WORLDQUANT UNIVERSITY

Compiled Content Module 5

MScFE 630 Computational Finance

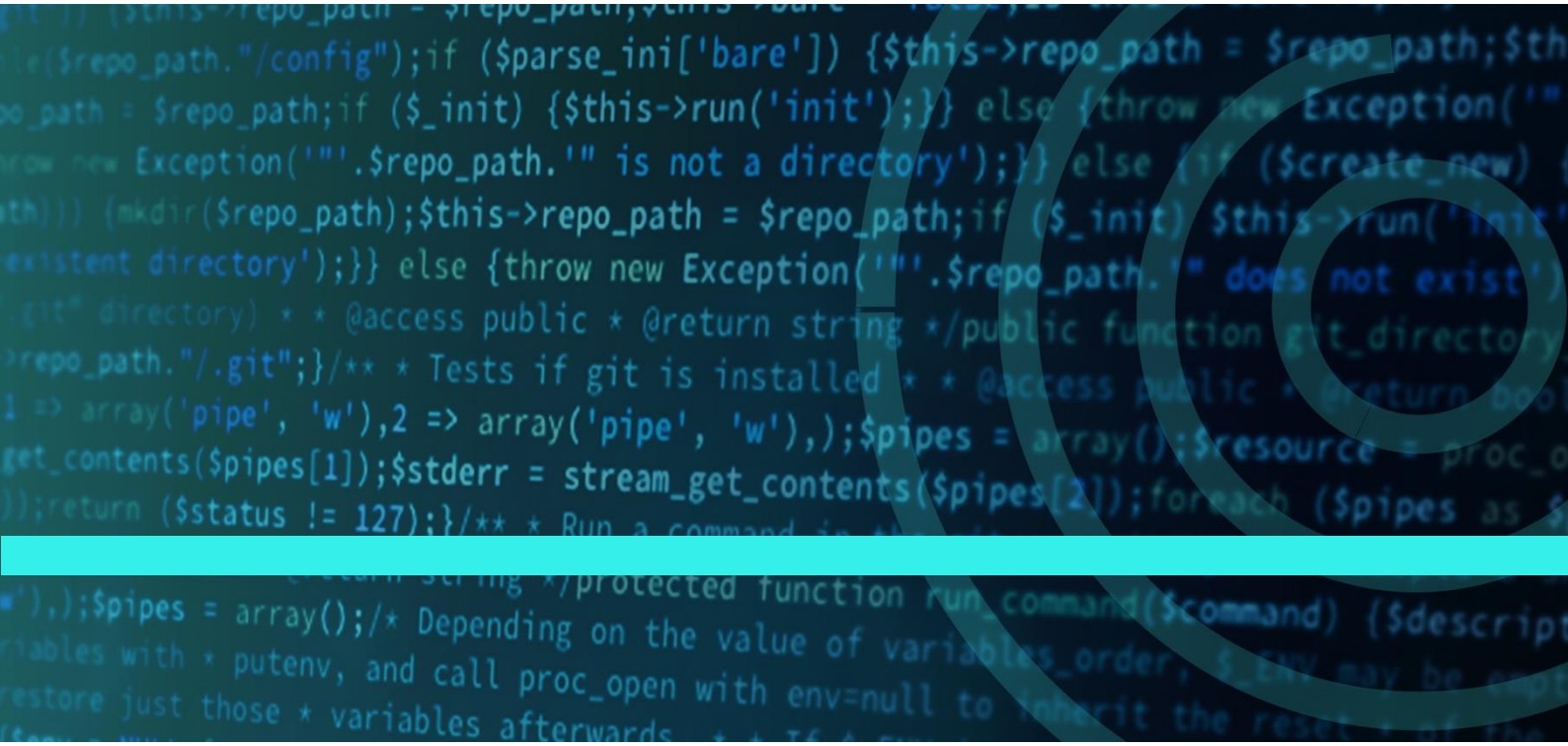


Table of Contents

Revised: 08/19/2019

Module 5: Monte Carlo Methods for Risk Management

Module 5 explores the result of relaxing the constant volatility assumption in the Black-Scholes model. The module begins by introducing local volatility, then continues by deriving Dupire's equation and introducing the Constant Elasticity of Variance (CEV) model. At the end of the module, stochastic volatility and the Heston model are introduced as well.

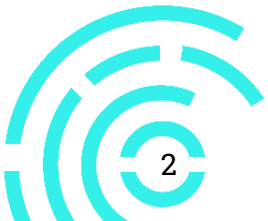
Unit 1: Local Volatility

Unit 1: Notes

Up until this point, we have been working within a Black-Scholes world where we have assumed our assets all have their own constant volatility, which we have usually denoted by σ. This means that, no matter the strike or maturity of an option, the volatility for the underlying is the same. In practice, we observe something different. Implied volatility is the constant volatility which would make the Black-Scholes model price for an option with given strike and maturity equal to the market price. This leads to something known as the **implied volatility surface**. The implied volatility surface considers different possible strike values and maturity times, and asks what the implied volatility is for each of these.

If the Black-Scholes model captured the market perfectly, the implied volatility surface would be flat – that is, the implied volatility would be the same for all strikes and maturities. Often, we see differences over strike and maturity. For example, the volatility smile is the name given to the phenomenon of deep in-the-money or out-of-the-money options sometimes having higher implied volatilities than at-the-money options. As a result of these non-constant implied volatility surfaces, we introduce two ways of modeling non-constant volatility. Our goal would usually be to choose these models so that the option prices they give have similar volatility surfaces to those seen in the market (the implied volatility surfaces). In this module, we will be introducing non-constant volatility as both deterministic and stochastic.

When we say an asset's volatility is deterministic, we mean that we can express the diffusion coefficient in the asset's stochastic differential equation (SDE) as a function of time and the asset's price at that time.



This means we can write the diffusion coefficient as a function $\sigma(S_t, t)$, where t is the time, S_t is the asset's value at time t, and $\sigma(S_t, t)$ is the diffusion coefficient for the asset for the given value and time. We will be keeping all of our other Black-Scholes assumptions the same – constant risk-free interest rate, no risk of default, frictionless trading, etc. The SDE which defines an asset's dynamics at time t will now be given by:

$$dS_t = \mu S_t dt + \sigma(S_t, t) S_t dW_t,\ S_0 = s_0, \tag{1}$$

where μ is a constant and W_t is a standard Brownian motion. Note that this is under the real-world measure – under the risk-neutral measure, the μ in equation (1) will be given by the risk-free continuously-compounded rate, r.

In order to price using these assumptions, we need to find a formula for $\sigma(S_t, t)$. We will be using market prices to do this. Before that, we need to introduce two new equations, which we will use together in the next unit.

Fokker-Planck equation

This derivation follows the form of Rouah (n.d.). Consider the following SDE:

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t,\ \ X_0 = x_0, \tag{2}$$

where W_t is standard Brownian motion. Let's consider a function, $H(\bullet)$, which has compact support and is twice differentiable.

For our purposes, this means $H(\infty) = H(-\infty) = 0$. We are going to find two expressions for the expectation of $\frac{\partial}{\partial t} H(X_t)$. Firstly, we can apply Ito's Lemma to $H(X_t)$:

$$dH(X_t) = \frac{\partial H(X_t)}{\partial x} dX_t + \frac{1}{2} \frac{\partial^2 H(X_t)}{\partial x^2} (dX_t)^2$$

$$= \frac{\partial H(X_t)}{\partial x} (\mu(X_t, t)dt + \sigma(X_t, t)dW_t) + \frac{1}{2} \frac{\partial^2 H(X_t)}{\partial x^2} (\sigma(X_t, t)^2 dt)$$

$$= \left(\frac{\partial H(X_t)}{\partial x} \mu(X_t, t) + \frac{1}{2} \frac{\partial^2 H(X_t)}{\partial x^2} \sigma(X_t, t)^2\right) dt \; + \frac{\partial H(X_t)}{\partial x} \sigma(X_t, t) dWt. \tag{3}$$

With this, we can write $H(X_t)$ in terms of integrals, and then calculate the expected value of $H(X_t)$:

$$H(X_t) = H(X_0) + \int_0^t \left(\frac{\partial H(X_s)}{\partial x} \mu(X_s, s) + \frac{1}{2} \frac{\partial^2 H(X_s)}{\partial x^2}\right) ds \; + \int_0^t \frac{\partial H(X_s)}{\partial x} \sigma(X_s, s) dW_s$$

$$\Rightarrow \mathbb{E}[H(X_t)] = H(X_0) + \mathbb{E}\left[\int_0^t \left(\frac{\partial H(X_s)}{\partial x} \mu(X_s, s) + \frac{1}{2} \frac{\partial^2 H(X_s)}{\partial x^2} \sigma(X_s, s)^2\right) ds\right], \tag{4}$$

since the expectation of any integral with respect to Brownian motion is 0, and X_0 and therefore $H(X_0)$ are constants. We can then take the derivative of this expression with respect to t:

$$\frac{\partial}{\partial t} \mathbb{E}[H(X_t)] = \mathbb{E}\left[\frac{\partial H(X_t)}{\partial x} \mu(X_t, t) + \frac{1}{2} \frac{\partial^2 H(X_t)}{\partial x^2} \sigma(X_t, t)^2\right], \tag{5}$$

which follows from the fundamental theorem of calculus. We can also write this expectation of $H(X_t)$ in terms of the probability density function of X_t. Suppose $\varphi(\bullet, t)$ is the density of X_t at time t. Then:

$$\mathbb{E}[H(X_t)] = \int_{-\infty}^{\infty} H(x)\varphi(x,t)dx. \tag{6}$$

Now, taking the derivative with respect to t:

$$\begin{aligned}\frac{\partial}{\partial t}\mathbb{E}[H(X_t)] &= \int_{-\infty}^{\infty} H(x)\frac{\partial \varphi(x,t)}{\partial t}dx \\ &= \int_{-\infty}^{\infty} \frac{\partial H(x)}{\partial x}\mu(x,t)\varphi(x,t) + \frac{1}{2}\frac{\partial^2 H(x)}{\partial x^2}\sigma(x,t)^2\varphi(x,t)dx.\end{aligned} \tag{7}$$

The second line in this equation comes from equation (5); we have changed the expectation into an integral by multiplying by the density of X_t.

We can use integration-by-parts on the two terms within the integral in this expression:

$$\begin{aligned}&\int_{-\infty}^{\infty} \frac{\partial H(x)}{\partial x}\mu(x,t)\varphi(x,t)dx \\ &= H(x)\mu(x,t)\varphi(x,t)|_{-\infty}^{\infty} - \int_{-\infty}^{\infty} H(x)\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x}dx \\ &= -\int_{-\infty}^{\infty} H(x)\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x}dx,\end{aligned} \tag{8}$$

from the fact that $H(\bullet)$ has a compact support, which means $H(\infty) = H(-\infty) = 0$. With the same reasoning, but applying the steps above twice:

$$\int_{-\infty}^{\infty} \frac{\partial^2 H(x)}{\partial x^2} \sigma(x,t)^2 \varphi(x,t) dx$$

$$= \int_{-\infty}^{\infty} H(x) \frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2} dx. \tag{9}$$

We sub equations (8) and (9) into equation (7):

$$\int_{-\infty}^{\infty} \frac{\partial H(x)}{\partial x} \mu(x,t)\varphi(x,t) + \frac{1}{2} \frac{\partial^2 H(x)}{\partial x^2} \sigma(x,t)^2 \varphi(x,t) dx$$

$$= \int_{-\infty}^{\infty} -H(x) \frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2} H(x) \frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2} dx$$

$$\Rightarrow \int_{-\infty}^{\infty} H(x) \left[-\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2} \frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2} - \frac{\partial \varphi(x,t)}{\partial t} \right] dx = 0$$

$$\Leftrightarrow -\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2} \frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2} - \frac{\partial \varphi(x,t)}{\partial t} = 0$$

$$\Leftrightarrow \frac{\partial \varphi(x,t)}{\partial t} = -\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2} \frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2}. \tag{10}$$

Line 3 in the above equation comes from subtracting the first line in equation (7) from the second line in the above equation. Line 4 in the above equation comes from the fact that the only restriction we placed on $H(\bullet)$ is that it needs a compact support – thus the integral will

only be 0 for all $H(\bullet)$ if all the integrands are 0. The final line in the above equation is known as the Fokker-Planck equation. It is also sometimes called the Kolmogorov forward equation.

Breeden-Litzenberger formulas

These equations were first derived by Breeden and Litzenberger (1978). This derivation follows from Davis (2011).

We are going to express a call price as a function of maturity time and strike price:

$$C(T,K) = e^{-rT}\mathbb{E}^{\mathbb{Q}}[(S_T - K)^+] = e^{-rT}\int_K^\infty (x-K)\varphi(x,T)dx, \tag{11}$$

where T is the maturity time, K is the strike price, r is the risk-free continuously compounded interest rate, S_t is the stock price at time T, and $\varphi(\bullet, T)$ is the probability density function of S_T. We can then calculate mathematical derivatives of this function, $C(T,K)$:

$$\begin{aligned}
\frac{\partial}{\partial K}C(T,K) &= e^{-rT}\int_K^\infty \frac{\partial}{\partial K}(x-K)\varphi(x,T)dx \\
&= e^{-rT}\int_K^\infty -\varphi(x,T)dx \\
&= -e^{-rT}\mathbb{P}[S_T > K] \\
&= -e^{-rT}(1-\mathbb{P}[S_T \le K]) \\
&= e^{-rT}\big(F_{S_T}(K,T) - 1\big),
\end{aligned} \tag{12}$$

where $F_{ST}\ (K,T\ )$ is the cumulative distribution function of S_T. We can differentiate again:

$$\begin{aligned}\frac{\partial^2}{\partial K^2}C(T,K) &= \frac{\partial}{\partial K}\frac{\partial}{\partial K}C(T,K)\\ &= \frac{\partial}{\partial K}e^{-rT}\left(F_{S_T}(K,T)-1\right)\\ &= e^{-rT}\varphi(K,T).\end{aligned} \tag{13}$$

The Breeden-Litzenberger formulas are thus given by $\frac{\partial}{\partial K}C(T,K) = e^{-rT}\left(F_{S_T}(K,T)-1\right)$ and $\frac{\partial^2}{\partial K^2}C(T,K) = e^{-rT}\varphi(K,T)$. We are going to use these equations in conjunction with the Fokker-Planck equations to derive a formula for $\sigma(S_t,t)$.

Unit 1 : Video Transcript

Hello, and welcome to the first video of Module 5. In this video we will briefly go through what local volatility is. Then, we will spend time developing the mathematics required to use a local volatility model.

Consider the standard Black-Scholes model. If we know the risk-free continuously compounded interest rate, and a stock's volatility, we can calculate the price of a vanilla call and put option for a given strike and maturity. What you might observe in the market, however, is that the volatility you need for the stock to match market option prices is different depending on the maturity and strike of the option. This is because the Black-Scholes model is not a perfect representation of reality. This leads to what is known as the implied volatility surface – the set of volatilities which give Black-Scholes model prices equal to the market prices for different strikes and maturities. So, we see that our Black-Scholes assumptions aren't completely accurate. In this module and the next, we will be relaxing two of our assumptions – namely, constant stock volatilities, and constant risk-free rate. Let's start by looking at a type of non-constant volatility, called local volatility.

What is local volatility?

A local volatility model is just one where the diffusion coefficient in the asset's stochastic differential equation is a deterministic function of time and the asset price. This means that, if you know what time it is, and what the current asset price is, you should be able to calculate the diffusion coefficient. You could take this one step further and allow for the diffusion co-efficient to be fully stochastic. This is dealt with in more detail later in this module. For now, though, let's focus on local volatility.

With our local volatility assumption, our new share price dynamics become:

$$dS_t = \mu S_t dt + \sigma(t, S_t) S_t dW_t,$$

where W_t is a standard Brownian Motion. Note that the only difference between this equation and the one that we have dealt with so far is that diffusion coefficient is now a function. This means that we are still assuming that the other parameters are constant.

Fokker-Planck equations

Now that we know what a local volatility model is, let's derive some mathematics that will allow us to use these models. The first set of equations that will help us to achieve this goal are the Fokker-Planck equations.

Suppose that we have an SDE of the form:

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t,$$

and that the density function of X_t is $\varphi(x, t)$. The Fokker-Planck equations give us a way to characterize this density function in terms of its partial derivatives. The final result is:

$$\frac{\partial \varphi(x,t)}{\partial t} = -\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2}\frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2}.$$

In order to arrive at this result, you will need to rely on some function having compact support. This means that the function is only non-zero over a finite range. So, as we evaluate the function over positive or negative infinity, the function evaluates to 0. Otherwise, the proof makes use of integration by parts and the Fundamental Theorem of Calculus. The full derivation of this result can be found in the notes.

The mathematics underlying this derivation isn't necessarily too complicated, but there is a fair amount to go through. We really encourage you to spend sufficient time trying to understand the proof.

Breeden-Litzenberger formulas

We now move onto the next set of equations that we will need to use local volatility models. These equations are known as the Breeden-Litzenberger formulas. These equations allow us to write the underlying density functions of our assets in terms of sensitivities of options. We first start by taking the definition of a call option:

$$C(T,K) = e^{-rT}\int_K^\infty (x-K)\,\varphi(x,T)dx.$$

Then, by taking derivatives, we are able to arrive at the following two equations:

$$\frac{\partial}{\partial K}C(T,K) = e^{-rT}\left(F_{S_T}(K,T)-1\right)$$

and

$$\frac{\partial^2}{\partial \mathrm{K}^2}C(T,K) = e^{-rT}\varphi(K,T).$$

Once again, the full derivation can be found in the notes for this section.

In the next video, we will look at a local volatility model known as the CEV model, as well as derive Dupire's equation. Note that the derivation of Dupire's equation relies heavily on the equations derived in this video, so please make sure that you are comfortable with the mathematics explored in this unit before moving on.

Unit 2: Dupire's Equation and the CEV Model

Unit 2 : Notes

In the previous unit, we introduced the concept of local volatility, and showed some equations which we are going to need in this section. We will firstly be deriving Dupire's equation using our Fokker-Planck and Breeden-Litzenberger equations. We will then briefly introduce the Constant Elasticity of Variance (CEV) model, which we will use to test the Dupire equation in the next section.

Dupire's equation

This section follows Davis (2011). Recall the following from the previous section. X_t is a stochastic process with the following SDE:

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t,\ X_0 = x_0, \tag{14}$$

where W_t is standard Brownian motion. If we let the probability density function of X_t be $\varphi(\bullet, t)$, then

$$\frac{\partial \varphi(x,t)}{\partial t} = -\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2}\frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2}, \tag{15}$$

which is known as the Fokker-Planck equation.

Now, we define $C(T,K)$ as previously, namely:

$$C(T,K) = e^{-rT}\int_K^\infty (x-K)\,\phi(x,T)dx, \tag{16}$$

which is the price of a call option with strike K, maturity time T, risk-free continuously-compounded interest rate r, and stock price at time T of S_T, whose probability density we define as $\phi(\bullet, T)$.

The Breeden-Litzenberger equations tell us that

$$\frac{\partial}{\partial K} C(T,K) = e^{-rT}\big(F_{S_T}(K,T) - 1\big), \tag{17}$$

where $F_{S_T}(\bullet, T)$ is the cumulative distribution function of S_T, and

$$\frac{\partial^2}{\partial K^2} C(T,K) = e^{-rT}\phi(K,T). \tag{18}$$

To derive the Dupire equation, we consider the derivative of $C(T,K)$ with respect to T:

$$\begin{aligned}\frac{\partial}{\partial T} C(T,K) &= -rC(T,K) + e^{-rT}\int_K^\infty (x-K)\frac{\partial}{\partial T}\phi(x,T)dx \\ &= -rC(T,K) + e^{-rT}\int_K^\infty (x-K)\left[-\frac{\partial \mu(x,T)\phi(x,T)}{\partial x} + \frac{1}{2}\frac{\partial^2 \sigma(x,T)^2\phi(x,T)}{\partial x^2}\right]dx,\end{aligned} \tag{19}$$

which we arrive at by applying the Fokker-Planck equation. We are going to evaluate this integral using integration-by-parts:

$$\int_K^\infty (K-x)\frac{\partial \mu(x,T)\phi(x,T)}{\partial x}dx = (K-x)\mu(x,T)\phi(x,T)|_K^\infty$$

$$-\int_K^\infty \frac{\partial (K-x)}{\partial x}\mu(x,T)\phi(x,T)dx \tag{20}$$

$$= \int_K^\infty \mu(x,T)\phi(x,T)dx,$$

where we have used the fact that $\phi(\infty,T)=0$.

$$\int_K^\infty (x-K)\frac{\partial^2 \sigma(x,T)^2\phi(x,T)}{\partial x^2}dx$$

$$= (x-K)\frac{\partial \sigma(x,T)^2\phi(x,T)}{\partial x}|_K^\infty - \int_K^\infty \frac{\partial (x-K)}{\partial x}\frac{\partial \sigma(x,T)^2\phi(x,T)}{\partial x}dx \tag{21}$$

$$= -\sigma(x,T)^2\phi(x,T)|_K^\infty = \sigma(K,T)^2\phi(K,T),$$

where we have applied integration-by-parts twice and used the fact that $\phi(\infty,T) = \frac{\partial \phi(\infty,T)}{\partial x} = 0$. In total, we have:

$$\frac{\partial}{\partial T}C(T,K) = -rC(T,K) + e^{-rT}\int_K^\infty \mu(x,T)\phi(x,T)dx + \frac{1}{2}e^{-rT}\sigma(K,T)^2\phi(K,T). \tag{22}$$

In order to implement the Breeden-Litzenberger equations, we need to make assumptions about the form of $\mu(x,T)$ – we are going to assume that $\mu(x,T) = rx$. Then,

$$e^{-rT}\int_{\mathrm{K}}^{\infty}\mu(x,T)\phi(x,T)dx = re^{-rT}\int_{K}^{\infty}x\phi(x,T)dx$$

$$= re^{-rT}\left[\int_{\mathrm{K}}^{\infty}(x-K)\phi(x,T)dx + K\int_{K}^{\infty}\phi(x,T)dx\right]$$

$$= rC(T,K) - rK\frac{\partial}{\partial K}C(T,K),$$

(23)

from equation (17), and

$$e^{-rT}\sigma(K,T)^2\phi(T,K) = \sigma(K,T)^2\frac{\partial^2}{\partial K^2}C(T,K),$$

(24)

from equation (18). Subbing these into equation (22) gives

$$\frac{\partial}{\partial T}C(T,K) = -rK\frac{\partial}{\partial K}C(T,K) + \frac{1}{2}\sigma(K,T)^2\frac{\partial^2}{\partial K^2}C(T,K)$$

$$\Rightarrow \sigma(K,T) = \sqrt{\frac{2\frac{\partial}{\partial T}C(T,K) + 2rK\frac{\partial}{\partial K}C(T,K)}{\frac{\partial^2}{\partial K^2}C(T,K)}}.$$

(25)

It is common to express $\sigma(K,T)$ as $K\bar{\sigma}(K,T)$, which makes our equation:

$$\bar{\sigma}(K,T) = \frac{\sqrt{2}}{K}\sqrt{\frac{\frac{\partial}{\partial T}C(T,K) + rK\frac{\partial}{\partial K}C(T,K)}{\frac{\partial^2}{\partial K^2}C(T,K)}}.$$

(26)

This is the Dupire equation, which we will be using to estimate volatility in our simulations.

The CEV model

The Constant Elasticity of Variance (CEV) model proposes stock dynamics given by:

$$dS_t = \mu S_t dt + \sigma S_t^{\gamma} dW_t, \tag{27}$$

where γ and σ are non-negative, and all other variables are defined as in the usual Black-Scholes model. If $\gamma = 1$, we are back to the standard Black-Scholes model. For $\gamma < 1$, stock volatility increases as prices decrease – an effect called leverage, which is often observed in practice.

If we were to express our stock dynamics as

$$dS_t = \mu S_t dt + S_t \sigma(S_t, t) dW_{t,} \tag{28}$$

we would have that $\sigma(S_t, t) = \sigma S_t^{\gamma - 1}$. Note that this does not depend on time, only on stock price. This is the expression for local volatility, which we will be comparing to our Dupire estimates to the next unit.

Unit 2: Video Transcript

In this video, we will be exploring Dupire's equation and the CEV model. The first part of the video will use the Fokker-Planck and Breeden-Litzenberger equations to derive Dupire's equation. The goal of this will be to determine a formula that we can use for evaluating the volatility of an asset. Finally, we will look at the Constant Elasticity of Variance, or CEV, model and the assumptions this model makes about asset volatility. We will be relying quite heavily on the previous unit's mathematics, so please make sure that you are comfortable with it before starting with this video.

Recall from the previous video that, if we define an SDE as:

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_{t,}$$

and let our X_0 value be a constant and W_t be a standard Brownian motion then we have:

$$\frac{\partial \varphi(x,t)}{\partial t} = -\frac{\partial \mu(x,t)\varphi(x,t)}{\partial x} + \frac{1}{2}\frac{\partial^2 \sigma(x,t)^2 \varphi(x,t)}{\partial x^2}.$$

Note that, in the previous equation, $\varphi(\bullet, t)$ is the probability density of function of X_t. Now, let's assume that we are pricing a call option. This means that we can write the price of the call option as:

$$C(T,K) = e^{-rT}\int_K^\infty (x-K)\phi(x,T)dx.$$

In this equation, we take $\phi(\bullet, T)$ to be the probability density function of S_T, K is the strike of the option, r is the risk-free interest rate, and T is the maturity time of the option.

The Breeden-Litzenberger now allows us to arrive at two equations:

$$\frac{\partial}{\partial K} C(T,K) \;=\; e^{-rT}(F_{S_T}(K,T) \;-\; 1)$$

and

$$\frac{\partial^2}{\partial \mathrm{K}^2}\, \mathrm{C}(\mathrm{T},\mathrm{K}) \;=\; \mathrm{e}^{-\mathrm{rT}}\phi(\mathrm{K},\mathrm{T}),$$

where $\phi(\bullet, T)$ is defined as before, and $F_{S_T}(\bullet, T)$ is the cumulative distribution function of S_T.

In order to get to the Dupire equation, we need to consider the derivative of the call option. We can write this as:

$$\frac{\partial}{\partial T} C(T,K) = -rC(T,K) + e^{-rT} \int_K^\infty (x-K) \frac{\partial}{\partial T} \phi(x,T) dx.$$

We can evaluate $\frac{\partial}{\partial T}\phi(x,T)$ using the Fokker-Planck equation. The next thing we do is apply integration-by-parts and use the fact that the density $\phi(\bullet, T)$ is compact. This is largely about rearranging equations, which is covered in the notes, so we won't go through this in the video.

Now, we are able to write the derivative of the call option with respect to maturity time as:

$$\frac{\partial}{\partial T} C(T,K) = -rC(T,K) + e^{-rT} \int_K^\infty \mu(x,T)\phi(x,T) dx + \frac{1}{2} e^{-rT} \sigma(K,T)^2 \phi(K,T).$$

Following on from this, before we can apply the Breeden-Litzenberger equations, we need to assume that $\mu(x,T)$ is of an appropriate form. We assume that $\mu(x,T) = rx$. This allows us to write the following equation:

$$e^{-rT} \textstyle\int_K^\infty \mu(x,T)\phi(x,T) dx = rC(T,K) - rK \frac{\partial}{\partial K} C(T,K).$$

Using this, and all the other equations that we have found, we can finally conclude that:

$$\sigma(K,T) = \sqrt{\frac{2\frac{\partial}{\partial T}C(T,K) + 2rK\frac{\partial}{\partial K}C(T,K)}{\frac{\partial^2}{\partial K^2}C(T,K)}}.$$

The last equation, which is known as the Dupire equation, follows from a few other manipulations which can be found in the notes. The most important thing that we are seeking to achieve in this video is an understanding of the assumptions underlying the Dupire equation. The actual mathematics involved in arriving at this equation is sufficiently covered in the notes.

The CEV model

We will now move onto working through the CEV model. In the CEV model, we assume that the volatility of the asset depends on the asset price at that time, in the following way:

$$dS_t = \mu S_t dt + \sigma S_t^{\gamma} dW_t,$$

where γ and σ are both non-negative, and all the other variables are defined in the way as in the Black-Scholes model. If we rewrite our dynamics as:

$$dS_t = \mu S_t dt + S_t \sigma(S_t, t) dW_t.$$

If we do this, then we should have it that $\sigma(S_t, t) = \sigma S_t^{\gamma-1}$. Note that we are assuming that our volatility process doesn't explicitly depend on time; only on the underlying asset price. We can now use this expression for volatility as a comparison to the Dupire estimate that we derived previously. The notes go into more detail of some of the problems one might encounter in implementing the Dupire equation in practice.

In the next video, we will be going through how to apply the equations derived in this video in Python.

Unit 3: Implementing Dupire's Equation

Unit 3: Notes

In this unit, we are going to be implementing Dupire's equation in Python and comparing it to the closed-form local volatility in the CEV model. For this example, we are comparing our Dupire values to something we know – namely, the CEV values. Usually, you would use market call and put prices and Dupire's equation to estimate local volatility values in a model, allowing you to account for an implied volatility surface. To begin, recall that the CEV dynamics of a stock are given by:

$$dS_t = \mu S_t dt + S_t \sigma(S_t, t) dW_{t,} \tag{29}$$

where $\sigma(S_t, t) = \sigma S_t^{\gamma-1}$ and all other variables are defined as usual in the Black-Scholes model. It can be shown that there is a closed-form solution to the value of a call option, given as:

$$C(S_0, K, T, \sigma, \gamma) = -S_0 \chi(y; z, x) + K e^{-rT}(1 - \chi(x; z-2, y)), \tag{30}$$

where $C(S_0, K, T, \sigma, \gamma)$ is the value of a call on a stock with initial value S_0, strike K, time to maturity T, and σ and γ as per the CEV volatility term, and

$$\kappa = \frac{2r}{\sigma^2(1-\gamma)e^{2r(1-\gamma)T-1}}$$

$$x = \kappa S_0^{2(1-\gamma)} e^{2r(1-\gamma)T}$$

$$y = \kappa K^{2(1-\gamma)}$$

$$z = 2 + \frac{1}{1-\gamma},$$

and $\chi(\bullet; d, \lambda)$ is the cumulative distribution function of a noncentral chi-squared distribution with d degrees of freedom and non-centrality parameter λ.
Details of the derivation of this equation can be found in Hsu *et al.* (2008), but you don't need to worry about how to derive this equation — all we will be doing is writing a function which implements it. You do, however, need to know what each argument in $\chi(\bullet; d, \lambda)$ stands for.

We start off by importing the relevant libraries:

In [ ]:
```
#Importing libraries
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import ncx2
```

Note that we need a new class from the `scipy.stats` library — `ncx2`. This class is used for noncentral chi-squared distributions. As you shall see, we will be using the cumulative distribution, which is called using `ncx2.cdf`. Let's now create our market and stock variables:

In [ ]:
```
#Variable declaration
S0 = 100
sigma = 0.3
gamma = 0.75
r = 0.1
T = 3
```

Recall we are in the CEV model with dynamics as per equation (29). We have set an initial stock price of S_0, a σ constant of 30%, a γ value of 0.75, a risk-free continuously compounded rate of 10%, and a time to maturity of 3 years. With these, we can create a function that finds the call price for a given strike and time to maturity using equation (30):

In [ ]:
```
#Call price under CEV
z = 2 + 1/(1-gamma)
def C(t,K):
    kappa = 2*r/(sigma**2*(1-gamma)*(np.exp(2*r*(1-gamma)*t)-1))
    x = kappa*S0**(2*(1-gamma))*np.exp(2*r*(1-gamma)*t)
    y = kappa*K**(2*(1-gamma))
    return S0*(1-ncx2.cdf(y,z,x))-K*np.exp(-r*t)*ncx2.cdf(x,z-2,y)
```

Notice the use of `ncx2.cdf` to find the cumulative distribution for the given parameters. We need to make this a function of time and stock price, since we need to find the derivative (and second derivative) of this function with respect to these two variables. We are going to use strikes that vary between 80 and 120 (inclusive):

In [ ]:
```
#Strikes to test volatility
test_strikes = np.linspace(80,120,41)
```

Recall that the Dupire equation is:

$$\overline{\sigma}(K,T) = \frac{\sqrt{2}}{K}\sqrt{\frac{\frac{\partial}{\partial T}C(T,K) + rK\frac{\partial}{\partial K}C(T,K)}{\frac{\partial^2}{\partial K^2}C(T,K)}}. \tag{31}$$

We need to estimate $\frac{\partial}{\partial t}C(T,K)$, $\frac{\partial}{\partial K}C(T,K)$, and $\frac{\partial^2}{\partial K^2}C(T,K)$. We do so by using the following estimates for derivatives:

$$\frac{\partial}{\partial a}f(a,b,c,\dots) \approx \frac{f(a+\Delta,b,c,\dots) - f(a-\Delta,b,c,\dots)}{2\Delta}$$

$$\frac{\partial^2}{\partial a^2}f(a,b,c,\dots) \approx \frac{f(a+2\Delta,b,c,\dots) - 2f(a+\Delta,b,c,\dots) + f(a,b,c,\dots)}{\Delta^2}, \tag{32}$$

where Δ is some small number. This approximation comes from the definition of a derivative,

$$\frac{d}{dx}f(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

We implement equation (32) as follows:

In [ ]:
```
#Estimating partial derivatives
delta_t = 0.01
delta_K = 0.01
dC_dT = (C(T+delta_t,test_strikes)-C(T-delta_t,test_strikes))/(2*delta_t)
dC_dK = (C(T,test_strikes+delta_K)-C(T,test_strikes-delta_K))/(2*delta_K)
d2C_dK2 = (C(T,test_strikes+2*delta_K)-2*C(T,test_strikes+delta_K)+C
        (T,test_strikes))/(delta_K**2)
```

We have used the same Δ for both our derivative with respect to T and to K. We can then use these estimates in our Dupire equation (31):

In [ ]:
```
#Estimating local volatility - Dupire Equation
vol_est = np.sqrt(2)/test_strikes*np.sqrt((dC_dT+r*test_strikes*dC_dK)
        /d2C_dK2)
```

We can plot these Dupire estimates against the CEV closed-form local volatility, which is given by $\sigma K^{\gamma-1}$:

In [ ]:
```
#Plotting closed-form and Dupire equation
plt.plot(test_strikes, sigma*test_strikes**(gamma-1))
plt.plot(test_strikes, vol_est, '.')
```

Figure (1) shows the plot of the closed-form versus Dupire equation estimates, with axis labels added. Note that, since we are estimating our derivatives with respect to T and K, we will have some error.

Figure (2) shows the error between the Dupire estimates and the closed-form local volatility. As you can see, the error is minimal, displaying the usefulness of the Dupire equation.

Note that, in this example, we already knew the local volatility, as it is given by the CEV model. We simply illustrated how the Dupire equation matches the true local volatility. In order to implement Dupire in practice, you would need to use liquid call options to find your estimates for $\frac{\partial}{\partial t}C(T,K)$, $\frac{\partial}{\partial K}C(T,K)$, and $\frac{\partial^2}{\partial K^2}C(T,K)$. You may not have evenly spaced strike values and maturity times, so you would need to be careful with how you use finite difference to find these estimates.

While this may pose some problems, it allows you to take account of the different implied volatilities often seen in practice.

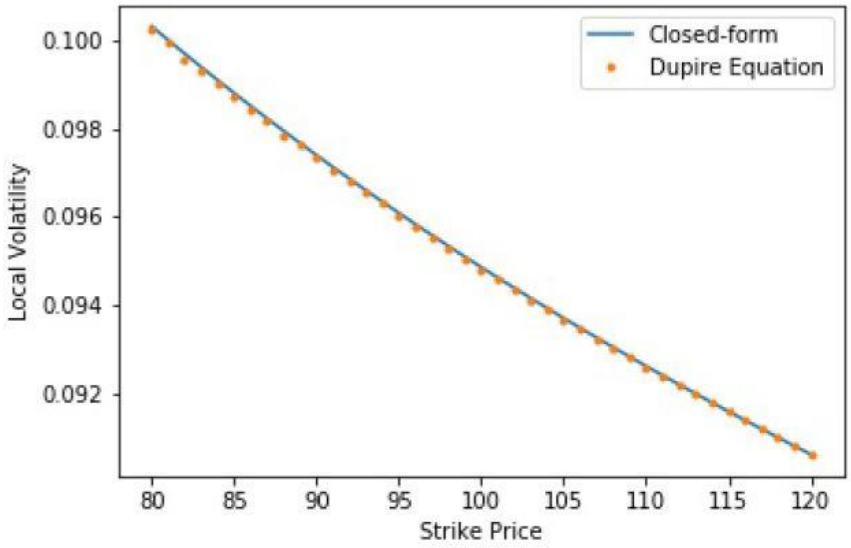


Figure 1: Local volatility for different strikes

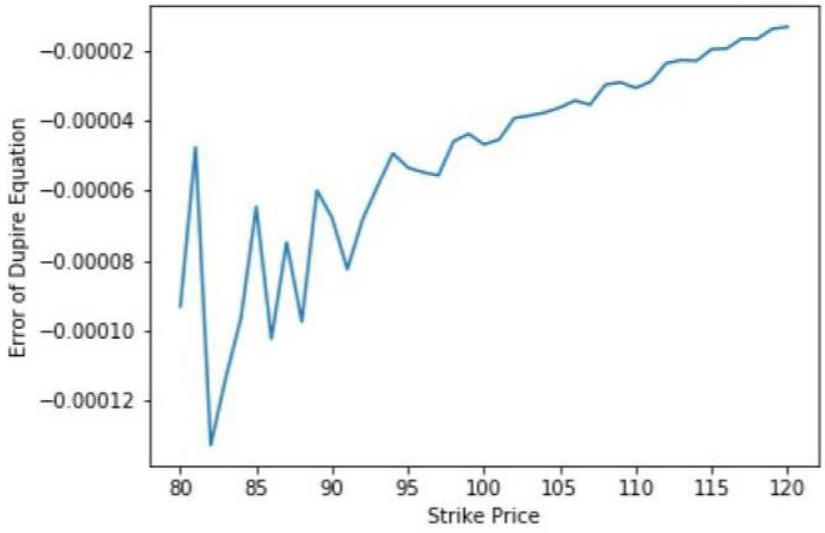


Figure 2: Error in local volatility for different strikes

The final thing to mention with regard to the Dupire equation is its applicability to put options. Equation (31) holds in exactly the same way if one replaces the call price function with a put price function – that is,

$$\overline{\sigma}(K,T) = \frac{\sqrt{2}}{K}\sqrt{\frac{\frac{\partial}{\partial T}P(T,K) + rK\frac{\partial}{\partial K}P(T,K)}{\frac{\partial^2}{\partial K^2}P(T,K)}}, \tag{33}$$

where $P(T,K)$ is the price of a put option with time to maturity T, and strike K.

Unit 3 : Video Transcript

Hello, in this video we will go through how to implement Dupire's equation and the CEV model in Python.

Let's start with the CEV model. Conveniently, this model has a closed-form solution for a European call option. The actual solution can be seen in the notes. The only point of complexity that comes in when dealing with this closed-form solution is that it requires the use of the cumulative distribution of a noncentral chi-squared distribution. Thankfully, Python has libraries which can handle various aspects of the chi-squared distribution. With this closed-form solution, we have something to compare the accuracy of the Dupire value against.

Now that we have a function for our call price with local volatility, we can move on to calculating an estimate for the Dupire equation. The problem we face here is in estimating the derivatives of the call option with respect to maturity, and strike, and the second derivative with respect to strike.

We can estimate these values using central-difference estimates. A central-difference estimate writes the first and second order partial derivative of a function as:

$$\frac{\partial}{\partial a} f(a, b, c, \dots) \approx \frac{f(a + \Delta, b, c, \dots) - f(a - \Delta, b, c, \dots)}{2\Delta}$$

$$\frac{\partial^2}{\partial a^2} f(a, b, c, \dots) \approx \frac{f(a + 2\Delta, b, c, \dots) - 2f(a + \Delta, b, c, \dots) + f(a, b, c, \dots)}{\Delta^2}.$$

We would use our call price function as the f function in the previous equation.

Let's go through the code which will allow for us to apply Dupire's equation. Note that we will be cycling through strike price when estimating volatilities.

The first thing we do, as always, is to import our libraries and declare our variables.

In [ ]:
```
#Importing libraries
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import ncx2
```

In [ ]:
```
#Variable declaration
S0 = 100
sigma = 0.3
gamma = 0.75
r = 0.1
T = 3
```

In [ ]:
```
#Strikes to test volatility
test_strikes = np.linspace(80,120,41)
```

The only new thing here is that we are importing an additional library from scipy. This allows us to deal with a noncentral chi-squared distribution, which we will need for pricing options using the CEV model.

Next, we define a function which can calculate our call price for a given strike and maturity time. We need this so that we can estimate the sensitivities of the call price, which is necessary to apply Dupire's equation. This function is:

In [ ]:
```
#Call price under CEV
z = 2 + 1/(1-gamma)
def C(t,K):
    kappa = 2*r/(sigma**2*(1-gamma)*(np.exp(2*r*(1-gamma)*t)-1))
    x = kappa*S0**(2*(1-gamma))*np.exp(2*r*(1-gamma)*t)
    y = kappa*K**(2*(1-gamma))
    return S0*(1-ncx2.cdf(y,z,x))-K*np.exp(-r*t)*ncx2.cdf(x,z-2,y)
```

We now need to estimate the partial derivatives of the call price with respect to maturity, and strike. In order to do this, we will be making use of the central difference estimates that we covered earlier in the video. The only additional parameters we need to apply this are step sizes for our strike and maturity time. The code to apply this is:

In [ ]:
```
#Estimating partial derivatives
delta_t = 0.01
delta_K = 0.01
dC_dT = (C(T+delta_t,test_strikes)-C(T-delta_t,test_strikes))/(2*delta_t)
dC_dK = (C(T,test_strikes+delta_K)-C(T,test_strikes-delta_K))/(2*delta_K)
d2C_dK2 = (C(T,test_strikes+2*delta_K)-2*C(T,test_strikes+delta_K)+C
        (T,test_strikes))/(delta_K**2)
```

The `delta_t` and `delta_K` variables are our step sizes. Then the code creates variables which store the partial derivative with respect to maturity, then strike, and finally the second partial derivative with respect to strike.

Finally, we can estimate our volatilities using the following code:

In [ ]:
```
#Estimating local volatility - Dupire Equation
vol_est = np.sqrt(2)/test_strikes*np.sqrt((dC_dT+r*test_strikes*dC_dK)
        /d2C_dK2)
```

If applied correctly, this should result in the following graph:

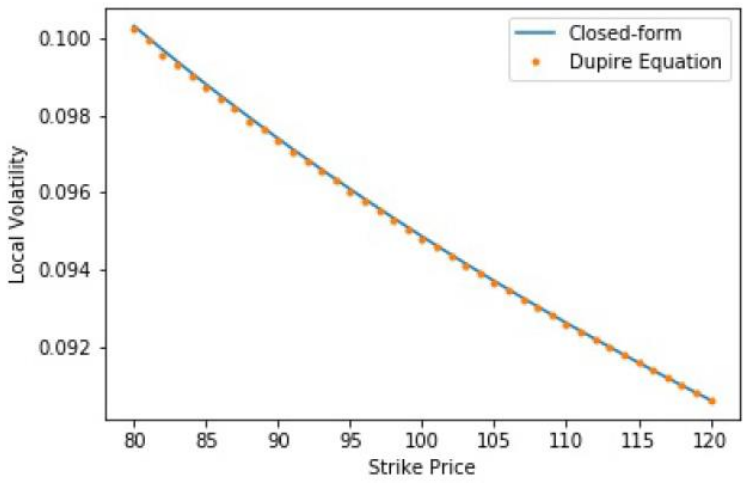


Figure 3: Local volatility for different strikes

Applying Dupire's equation for a put option

The last thing that we will go over in this video is how to apply Dupire's equation for a put option. Conveniently, the Dupire equation for a put option is essentially the same as the one for a call option. The only difference is that we replace the call price function with the put price function. Thus, our equation becomes:

$$\bar{\sigma}(K,T) = \frac{\sqrt{2}}{K}\sqrt{\frac{\frac{\partial}{\partial T}P(T,K) + rK\frac{\partial}{\partial K}P(T,K)}{\frac{\partial^2}{\partial K^2}P(T,K)}}.$$

The method for applying this version of the equation is the same as the one for the call option, which was covered in this video.

We have now covered local volatility models. The last section that we will cover in this module is stochastic volatility.

Unit 4 : The Heston Model

Unit 4 : Notes

In this unit, we present the model proposed by Heston (1993). While this wasn't the first stochastic volatility model to be proposed, it was the first to result in a closed-form characteristic function. This means that it can be used, alongside the pricing techniques presented in Module 4, to give analytical prices for vanilla options. The Heston model is extremely important as it is the first model that allowed for stochastic volatility, while still capable of being used for pricing. Note that no closed-form pricing solution exists for even vanilla calls and puts under the Heston model. However, it does result in a closed-form characteristic function, which means that we can use the techniques from the Fourier pricing section, along with some approximations, to estimate these prices.

The Heston model dynamics

The original Heston model assumed that risk-free rates are constant. Note that extensions to the model exist which allow for deterministic and even stochastic interest rates. For simplicity's sake, we will present the constant interest rate model which was presented in Heston's original paper in 1993 (Heston, 1993).

The Heston model assumes that the asset follows the following dynamics:

$$dS_t = \mu S dt + \sqrt{v_t} S dW_t^1,$$

where S_t is the asset value, W_t^1 is a standard Brownian motion, and the volatility term, v_t, follows a CIR, or square-root process:

$$d\sqrt{v_t} = -\beta\sqrt{v_t}dt + \sigma dW_t^2.$$

Now apply Ito's formula to the above volatility process. With some manipulation, this can be written as:

$$dv_t = \kappa(\theta - v_t)dt + \sigma\sqrt{v_t}dW_t^2,$$

where W_t^2 is a standard Brownian motion and has correlation ρ with W_t^1. You should note that the variance process is stochastic, and not the volatility process. By doing this, the volatility will always be positive.

Let $s_t = log(S_t)$. We will now present the characteristic function of s_t. Note that we will be presenting an adaption of the characteristic function proposed by Albrecher *et al.* (2007), as it is numerically more stable than the original proposed by Heston. The characteristic function can be written as:

$$\phi_{s_T} = \exp\big(C(\tau;u) + D(\tau;u)v_t + iulog(S_t)\big), \tag{34}$$

where

$$C(\tau;u) = ri\tau u + \theta\kappa\left[\tau x_- - \frac{1}{a}log\left(\frac{1-ge^{d\tau}}{1-g}\right)\right]$$

$$D(\tau;u) = \left(\frac{1-e^{d\tau}}{1-ge^{d\tau}}\right)x_-, \tag{35}$$

And

$$\tau = T - t,$$
$$g = \frac{x_-}{x_+},$$
$$x_\pm = \frac{b \pm d}{2a}$$
$$d = \sqrt{b^2 - 4ac}$$

$$c = -\frac{u^2 + ui}{2}$$
$$b = \kappa - \rho\sigma iu$$
$$a = \frac{\sigma^2}{2}.$$

(36)

Applying the Heston model in Python

The previous unit gives several equations. The difficulty in implementing the Heston model, given that you are relatively comfortable with characteristic function pricing methods, is in making sure that you code up all the equations correctly. Otherwise, it is a relatively straightforward problem.

Let's go through the code that allows us to estimate the call price on an asset that follows the Heston model dynamics. Using our characteristic function, we just need one more piece of mathematics before we can price a call option.

In the previous module, we applied a change of measure to the measure associated with the asset, S_T. This was relatively simple in the case of geometric Brownian motion because we only had one source of noise to deal with. With the Heston model, we now have two sources of noise, so the mathematics required for a change of measure is a little bit more complex.

Instead, we are going to apply a trick which allows us to convert our characteristic function directly. Let $\phi^S_{s_T}(u)$ be the characteristic function of s_T, where $s_T = log(S_T)$, under the measure associated with S_T. It can be shown that:

$$\phi^S_{s_T}(u) = \frac{\phi_{s_T}(u - i)}{\phi_{s_T}(-i)}.$$

Using this conversion, we can write the price of a European call price as:

$$c = S_0\mathbb{Q}^S[S_T > K] - e^{-rT}K\mathbb{Q}[S_T > K]$$

$$= S_0\left(\frac{1}{2} + \frac{1}{\pi}\int_0^\infty \frac{Im\left[e^{-it\ln K}\varphi_{M_2}(t)\right]}{t}dt\right)$$

$$-e^{-rT}K\left(\frac{1}{2} + \frac{1}{\pi}\int_0^\infty \frac{Im\left[e^{-it\ln K}\varphi_{M_1}(t)\right]}{t}dt\right), \tag{37}$$

where

$$\varphi_{M_1}(t) = \phi_{S_T}(t)$$
$$\varphi_{M_2}(t) = \frac{\phi_{S_T}(u-i)}{\phi_{S_T}(-i)}.$$

Now we just apply a discretization of the integrals such that:

$$\int_0^\infty \frac{Im\left[e^{-it\ln K}\varphi_{M_i}(t)\right]}{t}dt \approx \sum_{n=1}^{N} \frac{Im\left[e^{-it_n\ln K}\varphi_{M_i}(t_n)\right]}{t_n}\Delta t.$$

Now we can apply this in Python. First, we import the relevant libraries:

In [ ]:
```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
```

Then we input our parameter values:

In [ ]:

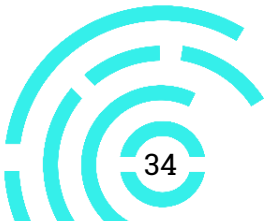
```
#Share specific information
S0 = 100
v0 = 0.06
kappa = 9
theta = 0.06
r = 0.03
sigma = 0.5
rho = -0.4

#Call Option specific information
K = 105
T = 0.5
k_log = np.log(K)

#Approximation information
t_max = 30
N = 100
```

Here $S0$ is S_0, $v0$ is v_0, kappa is κ, theta is θ, r is the risk-free rate, sigma is the volatility term of v_t, rho is ρ, K is the strike, T is the term of the option, `k_log` is the log-strike, `t_max` is the upper bound of our discretized interval for t, and N is the upper bound for the sum.

Now we need to define a few functions so that we can create a function (based on these other functions) for the characteristic function. The functions are:



In [ ]:

```
#Characteristic function code

a = sigma**2/2

def b(u):
    return kappa - rho*sigma*1j*u

def c(u):
    return -(u**2+1j*u)/2

def d(u):
    return np.sqrt(b(u)**2-4*a*c(u))

def xminus(u):
    return (b(u)-d(u))/(2*a)

def xplus(u):
    return (b(u)+d(u))/(2*a)

def g(u):
    return xminus(u)/xplus(u)

def C(u):
    val1 = T*xminus(u)-np.log((1-g(u)*np.exp(-T*d(u)))/(1-g(u)))/a
    return r*T*1j*u + theta*kappa*val1

def D(u):
    val1 = 1-np.exp(-T*d(u))
    val2 = 1-g(u)*np.exp(-T*d(u))
    return (val1/val2)*xminus(u)

def log_char(u):
    return np.exp(C(u) + D(u)*v0 + 1j*u*np.log(S0))

def adj_char(u):
    return log_char(u-1j)/log_char(-1j)
```

The functions are labelled the same as they are in equations (34), (35), and (36). The only variations are for the `log_char` function which is just ϕ_{s_T}, and the `adj_char` function which is just $\frac{\phi_{s_T}(u-i)}{\phi_{s_T}(-i)}$.

Now that we have our functions set up, we can create a few variables that we are going to use so that we can vectorize our code:

In [ ]:

```
delta_t = t_max/N
from_1_to_N = np.linspace(1,N,N)
t_n = (from_1_to_N-1/2)*delta_t
```

Then we can calculate an estimate for the integrals:

In [ ]:
```
first_integral = sum((((np.exp(-1j*t_n*k_log)*adj_char(t_n)).imag)/t_n)*delta_t)
second_integral = sum((((np.exp(-1j*t_n*k_log)*log_char(t_n)).imag)/t_n)*delta_t)
```

And finally, calculate the Fourier estimate of our call price:

In [ ]:
```
fourier_call_val = S0*(1/2 + first_integral/np.pi)-np.exp(-r*T)*K*(1/2 + second_integral/
                   np.pi)
```

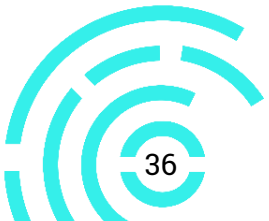
For the sake of comparison, we can also calculate the call price under the Black-Scholes assumptions as:

In [ ]:
```
#Code for analytical solution for vanilla European Call option
d_1_stock = (np.log(S0/K)+(r + sigma**2/2)*(T))/(sigma*np.sqrt(T))
d_2_stock = d_1_stock - sigma*np.sqrt(T)

analytic_callprice = S0*norm.cdf(d_1_stock)-K*np.exp(-r*(T))*norm.cdf(d_2_stock)
```

Now, if you implement the code correctly, you should get a Fourier call price estimate of 5.2773446.

A useful way to illustrate the power of the characteristic pricing method is to plot how quickly the additional terms in the integral approximation go to zero. If we plot the integrands for the two integrals in equation (37), we should get the figure (4).



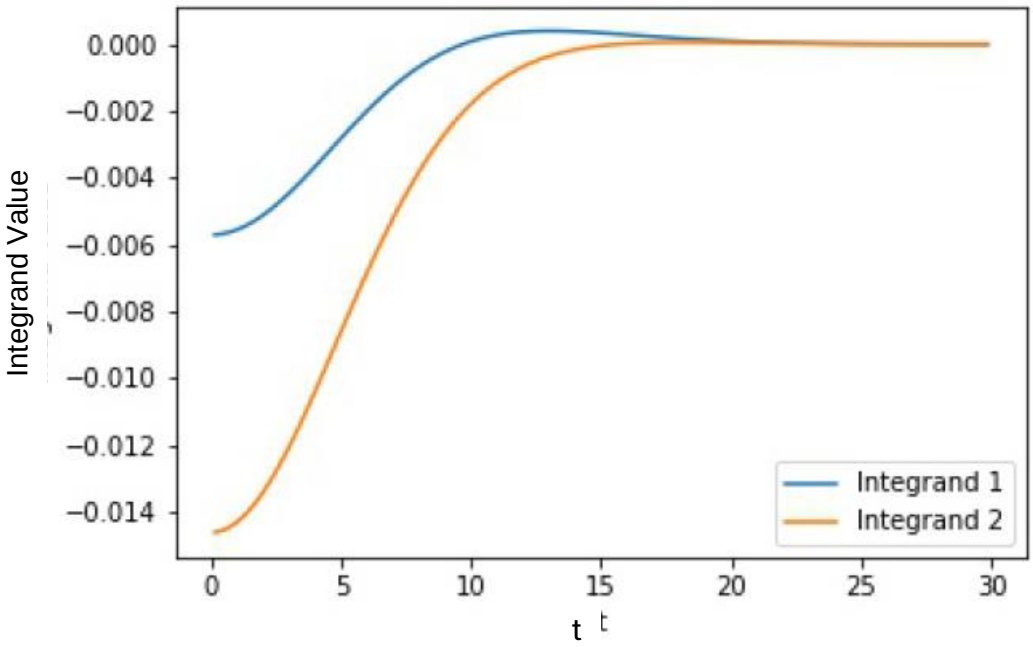


Figure 4: Value of additional terms in integrand estimate

```
plt.title("Value of additional terms in integrand estimate")
plt.plot((((np.exp(-1j*t_n*k_log)*adj_char(t_n)).imag)/t_n)*delta_t,
label="First integrand")
plt.plot((((np.exp(-1j*t_n*k_log)*log_char(t_n)).imag)/t_n)*delta_t,
label="Second integrand")
plt.xlabel("t")
plt.ylabel("Integrand value")
plt.legend()
plt.show()
```

Unit 4: Video Transcript

Hello, and welcome to the final video of Module 5. So far in this module, we have been focusing on local volatility. In this final unit, we will be introducing the Heston model, which models volatility stochastically. Under this model, there is a closed-form characteristic function, which we can use in conjunction with characteristic function pricing methods to price various options.

The Heston model is useful in that it results in what is known as the volatility smile, which we discussed in the first section of this module. The Black-Scholes model is incapable of capturing this characteristic which means that the Heston model allows for us to better capture what happens in markets.

The Heston model dynamics are given as follows:

$$dS_t = \mu S dt + \sqrt{v_t} S dW_t^1,$$

where S_t is the asset value, W_t^1 is a standard Brownian motion, and the volatility term, v_t, follows a CIR, or square-root process:

$$dv_t = \kappa(\theta - v_t)dt + \sigma\sqrt{v_t} dW_t^2,$$

where W_t^2 is a standard Brownian motion and has correlation ρ with W_t^1. You should note that the variance process is stochastic, and not the volatility process. By doing this, the volatility will always be positive.

Heston was able to show that the characteristic function of $\log(S_T)$, s_T, is given by:

$$\phi_{s_T} = \exp\big(C(\tau;u) + D(\tau;u)v_t + iulog(S_t)\big),$$

where

$$C(\tau;u) = ri\tau u + \theta\kappa\left[\tau x_- - \frac{1}{a}log\left(\frac{1-ge^{d\tau}}{1-g}\right)\right]$$

$$D(\tau;u) = \left(\frac{1-e^{d\tau}}{1-ge^{d\tau}}\right)x_-,$$

and

$$g = \frac{x_-}{x_+},$$

$$x_\pm = \frac{b \pm d}{2a}$$

$$d = \sqrt{b^2 - 4ac}$$

$$c = -\frac{u^2 + ui}{2}$$

$$b = \kappa - \rho\sigma iu$$

$$a = \frac{\sigma^2}{2}.$$

The above equation is an adaption of a modification of the model presented by Heston, which was proposed by Albrecher et al. Numerically, this equation is more stable. It is important to note that there have been various extensions to the Heston model by other authors as well. Let's have a look at the Python code to implement these functions.

We first import the libraries we will be using.

In [ ]:
```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
```

Let's suppose we are pricing a vanilla call option with strike \$105 and maturity six months. We assume the stock has an initial value, S_0, of \$100 and the risk-free continuously compounded interest rate, r, is 3%. Further, we assume that S can be modelled using the Heston model, with $v_0 = 0.06, \kappa = 9, \theta = 0.06, \sigma = 0.5$, and $\rho = -0.4$. We are going to use quadrature to estimate the value of the integrals we will be using with a maximum bound of 30 and 100 points of estimation:

In [ ]:
```
#Share specific information
S0 = 100
v0 = 0.06
kappa = 9
theta = 0.06
r = 0.03
sigma = 0.5
rho = -0.4

#Call Option specific information
K = 105
T = 0.5
k_log = np.log(K)

#Approximation information
t_max = 30
N = 100
```

We can then create the functions as defined by Heston for the characteristic function. These are the functions defined earlier in the video and are the crux of implementing the Heston model. The `adj_char` function is the characteristic function associated with the stock after a change in measure. The notes go into more detail about how and why this is done, but we are just going to use it here:

In [ ]:

```
#Characteristic function code

a = sigma**2/2

def b(u):
    return kappa - rho*sigma*1j*u

def c(u):
    return -(u**2+1j*u)/2

def d(u):
    return np.sqrt(b(u)**2-4*a*c(u))

def xminus(u):
    return (b(u)-d(u))/(2*a)

def xplus(u):
    return (b(u)+d(u))/(2*a)

def g(u):
    return xminus(u)/xplus(u)

def C(u):
    val1 = T*xminus(u)-np.log((1-g(u)*np.exp(-T*d(u)))/(1-g(u)))/a
    return r*T*1j*u + theta*kappa*val1

def D(u):
    val1 = 1-np.exp(-T*d(u))
    val2 = 1-g(u)*np.exp(-T*d(u))
    return (val1/val2)*xminus(u)

def log_char(u):
    return np.exp(C(u) + D(u)*v0 + 1j*u*np.log(S0))

def adj_char(u):
    return log_char(u-1j)/log_char(-1j)
```

Recall now, how, using Gil-Palaez, the price of a call is given by:

$$c = S_0\mathbb{Q}^S[S_T > K] - e^{-rT}K\mathbb{Q}[S_T > K]$$

$$= S_0\left(\frac{1}{2} + \frac{1}{\pi}\int_0^\infty \frac{Im\left[e^{-it\ln K}\varphi_{M_2}(t)\right]}{t}dt\right)$$

$$-e^{-rT}K\left(\frac{1}{2} + \frac{1}{\pi}\int_0^\infty \frac{Im\left[e^{-it\ln K}\varphi_{M_1}(t)\right]}{t}dt\right),$$

Where

$$\varphi_{M_1}(t) = \phi_{S_T}(t)$$
$$\varphi_{M_2}(t) = \frac{\phi_{S_T}(u-i)}{\phi_{S_T}(-i)}.$$

Now we just apply a discretization of the integrals such that:

$$\int_0^\infty \frac{Im\left[e^{-it\ln K}\varphi_{M_i}(t)\right]}{t}dt \approx \sum_{n=1}^{N} \frac{Im\left[e^{-it_n\ln K}\varphi_{M_i}(t_n)\right]}{t_n}\Delta t.$$

We then implement this in Python:

In [ ]:
```
delta_t = t_max/N
from_1_to_N = np.linspace(1,N,N)
t_n = (from_1_to_N-1/2)*delta_t

#Integral calculations
first_integral = sum((((np.exp(-1j*t_n*k_log)*adj_char(t_n)).imag)/t_n)*delta_t)
second_integral = sum((((np.exp(-1j*t_n*k_log)*log_char(t_n)).imag)/t_n)*delta_t)

#Call value
fourier_call_val = S0*(1/2 + first_integral/np.pi)-np.exp(-r*T)*K*(1/2 + second_integral/
                np.pi)
```

This gives us a call option price of $5.2773446.

That brings us to the end of the module. In Module 6, we will turn our attention to pricing interest rate options.

Gatheral (2006) explains the popularity of Heston model. The main advantages of this stochastic volatility model are the following:

- It has a quasi-closed form solution for European options as compared to other more realistic stochastic volatility models
- The computations are cheap as compared to models that require expensive Monte Carlo simulations

Gatheral (2006) noticed that all stochastic volatility models provide nearly the same shape of the implied volatility surface.

Bibliography

Albrecher, H., Mayer, P., Schoutens, W. and Tistaert, J. (2007). "The Little Heston Trap", *Wilmott* (1): 83–92.

Bauer, R. (2012). *Fast calibration in the Heston model*. Vienna Institute of Technology.

Breeden, D. T. and Litzenberger, R. H. (1978). "Prices of State-contingent Claims Implicit in Option Prices", *Journal of Business,* pp. 621–651.

Davis, M. H. (2011). "The Dupire Formula", *Finite Difference Methods Course material, Imperial College London.*

Gatheral, J. (2011). *The volatility surface: a practitioner's guide* (Vol. 357). John Wiley & Sons.

Heston, S. L. (1993). "A Closed-form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options", *The Review of Financial Studies* 6(2): 327–343.

Hsu, Y.-L., Lin, T. and Lee, C. (2008). "Constant Elasticity of Variance (cev) Option Pricing Model: Integration and Detailed Derivation", *Mathematics and Computers in Simulation* 79(1): 60–71.

Rouah, F. D. (n.d.). "Heuristic Derivation of the Fokker-Planck Equation", *www. frouah. com* 14: 15, (accessed on 21.09. 2015, at https://frouah.com/finance%20notes/Derivation%20of%20the%20FokkerPlanck%20equation.pdf)
.

Rouah, F. D. (2013). *The Heston Model and Its Extensions in Matlab and C*. John Wiley & Sons.

Appendix

Heston Parameters

Initial variance (v_0)

- This parameter impacts the height of the volatility smile. Higher initial volatility moves the implied volatility upwards.

Long run variance (θ)

- Has a similar impact as the initial variance.

Mean reversion (k)

- Shows the degree of volatility clustering, influencing the curvature of the volatility smile curve.
- Higher mean reversion – flatter volatility smile

Correlation (ρ)

- It shows the connection between log-returns and asset volatility.
- Impacts the skewness of the distribution

Volatility of *variance* (σ)

- Affects the peak (kurtosis) of the distribution
- If σ increases -> heavy tails on both sides

A full description of the topic can be found in Bauer (2012) and Rouah (2013).

The connection between Heston and Black-Scholes

According to Rouah (2013), BS is nested inside Heston model. The conditions to obtain the same price as BS are the following:

$\sigma = 0$ -> volatility becomes deterministic and time-varying

$\theta = v_0$ -> volatility becomes constant as indicated by BS

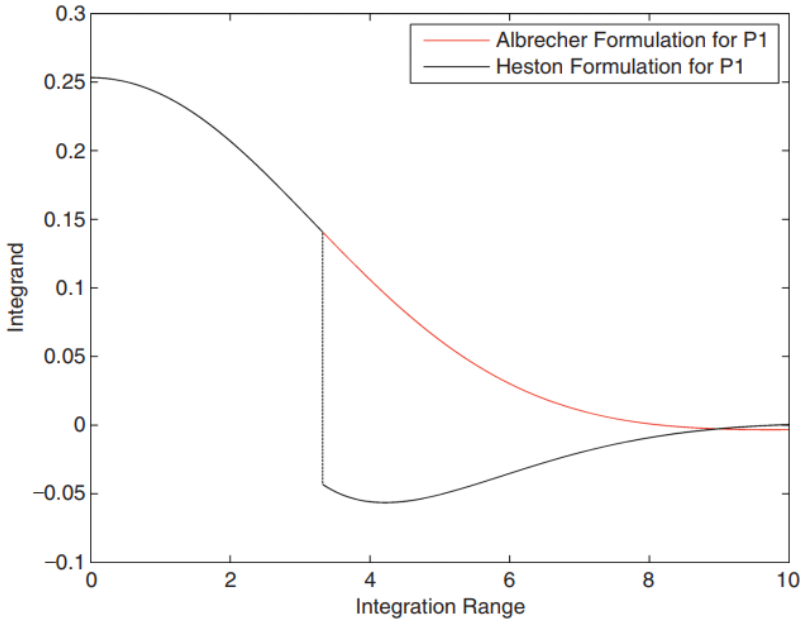
Heston approach provides the same price as BS in this case with $\sigma_{BS} = \sqrt{v_0}$

The Little Heston Trap

Albrecher et al. (2007) and Rouah (2013) show that there are two versions of the Heston characteristic function:

- The original one proposed by Heston (1993)
- A slightly revised one

Albrecher et al. (2007) consider that both approaches are correct, but there are certain numerical problems in the original Heston work. A discontinuity may appear when calculating the first integrand. A solution was provided by Albrecher et al. (2007) by showing a more smoothened approach.



Source: The Heston Model and its Extensions in Matlab and C# - Rouah (2013)

Advantages of the Heston model

- Asset price is not log-normally distributed, an assumption which is closer to market evolution. Takes in consideration high peak and fat tails of the distribution.
- Mean reverting volatility
- It has a semi-closed form solution for European options
- Can be calibrated fast to market data
- Captures the leverage effect

Disadvantages of the Heston model

- Calibrating model parameters is not an easy task
- Results are highly sensitive to model parameters indicated
- Does not perform very well for short-term maturities (inability to create short-term skew as the market one). Mikhailov and Nögel (2003) add jumps in their analysis in order to improve model performance for short-term maturities.