LOWER BOUND FOR BIDDING OF OPTION CHAIN PRICING

CFWin18-107-108

Project Report

APRIL 16, 2018

SUBMITTED BY:
SIRJANA KAUR (7813553) & MANMOHIT SINGH (7820006)

Abstract

Over the last few decades, it has become a great challenge for the investors to select the most appropriate option contract which could possibly lead them to earn maximum possible profits. The major reason behind this is that a company offers multiple option contracts with varied expiration dates, leaving the investor in a dilemma to choose the perfect one. To help with the decision making process of investing and gaining best possible profits, this project focuses on assisting a person/investor to compute option value on their own and bid accordingly for any option contract using some proposed bidding strategies. This system helps the investors in letting them know about the safe limit until which they opt to invest on a particular option contract. In this project, bidding has not been done for real world contracts and hence the results are limited to simulations and speculations using different virtual case studies.

1. Introduction

Recent years have seen a swift increase in the number of online auction sites that allow both individuals and businesses to trade goods within a virtual worldwide market (prominent examples include eBay, uBid, and Yahoo!). While there are many minor implementation differences between these online auction systems (e.g., the availability of proxy bidding services, the use of a fixed or variable auction duration, and the ability to set both starting and reserve prices), these auction systems have been modelled on real-world counterparts, and thus, in general, they all share two common features [3, 4]. First, they are predominantly based on the ascending price English auction, whereby bidders submit bids to an auctioneer in an open fashion and the auction price increases until no bidder is willing to bid higher. Second, they typically exhibit discrete bid levels, whereby the bids that the bidders may submit within the auction are restricted to certain levels either through a minimum bid increment that the next bid must exceed (as in eBay) or by forcing the auction price to increment through a set of predetermined price levels. Our work is to substantiate the process of buying option contracts for investors on their own, without using any third-party facility like brokers. The toughest part in purchasing financial assets like stocks and options is that many times people done know or don't have enough finance knowledge and therefore, are not aware about the company's past performances and the possible outcomes of their investments which often lead investors to suffering of losses.

For valuation of options, there exists several approaches like binomial model, Black-Scholes-Merton model and finite differencing and many more. Our work is specifically based on Black-Scholes option pricing model (also known as BSM model). For calculation of the option values, BSM model uses two major assumptions of constant volatility and constant interest rates. We propose solution for volatility for not taking a constant value, but a computed value. Volatility can be classified further into two categories: historical and implied. The historical volatility is computed based on past performance of company, considering the variations in stock prices. For more precision, implied volatility has been computed by using the market price of the option contract, the underlying stock price, the strike price, the time to expiration and the risk-free interest rate in the Black-Scholes model.

Using the system that is proposed in this study, investors can compute historical volatility using the monthly stock data of a company's stock, which is freely available on web. This computed volatility can be further used by investor to obtain the Call or Put option value using this system. The computed option value gives an investor the idea of estimate worth price of an option contract and furthermore, let them earn fruitful profits. By calculating implied volatility, we define the upper limit of auction bid for obtaining an option contract. Once the range of bidding is computed, the system that is proposed by us, suggests the variations required in bidding for opting a contract. For online bidding, an enhancement to one of the existing best algorithm called Doubling technique has been proposed as well as used. The improvisation includes the doubling of the difference between two consecutive bids every time, while maintaining competitive ratio close to the best found Competitive Ratio as well.

This report is divided into eight sections. This report starts with discussions on background, related work and techniques as well as algorithms which have been used in our project has been carried out (Section 2). Section 3 discusses the specific problem statement which also summarizes motivation and focus of the project. Section 4 describes the solution strategy and its implementation used in the project such as historical volatility, implied volatility and many more. Section 5 outlines the experimental approach by varying parametric conditions. Section 6 includes the timeline of the project. The most important section of the report is results section (Section 7) which has further two subsections explaining the environment as well as analysis of the project. Conclusion along with the future scope has been discussed in the final section, Section 8.

2. Background and Related Work

One of the earliest operations research studies of competitive online bidding was made by [6]. It was argued by the author that a bidder should study the past behavior of its competitors to discover the patterns that governed their bidding. This information could be used, in any (particular) competition, to estimate the probability distribution of any (particular) competitor i's bid b_i, as represented by the function describing how likely the bid b_i is to be less than any (particular) amount b.

The emphasis of much of recent bidding theory has been on ranking auctions based on the expected receipts they generate [7]. Sometimes this approach is taken to the extreme of determining the institutions that maximize expected receipts, on the grounds that such institutions will be the ones chosen by the auctioneer/seller. The results of these maximization problems are, for all but the simplest environments, auctions of unreasonably complicated forms involving payments by the seller to losers, required side bets among the bidders, and so on [13]. That such forms are not observed in practice indicates that the "optimal auctions" theory in which the auctioneer can tailor a specific institution to each environment may be a poor way to explain actual institutions. The various models differ in their conclusions about the efficiency of the allocations resulting from various kinds of auctions [14]. The Hansen endogenous quantities model assumes symmetry among the bidders before the auction and predicts that the outcome of the first-bid auction is more efficient. When the quantity traded is fixed and the bidders are not symmetrical, the second-bid auction always assigns the goods to the proper bidder, but the first-bid auction may fail to do so.

Of course, the final allocation itself is only one aspect of the efficiency of auctions. Another important aspect is the cost of preparing a bid. Complicated auctions and those that provide large returns to information gathering are likely to increase bid preparation costs. The English auction system, in which a bidder's optimal bidding strategy does not depend on how his/her competitors bid, economizes on information gathering and bid preparation costs [3]. In summary, at least for fixed quantity environments, the English auction possesses a variety of characteristics that help to explain its popularity. It generates more receipts on average than the Dutch/sealed-bid auction [15]. It leads to efficient outcomes in a wider range of environments and it economizes on information gathering and bid preparation costs.

There are many models that exist which determine the auction prices but, none of them satisfies the option contracts specifically. Existing fundamental auction bidding models are of two types, first ones are namely; weighted lognormal models (LN) and others are known as generalized gamma models (GG) [1,2]. Skitmore [2] compared and analyzed these two models and came up with results that these can be improved as the time taken for evaluation is considerably large and there was a lack of local bidding procedures or might be due to the intensity of market conditions. As the option contracts vary vigorously with market conditions.

There are many factors that influence the price of an option. Among the more influential and better-known factors are the price of the underlying security, the strike price of the option, the time remaining before he expiration, the volatility of the underlying asset, the dividend payout of that security (if any), and the risk-free interest rate. Two obvious variables that influence an option's price are the strike price of the option and the price of the underlying security. Algorithm that has been used in this project is BSM model as it has become more dominant pricing model than Binomial Pricing. This is because it is very easy to compute both option prices and the Greeks and because it is more effective as well as give good results [18]. The assumptions which are usually taken into account are; constant asset volatility as well as constant risk-free interest. In our approach to find the lower bound for bidding of option chain pricing, we used these two assumptions and compute them using given parameters of the option contract.

Black, Scholes and Merton (BSM) [8,10] presented a mode of analysis that revolutionized the theory of "corporate liability pricing". In particular, the option pricing formula given in [8] did not require knowledge of either investors' tastes or their beliefs about expected returns on the fundamental common stock [12]. Moreover, under speculated conditions, their formula must hold to bypass the creation of arbitrage probabilities.

It is normally assumed in the auction literature that the bid price is a continuous variable and it can take on any value in a certain range. The implication is that extremely small bid decrements may be used by the auctioneer to test the bidder's valuation of the object for sale and the auction could last for a long time [3]. Option contract is a right to sell or to buy any asset at a given price within a definite period of time. Permits to purchase decision-making stock options, common stock, put and call options are few of the examples of option contracts.

In online bidding [16], faced with some unknown integer threshold u, an auctioneer submits a sequence (d_l) of bids until one is greater than or equal to u, paying the sum of its bids. Any strategy is defined by its sequence of bids, and its competitive ratio is

$$\max_{u,k}\left\{\frac{d_0 + d_1 + \cdots + d_k}{u} : d_{k-1} < u \leq d_k\right\}$$

The best sequence (d_l) would be a natural approach which is to double the bid each time, that is $d_l = 2^l$ for $l \geq 0$. The worst case could be when $u = d_{k-1}$, and routine calculations show that this strategy's competitive ratio is 4 [17], which also turns out to be optimal in the deterministic setting and using this same doubling algorithm as the base of our research, a modification to this algorithm has been suggested. It is made to be suitable for online option bidding which we discuss in a later section.

3. Specific Problem Statement

Models do exist for determining the auction prices but, none of them satisfies the option contracts specifically. The aim of this project is to propose the concept of online bidding algorithms and to analyze their behavior for buying real-world option contracts and implementation of them for deriving the least bid amount for buying the option contract. As an example, the option contracts offered by APPL Inc. exist till 2019 as per NASDAQ. Our project suggests an investor an optimal price which should be paid in an auction for obtaining the contracts offered by that respective company.

As this is a team project, responsibilities were equally dissolved. Sirjana was mainly focused to come up with an appropriate algorithm for finding variations in consecutive bids whereas, Manmohit accurately analyzed the real-time option contracts and figured out the lower bound for bids. In general, both put their equal contribution in all tasks and at each step, both were aware of everything that was going on in the project. The source codes were written by both, without the help of each other, to compare the final results, which resulted the same. Many papers and books were closely followed by both, with and without the help of each other, to find the appropriate online algorithm and auction, that was required for online bidding process.

4. Solution Strategy and Implementation

The project is divided into two main phases. The first part focuses on calculating the accurate option pricing for which we used the Black-Scholes Model and other part of our project focuses on determining the bidding sequences required for assisting an investor for obtaining a prolific option contract. As mentioned in introduction, this project proposes a solution for one of the assumption in option value calculation when using the Black-Scholes model which is Volatility.

4.1 Volatility

Volatility is a measure of how much a stock characteristically moves in each unit of time, often stated as an annualized percentage. It is an arithmetical measure of the spreading of returns for a given security (or market) index. It refers to the amount of uncertainty involved in size of changes in a security's value. As every trader knows, stocks differ greatly in their volatility. A higher volatility means that a security's value can potentially be spread out over a larger range of values. This infers that the price of the security can change dramatically over a short time in either direction. A lower volatility means that a security's value does not fluctuate dramatically, but changes in value at a steady pace over a period [18].

Volatility can either be measured by using the standard deviation or variance between returns from that same security or market index. Volatility is further divided into historical and implied classifications.

4.1.1 Historical Volatility

This factor examines how quickly a stock price has varied in the past to predict the probable stock price variations in the future. A simple yet successful way which investors can use this system to compute historical volatility has been proposed in this project using. They must provide the monthly stock data of respective company they wish to invest, in a form of excel sheet and with the help of this system, it would automatically populate that stock data and compute historical volatility value for those investors.

4.1.2 Implied Volatility

Implied volatility is the volatility that, when entered an option pricing model, yields the observed market price of an option. It is a constraint part of an option pricing model, such as the Black-Scholes (BSM) model. Implied volatility shows that how the marketplace views where volatility should be in the future. Since implied volatility is forward-looking, it helps us gauge the sentiment about the volatility of a stock or the market [9]. However, implied volatility does not forecast the direction in which an option is headed.

It is usually computed for each of several active option contracts trading on a given security at a given time. Implied volatility is computed by taking the market price of the option, the underlying stock price, the strike price, the time to expiration and the risk-free interest rate, entering these factors into the Black-Scholes formula, and back-solving for the value of the volatility. The following function briefly explains the calculation of Implied volatility.

$$\sigma = \sqrt{\frac{ln(M2)}{T} - \frac{2ln(M1)}{T}}$$

here, M1 and M2 are the first and second moments of asset prices and T being the duration of option contract which explained in the source code provided in Appendix.

These classifications of volatility are used to simulate the Option value using the BSM. For the *second phase* of this project, there is need more knowledge about the online bidding strategies.

4.2 Online Bidding

Online bidding refers to the next upcoming bids that are unknown and the auctioneer is unaware of any bidding pattern or strategies followed by the investors. We have proposed the use of doubling algorithm (which is an appropriate online bidding strategy) for bidding in English auctions and it is further discussed in the following section.

4.2.1 English Auction

In English auctions, introducing a buy price, i.e., the seller's maximum price bid at which any bidder at any time can immediately win the auction, allows the seller to gain higher expected utility than that in a traditional auction when either the seller or the buyers are risk-averse [19]. If the seller sets the buy price high enough, the buy- price English auction is efficient and guarantees the highest bidder wins. The procedure is presented below:

- The auctioneer accepts increasingly higher bids from in online manner, consisting of buyers with an interest in the respective option contract.

- The highest bidder at any given moment is considered to have the standing bid, which can only be replaces by a higher bid from a competing buyer.
- And, if there is no competing bidder that challenges the standing bid within a given time frame, the standing bid becomes the winner, and the item is sold to the highest bidder at a price equal to his or her bid.

4.2.2 Doubling Algorithm

Doubling Algorithm is one of the most famous online bidding algorithm in which every next bid is doubled from the previous existing bid. Online bidding captures many but not all applications of the doubling method. The competitive ratio of the doubling algorithm is at most 4 for the online bidding problem and this is the best competitive ratio that an online algorithm can be achieve for any online bidding problem [17]. Since the price of option contracts is not significantly large to double the bid amount every time, So, the proposed technique is doubling the difference between two consecutive bids every time. For an instance,

$$Base, (Base + 2^0 p), (Base + 2^1 p), (Base + 2^2 p), \dots, (Base + 2^n p)$$

where, 'Base' refers to the base price of option contract as defined by the auctioneer and *p* is the increment amount in the next bid. We can observe that the variation is increasing and doubled every time with each new bid. For computing the competitive ratio, the problem depends upon two factors, *base* and *n*.

Algorithm 2 Proposed Doubling approach

$Bid \leftarrow Initial\ Bid$

$CurrentBid \leftarrow \text{Bid}$

$MaxBid \leftarrow Option\ Value\ using\ our\ model$

$p = \text{increment amount}$

$i \leftarrow 0$

while (Bid < MaxBid) **do**

  CurrentBid $+=$ Bid$+2^i * p$

  $i++$

end while

$FinalBid \leftarrow CurrentBid$

Algorithm 2: Algorithm of the improvised doubling approach

Total sum of online algorithm would be: $n * Base + p(2^{n+1} - 1)$

Total cost of offline optimal: $Base + p(2^{n-1} + \epsilon)$

Competitive ratio = $\frac{n*Base + p(2^{n+1}-1)}{Base+p(2^{n-1}+\epsilon)}$

There are two cases in this algorithm.

Case 1: If the number of bids are considered to be very large as compared to the Base price then we get competitive ratio close to the original doubling algorithm.

$If\ n \gg Base$

$$\frac{n * Base}{Base + p(2^{n-1} + \epsilon)} + \frac{p(2^{n+1} - 1)}{Base + p(2^{n-1} + \epsilon)}$$

In this, as Base is considered to be very small than the number of bids, the first part of the equation would be a value which is less than 1 and the second part of the equation would give us value = 4. This means that the result of this case would be something (very small value) more than 4 i.e. competitive ratio through this case would be at least 4, which is close to optimal solution.

Case 2: If the base price is taken to be large as compared to the number of bids then our preposition is more competitive as compared to the Case 1. This is a more realistic approach in purchasing option contracts as the number of bids is always going to be smaller than the price of first bid.

$If\ Base \gg n$

$$\frac{n * Base}{Base + p(2^{n-1} + \epsilon)} + \frac{p(2^{n+1} - 1)}{Base + p(2^{n-1} + \epsilon)}$$

5. Experimental Framework

In phase 1 of the project, we have discussed that using this system an investor can compute the historical volatility of a company on its own, on the basis of the information of past performance records of how that company's stock price changed in past months. Now, since this system has not been evaluated on bidding for real-world option contracts, therefore, the simulation of the working of this system has been done using three virtual people, who compute the historical volatility using the given system. Now, it is very significant to know that what would be the difference in their individual calculations. Thinking about this realistically, we isolate the calculation of historical volatility by concept of weighted average, which means that different investors would give different weightage to the monthly performance of respective companies while calculating historic volatility, as every person has his/her own tendency of considering and giving priorities while evaluating the past performances of that company.

For simulation, consider that these three virtual investors want to buy an option contract offered by AAPL Inc. (Apple). Consider that all three investors investigate the past one year monthly stock performance of AAPL.

- Investor 1 gives equal weightage to performance of AAPL for every month.
- Investor 2 gives 60% weightage to the performance of AAPL in last six months while giving 40% weightage to the remaining six months.
- Investor 3 gives significantly more weightage that of 80% to the last four months' performance of AAPL while giving only 20% weightage to remaining eight months.

Investor Sequence	Weightage	w.r.t months	Historic Volatility
Investor 1	8.33%	Every month	12.215
Investor 2	60% - 40%	6 - 6	12.403
Investor 3	80% - 20%	4-8	14.511

Table 1: Historical volatilities computed by three virtual investors.

Subsequently, after calculating historical volatility, the next step is to compute the option value. Using this system, these investors can compute the option values (i.e. the worth price) for buying any option contract offered irrespective of the expiration date offered by AAPL.

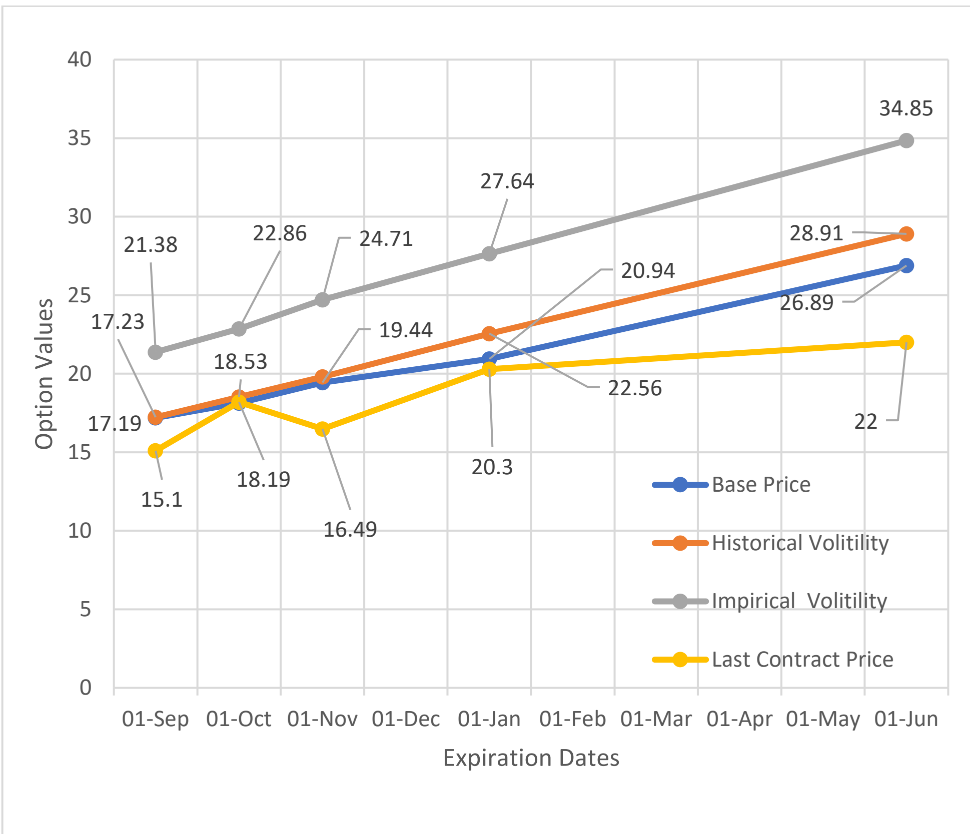
Using the BSM formula, one can compute the option value and the implied volatility. Investors need to input current asset value, historical volatility, strike price (i.e. the trade price at the end of option contract), contract tenure in months (i.e. the duration of option contract) and the derived value of historical volatility. Apart from these factors, this system assumes the value of risk-free interest to be 10% and the option contracts are non-dividend (which means that no guaranteed benefits).

The implied volatility is also computed using this Black-Scholes formula. For better simulations, the stock and option data from Bloomberg software has been used, that implicitly provides values of implied volatility for any company.

Now with computed parameters each of virtual investors can have the knowledge about the maximum value of an offered option contract. For illustration, consider the following graph 1, which depicts the option values computed for five option contracts offered by apple with varied expiration dates.

Expiration Date	Base Price	Option Price using Historical Volatility	Option Price using Implied Volatility	Last option Contract Price
21-Sep-2018	17.19	17.23	21.38	15.1
19-Oct-2018	18.14	18.53	22.86	18.19
16-Nov-2018	19.44	19.8	24.71	16.49
18-Jan-2019	20.94	22.56	27.64	20.3
21-Jun-2019	26.89	28.91	34.85	22

Table 2: Computations for virtual investor 3



Graph 1: Option value calculations done by virtual investor 3 and defines his range for bidding process.

Table 2, shows the statistical data used in Graph 1 and furthermore, represents the computed option price/value for respective option contract computed using both historic and implied volatilities for every available option contract. The second column in table 2 represents the base price set by auctioneer. Since we are considering English auctions in this study, the base price is provided for every online bidding/auction process whereas, the last option contract price refers to the last buying price for that respective contract.

In the above graph, the Base price curve signifies the initiation of bid price (i.e. from which the auction for opting an option contract starts or the first bid). The historical volatility curve is values of option price computed by Black-Scholes model using computed historical volatility by investor 3. The uppermost curve indicates the calculation of option value using implied volatility tells virtual investor 3 the peak limit till which he can invest in buying an offered option contract. Similarly, remaining two virtual investors will compute these ranges using this system.

Next phase is to figure out the required variations needed for placing the best bid for obtaining the option contract. Considering bidding for only the virtual investor 3 for this section, this system would predict the following bidding sequence. For an instance, she/he wants to bid for the AAPL Inc. option contract offered with expiration date Sept 21, 2018. Considering data from table 2, setting the value of p to be 10 and using the proposed strategy from section 4.2.3, the sequence would be:

17.19 , 17.29 , 17.49 , 17.89 , 18.69 , 20.29 , 23.49

Now, virtual investor 3 would not bid for an amount greater than 20.29 as the next bid value is more than that computed using the implied volatility from our system. Hence, he would place only 6 bids to buy this option contract.

6. Timeline of Activities

Tasks	Dates
• Project Proposal	Feb 09, 2018
• Background Research	Feb 28, 2018
• Evaluations of real-world data	March 15, 2018
• Progress Report	March 22, 2018
• Implementation on bidding models	March 30, 2018
• Final Report	April 16, 2018
• Project Presentation	April 18, 2018

7. Results

The principal objective of this project is to help investors to bid by themselves for obtaining an option contract offered by some company. This system helps investors by defining a productive range and suggesting a bidding strategy for them, to obtain and make successful profits from that option contract.

The first phase of the project successfully computes historical volatility, implied volatility, followed by option values using these two volatility constraints. By using values of historical volatility computed by the formula proposed above and using the Black-Scholes model, implied volatility is computed. We matched the values of implied volatility computed by our system with the values provided by the Bloomberg Database. The results were found out to be similar and are shown in table 3.

The second phase of the system improvised the most competitive online bidding algorithm known as the doubling technique. Two cases are defined for evaluation of the proposed bidding strategy. In one case, the best known online algorithm outperformed, this is because these auctions for Option contracts are not merely online and since we used the English Auction strategy as our base for bidding strategy, our system approach differed from doubling algorithm approach where the first bid is 1.

7.1 Experimental Environment

For this project, the standard Black-Scholes model has been used and one of its major assumptions on volatility has been computed instead of using any assumed value for it. For the case-study, the scenario with three virtual investors has been considered, while differed from each other by their varied weightage for the past performance of respective company in calculating the historical volatility as described in Table 1.

In general, no out of the ordinary resources are required, only the personal computers and some open-source software were required.

(a) PC with RAM 16GB, Hard Disk 512GB, Processor 64 bit (minimum)
(b) OS (Linux, OSX)
(c) MySQL/Microsoft Excel (to store database)
(d) ARX (to clean data)
(e) Bloomberg software: Bloomberg database has been used to analyze the value of implied volatility with the ones which our system computed.

7.2 Detailed Results and Analysis

For the first phase of the project, the option values for different option contracts varied by different expiration dates has been computed. Table 3 shows how this system computes the implied volatilities for the three virtual investors considered as a part of the case study for obtaining an option contract of AAPL with maturity date of Sept 21, 2016, with respect to the asset value as per March 28, 2018 and its coherence with the results obtained from Bloomberg database.

Sequence	Implied Volatility from our system	Implied Volatility from Bloomberg Database
Investor 1	26.49	26.01
Investor 2	26.33	26.01
Investor 3	26.16	26.01

Table 3: Comparison between implied volatilities computed and obtained from Bloomberg database

The next step is to compute the option values using historic and implied volatilities. Table 4 represents the option values computed by our three virtual investors using the system. These results from Table 4 are further used to determine the bidding capacities of each virtual investor.

Sequence	Expiration Date	Base Price	Option Price using Historical Volatility	Option Price using Implied Volatility	Historical Volatility
Investor 1	21-Sep-2018	17.19	16.60	21.38	12.21%
	19-Oct-2018	18.14	17.74	22.86	
	16-Nov-2018	19.44	19.03	24.71	
	18-Jan-2019	20.94	21.69	27.64	
	21-Jun-2019	26.89	27.51	34.85	
Investor 2	21-Sep-2018	17.19	16.66	21.38	12.40%
	19-Oct-2018	18.14	17.81	22.86	
	16-Nov-2018	19.44	19.10	24.71	
	18-Jan-2019	20.94	21.77	27.64	
	21-Jun-2019	26.89	27.60	34.85	
Investor 3	21-Sep-2018	17.19	17.23	21.38	14.51%
	19-Oct-2018	18.14	18.53	22.86	
	16-Nov-2018	19.44	19.8	24.71	
	18-Jan-2019	20.94	22.56	27.64	
	21-Jun-2019	26.89	28.91	34.85	

Table 4: Range for each virtual investor and the data is used to generate the bidding sequence

As discussed in experimental framework section, the bidding sequence for obtaining AAPL Inc. option contract with maturity date Sept 21, 2018 is going to be as follows:

17.19 , 17.29 , 17.49 , 17.89 , 18.69 , 20.29 , 23.49

For option contract with maturity date Oct 19, 2018 and value of p =10 (i.e. the variation in first two bids) the sequence comes out to be:

18.14 , 18.24 , 18.44 , 18.84 , 19.64 , 21.24 , 24.44

For Option Contract with maturity date Nov 16, 2018 and value of p = 10 (i.e. the variation in first two bids) the sequence comes out to be:

19.44 , 19.54 , 19.74 , 20.14 , 20.94 , 22.54 , 25.74

For option contract with maturity date Jan 18, 2019 and value of p =10 (i.e. the variation in first two bids) the sequence comes out to be:

20.94 , 21.04 , 21.24 , 21.64 , 22.44 , 23.64 , 26.84

For option contract with maturity date June 21, 2019 and value of p = 10 (i.e. the variation in first two bids) the sequence comes out to be:

26.89 , 26.99 , 27.19 , 27.59 , 28.39 , 29.59 , 32.79

These are the bidding sequences that would be generated using the proposed strategy of doubling the difference between every consecutive bid. Since, the third virtual investor in our case study has the higher range, therefore, she/he is always going to win among the three.

8. Conclusion and Future Work

This project successfully combines the concept of online bidding with the option pricing. The initial phase of this project accurately computes the option values, which serve as the upper limit for bidding of an option contract. In the second phase, a bidding strategy is proposed, which is also closely competitive with the best known online bidding strategy. The simulations and evaluations show that even without having access to costly software(s) like Bloomberg, one can compute near to efficient option values using our proposed system. The option value computed by using historical volatility cannot be always precise, since it is truly based on the past performance of a company. Hence, it summarizes that there is always a need to compute the option values using implied volatility.

For future work, we would like to automate this system for implicitly bidding whenever the previous bid is superseded by someone else's bid. This process is called "Auction Snipping", where the last bid is placed using a software just seconds before auction process is ending. We would also like to add some more bidding strategies to our system, like incrementing every new bid with the increment of a same static amount and study how such strategies influence competitive ratio. Moreover, our work is evaluated using virtual scenarios, and the simulations are limited to only 3 investors. So, for better results we would like to implement our prepositions on large number of simulations and testing our system, on purchasing real world option contracts. Our system does not consider ensured benefits or returns as the dividends were assumed to be zero. So, making our project compatible with non-zero dividends would be one of our major future attempts.

References

[1] Kumar, Sandeep, and Deepak Garg. “Online Financial Algorithms: Competitive Analysis.” International Journal of Computer Applications, vol. 40, no. 7, 2012, pp. 8–14., doi:10.5120/4974-7228.

[2] Skitmore, M. “Generalised gamma bidding model.” Journal of the Operational Research Society, vol. 65, no. 1, Jan. 2014, pp. 97–107.

[3] David, Esther, et al. "Optimal design of English auctions with discrete bid levels." ACM Transactions on Internet Technology (TOIT) 7.2 (2007): 12.

[4] Lucking-Reiley, David. "Auctions on the Internet: What’s being auctioned, and how?." *The journal of industrial economics* 48.3 (2000): 227-252.

[5] Cohen, Jerome B., Fischer Black, and Myron Scholes. "The valuation of option contracts and a test of market efficiency." *The Journal of Finance* 27.2 (1972): 399-417.

[6] Friedman, Lawrence. "A competitive-bidding strategy." *Operations research* 4.1 (1956): 104-112.

[7] Milgrom, Paul. "Auctions and bidding: A primer." *Journal of economic perspectives* 3.3 (1989): 3-22.

[8] Black, Fischer, and Myron Scholes. "The pricing of options and corporate liabilities." *Journal of political economy* 81.3 (1973): 637-654.

[9] Dumas, Bernard, Jeff Fleming, and Robert E. Whaley. "Implied volatility functions: Empirical tests." *The Journal of Finance*53.6 (1998): 2059-2106.

[10] Merton, Robert C. "Option pricing when underlying stock returns are discontinuous." *Journal of financial economics* 3.1-2 (1976): 125-144.

[11] Cox, John C., Stephen A. Ross, and Mark Rubinstein. "Option pricing: A simplified approach." *Journal of financial Economics*7.3 (1979): 229-263.

[12] Hull, John, and Alan White. "The pricing of options on assets with stochastic volatilities." *The journal of finance* 42.2 (1987): 281-300.

[13] Cong, Lin William. *Auctions of real options*. working paper, Stanford University, 2013.

[14] Milgrom, Paul. "Auctions and bidding: A primer." *Journal of economic perspectives* 3.3 (1989): 3-22.

[15] Li, Zhen, Jinfeng Yue, and Ching-Chung Kuo. "Design of discrete Dutch auctions with consideration of time." *European Journal of Operational Research* 265.3 (2018): 1159-1171.

[16] M. Chrobak, C. Kenyon, J. Noga, and N. Young. Online medians via online bidding. In Proc. 7th Latin American Theoretical Informatics Symp. (LATIN), volume 3887 of Lecture Notes in Comput. Sci., pages 311–322, 2006

[17] Chrobak, Marek, and Claire Kenyon-Mathieu. "Sigact news online algorithms column 10: Competitiveness via doubling." *ACM SIGACT News* 37.4 (2006): 115-126.

[18] Epps, Thomas W. *Pricing derivative securities*. World Scientific Publishing Company, 2007.

[19] Hidvegi, Zoltan, Wenli Wang, and Andrew B. Whinston. "Buy-price English auction." *Journal of Economic Theory* 129.1 (2006): 31-56.

Appendix A

A) The following is the source code for calculating Historical Volatility: This is a source code written in JAVA, for running this one needs to also add Appache.POI library files, which are required for reading data from excel files.

```
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.Iterator;
import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
public class Xcel {
        //-------------------for reading data from Excel file starts------------------------
public static double[] readXLSXFile() throws IOException {
        InputStream ExcelFileToRead = new FileInputStream
        ("C:\\Users\\manmo\\Desktop\\aapl.xlsx");
        XSSFWorkbook  wb = new XSSFWorkbook(ExcelFileToRead);
        XSSFWorkbook test = new XSSFWorkbook();
        XSSFSheet sheet = wb.getSheetAt(0);
        XSSFRow row;
        XSSFCell cell;
          double array[]= new double[12];
                int i=0;
                Iterator rows = sheet.rowIterator();
                while (rows.hasNext())  {
                        row=(XSSFRow) rows.next();
                        Iterator cells = row.cellIterator();
                        while (cells.hasNext())
                        {       cell=(XSSFCell) cells.next();
                                if (cell.getCellType() == XSSFCell.CELL_TYPE_STRING)
                                {
                                        System.out.print(cell.getStringCellValue()+" ");
                                }
                                else if(cell.getCellType() == XSSFCell.CELL_TYPE_NUMERIC)
                                {
                                        System.out.print(cell.getNumericCellValue()+" ");
                                        array[i]=cell.getNumericCellValue();
                                        System.out.print(array[i]+" month "+(i+1));
                                        i++;            }
                                else    {
                                }
                        }
```

```
                        System.out.println();   }
    return array;
}
    //-------------------for reading data from Excel file ends---------------------------
    public static void main(String[] args) throws IOException {
    double stocks[]= new double[12];
        stocks=readXLSXFile();
        System.out.println();
         int i=0;
        //--------------- Investor 1 ---------------
        System.out.println();
        System.out.println(" These are the credentials for GENERAL case with normal
    average : Investor 1  ");
        System.out.println("---------------------------------------------------------------");
        double average=0, avg=0, sd=0, vol=0, sd1=0;

        for(int j=0; j<12;j++)
        avg += stocks[j];
        average= avg/12;
        System.out.println("Per anum general average is : "+average);
                for(i=0; i<12;i++)
                 sd += ( (stocks[i]-average)*(stocks[i]-average) );
                sd1=sd/11;
                System.out.println("SD is : "+sd1);
                vol=Math.sqrt(sd1);
                System.out.println("volatility is : "+vol);
                System.out.println();

        //------------------ Investor 2-------------------------
         System.out.println();
        System.out.println(" These are the credentials for Investor 2 with 60-40 average  ");
        System.out.println("---------------------------------------------------------------");
        double sum1WA1=0, sum2WA1=0, wa1a=0, wa1b=0, wa1=0;
        for(int k=0; k<6; k++)
        {
            sum1WA1 += stocks[k];
        }
        wa1a= (sum1WA1/6)*60;       // gives 60% waitage to recent 6 months
        for(int k=6; k<12; k++)
        {
            sum2WA1 += stocks[k];
        }
        wa1b= (sum2WA1/6)*40;       // gives 40% waitage to remaining 6 months
        wa1= (wa1a+wa1b)/100;
        System.out.println("wieghted average by investor 2 is "+wa1);
        sd=0;
```

```
    sd1=0;
   for(i=0; i<12;i++)
    sd += ( (stocks[i]-wa1)*(stocks[i]-wa1) );
    sd1=sd/11;
    System.out.println("SD is : "+sd1);
    vol=Math.sqrt(sd1);
    System.out.println("volatility is : "+vol);
    System.out.println();

    //------------------ Investor 3-------------------------
    System.out.println();
    System.out.println(" These are the credentials for Investor 3 with 80-20 average  ");
    System.out.println("----------------------------------------------------------------");

    double sum1WA2=0, sum2WA2=0, wa2a=0, wa2b=0, wa2=0;
    for(int k=0; k<4; k++)
    {
       sum1WA2 += stocks[k];
    }
    wa2a= (sum1WA2/4)*80;       // gives 80% waitage to recent 4 months
    for(int k=4; k<12; k++)
    {
       sum2WA2 += stocks[k];
    }
    wa2b= (sum2WA2/8)*20;       // gives 20% waitage to remaining months
    wa2= (wa2a+wa2b)/100;
    System.out.println("wieghted average by Investor 3 is "+wa2);
    sd=0;
    sd1=0;
    for(i=0; i<12;i++)
    sd += ( (stocks[i]-wa2)*(stocks[i]-wa2) );
    sd1=sd/11;
    System.out.println("SD is : "+sd1);
    vol=Math.sqrt(sd1);
    System.out.println("volatility is : "+vol);
  }
}
```

B) The following is the source code for finding Implied Volatility and Option values using Black Scholes model also written in JAVA

```
public class BSM {

   public static void main(String[] args) {
      //some initializations
      double X, S, T, T2, T1, r, ba,b, v, toh;
```

```
double d1,d2,Xa, Sa, power1, power2, power3, power4, power5, M1, M2,c,p;
double pw1, pw2,pw3, pw4;
// input parameters
X=165;      //Strike price
T=1;
T2=0.5;     //expiration time
r=0.1;      // Risk free interest
T1=T-T2;
v=0.12;     // volatility
b=0.05;
toh = 0.5;
S=172.77;   // Asset Price
```

// This M1, is the value of first moment mentioned in section 4.I.c

```
M1= (Math.exp(b*T)-Math.exp(b*toh)) / (b*(T-toh));
//System.out.println("value of M1 is "+M1);
 power1= (2*b)+(v*v);
power2= b+(v*v);
power3= T-toh;
power4= (1/power1) - (Math.exp(b*power3)) / power2;
```

// This M2, is the value of second moment as mentioned in section 4.I.c

```
M2 = ((2*Math.exp(power1*T) ) / (power2*power1*(power3*power3)) ) + (
(2*Math.exp((power1)*toh) )/ b*(power3*power3)) * power4;
//System.out.println("value of M2 is "+M2);

ba = Math.log(M1)/T;
//System.out.println("value of ba is "+ba);

v= Math.sqrt( ( (Math.log(M2)) /T) - 2*ba);
double iv=v*10;

d1= (Math.log(S/X)+ (ba+ (v*v)/2)*T2 )/(v*Math.sqrt(T2));
//System.out.println("value of d1 is "+d1);

d2= d1- (v*Math.sqrt(T2));
//System.out.println("value of d2 is "+d2);
System.out.println("value of implied volatility is "+iv);

    power5=ba-r;
    c= S*Math.exp(power5)*T2*CND(d1)- X*Math.exp(-r*T2)*CND(d2);
    if(c>0)
    System.out.println("value of C is "+c);
    else
```

```
        System.out.println("value of C is "+0);

        p= X*Math.exp(r*T2)*CND(d2)-S*Math.exp(power5)*T2*CND(d1);
        if(p>0)
        System.out.println("value of p is "+p);
        else
        System.out.println("value of p is "+0);

    }
    private static double CND(double x)     {
      int neg = (x < 0d) ? 1 : 0;
      if ( neg == 1)
      x *= -1d;

      double k = (1d / ( 1d + 0.2316419 * x));
      double y = (((( 1.330274429 * k - 1.821255978) * k + 1.781477937) *
                k - 0.356563782) * k + 0.319381530) * k;
            y = 1.0 - 0.398942280401 * Math.exp(-0.5 * x * x) * y;

      return (1d - neg) * y + neg * (1d - y);
    }

}
```