Black-Scholes

In this chapter, we will study the value of European digital and share digital options and standard European puts and calls under the Black-Scholes assumptions. We will also explain how to calculate implied volatilities and the option Greeks. The Black-Scholes assumptions are that the underlying asset pays a constant dividend yield q and has price S satisfying

$$\frac{\mathrm{d}S}{S} = \mu\,\mathrm{d}t + \sigma\,\mathrm{d}B \tag{1}$$

for a Brownian motion B. Here σ is assumed to be constant (though we will allow it to vary in a non-random way at the end of the chapter) and μ can be a quite general random process. It is also assumed that there is a constant continuously-compounded risk-free rate r.

Under these assumptions, we will complete the discussion of **?@sec-s__introoptions** to derive option pricing formulas. Recall that, to price a European call option, all that remains to be done is to calculate the probabilities of the option finishing in the money when we use the risk-free asset and the underlying asset as numeraires. We will do this using the results of **?@sec-s__girsanov**. As in **?@sec-s__introoptions**, we will approach the pricing of call and put options by first considering their basic building blocks: digitals and share digitals.

Digital Options

A digital (or binary) option pays a fixed amount in a certain event and zero otherwise. Consider a digital that pays \$1 at date T if $S(T) > K$, where K is a number that is fixed by the contract. This means that the digital pays x dollars at date T where x is defined as

$$x = \begin{cases} 1 & \text{if } S(T) > K\ , \\ 0 & \text{otherwise}\ . \end{cases}$$

Using the risk-neutral pricing **?@eq-riskneutralformula**, the value of the digital at date~0 is $\mathrm{e}^{-rT}E^R[x]$. Note that

$$\begin{aligned} E^R[x] &= 1 \times \text{prob}^R(x = 1) \ + \ 0 \times \text{prob}^R(x{=}0) \\ &= \text{prob}^R(x = 1) \\ &= \text{prob}^R(S(T) > K)\ . \end{aligned}$$

So we need to calculate this probability of the digital finishing in the money.

In **?@sec-s__girsanov**—see **?@eq-riskneutral11**—we learned that under the Black-Scholes assumption Equation 1 we have

$$\frac{\mathrm{d}S}{S} = (r-q)\,\mathrm{d}t + \sigma\,\mathrm{d}B^* \;,$$

where B^* is a Brownian motion under the risk-neutral measure.[1] on the volatility coefficients and on B and B^* to distinguish the Brownian motion driving S from the Brownian motion driving Y and to distinguish their volatilities are not needed here. In **?@sec-s__geometricbrownianmotion**, we observed that this is equivalent to

$$\mathrm{d}\log S = \left(r - q - \frac{1}{2}\sigma^2\right)\,\mathrm{d}t + \sigma\,\mathrm{d}B^* \;.$$

Now using the formulas **?@eq-tailprob01**–**?@eq-tailprob3**, with $\alpha = r - q - \sigma^2/2$, we have $\text{prob}^R(S(T) > K) = \mathrm{N}(d_2)$ where

$$d_2 = \frac{\log\left(\frac{S(0)}{K}\right) + \left(r - q - \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}} \;. \tag{2}$$

The notation d_2 is standard notation from the Black-Scholes formula, and we use it—rather than a simple d—to distinguish the number Equation 2 from a similar number—to be called d_1 of course—that we will see in the next section.
We conclude:

Tip

The value of a digital option that pays \$1 when $S(T) > K$ is $\mathrm{e}^{-rT}\mathrm{N}(d_2)$, where d_2 is defined in Equation 2.

Consider now a digital that pays when the underlying asset price is low; i.e., consider a security that pays y dollars at date T where

$$y = \begin{cases} 1 & \text{if } S(T) < K \;, \\ 0 & \text{otherwise} \;. \end{cases}$$

Using risk-neutral pricing again, the value of this digital at date 0 is

$$\mathrm{e}^{-rT}E^R[y] = \mathrm{e}^{-rT}\text{prob}^R(y=1) = \mathrm{e}^{-rT}\text{prob}^R(S(T) < K) \;.$$

From this fact and **?@eq-tailprob02**, we conclude:

[1]There is no other risky asset price Y in this model, so the subscripts we used in **?@sec-s__girsanov**

Tip

The value of a digital option that pays \$1 when $S(T) < K$ is $\mathrm{e}^{-rT}\mathrm{N}(-d_2)$, where d_2 is defined in Equation 2.

Share Digitals

Consider a derivative security that pays one share of the underlying asset at date T if $S(T) > K$ and pays zero otherwise. This is called a share digital. As before, let

$$x = \begin{cases} 1 & \text{if } S(T) > K \;, \\ 0 & \text{otherwise} \;. \end{cases}$$

Then the payoff of the share digital at date T is $xS(T)$. Let $Y(t)$ denote the value of this claim for $0 \le t \le T$. We have $Y(T) = xS(T)$ and we want to find $Y(0)$.

From **?@sec-s_reinvestingdividends**, we know that $V(t) = \mathrm{e}^{qt}S(t)$ is the price of a non-dividend-paying portfolio. From our fundamental pricing **?@eq-formula**, using V as the numeraire, we have

$$\begin{aligned} Y(0) &= S(0)E^V\left[\frac{Y(T)}{\mathrm{e}^{qT}S(T)}\right] \\ &= \mathrm{e}^{-qT}S(0)E^V[x] \;. \end{aligned}$$

As in the previous section, $E^V[x] = \text{prob}^V(x = 1)$, so we need to compute this probability of the option finishing in the money.

We follow the same steps as in the previous section. From **?@eq-own11** we have

$$\frac{\mathrm{d}S}{S} = (r - q + \sigma^2)\,\mathrm{d}t + \sigma\,\mathrm{d}B^*,$$

where now B^* denotes a Brownian motion when V is the numeraire. This is equivalent to

$$\mathrm{d}\log S = \left(r - q + \frac{1}{2}\sigma^2\right)\,\mathrm{d}t + \sigma\,\mathrm{d}B^* \;. \tag{3}$$

Thus, from the formulas **?@eq-tailprob01**–**?@eq-tailprob3**, with $\alpha = r - q + \sigma^2/2$, we have

$$\text{prob}^V(S(T) > K) = \mathrm{N}(d_1) \;,$$

where

$$d_1 = \frac{\log\left(\frac{S(0)}{K}\right) + \left(r - q + \frac{1}{2}\sigma^2\right)T}{\sigma\sqrt{T}} \;. \tag{4}$$

This implies:

Tip

The value of a share digital that pays one share when $S(T) > K$ is $\mathrm{e}^{-qT}S(0)\mathrm{N}(d_1)$, where d_1 is defined in Equation 4.

Consider now a share digital that pays one share of the stock at date T if $S(T) < K$. Letting

$$y = \begin{cases} 1 & \text{if } S(T) < K\ , \\ 0 & \text{otherwise}\ , \end{cases}$$

the payoff of this option is $yS(T)$. Its value at date 0 is

$$\begin{aligned} \mathrm{e}^{-qT}S(0)E^V[y] &= \mathrm{e}^{-qT}S(0) \times \mathrm{prob}^V(y=1) \\ &= \mathrm{e}^{-qT}S(0) \times \mathrm{prob}^V(S(T) < K)\ , \end{aligned}$$

and from **?@eq-tailprob02** we have

$$\mathrm{prob}^V(S(T) < K) = \mathrm{N}(-d_1)\ .$$

We conclude:

Tip

The value of a share digital that pays one share when $S(T) < K$ is $\mathrm{e}^{-qT}S(0)\mathrm{N}(-d_1)$, where d_1 is defined in Equation 4.

Puts and Calls

A European call option pays $S(T)-K$ at date T if $S(T) > K$ and 0 otherwise. Again letting

$$x = \begin{cases} 1 & \text{if } S(T) > K\ , \\ 0 & \text{otherwise}\ , \end{cases}$$

the payoff of the call can be written as $xS(T) - xK$. This is equivalent to one share digital minus K digitals, with the digitals paying in the event that $S(T) > K$. The share digital is worth $\mathrm{e}^{-qT}S(0)\mathrm{N}(d_1)$ at date 0 and each digital is worth $\mathrm{e}^{-rT}\mathrm{N}(d_2)$. Note that equations Equation 2 and Equation 4 for d_1 and d_2 imply $d_2 = d_1 - \sigma\sqrt{T}$. Therefore, combining the results of the previous two sections yields the Black-Scholes formula:

Tip

The value of a European call option at date 0 is

$$\mathrm{e}^{-qT} S(0) \mathrm{N}(d_1) - \mathrm{e}^{-rT} K \mathrm{N}(d_2) \;, \tag{5}$$

where d_1 is defined in Equation 4 and $d_2 = d_1 - \sigma\sqrt{T}$.

A European put option pays $K - S(T)$ at date T if $S(T) < K$ and 0 otherwise. As before, let

$$y = \begin{cases} 1 & \text{if } S(T) < K \;, \\ 0 & \text{otherwise} \;. \end{cases}$$

The payoff of the put option is $yK - yS(T)$. This is equivalent to K digitals minus one share digital, all of the digitals paying when $S(T) < K$. Thus, we have:

Tip

The value of a European put option at date 0 is

$$\mathrm{e}^{-rT} K \mathrm{N}(-d_2) - \mathrm{e}^{-qT} S(0) \mathrm{N}(-d_1) \;, \tag{6}$$

where d_1 is defined in Equation 4 and $d_2 = d_1 - \sigma\sqrt{T}$.

Again, this is the Black-Scholes formula.

The values of the European put and call satisfy put-call parity, and we can also find one from the other by^[The put-call parity relation follows from the fact that both the left and the right-hand sides are the prices of portfolios that have value $\max(S(T), K)$ at the maturity of the option. To see this for the left-hand side, note that $\mathrm{e}^{-rT}K$ is sufficient cash to accumulate to K at date T, allowing exercise of the call when it is in the money and retention of the cash K otherwise. For the right-hand side, note that $\mathrm{e}^{-qT}S(0)$ is enough cash to buy e^{-qT} shares of the stock at date 0 which, with reinvestment of dividends, will accumulate to one share at date T, enabling exercise of the put if it is in the money or retention of the share otherwise.}

$$\mathrm{e}^{-rT} K + \text{Call Price} = \mathrm{e}^{-qT} S(0) + \text{Put Price} \;. \tag{7}$$

Greeks

The derivatives (calculus derivatives, not financial derivatives!) of an option pricing formula with respect to the inputs are commonly called Greeks. The most important Greek is the option delta. This measures the sensitivity of the option value to changes in the value of the underlying asset. The following table shows the standard Greeks, with reference to the Black-Scholes pricing formula.

Table 1: Black-Scholes Greeks

Input	**Input Symbol**	**Greek**	**Greek Symbol**
Stock price	S	delta	δ
delta	δ	gamma	Γ
- Time to maturity	$-T$	theta	Θ
Volatility	σ	vega	$\mathcal{V}$
Interest rate	r	rho	ρ

The second line of the above shows δ as an input.[2] Of course, it is not an input but instead is calculated. Gamma, the derivative of δ, is the second derivative of the option price with respect to the underlying asset price. The reason for calculating Θ as the derivative with respect to $-T$ instead of T is that the time-to-maturity T decreasing ($-T$ increasing) is equivalent to time passing, so Θ measures the change in the option value when time passes.

We can calculate these from the Black-Scholes formula using the chain rule from differential calculus. The derivative of the normal distribution function N is the normal density function nd defined as

$$\mathrm{n}d(d) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-d^2/2} \;.$$

One can easily verify directly that

$$\mathrm{e}^{-qT} S \mathrm{n}d(d_1) = \mathrm{e}^{-rT} K \mathrm{n}d(d_2) \;, \tag{8}$$

which simplifies the calculations for the Black-Scholes call option pricing formula. For this formula, the Greeks are as follows:

$$\begin{aligned}
\delta &= \mathrm{e}^{-qT}\mathrm{N}(d_1) + \mathrm{e}^{-qT} S \mathrm{n}d(d_1)\frac{\partial d_1}{\partial S} - \mathrm{e}^{-rT} K \mathrm{n}d(d_2)\frac{\partial d_2}{\partial S} \\
&= \mathrm{e}^{-qT}\mathrm{N}(d_1) + \mathrm{e}^{-qT} S \mathrm{n}d(d_1)\left(\frac{\partial d_1}{\partial S} - \frac{\partial d_2}{\partial S}\right) \\
&= \mathrm{e}^{-qT}\mathrm{N}(d_1) \;, \\
\Gamma &= \mathrm{e}^{-qT}\mathrm{n}d(d_1)\frac{\partial d_1}{\partial S} = \mathrm{e}^{-qT}\mathrm{n}d(d_1)\frac{1}{S\sigma\sqrt{T}} \;,
\end{aligned}$$

[2]The delta is frequently denoted by the upper case Δ, but we will use the lower case, reserving the upper case for discrete changes, e.g., Δt. One may have noticed also that the symbol for vega is a little different from the others; this reflects the fact that vega is not actually a Greek letter.

$$
\begin{aligned}
\Theta &= -\mathrm{e}^{-qT} S \mathrm{n}d(d_1) \frac{\partial d_1}{\partial T} + q\mathrm{e}^{-qT} S\mathrm{N}(d_1) \\
&\quad + \mathrm{e}^{-rT} K \mathrm{n}d(d_2) \frac{\partial d_2}{\partial T} - r\mathrm{e}^{-rT} K\mathrm{N}(d_2) \\
&= \mathrm{e}^{-qT} S \mathrm{n}d(d_1) \left(\frac{\partial d_2}{\partial T} - \frac{\partial d_1}{\partial T} \right) \\
&\quad + q\mathrm{e}^{-qT} S\mathrm{N}(d_1) - r\mathrm{e}^{-rT} K\mathrm{N}(d_2) \\
&= -\mathrm{e}^{-qT} S \mathrm{n}d(d_1) \frac{\sigma}{2\sqrt{T}} + q\mathrm{e}^{-qT} S\mathrm{N}(d_1) - r\mathrm{e}^{-rT} K\mathrm{N}(d_2) \;, \\
V &= \mathrm{e}^{-qT} S \mathrm{n}d(d_1) \frac{\partial d_1}{\partial \sigma} - \mathrm{e}^{-rT} K \mathrm{n}d(d_2) \frac{\partial d_2}{\partial \sigma} \\
&= \mathrm{e}^{-qT} S \mathrm{n}d(d_1) \left(\frac{\partial d_1}{\partial \sigma} - \frac{\partial d_2}{\partial \sigma} \right) \\
&= \mathrm{e}^{-qT} S \mathrm{n}d(d_1) \sqrt{T} \;, \\
\rho &= \mathrm{e}^{-qT} S \mathrm{n}d(d_1) \frac{\partial d_1}{\partial r} - \mathrm{e}^{-rT} K \mathrm{n}d(d_2) \frac{\partial d_2}{\partial r} + T\mathrm{e}^{-rT} K\mathrm{N}(d_2) \\
&= \mathrm{e}^{-qT} S \mathrm{n}d(d_1) \left(\frac{\partial d_1}{\partial r} - \frac{\partial d_2}{\partial r} \right) + T\mathrm{e}^{-rT} K\mathrm{N}(d_2) \\
&= T\mathrm{e}^{-rT} K\mathrm{N}(d_2) \;.
\end{aligned}
$$

We can calculate the Greeks of a European put option from the call option Greeks and put-call parity:

$$\text{Put Price} = \text{Call Price} + \mathrm{e}^{-rT} K - \mathrm{e}^{-qT} S(0) \;.$$

For example, the delta of a put is the delta of a call (with the same strike and maturity) minus e^{-qT}, and the gamma of a put is the same as the gamma of the corresponding call.

Delta Hedging

The ability to create a fully hedged (risk-free) portfolio of the stock and an option is the essence of the arbitrage argument underlying the Black-Scholes formula, as we saw in **?@sec-c__basics** for the binomial model. For a call option, such a portfolio consists of delta shares of the underlying asset and a short call option, or a short position of delta shares of the underlying and a long call option. These portfolios have no instantaneous exposure to the price of the underlying. To create a perfect hedge, the portfolio must be adjusted continuously, because the delta changes when the price of the underlying changes and when time passes. In practice, any hedge will therefore be imperfect, even if the assumptions of the model are satisfied.

We first consider the continuous-time hedging argument. Consider a European call option with maturity T, and let $C(S, t)$ denote the value of the option at date $t < T$ when the stock price is S at date t. Consider a portfolio that is short one call option and long δ shares of the underlying asset and that has a (short) cash position equal to $C - \delta S$. This portfolio has zero value at date t.

The change in the value of the portfolio in an instant $\mathrm{d}t$ is

$$-\mathrm{d}C + \delta\,\mathrm{d}S + q\delta S\,\mathrm{d}t + (C - \delta S)r\,\mathrm{d}t\;. \tag{9}$$

The first term reflects the change in the value of the option, the second term is the capital gain or loss on δ shares of stock, the third term is the dividends received on δ shares of stock, and the fourth term is the interest expense on the short cash position.

On the other hand, we know from Ito's formula that

$$\begin{aligned}\mathrm{d}C &= \frac{\partial C}{\partial S}\,\mathrm{d}S + \frac{\partial C}{\partial t}\,\mathrm{d}t + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}(\mathrm{d}S)^2\\ &= \delta\,\mathrm{d}S + \Theta\,\mathrm{d}t + \frac{1}{2}\Gamma\sigma^2 S^2\,\mathrm{d}t\;. \end{aligned} \tag{10}$$

Substituting Equation 10 into Equation 9 shows that the change in the value of the portfolio is

$$-\Theta\,\mathrm{d}t - \frac{1}{2}\Gamma\sigma^2 S^2\,\mathrm{d}t + q\delta S\,\mathrm{d}t + (C - \delta S)r\,\mathrm{d}t\;. \tag{11}$$

Several aspects of this are noteworthy. First, as noted earlier, the delta hedge (being long δ shares of the underlying) eliminates the exposure to changes in the price of the underlying—there is no $\mathrm{d}S$ term in Equation 11. Second, Θ will be negative, because it captures the time decay in the option value; being short the option means the portfolio will profit from time decay at rate $-\Theta$. Third, this portfolio is short gamma. We can also say it is short convexity, the term convexity referring to the convex shape of the option value as a function of the price of the underlying, which translates mathematically to a positive second derivative (gamma). The volatility in the stock makes convexity valuable, and a portfolio that is short convexity will suffer losses. Finally, the portfolio is earning dividends but paying interest.

It is straightforward to check, from the definitions of Θ, Γ and δ in the preceding section, that the sum of the terms in Equation 11 is zero. The time decay in the option value and dividends received on the shares of the underlying exactly offset the losses due to convexity and interest. Therefore, the delta hedge is a perfect hedge. The portfolio, which has a zero cost, neither earns nor loses money. This is true not only on average but for every possible change in the stock price.

To see how well this works with only discrete adjustments to the hedge, one can simulate the changes in S over time and sum the gains and losses over discrete rebalancing periods. One should input the actual (not risk-neutral) expected rate of return on the asset to compute the actual distribution of gains and losses. This is discussed further in **?@sec-blackscholes_python**.

Gamma Hedging

To attempt to improve the performance of a discretely rebalanced delta hedge, one can use another option to create a portfolio that is both delta and gamma neutral. Being delta neutral means hedged as in the previous section—the portfolio value has no exposure to changes in the underlying asset price. In other words, it means that the derivative of the portfolio value with respect to the price of the underlying (the portfolio delta) is zero. Being gamma neutral means that the delta of the portfolio has no exposure to changes in the underlying price, which is equivalent to the second derivative of the portfolio value with respect to the price of the underlying (the portfolio gamma) being zero. If the delta truly did not change, then there would be no need to rebalance continuously, and hence no hedging error introduced by only adjusting the portfolio at discrete times rather than continuously. However, there is certainly no guarantee that a discretely-rebalanced delta/gamma hedge will perform better than a discretely rebalanced delta hedge.

A delta/gamma hedge can be constructed as follows. Suppose we have written (shorted) a call option and we want to hedge both the delta and gamma using the underlying asset and another option, for example, another call option with a different strike. In practice, one would want to use a liquid option for this purpose, which typically means that the strike of the option will be near the current value of the underlying (i.e., the option used to hedge would be approximately at the money).

Let δ and Γ denote the delta and gamma of the written option and let δ' and Γ' denote the delta and gamma of the option used to hedge. Consider holding a of shares of the stock and b units of the option used to hedge in conjunction with the short option. The delta of the stock is one ($\mathrm{d}S/\mathrm{d}S = 1$), so to obtain a zero portfolio delta we need

$$0 = -\delta + a + b\delta' . \tag{12}$$

The gamma of the stock is zero ($\mathrm{d}^2S/\mathrm{d}S^2 = d\,1/\mathrm{d}S = 0$), so to obtain a zero portfolio gamma we need

$$0 = -\Gamma + b\Gamma' \,. \tag{13}$$

Equation Equation 13 shows that we should hold enough of the second option to neutralize the gamma of the option we have shorted; i.e.,

$$b = \frac{\Gamma}{\Gamma'}$$

Equation Equation 12 shows that we should use the stock to delta hedge the portfolio of options; i.e.,

$$a = \delta - \frac{\Gamma}{\Gamma'}\delta' \,.$$

Implied Volatilities

All of the inputs into the option pricing formulas are in theory observable, except for the volatility coefficient σ. We can estimate σ from historical data (see **?@sec-c_stochasticvolatility**), or estimate it from the prices of other options. The latter method exploits the fact that there is a one-to-one relationship between the price given by the Black-Scholes formula and the σ that is input, so one can take the price as given and infer σ from the formula. The σ computed in this way is called the implied volatility. The implied volatility from one option can be used to price another (perhaps non-traded or less actively traded) option. The calculation of implied volatilities is discussed in **?@sec-blackscholes_implied**.

Even if we acknowledge that the model is not correct, the computation of implied volatilities is still useful for characterizing market prices, because we can quickly describe an option as expensive or cheap depending on whether its implied volatility is large or small. Somewhat paradoxically, it is less easy to see if an option is expensive or cheap by looking at its price, because one must consider the price in the context of the exercise price and maturity. To some extent, the implied volatility normalizes the price relative to the exercise price and maturity. Of course, it does not always pay to sell expensive options or buy cheap options, unless they are expensive or cheap relative to an accurate model!

Term Structure of Volatility

The option pricing formulas in this chapter are derived from the fact that the natural logarithm of the stock price at maturity is normally distributed with a certain mean (depending on the numeraire) and variance equal to $\sigma^2 T$. It is not actually necessary that the volatility be constant. The formulas are still valid if

$$\frac{\mathrm{d}S(t)}{S(t)} = \mu(t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}B(t)$$

where $\sigma(t)$ is some non-random function of time (and again μ can be a quite general random process). In this case, the variance of $\log S(T)$ will be

$$\int_0^T \sigma^2(t)\,\mathrm{d}t\ , \tag{14}$$

which is essentially the sum of the instantaneous variances $\sigma^2(t)\,\mathrm{d}t$. In the d_1's and d_2's in the option pricing formulas, $\sigma^2 T$ should be replaced by Equation 14. A convenient way of expressing this is as follows. Let σ_{avg} be the positive number such that

$$\sigma^2_{\text{avg}} = \frac{1}{T}\int_0^T \sigma^2(t)\,\mathrm{d}t\ . \tag{15}$$

Then we simply need to input σ_{avg} as `sigma` in our option pricing functions. We will call σ_{avg} the average volatility, though note that it is not really the average of $\sigma(t)$ but instead is the square root of the average of $\sigma^2(t)$.

It is important to recognize that, throughout this chapter, date 0 means the date at which the option is being valued. It is not necessarily the date at which the option was first bought or sold. So σ_{avg} is the average (in a sense) volatility during the remaining lifetime of the option, which need not be the same as the average during the option's entire lifetime. It is this remaining volatility that is important for pricing and hedging. Moreover, it is a mistake at date 0 to use $\sigma(0)$ as the volatility to compute prices and hedges. Instead, prices and hedges should be based on σ_{avg}.

These considerations provide a way to address the following situation. If we compute implied volatilities for options with different maturities, we will normally get different numbers. For example, consider two at-the-money options with maturities T_1 and T_2 where $T_2 > T_1$. Denote the implied volatilities by $\hat{\sigma}_1$ and $\hat{\sigma}_2$. We want to interpret these as average volatilities for the time periods $[0, T_1]$ and $[0, T_2]$ respectively. This requires the existence of a function $\sigma(t)$ such that

$$\hat{\sigma}_1^2 = \frac{1}{T_1}\int_0^{T_1} \sigma^2(t)\,\mathrm{d}t \quad \text{and} \quad \hat{\sigma}_2^2 = \frac{1}{T_2}\int_0^{T_2} \sigma^2(t)\,\mathrm{d}t\ .$$

This would imply

$$\hat{\sigma}_2^2 T_2 - \hat{\sigma}_1^2 T_1 = \int_{T_1}^{T_2} \sigma^2(t)\,\mathrm{d}t\ ,$$

which requires

$$\hat{\sigma}_2^2 T_2 - \hat{\sigma}_1^2 T_1 \geq 0\ .$$

Equivalently,

$$\hat{\sigma}_2 \geq \sqrt{\frac{T_1}{T_2}}\hat{\sigma}_1\ .$$

Provided this last inequality is satisfied, we can easily construct the function $\sigma(t)$ as

$$\sigma(t) = \begin{cases} \hat{\sigma}_1 & \text{for } t \leq T_1 \\ \sqrt{\frac{\hat{\sigma}_2^2 T_2 - \hat{\sigma}_1^2 T_1}{T_2 - T_1}} & \text{for } T_1 < t \leq T_2. \end{cases}$$

More generally, given a sequence of at-the-money options with maturities $T_1 < T_2 < \cdots T_N$ and implied volatilities $\hat{\sigma}_1, \ldots, \hat{\sigma}_N$, we define

$$\sigma(t) = \sqrt{\frac{\hat{\sigma}_{i+1}^2 T_{i+1} - \hat{\sigma}_i^2 T_i}{T_{i+1} - T_i}}$$

for $T_i < t \leq T_{i+1}$, provided the expression inside the square root symbol is positive. This $\sigma(t)$ is often called the term structure of (implied) volatilities. Generally, we may expect $\sigma(t)$ to be a decreasing function of time t when the current market is especially volatile and to be an increasing function when the current market is especially quiet.

Smiles and Smirks

If we compute implied volatilities for options with the same maturity but different strikes, we will again obtain different implied volatilities for different options. If we plot implied volatility against the strike, the pattern one normally sees for equities and equity indices is the implied volatility declining as the strike increases until the strike is somewhere near the current value of the underlying (so the option is at the money). The implied volatility will then generally flatten out or increase slightly at higher strikes. The graph looks like a twisted smile (smirk). This pattern has been very pronounced in equity index option prices since the crash of 1987. In contrast to the term structure of implied volatilities, this moneyness structure of implied volatilities is simply inconsistent with the model. It suggests that the risk-neutral return distribution is not lognormal but instead exhibits a higher likelihood of extreme returns than the lognormal distribution (i.e., it has fat tails) with the likelihood of extreme negative returns being higher than the likelihood of extreme positive returns (i.e., it is skewed). We will return to this subject in **?@sec-s_smilesagain**.

Calculations in Python

The following calculates the Black Scholes call, put, call delta, call gamma, and impliied volatility.

```python
import numpy as np
from scipy.stats import norm
import scipy.optimize as optimize

def black_scholes_call(S, K, r, sigma, q, T):
    """
    Inputs:
    S = initial stock price
    K = strike price
    r = risk-free rate
    sigma = volatility
    q = dividend yield
    T = time to maturity
    """
    if sigma == 0:
        return max(0, np.exp(-q * T) * S - np.exp(-r * T) * K)
    else:
        d1 = (np.log(S / K) + (r - q + 0.5 * sigma ** 2) * T) / (sigma * np.sqrt(T))
        d2 = d1 - sigma * np.sqrt(T)
        N1 = norm.cdf(d1)
        N2 = norm.cdf(d2)
        return np.exp(-q * T) * S * N1 - np.exp(-r * T) * K * N2

def black_scholes_put(S, K, r, sigma, q, T):
    """
    Inputs:
    S = initial stock price
    K = strike price
    r = risk-free rate
    sigma = volatility
    q = dividend yield
    T = time to maturity
    """
    if sigma == 0:
        return max(0, np.exp(-r * T) * K - np.exp(-q * T) * S)
    else:
        d1 = (np.log(S / K) + (r - q + 0.5 * sigma ** 2) * T) / (sigma * np.sqrt(T))
        d2 = d1 - sigma * np.sqrt(T)
        N1 = norm.cdf(-d1)
        N2 = norm.cdf(-d2)
        return np.exp(-r * T) * K * N2 - np.exp(-q * T) * S * N1

def black_scholes_call_delta(S, K, r, sigma, q, T):
    """
    Inputs:
    S = initial stock price
    K = strike price
    r = risk-free rate
    sigma = volatility
    q = dividend yield
    T = time to maturity
    """
    d1 = (np.log(S / K) + (r - q + 0.5 * sigma ** 2) * T) / (sigma * np.sqrt(T))
    return np.exp(-q * T) * norm.cdf(d1)

def black_scholes_call_gamma(S, K, r, sigma, q, T):
    """
    Inputs:
    S = initial stock price
    K = strike price
    r = risk-free rate
    sigma = volatility
    q = dividend yield
    T = time to maturity
    """
    d1 = (np.log(S / K) + (r - q + 0.5 * sigma ** 2) * T) / (sigma * np.sqrt(T))
    nd1 = np.exp(-d1 ** 2 / 2) / np.sqrt(2 * np.pi)
    return np.exp(-q * T) * nd1 / (S * sigma * np.sqrt(T))

def black_scholes_call_implied_vol(S, K, r, q, T, CallPrice):
    """
    Inputs:
    S = initial stock price
    K = strike price
    r = risk-free rate
    q = dividend yield
    T = time to maturity
    CallPrice = call price
    """
    def objective(sigma):
        return black_scholes_call(S, K, r, sigma, q, T) - CallPrice

    if CallPrice < np.exp(-q * T) * S - np.exp(-r * T) * K:
        raise ValueError("Option price violates the arbitrage bound.")

    tol = 1e-6
    lower = 0
    upper = 1
    fupper = objective(upper)

    while fupper < 0:
        upper *= 2
        fupper = objective(upper)

    implied_vol = optimize.bisect(objective, lower, upper, xtol=tol)
    return implied_vol


# Example usage (you can replace these with input values)
S = 100  # Initial stock price
K = 100  # Strike price
r = 0.05  # Risk-free rate
sigma = 0.2  # Volatility
q = 0.02  # Dividend yield
T = 1  # Time to maturity in years
```

```python
CallPrice = 10  # Call price for implied volatility calculation

# Calculate Black-Scholes call and put prices
call_price = black_scholes_call(S, K, r, sigma, q, T)
put_price = black_scholes_put(S, K, r, sigma, q, T)
print(f"Call Price: {call_price}, Put Price: {put_price}")

# Calculate Delta and Gamma for the call option
call_delta = black_scholes_call_delta(S, K, r, sigma, q, T)
call_gamma = black_scholes_call_gamma(S, K, r, sigma, q, T)
print(f"Call Delta: {call_delta}, Call Gamma: {call_gamma}")

# Calculate implied volatility for a given call price
implied_vol = black_scholes_call_implied_vol(S, K, r, q, T, CallPrice)
print(f"Implied Volatility: {implied_vol}")
```

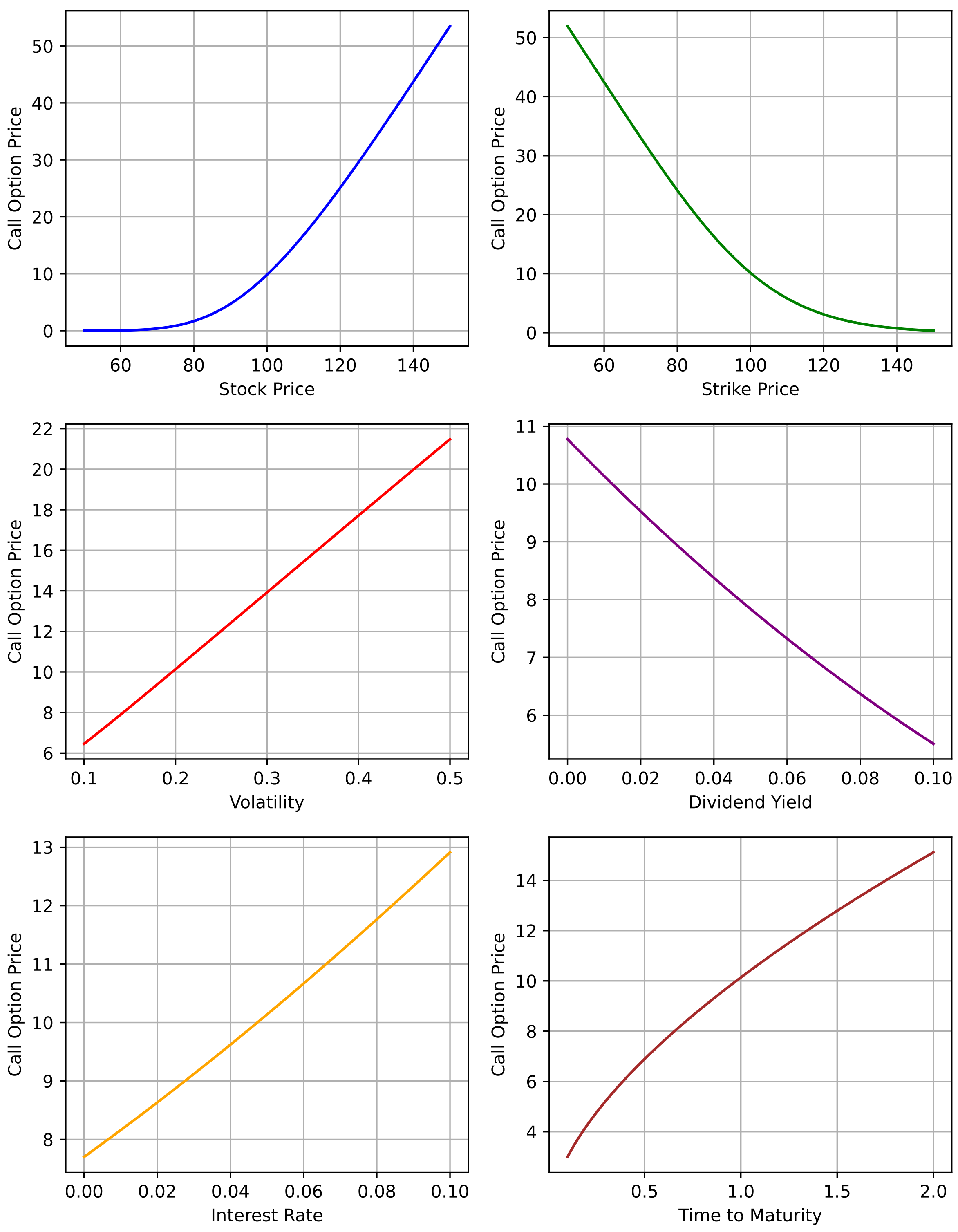
```
Call Price: 9.227005508154036, Put Price: 6.330080627549918
Call Delta: 0.586851146134764, Call Gamma: 0.018950578755008718
Implied Volatility: 0.2203836441040039
```

The following plot the Black Scholes call price against the stock price, the strike price, volatility, dividend yield, interest rate, and time to maturity.

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

black_scholes_call(S, K, r, sigma, q, T)

# Parameters
S = np.linspace(50, 150, 100)
K = 100
T = 1
r = 0.05
sigma = 0.2
q = 0.01

# Plot Black-Scholes call price against various parameters
fig, axs = plt.subplots(3, 2, figsize=(8, 10))

# Plot against stock price
```

```
call_prices_S = [black_scholes_call(s, K, r, sigma, q, T) for s in S]
axs[0, 0].plot(S, call_prices_S, label='Call Option Price', color='blue')
axs[0, 0].set_title('')
axs[0, 0].set_xlabel('Stock Price')
axs[0, 0].set_ylabel('Call Option Price')
axs[0, 0].grid(True)

# Plot against strike price
strike_prices = np.linspace(50, 150, 100)
call_prices_K = [black_scholes_call(S[50],k, r, sigma, q, T) for k in strike_prices]
axs[0, 1].plot(strike_prices, call_prices_K, label='Call Option Price', color='green')
axs[0, 1].set_title('')
axs[0, 1].set_xlabel('Strike Price')
axs[0, 1].set_ylabel('Call Option Price')
axs[0, 1].grid(True)

# Plot against volatility
volatilities = np.linspace(0.1, 0.5, 100)
call_prices_sigma = [black_scholes_call(S[50],K, r, sigma, q, T) for sigma in volatilities]
axs[1, 0].plot(volatilities, call_prices_sigma, label='Call Option Price', color='red')
axs[1, 0].set_title('')
axs[1, 0].set_xlabel('Volatility')
axs[1, 0].set_ylabel('Call Option Price')
axs[1, 0].grid(True)

# Plot against dividend yield
dividend_yields = np.linspace(0, 0.1, 100)
call_prices_q = [black_scholes_call(S[50],K, r, sigma, q, T) for q in dividend_yields]
axs[1, 1].plot(dividend_yields, call_prices_q, label='Call Option Price', color='purple')
axs[1, 1].set_title('')
axs[1, 1].set_xlabel('Dividend Yield')
axs[1, 1].set_ylabel('Call Option Price')
axs[1, 1].grid(True)

# Plot against interest rate
interest_rates = np.linspace(0, 0.1, 100)
call_prices_r = [black_scholes_call(S[50],K, r, sigma, q, T) for r in interest_rates]
axs[2, 0].plot(interest_rates, call_prices_r, label='Call Option Price', color='orange')
axs[2, 0].set_title('')
axs[2, 0].set_xlabel('Interest Rate')
axs[2, 0].set_ylabel('Call Option Price')
axs[2, 0].grid(True)
```

```python
# Plot against time to maturity
times_to_maturity = np.linspace(0.1, 2, 100)
call_prices_T = [black_scholes_call(S[50],K, r, sigma, q, T) for T in times_to_maturity]
axs[2, 1].plot(times_to_maturity, call_prices_T, label='Call Option Price', color='brown')
axs[2, 1].set_title('')
axs[2, 1].set_xlabel('Time to Maturity')
axs[2, 1].set_ylabel('Call Option Price')
axs[2, 1].grid(True)

plt.tight_layout()
plt.show()
```



Call Option Price
Stock Price
Strike Price
Volatility
Dividend Yield
Interest Rate
Time to Maturity

Example: Replicating Portfolios and Simulating Portfolio Insurance

Another derivation of the Black Scholes formula is provided by Merton. He asked the question whether by trading the stock and the risk free asset whether the payoff to a European call option can be replicated. Let θ_t be the number of shares of the stock held at time t and α_t the number of shares of an initial investment of one dollar in the risk free asset. Then the portfolio is worth $\alpha_t R_t + \theta_t S_t$ where $R_t = e^{rt}$ is the time t value of an initial time 0 investment of one dollar in the risk free asset. The portfolio should start with an initial value, should not have any cash inflows or outflows and have a terminal value equal to a call payoff so the changes in value are completely dictated by the changes in the value of the assets. That is, assuming continuous trading,

$$dW_t = \theta_t dS_t + \alpha_t dR_t = \theta_t \left(\mu S_t dt + qS_t dt + \sigma S_t dB_t\right) + \alpha_t r R_t dt$$

with terminal condition

$$W_T = \alpha_T R_T + \theta_T S_T = (S_T - K)^+$$

The problem is to find θ_t and α_t for all times and states. If we can accomplish this, then by 'no-arbitrage' the call price must be the value of the initial investment. Assume the call price is a function of the stock price and time: $C(t, S_t)$. Then by Ito's Lemma

$$dC(t,S_t) = \left(\frac{\partial C}{\partial t} + \frac{\partial C}{\partial S}(\mu - q)S_t + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S_t^2\right)dt + \frac{\partial C}{\partial S}\sigma S_t dB_t$$

It should be apparent that we want to hold $\theta = \frac{\partial C}{\partial S}$, which is the delta of the call option. By doing so, we match the diffusion term in thw change in wealth and the change in the call option. Then matching the drift terms in both expressions

$$\frac{\partial C}{\partial S}\mu S_t + \alpha_t r R_t = \frac{\partial C}{\partial t} + \frac{\partial C}{\partial S}(\mu - q)S_t + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S_t^2$$

which can be solved to give

$$\alpha_t r R_t = r\left(W_t - \frac{\partial C}{\partial S}S_t\right) = \frac{\partial C}{\partial t} - \frac{\partial C}{\partial S}qS_t + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S_t^2$$

which gives the equation

$$rW_t = \frac{\partial C}{\partial t} + \frac{\partial C}{\partial S}(r - q)S_t + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S_t^2$$

with a boundary condition $W_T = (S_T - K)^+$. However, no-arbitrage suggests $W_t = C(t, S_t)$ which gives us the partial differential equation

$$rC = \frac{\partial C}{\partial t} + \frac{\partial C}{\partial S}(r - q)S + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S^2$$

with a boundary condition $C(T, S_T) = (S_T - K)^+$. This is a partial differential equation and a fairly tedious set of calcuations show the Black Scholes formula is a solution (in fact it is the only positive solution). Close observation of the right hand side we see this is the drift term of Ito expansion for C if we work in the risk neutral measure. The right hand side then says in the risk neutral measure, the call option earns the risk free return.

However, there is nothing special about a call option. The same argument will apply for any European style option. The only difference is the boundary condition. This procedure allows us to replicate the payoff of any European option even for those which might not be traded. This observation had a profound effect on practice. A particularly popular example is portfolio insurance.

Recall, that a protective put position buys a put and buys a share and the payoff at the expiration of the put is given by $\max(K, S_T)$. The reason for the name protective put is apparent since the position can pay off no less than K. The cost of this insurance is the price of the put. However, if the put is not traded, we can synthetically replicate this payoff using the prodedure above assuming we can trade continuously. The basic recipe is to start with inital wealth equal to that for a protective put position: $W_0 = P(0, S_0) + S_0$. The delta of the protective put position can be calcuated to be the delta of the put plus 1 which is $N(d_1)$, where d_1 is calculated at each point in time. However, in practice we cannot trade continuously. A simple discrete strategy would rebalance at intervals Δt. The strategy calculates $N(d_1)$ at time 0 and holds $P(0, S_0) + S_0 - N(d_1)S_0$ dollars in the risk free asset and $N(d_1)$ shares of the asset. Thereafter these holdings are adjusted. The change in portfolio value over the interval Δt is

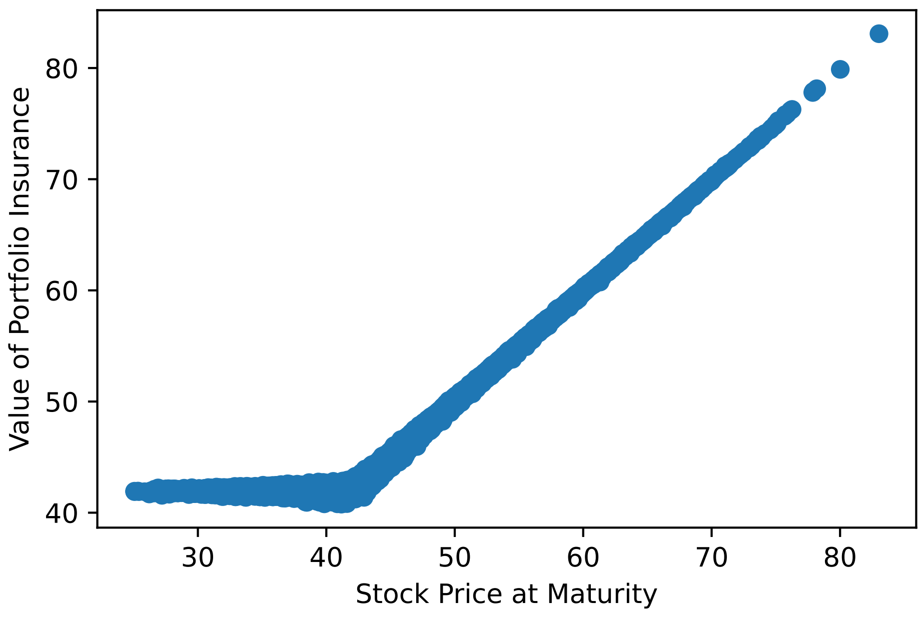
$$\Delta W = W_{i\Delta t} - W_{(i-1)\Delta t}$$

$$= (P((i-1)\Delta t, S_{(i-1)\Delta t}) + S_{(i-1)\Delta t} - N(d_1-)S_{(i-1)\Delta t})(R_{i\Delta t} - R_{(i-1)\Delta t}) + N(d1-)(S_{i\Delta t} - S_{(i-1)\Delta t})$$

where $N(d1-)$ is the delta chosen at time $(i-1)\Delta t$. The question is if the Black Scoles model is correct, how accurate can a discrete rebalancing scheme be? This is simulated in the following code:

```python
import numpy as np
# from bsfunctions import *
import matplotlib.pyplot as plt
import time
from math import pow, exp, sqrt
from scipy import stats
# incs = np.genfromtxt('incs.csv',delimiter=",",skip_header=1)
def blackscholes(S0, K, r, q, sig, T, call = True):
    '''Calculate option price using B-S formula.

    Args:
        S0 (num): initial price of underlying asset.
        K (num): strick price.
        r (num): risk free rate.
        q (num): dividend yield
        sig (num): Black-Scholes volatility.
        T (num): maturity.
        call (bool): True returns call price, False returns put price.

    Returns:
        num
    '''
    d1 = (np.log(S0/K) + (r -q + sig**2/2) * T)/(sig*np.sqrt(T))
    d2 = d1 - sig*np.sqrt(T)
#     norm = sp.stats.norm
    norm = stats.norm
    if call:
        return np.exp(-q*T)*S0 * norm.cdf(d1,0,1) - K * np.exp(-r * T) * norm.cdf(d2,0, 1)
    else:
        return np.exp(-q*T)*S0 * -norm.cdf(-d1,0,1) + K * np.exp(-r * T) * norm.cdf(-d2,0, 1

def blackscholes_delta(S0, K, r, q, sig, T, call = True):
    '''Calculate option price using B-S formula.

    Args:
        S0 (num): initial price of underlying asset.
        K (num): strick price.
        r (num): risk free rate.
        q (num): dividend yield
        sig (num): Black-Scholes volatility.
        T (num): maturity.
        call (bool): True returns call price, False returns put price.

    Returns:
        num
    '''
    d1 = (np.log(S0/K) + (r -q + sig**2/2) * T)/(sig*np.sqrt(T))
    d2 = d1 - sig*np.sqrt(T)
#     norm = sp.stats.norm
    norm = stats.norm
    if type(call) == bool:
        if call:
            return np.exp(-q*T)*norm.cdf(d1,0,1)
        else:
            return np.exp(-q*T)*norm.cdf(-d1,0,1)
    else:
        print("Not a valid value for call")

# parameters
# number of paths
# n = incs.shape[1]
n = 100000
# number of divisions
# m = incs.shape[0]
m = 100
# interest rate
r = .1
# dividend yield
q=0.0
# true drift
mu = .15
# volatility
sig = .2
# Initial Stock Price
S0 = 42
# Strike Price
K = 42
# Maturity
T = 0.5


# seed for random generator
seed= 1234
# define a random generator
rg = np.random.RandomState(seed)
# initialize


# generate normal random vairables
dt= T/m
vol=sig*np.sqrt(dt)
incs = rg.normal(0,vol,[m,n])


tline = np.linspace(0,T,m+1)
```

```
St = np.zeros((m+1,n))
#St1 = np.zeros((m+1,n))

V_vec = np.zeros((m+1,n))

delta = np.zeros((m,n))

put= blackscholes(S0,K,r, q, sig,T,call=False)

incs_cumsum =  np.concatenate((np.zeros((1,n)),incs),axis=0).cumsum(axis=0)
V_vec = np.zeros((m+1,n))
t_mat =  np.repeat(tline.reshape((m+1,1)), n, axis=1)
drift_cumsum = (mu -q -0.5*sig**2) * t_mat

St = S0 * np.exp(incs_cumsum + drift_cumsum)

delta = blackscholes_delta(St[:-1,:],K,r, q, sig,T-t_mat[:-1,:])

V_vec[0,:] = S0 + put

for i in range(1,m+1):
    V_vec[i,:] = V_vec[i-1,:] + (np.exp(r*dt)-1) * (V_vec[i-1,:] - delta[i-1,:] * St[i-1,:])

# Uses actual simulated changes in riskfree and stock price not the dt and dB approximations
# plot ST versus VT
plt.scatter(St[m,:],V_vec[m,:])
plt.xlabel('Stock Price at Maturity')
plt.ylabel('Value of Portfolio Insurance')
plt.show()
```



With $m = 100$ rebalancing dates over $T = 0.5$ for the parameters chosen the repliacting strategy does a pretty good job. The hedging errors occur when the stock price is close to the strike price. This is not surprising since the delta changes (measured by the gamma) fastest around this point. A gamma hedge would potentially improve the performance.

The portfolio insurance rebalancing scheme involves sell stock and buying bonds when the stock price goes down and buying stocks and selling bonds when the stock price goes up. This can be destabilizing and was identified as a contributor to the 1987 stock market crash.

Discretely-Rebalanced Delta Hedges

To compute the real-world distribution of gains and losses from a discretely-rebalanced delta hedge, we input the expected rate of return μ. We consider adjusting the hedge at dates $0 = t_0 < t_1 < \cdots < t_N = T$, with $t_i - t_{i-1} = \Delta t = T/N$ for each i. The changes in the natural logarithm of the stock price between successive dates t_{i-1} and t_i are simulated as

$$\Delta \log S = \left(\mu - q - \frac{1}{2}\sigma^2\right) \Delta t + \sigma \, \Delta B \, ,$$

where ΔB is normally distributed with mean zero and variance Δt. The random variables ΔB are simulated as standard normals multiplied by $\sqrt{\Delta t}$. We begin with the portfolio that is short a call, long δ shares of the underlying, and short $\delta S - C$ in cash. After the stock price changes, say from S to S', we compute the new delta δ'. The cash flow from adjusting the hedge is $(\delta - \delta')S'$. Accumulation (or payment) of interest on the cash position is captured by the factor $e^{r\Delta t}$. Continuous payment of dividends is modelled similarly: the dividends earned during the period Δt is taken to be $\delta S\left(e^{q\Delta t}-1\right)$. The cash position is adjusted due to interest, dividends, and the cash flow from adjusting the hedge. At date T, the value of the portfolio is the cash position less the intrinsic value of the option.

To describe the distribution of gains and losses, we compute percentiles of the distribution. You should see that the hedge becomes more nearly perfect as the number of periods N is increased. Note that this is true regardless of the μ that is input, which reaffirms the point that option values and hedges do not depend on the expected rate of return of the underlying. The percentile is calculated with the Excel `Percentile` function.[3]

```python
import numpy as np
from scipy.stats import norm
import scipy.optimize as optimize

def simulated_delta_hedge_profit(S0, K, r, sigma, q, T, mu, M, N, pct):
    """
    Inputs:
    S0 = initial stock price
    K = strike price
    r = risk-free rate
    sigma = volatility
    q = dividend yield
    T = time to maturity
    mu = expected rate of return
    N = number of time periods
    M = number of simulations
    pct = percentile to be returned
    """
    dt = T / N
    SigSqrdt = sigma * np.sqrt(dt)
    drift = (mu - q - 0.5 * sigma ** 2) * dt
    Comp = np.exp(r * dt)
    Div = np.exp(q * dt) - 1
    LogS0 = np.log(S0)
    Call0 = black_scholes_call(S0, K, r, sigma, q, T)
    Delta0 = black_scholes_call_delta(S0, K, r, sigma, q, T)
    Cash0 = Call0 - Delta0 * S0
    Profit = np.zeros(M)
```

[3]If numsims = 11 and pct =0.1, the percentile function returns the second lowest element in the series. The logic is that 10% of the numbers, excluding the number returned, are below the number returned—i.e., 1 out of the other 10 are below—and 90% of the others are above. In particular, if pct = 0.5, the percentile function returns the median. When necessary, the function interpolates; for example, if numsims = 10 and pct=0.1, then the number returned is an interpolation between the lowest and second lowest numbers.

```python
    for i in range(M):
        LogS = LogS0
        Cash = Cash0
        S = S0
        Delta = Delta0

        for j in range(1, N):
            LogS += drift + SigSqrdt * np.random.randn()
            NewS = np.exp(LogS)
            NewDelta = black_scholes_call_delta(NewS, K, r, sigma, q, T - j * dt)
            Cash = Comp * Cash + Delta * S * Div - (NewDelta - Delta) * NewS
            S = NewS
            Delta = NewDelta

        LogS += drift + SigSqrdt * np.random.randn()
        NewS = np.exp(LogS)
        HedgeValue = Comp * Cash + Delta * S * Div + Delta * NewS
        Profit[i] = HedgeValue - max(NewS - K, 0)

    return np.percentile(Profit, pct * 100)

# Example usage (you can replace these with input values)
S = 100  # Initial stock price
K = 100  # Strike price
r = 0.05  # Risk-free rate
sigma = 0.2  # Volatility
q = 0.02  # Dividend yield
T = 1  # Time to maturity in years
CallPrice = 10  # Call price for implied volatility calculation

# Simulate delta hedging profit
S0 = 100  # Initial stock price
mu = 0.1  # Expected rate of return
M = 1000  # Number of simulations
N = 252  # Number of time periods
pct = 0.95  # Percentile to be returned

delta_hedge_profit = simulated_delta_hedge_profit(S0, K, r, sigma, q, T, mu, M, N, pct)
print(f"Delta Hedge Profit (95th percentile): {delta_hedge_profit}")
```

```
Delta Hedge Profit (95th percentile): 0.6721233748183916
```

Exercises

Exercise 0.1. Create a Python code which inputs K, r, σ, q and T. Compute the delta of a call option for stock prices $S = .01K$, $.02K$, ..., $1.99K$, $2K$ (i.e., $S = iK/100$ for $i = 1, \ldots 200$) and plot the delta against the stock price.

Exercise 0.2. The delta of a digital option that pays \$1 when $S(T) > K$ is

$$\frac{\mathrm{e}^{-rT}\mathrm{n}d(d_2)}{\sigma S\sqrt{T}}\;.$$

Repeat the previous problem for the delta of this digital. Given that in reality it is costly to trade (due to commissions, the bid-ask spread and possible adverse price impacts for large trades), do you see any problems with delta hedging a short digital near maturity if it is close to being at the money?

Exercise 0.3. Modify the Python code for replicating portfolio insurance to simulate a discrete replication of a digital option using the delta in the previous problem. Run the code for $10, 20, 100, 1000$ rebalancing dates. When does the strategy do a good job and when does it fail?

Exercise 0.4. Repeat Exercise 0.1 for the gamma of a call option.

Exercise 0.5. Consider delta and gamma hedging a short call option, using the underlying and a put with the same strike and maturity as the call. Calculate the position in the underlying and the put that you should take, using the analysis in Section . Will you ever need to adjust this hedge? Relate your result to put-call parity.

Exercise 0.6. The delta of a share digital that pays one share when $S(T) > K$ is

$$\mathrm{e}^{-qT}\mathrm{N}(d_1) + \frac{\mathrm{e}^{-qT}\mathrm{n}d(d_1)}{\sigma\sqrt{T}}\;.$$

Repeat Exercise 0.1 for the delta of this share digital.

Exercise 0.7. Compute the value of an at-the-money call option ($S = K$) using the Python code for volatilities $\sigma = .01, .02, \ldots, 1.0$. Plot the call value against the volatility.

Exercise 0.8. Repeat the previous problem for $S = 1.2K$ (an example of an in-the-money call option).

Exercise 0.9. The file CBOEQuotes.txt (available at `www.kerryback.net`) contains price data for call options on the S&P 500 index. The options expired in February, 2003, and the prices were obtained on January 22, 2003. The first column lists various exercise prices. The second column gives the bid price and the third column the ask price. Import this data into an Excel worksheet and compute and plot the implied volatility against the exercise price using this data. Use the ask price as the market price for the option. The options have 30 days to maturity (so $T = 30/365$). At the time the quotes were downloaded, the S&P 500 was at 884.25. According to the CBOE, the dividend yield on the S&P 500 was 1.76%. Use 1.25% for the risk-free interest rate.

Exercise 0.10. Attempt to repeat the previous problem using the bid price as the market price of the option. If this doesn't work, what is wrong? Does this indicate there is an arbitrage opportunity?

::: Suppose an investor invests in a portfolio with price S and constant dividend yield q. Assume the investor is charged a constant expense ratio α (which acts as a negative dividend) and at date T receives either his portfolio value or his initial investment, whichever is higher. This is similar to a popular type of variable annuity. Letting D denote the number of dollars invested in the contract, the contract pays

$$\max\left(D, \frac{D\mathrm{e}^{(q-\alpha)T}S(T)}{S(0)}\right) \tag{16}$$

at date T.
We can rearrange the expression Equation 16 as

$$\begin{aligned}\max\left(D, \frac{D\mathrm{e}^{(q-\alpha)T}S(T)}{S(0)}\right) &= D + \max\left(0, \frac{D\mathrm{e}^{(q-\alpha)T}S(T)}{S(0)} - D\right) \\ &= D + \mathrm{e}^{-\alpha T}D\max\left(0, \frac{\mathrm{e}^{qT}S(T)}{S(0)} - \mathrm{e}^{\alpha T}\right) \,. \end{aligned} \tag{17}$$

Thus, the contract payoff is equivalent to the amount invested plus a certain number of call options written on the gross holding period return $\mathrm{e}^{qT}S(T)/S(0)$. Note that $Z(t) = \mathrm{e}^{qt}S(t)/S(0)$ is the date–t value of the portfolio that starts with $1/S(0)$ units of the asset (i.e., with a \$1 investment) and reinvests dividends. Thus, the call options are call options on a non-dividend paying portfolio with the same volatility as S and initial price of \$1. This implies that the date–0 value of the contract to the investor is $\mathrm{e}^{-rT}D$ plus

$e^{-\alpha*T} * D*$`Black_Scholes_Call`$(1, e^{-\alpha*T}, r, sigma, q, T)$

1. Create a Python function to compute the fair expense ratio; i.e., find α such that the date–0 value of the contract is equal to D. Hint: Modify the

```
Black_Scholes_Call_Implied_Vol
```

function. You can use $\alpha = 0$ as a lower bound. Because the value of the contract is decreasing as α increases, you can find an upper bound by iterating until the value of the contract is less than D. 2. How does the fair expense ratio vary with the maturity T? Why?

::: ::: {#exr-nolabel} Modify the function `Simulated_Delta_Hedge_Profit` to compute percentiles of gains and losses for an investor who writes a call option and constructs a delta and gamma hedge using the underlying asset and another call option. Include the exercise price of the call option used to hedge as an input, and assume it has the same time to maturity as the option that is written. Hint: In each period `j = 1 to N-1`, the updated cash position can be calculated as

```
Cash = exp(r*dt)*Cash + a*S*(exp(q*dt)-1) - (Newa-a)*NewS _
     - (Newb-b)*PriceHedge ,
```

where `a` denotes the number of shares of the stock held, `b` denotes the number of units held of the option that is used for hedging, and `PriceHedge` denotes the price of the option used for hedging (computed from the Black-Scholes formula each period). This expression embodies the interest earned (paid) on the cash position, the dividends received on the shares of stock and the cash inflows (outflows) from adjusting the hedge. At the final date `N`, the value of the hedge is

```
exp(r*dt)*Cash + a*S*(exp(q*dt)-1) + a*NewS _
     + b*Application.Max(NewS-KHedge,0) ,
```

and the value of the overall portfolio is the value of the hedge less

```
Application.Max(NewS-KWritten,0) ,
```

where `KHedge` denotes the strike price of the option used to hedge and `KWritten` denotes the strike of the option that was written. :::