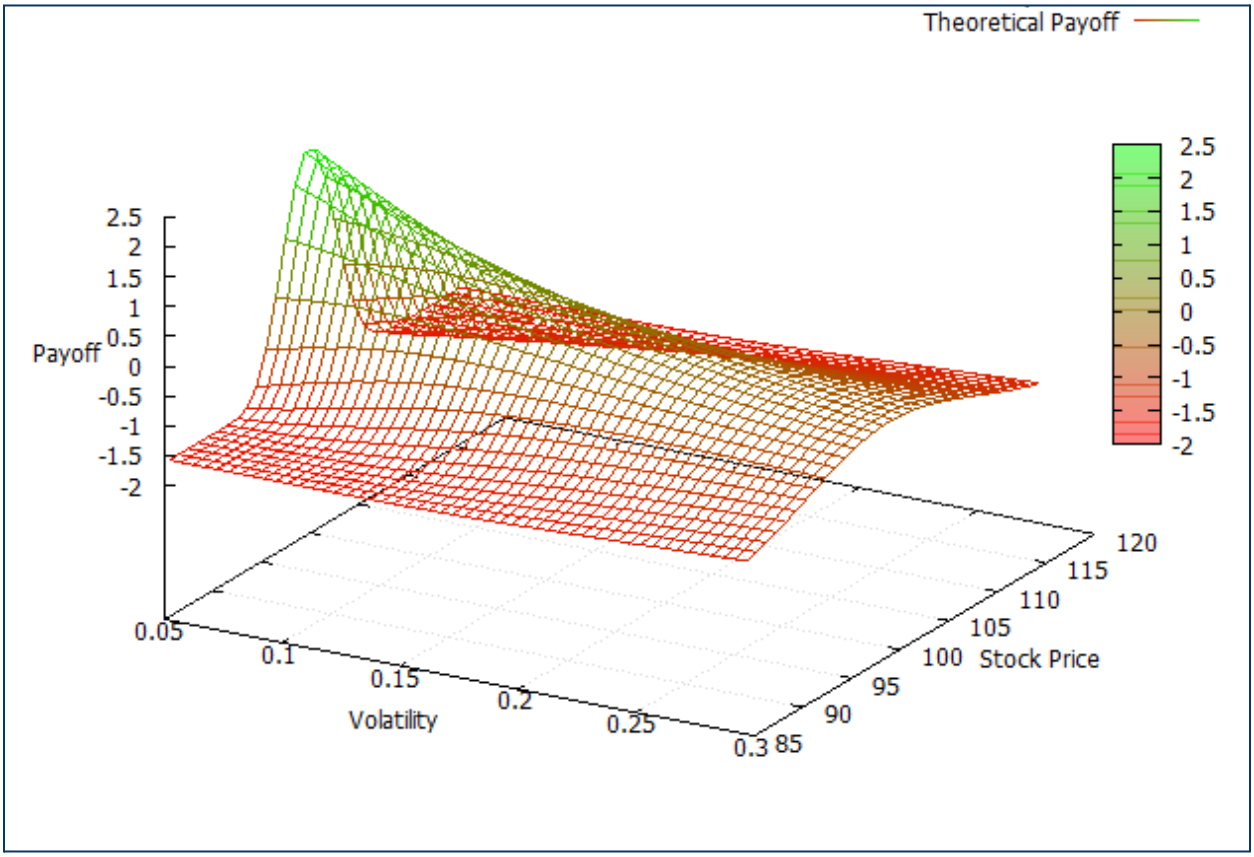
RESEARCH

2023 October

Analyzing Option & Underlying Positions Quantitatively w/ C++



Communication enquiries should be directed to the following contact:
Matas Urbonavicius
+370 6 196 2244
matas.urbonavicius@gmail.com
linkedin.com/in/matasurbonavicius/

Disclaimer: This paper is written for informational and educational purposes only. While every effort has been made to ensure the accuracy and reliability of the information presented, the author accepts no responsibility for errors, omissions, or any consequences arising from the use of the information herein. Readers are encouraged to consult primary sources and seek expert advice before drawing any conclusions based on this work.

Analyzing Option Positions And Constructing A Program in C++

Abstract

Author will show how to create a code in C++ when making an options position analyzer which includes various payoff charts, greeks. Additionally, the author will dive into information on the option strategies and when they work, what to look for, what is the right strategy.

Before you go further

It is assumed that the reader is relatively knowledgeable about financial markets and is able to read mathematical formulas, code in C++. If not, This paper may be more difficult to read as the author does not explain everything in great detail and concentrates into the intuition part. Having said that, if you follow along, I am sure you will get there.

Table of Contents

Important Aspects of the Option Positions

Theoretical Value vs Implied Value: Since options are publicly traded instruments, value will often differ to what we perceive as "correct". But how do we perceive the value in the first place? For example valuing stocks is relatively straightforward - check if the product of the company is good, look through financial statements, sentiment analysis, market charting and so forth. You got me there - maybe not too straightforward, however, it is easy to understand the logic behind the valuation of stocks. But what about options?

Options, having different maturities and strike prices for each underlying can seem intimidating at first. Then there is a natural thought "well, how do I value this thing?". Gladly for us, we don't have to invent the bicycle - though, we must assemble it. In 1973, a model Black-Scholes was invented which is still used today. The model is a relatively simple formula where we simply need to plug in values and it will spit out the price of an option. As a matter of fact, the Black-Scholes model will be the backbone of this paper.

$$C = SN(d_1) - Ke^{-rt}N(d_2)$$

where:

$$d_1 = \frac{ln\frac{S}{K} + (r + \frac{\sigma_v^2}{2})t}{\sigma_s\sqrt{t}}$$

and

$$d_2 = d_1 - \sigma_s\sqrt{t}$$

and where:

C = Call option price

S = Current stock (or other underlying) price

K = Strike price

r = Risk-free interest rate

t = Time to maturity

N = A normal distribution

Fig. 1 - Black-Scholes Formula[1]

The next logical step is to put this formula into the code. Note that while this article is used to construct the program in C++, the logic will hold throughout all programming languages. Code of this formula will look like the one, given below:

```
double callPrice() const {
      return S * N(d1) - K * exp(-r * T) *
N(d2);
}
```

[1]https://www.investopedia.com/terms/b/blackscholes.asp

Where:

```
void computeD1D2() {
     d1 = (log(S / K) + (r + 0.5 * sigma *
sigma) * T) / (sigma * sqrt(T));
    d2 = d1 - sigma * sqrt(T);
}
```

Our theoretical value will be the price from the model. We will come back to the exact function later in the article. What about the Implied value? Well, this is simply the Bid/Ask price of the option- its premium. The main difference that we will often need to address is the volatility component of the model. For the purposes of this paper, I will take volatility as a long term average of the standard deviation. For SPX, this is 18%.

Using the implied value, we can backwards-calculate the volatility which should be plugged into the model to get this price. This is called Implied Volatility. Looking into any options chain currently trading, we will see that the implied volatility is different for different strike prices but a constant maturity.

This is called a volatility smile (or a volatility skew) and is shown in the Fig. 2 = Volatility smile. The phenomenon is interesting, as the market essentially is saying that annualized volatility differs by the length of the option, while we know it is not true! If both options expire in 1 month, their realized annualized volatility will be the same!

Speculation about the reasons for the volatility smile is beyond this paper, however, keep in mind that volatility, being the most important component of the option, is also the most difficult to estimate!

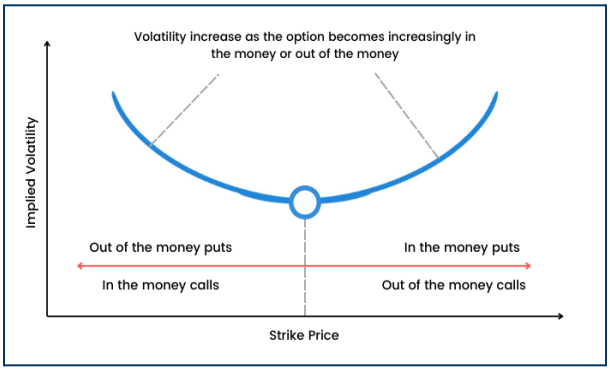


Fig. 2 - Volatility Smile[2]

That means the implied volatility is set by the market as it is known, that market often over- or under-prices assets due to the nature of psychology and other effects happening on a continuous basis. Hence, we can sometimes assume that our theoretical volatility (the long-term average) will be more correct than the one set by the market and as a matter of fact, market over-prices options the absolute majority of the time. The effect is clear- insurance is not cheap. Effect is shown below:

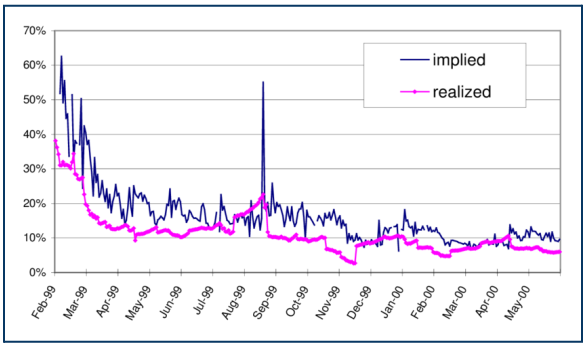


Fig. 3 - Implied vs Realized Volatility[3]

The conclusion can be drawn clearly: By selling volatility we can expect to make money, based on historical data. As we will see later on, volatility is by far the most important assumption

[2]https://www.elearnmarkets.com/school/units/option-greeks-1/volatility-smile
[3]https://www.researchgate.net/figure/displays-in-percent-per-annum-the-time-variation-of-implied-volatility-and-realized_fig1_4906338

we make when calculating the price for the option.

Time value: Options can be thought of as insurance contracts. The longer the duration of the insurance- the more you pay! However, the bigger chances to catch a movement which will put the current option contract in the profit. Our position analyzer should be able to correctly calculate the effect of time passage, given that underlying conditions remain the same.

Payoff Chart: Lastly, the payoff chart of the option is probably the most important visual we can have when analyzing the position. It shows the profit/loss at expiration. By the payoff chart, we can immediately spot the strategy used and where we want the price to go. Problem is, however, that a payoff chart is at a given point of time, which is at expiration. If we want to see the price (PnL) of the position at the other times, another calculation method is needed.

The Greeks

So far we have slightly covered the volatility and the time passage, but there is clearly more to it! The so-called "Greeks" describe several key characteristics about the options. A small summary is shown in Fig. 4.

Name	Dependent Variable	Independent Variable
Delta	Option price	Value of underlying asset
Gamma	Delta	Value of underlying asset
Vega	Option price	Volatility
Theta	Option price	Time to maturity
Rho	Option price	Interest rate

Fig. 4 - Greeks[4]

The Greeks are some of the most important measures used when describing how option behaves. They are widely used for risk management or construction of positions, for which reason we will now dive deeper into their calculation and meaning.

Delta

Measures how much the option will move in comparison with the underlying. If Delta = 0.2, then for every $move of the underlying, the option will move $0.2. Delta can also be viewed as a probability of option being in-the-money by the expiration. In our case- roughly 20%. Mathematically, formula is expressed as:

$$\Delta = \frac{\partial V}{\partial S}$$

Fig. 5 - Delta Mathematical[5]

Where ∂ is the first derivative, V is the theoretical option's price (so Black Scholes Formula output) and S being the underlying asset's price. However, practically, Delta can be calculated as the Normal Distribution value of d1. Code snippet below:

```
double callDelta() const {
    return N(d1);
}
```

Where:

```
void computeD1() {
     d1 = (log(S / K) + (r + 0.5 * sigma     *
sigma) * T) / (sigma * sqrt(T));
}

// Cumulative Distrib Function
double N(double x) const {
    return 0.5 * erfc(-x * sqrt(0.5));
}
```

Since computing the CDF using its regular formula is difficult, we use an approximation method called error function, commonly denoted as erfc..

[4]https://corporatefinanceinstitute.com/resources/derivatives/option-greeks/

[5]https://corporatefinanceinstitute.com/resources/derivatives/option-greeks/

This is because the d1 component of the Black-Scholes model can be thought of as a normalized measure of the distance between the current stock price S and the present value of the strike price K. The closer S is to the K, d1 approaches 0.5 which is actually a major assumption in the Black-Scholes model- market is efficient and it is not known in advance the direction that market will take, so at current price there is a 50/50 chance of market going either direction.

Gamma

Measures the change of Delta for every $move. It is defined as a rate of change or as a first derivative. Mathematically it is expressed as:

$$\Gamma = \frac{\partial \Delta}{\partial S} = \frac{\partial^2 V}{\partial^2 S}$$

Fig. 6 - Gamma Mathematical[6]

Which shows that it is the second derivative of V/S, meaning the first derivative of Delta.Good thing for us is that the calculation of Gamma is the same for Put and Call options. The function D1 I have shown above represents CDF- an area under the curve to the left of x. For the Delta we want to calculate a PDF which tells the height of the curve at the point x. Then divide it by variation for a small time increment to get the derivative. PDF can be calculated with the function below:

```
double n(double x) const {
        return (1.0 / sqrt(2.0 * PI)) *
exp(-0.5 * x * x);
}
```

Which is derived from the general normal distribution formula:

$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Fig. 7 - PDF formula[7]

With this in mind, we can calculate the Gamma:

```
double gamma() const {
    return (n(d1) / (S * sigma * sqrt(T)));
}
```

Note, that S * sigma * sqrt(T) simply represents how sensitive stock price is to change, hence by dividing it from PDF we get sensitivity of Delta - Gamma.

Theta

Measures how the passage of time will influence position. Is expressed as a change in options value per 1 day. Long positions will generally have a negative Theta while short positions will have a positive one. Mathematically it is expressed as:

$$\theta = -\frac{\partial V}{\partial \tau}$$

Fig. 8 - Theta Mathematical[8]

And it is described as the negative rate of change of the option price with respect to the option's time to maturity. Practically, calculation is this formula:

$$\theta = -\frac{S\phi\,(d1)\sigma}{2\sqrt{t}} - rKe^{-rt}N\,(d2)$$

Fig. 9 - Theta Practical[9]

[6]https://corporatefinanceinstitute.com/resources/derivatives/option-greeks/

[7]https://www.cuemath.com/data/probability-density-function/

[8]https://corporatefinanceinstitute.com/resources/derivatives/option-greeks/

[9]https://www.iotafinance.com/en/Formula-Theta-of-a-Call-Option.html

Which can be simply implemented into the code as a function:

```
double callTheta() const {
      return ((-S * sigma * n(d1) / (2 *
sqrt(T))) - (r * K * exp(-r * T) * N(d2)));
}
```

Vega

Measures how the change of volatility will impact position. Is expressed as a $change for every 1% of volatility increase/decrease. A positive Vega means option will gain value as volatility increases. Mathematically it can be expressed like this:

$$\nu = \frac{\partial V}{\partial \sigma}$$

Fig. 10 - Vega Mathematical[10]

It is described as a rate of change given change of volatility. Practically, formula looks like this:

$$\nu = S\phi\,(d1)\sqrt{t}$$

Fig. 11 - Vega Practical[11]

Plugging this formula into the code and wrapping a function around it, we get a resulting code:

```
double optionVega() const {
    return (S * sqrt(T) * n(d1));
}
```

Rho

Measures how the change of interest rates will impact the position. Is expressed as a $change in position given 1% change in the risk free rate (interest rate). It's mathematical formula is given as:

$$\rho = \frac{\partial V}{\partial r}$$

Fig. 12 - Rho Mathematical[12]

It is expressed as a rate of change in options price with respect to the change in the risk free rate. Practical formula is shown below:

$$\rho = Kte^{-rt}N\,(d2)$$

Fig. 13 - Rho Practical[13]

And thus can be implemented into the code easily as shown below:

```
double callRho() const {
    return (K * T * exp(-r * T) * N(d2));
}
```

Integrating all code into one

In object oriented programming we can encapsulate all of our option calculation logic into one module which is ideal for a task like the option price, greeks calculation.

In this paper's example where the author is using C++ language- this would mean putting all logic into a class. We can then declare the key variables and add calculations of key formulas as private members. Functions to calculate the price and greeks can be used as public members.

[10]https://corporatefinanceinstitute.com/resources/derivatives/option-greeks/
[11]https://www.iotafinance.com/en/Formula-Vega-of-an-option.html
[12]https://corporatefinanceinstitute.com/resources/derivatives/option-greeks/
[13]https://www.iotafinance.com/en/Formula-Rho-of-a-call-option.html

The structure would look like this:

```
Class BlackScholes{
private:
        void compute D1D2;
        double N(); // CDF
        double n()  // PDF
public:
        double callPrice();
        putPrice();
        callDelta();
        //…etc.
}
```

Once everything is put together, we can compute the price & the greeks with the function call shown below:

```
double T = daysTillMaturity / 365.0;

BlackScholes option(stockPrice, strikePrice, riskFreeRate, T, annualizedVolatility);

callPrice = option.callPrice()
```

To test it, we can take a look into a SPY options chain in Yahoo and put some values to get the theoretical price of the option. We can then compare it with the implied price.

424.00	9.28	9.24	9.27	+0.88	+10.48%	81	664	18.40%
425.00	8.82	8.64	8.67	+1.00	+12.79%	154	1,885	18.16%
426.00	7.95	8.25	8.28	+0.72	+9.96%	65	877	18.31%
427.00	7.44	7.50	7.52	+0.90	+13.76%	3	815	17.69%
427.50	7.11	7.25	7.28	+0.87	+13.94%	103	728	17.64%
428.00	7.07	6.97	7.00	+1.08	+18.03%	29	1,169	17.51%
429.00	6.57	6.48	6.51	+1.13	+20.77%	13	652	17.35%
430.00	6.04	5.90	5.92	+0.73	+13.75%	125	3,160	16.94%

Fig. 14 - Current SPY Option Chain. Columns = Strike; Last, Bid, Ask, Change, Change%, Volume, Open Interest, Implied Volatility [14]

[14]https://finance.yahoo.com/quote/SPY/options?date=1700784000

Plugging these values into the C++ program, we get these numbers:

```
|-------------------------------------------------------------------|
|                        -- Global Inputs --                        |
| How many different options do you have? 1
| What is the risk free rate? (if 1%, then enter 0.01) - 0.05
| What is the current underlying price? - 424.24
| How much of underlying do you own? - 0

| Details for Option 1:
| What is the option? (1 for call & 0 for put) - 1
| What is the quantity? (Negative for sold) - 1
| What is the current price for the option? - 7
| What is the strike price? - 428
| How many days till maturity? - 30
| What is the annualized volatility? (if 10%, then enter 0.1) - 0.175
|-------------------------------------------------------------------|

|-------------------------------------------------------------------|
| -- Option 1 --
|
| The current call option price is: $7.00
| The theoretical Black Scholes call option price is: $7.55
| Delta for the call option is: 0.47
| Gamma for the option is: 0.02
| Theta for the call option is: $-0.17 per day
| Vega for the call option is: 0.48 per 1% volatility change
| Rho for the call option is: 15.86 per 1% interest rate change
|-------------------------------------------------------------------|
```

Fig. 15 - Black Scholes C++ UI

The current call option price is $7 whereas the theoretical price is $7.55, meaning that in theory, this option is currently underpriced. We would look to the strategies that play on the increase in volatility, to capture this difference.

Additionally, our greeks formulas calculate these values:

```
|-------------------------------------------------------------------|
| -- Position Greeks --
|
| Position Delta: 0.47
| Position Gamma: 0.02
| Position Theta -0.61
| Position Vega: 0.48
| Position Rho: 0.16
|...Generating Charts
|-------------------------------------------------------------------|
```

Fig. 16 - Black Scholes C++ UI

Everything looks as expected. Since the option is close to the current value, Delta is close to 0.5. Gamma shows that for every SPY $move, we will have an increase of Delta by 0.02. Theta is quite strongly negative- as expected. Option is bought and insurance costs to hold! Vega tells us that the increase in volatility by 1% will give $0.48 of value to our option, while an increase in interest rates by 1% will give the option $0.16 of value..

Creating a Payoff Chart

A payoff chart is simply an options profit or loss at the day of it's expiration, given some price. Let's say we pay $5 for a call option, strike price 100. Anything below 100 and the option will expire worthless, hence our loss of $5. Anything above that and the option will grow at a rate equal that of the underlying. The chart will look something like this, shown in Fig. 17:

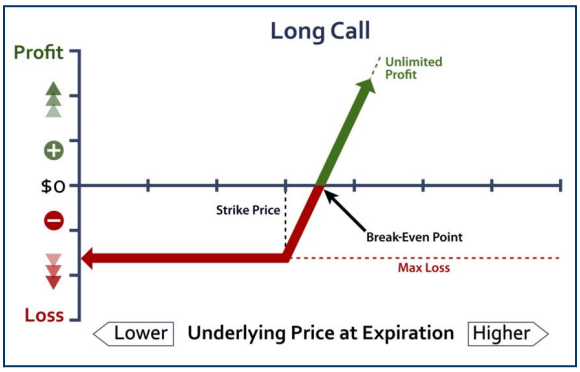


Fig. 17 - Call payoff chart[15]

So in essence, the payoff chart is simply the option's intrinsic value - premium. To account for the possibility of buy vs selling options, we can add a direction variable which is negative when we sell. With this in mind, we can loop through the prices and for each price, calculate the payoff with this formula:

```
If (option.type == "Call") {
        payoff += quantity * max<double>(0,
        (stockPrice - strikePrice)) -
        option.callPrice())
}
```

Notice the max function. This is because our maximum loss is always the premium paid. The similar will be for the put option. Next step is to construct two for loops: one for each price, one for each currently held option; then sum the payoff for each price and we will get the payoff chart. Using GnuPlot software we can create simple looking charts. See Fig. 18 below:

[15]https://www.optionsbro.com/long-call-option-strategy-example/

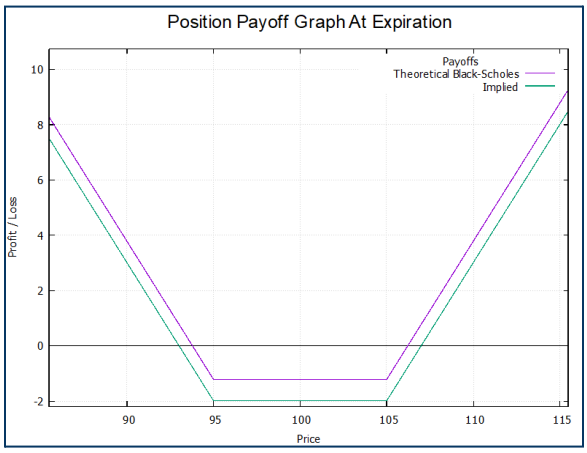


Fig. 18 - Strangle payoff chart in GnuPlot

This is a strangle position which means buying one call and one put. Traders enter strangle positions when they think a large move is going to appear, however they don't know in which direction. Notice the increasing "legs" to both directions

Payoff charts are a great way to visualize a strategy and what we are looking for from the price to do. But how do they look at different dates? For example, today. What is the payoff chart for the option position today? Or in two weeks?

The formula will be conceptually different, as we are not holding to maturity and exercising. The intrinsic value component will not be there, so all that is left is the options price! If we have a position and we want to sell it- we just need to pay the market price! Code will look like this:

```
If (option.type == "Call") {
        payoff += quantity * -
        option.callPrice())
}
```

Combining a couple of different dates and an expiration calculation, the chart looks like the one below, given a strangle:

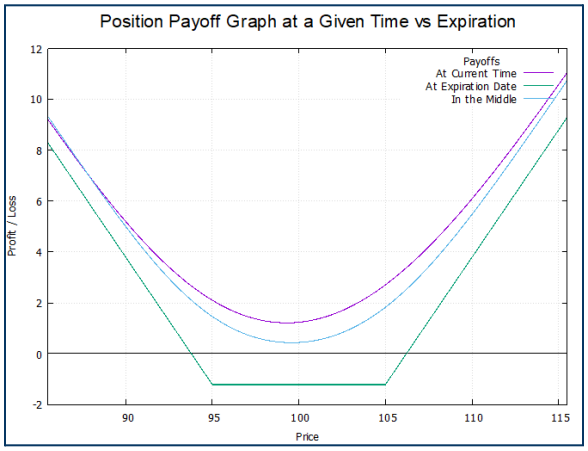


Fig. 19 - Strangle payoff at different dates

The chart essentially demonstrates the price to obtain a position, or.. The price to sell it at a given time.

To conclude, the payoff charts are a great tool to visualize a position. They are especially useful because options can have non-linear payoffs, which can sometimes be counterintuitive.

Creating a Payoff Chart for Time Spreads

A calendar spread is buying 1 call and selling 1 call that are the same strike price, but different maturities. This is done because the closer to maturity, the sooner option loses value, so the goal with time spreads is to "catch" that movement with the option we sold. If we try to graph the payoff chart we will simply get a straight line. Time spreads need a different way of calculation.

A time spread is calculated by looking at the expiry date of the shorter expiration contract and graphing the payoff. However, one option must be calculated using the "at-maturity" formula, while the other is "What we paid - what the market offers now" while the other is the payoff at-expiry formula which we already covered.

What we paid is premium, that we paid at the position establishment and what the market offers now can only be calculated by plugging the time left to expiration at the time of the short option expiry. Let's take a look into the overall logic of this code:

```
double T = optDetail.daysTillMaturity / 365.0;
double T_ = *smallest_expiry;

BlackScholes option_bought(stockPrice,
strikePrice, riskFreeRate, T, volatility);

BlackScholes option_at_expiry(stockPrice,
strikePrice, riskFreeRate, T_, volatility);

if (daysTillMaturity / 365 == T_) {
        if (optDetail.type == 'Call") {
                PnL_Position += |At_Exp_Calc|
        }
}
Else {
        if (optDetail.type == 'Call") {
                PnL_Positions += |Paid-Bought|
```

For visual and cleanness reasons I have left out parts of the code, however the general idea is this- we split the calculations and then sum them up. A typical time spread will look like the one provided in Fig. 20 below:

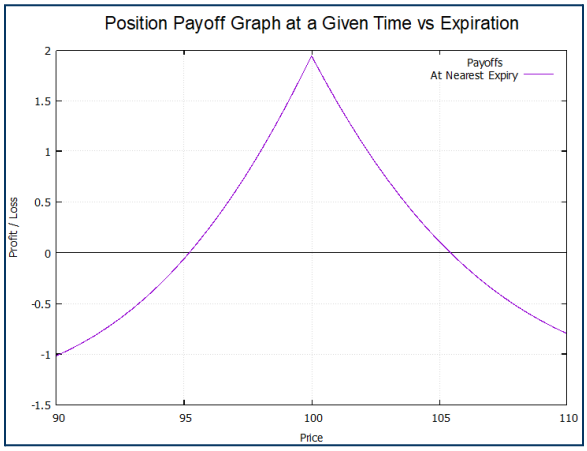


Fig. 20 - Call Calendar Spread

Note, that this payoff chart does not make sense for other positions, therefore it is best to add an if statement which will skip this chart if there is only one maturity date.

Change in PnL Given Different Volatility

Since volatility is one of the most important components of Black-Scholes model, it would be wise to visualize how the position will do for different given volatilities. What if we have 10 different options and a sudden sharp jump will appear? How quickly will our PnL change? One way to visualize that is to look into Vega; however, the volatility function is nonlinear, therefore Vega is only a good approximation at near intervals.

To chart it, we can adjust the for loop to go through volatility instead of prices. But how about the price component then? The answer is- take one price, for example strike price and given that price, run through the model for each volatility value, to construct a chart. A long call option volatility chart, given its strike price, will look like the Fig. 21 shown below:

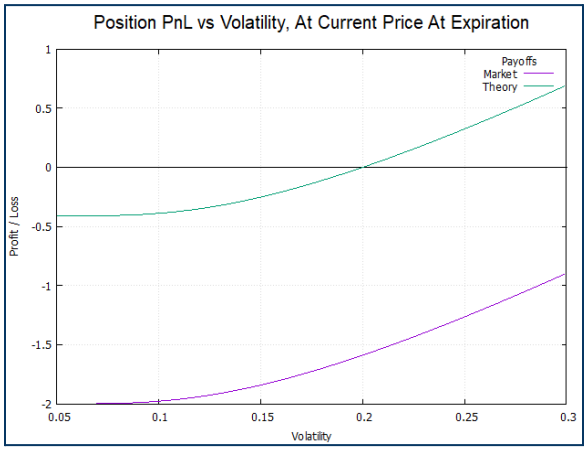


Fig. 21 - Call Option Payoff at Strike K Given Volatilities

It is clearly visible that by increasing the volatility, options are valued more. By 5% of volatility increase, slope is around 0.5 of increase in Profit / Loss.

3D Option Payout Surface - Price vs Volatility

The last chart of this paper will be the surface chart. We have so far covered the payout when we iterate through the price. We have just covered the graph when we are iterating through volatility. A clear next step is to merge those two to see how the payout will change for each price at expiration and also for each volatility. Chart is displayed below, as Fig. 22.

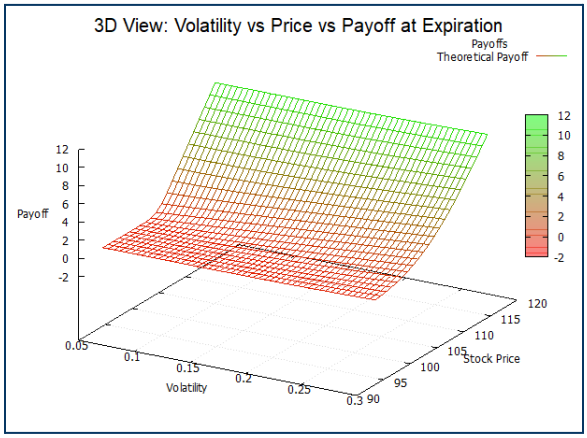


Fig. 22 - Call Option Surface

Notice, how the payout shape nears the regular option payout 2D graph when volatility goes closer to 0. The reason is because when volatility is near 0, it means the stock price is not expected to change much from its current value. This is equivalent to a scenario where the expiration of the option is very near, and there's very little time for the stock to move. Hence, the expected value of the call option will resemble its intrinsic value, which is the value if exercised immediately.

In contrast, as volatility increases, there's greater uncertainty about where the stock price will be at expiration. This results in a wider range of potential payoffs, represented by the more extensive surface area in the plot. However, since this is a chart for the payoff at expiration, the effects of volatility on option pricing (like the time value) aren't reflected directly in the surface shape, but rather in the possible range of stock prices and associated payoffs.

Adding the Underlying

Sometimes, we will have a position that includes the underlying itself. An example: a covered call. A covered call is buying the underlying and selling a call for the same amount. In the coding environment, this is actually easy to do. To each of our payoffs, we need to add the amount of underlying we have times the current price. A snippet of the code is shown below:

```
// Addition of the underlying position
+ (underlying_qty * (stockPrice_iteration - stockPrice_bought))
```

Now, if we have an underlying position and we have sold an option against it (a Covered call), the payout chart will look like this:

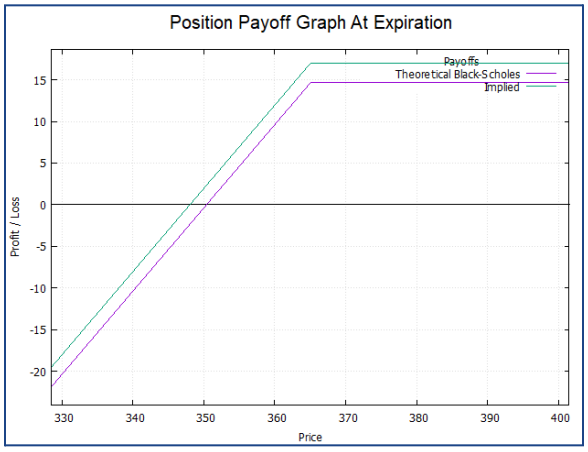


Fig. 23 - Covered Call Payoff Graph

To conclude, once the program is done and running, we can happily conclude that we have a tool to analyze option positions. We will clearly see what to expect from each position.

Characteristics of Volatility Spreads

We can create the most popular option combinations and make a small analysis of the Greeks. Note, that when creating a position, we can only determine the initial Greeks. They, however, will generally continue to have the same direction throughout the options life. A positive Gamma means a large move will generally help the position. Negative Theta will mean that passage of time will hurt the position, while a positive Vega will illustrate that position will be helped by an increase in implied volatility. See the table below:

Position Name	Initial Gamma	Initial Theta	Initial Vega
Call Backspread	+	-	+
Put Backspread	+	-	+
Call Ratio Vertical	-	+	-
Put Ratio Vertical	-	+	-
Long Straddle	+	-	+
Short Straddle	-	+	-
Long Strangle	+	-	+
Short Strangle	-	+	-
Long Butterfly	-	+	-
Short Butterfly	+	-	+
Long Time Spread	-	+	+
Short Time Spread	+	-	-

Fig. 24 - Greeks of the Most Popular Option Strategies

Choosing the Right Strategy[16]

When trading options, there are two major components that the trader should care about the most- volatility & market direction. The implied volatility component, whether low, moderate or high, is in comparison to the trader's own volatility forecast. If you believe that 15% is a reasonable forecast but implied volatility is 13%, then it is low. Similarly if you believe that 20% is reasonable, but implied volatility is 25%- in that case implied volatility is too big! If our predicted volatility is approximately in-line with the implied volatility, we can then say that implied volatility is moderate. The same way, we can classify the directional bias of the trader with bearish, neutral and bullish. We can illustrate the right strategy, given the current view, in a table below:

[16] Idea From: Option Pricing And Volatility - Advanced Strategies And Trading Techniques - Sheldon Natenberg - (1994)

	Implied Volatility		
	Low	**Moderate**	**High**
	Buy Naked Puts	Sell The Underlying	Sell Naked Calls
	Bear Vertical Spreads		Bear Vertical Spreads
	Sell OTM Call butterflies		Buy ITM Call Butterflied
	Buy ITM Call Time Spreads		Sell OTM call Time Spreads
	Backspreads	Go On Vacation	Ratio Vertical Spreads
	Buy Straddles/Strangles		Sell Straddles/Strangles
	Sell ATM Call or Put Butterflies		Buy ATM Call or Put Butterflied
	Buy ATM Call or Put Time Spreads		Sell ATM Call or Put Time Spreads
	Buy Naked Calls	Buy The Underlying	Sell Naked Puts
	Buy Vertical Spreads		Buy Vertical Spreads
	Sell ITM Call Butterflies		Buy OTM Call Butterflies
	Buy OTM Call Time Spreads		Sell ITM Call Time Spreads

Fig. 25 - Right Strategy Given Conditions, where: Red: Bearish outlook, Yellow: Neutral outlook, Green: Bullish outlook

Closing Remarks

This paper should be used as a guide when learning about options. It helped me to strengthen and deepen the knowledge on these derivative instruments. It is important to note, that this article only covered the Black-Scholes model & its integration into the code. While being far from perfect and originally constructed for the European options, Black-Scholes model works well for American too, as usually there isn't much motivation to exercise an option early. Lastly, as volatility being the major component, one should worry more about the estimate of this variable, rather than diving into more complex models.

Conclusion

We successfully demonstrated how to create an option position analysis tool & have evaluated several key option strategies

The full code can be found in the GitHub provided by MercatusCapitis.com or by contacting Matas Urbonavicius at matas.urbonavicius@gmail.com

About The Author

Matas is an aspiring capital markets professional with experience ranging from M&A in NYC to Prop trading for a Family office located in Luxembourg