# Interim Project Report

Kenny McAvoy

10/19/2020

## Project Directive

This project is an implementation of the Dynamic Delta Hedging Strategy with call options. The delta strategy offsets long and short positions in order to reduce risk associated with the underlying asset. In the case of buying call options, that would mean long on the option and short on the underlying asset. This may be reflected in taking opposing positions in the call option and the underlying asset. This is assuming the underlying asset price and the option price are correlated. The delta of the Delta Hedging Strategy is derived from the Black Scholes (BS) model.

## Model Implementation

The Project file tree is as follows:

Project File Tree

```
|___interimProject.o (Compiled Executable)
|___InterimProject.cpp
|___pricing_method.h
|___option.h
|___option.cpp
|___option_price.h
|___option_price.cpp
|___data
        | |___sec_GOOG.csv
        | |___interest.csv
        | |___op_GOOG.csv
|___Output
        | |___bsCalldelta.csv (Generated)
        | |___HE.csv (Generated)
        | |___pricingPaths.csv (Generated)
        | |___Bvalues.csv (Generated)
        | |___bsCallprices.csv (Generated)
        | |___results.csv (Generated)
```

The project was compiled using the following command on a MacOS system:

g++ -o interimProject.o InterimProject.cpp option.cpp option_price.cpp -I /usr/local/Cellar/boost/1.74.0/include -I /usr/local/include/ql -L /usr/local/lib/ -lQuantLib

Boost Library – Verison 1.74

Quantlib Library – Version 1.19

## File Descriptions

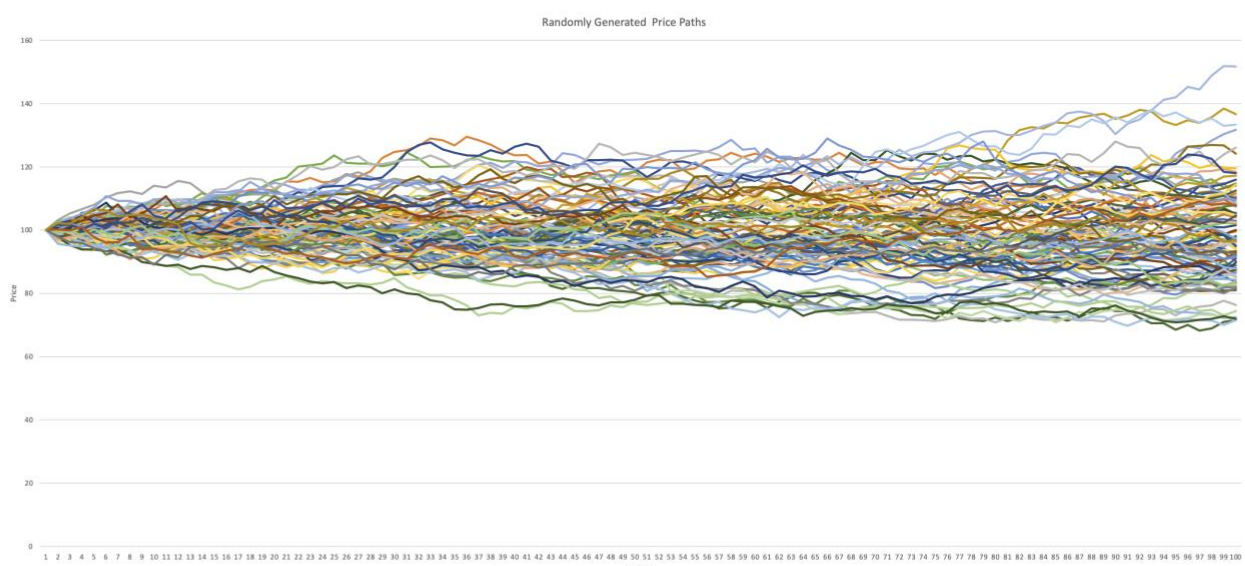
### InterimProject.cpp

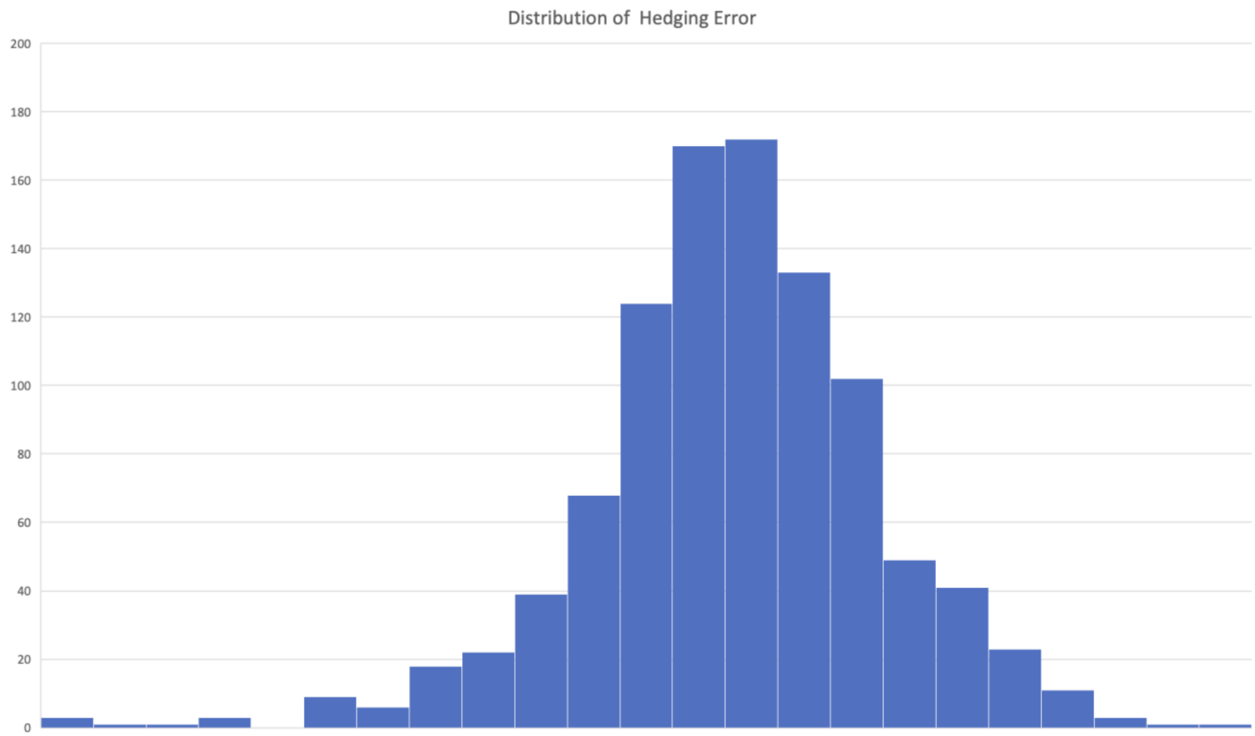
This file contains the main source of importing each file and running each step of the project requirements. Each section is labelled in the code document by part. In part 1, using a random number generator, I create 1000 stocks with varying price paths over an interval of 100 equal periods. The prices are generated through a random change factor (Z) between each interval following this equation:

$$S_{t+\Delta t} = S_t + \mu S_t \Delta t + \sigma S_t \sqrt{\Delta t} Z_t,$$

(Source: Project Instruction)

Following this, an option pricing object based on the option.h and option_pricing.h files is generated for each stock over the period based on S0 = 100, T = 0.4, μ = 0.05, σ = 0.24, r = 0.025. Following the initialization of the option pricing class, the BS call option price, delta, beta, and hedging errors are calculated. These calculations are saved to the output folder for reference if necessary.






Following this section, part II is started by importing each of the Google data CSV files, based on the input of the user for option dates, maturities, and strike prices. The implied volatility, option delta, beta, hedging error, PNL, and PNL without hedging are calculated using functions from the option_pricing.h/.cpp file for each of the options meeting the criteria set by the user. The calculations are saved to a results.csv file in the output folder. An example of what this file may look like is presented below:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Option Window Start Date: 2011-07-05 | | | | | | | |
| Option Window End Date: 2011-07-29 | | | | | | | |
| Option Maturity Date: 2011-09-17 | | | | | | | |
| Option Strike Price: 500 | | | | | | | |
| date | S | V | implied vol | delta | hedgingerror | PNL | PNLwHedge |
| 2011-07-05 | 532.44 | 44.2 | 0.259686 | 0.722654 | 0 | 0 | 0 |
| 2011-07-06 | 535.36 | 46.9 | 0.269337 | 0.733318 | -0.592419 | -2.7 | -0.592419 |
| 2011-07-07 | 546.6 | 55.3 | 0.269861 | 0.787576 | -0.160121 | -11.1 | -0.752541 |
| 2011-07-08 | 531.99 | 43.95 | 0.268563 | 0.719396 | -0.159464 | 0.25 | -0.912005 |
| 2011-07-11 | 527.28 | 41 | 0.27595 | 0.691509 | -0.440645 | 3.2 | -1.35265 |
| 2011-07-12 | 534.01 | 46.4 | 0.286788 | 0.722767 | -0.748334 | -2.2 | -2.10098 |
| 2011-07-13 | 538.26 | 49.3 | 0.287137 | 0.745056 | 0.16932 | -5.1 | -1.93166 |
| 2011-07-14 | 528.94 | 41.15 | 0.272178 | 0.706903 | 1.20383 | 3.05 | -0.727832 |
| 2011-07-15 | 597.62 | 99.65 | 0.28149 | 0.940744 | -9.95189 | -55.45 | -10.6797 |
| 2011-07-18 | 594.94 | 97.65 | 0.300413 | 0.926444 | -0.524012 | -53.45 | -11.2037 |
| 2011-07-19 | 602.55 | 103.8 | 0.266228 | 0.96029 | 0.89747 | -59.6 | -10.3063 |
| 2011-07-20 | 595.35 | 97.8 | 0.299707 | 0.931925 | -0.91736 | -53.6 | -11.2236 |
| 2011-07-21 | 606.99 | 108.15 | 0.275666 | 0.964257 | 0.494075 | -63.95 | -10.7295 |
| 2011-07-22 | 618.23 | 118.7 | 0.24842 | 0.985999 | 0.284382 | -74.5 | -10.4452 |
| 2011-07-25 | 618.98 | 119.95 | 0.294243 | 0.971588 | -0.51448 | -75.75 | -10.9596 |
| 2011-07-26 | 622.52 | 123.25 | 0.28697 | 0.978582 | 0.135515 | -79.05 | -10.8241 |
| 2011-07-27 | 607.22 | 108.65 | 0.304805 | 0.957691 | -0.376445 | -64.45 | -11.2006 |
| 2011-07-28 | 610.94 | 112.1 | 0.302652 | 0.964977 | 0.108576 | -67.9 | -11.092 |
| 2011-07-29 | 603.69 | 106.8 | 0.370045 | 0.924709 | -1.70015 | -62.6 | -12.7922 |

The hedging error and beta are calculated according to the following formulas:

$$HE_i = \delta_{i-1} S_i + B_{i-1} e^{r_{i-1} \Delta t} - V_i$$

$$B_i = \delta_{i-1} S_i + B_{i-1} e^{r_{i-1} \delta t} - \delta_i S_i \ (i \geq 1)$$

$$B_0 = V_0 - \delta_0 S_0.$$

(Source: Project Instruction)

Si, Vi, ri are the stock price, option price, and risk-free rate at each period i. delta t represents one business day.

The BS option price and delta (Delta is the normal CDF of D1) are calculated according to the following formula:

$$C = N(d_1) S_t - N(d_2) K e^{-rt}$$

$$\text{where } d_1 = \frac{\ln \frac{S_t}{K} + (r + \frac{\sigma^2}{2}) t}{\sigma \sqrt{t}}$$

$$\text{and } d_2 = d_1 - \sigma \sqrt{t}$$

(Source: Wikipedia – Black Scholes Model)

### Option.cpp/Option.h

The option.h and option.cpp files define the option class and provide simple retrieval function for private members. The members are: Time to Maturity, Strike Price, Asset Price, Risk Free Rate, and Volatility.

### Pricing_method.h

This header file simply contains two virtual functions connecting to the option_pricing.h/.cpp files.

### Option_price.cpp/option_price.h

This file contains the main functionality for calculations in the interimProject.cpp file. The class inherits from the option class so that the required parameters are easily retrieved using the previously mentioned option.cpp functions. Following, 3 main function in the context of this project are defined: Black_Scholes_Option_Price(), option_delta(), and impliedvol(). The BS option pricing function performs the BS method using the private members of the option class, according to the BS pricing model.

Delta is calculated from the same private members as the normal CDF of d1 in the BS pricing model. Finally impliedvol() takes additional parameters of the lowerlimit, upperlimit, tolerance, and price_target. The function then iteratively uses the option_price class and Black_Scholes_Option_Price() function to deduce the approximate implied volatility of the option according to the tolerance and option price.

## Unit Test

The following is a screen shot of a unit test. The results.csv file would represent the same results shown earlier in this document:

```
kennethmcavoy@Kennethinch2017 Project1 % ./interimProject.o
Interim Project - Part 1 - 1
File saved in output folder as pricingPaths.csv
Interim Project - Part 1 - 1 - Complete

Interim Project - Part 1 - 2
File saved in output folder as bsCallprices.csv
File saved in output folder as bsCalldelta.csv
File saved in output folder as Bvalues.csv
File saved in output folder as HE.csv
Interim Project - Part 1 - 2 - Complete

Interim Project - Part 2
Please Specify the Parameters
Enter the option window start date (yyyy-mm-dd):
2011-07-05
Enter the option window end date (yyyy-mm-dd):
2011-07-29
Enter the time to maturity (yyyy-mm-dd):
2011-09-17
Enter the strike price:
500

Interest file opened
Interest file Imported
Google Closing Price file opened
Google Closing Price file Imported
Google Option file opened
Google Option file Imported

Analysis Complete
File saved in output folder as results.csv
Interim Project - Part 2 - Complete
Interim Project - End
kennethmcavoy@Kennethinch2017 Project1 %
```

## Outcome of Implementation

The implementation seemed to have promising results. The cumulative hedging error of the first part is unsurprisingly small, as the option prices were calculated based on the BS model and hedging errors generated on parameters of the same model.

In part II, using data from Google's asset price and option price contracts, the model had a much larger cumulative hedging error of $-12.79. This is a relatively small amount of error given the performance of a portfolio without hedging, $-62.60. This would imply that this model does not completely negate risk but does reduce the possible loss and provides some security to the movement of the underlying asset. The reason the hedging does not work exactly as it should may be because the option price and underlying asset price are not always directly correlated. The results shows that in the given market, the option and underlying stock values are correlated enough that hedging according to underlying asset movements can reduce the risk of the portfolio, just not completely.