A Project on the CEV Model

Christos Saltapidas and Filip Hallqvist

March 6, 2018

Part I

1 Introduction

1.1 The CEV Model

The Constant Elasticity of Variance (CEV) pricing model is a one-dimensional diffusion model with the instantaneous volatility specified to be a power function of the underlying stock price, $S(t)$. The model has been introduced John Cox in 1975 as one of the early alternatives to the geometric Brownian motion to model asset prices. Options prices in the CEV model exhibit implied volatility skews and, for $\delta < 1$, there is a positive probability of hitting zero (bankruptcy).

The CEV model is a one dimensional diffusion process that solves a stochastic differential equation:

$$dS(t) = rS(t)dt + \sigma S(t)^{\delta} d\widetilde{W} \tag{1}$$

with $r \in \mathbb{R}$, $\sigma > 0$, $\delta > 0$ and the instantaneous volatility $\beta(t, S(t)) = \sigma S(t)^{\delta}$ as stated above. Here δ is the elasticity parameter of the local volatility. For $\delta = 1$ the CEV model reduces to the constant volatility Geometric Brownian motion process employed in the Black, Scholes and Merton model.

The result of the relationship between the price and volatility is the implied volatility skew exhibited by options prices in the CEV model. The elasticity parameter δ controls the steepness of the skew (the larger the $|\delta|$, the steeper the skew). This ability to capture the skew has made the CEV model popular in Equity Options Markets.

1.2 Report Outline

In this project we are going to use the CEV model to price derivatives with MATLAB. In the CEV model the price of an Option at time t with maturity $T > 0$ and payoff $Y = g(S(T))$ is given by $\Pi_Y = e^{-r(T-t)}u(t, S(t))$ where $u(t, x)$ solves:

$$\begin{aligned} &\partial_t u + rx\partial_x u + \frac{\sigma^2}{2}x^{2\delta}\partial_x^2 u = 0 \\ &u(T, x) = g(x),\ x > 0 \end{aligned}$$

A more convenient way to work, is to use the equivalent problem which is obtained by the change of variable $t \to T - t$.

$$
\begin{aligned}
&-\partial_t u + rx\partial_x u + \frac{\sigma^2}{2}x^{2\delta}\partial_x^2 u = 0 \qquad (\star)\\
&u(0,x) = g(x),\ x > 0
\end{aligned}
$$

Firstly, we are going to obtain a numerical solution of this PDE using the Crank-Nicholson Method. With this solution then, it is possible for the price of the derivative to be calculated at time t.

MATLAB implementation will help us later to plot initial prices for Vanilla Options and derive Implied Volatility skews.

2 The Crank-Nicholson Approach

This method combines 2 different methods for solving PDE's. It is in fact, the semi-sum of the forward and backward Euler method solutions. More details are presented in the following subsections.

2.1 Forward in time Centered in Space

In this method we introduce the following approximations for $\partial_t u$, $\partial_x u$ and $\partial_x^2 u$.

If $0 = x_0 < x_1 < \cdots < x_m = X$ and $0 = t_0 < t_1 < \cdots < t_n = T$ are the partitions of space and time respectively,

$$
\begin{aligned}
\partial_t u(t,x) &= \frac{u(t+\Delta t, x) - v(t,x)}{\Delta t} + O(\Delta t)\\
\partial_x u(t,x) &= \frac{u(t, x+\Delta x) - v(t,x-\Delta x)}{2\Delta x} + O(\Delta x^2)\\
\partial_x^2 u(t,x) &= \frac{u(t, x+\Delta x) - 2v(t,x) + v(t,x-\Delta x)}{\Delta x^2} + O(\Delta x),
\end{aligned}
$$

where $\Delta x = X/m$ and $\Delta t = T/n$. Substitute them in (*) and after some technical steps we face the following iterative result:

$$
u_{i+1,j} = u_{i,j} + rx\frac{h}{2}(u_{i,j+1} - u_{i,j-1}) - \frac{\sigma^2}{2}x^{2\delta}d(u_{i,j+1} - 2u_{i,j} + u_{i,j-1})
$$

with $j = 0, 1, ..., m-1$, $i = 0, 1, ..., n-1$, $d = \frac{\Delta t}{\Delta x^2}$ and $h = \frac{\Delta t}{\Delta x}$.

2.2 Backward in time Centered in Space

We introduce another approximation for $\partial_t u$ (backwards in time). The rest of them, are the same with section 2.1:

$$
\partial_t u(t,x) = \frac{u(t,x) - v(t-\Delta t, x)}{\Delta t} + O(\Delta t)
$$

With exactly the same steps we derive following equation:

$$
u_{i+1,j} = u_{i,j} + rx\frac{h}{2}(u_{i+1,j+1} - u_{i+1,j-1}) - \frac{\sigma^2}{2}x^{2\delta}d(u_{i+1,j+1} - 2u_{i+1,j} + u_{i+1,j-1})
$$

This is an implicit solution and we need to solve for the solution at time $i+1$ in terms of the solution at time i.

In this case, a $m+1 \times m+1$ matrix will be used to help us re-write the equation and solve for time $i+1$, which was our initial goal.

The equation now in matrix form is:

$$Au_{i+1} = u_i$$

and the elements of the matrix A are defined as:

$$\begin{aligned}
A_{0,0} &= A_{m,m} = 1 \\
A_{k,k} &= 1 + d\sigma^2 x^{2\delta} \\
A_{k,k-1} &= \frac{hrx - dx^{2\delta}\sigma^2}{2}, \\
A_{k,k+1} &= \frac{-hrx + dx^{2\delta}\sigma^2}{2}.
\end{aligned}$$

And finally after that step the backwards method gives the solution in matrix form:

$$u_{i+1} = A^{-1}u_i,$$

which is easier to implement in MATLAB.

2.3 Crank-Nicholson

This method is more accurate, because it basically giving us the 'mean' of the two previous methods. It is obtained by averaging between the methods given in sections 2.1 and 2.2. In other words:

$$u_{i+1,j}(CK) = \frac{1}{2}u_{i+1,j}(FW) + \frac{1}{2}u_{i+1,j}(BW),$$

where $u_{i+1,j}(FW)$ and $u_{i+1,j}(BW)$ are the solutions for the PDE obtained by the forward and the backward method, respectively.

Part II

2.4 The Put-Call Parity

When using finite difference methods to price options using the CEV model, there are three boundary conditions which must be set beforehand. These are $u(0, X)$, $u(t, 0) = u_L(t)$ and $u(t, X) = u_R(t)$. In order to set define these boundaries we will use the put-call parity as defined below.

The put-call parity defines a relationship between the price of a European Call Option and European Put Option, both with the identical Strike price and maturity. The put-call parity is defined as the following:

$$C - P = x - Ke^{-rt},$$

where C is the price of the Call Option at time t, P is the price of the Put Option at time t, x is the Stock price and K is the strike. Remember that we are working with an equivalent problem, where there was a change of variable $t \to T - t$.

Furthermore, as a numerical approach is used for the CEV model, the put-call parity will not always hold. In particular the error increases as the price reaches the upper bound (see figure 1).

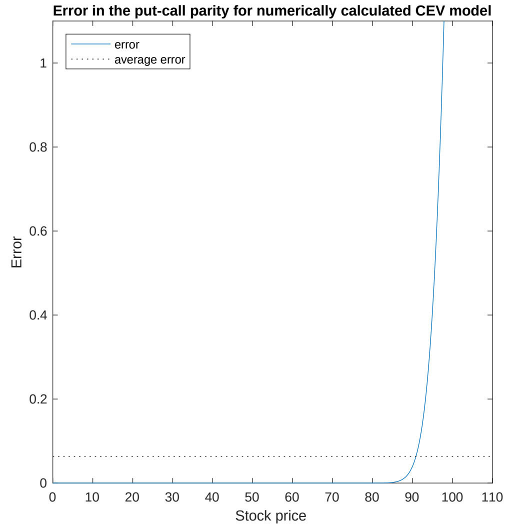

Figure 1: Error in the put-call parity. The put-call parity seem to hold even for the numerical method (until the stock price reaches the upper bound X).

2.5 Boundary Conditions

In order to price an Option, we first have to solve the given PDE. It is know that, for solving PDE's with the methods we used, we need to create a grid and also determine the conditions on the boundary. In this section we are going to do this for Call and Put Options.

These boundary conditions are $u(t, 0) = u_L(t)$ and $u(X, t) = u_R(t)$ for a Call and a Put Option.

Note that $u(t, 0)$ occurs when the underlying stock has defaulted (the price of the stock is 0). A call option on a defaulted stock is worthless, and so we set $u(t, 0) = 0$. In the case of a put option however, we use the put-call parity to find a proper value. By the put-call parity we have $C - P = x - Ke^{-rt}$, and since $C = 0$ we get $u(t, 0) = Ke^{-rt} - x = Ke^{-rt}$, where the last equality holds since $x = 0$.

Moreover, $u(t, X)$ is the value for when the stock price is at its highest possible value. In reality, this value is infinity, but for our purposes it suffices to set it to any large number. We will again make use of the put-call parity, but this time for a call option. By the put-call parity we have $C - P = x - Ke^{-rt}$, and since $P = 0$ we get $u(t, X) = X - Ke^{-rt}$. For a put option we set $u(t, X)$ since clearly the opportunity to sell an asset at any price, when the underlying stock is priced at infinity (or at least a very large value) is worthless.

Lastly, recall that the $u(0, x)$ is the non-discounted value of the option at time of expiration. Clearly this is just the pay-off, $u(0, x) = \max(0, g(x))$.

To sum up, in a more convenient form, our boundary conditions are:

$$u_{i,0} = 0, \quad u_{i,m+1} = X - Ke^{-rt}$$

for the Calls,

$$u_{i,m+1} = 0, \quad u_{i,0} = Ke^{-rt}$$

for the Put Options, and

$$u_{0,x} = 0, \quad \max(0, g(x))$$

for both.

3 CEV Implementation

3.1 Initial Price of Call and Put Options

By using the formulas introduced in section 2, we can calculate the initial price of call and put options as a function of the stock price. By figure 2 we see that when we have $\delta = 1.0$ the CEV model and the Black-Scholes model are equal. However, as $x \to X$ the CEV model starts to deviate. We can mitigate this error by setting $X >> 1$.

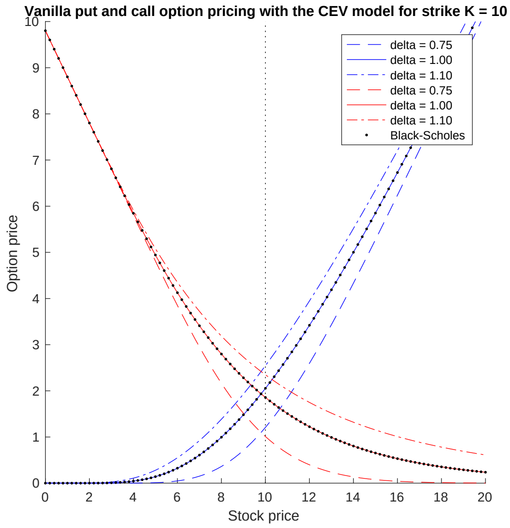

Figure 2: Initial prices for vanilla call and put options as given by the CEV model. Parameters used: $r = 0.02, T = 1, K = 10, \sigma = 0.5$.

3.2 Implied Volatility

If the Black-Scholes model is used to price the option, the implied volatility plot will show a straight line. On the other hand, if the CEV model is used, as in figure 3, we see a volatility smile which is a better model of reality.

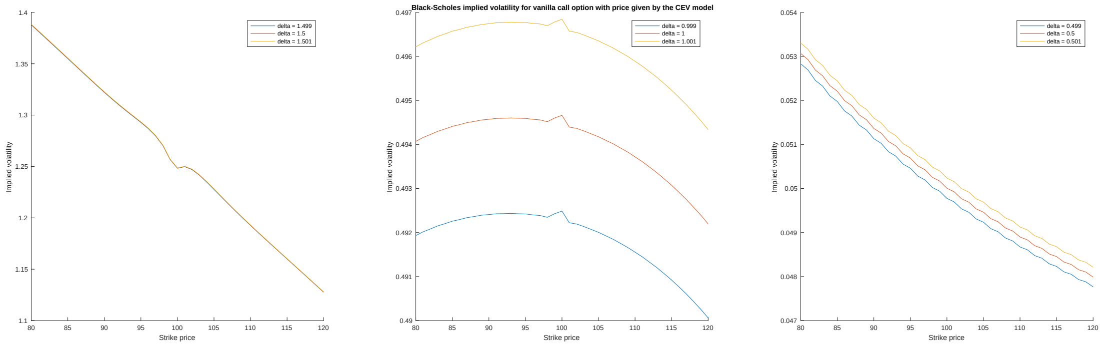

Figure 3: The implied volatility as a function of strike prices. Parameters used: $r = 0.02, T = 1, S_0 = 99.6, \sigma = 0.5$.