Static Replication Methods for Vanilla Barrier Options

with

MATLAB Implementation

Shivum Patel

November 28, 2005
An Honours Project

submitted to

Faculty of Science,
University of the Witwatersrand,
Johannesburg,
South Africa,

in partial fulfillment
of the requirements for a

Bachelor of Science Honours,
Programme in Advanced Mathematics of Finance

by

Shivum Patel
shivum@mweb.co.za

Acknowledgements

I would like to express my sincere thanks and gratitude to Moepa Malataliana from Standard Bank for providing me with extremely useful papers and references on the topic as well as for all his advice and suggestions over the course of this project. My thanks also goes out to my internal supervisor, Prof. David Taylor, for his guidance in ensuring that the scope of the project stayed within realistic bounds and remained on track over the duration that it was completed. Nimalin Moodley needs to be given special thanks for all his assistance with issues pertaining to LaTeX. Finally, I would like to thank Marchand van Rooyen from Standard Bank for proposing the topic; it turned out to be very interesting.
This project discusses two methods for obtaining static replicating portfolios for barrier options. The first method discussed is the method of (Carr & Chou 1997a) and the second is the method of (Derman, Ergener & Kani 1995). The methods are tackled from both a theoretical point of view as well as from a practical implementation point of view. Hence, MATLAB code has also been provided implementing these methods. The inputs and outputs of this code is also discussed. The type of barrier options dealt with in this project are vanilla barrier options. That is, your basic up-and-(out/in) and down-and-(out/in), constant-barrier, standard European put and call options.
# Contents

## 1 Introduction

## 2 The Carr and Chou Method

2.1 Introduction .................................................. 5

2.2 General result for replicating European securities .............. 6

2.3 Determining the equivalent European payoff for vanilla barrier options in the Black-Scholes model .......................... 8

2.4 Implementation .................................................. 12

2.4.1 Inputs & outputs for the MATLAB function **CarrChou.m** 18

2.5 Results on convergence from running the code .................. 20

## 3 The Derman-Ergener-Kani (DEK) Method

3.1 Introduction .................................................. 25

3.2 Justification for the procedure ................................ 25

3.3 Description of the Method ..................................... 27

3.4 Implementation for the CEV & Black-Scholes model .............. 33

3.4.1 The CEV model ............................................. 34

3.4.2 Inputs & outputs for the MATLAB function **DEKCEV.m** . 35

3.5 Results on convergence from running the code .................. 37

## 4 Conclusion

## Appendices

### A No-Arbitrage PDE for Local Volatility Models

A.1 Derivation of the PDE for a derivative’s value ................ 45

A.2 Solving the PDE for the CEV model using a Backward-Time Central-Space Finite Difference Scheme .......................... 46

A.2.1 Inputs & outputs for the MATLAB function **CEVFDS.m** . 49

### B MATLAB CODE

B.1 CarrChou.m .................................................... 51

B.2 DEKCEV.m ..................................................... 52

B.3 CEVprice.m ..................................................... 54

B.4 CEVFDS.m ..................................................... 56

## References

58
1 Introduction

Barrier options have become one of the most heavily traded exotic derivatives in the OTC market, (Carr & Chou 1997b). From a banks point of view an important issue becomes the hedging of these options. The traditional method of doing this is to dynamically replicate the (opposite) option (position) by trading in the underlying. However, there are certain practical difficulties associated with this particular method.

Firstly, in theory, this method of replicating an option requires continuous trading in the underlying. This is not possible in practice and therefore traders compromise by trading in the underlying at “regular” intervals - this creates a hedging error\(^1\). Secondly, the fact that one has to trade in the underlying at regular intervals also means that transaction costs will be incurred. This has a further impact on the hedging error. Moreover, the impact of transaction costs is greatly increased if the gamma of the option being replicated is large. i.e. a high gamma means that the delta\(^2\) of the option is very sensitive to changes in the price of the underlying. Hence, traders need to trade in the underlying more often. i.e. for small changes in the price of the underlying their deltas, before the change, deviates a lot from what it should be after the change in price; implying that they are not adequately hedged. Thirdly, the underlying has a zero vega while the option being hedged certainly does not. This means that should volatility change, the value of the option will change but the value of the replicating portfolio will not be affected. The result: a greater impact on the hedging error due to changes in volatility.

Given that the vega of barrier options is often very high, (Carr & Chou 1997b), and that the gamma of barrier options is also very large over certain regions of the underlying’s value, (Carr & Chou 1997a) and (Carr, Ellis & Gupta 1998), the last two points are important when replicating barrier options dynamically.

There is an alternative to dynamically replicating barrier options, namely the static replication of barrier options with positions in vanilla European options. This method of replication overcomes (or at least reduces the impact of) some of the difficulties associated with dynamic hedging. Firstly, the fact that the hedge is static clearly implies regular trading in the options is not necessary, hence, reducing the impact of transaction costs. The reason why it only reduces the impact of transaction cost is because, as will become apparent in later sections,

\(^1\)Note that in practice any hedging error needs to some how be quantified and added to the price of an option.

\(^2\)The position that needs to be held in the underlying in order to be (delta-) hedged.

1. INTRODUCTION

the static replicating portfolio needs to be liquidated when the stock price hits the barrier level if it is to replicate the barrier option. The second difficulty that is overcome is the mismatch between the vega of the replicating portfolio and the vega of the option being replicated. In other words, since vanilla options are being used to hedge, which certainly have non-zero vegas, one can expect the impact of a changing volatility on the replicating portfolio and the option being replicated to be similar.

However, with this new technique comes new problems. The problems that arise are as follows. Firstly, as would naturally be expected, static replicating portfolios for barrier options are model dependent. That is, in addition to the standard assumptions of no-arbitrage and frictionless markets, assumptions are also made about the process governing the underlying of the derivative. It is then under this model that a static replicating portfolio is obtained. It therefore becomes necessary to test the robustness of a replicating portfolio, obtained from a model, under more realistic and complicated processes for the underlying - both relative to a dynamic strategy (with the same model) as well as against static replicating portfolios obtained from other models. Secondly, as discussed in later sections, to exactly replicate a barrier option one requires an infinite number of positions in vanilla option - obviously this is possible only in theory. However, it will be shown that using a finite number of options, one can get a reasonable replicating portfolio for a barrier option - with convergence occurring as the number of options used increases. Another problem that arises is that the options in which positions need to be taken may not exist in the market. However, the methods discussed are quite flexible in the sense that one has a reasonable amount of freedom in choosing the options with which to replicate the barrier option. Be as it may, it will be impossible to create a reasonable static replicating portfolio for certain barrier options. Another important requirement in order to statically replicate a barrier option, is that the the market in vanilla European puts and call must be liquid. This is because, as already mentioned, in order to replicate a barrier option using the methods that will be discussed, one needs to liquidate the portfolio at the instant that the stock price hits the barrier.

Another application of static replication, besides providing an alternative to dynamic replication, is to price barrier options. What is more, it allows one to price the barrier option off the volatility surface for vanilla European options. I.e. each of the individual vanilla European options making up the static portfolio can be priced using the appropriate implied volatilities for that option in the market. This can then be compared to the price of the barrier option in the market.

\footnote{Though, this is true for dynamic replication as well.}
1. INTRODUCTION

In this project, two methods of statically replicating barrier options are discussed. With regards to these methods, this project concentrates only on vanilla barrier options. More specifically, in the remainder of this project, when referring to “vanilla barrier options”, reference is being made to barrier options with the following characteristics:

- the barrier level is constant;
- the barrier option is either an up-and-(out\in) or a down-and-(out\in) option and
- the barrier option either gives rise to (i.e. knocks-in) or takes away (i.e. knocks-out) a standard European put or call option.

In certain instances, however, the discussions are slightly more general. The following papers deal with more exotic barrier options for the methods or talk about the methods under more general circumstances: (Carr & Chou 1997b), (Carr et al. 1998), (Derman et al. 1995) and (Andersen, Andreasen & Eliezer 2002).

As mentioned earlier, due to the model dependence of a static replicating portfolio, it is necessary to test the robustness of a replicating portfolio under more realistic and complicated processes for the underlying. In order to do such testing for different types of vanilla barrier options, one obviously requires the replicating portfolio under the relevant methods for which testing is being done. Hence, MATLAB functions have been written, which implement the methods for vanilla barrier options. Note that the functions were written in MATLAB 7.0. This is important as the functions may give errors in other versions of the program. Furthermore, the functions require both the financial and statistical packages to be installed. The inputs and the form of the outputs for these functions are also discussed in this project. It is hoped that this will ease the task, in a future project, of performing such tests.

The first method that will be discussed is the method developed by (Carr & Chou 1997a). This will be done in Section 2. The second method that will be discussed is the method developed by (Derman et al. 1995). This is done in Section 3. For each of these methods, first the theoretical reasoning behind the method is given. This is followed by a discussion on the practical implementation of the methods. In other words, how does one actually obtain the replicating portfolio. This is necessary as it is not at all obvious how one gets the replicating portfolio from the theoretical results. In each of the sections, the inputs and outputs for the function written for the particular method is also discussed. The
sections are ended with a brief discussion on certain convergence results, which are obtained from running the implemented functions.
2 THE CARR AND CHOU METHOD

2.1 Introduction

The method of (Carr & Chou 1997a) is set in the Black-Scholes world. It converts the problem of replicating a barrier option to a problem of replicating a *European security*\(^4\), which turns out to have a *non-linear*\(^5\) payoff profile. Given that, in theory, any European security can be replicated with a combination of puts, calls, forwards and bonds, it implies that one can therefore replicate the barrier option. The difficulty is that, in theory, to replicate a European security with a non-linear payoff profile requires an infinite number of positions in vanilla instruments. This means that in practice one can only *approximately* replicate a European security with a non-linear payoff profile, and hence only approximately replicate the barrier option. This approximation is achieved by matching the replicating portfolio’s payoff and the (non-linear) payoff of the security it is replicating, at a finite number of points for the value of the underlying at maturity. As the number of points, at which the matching takes place, increases so the replicating portfolio’s payoff will converge to the payoff of the security being replicated. Hence, so will the value and behaviour (within the model) of the replicating portfolio converge to that of the security being replicated.

The subsections that follow will begin by deriving the result which illustrates, under no major assumptions, that any European security can be replicated with a combination of zero coupon bonds (ZCBs), forwards and vanilla European put and call options. Next it will be shown that under the assumptions of the Black-Scholes model a barrier option can be replicated with a European security with a specifically chosen payoff function. Once such a payoff function is obtained, one can then replicate the barrier option (using puts and calls) by replicating the European security (using puts and calls). It is then shown how one would go about replicating the European security in practice. This will be followed by a description of the inputs and outputs of the MATLAB function which implements this method. Finally, this section is ended with some results, from running the implemented functions, on the convergence of the replicating portfolio’s value and behaviour to that of the barrier option (within the Black-Scholes model).

The subsections that follow have been written with the assistance of the following papers: (Bowie & Carr 1994), (Carr & Chou 1997a), (Carr & Chou 1997b), (Carr

---

\(^4\)That is, a security which entitles the holder to a payoff \(f(S_T)\) at the maturity, \(T\), of the security - with no payments in the interim between initiation and maturity. e.g. a vanilla European put/call option.

\(^5\)In some special cases the payoff profile may be completely linear.
2.2 General result for replicating European securities

The following derivation is taken from the appendix of (Carr & Picron 1999). The assumptions made in the derivation are:

- The payoff function \( f(S_T) \) is twice differentiable.
- There is no arbitrage and markets are frictionless.

Now, with \( I_{\{\cdot\}} \) representing the indicator function, one can proceed as follows:

\[
\begin{align*}
f(S_T) &= f(S_T)(I_{\{S_T \leq k\}} + I_{\{S_T > k\}}) + f(k) - f(k)(I_{\{S_T < k\}} + I_{\{S_T > k\}}) \\
&= f(k) - I_{\{S_T \leq k\}}[f(k) - f(S_T)] + I_{\{S_T > k\}}[f(S_T) - f(k)] \\
&= f(k) - I_{\{S_T \leq k\}} \left[ \int_{S_T}^{k} f'(u)du \right] + I_{\{S_T > k\}} \left[ \int_{k}^{S_T} f'(u)du \right] \\
&= f(k) - I_{\{S_T \leq k\}} \left[ \int_{k}^{u} f'(k)du + f'(u) - f'(k) \right] + \\
&\quad I_{\{S_T > k\}} \left[ \int_{k}^{S_T} f'(k)du \right] \\
&= f(k) - I_{\{S_T \leq k\}} \left[ \int_{S_T}^{k} f'(k)du - \int_{k}^{u} f''(v)dv \right] + \\
&\quad I_{\{S_T > k\}} \left[ \int_{k}^{S_T} f'(k)du + \int_{u}^{S_T} f''(v)dv \right] \\
&= f(k) - I_{\{S_T \leq k\}} \int_{S_T}^{k} f'(k)du + I_{\{S_T > k\}} \int_{k}^{S_T} f'(k)du \\
&\quad + I_{\{S_T \leq k\}} \int_{S_T}^{k} f''(v)dvdu + I_{\{S_T > k\}} \int_{k}^{u} f''(v)dvdu
\end{align*}
\]

Since \( f'(k) \) is not dependant on \( u \), the second and third terms can easily be seen to simplify to \( +I_{\{S_T \leq k\}} f'(k)(S_T - k) + I_{\{S_T > k\}} f'(k)(S_T - k) = f'(k)(S_T - k) \). Next, we change the order of integration for the fourth and fifth terms and then
2.2 General result for replicating European securities

integrate with respect to $u$, giving us:

$$f(S_T) = f(k) + f'(k) (S_T - k)$$
$$+ I_{\{S_T \leq k\}} \int_{S_T}^{k} f''(v) dv + I_{\{S_T > k\}} \int_{k}^{S_T} f''(v) dv$$
$$= f(k) + f'(k) (S_T - k)$$
$$+ I_{\{S_T \leq k\}} \int_{S_T}^{k} f''(v) (v - S_T) dv + I_{\{S_T > k\}} \int_{k}^{S_T} f''(v) (S_T - v) dv$$

Now note that the third term in the last equation is equal to:

$$I_{\{S_T \leq k\}} \int_{S_T}^{k} f''(v) (v - S_T) dv = \begin{cases} \int_{S_T}^{k} f''(v) (v - S_T) dv & \text{if } S_T \leq k \\ 0 & \text{if } S_T > k \end{cases}$$
$$= \begin{cases} \int_{0}^{k} f''(v) (v - S_T)^+ dv & \text{if } S_T \leq k \\ \int_{0}^{k} f''(v) (v - S_T)^+ dv & \text{if } S_T > k \end{cases}$$
$$= \int_{0}^{k} f''(v) (v - S_T)^+ dv$$

Similarly the forth term can be written as $\int_{k}^{\infty} f''(v) (S_T - v)^+ dv$.

$f(S_T)$ can therefore be simplified further to:

$$f(S_T) = f(k) + f'(k) (S_T - k)$$
$$+ \int_{0}^{k} f''(v) (v - S_T)^+ dv + \int_{k}^{\infty} f''(v) (S_T - v)^+ dv$$

Thus a European security’s payoff can be viewed as the payoff arising from a static position in $f(k)$ ZCB’s, $f'(k)$ long forwards and an infinite continuum of put and call options. The value of this portfolio at time $t \ (< T)$, assuming no arbitrage, is therefore:

$$V_t = f(k) B(t, T) + f'(k) (S_T - kB(t, T))$$
$$+ \int_{0}^{k} f''(v) P(t, T, v) dv + \int_{k}^{\infty} f''(v) C(t, T, v) dv$$

where

$B(t, T)$ is the price of a ZCB at time $t$ with maturity $T$,
$P(t, T, v)$ is the price of a put at time $t$, with maturity $T$ and strike $v$ and
$C(t, T, v)$ is the price of a call at time $t$, with maturity $T$ and strike $v$. 

7
2.3 Determining the equivalent European payoff for vanilla barrier options in the Black-Scholes model

This subsection therefore illustrates that the payoff of a European security can be replicated with positions in forwards, ZCB’s and vanilla European options. This result in fact leads to put-call parity when the payoff, \( f(S_T) \), is that of a European put/call option, see (Carr & Chou 1997b).

2.3 Determining the equivalent European payoff for vanilla barrier options in the Black-Scholes model

We begin with a lemma taken from (Carr & Chou 1997b)\(^6\), which underlies the discussion in this subsection.

**Lemma 2.1.** Under the assumptions of the Black-Scholes model, consider a European security expiring at time \( T \) with payoff:

\[
f_1(S_T) = \begin{cases} 
  g(S_T) & S_T \in (A, B) \\
  0 & \text{otherwise}
\end{cases}
\]

Further, for \( H > 0 \), consider another European security with maturity \( T \) and payoff:

\[
f_2(S_T) = \begin{cases} 
  \left( \frac{S_T}{H} \right)^p g(\frac{H^2}{S_T}) & S_T \in (\frac{H^2}{B}, \frac{H^2}{A}) \\
  0 & \text{otherwise}
\end{cases}
\]

where \( p = 1 - \frac{2(r-q)}{\sigma^2} \) and \( r, q \) and \( \sigma \) are the constant NACC interest rate, NACC dividend yield and volatility, respectively. Then, for any \( t \in [0, T] \) the value of these two securities is equal when \( S_t = H \). Note, \( A \) can be 0 and/or \( B \rightarrow \infty \).

**Proof.** Using risk-neutral pricing\(^7\), the value of the payoff \( f_1(S_T) \) when spot at time \( t \) is \( H \), is:

\[
V_1(H, t) = e^{-r(T-t)} \int_A^B g(S_T) \frac{1}{S_T \sqrt{2\pi \sigma^2(T-t)}} \times 
\exp\left(-\frac{(\ln(S_T/H) - (r - q - \frac{1}{2}\sigma^2)(T-t))^2}{2\sigma^2(T-t)}\right) dS_T
\]

Now let \( S_T = \frac{H^2}{S} \) with \( dS_T = -(H^2/\hat{S}^2) d\hat{S} \). Therefore:

\(^6\)But with far more detail included - all done with the compassionate intention of preventing the agony I went through, in filling in the details.

\(^7\)Remember that under the risk-neutral measure \( \ln(S_T) \sim N(\ln(H) + (r - q - \frac{1}{2}\sigma^2)(T-t), \sigma^2(T-t)) \)
2.3 Determining the equivalent European payoff for vanilla barrier options in the Black-Scholes model

\[ V_1(H, t) = -e^{-r(T-t)} \int_{H^2}^{\infty} g \left( \frac{H^2}{S} \right) \frac{1}{S \sqrt{2\pi \sigma^2(T-t)}} \times \exp \left( -\frac{(\ln(H/S) - (r - q - \frac{1}{2}\sigma^2)(T-t))^2}{2\sigma^2(T-t)} \right) d\tilde{S} \]

\[ = e^{-r(T-t)} \int_{H^2}^{\infty} g \left( \frac{H^2}{S} \right) \frac{1}{S \sqrt{2\pi \sigma^2(T-t)}} \times \exp \left( -\frac{[\ln(H/S) + (r - q - \frac{1}{2}\sigma^2)(T-t)]^2}{2\sigma^2(T-t)} \right) d\tilde{S} \]

\[ = e^{-r(T-t)} \int_{H^2}^{\infty} \exp \left( \frac{\ln(H/S) - (r - q - \frac{1}{2}\sigma^2)(T-t)\ln(H/S)}{2\sigma^2(T-t)} \right) \times \exp \left( -\frac{[\ln(H/S) + (r - q - \frac{1}{2}\sigma^2)(T-t)]^2}{2\sigma^2(T-t)} \right) d\tilde{S} \]

But this is exactly the risk-neutral valuation of \( f_2(S_T) \). Hence when \( S_t = H \) we have that \( V_1(H, t) = V_2(H, t) \).

We now use this result to show that a vanilla barrier option can be replicated with a European security with a specifically chosen payoff function. Note that this argument only holds in the Black-Scholes world, which is the main assumption in the previous lemma. We begin with the case of a down-and-out barrier with payoff function \( g(S_T) \) and barrier level \( H (< S_0) \).

**DOWN-AND-OUT OPTION**

Consider a long position in two European securities with payoff functions at time \( T \) as follows:

\[ f_*(S_T) = \begin{cases} g(S_T) & S_T \in (H, \infty) \\ 0 & \text{otherwise} \end{cases} \quad f_2(S_T) = \begin{cases} -\left( \frac{S_T}{H} \right)^p g\left( \frac{H^2}{S_T} \right) & S_T \in (0, H) \\ 0 & \text{otherwise} \end{cases} \]
2.3 Determining the equivalent European payoff for vanilla barrier options in the Black-Scholes model

We claim that the value of these two securities is the same as the value of a down-and-out barrier option with payoff function \( g(S_T) \) and barrier level \( H (< S_0) \). In order to prove the claim, we now consider two mutually-exclusive and exhaustive scenarios for the stock price path over the term of these two securities:

1. The stock price never crosses the barrier level \( H \) over the term of these European securities.

2. The stock price does cross the barrier level \( H \) at some time over the life of these European securities.

Under scenario 1, the payoffs at maturity are \( f_2(S_T) = 0 \) and \( f_\star(S_T) = g(S_T) \). This gives a combined portfolio payoff, at maturity, exactly equal to that of the barrier option under this scenario.

We now consider scenario 2. We will make use of the previous lemma to show that the portfolio consisting of the above two securities has the same payoff as the barrier option under this scenario.

Note from the lemma that at the time the stock hits the barrier, the value of the European security with payoff \( f_2(S_T) \) is exactly the same as the value of a European security with payoff function,

\[
f_1(S_T) = \begin{cases} 
-g(S_T) & S_T \in (H, \infty) \\
0 & \text{otherwise}
\end{cases}
\]

Hence, at this time (i.e. \( S_t = H \)), we can sell the payoff \( f_2(S_T) \) and buy the payoff \( f_1(S_T) \) at no cost. The new payoff of the portfolio at maturity, once this is done, is zero: exactly the payoff of the barrier option under this scenario. That is,

\[
f_1(S_T) + f_\star(S_T) = \begin{cases} 
g(S_T) - g(S_T) = 0 & S_T \in (H, \infty) \\
0 & \text{otherwise}
\end{cases} = 0
\]

Following from the strategy described in the previous paragraph, it can equivalently be said that when the stock price hits the barrier, we should liquidate our portfolio. The gain/loss of doing this will be zero. This is exactly a consequence of the argument given in the previous paragraph. The “payoff” at maturity after doing this will obviously be zero: exactly the payoff of the barrier option under this scenario.

From the arguments given above for the two mutually-exclusive and exhaustive scenarios, together with the no-arbitrage condition, we have that the down-and-out vanilla barrier option can be replicated with a European security with payoff
2.3 Determining the equivalent European payoff for vanilla barrier options in the Black-Scholes model

function

\[ f(S_T) := f_1(S_T) + f_2(S_T) = \begin{cases} 
  g(S_T) & S_T \in (H, \infty) \\
  -\left(\frac{S_T}{H}\right)^p g\left(H^2 \frac{S_T}{H^2}\right) & S_T \in (0, H) 
\end{cases} \]  

Therefore replicating a European security with such a payoff function is the same as replicating the barrier option. Furthermore, we know that the former can be done, from the result in subsection 2.2.

**Definition 2.2. (Adjusted Payoff)** The payoff function of a European security that can be used to replicate a barrier option, is known as the adjusted payoff function.

The payoff function (1) is an example of an adjusted payoff function.

**Down-and-in Option**

Using in-out parity, the adjusted payoff function for a down-and-in vanilla barrier option with payoff function \( g(S_T) \) and barrier level \( H (< S_0) \) is:

\[ f(S_T) = \begin{cases} 
  0 & S_T \in (H, \infty) \\
  g(S_T) + \left(\frac{S_T}{H}\right)^p g\left(H^2 \frac{S_T}{H^2}\right) & S_T \in (0, H) 
\end{cases} \]  

**Up-and-out Option**

Similar arguments to the down-and-out case can be used to show that the adjusted payoff function for an up-and-out option is:

\[ f(S_T) = \begin{cases} 
  -\left(\frac{S_T}{H}\right)^p g\left(H^2 \frac{S_T}{H^2}\right) & S_T \in (H, \infty) \\
  g(S_T) & S_T \in (0, H) 
\end{cases} \]  

**Up-and-in Option**

Once again, using in-out parity the adjusted payoff function for an up-and-in barrier option is:

---

8Note that the replicating portfolio for the European security with the adjusted payoff will have to be liquidated in order for it to replicate the barrier option. That is, since this portfolio is replicating the European security with the adjusted payoff, it too will result in a zero gain/loss if liquidated when \( S_t = H \) and hence result in a “payoff” of zero under scenario 2 - exactly the payoff of the barrier option under this scenario.

9That is, the barrier option knocks-in if the stock price hits the level \( H \), and gives rise to a European security with payoff function \( g(S_T) \). An analogous meaning is intended for the case of an up-and-out and up-and-in option, which follow.
2.4 Implementation

\[ f(S_T) = \left\{ \begin{array}{ll}
g(S_T) + \left( \frac{S_T}{H} \right)^p g\left( \frac{H^2}{S_T} \right) & S_T \in (H, \infty) \\
0 & S_T \in (0, H)
\end{array} \right. \]

This subsection is concluded by noting that in order to statically replicate a vanilla barrier option (using this method) all one needs to do is statically replicate a European security providing the adjusted payoff. Remember that holding a security with such a payoff is equivalent to holding the vanilla barrier option. Furthermore, it known from subsection 2.2 that one can statically replicate any European security.

It is important to point out that in all the cases dealt with above, the adjusted payoff functions are non-linear over certain regions of their domains. This is because \( p \), in the adjusted payoff functions, is not in general equal to zero or one. Hence, replicating a European security that provides such a payoff is not obvious and requires some discussion\(^{10}\). This is done next.

2.4 Implementation

An example will now be given so as to make more concrete the ideas presented in the previous two subsections as well as to explain how this method would be implemented in practice. In other words, how does one actually create the static replicating portfolio of vanilla puts and calls; this is not obvious from the previous subsections. This subsection fully develops, with a lot more detail, a brief explanation of how one might go about doing this found in (Nalholm & Poulsen 2005).

The option that will be replicated in this example is a down-and-out European call option with barrier level \( H \) and strike \( K \) (\( < H \)). The adjusted payoff in this case is:

\[ f(S_T) = \left\{ \begin{array}{ll}
(S_T - K)^+ & S_T \in (H, \infty) \\
\left( \frac{S_T}{H} \right)^p \left( \frac{H^2}{S_T} - K \right)^+ & S_T \in (0, H)
\end{array} \right. \]

When plotted this payoff looks something like the function in Figure 1.

Recalling the conclusion in the previous subsection, a security with such a payoff function can be made to replicate a down-and-out call by taking the appropriate course of action when the share price hits the barrier level - namely, the liquidation of the position. Hence replicating this payoff function is equivalent to replicating\(^{10}\) it should be obvious that any linear payoff profile can be replicated exactly with a finite number of positions in vanilla European puts and calls and binary European puts and calls.

\(^{10}\)
the barrier option\footnote{As mentioned in a previous footnote, this replicating portfolio of the adjusted payoff will also have to be liquated when the stock price hits the barrier level if it is to replicate the barrier option.}. One, therefore, proceeds to replicate this adjusted payoff. Remember that the theme of this project is to obtain a replicating portfolio of \textit{vanilla put and call options} for the barrier option. Hence, the aim will be to replicate the adjusted payoff using vanilla put and call options.

\textbf{Replicating The Adjusted Payoff with Vanilla European Puts and Calls}

Figure 2 will be useful in explaining how one goes about doing this.

This payoff function is replicated in two parts. First the adjusted payoff above the barrier level is replicated with vanilla puts and calls - this can be done exactly for vanilla barrier options. Once this is done, the aim is to replicate the payoff below the barrier with vanilla puts and calls. The latter is more tricky, due to the non-linearity of the payoff function in this region, and therefore requires more thought. In fact, from a practical point of view, one can only replicate this non-linear portion of the payoff, using vanilla European options, at a finite number of points. Of course, this approximation becomes better and better as one increases
2.4 Implementation

Figure 2: Replicating the adjusted payoff for a down-and-out call with $K < H$

The number of points at which the portfolio payoff is matched to the adjusted payoff. This is illustrated in Figure 3\textsuperscript{12}.

The procedure to obtain a portfolio of vanilla put and call options that (approximately) replicates the adjusted payoff is explained next. Please refer to Figure 2 when reading, and trying to understand, the arguments and explanations that follow.

To replicate the adjusted payoff above the barrier is simple. One simply needs to long a non-barrier version of the call option and we are done.

Next, it is explained how to create a replicating portfolio of vanilla put and call options to match the adjusted payoff below the barrier. We will begin by replicating the adjusted payoff at one, of an entire range, of final share price values below the barrier. Let this point be $x_1$. This is done by taking a position in a vanilla option such that the payoff of this option plus the payoff of the call (purchased to replicate the adjusted payoff above the barrier) will have the same

\textsuperscript{12}The technique explained next, to replicate the adjusted payoff function, was used to create this figure. Note further that all the points at which the replicating portfolio’s payoff matches the exact payoff may not be visible as the domain over which the figure is plotted has been restricted.
2.4 Implementation

Figure 3: Illustration of the convergence of the portfolio payoff to the exact adjusted payoff as the number of points at which the adjusted payoff is matched, increases. Note only the non-linear portion of the payoff is shown, as the linear portion is exactly matched by a single position in the non-barrier version of the barrier option. Hence, the “matching” takes place only in the non-linear region of the adjusted payoff.

payoff as the adjusted payoff should the final share price be $x_1$. Now there are a couple of points to note about this option in which the position will be taken:

1. To ensure that the second vanilla option position does not interfere with the already matched payoff above the barrier, this position will have to be in a put option with a strike below the barrier.

2. Furthermore, the strike of this put will have to be greater than $x_1$. If this is not the case then the option will always have a value of zero should the final share price be $x_1$ and, hence, no position in this option can be taken to target a certain payoff at the point $x_1$.

Let the strike of this put option be $K_1$.

Now, let $\alpha_1$ be the position required in this put option to ensure that the portfolio (consisting of the call and this put) will have the same payoff as the adjusted payoff if the final share price is $x_1$. The equation that needs to be solved to determine $\alpha_1$ is:

$$\alpha_1(K_1 - x_1)^+ + (x_1 - K)^+ = f(x_1)$$

$$\Rightarrow \quad \alpha_1(K_1 - x_1)^+ = f(x_1) - (x_1 - K)^+$$
2.4 Implementation

Hence, a portfolio of vanilla European options has been created which has the same payoff as the adjusted payoff everywhere above the barrier and at a single final share price value below the barrier.

Next the aim is to achieve this matching\textsuperscript{13} for a second final share price value below the barrier, \( x_2 \) say. For the same reason as before, a put option is used i.e. so as not to interfere with the matched payoff above the barrier. Moreover, the strike of this second put must be less that \( x_1 \) so that the matched payoff at \( x_1 \) is not altered. Finally, as before, the strike of the second put must be greater than \( x_2 \). The equation which now needs to be solved to obtain \( \alpha_2 \) (the position required in a second put so as to match the payoff of the portfolio and the adjusted payoff for a final share price of \( x_2 \)) is:

\[
\alpha_2(K_2 - x_2)^+ + \alpha_1(K_1 - x_2)^+ + (x_2 - K)^+ = f(x_2)
\]

\[
\alpha_1(K_1 - x_2)^+ + \alpha_2(K_2 - x_2)^+ = f(x_2) - (x_2 - K)^+
\]

Continuing in this way for final share price values \( x_3 \) and \( x_4 \) below the barrier, one obtains the following set of equations:

\[
\alpha_1(K_1 - x_1)^+ = f(x_1) - (x_1 - K)^+
\]

\[
\alpha_1(K_1 - x_2)^+ + \alpha_2(K_2 - x_2)^+ = f(x_2) - (x_2 - K)^+
\]

\[
\alpha_1(K_1 - x_3)^+ + \alpha_2(K_2 - x_3)^+ + \alpha_3(K_3 - x_3)^+ = f(x_3) - (x_3 - K)^+
\]

\[
\alpha_1(K_1 - x_4)^+ + \alpha_2(K_2 - x_4)^+ + \alpha_3(K_3 - x_4)^+ + \alpha_4(K_4 - x_4)^+ = f(x_4) - (x_4 - K)^+
\]

This can be written in matrix notation as:

\[
\begin{pmatrix}
(K_1 - x_1)^+ & 0 & 0 & 0 \\
(K_1 - x_2)^+ & (K_2 - x_2)^+ & 0 & 0 \\
(K_1 - x_3)^+ & (K_2 - x_3)^+ & (K_3 - x_3)^+ & 0 \\
(K_1 - x_4)^+ & (K_2 - x_4)^+ & (K_3 - x_4)^+ & (K_4 - x_4)^+
\end{pmatrix}
\begin{pmatrix}
\alpha_1 \\
\alpha_2 \\
\alpha_3 \\
\alpha_4
\end{pmatrix}
= \begin{pmatrix}
f(x_1) - (x_1 - K)^+ \\
f(x_2) - (x_2 - K)^+ \\
f(x_3) - (x_3 - K)^+ \\
f(x_4) - (x_4 - K)^+
\end{pmatrix}
\]

Hence one can now easily obtain the positions required in the put options so as to match the adjusted payoff for final share price values of \( x_1, x_2, x_3 \) and \( x_4 \). Having completed this exercise, an approximate replicating portfolio for a European security providing the adjusted payoff (and hence an approximate replicating portfolio for the barrier option) is:

- A single long position in the non-barrier version of the barrier option being replicated - in this case, a vanilla European call option.
- A position \( \alpha_1 \) in a put with strike \( K_1 \) and maturity \( T \).
- A position \( \alpha_2 \) in a put with strike \( K_2 \) and maturity \( T \).
- A position \( \alpha_3 \) in a put with strike \( K_3 \) and maturity \( T \).

\textsuperscript{13}i.e. between the replicating portfolio’s payoff and the adjusted payoff.
2.4 Implementation

- A position $\alpha_4$ in a put with strike $K_4$ and maturity $T$.

This approximation can be made more accurate by increasing the number of points, $x_i$, at which the replicating portfolio’s payoff is matched to the adjusted payoff in the non-linear region of the latter.

Note that the maturities of the vanilla European options making up the replicating portfolio are the same as maturity of the barrier option being replicated. The strikes, however, vary. This is in contrast to the method of (Derman et al. 1995), discussed in section 3, where the strikes of the vanilla options (making up the replicating portfolio) are all the same but the maturities vary.

Notes On The Procedure For Other Types Of Vanilla Barrier Options

1. A similar procedure can be used for down-and-out put options.

2. Given the replicating portfolio for down-and-out options, in-out-parity allows one to obtain the replicating portfolio for down-and-in call and put option. Therefore, once the above procedure has been implemented for down-and-out options there is no need to implement the whole thing for down-and-in options. The replicating portfolio for down-and-in options is simply a call/put\(^{14}\) minus the replicating portfolio for the corresponding down-and-out option.

Figure 4 is useful for highlighting points about the procedure for up-and-out options, which otherwise is very similar.

1. The payoff below the barrier is easily replicated by a single position in a non-barrier version of the barrier option being replicated.

2. To replicate the payoff above the barrier, we need to use call options. The reason for this is to prevent altering the already matched payoff below the barrier.

3. For points above the barrier at which one wants to match the payoff of the replicating portfolio to the adjusted payoff, the strikes needs to be less than these points. For example, $K_4 < x_4$. The reason for this is the same as the down-and-out case.

4. Finally, in-out parity can once again be used to obtain the replicating portfolio for an up-and-in vanilla barrier option.

---

\(^{14}\)Depending on whether it is a down-and-in call or down-and-in put.
2.4 Implementation

Figure 4: Adjusted payoff for a up-and-out call with $K < H$

This method has been implemented in MATLAB for vanilla barrier options. The name of the function is `CarrChou.m` and the code for this function can be found in Appendix B.1. The inputs for this function are discussed next.

2.4.1 Inputs & outputs for the MATLAB function `CarrChou.m`

The function

\[ [\text{output value}] = \text{CarrChou}(\text{int}, \text{down}, \text{out}, \text{So}, \text{strike}, B, K, x, \sigma, r, q, \text{term}) \]

implemented in MATLAB, calculates the replicating portfolio for any vanilla barrier option using the methodology of Carr & Chou as described in subsection 2.4. The inputs of this function represent:

1. **int** - takes on the value 1 if the vanilla barrier option to be replicated is a call version and the value -1 if it is a put version.
2. **down** - takes on the value 1 if a **down** version of the vanilla barrier option is to be replicated and -1 if an **up** version of the vanilla barrier option is to be replicated.
3. **out** - takes the value 1 for a **knock-out** option and 0 for a **knock-in** option.
2.4 Implementation

4. So, strike, B, r, sigma, q and term - these are the initial share price, the strike price, the barrier level, the NACC risk free rate, the volatility, the NACC dividend yield and the term of the option, respectively.

5. K - This is a row vector of the strike prices of the options with which one wants to replicate the barrier option. i.e. the \( K_i \)'s in Figures 2 and 4 in subsection 2.4.

6. x - This is a row vector of the final share price values at which the replicating portfolio (of put and/or call options) should match the adjusted payoff function. i.e. the \( x_i \)'s in Figure 2 and Figure 4 in subsection 2.4.

The restrictions on the last two inputs, K and x, are as described in subsection 2.4. Namely, for down options the entries in the K vector must be greater than the corresponding entries in the x vector and for up options the entries should be less than the corresponding entries in the x vector. Another important point about the last two inputs is that the row entries of these vectors should be in increasing order. Therefore, for the payoff functions in Figures 2 and 4, these inputs would be 

\[
K = [K_4, K_3, K_2, K_1] \quad \text{and} \quad x = [x_4, x_3, x_2, x_1].
\]

When using this function for “up” options, the range over which the \( K_i \)'s are chosen would be between the barrier level and some “large” value greater than the barrier level. Note that if this “large” value is not sufficiently large the output of this function may give ridiculous answers - so take care\(^\text{15}\). When using this function for “down” options, the range over which the \( K_i \)'s are chosen is between zero and the barrier level, and no problems should be experienced. Furthermore, matching the replicating portfolio’s payoff to the adjusted payoff for values of the final stock price “far” away from the barrier level (i.e. close to zero for down-options and close to the “large” value for up-options), has a very small impact on the value and behaviour of the replicating portfolio - experiment with the functions provided to see this.

The outputs of this function are output and value. output is a matrix with the first column giving the positions that must be taken in various vanilla options in order to replicate the barrier option. The second column gives the strike prices of the options in which the positions must be taken. The maturity of all these options is equal to the maturity of the barrier option being replicated. Therefore, the maturities of the options are not included in the matrix. The type of option in

\[^{15}\text{A technique to test if a value is “large” enough is to plot the blue graphs in Figure 5 or Figure 6 but for an up-option - in these figures it is drawn for down-options. If such a graph oscillates violently then it means that the chosen value is too small. This conclusion was reached via experimentation.}\]
which the position must be taken is indicated in the third column of this matrix. In this column, 1 indicates that the position must be taken in a call option and -1 indicates that it must be taken in a put option. Finally, the last column gives the price of the options under the Black-Scholes model.

The other output of this function, value, gives the total value of the option positions (i.e. the value of the replicating portfolio) using the Black-Scholes pricing formulae for put and call options\(^{16}\). This can be compared to the exact price of the barrier option under the Black-Scholes model - for which closed-form option pricing formulae exist, see (Hull 2003).

### 2.5 Results on convergence from running the code

In this subsection, the convergence of the value/price and the behaviour of the static replicating portfolio, to that of the barrier option, is illustrated. The function `CarrChou.m`, discussed in the previous subsection, is used to obtain the replicating portfolios needed to create the diagrams. It is shown that as the number of points at which the two payoffs (i.e. the replicating portfolio’s payoff and the adjusted payoff) are matched (in the non-linear region) increases, so the replicating portfolio’s value and behaviour tends to that of the barrier option.

We begin by showing the convergence of the price of the replicating portfolio to the price of the barrier option. These prices are determined using the Black-Scholes model because the method of (Carr & Chou 1997a) is derived in this world, and therefore convergence is expected in this world.

Figure 5 illustrates the convergence in price for a down-and-in call option with parameters \(S_0 = 50, K = 40, H = 45, T = 2, \sigma = 0.25, r = 0.1\) and \(q = 0\). It can be seen that the price of the replicating portfolio converges to within 2% of the exact price. More importantly, though, is that the price converges after matching the two payoffs (in the non-linear region) at approximately 40 to 45 points (checked with the actual data used to create the plot).

Figure 6 illustrates the convergence in price for a down-and-out option with the same parameters as before. In this case the convergence is to within 1.6% of the exact price. Again, convergence occurs after matching the two payoffs at approximately 40 to 45 points (checked with the actual data used to create the plot).

Note that the static replicating portfolio underestimates the price of the “in”-barrier option and overestimates the price of the “out”-barrier option. This is

\(^{16}\)i.e. the dot product of the first column and last column of the output matrix.
2.5 Results on convergence from running the code

Figure 5: Convergence of the value of the replicating portfolio to the exact price of the barrier option. The barrier option in this case is a down-and-in call option. The following parameters were used to obtain the diagram: \( S_0 = 50, K = 40, H = 45, r = 0.1, q = 0, T = 2 \) and \( \sigma = 0.25 \).

Figure 6: Convergence of the value of the replicating portfolio to the exact price of the barrier option. The barrier option in this case is a down-and-out call option. The following parameters were used to obtain the diagram: \( S_0 = 50, K = 40, H = 45, r = 0.1, q = 0, T = 2 \) and \( \sigma = 0.25 \).

useful as, in practice, the price of a barrier option obtained using closed-form solutions (within the Black-Scholes model), overestimates the price of “in” barrier options and underestimates the price of “out” barrier options. This over/under-
2.5 Results on convergence from running the code

estimation is significant, see (West 2005). This is due to the fact that these closed form solutions are obtained under the assumption that the barrier is observed continuously, whereas in practice this is not the case.

The set of surfaces in Figure 8, illustrates the behaviour, within the Black-Scholes model, of the replicating portfolio as the number of options used to create a replicating portfolio is increased\textsuperscript{17}. The option being replicated for this illustration is an up-and-out call and the parameters being used are $S_0 = 35$, $K = 30$, $H = 60$, $T = 2$, $\sigma = 0.25$, $r = 0.1$ and $q = 0$. The first row of surfaces gives the results when 21 options are used to replicate the barrier option. The second row gives the results when 51 options are used and the last row is when 101 options are used.\textsuperscript{18} In each row, the first surface (i.e. in the first column) gives the value of the replicating portfolio for varying times to maturity and stock prices. The second surface in each row (i.e. in the second column) gives the absolute error, as a % of the exact value, between the value of the replicating portfolio and the exact value of the barrier option, for varying times to maturity and stock prices. The surface of the exact value of the barrier option is given in Figure 7.

\textbf{Figure 7: Surface of the exact value of the barrier option at various points in time and “space” (i.e. stock price).}

\textsuperscript{17}That is, as the number of points (along the barrier) at which the replicating portfolio’s value is matched to the value of the barrier option is increased.

\textsuperscript{18}In each case, one of the options matches the replicating portfolio’s payoff and the adjusted for all final stock price values below the barrier. The remaining options match the payoffs at a finite number of points above the barrier, where the adjusted payoff is non-linear.
2.5 Results on convergence from running the code

Figure 8: Surfaces of the value of replicating portfolio, together with the corresponding surface of the absolute error between the replicating portfolio’s value and the exact value of the barrier option. The surfaces in the first row are for a replicating portfolio consisting of 21 options and the second and third rows give the surfaces when 51 and 101 options are used, respectively. The replicating portfolio’s are obtained using the Carr and Chou method.
2.5 Results on convergence from running the code

The following observations can be made from Figure 8:

- The convergence in shape and level of the surface of the value of the replicating portfolio, is evident on comparing the surfaces in the first column of Figure 8 and the surface in Figure 7. Note that the axes for all the surfaces are on the same scale. This is also true for the colour bars, which give an indication of level of the surface on the z-axis. These two features of the plots in question allows a comparison to be made.

- The second column of surfaces is more revealing. These surfaces illustrate the percentage discrepancy between the exact surface and that of the replicating portfolio. In all cases, a prominent feature is the “blow up” in the discrepancy (though, on different scales) when the share price is close to the barrier level. This blow up occurs for all values of time when the share price is close to the barrier - though it is far more exaggerated as the maturity of the barrier option is approached. Note that the dark brown in these figures are regions where the discrepancy is totally off the colour bar’s scale.

- Lastly, note how the level of the values for the discrepancy is greatly reduced as the number of options used to replicate is increased. That is, when 21 options are used to replicate, the discrepancy near the barrier is well above the 30% mark. In this case, the discrepancy is large even when the stock price is not very close to the barrier level - e.g. when the stock price in the range 20-40, there are times where the discrepancy is approximately 10-20%. However, when we use 51 options to create a replicating portfolio, the discrepancy near the barrier plummets to around the 5-8% level. And when a 101 options are used, this discrepancy becomes negligible - except of course when the stock price is very close to the barrier and particularly so as the option approaches maturity. However, even under such a scenario (i.e. close to maturity and/or the stock price near the barrier) the discrepancy, of at most 8%, is still reasonable.
3 The Derman-Ergener-Kani (DEK) Method

3.1 Introduction

This method can be implemented for any model for which the problem of pricing a derivative can be reduced to solving a single no-arbitrage PDE with certain boundary conditions\(^ {19}\). This is true of models where, in addition to the standard assumptions of no-arbitrage and frictionless markets, the process for the underlying is of the form:

\[
\frac{dS_t}{S_t} = \mu(t, S_t)dt + \sigma(t, S_t)dW_t
\]

Such models are usually called local volatility models. Given that, in general, such a model does produce a volatility smile, a replicating portfolio (of puts and call) for a barrier option obtained from such a model would be preferred to a replicating portfolio obtained from a model that did not produce a smile. This does suggest calibrating such a local volatility model using market information and then using the calibrated model to obtain a replicating portfolio using this method. However, the feasibility of doing this would need to be investigated.

In the subsections that follow, first a justification will be given for the method of (Derman et al. 1995). This will be followed by a description of how one actually gets the static replicating portfolio of vanilla puts and calls (for vanilla barrier options) using this method. The description given is quite different to the one given in the original paper by (Derman et al. 1995)\(^ {20}\), and follows more along the lines of the description given in (Nalholm & Poulsen 2005). Following the discussion of the method, the inputs and outputs for the MATLAB function implementing this method for the CEV and Black-Scholes model, will be given. Finally, this section is ended with some results, from running the implemented functions, on the convergence of the replicating portfolio’s value and behaviour to that of the barrier option.

3.2 Justification for the procedure

The justification for the methodology will be given in the case of an up-and-out call. The reasoning, however, is easily extended for other types of vanilla barrier options.

\(^{19}\)The boundary conditions are dependent on the derivative for which a price is required.

\(^{20}\)The explanation in this paper was given in a binomial tree setting.
3.2 Justification for the procedure

For a local volatility model of the form (5), a no-arbitrage PDE can be obtained for the price of a derivative. This PDE is given by:

\[
\frac{\partial f}{\partial t} + rS_t \frac{\partial f}{\partial S_t} + \frac{1}{2} \sigma^2(t, S_t) S_t^2 \frac{\partial^2 f}{\partial S_t^2} = r f
\]  

(6)

The derivation of this PDE can be found is Appendix A.1. Let us define the linear partial differential operator \( L \) as follows:

\[
L = \frac{\partial}{\partial t} + rS_t \frac{\partial}{\partial S_t} + \frac{1}{2} \sigma^2(t, S_t) S_t^2 \frac{\partial^2}{\partial S_t^2} - r
\]

This will be useful in the explanations that follows.

Now remember: a solution to such a PDE is uniquely determined by a set of boundary conditions for \( f \). In the case of an up-and-out call option the boundary conditions are as follows\(^{21}\):

\[
\begin{align*}
    f(T, S_T) &= (S_T - K)^+ \quad \text{for} \quad S_T < H \\
    f(t, 0) &= 0 \quad \text{for} \quad t \leq T \\
    f(t, H) &= 0 \quad \text{for} \quad t \leq T \\
\end{align*}
\]

Theorem 3.1. (The Superposition Principle) Let \( L \) and \( B \) be linear partial differential operators. If \( u_1, u_2, ..., u_k \) satisfy the linear partial differential equations \( L(u_i) = 0 \) for \( i = 1, ..., k \) and if \( c_1, ..., c_k \) are any constants then \( u = c_1 u_1 + c_2 u_2 + ... + c_k u_k \) satisfies \( L(u) = 0 \) and \( B(u) = 0 \). This notion of being able to add known solutions to obtain a new solution to a related linear partial differential equation is known as the superposition principle. (Lotter 2004)

Now suppose that one can evaluate the price of a call option, which satisfies the PDE (6). That is, one can evaluate the function \( C(t, T, S_t, K) \) at time \( t \) such that:

\[
\begin{align*}
    L(C(t, T, S_t, K)) &= 0 \quad \text{for} \quad t \leq T \text{ and } S_t \geq 0 \\
    C(T, T, S_t, K) &= (S_T - K)^+ \quad \text{for} \quad S_T > 0 \\
    \lim_{S_t \to -\infty} C(t, T, S_t, K) &= S_t \quad \text{for} \quad t \leq T \\
    C(t, T, 0, K) &= 0 \quad \text{for} \quad t \leq T \\
\end{align*}
\]

Given that \( C(t, T, S_t, K) \) can be evaluated, suppose further that this can be used to construct a portfolio of call options

\[
\pi(t, S_t) = \sum_{i=1}^{n} \alpha_i C_i(t, \tau_i, S_t, K_i)
\]

\(^{21}\)The last boundary condition should, more precisely, be \( f(t, S_t) = 0 \) for \( t \leq T \) and \( S_t \geq H \). However, the condition given is sufficient. This is because a security satisfying \( f(t, H) = 0 \) for \( t \leq T \) can be liquidated at the instant \( S_t = H \) so that it satisfies \( f(t, S_t) = 0 \) for \( t \leq T \) and \( S_t \geq H \).
such that:

\[
\begin{align*}
\pi(T, S_T) &= (S_T - K)^+ \quad \text{for} \quad S_T < H \\
\pi(t, 0) &= 0 \quad \text{for} \quad t \leq T \\
\pi(t_j, H) &= 0 \quad \text{for} \quad t_j (< \tau_j < T), \quad j = 1, 2, \ldots, n - 1
\end{align*}
\]  

(7)

Now since \( L_i(C_i(t, \tau_i, S_t, K_i)) = 0 \) for \( i = 1, \ldots, n \), from the superposition principle, it is true that \( L(\pi) = 0 \). Furthermore, by construction, the boundary conditions for \( \pi \) are as given in (7). Note that the first two conditions are exactly the same as those for the up-and-out call option. The last boundary condition, however, only agrees with the third boundary condition of the up-and-out call at a finite number of points \( t_j \) for \( j = 1, 2, \ldots, n - 1 \). Remember, as mentioned earlier, a solution to a PDE is uniquely determined by a set of boundary conditions. Therefore, as the number of points \( t_j \) at which the portfolio \( \pi \) agrees with the third boundary condition of an up-and-out barrier option, increases, so \( \pi(t, S_t) \) will tend to the value of the up-and-out call for all \( t \leq T \) and \( S_t \leq H \). In other words, matching the value along the boundaries ensures that the value within the boundaries is the same. Hence the portfolio, \( \pi \), can be taken as a replicating portfolio for the up-and-out call option.

This concludes the justification of the procedure described in the next subsection, for obtaining a replicating portfolio of puts and/or calls for vanilla barrier options.

### 3.3 Description of the Method

This method is best understood and explained with an example. The description given in this subsection is based on the one found in (Nalholm & Poulsen 2005) - though with a lot more of the finer details included. We begin by explaining the method in the case of an up-and-out call with barrier level \( H \), strike \( K \) and initial share price \( S_0 \). Consider Figure 9:

There are two mutually-exclusive and exhaustive scenarios that need to be considered when analysing the payoff of a barrier option:

1. The stock price never hits the barrier level over the life of the option (stock price path A). In this case, the payoff of the barrier option is the same as the payoff of a call option with strike \( K \).

2. The stock price hits the barrier level \( H \) at some time \( t \) before maturity (stock price path B). In this case, the value of the option when this happens is zero.

The aim of the DEK method is to construct a portfolio of vanilla European options which replicates the payoff of the barrier option under the above two scenarios.
3.3 Description of the Method

In other words, should the stock price never hit the barrier, the portfolio must provide a payoff equal to that of a call with strike $K$. However, if the stock price does hit the barrier, the value of the portfolio when this occurs must be zero. In the latter case, we would liquidate this portfolio at the point the barrier is hit. The reason being, should the stock price move to another level after hitting the barrier level, this portfolio will no longer have a value of zero, while the barrier option will. Hence by liquidating the portfolio we are ensuring that the barrier option’s payoff of zero under such a scenario is replicated.

By creating a portfolio (of vanilla European options) that provides the same pay-off as the barrier option under the above two scenarios, one is essentially ensuring that the boundary values of the replicating portfolio are the same as that of the barrier option. Hence, following from the arguments given in subsection 3.2, it is known that this will ensure that the value of the replicating portfolio will match that of the barrier option for all $t \leq T$ and $S_t \leq H$ (within the local volatility model being used).

Let $C(t, T, S_t, K)$ denote the value of a vanilla European call at time $t$, with initial share price $S_t$, maturity $T$ and strike $K$ - this value is the value of the option within the local volatility model being used to create a replicating portfolio. Now, consider Figure 10:
3.3 Description of the Method

Figure 10: Constructing a portfolio of vanilla options with (approximately) the same boundary conditions as an up-and-out barrier option.

We begin by targeting the barrier option’s payoff under the first scenario. Should the share price never hit the barrier, the payoff at maturity is equal to that of a non-barrier version of the barrier option. Therefore, to achieve the payoff under this scenario we simply need to be long a non-barrier version of the barrier option i.e. a vanilla European call option with strike $K$ and maturity $T$ (in our example). We have therefore achieved the barrier option’s payoff for all paths of type $A$ in Figure 10.

Now for scenario two. We begin by targeting the barrier options value should the stock price hit the barrier at time $t_3$ in Figure 10. Note that we do not begin at time $t_4$ - this point is discussed later. Currently, holding just the vanilla European call option (in order to replicate the payoff under scenario 1), the value of the portfolio at this time is $C(t_3, T, H, K)$. However, we can take a position, $\alpha_3$, in an option with maturity $\tau_3$ which will ensure that the value of the portfolio at $(H, t_3)$ is zero.

Two important points to note are:

1. The option used to achieve this will have to be a call with strike, $K_3$, greater than or equal to the barrier level $H$. Suppose this is not true. Then if scenario 1 occurs, a payoff may arise from this option at time $\tau_3$, which will have some non-zero value at time $T$. This will interfere with the already matched payoff of the portfolio under scenario 1.
2. Given that the previous point is satisfied, the maturity of this option, $\tau_3$, must be strictly greater than $t_3$. If it is equal to $t_3$, then at the point $(H, t_3)$ the option will always have a value of zero (provided condition 1 holds). This makes it impossible to achieve the target value of zero for the portfolio at the point $(H, t_3)$.

The equation that needs to be solved to determine $\alpha_3$ is then:

$$\alpha_3 C(t_3, \tau_3, H, K_3) + C(t_3, T, H, K) = 0 \quad (8)$$

**Practical point when selecting strike price $K_3$.**

*Note from (8) that*

$$\alpha_3 = -\frac{C(t_3, T, H, K)}{C(t_3, \tau_3, H, K_3)}$$

Since the price of a call option decreases as the strike price increases, one can see that the larger the chosen value for $K_3$, the larger will be the required position, $\alpha_3$, in order to zeroise the payoff at $(H, t_3)$ - i.e. $\alpha_3$ blows up. Hence, it would be best to choose $K_3$ (which must be $\geq H$, from point 1. above) to be smallest possible strike price. It is for this reason that the strike price of the options used to replicate along the barrier are usually, in theoretical work, equal to be the barrier level.

Next we target the barrier options value should the stock price hit the barrier at time $t_2$ in Figure 10. The current value of the replicating portfolio at this point - i.e. $(t_3, H)$ - is

$$\alpha_3 C(t_2, \tau_3, H, K_3) + C(t_2, T, H, K)$$

To achieve a value of zero at this point, we need to take a position, $\alpha_2$, in a call with strike greater than or equal to $H$ and maturity $\tau_2$ greater than $t_2$. The equation which needs to be solved to determine $\alpha_2$ is then:

$$\alpha_2 C(t_2, \tau_2, H, K_2) + \alpha_3 C(t_2, \tau_3, H, K_3) + C(t_2, T, H, K) = 0$$

Continuing in this way, the equations that need to be solved to determine $\alpha_1$ and $\alpha_0$ in order to target the barrier option’s value of zero at the points $(H, t_1)$ and $(H, t_0)$ are:

$$\alpha_1 C(t_1, \tau_1, H, K_1) + \alpha_2 C(t_1, \tau_2, H, K_2) + \alpha_3 C(t_1, \tau_3, H, K_3) + C(t_1, T, H, K) = 0$$

---

22This blow up can also be seen by running the code provided (discussed later), which implements this particular method for the CEV and Black-Scholes model. That is, MATLAB returns the output NaN when the strike prices chosen are “too large”.

23For the same reasons as before.

24That is, the positions required in call options with strikes $\geq H$ and which mature at $\tau_1$ and $\tau_0$, respectively.
3.3 Description of the Method

\[ \alpha_0 C(t_0, \tau_0, H, K_0) + \alpha_1 C(t_0, \tau_1, H, K_1) + \alpha_2 C(t_0, \tau_2, H, K_2) + \alpha_3 C(t_0, \tau_3, H, K_3) + C(t_0, T, H, K) = 0 \]

In matrix notation these equations can be written as:

\[
\begin{pmatrix}
-C(t_3, T, H, K) \\
-C(t_2, T, H, K) \\
-C(t_1, T, H, K) \\
-C(t_0, T, H, K)
\end{pmatrix}
= \begin{pmatrix}
0 & 0 & 0 & C(t_3, \tau_3, H, K_3) \\
0 & 0 & C(t_2, \tau_2, H, K_2) & C(t_2, \tau_3, H, K_3) \\
0 & C(t_1, \tau_1, H, K_1) & C(t_1, \tau_2, H, K_2) & C(t_1, \tau_3, H, K_3) \\
C(t_0, \tau_0, H, K_0) & C(t_0, \tau_1, H, K_1) & C(t_0, \tau_2, H, K_2) & C(t_0, \tau_3, H, K_3)
\end{pmatrix}
\begin{pmatrix}
\alpha_3 \\
\alpha_2 \\
\alpha_1 \\
\alpha_0
\end{pmatrix}
\]

This can easily be solved to obtain \(\alpha_0, \alpha_1, \alpha_2\) and \(\alpha_3\), provided that \(C(t, T, S_t, K)\) can be evaluated accurately and efficiently within the local volatility model being used to create a static replicating portfolio. This, therefore, is an additional requirement for a local volatility model, if the method of (Derman et al. 1995) is going to be used to replicate barrier options.

Given that \(\alpha_0, \alpha_1, \alpha_2\) and \(\alpha_3\) are obtainable (i.e. \(C(t, T, S_t, K)\) can be evaluated accurately and efficiently), we have created a portfolio of options that replicates the up-and-out option for all cases of scenario 1 and at the times \((H, t_0), (H, t_1), (H, t_2)\) and \((H, t_3)\) under scenario 2. This portfolio consists of the following positions:

- Long a European call with strike \(K\) and maturity \(T\) - i.e. the non-barrier version of the barrier option being replicated.
- \(\alpha_3\) of a call with maturity \(\tau_3\) and strike \(K_3\)
- \(\alpha_2\) of a call with maturity \(\tau_2\) and strike \(K_2\)
- \(\alpha_1\) of a call with maturity \(\tau_1\) and strike \(K_1\)
- \(\alpha_0\) of a call with maturity \(\tau_0\) and strike \(K_0\)

Note, compared to the Carr & Chou method, the replicating portfolio here consists of options with different maturities.

Recall, we began by zeroising the initial replicating portfolio’s value (i.e. after having achieved the payoff under scenario 1) at the point \((H, t_3)\) and not \((H, t_4)\). The reason for this is that it is (practically) impossible to zeroise the (initial) replicating portfolio’s value at this point using vanilla European calls and puts. This is because, at this point, the payoff of the barrier option being replicated is
3.3 Description of the Method

discontinuous and therefore would require an infinite number of number of vanilla
calls to be shorted in order to zeroise the initial payoff (i.e. the payoff of the call
option which is purchased to replicate the payoff of the barrier option under sce-
nario 1). However, if one is permitted to use European binary options, then the
payoff at the point \((H, t_4)\) can be zeroised by shorting a call option with strike \(H\)
and maturity \(T\) and shorting some amount, \(\alpha_4\), of a binary European call with
strike \(H\) and maturity \(T\). Note \(\alpha_4\) is solved by setting up an appropriate equa-
tion. One would then proceed using the same methodology as described above
to zeroise the resulting (i.e. after zeroising at the point \((H, t_4)\)) replicating port-
folio value at the points \(t_3, t_2, t_1\) and \(t_0\) - in this order, as the methodology requires.

By increasing the number of points along the barrier at which the payoff of the
portfolio of vanilla options matches the payoff of the barrier option, one can
replicate the barrier option’s payoff for a large number of possible stock price
paths of the type of scenario 2. In the limit, which would require an infinite
number of options, we can identically replicate the barrier option’s payoff under
all possible stock price paths. Obviously, from a practical point of view, one would
only match the portfolio value (under scenario 2) at a finite number of points
along the barrier. Furthermore, the choice of these points will be influenced by
the availability of options with the required maturities and strike prices.

Notes On The Procedure For Other Types
Of Vanilla Barrier Options

For up-and-out puts, the above formulation and comments are exactly the same
with the only difference being: instead of longing a call to achieve the barrier
option’s payoff under scenario 1 you would need to long a vanilla European put
(i.e. a non-barrier version of the up-and-out put).

For down-and-out puts/calls, the procedure is very similar. The following notes
highlight important differences and similarities:

1. The payoff under scenario 1 is easily obtained by simply taking a long
   position in the non-barrier version of the option.

2. To achieve the value of the barrier option along the barrier (i.e. scenario
   2), instead of using call options one needs to use put option. This, again,
   is to ensure that in the process of matching the portfolio’s payoff along the
   barrier, one does not interfere with the already matched payoff above the
   barrier.

3. The strike prices of these put options must be less than or equal to the
   (down-) barrier level \(H\). As before, if this is not true, we could have a
3.4 Implementation for the CEV & Black-Scholes model

situation where scenario 1 occurs and a payoff from a put position occurs, resulting in a non-zero value for this payoff at time $T$. This will then affect the already matched payoff under scenario 1.

4. The comments regarding the “blow up” for up-options, which occurs when the strike price of the call options used to replicate along the barrier are too large, also applies here. Except this time the blow up occurs when the strike price is too small\(^{25}\).

5. The maturity of an option, $\tau_i$ say, being used to achieve a certain value for the replicating portfolio along the barrier at a certain time, $t_i$ say, must be strictly greater than $t_i$. The reason for this is the same as before.

For “in” versions of the above barriers, one can obtain the replicating portfolio using in-out parity. Alternatively, the above procedure can be used keeping the following points in mind:

1. The replicating portfolio must result in a payoff of zero under scenario 1. Hence, one does not take any position in the non-barrier version of the barrier option.

2. Along the “in”-barrier, instead of aiming for a value of zero for the replicating portfolio, one will be aiming for the value of the non-barrier version of the barrier option being replicated.

3. The comments about using calls to replicate along an “up”-barrier and puts when replicating along a “down”-barrier, still holds.

3.4 Implementation for the CEV & Black-Scholes model

The DEK method has been implemented in MATLAB for two option pricing models: the constant elasticity of variance (CEV) model and the Black-Scholes model. In fact, the Black-Scholes model is a special case of the CEV model. However, incorporating this into the code is not trivial and one has to explicitly take account of the two models. The name of the function implementing the method of (Derman et al. 1995) for these two models is DEKCEV.m. The actual MATLAB code can be found in Appendix B.2. The inputs for this function, and the outputs, are discussed in subsection 3.4.2. Please note that the use of binary options, when creating static replicating portfolios, has not been incorporated into this code - though is can be done relatively easily.\(^{26}\) A brief discussion of the CEV model is now given.

\(^{25}\)Remember, the price of put options decrease as the strike price decreases.

\(^{26}\)The realisation of the usefulness of binary options occurred towards the end of the project. Hence, time constraints prevented the incorporation of their use into the code.
3.4 Implementation for the CEV & Black-Scholes model

3.4.1 The CEV model

The assumption made in the CEV model for the process of the underlying asset of the derivative is:

\[ dS_t = \mu S_t + \sigma S_t^\gamma dW_t \]  

(9)

where \( \gamma \) is a positive constant, \( \mu \) is the drift rate and \( \sigma \) is a volatility parameter. Note that the volatility of the instantaneous return on \( S_t \) is equal to \( \sigma S_t^{\gamma-1} \). This means that when \( \gamma > 1 \), the volatility of the stock returns increases as the stock price increases, and when \( 0 < \gamma < 1 \) the volatility decreases as stock prices increase. All other assumptions of the model are exactly the same as the standard Black-Scholes model assumptions. Note that when \( \gamma = 1 \) the process followed by the underlying is exactly GBM and hence the model reduces to the Black-Scholes model for option pricing.

These assumptions lead to the following option pricing formulae for put and call options, (Hull 2003). For \( 0 < \gamma < 1 \) we have

\[
C(t, T, S_0, K, r, \sigma) = S_0 e^{-q(T-t)} \left[ 1 - \chi^2(a+b+2, c) \right] - K e^{-r(T-t)} \chi^2(c, b, a) \\
P(t, T, S_0, K, r, \sigma) = K e^{-r(T-t)} \left[ 1 - \chi^2(c, b, a) \right] - S_0 e^{-q(T-t)} \chi^2(a+b+2, c)
\]

and for \( \gamma > 1 \) we have

\[
C(t, T, S_0, K, r, \sigma) = S_0 e^{-q(T-t)} \left[ 1 - \chi^2(c, -b, a) \right] - K e^{-r(T-t)} \chi^2(a, 2-b, c) \\
P(t, T, S_0, K, r, \sigma) = K e^{-r(T-t)} \left[ 1 - \chi^2(a, 2-b, c) \right] - S_0 e^{-q(T-t)} \chi^2(c, -b, a)
\]

where

\[
\nu = \frac{\sigma^2}{2(r-q)(\gamma-1)[e^{2(r-q)(\gamma-1)(T-t)}]} \\
a = \frac{K e^{-r(T-t)}(\gamma-1)^2(1-\gamma)}{(1-\gamma)^2 \nu} \\
b = \frac{1}{1-\gamma} \\
c = \frac{S_0^2(1-\gamma)}{(1-\gamma)^2 \nu}
\]

and \( \chi^2(a, b, c) \) is the CDF of the non-central chi-squared distribution, evaluated at \( a \), with \( b \) degrees of freedom and non-centrality parameter \( c \). These formulae can now be used, as described in subsection 3.3, to obtain a replicating portfolio (of puts and/or calls) for vanilla barrier options.

The value of the replicating portfolio should approach the exact value of the barrier option, under the CEV or Black-Scholes model, as the number of points at which we match the barrier option’s value, along the barrier, increases.

In order to check this one requires the exact value of a barrier option under the CEV model and the Black-Scholes model. Remember that exact formulae for vanilla barrier options exist under the Black-Scholes model, see (Hull 2003). In the case of the CEV model this is not true. However, the price of a barrier option...
3.4 Implementation for the CEV & Black-Scholes model

in this model can be obtained by numerically solving the no-arbitrage PDE (with
the relevant boundary conditions) for the value of a derivative, which in the case
of the CEV model is:

\[
\frac{\partial f}{\partial t} + r S_t \frac{\partial f}{\partial S_t} + \frac{1}{2} \sigma^2 S_t^\gamma \frac{\partial^2 f}{\partial S_t^2} = rf
\]

A function implementing the Backward-Time-Central-Space finite difference scheme
to this PDE, to obtain the price of barrier options, has been written. The name
of the function is CEVFDS.m\(^{27}\) and the code for it can be found in Appendix B.4.
A discussion of how the PDE is solved numerically, together with a discussion of
the inputs of the implemented function, can be found in Appendix A.2

3.4.2 Inputs & outputs for the MATLAB function DEKCEV.m

The function

\[
\text{[output value]}=\text{DEKCEV}(\text{int},\text{up},\text{out},\text{So},\text{strike},\text{B},\text{term},\text{sigma},\text{r},\text{q},\text{Y},\text{time},\text{tau})
\]

implemented in MATLAB calculates the replicating portfolio for vanilla barrier
options using the methodology described in subsection 3.3. We now describe the
inputs for this function together with the restrictions on their value:

1. int - This indicates whether the vanilla barrier option is a vanilla call
   barrier option, \(\text{int}=1\), or a vanilla put barrier option, \(\text{int}=0\).
2. up - This indicates whether the vanilla barrier option is an “up” option,
   \(\text{up}=1\), or a “down” option, \(\text{up}=0\).
3. out - This indicates whether the option is a knock-out option, \(\text{out}=1\), or a
   knock-in option, \(\text{out}=0\).
4. So, strike and B- These are the initial share price, the strike price and
   the barrier level of the vanilla barrier option, respectively.
5. term - This is the term of the barrier option.
6. sigma - This is the volatility parameter \(\sigma\) for the CEV model in equation
   (9), subsection 3.4.1. Note that if the volatility of the return in the market
   is currently \(\nu\), then \(\sigma\) in (9) must be chosen such that \(\sigma S_t^\gamma - 1 = \nu\).
7. r and q - These are the constant NACC, risk free and dividend yields,
   respectively.

\(^{27}\) i.e. CEV Finite Difference Scheme.
8. \( Y \) - This is the parameter \( \gamma \) for the CEV model in equation (9), subsection 3.4.1. When \( Y=1 \), the function determines the replicating portfolio in the Black-Scholes model.

9. \textbf{time} - This is the vector indicating the times at which one requires the payoff of the replicating portfolio to match the payoff of the barrier option along the barrier. In other words, it would be a vector containing \( t_0, t_1, t_2 \) and \( t_3 \) in Figure 10, subsection 3.3.

10. \textbf{tau} - This would be the vector containing the maturities of the options that are required to be used to replicate the barrier option’s value along the barrier. In other words, it would be a vector containing \( \tau_0, \tau_1, \tau_2 \) and \( \tau_3 \) in Figure 10, subsection 3.3.

11. \( K \) - This is the vector containing the strike prices of the options which will be used to replicate the payoff along the barrier. i.e. it would be a vector containing the strike prices \( K_0, K_1, K_2 \) and \( K_3 \) in Figure 10, subsection 3.3. Remember, they are all usually equal to the barrier level so as to prevent the “blow-up” discussed earlier.

Further points to note are:

- \textbf{time} and \textbf{tau} must be \textit{row} vectors.

- The entries in both these vectors should be increasing from left to right. In other words, using the time and maturity points in Figure 10, we would have \( \text{time}=[t_0, t_1, t_2, t_3] \) and \( \text{tau}=[\tau_0, \tau_1, \tau_2, \tau_3] \).

- Each entry in the \textbf{time} vector must be less than the corresponding entry in the \textbf{tau} vector. The reason for this was explained in subsection 3.3.

- The entries in the vector \( K \) should be the strike prices corresponding to the maturity dates in the vector \textbf{tau}. i.e. \( K=[K_0, K_1, K_2, K_3] \)

The function produces two outputs, \textbf{output} and \textbf{value}. The output, \textbf{output}, is a matrix with the first column being the positions that need to be taken in various vanilla options\textsuperscript{28}. The second column contains the corresponding strike prices of the options in which the positions (in the first column) must be taken. The third column contains the corresponding maturity of these options and the forth column indicates whether the position should be taken in a put option or a call option. \( 1 \) indicates that the position must be taken in a call option and \(-1\) indicates that it must be taken in a put option. The last column gives the

\textsuperscript{28}The second and subsequent entries are exactly equal to the \( \alpha_i \)’s from subsection 3.3
3.5 Results on convergence from running the code

price of the options under the CEV or Black-Scholes model - depending on which model was used to obtain the replicating portfolio (i.e. whether \( Y = 1 \) or not). The output, value, is the value of this replicating portfolio of vanilla options calculated using the CEV (or Black-Scholes - again, depending on the value for \( Y \)) model’s pricing formulae for puts and calls.

3.5 Results on convergence from running the code

This subsection briefly illustrates, with the use of diagrams, the convergence of the replicating portfolio’s value/price and behaviour\(^{29} \) to that of the barrier option being replicated - the replicating portfolio, obviously, obtained using the methodology of (Derman et al. 1995).

First, the convergence of the value of the replicating portfolio is illustrated. Figures 11 and 12 plot the value of the replicating portfolio against the number of points, along the barrier, at which the payoff of the replicating portfolio and the barrier option being replicated are matched. In the first figure, the replicating portfolio was obtained using the DEK methodology for the Black-Scholes model - hence the value of the replicating portfolio is also determined in this model. In the second figure, the replicating portfolio was obtained using the DEK methodology for the CEV model - again, the value of the replicating portfolio is therefore also determined in this model. The barrier option being replicated is an up-and-out call and the parameters used are as follows: \( S_0 = 50, K = 45, H = 60, T = 2, r = 0.1, q = 0 \) and \( \sigma = 0.25 \) for the Black-Scholes model and \( \sigma = 1.76777^\text{30} \) for the CEV model. For the CEV model \( \gamma = 0.5 \).

From these plots it can be seen that convergence within both models does occur - as expected. Further, for both models, after “matching” at 70 points along the barrier, the value of the replicating portfolio is approximately 4% of the exact value of the barrier option.

Next the behaviour of the replicating portfolio, in the case of the Black Scholes model, is discussed. This is not done for the CEV model as an exact solution for the price of an up-and-out barrier option (or any other barrier option) does not exist and therefore constructing surfaces is not feasible - the finite difference scheme would have to be run numerous times; this is time consuming. However, one can expect the results to be similar to that of the Black-Scholes model; after all, the Black-Scholes model is a special case of the CEV model.

\(^{29}\)Within the local vol model being used

\(^{30}\)This ensures that the initial volatility in the model is 0.25. i.e. \( \sigma S_0^{-1} = 0.25 \)
3.5 Results on convergence from running the code

Figure 11: Convergence of the value of the replicating portfolio of vanilla options (obtained under the Black-Scholes model), to the Exact value of an Up-and-out call, as the number of matched points along the barrier increases. The parameters used were $K = 45$, $H = 60$, $S_0 = 50$, $T = 2$, $r = 0.1$, $q = 0$ and $\sigma = 0.25$.

Figure 12: Convergence of the value of the replicating portfolio of vanilla options (obtained under the CEV model), to the “Exact” value of an Up-and-out call, as the number of matched points along the barrier increases. The parameters used were $K = 45$, $H = 60$, $S_0 = 50$, $T = 2$, $r = 0.1$, $q = 0$, $\sigma = 1.76777$ and $\gamma = \frac{1}{2}$.

Figure 14 contains surfaces of the value of replicating portfolios together with the corresponding absolute error surfaces (i.e. the difference between the exact value of the barrier option and the value of the replicating portfolio - given as a % of the exact value.). These surfaces are in the case where 21 options are used to create a replicating portfolio (the first row of surfaces in Figure 14), the case where 51 options are used (the second row of surfaces in Figure 14) and finally the
3.5 Results on convergence from running the code

case where 101 options are used (the third row of surfaces in Figure 14). The option being replicated for this illustration is an up-and-out call and the parameters being used are $S_0 = 35$, $K = 30$, $H = 60$, $T = 2$, $\sigma = 0.25$, $r = 0.1$ and $q = 0$.

Note that the option and the parameters being used, are exactly the same as those used for the same illustration in the case of the Carr and Chou method - i.e. to determine Figures 7 and 8. This means that comparisons can be made between Figures 8 and 14. For convenience, the surface of the exact value of the up-and-out barrier option in question is repeated in Figure 13.

Figure 13: Surface of the exact value of the barrier option at various points in time and “space” (i.e. stock price).

---

In each case, one of the option is used to replicated the payoff of the up-and-out barrier option for all cases of scenario 1. The remainder aim to achieve the payoff of the barrier option along the barrier i.e. they do so at a finite number of points along the barrier.
3.5 Results on convergence from running the code

**Figure 14:** Surfaces of the value of replicating portfolio, together with the corresponding surface of the absolute error between the replicating portfolio's value and the exact value of the barrier option. The surfaces in the first row are for a replicating portfolio consisting of 21 options and the second and third rows give the surfaces when 51 and 101 options are used, respectively. The replicating portfolio’s are obtained using the DEK method.
3.5 Results on convergence from running the code

From Figures 8 and 14 the following observations can be made:

- As in the case of the Carr and Chou method, convergence in *shape* and *level* of the surface of the value of the replicating portfolio, is evident on comparing the surfaces in the first column of Figure 14 and the surface in Figure 13. As before, the axes and colour bars for all surfaces of the value of the replicating portfolio (including the exact value in Figure 13) are on the same scale. This allows comparisons to be made.

- The second column of surfaces is, again, more revealing. These surfaces illustrate the percentage discrepancy between the exact surface (i.e. Figure 13) and that of the replicating portfolio. In the case of the DEK method, it can be seen that when only 21 options are used to create a replicating portfolio, the discrepancy over most of the region is below 7%. This is in stark contrast to the discrepancy surface when 21 options are used for the Carr and Chou method - in this case, the discrepancy over a significant part of the region is well above the 10% level. More noticeably, though, is that for the DEK method, the “blow-up” in the discrepancy as the stock price approaches the barrier, occurs only when the option is relatively close to expiry. In the case of the Carr and Chou method this “blow up” occurs for all times as the stock price approaches the barrier; even more so as maturity is approached. This may have something to do with the fact that the maturities of the vanilla options making up the replicating portfolio for the Carr and Chou method are all the same; in contrast, for the DEK method, the vanilla options making up the replicating portfolio all have different maturities.

- Lastly, note how the level of the values for the discrepancy is greatly reduced as the number of options used to replicate are increased. That is, when 21 options are used to replicate, the discrepancy near the barrier is around the 6-7% range - excluding the points close maturity. Elsewhere, the discrepancies are far lower. However, when 51 options are used to create a replicating portfolio, the discrepancy near the barrier plunges to around the 2.5-3% level. And when a 101 options are used, this discrepancy becomes negligible. In this case, even when the option is close to maturity and the stock price near the barrier, the maximum discrepancy of 6% is acceptable. Generally speaking, the DEK method seems to converge far quicker that the Carr and Chou method. This can be seen by comparing the *level* of the discrepancy surfaces, for a *given number of options used to create a replicating portfolio*, for the respective methods. E.g. compare the
3.5 Results on convergence from running the code

discrepancy graph for the DEK method to the discrepancy graph for the Carr and Chou method when 21 options are used to create the replicating portfolios for both methods.
4 Conclusion

Statically replicating barrier options provides an alternative to dynamically replicating such options. It overcomes (or at least reduces the extent of) the difficulties associated with the latter. Some of these include: the need to trade regularly in the underlying, the resulting impact of transaction costs incurred from doing this and the added difficulty of transaction costs when the gamma of the underlying in large - as is the case for barrier options. Another difficulty that is overcome is the mismatch between the vega of the replicating portfolio (equal to zero) and that of the barrier option being replicated (which is, of course, non-zero).

However, with this new technique comes new problems - namely, the need for a large range of traded vanilla options, model dependence of the static replicating portfolio\textsuperscript{32} and the need for liquidity in the vanilla option’s market.

A second application of the static replication of barrier options is for pricing such option - one can use the vol surface of vanilla options trading in the market to price the vanilla options making up the static replicating portfolio.

Two methods of creating a static replicating portfolio are covered in this project. The first is the method by (Carr & Chou 1997\textit{a}) and the second is by (Derman et al. 1995).

The first method of obtaining a replicating portfolio is set in the Black-Scholes world. It is shown that a vanilla barrier option can be replicated with a European security, which has a payoff function chosen specifically for the barrier option being replicated - known as the \textit{adjusted payoff function}. Therefore, being able to replicate the European security is equivalent to being able to replicate the barrier option. A result is proved which shows that any European security can be replicated with vanilla puts and calls, bonds and forwards; hence, the European security and, therefore the barrier option, can be replicated. It is also explained how one would actually go about obtaining a replicating portfolio in practice - this is not obvious from the theoretical results. Finally, MATLAB code written for this method is discussed, followed by a brief discussion on some convergence results (obtained using the MATLAB code).

The second method discussed, allows one to get a static replicating portfolio for a barrier option within any local volatility model (which includes the Black-Scholes model). This is because under such models the problem of pricing derivatives

\textsuperscript{32}Though, this is true for dynamic replication as well.
can be reduced to solving a single PDE with specific boundary conditions. The replicating portfolio is then obtained by aiming to create a portfolio of vanilla options such that the portfolio satisfies the same boundary conditions as the barrier option being replicated. The justification of why such a portfolio replicates the barrier option (within the local volatility model being used to create a static replicating portfolio) is also given. This is done with the aid of the superposition principle. A requirement though, in order to practically implement this method, is that vanilla options need to be calculable both efficiently and accurately for the particular local volatility model being used to create a static replicating portfolio. This method is implemented in MATLAB for both the CEV model and the Black-Scholes model. The inputs and outputs for the MATLAB function is discussed and again some convergence results (from running this function) are provided.

In terms of the type of replicating portfolios produced under the two methods, the first method results in a replicating portfolio where the vanilla options have varying strikes but the same maturities. The opposite is true of the second method. It results in a portfolio where the vanilla options all have the same strikes but varying maturities.

In both cases, exact replication is possible only in theory. This is because exact replication requires an infinite number of positions to be taken in vanilla options. However, reasonable replicating portfolio can be achieved by taking positions in a finite number of vanilla options. Moreover, convergence of the value and behaviour of the replicating portfolio to the that of the barrier option (within the relevant model), occurs relatively quickly as the number of vanilla options used, increases. In some sense, this problem is equivalent to the impracticability of trading in the underlying continuously, when dynamically replicating options.

\footnote{This is necessary only to prevent a “blow-up” in the value of the positions needed to set up the static replicating portfolio.}
Appendices

A No-Arbitrage PDE for Local Volatility Models

A.1 Derivation of the PDE for a derivative’s value

This section was written with the aid of work covered in (Šešic & Taylor 2005).
Suppose we have a process for the underlying:

\[ dS_t = (\mu(t, S_t)S_t - qS_t)dt + \sigma(t, S_t)S_tdW_t \]  

(10)

where \( q \) is the constant continuously payable dividend yield. Using Itô’s lemma we get the process for a derivative \( f(t, S_t) \) i.e. a security whose value depends on that of another. Doing this we get:

\[ df_t = \left( \frac{\partial f}{\partial t} + (\mu(t, S_t)S_t - qS_t) \frac{\partial f}{S_t} + \frac{1}{2} \sigma^2 S_t^2 \frac{\partial^2 f}{S_t^2} \right) dt + \sigma(t, S_t)S_t \frac{\partial f}{S_t} dW_t \]  

(11)

Now if one longs the derivative and shorts \( \Psi_t \) of the underlying at time \( t \), the value of the portfolio at time \( t \) is \( V_t = f(t, S_t) - \Psi_t S_t \). Applying Itô’s lemma to this we get:

\[ dV_t = df_t - \Psi_t (dS_t + qS_t dt) - S_t d\Psi_t - d\Psi_t dS_t \]

If we require the portfolio to be self-financing it means \( S_t d\Psi_t + d\Psi_t dS_t = 0 \), and hence using (10) and (11) we get:

\[ dV_t = \left( \frac{\partial f}{\partial t} + \frac{1}{2} \sigma^2(t, S_t)S_t^2 \frac{\partial^2 f}{S_t^2} - qS_t \frac{\partial f}{S_t} \right) dt + \sigma(t, S_t)S_t \frac{\partial f}{S_t} dW_t \]

If \( \Psi_t = \frac{\partial f}{\partial S_t} \), the delta of the option, then the stochastic term of this equation is eliminated and the equation reduces to:

\[ dV_t = \left( \frac{\partial f}{\partial t} + \frac{1}{2} \sigma^2(t, S_t)S_t^2 \frac{\partial^2 f}{S_t^2} - qS_t \frac{\partial f}{S_t} \right) dt \]

Since the stochastic term is eliminated, that is the uncertainty of the portfolio’s value in a small interval \( dt \), the return on this portfolio over this interval must be exactly the risk free rate. This means:

\[ \left( \frac{\partial f}{\partial t} + \frac{1}{2} \sigma^2(t, S_t)S_t^2 \frac{\partial^2 f}{S_t^2} - qS_t \frac{\partial f}{S_t} \right) dt = (r - \frac{\partial f}{\partial S_t} S_t) dt \]
Simplifying this we get the no arbitrage PDE for the derivatives value:

\[ \frac{\partial f}{\partial t} + (r - q)S_t \frac{\partial f}{\partial S_t} + \frac{1}{2} \sigma^2(t, S_t)S_t^2 \frac{\partial^2 f}{\partial S_t^2} = rf \]  \hspace{1cm} (12)

**A.2 Solving the PDE for the CEV model using a Backward-Time Central-Space Finite Difference Scheme**

In the case of the CEV model where \( \sigma(t, S_t) = \sigma S_t^{\gamma-1} \) equation (12) becomes:

\[ \frac{\partial f}{\partial t} + (r - q)S_t \frac{\partial f}{\partial S_t} + \frac{1}{2} \sigma^2 S_t^{2\gamma} \frac{\partial^2 f}{\partial S_t^2} = rf \]

This remainder of this section is adapted from worked covered in (Lotter 2004). We begin by transforming the PDE into a backward time PDE.

Let \( u(\tau, S_t) = f(t, S_t) \) where \( \tau = T - t \). Then

\[ \frac{\partial f}{\partial t} = -\frac{\partial u}{\partial \tau}, \quad \frac{\partial f}{\partial S_t} = \frac{\partial u}{\partial S_t}, \quad \frac{\partial^2 f}{\partial S_t^2} = \frac{\partial^2 u}{\partial S_t^2} \]

hence

\[ -\frac{\partial u}{\partial \tau} + (r - q)S_t \frac{\partial u}{\partial S_t} + \frac{1}{2} \sigma^2 S_t^{2\gamma} \frac{\partial^2 u}{\partial S_t^2} = ru \]  \hspace{1cm} (13)

Suppose we want to solve this PDE on a region as indicated in Figure 15. Note we are given data along the the boundaries at \( S_{max}, S_{min} \) and along the boundary at \( \tau = 0 \).

---

**Figure 15:** Region over which reversed-time no-arbitrage PDE is to be solved.
We begin by subdividing the space axis into \( N_x \) pieces and the time axis into \( N_t \) pieces. That is, let

\[
\Delta \tau = \frac{t}{N_t} \quad \text{and} \quad \Delta S_t = \frac{S_{\max} - S_{\min}}{N_x}
\]

We now approximate the time derivative using a backward time approximation and the space derivative using a central time approximation. That is,

\[
\frac{\partial u}{\partial \tau}(\tau, S_t) \approx \frac{u(\tau - \Delta \tau, S_t) - u(\tau, S_t)}{\Delta \tau}
\]

\[
\frac{\partial u}{\partial S_t}(\tau, S_t) \approx \frac{u(\tau, S_t - \Delta S_t) - u(\tau, S_t + \Delta S_t)}{2\Delta S_t}
\]

If \( \tau = mk \) for some \( m = 1, ..., N_t \) and \( S_t = nh \) for some \( n = 1, ..., N_x \) then this can be written as,

\[
\frac{\partial u}{\partial \tau}(mk, nh) \approx \frac{u(mk - k, nh) - u(mk, nh)}{k}
\]

\[
\frac{\partial u}{\partial S_t}(mk, nh) \approx \frac{u(mk, nh - h) - u(mk, nh + h)}{2h}
\]

If we denote \( u(mk, nh) \) as \( u^m_n \) we can write this as

\[
\frac{\partial u}{\partial \tau}(mk, nh) \approx \frac{u^{m-1}_n - u^m_n}{k}
\]

\[
\frac{\partial u}{\partial S_t}(mk, nh) \approx \frac{u^m_{n-1} - u^m_{n+1}}{2h}
\]

In the same way we can write \( \frac{\partial^2 u}{\partial S_t^2}(mk, nh) \approx \frac{u^{m+1}_{n+1} - 2u^m_n + u^{m-1}_{n-1}}{k^2} \). Substituting these into (13) and simplifying we get,

\[
-\frac{u^{m-1}_n - u^m_n}{k} + \frac{u^m_{n-1} - u^m_{n+1}}{h} (r - q)(S_{\min} + nh) + \frac{1}{2} \sigma^2 (S_{\min} + nh)^{2\gamma} \frac{u^{m+1}_{n+1} - 2u^m_n + u^{m-1}_{n-1}}{h^2} = ru^m_n
\]

\[
\Rightarrow -\frac{u^m_n - u^{m+1}_n}{k} + \frac{u^m_{n+1} - u^m_{n-1}}{h} (r - q)(S_{\min} + nh) + \frac{1}{2} \sigma^2 (S_{\min} + nh)^{2\gamma} \frac{u^{m+1}_{n+1} - 2u^m_n + u^{m+1}_{n-1}}{h^2} = ru^{m+1}_n
\]

\[
\Rightarrow u^m_n = u^{m+1}_{n-1} \left[ (r - q)k(S_{\min} + nh) + \frac{1}{2h} - \frac{\sigma^2 k}{2h^2} (S_{\min} + nh)^{2\gamma} \right] + \frac{1}{2} \sigma^2 (S_{\min} + nh)^{2\gamma} \frac{u^{m+1}_{n+1} - 2u^m_n + u^{m+1}_{n-1}}{h^2}
\]

\[
\Rightarrow u^{m+1}_n = u^m_{n-1} \left[ 1 + rk + \frac{\sigma^2 k}{h^2} (S_{\min} + nh)^{2\gamma} \right] - \frac{1}{2} \sigma^2 (S_{\min} + nh)^{2\gamma} \frac{u^{m+1}_{n+1} - 2u^m_n + u^{m+1}_{n-1}}{h^2} + (r - q)k(S_{\min} + nh) \frac{1}{2h} + \frac{\sigma^2 k}{2h^2} (S_{\min} + nh)^{2\gamma}
\]
for \( m = 1, \ldots, N_t \).

For a fixed \( m \), this equation can be written for each of the share price values \( nh, n = 1, \ldots, N_x - 1 \). These equations can in turn be written in matrix form as follows. Let

\[
U^m = \begin{pmatrix}
  u^m_1 \\
  u^m_2 \\
  \vdots \\
  u^m_{N_x-2} \\
  u^m_{N_x-1}
\end{pmatrix}, \quad b^m = \begin{pmatrix}
  u^m_0 \\
  (r-q)k(S_{\text{min}} + h) \frac{1}{2h} - \frac{\sigma^2}{2h^2}(S_{\text{min}} + h)^{2\gamma} \\
  \vdots \\
  0 \\
  (r-q)k(S_{\text{min}} + (N_x-1)h) \frac{1}{2h} - \frac{\sigma^2}{2h^2}(S_{\text{min}} + (N_x-1)h)^{2\gamma}
\end{pmatrix}
\]

\[
D_1 = S_{\text{min}} + \text{diag}(1, 2, \ldots, N_x-1)h \\
D_2 = [S_{\text{min}} + \text{diag}(1, \ldots, N_x-1)h]^{2\gamma}
\]

\[
T_1 = \begin{pmatrix}
  0 & 1 & 0 & 0 & 0 & \ldots & 0 \\
  -1 & 0 & 1 & 0 & 0 & \ldots & 0 \\
  0 & -1 & 0 & 1 & 0 & \ldots & 0 \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & \ldots & 0 & 0 & -1 & 0 & 1 \\
  0 & \ldots & 0 & 0 & -1 & 0 & \ldots
\end{pmatrix}, \quad T_2 = \begin{pmatrix}
  -2 & 1 & 0 & 0 & 0 & \ldots & 0 \\
  1 & -2 & 1 & 0 & 0 & \ldots & 0 \\
  0 & 1 & -2 & 1 & 0 & \ldots & 0 \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & \ldots & 0 & 0 & 1 & -2 & 1 \\
  0 & \ldots & 0 & 0 & 0 & 1 & -2
\end{pmatrix}
\]

Then

\[
U^{m+1}F + b^{m+1} = U^m \quad \text{for} \quad m = 1, \ldots, N_t
\]

where

\[
F = I_{N_x-1}(1 + rk) - \frac{1}{2} \sigma^2 \mu D_2 T_2 - (r-q)k \frac{1}{2h} D_1 T_1
\]

Since the boundary values of the function \( u(\tau, S_t) \) are known, we can solve iteratively for \( U^{N_t} \). As an example, let’s consider the boundary data for a down-and-out call with strike \( K \) and barrier level \( B \). In this case we have,

\[
U^0 = \begin{pmatrix}
  u^0_1 \\
  u^0_2 \\
  \vdots \\
  u^0_{N_x-2} \\
  u^0_{N_x-1}
\end{pmatrix} = \begin{pmatrix}
  (h-K)^+ \\
  (2h-K)^+ \\
  \vdots \\
  ((N_x-2)h-K)^+ \\
  ((N_x-1)h-K)^+
\end{pmatrix}
\]
A.2 Solving the PDE for the CEV model using a Backward-Time Central-Space Finite Difference Scheme

\[ \begin{align*} 
U^m_{S_{\min}} &= \begin{pmatrix} u^1_{S_{\min}} \\ u^2_{S_{\min}} \\ \vdots \\ u^{N_x-2}_{S_{\min}} \\ u^1_{S_{\min}} \\ \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \end{pmatrix} \\
U^m_{S_{\max}} &= \begin{pmatrix} u^1_{S_{\max}} \\ u^2_{S_{\max}} \\ \vdots \\ u^{N_x-2}_{S_{\max}} \\ u^1_{S_{\max}} \\ \end{pmatrix} = \begin{pmatrix} S_{\max} - Ke^{-r_k} \\ S_{\max} - Ke^{-r_{2k}} \\ \vdots \\ S_{\max} - Ke^{-r(N_t-1)k} \\ S_{\max} - Ke^{-rN_tk} \end{pmatrix} 
\end{align*} \]

Note, \( S_{\min} \) is actually the barrier level. Using this information, we then recursively use

\[ U^{m+1} = F^{-1}(U^m - b^{m+1}) \]

to obtain

\[ U^{N_t} = \begin{pmatrix} u^1_{N_t} \\ u^2_{N_t} \\ \vdots \\ u^{N_t}_{N_x-1} \end{pmatrix} = \begin{pmatrix} u(T, h) \\ u(T, 2h) \\ \vdots \\ u(T, (N_x - 1)h) \end{pmatrix} \]

Finally, remember that \( f(0, S_0) = u(T, S_0) \). So, in order to obtain the price of the derivative, \( f(0, S_0) \), we must interpolate between the relevant values in this vector.

The MATLAB code for the function implementing this finite difference scheme, called CEVFDS.m, can be found in Appendix B.4. A discussion of its inputs is now given.

A.2.1 Inputs & outputs for the MATLAB function CEVFDS.m

The function is accessed using the syntax:

\[ \text{price} = \text{CEVFDS}(N_x, N_t, Y, S, K, r, q, \sigma, T, S_{\max}, S_{\min}, \text{type}). \]

We now describe what the inputs represent:

1. \( N_x \) - This is the number of space (i.e. share price) steps you require the finite difference scheme to use.
2. \( N_t \) - This is the number of time steps you require the finite difference scheme to use.
3. \( Y \) - This is the parameter \( \gamma \) in the CEV model.
4. \( S, K \) and \( \sigma \) - These are the initial share price, the strike price of the option, and the parameter \( \sigma \) in the CEV model, respectively.
A.2 Solving the PDE for the CEV model using a Backward-Time Central-Space Finite Difference Scheme

5. \( r \) and \( q \) - These are the constant NACC, risk free rate and dividend yield, respectively.

6. \( T \) - This is the term of the option.

7. \( S_{\text{max}} \) - This is the upper cut off point in the finite difference scheme for the share price. Remember that the finite difference scheme method can only be applied if one has boundary data along a bounded region in time and space. This parameter is equal to the barrier level if the option is an “up” option and a “large” value otherwise\(^{34}\).

8. \( S_{\text{min}} \) - This is the lower cut of point in the finite difference scheme for the share price. This parameter is equal to the barrier level if the option is a “down” option and zero otherwise.

9. \( \text{type} \) - This is a string indicating the type of option for which a price is required. The possibilities for this string are:
   
   - ‘call’ - for a call option
   - ‘put’ - for a put option
   - ‘doc’ - for a down-and-out call option
   - ‘dop’ - for a down-and-out put option
   - ‘uoc’ - for an up-and-out call option
   - ‘uop’ - for an up-and-out put option

Note that the parameter values for \( S, K \) and the barrier level must be consistent with the type of option for which a price under the CEV model is required. For example, one cannot have an up-and-out option with \( S_{\text{0}} > B \).

Note that this function does not calculate knock-in barrier option prices (under the CEV model). However, prices for such options can be obtained using in-out parity. Finally, letting \( Y=1 \), one can obtain the price of the above mentioned options under the Black-Scholes model - though closed form solutions do exist for this model, see (Hull 2003).

\(^{34}\)Two or three times the current share price is adequate.
B.1 CarrChou.m

function [output value]=CarrChou(int,down,out,So,strike,...
B,K,x,sigma,r,q,term)

K=fliplr(K);
x=fliplr(x);
p=1-2*(r-q)/sigma^2;
n=length(K);

P=zeros(n,n);
for i=1:n;
P(:,i)=max(down*(K(1,i)-x'),0);
end;

gvec=(-1*(x'./B).^p).*max(int*(B^2./x'-strike),0);
gvec=gvec-max(int*(x'-strike),0);

alphaout=P
gvec;

[call put]=blsprice(So,K,r,term,sigma,q);
[C1 P1]=blsprice(So,strike,r,term,sigma,q);
priceout=(put*(0.5*down+0.5)+call*(-0.5*down+0.5))*alphaout+...
    C1*(0.5*int+0.5)+P1*(-0.5*int+0.5);

alpha=[1;alphaout]*out+[0;-alphaout]*(1-out);
TypeOfOption=[int;ones(length(alphaout),1)]*(-down);
pricevec=[C1*(0.5*int+0.5)+P1*(-0.5*int+0.5);...
    (put*(0.5*down+0.5)+call*(-0.5*down+0.5))]
output=[alpha [strike K' TypeOfOption pricevec];
value=priceout*out+...
    (C1*(0.5*int+0.5)+P1*(-0.5*int+0.5)-priceout)*(1-out);

disp(' Position K Type Price');
disp(output)
function [output value]=DEKCEV(int,up,out,So,strike,B,term,...
sigma,r,q,Y,time,tau,K)

n=length(time);
P=zeros(n,n);
M=term; [Call,Put] = CEVprice(B,strike,r,term,sigma,time,Y,q);
NBV=flipud(((Call.*int+Put.*(1-int))')');

for i=1:n;
    [Call,Put] = CEVprice(B,K(1,n+1-i),r,tau(1,n-i+1),sigma,...
time(1:1,1:n-i+1),Y,q);
P(i:i,1:n-i+1)=Call.*up+Put.*(1-up);
end;
P=flipud(fliplr(P'));
alphaout=P\(-NBV);

M=term; [Call,Put] = CEVprice(B,strike,r,term,sigma,time,Y,q);
[Cl,P1] = CEVprice(So,strike,r,term,sigma,0,Y,q);
pricevec=[Cl*int+P1*(1-int),Call*up+Put*(1-up)];
valueout=dot(pricevec',[1;alphaout]);

% Output 1: Value of replicating portfolio.
% ----------------------------------------------------------
value=(Cl*int+P1*(1-int)-valueout)*(1-out)+valueout*out;

% Output 2: Positions that need to be taken.
% ----------------------------------------------------------
alpha=[(1;zeros(n,1)\[-1;alphaout])*(1-out)+[1;alphaout]*out;
Type=[2*int-1; (up*2-1)*ones(n,1)];
output=[alpha [strike K]’ [term tau]’ Type pricevec’];
disp(’ Position Strike Maturity Type Price’);
disp(output);

Note that the function DEKCEV makes use of the function CEVprice. This function calculates the price of a call and a put option under the price dynamics of the CEV model. Furthermore, it calculates the price of a call and a put option under the Black-Scholes model when the parameter Y is equal to 1. The code for this
function is presented next together with a description of its inputs.
function [Call Put] = CEVprice(So,strike,r,T,sigma,t,Y,q)

v=sigma^2/(2*(r-q)*(Y-1)).*(exp(2*(r-q)*(Y-1).*((T-t))-1));
b=1/(1-Y);

warning('off','MATLAB:divideByZero')

% The CEV vanilla option pricing formulae in (Hull 2003),
% don’t work when Y=0. I.e. we end up dividing by zero in
% the formulae. The "if" statement below, is taking into
% account the limiting values of the option pricing formulae
% in this case. The same problem occurs when T=0.
% The limiting value of the option pricing formulae in this
% case are also taken into account in the "if" statements
% below. Hence, though a Warning occurs, one need not worry
% about it – for this reason it has been switched off.

if (0<Y)&(Y<1)
    a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=0;
    c(find(isinf(c)))=inf;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(a,b+2,c))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(c,b,a);
    Call=(Call>0).*Call;
elseif (Y>1)
    a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,-b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2-b,c);
    Call=(Call>0).*Call;
elseif (Y<0)
    a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2+b,c);
    Call=(Call>0).*Call;
else
    a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,-b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2-b,c);
    Call=(Call>0).*Call;
end

Put=strike.*exp(-r.*(T-t)).*(1-ncx2cdf(a,b+2,c))-
      So.*exp(-q.*(T-t)).*ncx2cdf(c,b,a);
Put=(Put>0).*Put;
}

a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,-b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2-b,c);
    Call=(Call>0).*Call;
elseif (Y<0)
    a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2+b,c);
    Call=(Call>0).*Call;
else
    a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,-b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2-b,c);
    Call=(Call>0).*Call;
end

a=(strike.*exp(-(r-q).*(T-t))).^((2*(1-Y))./((1-Y).^2* v));
    c=So^((2*(1-Y))./((1-Y).^2*v));
    a(find(isinf(a)))=inf;
    c(find(isinf(c)))=0;
    Call=So.*exp(-q.*(T-t)).*(1-ncx2cdf(c,b,a))-
    strike.*exp(-r.*(T-t)).*ncx2cdf(a,2+b,c);
    Call=(Call>0).*Call;
B.3 CEVprice.m

\[ c = \frac{S_0^2(1-Y)}{((1-Y)^2v)}; \]
\[ a(\text{find(isinf}(a))) = 0; \]
\[ c(\text{find(isinf}(c))) = \text{inf}; \]
\[ \text{Put} = \text{strike} \cdot \exp(-r \cdot (T-t)) \cdot (1 - \text{ncx2cdf}(a, 2-b, c)) - \ldots \]
\[ \quad \text{So} \cdot \exp(-q \cdot (T-t)) \cdot \text{ncx2cdf}(c, -b, a); \]
\[ \text{Put} = (\text{Put} > 0) \cdot \text{Put}; \]
\[ \text{elseif } Y == 1 \]
\[ [\text{Call}, \text{Put}] = \text{blsprice}(\text{So}, \text{strike}, r, T-t, \text{sigma}, q); \]
\[ \text{else} \]
\[ '\text{error}' \]
\[ \text{end} \]

The inputs of this function represent the following:

- \text{So, strike, r and q} - These are the initial share price, the strike price, the constant risk-free rate and the constant dividend yield, respectively.
- \text{T and t} - These are the maturity time and the initial time, respectively.
- \text{Y} - This is the \( \gamma \) parameter for the CEV model.
function price=CEVFDS(Nx,Nt,Y,S,K,r,q,sigma,T,Smax,Smin,type)

h=(Smax-Smin)/Nx; k=T/Nt; mu=k/h^2;

D1=(diag(1:Nx-1).*h+diag(ones(1,Nx-1)).*Smin);
D2=(diag(1:Nx-1).*h+diag(ones(1,Nx-1)).*Smin).^(2*Y);
T1=diag(ones(Nx-2,1),1)+diag(-ones(Nx-2,1),-1);
T2=diag(ones(Nx-1,1)*(-2))+diag(ones(Nx-2,1),-1)+...
    diag(ones(Nx-2,1),1);
B=(1+r*k)*eye(Nx-1)-0.5*mu*sigma^2*D2*T2-...
    (r-q)*k/(2*h)*D1*T1;

S0=Smin+h:h:Smax-h;
Tvec=0:k:T;

% The boundry value changes depending on the type of
% option for which a price is required. Hence, the use
% of the "switch" function.
switch type
    case 'call'
        V0n=max((S0-K),0)';
        VmSmin=(0*Tvec)';
        VmSmax=(Smax-K*exp(-r*Tvec))';
    case 'put'
        V0n=max((K-S0),0)';
        VmSmin=K*exp(-r*Tvec)';
        VmSmax=zeros(1,Nt+1)';
    case 'doc'
        V0n=max((S0-K),0)';
        VmSmin=(0*Tvec)';
        VmSmax=(Smax-K*exp(-r*Tvec))';
    case 'dop'
        V0n=max((K-S0),0)';
        VmSmin=(0*Tvec)';
        VmSmax=zeros(1,Nt+1)';
    case 'uoc'
        V0n=max((S0-K),0)';
        VmSmin=(0*Tvec)';
        VmSmax=zeros(1,Nt+1)';
    case 'uop'
        V0n=max((K-S0),0)';
        VmSmin=K*exp(-r*Tvec)';
end
B.4 CEVFDS.m

```
Vmsmax=zeros(1,Nt+1)';
end

bm=zeros(1,Nx-1)';
sol=zeros(Nt,Nx-1);
Vm=V0n;

for i=1:Nt;
    bm(1,1)=Vmsmin(i+1,1)*((r-q)*k/(2*h)*(Smin+h)-
    sigma^2*mu/2*(Smin+h)^(2*Y));
    bm(Nx-1,1)=Vmsmax(i+1,1)*((r-q)*k/(2*h)*(Smin+(Nx-1)*h)-
    sigma^2*mu/2*(Smin+(Nx-1)*h)^(2*Y));
    Vmplus1=B\(Vm-bm);
    sol(i,:)=Vmplus1';
    Vm=Vmplus1;
end;

sol=flipud(sol);
sol=[sol;V0n'];
sol=[flipud(Vmsmin) sol flipud(Vmsmax)];
price=interp1([Smin S0 Smax],sol(:,:),S);
```
REFERENCES


