Algorithm for payoff calculation for option trading strategies using vector terminology

Pankaj Sinha and Archit Johar

Faculty of Management Studies, University of Delhi, Delhi

15. May 2009

Online at http://mpra.ub.uni-muenchen.de/15264/
MPRA Paper No. 15264, posted 17. May 2009 00:29 UTC
Algorithm for payoff calculation for option trading strategies using vector terminology

Pankaj Sinha
Faculty of Management Studies, University of Delhi, Delhi

and

Archit Johar
Department of Computer Engineering, Netaji Subhas Institute of Technology, New Delhi

Abstract

The aim of this paper is to develop an algorithm for calculating and plotting payoff of option strategies for a portfolio of path independent vanilla and exotic options. A general algorithm for calculating the vector matrix for any arbitrary combination strategy is also developed for some of the commonly option trading strategies.

1.0 Introduction

Hull [1] discusses the payoffs for long and short positions in Call and Put options by using algebraic techniques. J.S. Chaput & L.H. Ederington [3], Natenberg [2] and Hull [1] contain the bibliographies and survey of literature on the theoretical background of option strategies for path independent vanilla and exotic options such as European, Bermuda, Forward Start, Digital/Binary and Quanto options. There are various open source option strategy calculators like “Option” [4] that only rely on algebraic analytical and graph superposition techniques to plot graphs for overall profit/loss. We in this paper develop an algorithm using vector terminology to plot final profit/loss graph of various option strategies.
### 2.1 Option strategies using vector notation

For a spot price $S_T$ at time $T$ and a strike price $K$, the payoff for a long position in call option is given by $\max(S_T - K, 0)$ and the payoff is $\min(S_T - K, 0)$ for the short position in the call option. Similarly the payoff for a long position in put is $\max(K - S_T, 0)$ whereas it is $\min(K - S_T, 0)$ for a short position in the put option. We can represent a vector payoff matrix for any option strategy as a $2 \times N$ matrix.

<table>
<thead>
<tr>
<th>Vector $V_i$</th>
<th>$V_1$</th>
<th>$V_2$</th>
<th>$\ldots$</th>
<th>$V_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strike Price $K_i$</td>
<td>$K_1$</td>
<td>$K_2$</td>
<td>$\ldots$</td>
<td>$K_n$</td>
</tr>
</tbody>
</table>

In the above matrix the strike prices $K_1, K_2, \ldots, K_n$ for combination of options are in the ascending order, i.e., $K_1 < K_2 < \ldots < K_n$. The vector $V_i$ can be interpreted as slope of the payoff graph of option strategy. By default the smallest strike price is always taken to be zero i.e., $K_1 = 0$. The vector is always an integer in the interval $(-\infty, \infty)$. We can interpret the above matrix in terms of slope of the profit/loss curve obtained for option strategies.

$$slope = \begin{cases} V_i, & \text{for } K_i < K < K_{i+1} \text{ and } i < n \\ V_i, & \text{for } K > K_i \text{ and } i = n \end{cases}$$

Vector matrix for long and short position is given by

**Long Position**

<table>
<thead>
<tr>
<th>$V_1$</th>
<th>$V_2$</th>
<th>$\ldots$</th>
<th>$V_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_1$</td>
<td>$K_2$</td>
<td>$\ldots$</td>
<td>$K_n$</td>
</tr>
</tbody>
</table>

**Short Position**

<table>
<thead>
<tr>
<th>$-V_1$</th>
<th>$-V_2$</th>
<th>$\ldots$</th>
<th>$-V_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_1$</td>
<td>$K_2$</td>
<td>$\ldots$</td>
<td>$K_n$</td>
</tr>
</tbody>
</table>

Using the above vector notation we can represent long and short position in call option as under

**Long call**

<table>
<thead>
<tr>
<th>0</th>
<th>+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$K_1$</td>
</tr>
</tbody>
</table>

**Short Call**

<table>
<thead>
<tr>
<th>0</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$K_1$</td>
</tr>
</tbody>
</table>

For long position in call, profit/loss curve has two slopes 0 and +1 whereas for a short position the slope of profit/loss curve has two slopes 0 and -1.

![Figure 1: Long Position in Call Option](image-url)
Figure 2: Short position in Call Option

Similarly, the vector matrix for long and short position in put options are:

<table>
<thead>
<tr>
<th>Long Put</th>
<th>Short Put</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>+1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>K_1</td>
<td>K_1</td>
</tr>
</tbody>
</table>

For long position in stock, the slope of profit/loss curve is +1 and strike price is assumed to be zero whereas for short position in stock, the slope of profit/loss curve is -1 and strike price is assumed to be zero. The vector matrix notation is given as:

<table>
<thead>
<tr>
<th>Long Stock</th>
<th>Short Stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

When we trade in n units of options using a particular option strategy, the entire vector row is multiplied by n.

<table>
<thead>
<tr>
<th>n*V_1</th>
<th>n*V_2</th>
<th>....</th>
<th>n*V_n</th>
</tr>
</thead>
<tbody>
<tr>
<td>K_1</td>
<td>K_2</td>
<td>.....</td>
<td>K_n</td>
</tr>
</tbody>
</table>

The data set for a portfolio using n option strategies can be represented as

**Strategy 1**

<table>
<thead>
<tr>
<th>V_{i1}</th>
<th>V_{i2}</th>
</tr>
</thead>
<tbody>
<tr>
<td>K_{i1}</td>
<td>K_{i2}</td>
</tr>
</tbody>
</table>

**Strategy 2**

<table>
<thead>
<tr>
<th>V_{21}</th>
<th>V_{22}</th>
</tr>
</thead>
<tbody>
<tr>
<td>K_{21}</td>
<td>K_{22}</td>
</tr>
</tbody>
</table>

...  

**Strategy i**

<table>
<thead>
<tr>
<th>V_{i1}</th>
<th>V_{i2}</th>
<th>....</th>
<th>V_{ik}</th>
<th>....</th>
</tr>
</thead>
<tbody>
<tr>
<td>K_{i1}</td>
<td>K_{i2}</td>
<td>.....</td>
<td>K_{ik}</td>
<td>.....</td>
</tr>
</tbody>
</table>

...  

**Strategy n**

<table>
<thead>
<tr>
<th>V_{n1}</th>
<th>V_{n2}</th>
<th>....</th>
<th>V_{nm}</th>
</tr>
</thead>
<tbody>
<tr>
<td>K_{n1}</td>
<td>K_{n2}</td>
<td>.....</td>
<td>K_{nm}</td>
</tr>
</tbody>
</table>
Note that the number of columns in each option strategy can be different. We can use the above derived vector matrices to form profit/loss function for any combination of option strategies using the following algorithm:

**Algorithm**

To plot the overall payoff strategy we need the initial Y intercept of the strategy apart from the resultant vector matrix. This Y intercept can be calculated using matrices of length greater than one using the formula

\[
Y_{\text{intercept}} = \sum (-1 \times \text{Vector}(A[j]) \times \text{Strike}_\text{price}(A[j+1]))
\]

\[
Y_{\text{intercept}} = Y_{\text{intercept}} + \text{Net}_\text{Premium}_\text{Paid}
\]

**Step 1**

For I = 1 to no_of_options
  For j = 1 to length_of_option_matrix
    Insert A[j] in Result_matrix in sorted increasing order on the basis of Strike_price(A[j]).

**Step 2**

For k = 1 to length_of_Result_matrix
  Vector(B[k])=0
  For I = 1 to no_of_options
    For j = 1 to length_of_option_matrix
      If Strike_price(B[k]) = Strike_price(A[j])
        Vector(B[k]) = Vector(B[k]) + Vector(A[j])
      ElseIf j < length_of_option_matrix
        If Strike_price(A[j]) < Strike_price(B[k]) < Strike_price(A[j+1])
          Vector(B[k]) = Vector(B[k]) + Vector(A[j])
        Else
          Vector(B[k]) = Vector(B[k]) + Vector(A[j])
      End If
    End For
  End For

**Step 3**

For I = 1 to no_of_options
  j=1
  If length_of_option_matrix > 1
    Yint = Yint + -1 * Vector(A[j]) * Strike_price(A[j+1])
  End If
Yint = Yint + NetPremium

**Step 4**

For k = 1 to length_of_Result_matrix - 1
  Plot line with slope Vector(B[k]) & Y Intercept Yint between points Strike_price(B[k]) & Strike_price(B[k+1])
  ypoint=Vector(B[k])*( Strike_price(B[k+1]) - Strike_price(B[k]) ) + Yint
  Yint = ypoint - Vector(B[k+1])* Strike_price(B[k+1])
  k = length_of_Result_matrix
  Plot line with slope Vector(B[k]) between points Strike_price(B[k]) & infinity

The source code for the above algorithm is written and implemented on VC++.Net 2005 using open source graph plotting utility Gnuplot.
Illustration 1: An investor buys $3 put with strike price $35 and sells for $1 a put with a strike price of $30. (Example 10.2, page 224 given in Hull [1])
The above data can be represented as:

<table>
<thead>
<tr>
<th>Buy Put</th>
<th>+</th>
<th>Sell Put</th>
<th>=</th>
<th>Payoff (Bear Spread)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>0</td>
<td>+1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>35</td>
<td>0</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

Initial Y intercept is \(-1*(-1*35) + -1*(1*30) – 3 + 1 = 35 -30 -3 + 1 = 3\)

One can use the following form to input the data of his/her option strategy:

Figure 3: Input Screen

The following is the output of the final payoff of combination of option strategy in vector notation as discussed above.

Figure 4: Vector Payoff Matrix
The algorithm gives the following resultant profit/loss graph of the above combination of option strategies in the form of a bear put spread.

![Payoff Graph](image.png)

Figure 5: Payoff Graph

The loss is $2 if stock price is above $35 and the profit is $3 if stock price below $30.

### 2.2 Some More Complex Strategies

The following are the vector matrices for some of the commonly traded strategies:

**Long Combo**

\[
\begin{array}{c|c|c}
\text{Sell Put} & \text{Buy Call} & \text{Long Combo} \\
+1 & 0 & \begin{array}{c}
\text{0} \\
K_1
\end{array} \\
0 & +1 & \begin{array}{c}
\text{0} \\
K_2
\end{array} \\
\end{array} = \begin{array}{c|c|c}
\text{+1} & 0 & \text{+1} \\
0 & K_1 & K_2
\end{array}
\]

**Long Straddle**

\[
\begin{array}{c|c|c}
\text{Buy Put} & \text{Buy Call} & \text{Long Straddle} \\
-1 & 0 & \begin{array}{c}
\text{0} \\
K_1
\end{array} \\
0 & +1 & \begin{array}{c}
\text{0} \\
K_1
\end{array} \\
\end{array} = \begin{array}{c|c|c}
\text{-1} & \text{+1} \\
0 & K_1
\end{array}
\]

**Short Straddle**

The vector matrix of short straddle is negative of that of long straddle

\[
\begin{array}{c|c|c}
\text{Buy Put} & \text{Buy Call} & \text{Short Straddle} \\
+1 & -1 & \begin{array}{c}
\text{0} \\
K_1
\end{array} \\
0 & +1 & \begin{array}{c}
\text{0} \\
K_1
\end{array} \\
\end{array}
\]
### Strip
Buy call + Buy 2 puts = Strip

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>+1</th>
<th></th>
<th>-2</th>
<th>0</th>
<th></th>
<th>-2</th>
<th>+1</th>
<th>0</th>
<th>K₁</th>
<th>0</th>
<th>K₁</th>
</tr>
</thead>
</table>

### Strap
Buy 2 calls + Buy put = Strap

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>+2</th>
<th></th>
<th>-1</th>
<th>0</th>
<th></th>
<th>-1</th>
<th>+2</th>
<th>0</th>
<th>K₁</th>
<th>0</th>
<th>K₁</th>
</tr>
</thead>
</table>

### Long Strangle
(0<K₁<K₂)
Buy put + Buy call = Long Strangle

<table>
<thead>
<tr>
<th></th>
<th>-1</th>
<th>0</th>
<th></th>
<th>0</th>
<th>+1</th>
<th></th>
<th>-1</th>
<th>0</th>
<th>+1</th>
<th>0</th>
<th>K₁</th>
<th>K₂</th>
</tr>
</thead>
</table>

### Short Strangle
The vector matrix of short strangle is negative of that of short strangle. (0<K₁<K₂)

<table>
<thead>
<tr>
<th></th>
<th>+1</th>
<th>0</th>
<th>-1</th>
<th>0</th>
<th>K₁</th>
<th>K₂</th>
</tr>
</thead>
</table>

### Collar
(0<K₁<K₂)
Long Stock + Buy Put + Sell call = Collar

<table>
<thead>
<tr>
<th></th>
<th>+1</th>
<th>0</th>
<th>-1</th>
<th>0</th>
<th></th>
<th>K₂</th>
</tr>
</thead>
</table>

### Box Spread
(0<K₁<K₂)
Buy Call + Sell call + Sell Put + Buy Put = Box Spread

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>K₁</th>
<th>K₂</th>
</tr>
</thead>
</table>

### Long Call Butterfly
(0<K₁<K₂<K₃)
Buy Call + Sell 2 call + Buy Call = Long Call Butterfly

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>+1</th>
<th>-1</th>
<th>0</th>
<th>0</th>
<th>K₁</th>
<th>K₂</th>
<th>K₃</th>
</tr>
</thead>
</table>

|        | 0  | 0  | 0  | 0  | K₁ | K₂ | K₃ |
**Short Call Butterfly**
The vector matrix of short call butterfly is negative of that of long call butterfly (0<K_{1}<K_{2}<K_{3})

\[
\begin{array}{cccc}
0 & -1 & +1 & 0 \\
0 & K_{1} & K_{2} & K_{3} \\
\end{array}
\]

**Long Call Condor**
\((0<K_{1}<K_{2}<K_{3}<K_{4})\)

Buy Call : +
\[
\begin{array}{ccc}
0 & +1 & 0 \\
0 & K_{1} & K_{2} \\
\end{array}
\]

Sell Call : -
\[
\begin{array}{ccc}
0 & -1 & 0 \\
0 & K_{3} & K_{4} \\
\end{array}
\]

**Short Call Condor**
The vector matrix of short call condor is negative of that of long call condor (0<K_{1}<K_{2}<K_{3}<K_{4})

\[
\begin{array}{cccc}
0 & -1 & 0 & +1 & 0 \\
0 & K_{1} & K_{2} & K_{3} & K_{4} \\
\end{array}
\]

**Illustration 2:** Let a certain stock is selling at $77. An investor feels that significant change in price is un-likely in the next 3 months. He observes market price of 3 month calls as

<table>
<thead>
<tr>
<th>Strike Price($)</th>
<th>Call Price($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>75</td>
<td>12</td>
</tr>
<tr>
<td>80</td>
<td>8</td>
</tr>
<tr>
<td>85</td>
<td>5</td>
</tr>
</tbody>
</table>

The investor decided to go long in two calls each with strike price $75 and $85 and writes two calls with strike price $80. Payoff for different levels of stock prices is given as

**Figure 6: Payoff Graph**
The profit /loss when stock price is at maturity is

<table>
<thead>
<tr>
<th>Stock Price($)</th>
<th>Profit/Loss($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>-1</td>
</tr>
<tr>
<td>68</td>
<td>-1</td>
</tr>
<tr>
<td>73</td>
<td>-1</td>
</tr>
<tr>
<td>78</td>
<td>2</td>
</tr>
<tr>
<td>83</td>
<td>1</td>
</tr>
</tbody>
</table>

References