Project 1: Implied Volatility
IOE 553 Financial Engineering II

February 16, 2017

Dongxu Li
UMID: 73534478

Pattarapol Koosalapeerom
UMID: 38787086

Introduction

Volatilities play an important role in financial engineering and especially in the valuation of various forms of options. In the Black and Scholes (1973) model option prices are represented in terms of the following variables

- S_0: the current price of the underlying asset
- K: the option's strike price
- T: the option's time to expiration
- r: risk-free interest rate
- σ: the volatility of the underlying asset

Data Description

In this project, we analyze the implied volatility of the call options for Alphabet Inc (Ticker: GOOG US Equity), also known as Google. All the data are retrieved from the Bloomberg Terminal on Feb 15, 2017 and from now on, the trading date is set to be Feb 14, 2017. Below are the examples of the data we selected for this project.

GOOG 2/17/17 call option			**GOOG 3/17/17 call option**		
Strike	**T** (days)	**Option Price**	**Strike**	**T** (days)	**Option Price**
810	3	11	800	31	27.83999634
812.5	3	11.30000019	805	31	25
815	3	6.100000381	810	31	19.25
817.5	3	4.949999809	815	31	16.30000305
820	3	3.699999809	820	31	13.67000008
822.5	3	2.539999962	825	31	11.60000038
825	3	1.899999619	830	31	8.75
827.5	3	1.050000191	835	31	6.940000534
830	3	0.680000007	840	31	6.199999809
832.5	3	0.899999976	845	31	5

All together there are 50 data points. They are generated in a table format where different row corresponds to each call option with different maturity date, strike price, and the last price of the option. Note that, all of them are call options. There are 9 missing data from the column of the option's last price. We replaced these by the mid price, an average of the bid and ask prices being quoted. The closing price of GOOG US Equity as of Feb 14, 2017 is $S_0 = 820.45$. For simplicity, we set the default risk-free interest rate to be $r = 2\%$.

Analysis

1. Implied volatility

First of all, we import the data table into MATLAB and iteratively use the function **blsimpv()** to calculate the implied volatility for each option.

$$\text{Implied Volatility} = \textbf{blsimpv}(Price, Strike, Rate, Time, Value)$$

The MATLAB code is shown below

```
N = 50;
imp_vol = zeros(N,1);
r = 0.02;
s = 820.45;
for i = 1:N
    Volatility = blsimpv(s, strike(i), r, T(i)/365, price(i));
    imp_vol(i) = Volatility;
end
```

We store the volatility in the vector imp_vol. The simple average of the imp_vol is 15.56%.

2. Historical volatility

In this section, we estimate the historical volatility from a stock price by computing the 90-day standard deviation of stock prices from Aug 15, 2016 to Feb 15, 2017. $Std(stock\ price) \times \sqrt{365} = 18.3\%$. The difference between the implied volatility and the historical volatility suggest that the market expecting volatility to be lower than the past.

Furthermore, we also generate the 90-day volatility data from the Bloomberg Terminal for the period from Aug 15, 2016 to Feb 15, 2017. They are calculated based on 90-day historical data of the last price. The line plot of the data is shown below,

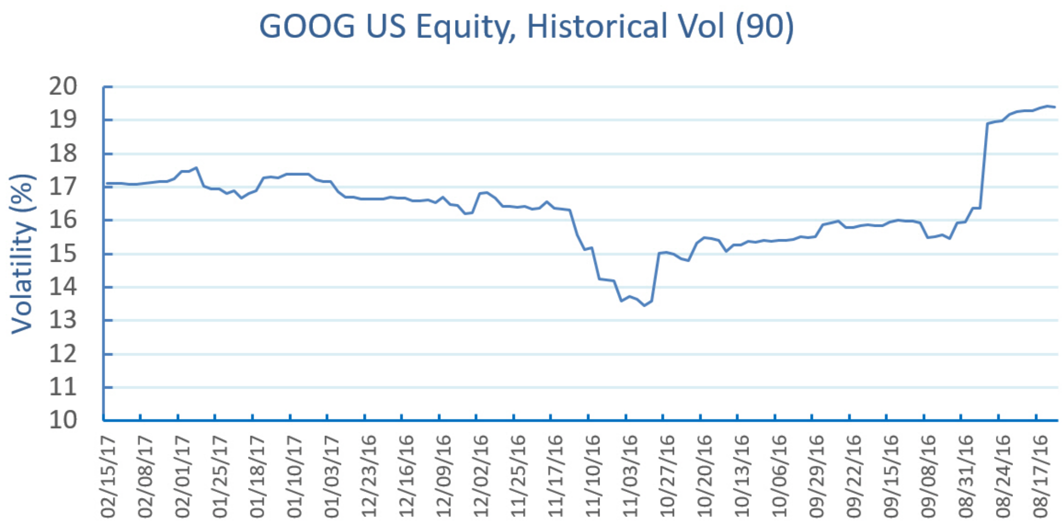


Figure 1: 90-day Historical Volatility of GOOG US Equity

3. Implied Volatility Surface

We construct the implied volatility surface as a function of strike price and time to expiration then we use the MATLAB Toolbox to generate a 3D plot. The results are shown below

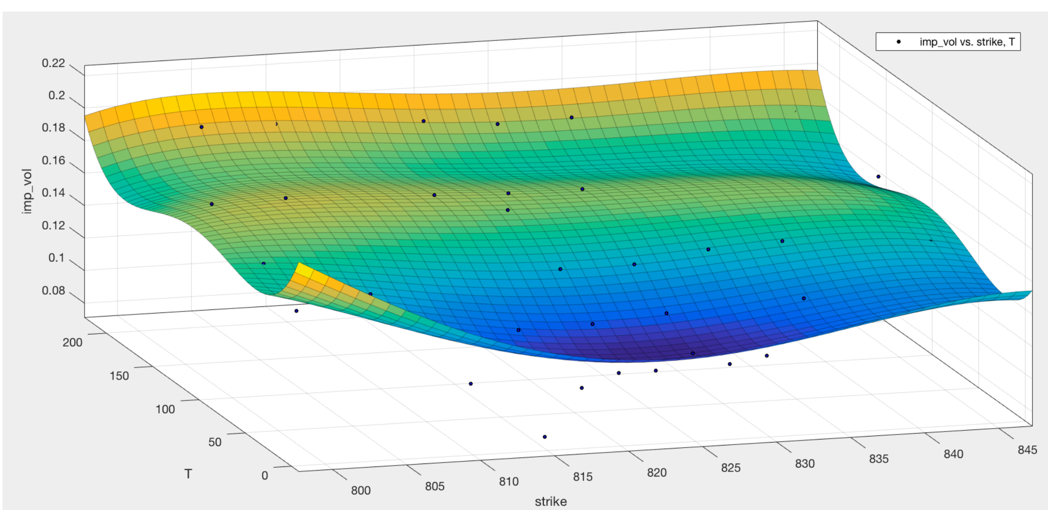


Figure 2: Implied Volatility Surface (1)

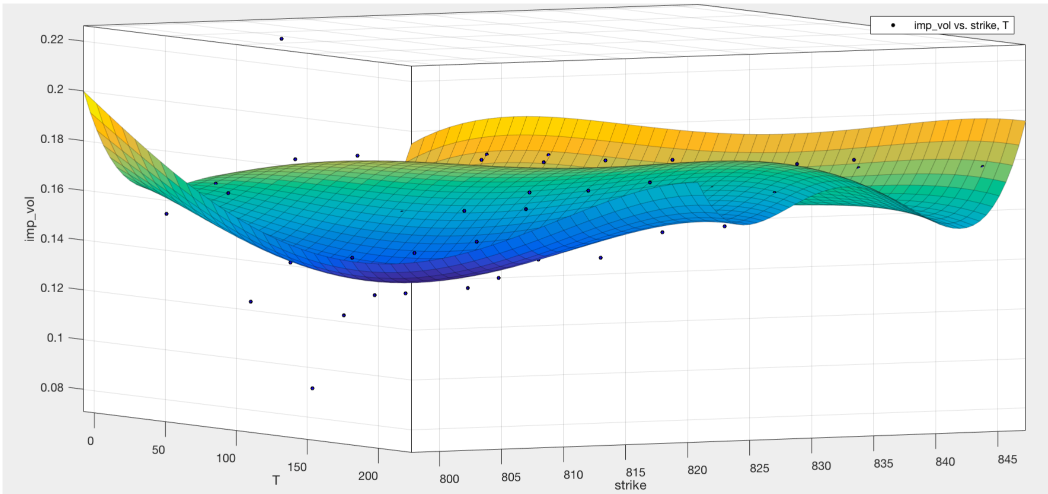


Figure 3: Implied Volatility Surface (2)

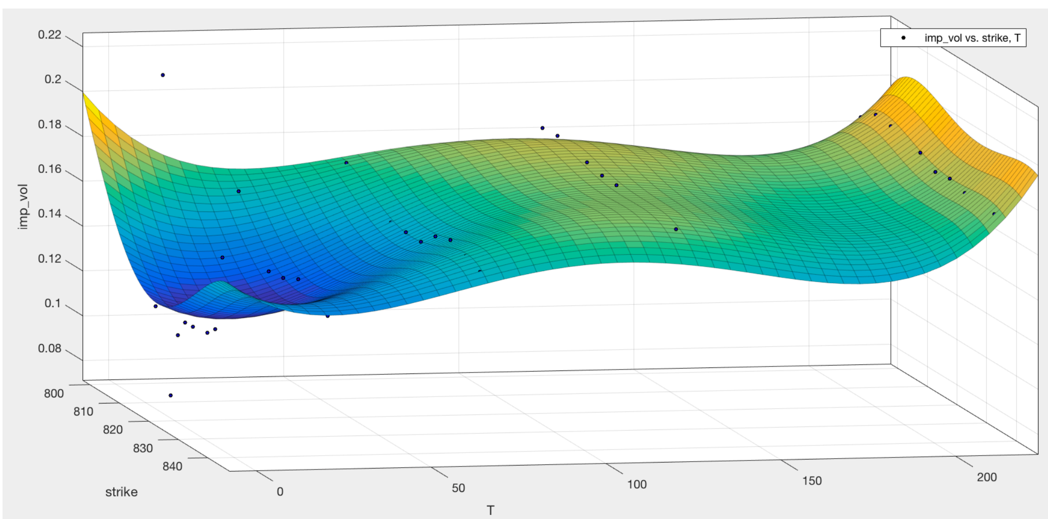


Figure 4: Implied Volatility Surface (3)

We use the 4th-degree polynomials to construct and parameterize the surface using MATLAB Toolbox. Below is the result of the surface function which we can conveniently use to approximate the data point:

$$\begin{aligned} f(x,y) = & p_{00} + p_{10}x + p_{01}y + p_{20}x^2 + p_{11}xy + p_{02}y^2 + \\ & p_{30}x^3 + p_{21}x^2y + p_{12}xy^2 + p_{03}y^3 + \\ & p_{40}x^4 + p_{31}x^3y + p_{22}x^2y^2 + p_{13}xy^3 + p_{04}y^4 \end{aligned}$$

Coefficients (with 95% confidence bounds):

$p_{00} = -2.583e + 04(-1.3\text{e+05}, 7.835\text{e+04})$

$p_{10} = 126.6(-380.1, 633.4)$

$p_{01} = -5.782(-27.04, 15.48)$

$p_{20} = -0.2327(-1.157, 0.6915)$

$p_{11} = 0.02011(-0.05729, 0.09752)$

$p_{02} = 0.002645(-0.003462, 0.008753)$

$p_{30} = 0.00019(-0.0005592, 0.0009392)$

$p_{21} = -2.328\text{e-05}(-0.0001173, 7.072\text{e-05})$

$p_{12} = -6.261\text{e-06}(-2.105\text{e-05}, 8.533\text{e-06})$

$p_{03} = -4.561\text{e-07}(-2.574\text{e-06}, 1.662\text{e-06})$

$p_{40} = -5.813\text{e-08}(-2.858\text{e-07}, 1.696\text{e-07})$

$p_{31} = 8.961e\text{e-09}(-2.912\text{e-08}, 4.705\text{e-08})$

$p_{22} = 3.737\text{e-09}(-5.276\text{e-09}, 1.275\text{e-08})$

$p_{13} = 3.071\text{e-10}(-2.241\text{e-09}, 2.855\text{e-09})$

$p_{04} = 4.889\text{e-10}(-6.331\text{e-10}, 1.611\text{e-09})$

Goodness of fit:

SSE : 0.01247

R-square : 0.6538

Adjusted R-square : 0.5153

RMSE : 0.01887

Prediction

From the previous section, we have generated a polynomial surface for implied volatility using 4th degree polynomials. Now we will discuss on how can we adapted the function so that we can implement time series analysis to predict the change of the parameters of the surface from this polynomial.

To achieve this, we create a partition, $0 = t_0 < t_1 < ... < t_n = T$ and $||\Pi|| = \Delta t$, for the period for which we have the data.

Observe that for each time period, there is a corresponding surface, denoted by $f(x, y)^k$, with the corresponding parameters p_{ij}. Our goal is to predict the change in p_{ij}. Rewrite each of p_{ij} in a matrix form as

$$M_n = \begin{bmatrix} p_{11}^n & p_{12}^n & p_{13}^n & p_{14}^n \\ p_{21}^n & p_{22}^n & p_{23}^n & p_{24}^n \\ p_{31}^n & p_{32}^n & p_{33}^n & p_{34}^n \\ p_{41}^n & p_{42}^n & p_{43}^n & p_{44}^n \end{bmatrix}$$

where n denotes the time period.

Let $(M_k)_{k=1}^n$ be the sequence of these matrices. Therefore, we obtain a time-series like data which can be predicted using time series techniques, for example the AR (Autoregressive) model or MA (Moving Average) model.

Project 2: Modeling of Yield Curve Dynamics
IOE 553 Financial Engineering II

March 23, 2017

Dongxu Li
UMID: 73534478

Pattarapol Koosalapeerom
UMID: 38787086

Introduction

Modeling short rates and bond prices are both crucial and challenging as they are the basis for economic analysis and pricing model for many kinds of securities, but yet are one of the most difficult object to model. Historically, many academics have been proposed numerous models for short rates under the risk-neutral space, beginning with Merton's model (1973),

$$r_t = r_0 + at + \sigma W_t^{\mathbb{Q}}$$

where $W_t^{\mathbb{Q}}$ is a one-dimensional Brownian motion under the martingale measure $\mathbb{Q}$. At the present, there are several standard models that have been used and studied frequently including Vasiček (1977), Dothan (1978), Cox–Ingersoll–Ross (1985), Ho–Lee (1986), Hull–White (1990, 1993), Black–Derman–Toy (1990).

For this project, we selected Vasiček, Cox–Ingersoll–Ross (CIR), and Hull-White models to model the bond price function given the observed data from the market. The important property for these 3 models is that they satisfy the affine term structure, implying that we are able to derive a close form formula for bond prices corresponding to the model. The followings are important settings for each models. Note also that from now on we will write W_t as a one-dimensional Brownian motion under the martingale measure.

Vasiček Model $\quad dr_t = (b - ar_t)dt + \sigma dW_t$

Corresponding term structure:

$$\begin{aligned} p(t,T) &= e^{A(t,T)-B(t,T)r_t} \\ B(t,T) &= \frac{1}{a}(1 - e^{a(T-t)}) \\ A(t,T) &= \frac{(B(t,T) - T + t)(ab - \frac{1}{2}\sigma^2)}{a^2} - \frac{\sigma^2 B^2(t,T)}{4a} \end{aligned}$$

CIR Model $\quad dr_t = a(b - r_t)dt + \sigma\sqrt{r}dW_t$

Corresponding term structure:

$$\begin{aligned} p(t,T) &= A(t,T)e^{-B(t,T)r_t} \\ B(t,T) &= \frac{2(e^{\gamma x} - 1)}{(\gamma +)(e^{\gamma x} - 1) + 2\gamma} \\ A(t,T) &= \left[\frac{2\gamma e^{(a+\gamma)(\frac{x}{2})}}{(\gamma + a)(e^{\gamma x} - 1) + 2\gamma}\right]^{\frac{2ab}{\sigma^2}} \\ \gamma &= \sqrt{a^2 + 2\sigma^2} \end{aligned}$$

Hull-White Model $dr_t = (\Theta_t - ar_t)dt + \sigma dW_t$

Corresponding term structure:

$$p(t,T) = \frac{p^*(t,T)}{p^*(0,t)} \exp\left(B(t,T)f^*(0,t) - \frac{\sigma^2}{4a}B^2(t,T)(1-e^{-2at}) - B(t,T)r_t\right)$$

$$B(t,T) = \frac{1}{a}(1-e^{a(T-t)}) \quad , \quad p^* \text{is an observed price}$$

Data Description

For this project, we use close prices of U.S. government treasury coupon STRIPS (Separate Trading of Registered Interest and Principal of Securities). One of the benefits for using STRIPS is that we can immediately represent these data as a zero-coupon bonds without using bootstrapping technique. During the work for this project, the authors found that getting zero-coupon treasury bonds with TTM greater than 1 year is a crucial building block for modeling a bond price.

The data are retrieved in a tabular format with a range of time to maturities (TTM) from 2 months to 30 years. Moreover, we also retrieved prices from 7 different pricing dates with an equal interval of 7 days beginning from 31 Jan, 2017 i.e. 31 Jan, 7 Feb, 14 Feb etc. We will use these data later on when we analyze the stability of the parameters generated from the model. All the data are retrieved from the Bloomberg Terminal on Mar 17, 2017. Below are the examples of the data we selected for this project.

U.S. Govt STRIPS Last Price

Maturity	as of Jan 31, 17	as of Feb 2, 17	as of Feb 14, 17
Mar 31, 17	99.896	99.918	99.918
Apr 15, 17	99.896	99.888	99.888
Apr 30, 17	99.838	99.857	99.857
May 15, 17	99.844	99.864	99.864
May 31, 17	99.776	99.792	99.792
Jun 15, 17	99.746	99.761	99.761
Jun 30, 17	99.751	99.772	99.772
Jul 15, 17	99.678	99.691	99.691
Jul 31, 17	99.641	99.649	99.649
Aug 15, 17	99.616	99.654	99.654

All together there are 1,575 data points consisting of 7 pricing dates where each of them includes 225 bonds with different TTM.

Model fitting

First of all, we import the data table from .csv file into python's Jupyter notebook and converted them into numpy array. To fit the model with the observed data, we utilize *curve_fit* method from SciPy's optimization library.

$$\textbf{scipy.optimize.curve\_fit}(f, xdata, ydata)$$

It uses non-linear least squares to fit a function, f, to data. The full python code is shown in the Appendix. Below are the results we obtained.

Vasiček Model

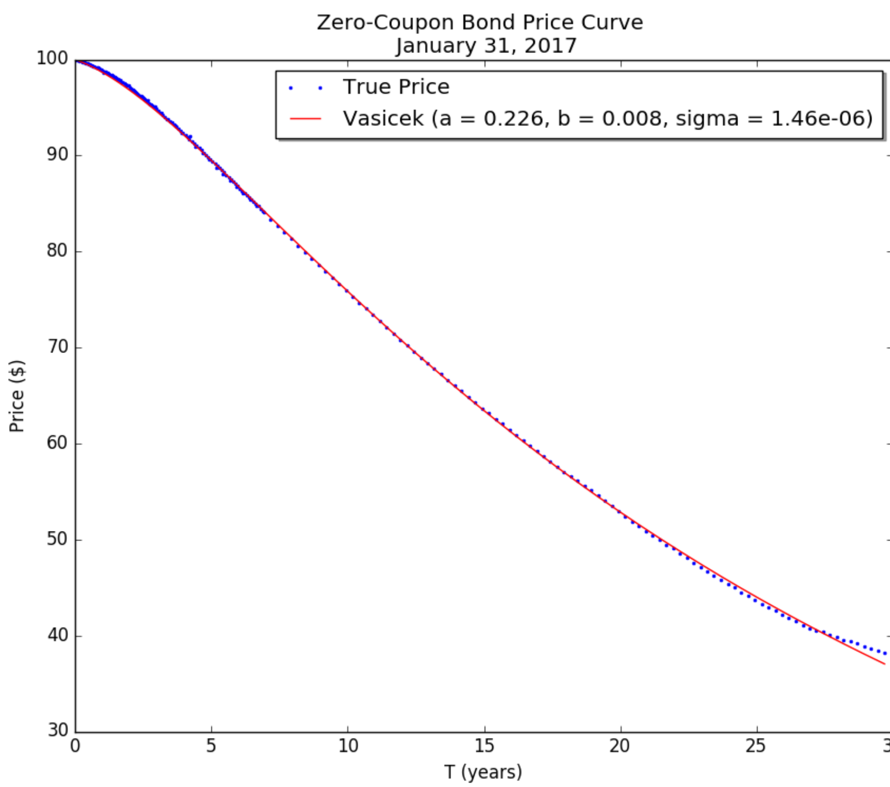


Figure 1: Vasiček Model Fitting, as of Jan 31, 2017

CIR Model

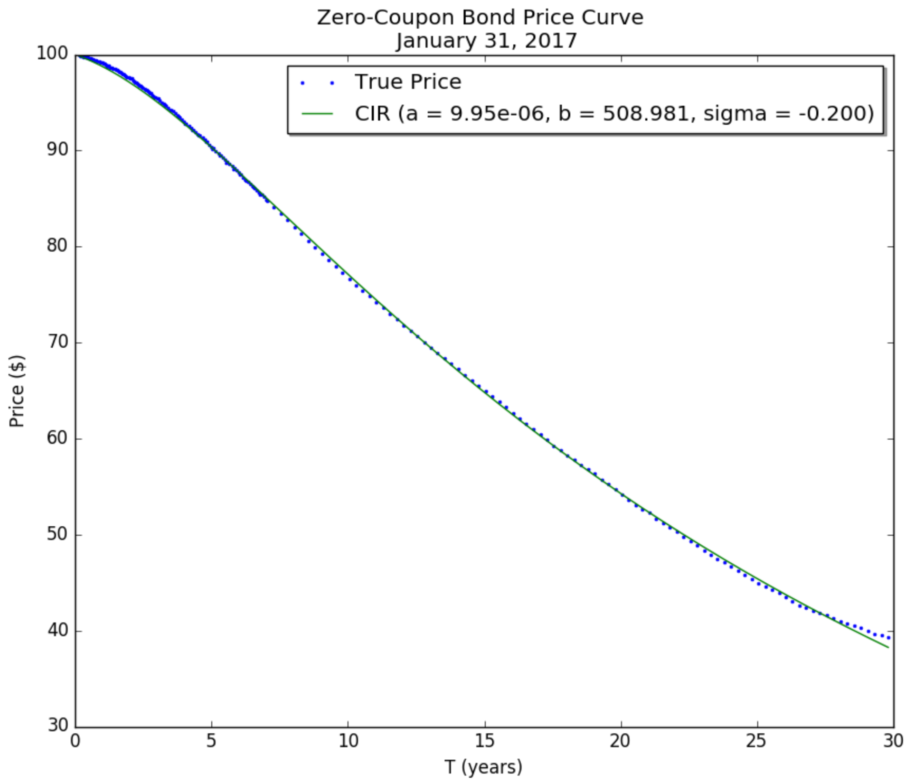


Figure 2: CIR Model Fitting, as of Jan 31, 2017

Hull-White Model

For the Hull-White model, we need a different approach to fit the observed price since the model's purpose is to estimate the forward price i.e. $t > 0$. If we model the today spot price $p(0, T)$, the result will be trivial i.e. the theoretical price will be exactly identical to the data, which is uninteresting. Hence, for this model, we estimate the forward price instead. The followings are the complications we faced during the analysis.

1. $f^*(0, t)$. We estimate $f^*(0, t)$ by using the fact that

$$\begin{aligned} f(0,t) &= -\frac{\partial \log p(0,T)}{\partial T} \\ &= -\left(\frac{\log p(0,T_{i+1}) - p(0,T_i)}{T_{i+1} - T_i}\right) \end{aligned}$$

Using MATLAB, we get the function of $f(0, t)$ as following

$$f(0, t) = 0.7684t^{-0.96}$$

2. $p^*(0, T)$. Since we need to get the price for <u>all</u> T in order to fit the model, we decided to use Vasiček Model to approximate this.

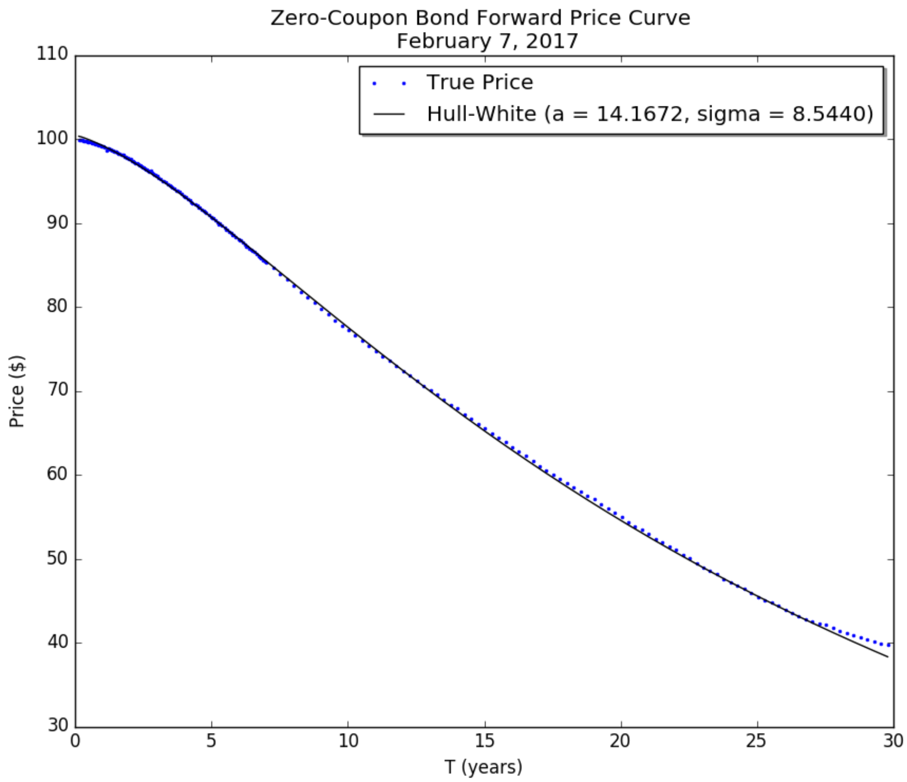


Figure 3: Hull-White Model Fitting, as of Feb 7, 2017

Stability

In this part, we utilize the rest of the data and apply the previous method to obtain parameters corresponded to each pricing dates. The results are shown below.

Vasiček Model parameters

Pricing date	a	b	σ
Jan 31, 17	0.2262	0.0082	1.4573e-06
Feb 7, 17	0.2097	0.0076	-1.9011e-06
Feb 14, 17	0.2417	0.0087	6.0190e-06
Feb 21, 17	0.2274	0.0082	1.2428e-06
Feb 28, 17	0.2224	0.0079	9.7369e-07
Mar 7, 17	0.2504	0.0091	1.4738e-05
Mar 14, 17	0.2751	0.0101	1.6925e-06
Std deviation	0.0217	0.0008	5.93e-06

CIR Model parameters

Pricing date	a	b	σ
Jan 31, 17	9.9520e-06	508.9811	-0.2004
Feb 7, 17	1.0024e-05	467.8315	-0.1867
Feb 14, 17	1.1813e-05	450.5133	-0.2124
Feb 21, 17	1.2954e-05	390.1090	0.2011
Feb 28, 17	1.3634e-05	356.8418	-0.1961
Mar 7, 17	0.0001	41.6972	0.2210
Mar 14, 17	4.5025e-05	137.9830	0.2418
Std deviation	3.37e-05	177.7647	0.2250

Hull-White Model parameters

Pricing date	a	σ
Feb 7, 17	14.1672	8.5440
Feb 14, 17	88.8060	47.6780
Feb 21, 17	33.9378	15.9506
Feb 28, 17	16.8112	9.3481
Mar 7, 17	3.4263	6.8282
Mar 14, 17	1.6780	4.8015
Std deviation	32.6553	16.1958

We can see that the parameters from Vasiček model are quite stable through time i.e. the standard deviation of parameters are small. However, for CIR and Hull-White model, the standard deviations are large for some parameters i.e. parameter b of CIR (177.7647) and a of Hull-White (32.6553).

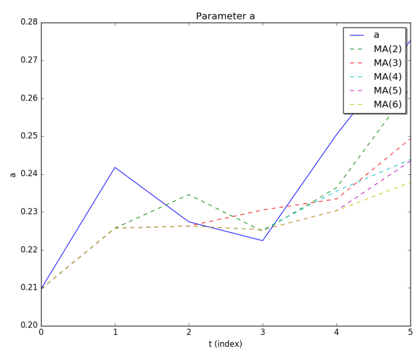
Prediction

From the previous section, we have generated a time series of 3-dimensional vector of parameters denoted by

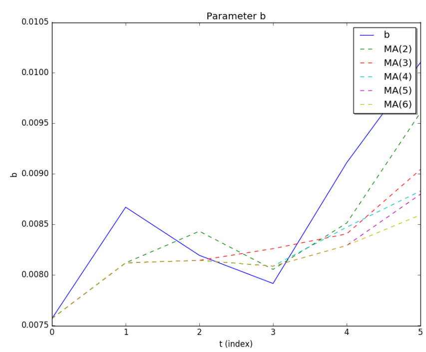
$$v_{t_n} = (a_{t_n}, b_{t_n}, \sigma_{t_n}) \quad n = 0, 1, ..., 6$$

where t_0 represents the pricing date Jan 31, 17. t_1 represents the pricing date Feb 7, 17 and so on. Furthermore, we also obtained the stability of the parameters and conclude that the Vasiček model is stable. Therefore, in this section we choose to apply time series analysis to predict the parameters of only for the Vasiček model.

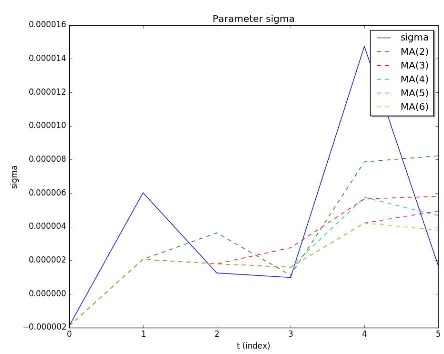
There are several time series models available including MA, AR, ARMA, ARIMA, ARCH, GARCH, to name a few. Since the number of data we have are limited, we choose the simple moving (running) average to predict the parameters of the next pricing data. Below are the plots for moving average fit associate with number of periods.



(a) Parameter a



(b) Parameter b



(c) Parameter σ

Figure 4: Parameters fit using moving average

After observing the plots, we decided to choose 2-period moving average to predict the next period parameters. The result is shown below.

Vasiček Model predicted parameters

a	b	σ
0.2628	0.0096	8.2e-06

Trading Strategy

Once we know the predicted price of bond for the next period, it's simple to form a trading strategy. Using the predicted parameters we obtained from the previous part, we can compute the predicted bond price of the next period as shown below.

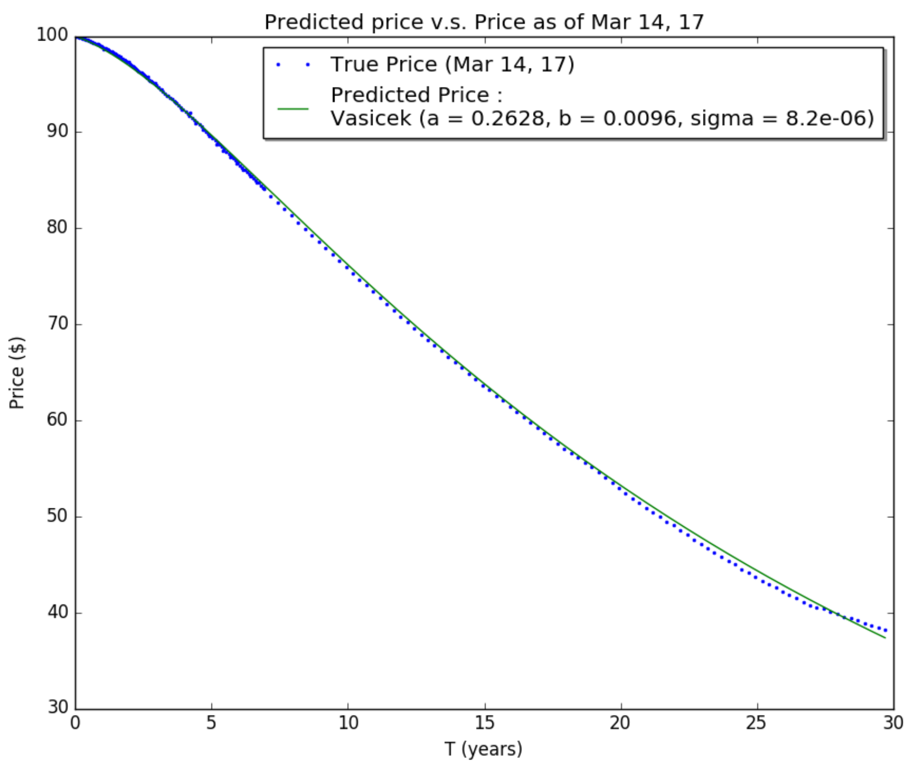


Figure 5: Predicted price (Vasicek) v.s. Price as of Mar 14, 2017

<u>Strategy</u> : For each of the T-bond, if the predicted price (green line in the figure 5) is higher than the current price (blue dots), we buy the bond. If otherwise for the range of T-bonds that the green line below the blue dots, we sell the bond.

Project 3: Option Pricing under Stochastic Interest Rates

IOE 553 Financial Engineering II

April 18, 2017

Dongxu Li

UMID: 73534478

Pattarapol Koosalapeerom

UMID: 38787086

Introduction

The Black-Scholes-Merton formula (1973) is one of the most famous and widely used in option pricing. However, the assumptions of the model still not completely complying with the observed data from the real-world market. Under the basic Black-Scholes model, it is assumed that the stock price dynamic follows geometric Brownian motion with constant drift rate and volatility, and more importantly the constant risk-free rate. It is clear that the assumption about risk-free rate is not the case in reality. The Black-Scholes-Merton formula for European call option is given by

$$C(S_t, t) = S_t N(d_1) - K e^{-r(T-t)} N(d_2)$$

$$d_1 = \frac{1}{\sigma\sqrt{T-t}} \left[ln\left(\frac{S_t}{K}\right) + \left(r + \frac{\sigma^2}{2}(T-t)\right) \right]$$

$$d_2 = d_1 - \sigma\sqrt{T-t}$$

- $N(\cdot)$ is the cumulative distribution function of the standard normal distribution
- $T - t$ is the time to maturity (expressed in years)
- S_t is the spot price of the underlying asset
- K is the strike price
- r is the risk free rate (annual rate, expressed in terms of continuous compounding)
- σ is the volatility of returns of the underlying asset

The goal of this project is to study the option pricing under the inconstant interest rate environment and compare it with the classic Black-Scholes-Merton formula. The formula we use to price an option under dynamic short rate is the General Option Pricing formula. It incorporates the short rates dynamic through the pricing of T-bond $p(0, T)$. The formula is given by

$$\Pi(0; X) = S_0 \mathbb{Q}^S(S_T \geq K) - K p(0, T) \mathbb{Q}^T(S_T \geq K)$$

where X is a claim, in this case a call option, $\mathbb{Q}^T$ denotes the T-forward measure, and $\mathbb{Q}^S$ denotes the martingale measure for the numeraire process $S(t)$. Also, define

$$Z_t = \frac{S_t}{p(t, T)}$$

$$dZ_t = Z_t m_t dt + Z_t \sigma_t dW_t$$

then finally we have,

$$\mathbb{Q}^S(S_T \geq K) = N(d_1)$$
$$\mathbb{Q}^T(S_T \geq K) = N(d_2)$$

where

$$d_2 = \frac{\ln\left(\frac{S_0}{Kp(0,T)}\right) - \frac{1}{2}\int_0^T \sigma_t^2 dt}{\sqrt{\int_0^T \sigma_t^2 dt}}$$
$$d_1 = d_2 + \sqrt{\int_0^T \sigma_t^2 dt}$$

Note that there are tons of short rate models to choose from, however, in order to get a closed form formula for $p(0,T)$, it is necessary to choose the short rate model that satisfy an Affine Term Structure. For example, Vasiček (1977), Cox–Ingersoll–Ross (1985).

Assumptions

We begin by making an assumption about the form of stock price and short rate dynamics. In this project, we assume the stock price follows a geometric Brownian motion with constant drift and volatility. It is then given by

$$dS_t = \mu S_t dt + \sigma_s S_t dW_t$$

where $\mu, \sigma \in \mathbb{R}$ and W_t a standard one dimensional Wiener process.
For the short rate, we choose Vasiček model as an underlying dynamic, given by

$$dr_t = (b - ar_t)dt + \sigma_r dW_t$$

where a, b, and $\sigma_r \in \mathbb{R}$. The corresponding term structure is then in the following form

$$\begin{aligned}
p(t,T) &= e^{A(t,T)-B(t,T)r_t} \\
B(t,T) &= \frac{1}{a}(1 - e^{a(T-t)}) \\
A(t,T) &= \frac{(B(t,T) - T + t)(ab - \frac{1}{2}\sigma^2)}{a^2} - \frac{\sigma^2 B^2(t,T)}{4a}
\end{aligned}$$

Note further that r_t in the term structure formula is also assumed to be fixed at 1%.

Data Description

We choose to study the pricing of the call option on Apple Incorporation stock (AAPL). The data consist of historical daily price of AAPL of roughly one year from April 18, 2016 to April 13, 2017. The close stock price as of April 13, 2017 is $141.05. For the call option on AAPL, we select the in-the-money call option with a strike $140. Also, we choose to analyze 4 different time to maturities of the option including May 12, July 21, December 15, 2017, and Jan 19, 2018. Both the stock price and option price data are retrieved from Yahoo! Finance website on April 14, 2017.

Finally, to fit the term structure of a short rate model, we retrieved the Treasury STRIP price data from Wall Street Journal website on April 14, 2017 including 70 data with various time to maturity from 1 month to 30 years. The data come with Bid and Ask prices. Hence, for simplicity, the Mid prices are computed and used throughout the project. Below are the example of the data we select.

U.S. Govt STRIPS Price

Maturity	as of Apr 14, 17
May 15, 17	99.939
Aug 15, 17	99.732
May 15, 18	98.890
Nov 15, 18	98.364

Call Options on AAPL with Strike $140

Maturity	Price ($)
May 12, 17	4.25
Jul 21, 17	6.1
Dec 15, 17	4.9
Jan 19, 18	11.15

AAPL Close Price

Date	Price ($)
Apr 13, 17	141.05
Apr 12, 17	141.8
Apr 11, 17	141.63
Apr 10, 17	143.17
Apr 7, 17	143.34
Apr 6, 17	143.66
Apr 5, 17	144.02
Apr 4, 17	144.77
Apr 3, 17	143.7

Model fitting

Stock Price

To find the stock dynamic, we need to compute μ and σ. First, we prepare the time series of daily log returns then calculate the simple mean and (scaled) standard deviation per year. The results are $\mu = 0.1087\%$ and $\sigma = 1.2108\%$.

Short Rates

We import the STRIP data table from .csv file into Python's Jupyter notebook and converted them into numpy array. We then utilizing *curve_fit* method from SciPy's optimization library to fit the Vasiček model. We obtain $a = 0.1028, b = 0.0048, \sigma = -0.0143$.

Stock Price under the Numeraire $p(t, T)$

Denote $Z_t = \frac{S_t}{p(t,T)}$. To get the dynamic of Z_t, we apply Ito's formula knowing that $p(t, T)$ has the form $e^{A(t,T)-B(t,T)r_t}$ where A, B are deterministic and $r_t \equiv 1\%$. We get

$$\begin{aligned} dZ_t &= \frac{1}{p(t,T)}dS_t - \frac{S_t}{p(t,T)^2}dp(t,T) + \frac{S_t}{p(t,T)^3}(dp(t,T))^2 - \frac{1}{p(t,T)^2}dS_t dp(t,T) \\ &= \{...\}Z_t dt + \Big\{\sigma_s + \frac{\sigma_r}{a}\big(1 - e^{-a(T-t)}\big)\Big\}Z_t dW_t \end{aligned}$$

Now we have the volatility function σ_t for Z_t which is the important step in estimating option price using the general option pricing formula.

Analysis

Black-Scholes-Merton formula

Fist we analyze the theoretical call option price using Black-Scholes-Merton formula. Note that we fix the risk-free rate at 1%. The results are shown below.

Black-Scholes-Merton formula

Maturity	Theoretical Price	Observed Price	Absolute Difference
May 15, 17	4.166	4.25	0.084
Aug 15, 17	5.372	6.1	0.728
May 15, 18	7.438	4.9	2.583
Nov 15, 18	11.997	11.15	0.847

General Option Pricing formula

Next, we analyze the theoretical call option price using General Option Pricing formula. The difficulty here is to compute the definite integral $\int_0^T \sigma_t^2 dt$. We get around this by utilizing numerical integration using Python's *numpy.quad* function. Below are the results.

General Option Pricing formula

Maturity	Theoretical Price	Observed Price	Absolute Difference
May 15, 17	4.194	4.25	0.056
Aug 15, 17	5.379	6.1	0.721
May 15, 18	7.405	4.9	2.505
Nov 15, 18	11.828	11.15	0.678

The results from General Option Pricing formula are promising. For every maturity date we choose, the theoretical price are closer to the observed price than the Black-Scholes-Merton formula. Hence, we can conclude that the general option pricing model explain option prices better.

Implied Volatility

We define implied volatility by $\sqrt{\int_0^T \sigma_t^2 dt}$. Then, we compute implied volatility based on both Black-Scholes-Merton and General Option Pricing formula. Note that, there is no explicit formula for this. By numerical method, we can obtain the implied volatility as follow. (Here, we denote BS as Black-Scholes-Merton, and GOP as General Option Pricing formula)

Implied Volatility

Maturity	BS	GOP
May 15, 17	0.237	0.0648
Aug 15, 17	0.2671	0.097
May 15, 18	0.3514	0.1817
Nov 15, 18	0.221	0.186

To decide if the implied volatility from General Option Pricing give us any improvement, we use the implied volatility we obtained to compute the price of the option with a different strike and then compare which one give a more accurate price. Hence, we we will do this scheme for the strike \$135. The results are shown below.

Theoretical Price (Strike \$135) using the Implied vol

Maturity	BS	GOP	Observed Price
May 15, 17	7.46	7.46	8.06
Aug 15, 17	8.504	9.045	9.3
May 15, 18	10.422	13.517	13.2
Nov 15, 18	14.858	14.152	13.85

Overall, the General Option Pricing formula performs better than the Black-Scholes-Merton. We can see that for the maturity May 15, 17, both formulas give the same result. However, the other three, the prices from General Option Pricing formula are closer to the observed price. We can say that the implied volatility from General Option Pricing formula slightly improve the accuracy from Black-Scholes-Merton's implied volatility.

The Greeks

Delta, Δ, measures the rate of change of the theoretical option value with respect to changes in the underlying asset's price. Mathematically, delta is the first derivative of the value of the option (V) with respect to the underlying instrument's price (S), i.e. $\frac{\partial V}{\partial S}$. The close form formula for both Black-Scholes-Merton and General Option Pricing formula are given by

$$\Delta = N(d_1)$$

Gamma, Γ , measures the rate of change in the delta with respect to changes in the underlying price. Mathematically, gamma is the second derivative of the value function with respect to the underlying price i.e. $\frac{\partial^2 V}{\partial S^2}$. For Black-Scholes-Merton formula, the close form formula for gamma is given by

$$\Gamma = \frac{N(d_1)}{S\sigma\sqrt{\tau}}$$

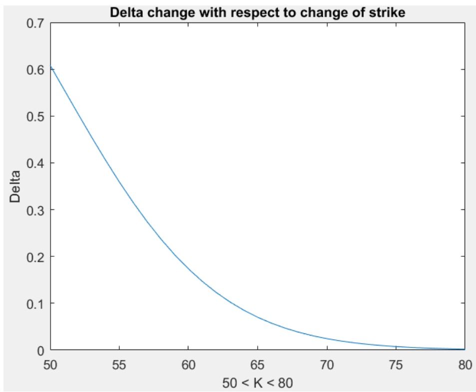
For General Option Pricing formula, the gamma is given by

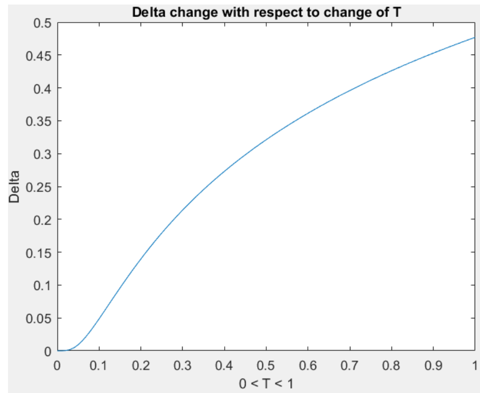
$$\Gamma = e^{-\frac{d_1^2}{2}} \left[\frac{1}{S_0\sqrt{\int_0^T \sigma_t^2 dt}} \right]$$

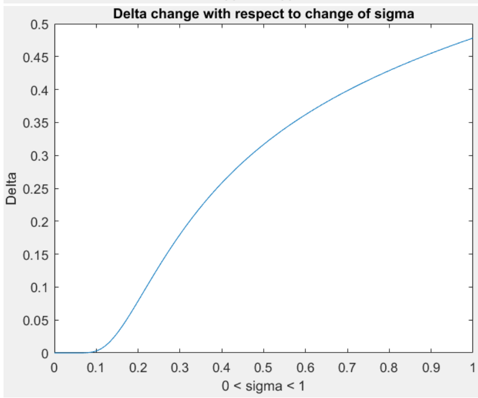
The objective for this part is to analyze the change of Δ and Γ with respect to the change of other parameters. These include time to maturity $\boldsymbol{T}$, strike $\boldsymbol{K}$, volatility $\boldsymbol{\sigma}$. To be precise, we wish to see the movement of

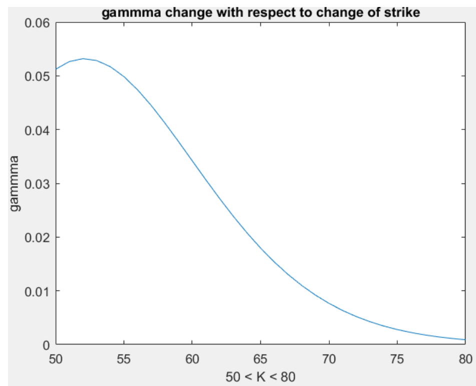
$$\frac{\partial \Delta}{\partial T}, \frac{\partial \Delta}{\partial K}, \frac{\partial \Delta}{\partial \sigma}$$
$$\frac{\partial \Gamma}{\partial T}, \frac{\partial \Gamma}{\partial K}, \frac{\partial \Gamma}{\partial \sigma}$$

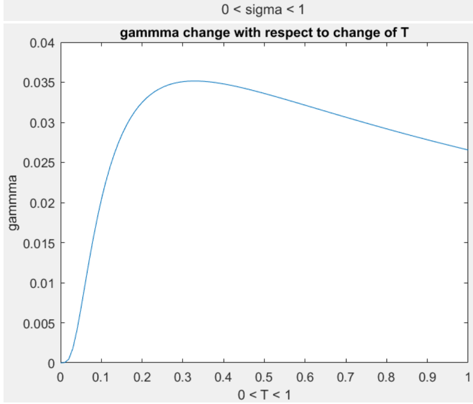
Below are the results we obtained from MATLAB.

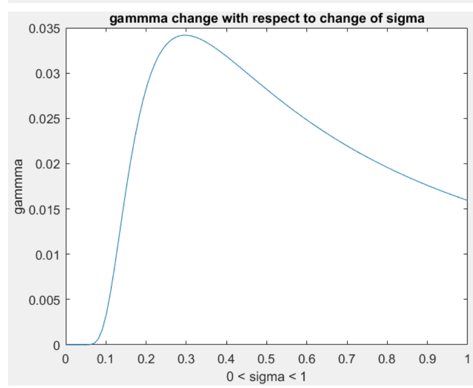

Delta change with respect to change of strike
Delta
50 < K < 80


Delta change with respect to change of T
Delta
0 < T < 1


Delta change with respect to change of sigma
Delta
0 < sigma < 1


gammma change with respect to change of strike
gammma
50 < K < 80


gammma change with respect to change of T
gammma
0 < T < 1


gammma change with respect to change of sigma
gammma
0 < sigma < 1

Appendix

Python Code

In [1]:
```
from scipy.optimize import curve_fit
import numpy as np
import matplotlib.pyplot as plt
import warnings
plt.rcParams['figure.figsize'] = (10.0, 8.0)
warnings.filterwarnings('ignore')
r = 0.01

data = np.genfromtxt('data for project 3.csv', delimiter=',')
T = data[:,0]/365
Price = data[:,1]
```

In [2]:
```
## Vasicek ##

def B_Vasicek(T, a):
    return (1-np.exp(-a*T))/a

def A_Vasicek(T, a, b, sigma):
    return (B_Vasicek(T, a)-T)*(a*b-(sigma**2)/2)/(a**2) - \
            (sigma**2)*np.power(B_Vasicek(T, a),2)/(4*a)

def Vasicek(T, a, b, sigma):
    return 100*np.exp(A_Vasicek(T, a, b, sigma)-B_Vasicek(T, a)*r)

params_Vasicek, _ = curve_fit(Vasicek, T, Price)
print('a =', params_Vasicek[0], ', b =', params_Vasicek[1], ', sigma =', params_Vasicek[2])
```

```
a = 0.102764482876 , b = 0.00483254924203 , sigma = -0.0143299375378
```

In [3]:
```
from scipy.integrate import quad

sigma2 = 0.012120785*np.sqrt(365)
sigma1 = params_Vasicek[2]
a = params_Vasicek[0]
s = 141.05
K = 140

def integrand(x, sigma2, sigma1, a, T):
    return (sigma2 + sigma1/a*(1-np.exp(-a*(T-x))))**2
```

In [4]:
```
## general option pricing formula

from scipy.stats import norm

Tc = [28/365, 49/365, 98/365, 262/365]

for T in Tc:
    sigma_T_square, err = quad(integrand, 0, T, args=(sigma2, sigma1, a, T))
    P_0_T = Vasicek(T, params_Vasicek[0], params_Vasicek[1], params_Vasicek[2])/100
    d2 = (np.log(s/(K*P_0_T)) - (1/2)*sigma_T_square)/np.sqrt(sigma_T_square)
    d1 = d2 + np.sqrt(sigma_T_square)
    General_price_0 = s* norm.cdf(d1)- K*P_0_T*norm.cdf(d2)
    print(General_price_0)
```

```
4.19431519585
5.37944633052
7.40518935026
11.8279416841
```

In [5]:
```
## black-scholes

r = 0.01;

for T in Tc:
    d1 = (np.log(s/K)+(r+1/2*(sigma2**2)*T))/sigma2/np.sqrt(T)
    d2 = d1 - sigma2*np.sqrt(T)
    Price_0 = s*norm.cdf(d1) - np.exp(-r*T)*K*norm.cdf(d2)
    print(Price_0)
```

```
4.16562919449
5.37211065263
7.43811145237
11.9969911487
```

In [13]:
```
## implied vol black-scholes

Tc = [28/365, 49/365, 98/365, 262/365]
actual = [4.25,6.1,10.9,11.5]

r = 0.01;
guess = [i/10000 for i in range(0,4000,1)]
implied_vol_BS = []
i = 0
temp = []
for T in Tc:
    BS = []
    for sigma in guess:
        d1 = (np.log(s/K)+(r+1/2*(sigma**2)*T))/sigma/np.sqrt(T)
        d2 = d1 - sigma*np.sqrt(T)
        BS.append(s*norm.cdf(d1) - np.exp(-r*T)*K*norm.cdf(d2))
    temp = [abs(n - actual[i]) for n in BS]
    i = i + 1
    vol = guess[temp.index(min(temp))]
    print(vol)
    implied_vol_BS.append(vol)
```

```
0.237
0.2671
0.3514
0.221
```

In [12]:
```
## general option pricing formula, implied vol

Tc = [28/365, 49/365, 98/365, 262/365]
actual = [4.25,6.1,10.9,11.5]
guess = [i/10000 for i in range(0,2000,1)]
K = 140
implied_vol_G = []
i = 0
temp = []
for T in Tc:
    General_price = []
    for sigma_T_square in guess:
        P_0_T = Vasicek(T, params_Vasicek[0], params_Vasicek[1],
params_Vasicek[2])/100
        d2 = (np.log(s/(K*P_0_T)) -
(1/2)*sigma_T_square)/np.sqrt(sigma_T_square)
        d1 = d2 + np.sqrt(sigma_T_square)
        General_price.append(s* norm.cdf(d1)- K*P_0_T*norm.cdf(d2))
    temp = [abs(n - actual[i]) for n in General_price]
    i = i + 1
    vol = np.sqrt(guess[temp.index(min(temp))])
    print(vol)
    implied_vol_G.append(vol)
```

```
0.0648074069841
0.0974679434481
0.181659021246
0.186010752377
```

In [8]:
```
## Compare the price with another Strike
# using the implied vol we obtained
K = 135

## general option pricing formula
i = 0
for T in Tc:
    sigma_T_square = implied_vol_G[i]**2
    P_0_T = Vasicek(T, params_Vasicek[0], params_Vasicek[1],
params_Vasicek[2])/100
    d2 = (np.log(s/(K*P_0_T)) - (1/2)*sigma_T_square)/np.sqrt(sigma_T_square)
    d1 = d2 + np.sqrt(sigma_T_square)
    General_price = s* norm.cdf(d1)- K*P_0_T*norm.cdf(d2)
    print(General_price)
    i = i + 1
```

```
7.45702612584
9.04529561731
13.516760483
14.1520269951
```

In [9]:
```
# black-scholes

i = 0
for T in Tc:
    sigma = implied_vol_BS[i]
    d1 = (np.log(s/K)+(r+1/2*(sigma**2)*T))/sigma/np.sqrt(T)
    d2 = d1 - sigma*np.sqrt(T)
    Price = s*norm.cdf(d1) - np.exp(-r*T)*K*norm.cdf(d2)
    print(Price)
```

```
7.46575485996
8.50437912963
10.4219631816
14.8580158256
```

MATLAB code

```
%% blsdelta(Price,Strike,Rate,Time,Volatility,Yield)
delta = [];
for i = 50:80
   [CallDelta, PutDelta] = blsdelta(50, i, 0.12, 0.25, 0.3, 0);
   delta = [delta,CallDelta];
end
figure(1)
x = linspace(50,80,length(delta));
plot(x,delta);
title('Delta change with respect to change of strike')
xlabel('50 < K < 80')
ylabel('Delta')

delta = [];
for i = 1:100
   [CallDelta, PutDelta] = blsdelta(50, 60, 0.12, i/100, 0.3, 0);
   delta = [delta,CallDelta];
end
figure(2)
x = linspace(0,1,length(delta));
plot(x,delta);
title('Delta change with respect to change of T')
xlabel('0 < T < 1')
ylabel('Delta')


delta = [];
for i = 1:100
   [CallDelta, PutDelta] = blsdelta(50, 60, 0.12, 0.25 , i/100, 0);
   delta = [delta,CallDelta];
end
figure(3)
x = linspace(0,1,length(delta));
plot(x,delta);
title('Delta change with respect to change of sigma')
xlabel('0 < sigma < 1')
ylabel('Delta')

%% Price,Strike,Rate,Time,Volatility,Yield
gammma = [];
for i = 50:80
   Gammma  = blsgamma(50, i, 0.12, 0.25, 0.3, 0);
   gammma = [gammma,Gammma];
end
figure(4)
x = linspace(50,80,length(gammma));
plot(x,gammma);
title('gammma change with respect to change of strike')
xlabel('50 < K < 80')
ylabel('gammma')

gammma = [];
for i = 1:100
   Gammma  = blsgamma(50, 60, 0.12, i/100, 0.3, 0);
   gammma = [gammma,Gammma];
end
figure(5)
x = linspace(0,1,length(gammma));
plot(x,gammma);
title('gammma change with respect to change of T')
xlabel('0 < T < 1')
ylabel('gammma')



gammma = [];
for i = 1:100
   Gammma  = blsgamma(50, 60, 0.12, 0.25 , i/100, 0);
   gammma = [gammma,Gammma];
end
figure(6)
x = linspace(0,1,length(gammma));
plot(x,gammma);
title('gammma change with respect to change of sigma')
xlabel('0 < sigma < 1')
ylabel('gammma')
```