P_j}{\partial x^2}+\rho\sigma v\frac{\partial^2 P_j}{\partial x\partial v}+\frac{1}{2}\sigma^2 v\frac{\partial^2 P_j}{\partial v^2}+\left(r+u_j v\right)\frac{\partial P_j}{\partial x}\\&+\left(a-b_j v\right)\frac{\partial P_j}{\partial v}+\frac{\partial P_j}{\partial t}=0,\qquad j=1,2\end{aligned}\tag{A1.4}$$

where $u_1=\frac{1}{2}$, $u_2=-\frac{1}{2}$, $a=\kappa\eta$, $b_1=\kappa+\lambda-\rho\sigma$, $b_2=\kappa+\lambda$.

Considering the payoff of the option, they are subject to the initial condition

$$P_j\left(x,v,T;\ln K\right)=1_{x\geq \ln K} \tag{A1.5}$$

P_j are the conditional probability that the option will expire in the money. Heston showed that the characteristic function of P_j, $f_j\left(x,v,t;\varphi\right)$, satisfies the PDE (A1.4). The terminal condition is

$$f_j\left(x,v,t;\varphi\right)=e^{i\varphi x} \tag{A1.6}$$

So the solution of characteristic function is

$$f_j\left(x,v,t;\varphi\right)=e^{C(T-t;\varphi)+D(T-t;\varphi)v+i\varphi x} \tag{A1.7}$$

where (Let $\tau=T-t$)

$$\begin{aligned} C\left(\tau;\varphi\right)&=r\varphi i\tau+\frac{a}{\sigma^2}\left\{\left(b_j-\rho\sigma\varphi i+d\right)\tau-2\ln\left[\frac{1-ge^{d\tau}}{1-g}\right]\right\}\\ D\left(\tau;\varphi\right)&=\frac{b_j-\rho\sigma\varphi i+d}{\sigma^2}\left[\frac{1-e^{d\tau}}{1-ge^{d\tau}}\right] \end{aligned} \tag{A1.8}$$

and

$$\begin{aligned} g&=\frac{b_j-\rho\sigma\varphi i+d}{b_j-\rho\sigma\varphi i-d}\\ d&=\sqrt{\left(\rho\sigma\varphi i-b_j\right)^2-\sigma^2\left(2u_j\varphi i-\varphi^2\right)} \end{aligned} \tag{A1.9}$$

Convert the characteristic function back, we can get

$$P_j\left(x,v,T;\ln K\right)=\frac{1}{2}+\frac{1}{\pi}\int_0^\infty \mathrm{Re}\left[\frac{e^{-i\varphi\ln K}f_j\left(x,v,T;\varphi\right)}{i\varphi}\right]d\varphi \tag{A1.10}$$

Then solution of Heston PDE is a direct result.

2. SHAW'S APPROACH FOR EXACT SOLUTION

Start from the Heston PDE, we make transformation:

$$\begin{aligned} \tau &= T - t \\ x &= \lg s + r\tau \\ W &= Ve^{r\tau} \end{aligned} \tag{A2.1}$$

Then use the Fourier transform

$$\begin{aligned} W(x,v,\tau) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-i\omega x}\tilde{W}(\omega,v,\tau)d\omega \\ \tilde{W}(\omega,v,\tau) &= \int_{-\infty}^{\infty} e^{i\omega x}W(x,v,\tau)dx \end{aligned} \tag{A2.2}$$

The initial condition of $\tilde{W}(\omega,v,\tau)$ for European call is

$$\begin{aligned} \tilde{W}(\omega,v,0) &= \int_{-\infty}^{\infty} e^{i\omega x}W(x,v,0)dx = \int_{-\infty}^{\infty} e^{i\omega x}V(x,v,0)dx \\ &= \int_{-\infty}^{\infty} e^{i\omega x}\max(e^x - K,0)dx = \int_{\lg K}^{\infty} e^{i\omega x}(e^x - K)dx \\ &= \int_{\lg K}^{\infty}\left(e^{(1+i\omega)x} - Ke^{i\omega x}\right)dx = \frac{K^{1+i\omega}}{i\omega - \omega^2} \end{aligned} \tag{A2.3}$$

The integral is evaluated at $\mathrm{Im}(\omega) > 1$.

The Fourier transform satisfies PDE

$$\frac{\partial\tilde{W}}{\partial t} = \frac{1}{2}\sigma^2 v\frac{\partial^2\tilde{W}}{\partial v^2} + \left[\kappa(\eta - v) - i\omega\sigma\rho v\right]\frac{\partial\tilde{W}}{\partial v} - \frac{1}{2}v(\omega^2 - i\omega)\tilde{W} \tag{A2.4}$$

Now introduce the Green's function $G(\omega,v,\tau)$. It satisfies the above PDE and it has specific initial condition $G(\omega,v,0) = 1$. So

$$V = \frac{1}{2\pi}e^{-r\tau}\int_{ic-\infty}^{ic+\infty} e^{-i\omega x}\tilde{W}(\omega,v,0)G(\omega,v,\tau)d\omega \tag{A2.5}$$

The Green's function has form

$$G = e^{C(\tau,\omega)+vD(\tau,\omega)} \tag{A2.6}$$

in which

$$C(\tau,\omega)=\frac{\kappa\eta}{\sigma^2}\left\{(\kappa+\lambda+\rho\sigma\omega i+d)\tau-2\ln\left[\frac{1-ge^{d\tau}}{1-g}\right]\right\}$$
$$D(\tau,\omega)=\frac{\kappa+\lambda+\rho\sigma\omega i+d}{\sigma^2}\left[\frac{1-e^{d\tau}}{1-ge^{d\tau}}\right] \qquad \text{(A2.7)}$$

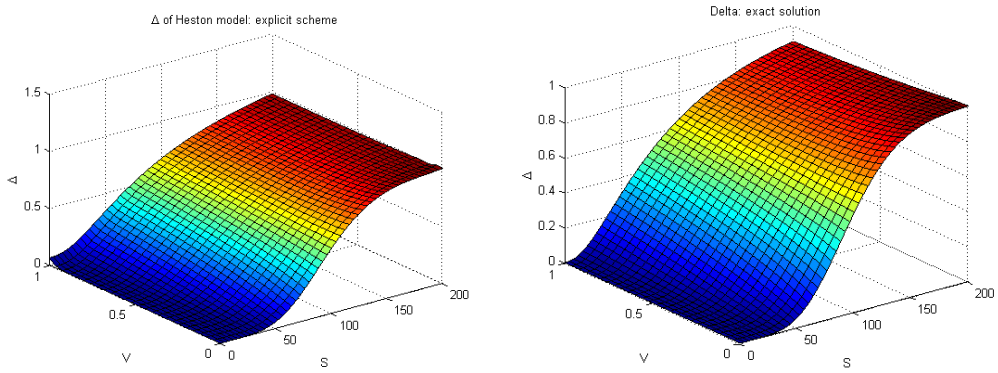
and

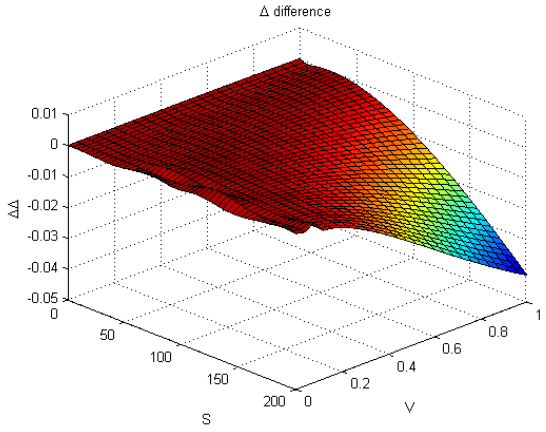
$$g=\frac{\kappa+\lambda+\rho\sigma\omega i+d}{\kappa+\lambda+\rho\sigma\omega i-d}$$
$$d=\sqrt{(\kappa+\lambda+\rho\sigma\omega i)^2-\sigma^2(\omega i-\omega^2)} \qquad \text{(A2.8)}$$

The advantage of Shaw's approach is that the computation of the Greeks is straightforward. Δ and Γ can be got just by multiplying the integrand of (A2.5) with $-\frac{i\omega}{s}$ and $-\frac{\omega^2}{s^2}$ respectively.

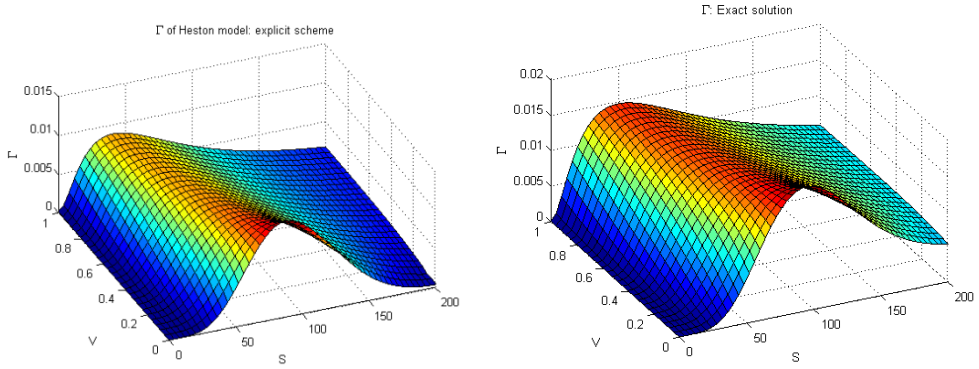
3. THE GREEKS COMPUTED BY EXPLICIT SCHEME AND BY SHAW'S APPROACH.

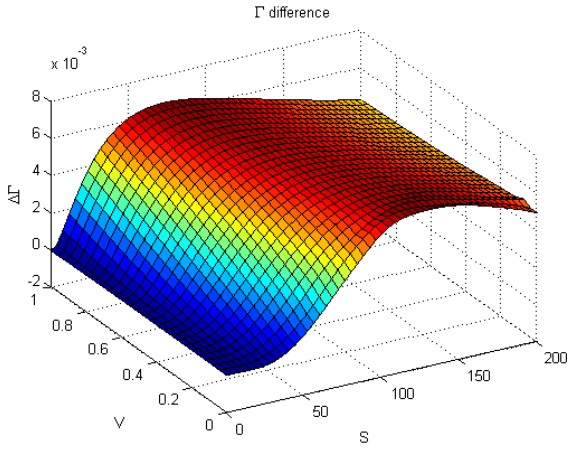
Delta:



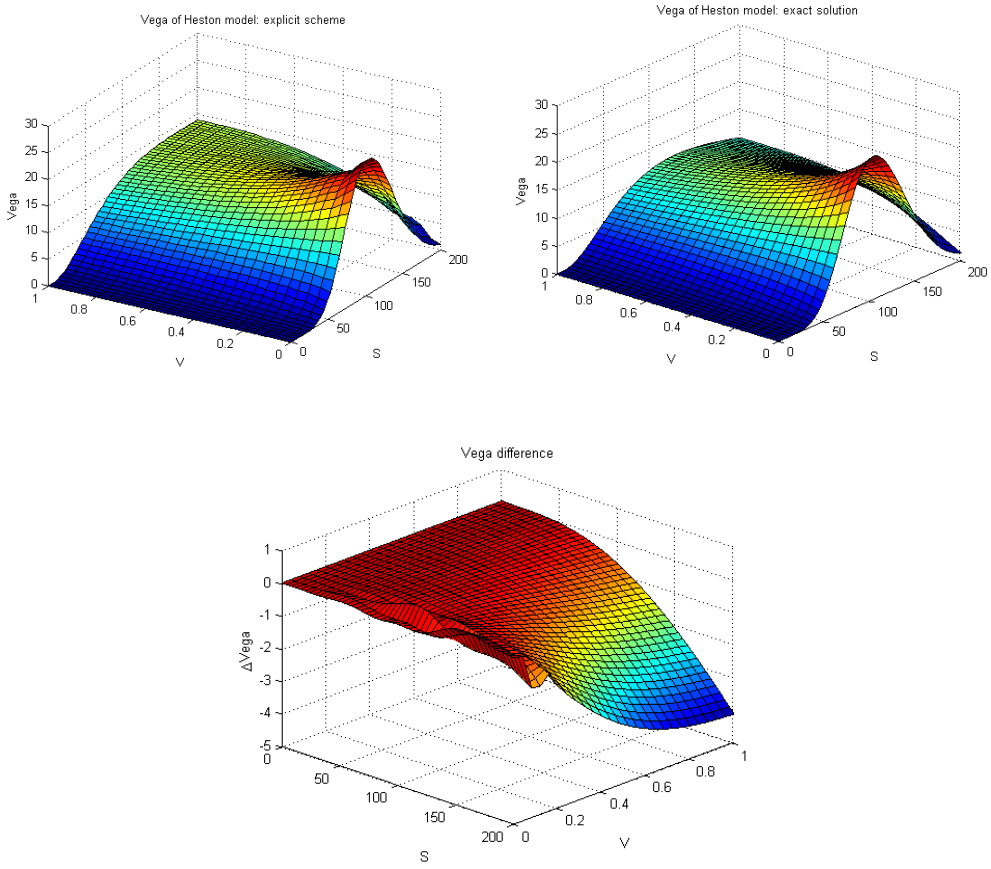

Δ difference
ΔΔ
0.01
0
-0.01
-0.02
-0.03
-0.04
-0.05
0
50
100
150
200
S
0
0.2
0.4
0.6
0.8
1
V

Gamma:


Γ of Heston model: explicit scheme
Γ
0.015
0.01
0.005
0
V
S
Γ: Exact solution
0.02
0.015
0.01
0.005


Γ difference
x 10^-3
ΔΓ
8
6
4
2
0
-2
V
S

Vega:



4. THE CODE IN MATLAB

The code for exact solution:

```
function R3 = HestonCall(S,K,v,r,t,kp,et,sm,rh,ld,trunc,greek)

if nargin < 11, trunc = 100; greek=1; end

I = sqrt(-1);
x = log(S)+r*t;
switch greek
   case 1  % price
      inte = @(w)
(exp(-I*w*x).*(K.^(1+I*w)).*Ffun(t,w,v,kp,ld,et,sm,rh)...
         ./(I*w-w.^2));
   case 2  % delta
      inte = @(w) (-I*w./S.*exp(-I*w*x).*(K.^(1+I*w))...
         .*Ffun(t,w,v,kp,ld,et,sm,rh)./(I*w-w.^2));
   case 3 % gama
```

```
        inte = @(w) (-(w.^2)./(S^2).*exp(-I*w*x).*(K.^(1+I*w))...
            .*Ffun(t,w,v,kp,ld,et,sm,rh)./(I*w-w.^2));
    case 4 % vega
        inte = @(w) (exp(-I*w*x).*(K.^(1+I*w))...
            .*Ffun(t,w,v,kp,ld,et,sm,rh,1)./(I*w-w.^2));
end

R3t = exp(-r*t)*quadgk(inte,-trunc+2i,trunc+2i)/(2*pi);
R3 = real(R3t);

end


function R1 = Ffun(t,w,v,kp,ld,et,sm,rh,Indi)

if nargin<9
    Indi = 0;
end
I = sqrt(-1);

d = sqrt((rh*sm*I*w+kp+ld).^2+(w.^2-I*w).*(sm^2));
g = (kp+ld+rh*sm*I*w+d)./(kp+ld+rh*sm*I*w-d);
D = ((kp+ld+rh*sm*I*w+d)/(sm^2)).*(1-exp(d*t))./(1-g.*exp(d*t));
C =
kp*et*((kp+ld+rh*sm*I*w+d)*t-2*log((1-g.*exp(d*t))./(1-g)))/(sm^2);

if Indi ==0
    R1 = exp(C+D.*v);
else R1 =D.*exp(C+D.*v);
end

end
```

The code for the explicit scheme:

```
% Solutions to Heston PDE using explicit scheme.

function Un=hestonex(kappa,theta,sigma,rho,V,J,r,T,dt,S,I,K,type)

nt=ceil(T/dt);
% dt = T/nt;

% the lower bounds of s and v are both 0
```

```
sbound = S*1.2;
vbound = V*1.3;
ds = S/I; % step length of s
dv = V/J; % step length of v
svalue = 0:ds:sbound;
vvalue = 0:dv:vbound;
ns = size(svalue,2)-1;
nv = size(vvalue,2)-1;
iline = 0:1:ns;
jline = 0:1:nv;


temp = max(0, svalue-K); % as usual, the payoff of European call option.
uinitial = temp'*ones(1,nv+1); % to construct the initial condition of
u.
U = uinitial; % u in dimension of (ns+1)*(nv+1).

for n=1:nt

   % the interior elements. Cross term part.
   Upp = U(3:ns+1,3:nv+1);
   Umm = U(1:ns-1,1:nv-1);
   Ump = U(1:ns-1,3:nv+1);
   Upm = U(3:ns+1,1:nv-1);
   Ut = Upp+Umm-Ump-Upm;
   G0 = rho*sigma*iline'*jline/4;
   G0t = G0(2:ns,2:nv);
   U1t = G0t.*Ut*dt;
   U1 = [zeros(1,nv+1);zeros(ns-1,1)  U1t
zeros(ns-1,1);zeros(1,nv+1)];

   % the elements in S direction.
   B1 = iline(2:ns).*iline(2:ns)/2; B1 = [B1';0;0];
   B2 = -iline(2:ns).*iline(2:ns); B2 = [0;B2';0];
   B3 = iline(2:ns).*iline(2:ns)/2; B3 = [0;0;B3'];
   B = spdiags([B1 B2 B3],[-1 0 1],ns+1,ns+1);

   C1 = -r*iline(2:ns)/2; C1 = [C1';0;0];
   C2 = -r*ones(ns+1,1)/2;
   C3 = r*iline(2:ns)/2; C3 = [0;0;C3'];
   C = spdiags([C1 C2 C3],[-1 0 1],ns+1,ns+1);

   for j=2:nv
```

```
        A = U(:,j);
        G1 = vvalue(j)*B+C;
        U2temp = dt*G1*A; % the interior elements along the j-th sub-vector.
        U2temp(ns+1) = 0; % where s=S;
        U2temp(1) = 0; % where s=0;
        if j==2
            U2 = U2temp;
        else U2 = [U2 U2temp];
        end
    end

    U2 = [zeros(ns+1,1) U2 zeros(ns+1,1)];


%    elements in V direction
    D1 =
sigma*sigma*vvalue(2:nv)/(2*dv*dv)-kappa*(theta-vvalue(2:nv))/(2*dv);
    D1 = [D1';0;0];
    D2 = -sigma*sigma*vvalue(2:nv)/(dv*dv)-r/2;
    D2 = [0;D2';0];
    D3 =
sigma*sigma*vvalue(2:nv)/(2*dv*dv)+kappa*(theta-vvalue(2:nv))/(2*dv);
    D3 = [0;0;D3'];
    D = spdiags([D1 D2 D3],[-1 0 1],nv+1,nv+1);

    for i = 2:ns
        A = U(i,:)';
        U3temp = dt*D*A;
        U3temp(1) = 0;
        U3temp(nv+1) = 0;
        if i==2
            U3 = U3temp';
        else U3 = [U3; U3temp'];
        end
    end

    U3 = [zeros(1,nv+1);U3;zeros(1,nv+1)];


    Uvtank = U(:,1); % the first column of the previous U
    U = U + U1 + U2 + U3 ;
    U(1,:) = zeros(1,nv+1);
    U(:,nv+1) = svalue';
```

```
    % elements where v=0
    for i=2:ns

U(i,1)=(Uvtank(i)-r*i*dt*U(i-1,1)+kappa*theta*dt*U(i,2)/dv)/ ...
          (1-r*i*dt+kappa*theta*dt/dv+r*dt);
    end
    U(ns+1,2:nv) = (2*ds+4*U(ns,2:nv)-U(ns-1,2:nv))/3;

end

if type==1 % price
    Un = U(1:I+1,1:J+1);
elseif type==2 % delta
    Un(1,1:J+1)= (U(2,1:J+1)-U(1,1:J+1))/ds;
    Un(2:I+1,1:J+1)=(U(3:I+2,1:J+1)-U(1:I,1:J+1))/(2*ds);
elseif type ==3 % gamma
    Un(1,1:J+1)= U(2,1:J+1)/(ds^2);
    Un(2:I+1,1:J+1) =
(U(3:I+2,1:J+1)-2*U(2:I+1,1:J+1)+U(1:I,1:J+1))/(ds*ds);
elseif type ==4 % vega
    Un(1:I+1,1) = (U(1:I+1,2)-U(1:I+1,1))/dv;
    Un(1:I+1,2:J+1)=(U(1:I+1,3:J+2)-U(1:I+1,1:J))/(2*dv);
end


end
```

The code for ADI scheme:

```
% DR Scheme

function f = douglas(theta,r,rho,sigma, kappa,eta,V,nv,S,ns,T,nt,K)

dt = T/nt;
dv = V/nv;
ds = S/ns;
svalue = 0:ds:S;
vvalue = 0:dv:V;

u0 = max(svalue'*ones(1,nv+1)-K,0);
```

```
for n = 1:nt
    bnd = boundary(u0,dt,r,V,S,kappa,eta);
    stbnd = u0(ns+1,:);
%    stbnd = stbound(u0,dt,r,V,S,kappa,eta,sigma,theta);
    u1 = u0 + TransA0(u0,rho,sigma,dt) ...
        + (1-theta)*TransA1(u0,r,V,dt) ...
        + TransA2(u0,r,sigma, kappa,eta,V,dt);
    Y = MasA1(u1,theta,r,V,S,dt,stbnd);
    u2 = Y - theta*TransA2(u0,r,sigma, kappa,eta,V,dt);
    u0 = MasA2(u2,theta,r,sigma, kappa,eta,V,S,dt,bnd);
    u0(ns+1,:) = (2*ds+4*u0(ns,:)-u0(ns-1,:))/3;

end

f = u0;

end


% CS scheme

function f = craig(theta,r,rho,sigma, kappa,eta,V,nv,S,ns,T,nt,K)

clear f;
dt = T/nt;
dv = V/nv;
ds = S/ns;
svalue = 0:ds:S;
vvalue = 0:dv:V;
ze = 0.5;

u0 = max(svalue'*ones(1,nv+1)-K,0);


for n = 1:nt
    bnd = boundary(u0,dt,r,V,S,kappa,eta);
    stbnd = u0(ns+1,:);
    u1 = u0 + TransA0(u0,rho,sigma,dt) ...
        + (1-theta)*TransA1(u0,r,V,dt) ...
        + TransA2(u0,r,sigma, kappa,eta,V,dt);
    Y1 = MasA1(u1,theta,r,V,S,dt,stbnd);
    u2 = Y1 - theta*TransA2(u0,r,sigma, kappa,eta,V,dt);
```

```
    Y2 = MasA2(u2,theta,r,sigma, kappa,eta,V,S,dt,bnd);
    Y2(ns+1,:) = (2*ds+4*Y2(ns,:)-Y2(ns-1,:))/3;
    u3 = u1 + ze*TransA0(Y2-u0,rho,sigma,dt);
    Y3 = MasA1(u1,theta,r,V,S,dt,stbnd);
    u4 = Y3 - theta*TransA2(u0,r,sigma, kappa,eta,V,dt);
    u0 = MasA2(u4,theta,r,sigma, kappa,eta,V,S,dt,bnd);
    u0(ns+1,:) = (2*ds+4*u0(ns,:)-u0(ns-1,:))/3;

end

f = u0;

end


% HV scheme

function f = hunds(theta,r,rho,sigma, kappa,eta,V,nv,S,ns,T,nt,K)

clear f;
dt = T/nt;
dv = V/nv;
ds = S/ns;
svalue = 0:ds:S;
vvalue = 0:dv:V;

u0 = max(svalue'*ones(1,nv+1)-K,0);

mew = 0.5;

for n = 1:nt
    bnd = boundary(u0,dt,r,V,S,kappa,eta);
    stbnd = u0(ns+1,:);
    u1 = u0 + TransA0(u0,rho,sigma,dt) ...
       + (1-theta)*TransA1(u0,r,V,dt) ...
       + TransA2(u0,r,sigma, kappa,eta,V,dt);
    Y1 = MasA1(u1,theta,r,V,S,dt,stbnd);
    u2 = Y1 - theta*TransA2(u0,r,sigma, kappa,eta,V,dt);
    Y2 = MasA2(u2,theta,r,sigma, kappa,eta,V,S,dt,bnd);
    Y2(ns+1,:) = (2*ds+4*Y2(ns,:)-Y2(ns-1,:))/3;
    u3 = u1 + mew*TransA0(Y2-u0,rho,sigma,dt) ...
       +mew*(1-theta)*TransA1(Y2-u0,r,V,dt);
    Y3 = MasA1(u1,theta,r,V,S,dt,stbnd);
```

```
    u4 = Y3 - theta*TransA2(u0,r,sigma, kappa,eta,V,dt);
    u0 = MasA2(u4,theta,r,sigma, kappa,eta,V,S,dt,bnd);
    u0(ns+1,:) = (2*ds+4*u0(ns,:)-u0(ns-1,:))/3;

end

f = u0;

end



function f = TransA0 (u0,rho,sigma,dt)

clear f;
ns = size(u0,1)-1;
nv = size(u0,2)-1;
iline = 0:1:ns;
jline = 0:1:nv;

u1 = u0(3:ns+1,3:nv+1) + u0(1:ns-1,1:nv-1) - u0(1:ns-1,3:nv+1)...
    - u0(3:ns+1,1:nv-1);
u2 = zeros(ns+1,nv+1);
u2(2:ns,2:nv) = u1;

f = rho*sigma*dt*(iline'*jline).*u2/4;


end


function f = TransA1(u0,r,V,dt)

clear f;
ns = size(u0,1)-1;
nv = size(u0,2)-1;
dv = V/nv;
vvalue = 0:dv:V;
iline = 1:ns-1;

f = zeros(size(u0));

B1 = iline.^2/2; B1 = [B1';0;0];
```

```
B2 = -iline.^2; B2 = [0;B2';0];
B3 = iline.^2/2; B3 = [0;0;B3'];
B = spdiags([B1 B2 B3],[-1 0 1],ns+1,ns+1);

C1 = -r*iline/2; C1 = [C1';0;0];
C2 = -r*ones(ns-1,1)/2;C2 = [0;C2;0];
C3 = r*iline/2; C3 = [0;0;C3'];
C = spdiags([C1 C2 C3],[-1 0 1],ns+1,ns+1);

    for j=2:nv
        A = u0(:,j);
        G1 = vvalue(j)*B+C;
        U2temp = dt*G1*A; % the interior elements along the j-th sub-vector.
        f(2:ns,j) = U2temp(2:ns);
    end


end


function f = TransA2(u0,r,sigma, kappa,eta,V,dt)

clear f;
ns = size(u0,1)-1;
nv = size(u0,2)-1;
dv = V/nv;
jline = 1:nv-1;

f = zeros(size(u0));

D1 = sigma^2*jline/(2*dv)-kappa*eta/(2*dv)+kappa*jline/2; D1 =
[D1';0;0];
D2 = -sigma^2*jline/dv-0.5*r; D2 = [0;D2';0];
D3 = sigma^2*jline/(2*dv)+kappa*eta/(2*dv)-kappa*jline/2; D3 =
[0;0;D3'];
D = spdiags([D1 D2 D3],[-1 0 1],nv+1,nv+1);

    for i = 2:ns
        A = u0(i,:)';
        U3temp = dt*D*A;
        f(i,2:nv)=U3temp(2:nv);
    end
```

```
end


function f = MasA1(u0,theta,r,V,S,dt,stbnd)

clear f;
ns = size(u0,1)-1;
nv = size(u0,2)-1;
% ds = S/ns;
dv = V/nv;
vvalue = 0:dv:V;
% svalue = 0:ds:S;
iline = 1:ns-1;

f = zeros(size(u0));

for j=2:nv
    stp = u0(2:ns,j);
    a = -theta*(0.5*vvalue(j)*iline.^2*dt - 0.5*r*iline*dt);
    b = 1 + theta*(vvalue(j)*iline.^2*dt + 0.5*r*dt);
    c = -theta*(0.5*vvalue(j)*iline.^2*dt + 0.5*r*iline*dt);
    stp(ns-1) = stp(ns-1) - c(ns-1)*stbnd(j);

    for k=2:ns-1
        m = a(k)/b(k-1);
        b(k) = b(k) - m*c(k-1);
        stp(k) = stp(k) - m*stp(k-1);
    end

    f(ns,j) = stp(ns-1)/b(ns-1);

    for k=ns-2:-1:1
        f(k+1,j) = (stp(k)-c(k)*f(k+1,j))/b(k);
    end


end

end


function f = MasA2(u0,theta,r,sigma, kappa,eta,V,S,dt,bnd)
```

```
clear f;
ns = size(u0,1)-1;
nv = size(u0,2)-1;
ds = S/ns;
dv = V/nv;
svalue = 0:ds:S;
jline = 1:nv-1;

f = zeros(size(u0));

f(:,1) = bnd;

for j=2:ns
    stp = u0(j,2:nv);
    a = -theta*(0.5*sigma^2*jline*dt/dv - 0.5*kappa*eta*dt/dv ...
        + 0.5*kappa*jline*dt);
    b = 1 + theta*(sigma^2*jline*dt/dv + 0.5*r*dt);
    c = -theta*(0.5*sigma^2*jline*dt/dv + 0.5*kappa*eta*dt/dv ...
        - 0.5*kappa*jline*dt);
    stp(1) = stp(1) - a(1)*bnd(j);
    stp(nv-1) = stp(nv-1) - c(nv-1)*svalue(j);

    for k=2:nv-1
        m = a(k)/b(k-1);
        b(k) = b(k) - m*c(k-1);
        stp(k) = stp(k) - m*stp(k-1);
    end

    f(j,nv) = stp(nv-1)/b(nv-1);

    for k=nv-2:-1:1
        f(j,k+1) = (stp(k)-c(k)*f(j,k+1))/b(k);
    end


end

f(:,nv+1) = svalue';

end


% Boundary condition for v=0
```

```
function f = boundary(u0,dt,r,V,S,kappa,eta)

ns = size(u0,1)-1;
nv = size(u0,2)-1;
dv = V/nv;
ds = S/ns;
iline = 0:1:ns;
%
% for i=2:ns
%
f(i,1)=(Uvtank(i)-r*i*dt*f(i-1,1)+kappa*theta*dt*f(i,2)/dv)/ ...
%           (1-r*i*dt+kappa*theta*dt/dv+r*dt);
%    end

% Del = r*iline(2:ns)'.*(u0(2:ns,1)-u0(1:ns-1,1))*dt;
% fin = Del + kappa*eta*(u0(2:ns,2)-u0(2:ns,1))*dt/dv ...
%    - r * u0(2:ns,1)*dt + u0(2:ns,1);

Dels = r*iline(2:ns)'.*(u0(3:ns+1,1)-u0(1:ns-1,1))*dt/2;
Delv = kappa*eta*(4*u0(2:ns,2)-3*u0(2:ns,1)-u0(2:ns,3))*dt/(2*dv);
fin = Dels + Delv - r*u0(2:ns,1)*dt +u0(2:ns,1);

last = (4*fin(ns-1)-fin(ns-2)+2*ds)/3;
f = [0;fin;last];

end
```