Documentation for Matlab Programs on Github

1 Simulation

These programs use Monte-Carlo simulations to price various options.

1.1 monte-carlo-options.m

This programs uses the following inputs

1. S0 is the initial value of the underlying asset
2. K is the strike price. For binary options it is the value of the payoff in the future
3. g is the drift growth rate of the underlying asset
4. sig is the volatility
5. r is the rate of interest
6. expiry-days is the number of days left till the expiry of the option
7. option-type specifies the type of option to be evaluated
8. simulation-count is the number of asset paths being simulated

The option-type takes seven types of option value as string input such as

- *"european"*: Payoff is computed using the $max(K - S, 0)$ for puts and $max(S - K, 0)$ for calls
- *"lookback"*: Payoff is computed using the $max(S_{max} - S_T, 0)$ for puts and $max(S_T - S_{min}, 0)$ for calls
- *"asian"*: Payoff is computed using the $max(K - S_{average}, 0)$ for puts and $max(S_{average} - K, 0)$ for calls
- *"binary"*: Payoff is subject to certain event happening and if it does then the payoff is K otherwise it is 0.

- *"barrier"*: An example of binary option is a barrier option. A knock in barrier is triggered when asset price hits a certain benchmark and once it does the option is exercisable. Similarly, if a knock out barrier is triggered the option is no longer exercisable.
- *"double-trigger"*: Payoff is subject two events (related or unrelated) happening and if they do then the payoff is K otherwise it is 0.
- *"range"*: An example of double-trigger option is a barrier option. If the price of the underlying asset remains within a pre-specified range then it can be exercised at the end of the period with the payoff same as a European option, otherwise the payoff is 0.

1.2 spread-options.m

The spread option is computed over a basket of assets and the payoff is computed between the maximum spread between the two assets over the lifetime of the option. If the spread is less than allowable spread - a constant pre-specified maximum allowable spread - then the payoff is the maximum spread otherwise it is 0.

Correlated asset paths are simulated and the average of the spread of the paths is taken for payoff computation.

$$payoff = \begin{cases} spread_{max} & spread_{max} < spread_{allowed} \\ 0 & spread_{max} \geq spread_{allowed} \end{cases}$$

1.3 longstaff-schwartz.m

This program computes the value of American Puts using the Longstaff-Schwartz approach. This function uses several basis functions to perform regressions instead of using polyfit function as in the Bermuda Options. The process includes comparing payoff of two periods by estimation through regression. All the payoffs from each simulation is averaged to give the theoretical value of the American Put.

1.4 bermuda-options.m

This program uses the regression approach presented in the paper by Longstaff and Schwartz for evaluating Bermuda Options with a finite number of exercise times until the maturity period.

First, a number of paths for asset prices are simulated and a cash flow matrix C checks if the path is in the money at maturity. Then the program works backward in time by using regression to see whether the present value of exercise in the future is better than the current period.

Finally, the C matrix displays the best time to exercise the option for each path and has zero values for all other time steps.

1.5 heston-model.m

Heston model describes evolution of the volatility of an underlying asset and can be used to price options.

It assumes that volatility is not constant and and introduces two stochastic variables that exhibit a geometric Brownian motion.

In this program, we can simulate both stochastic processes to get the volatility of the underlying asset and the value of the asset. (Figure 1)

The parameters of the model are as follows:

1. S0 is the initial value of the underlying asset
2. V0 is the current variance of the underlying asset
3. mu is the average return of the asset
4. kappa is the rate of reversion of the volatility
5. theta is the long run mean of the volatility
6. sig is the volatility of the volatility
7. rho is the correlation of the two Stochastic Processes
8. T is the time to maturity
9. N is the number of time steps per path
10. simulation-count is the number of asset paths being simulated

The equations used to compute the Asset Price S and Variance V are:

$$S_t = S_{t-1} + \mu S_{t-1}\Delta t + S_{t-1}\sqrt{V_t}\sqrt{\Delta t}F(t) \tag{1}$$

$$V_t = V_{t-1} + \kappa[\theta - V_{t-1}]\Delta t + \sigma\sqrt{V_{t-1}}\sqrt{\Delta t}G(t) \tag{2}$$

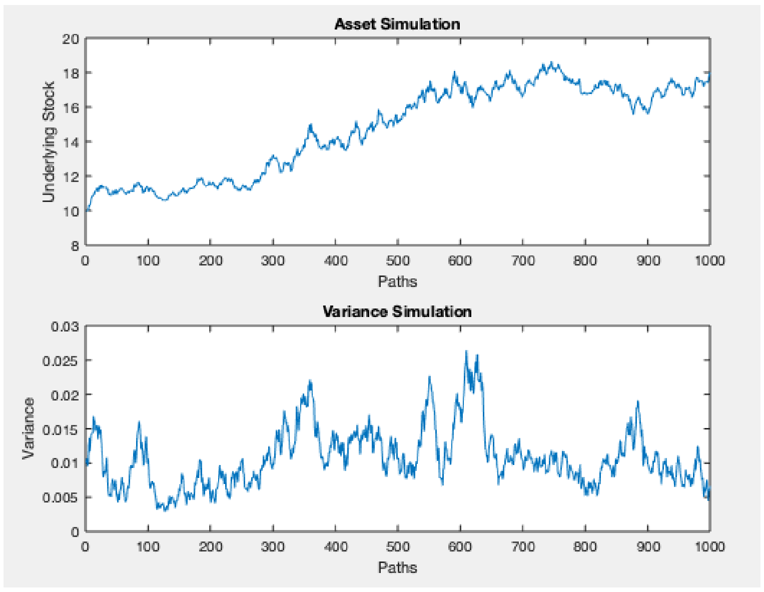

Figure 1: Heston Model Simulated

2 Numerical

A Black-Scholes Model

A.1 black-scholes-function.m

This program employs the explicit method of the finite difference scheme to compute the value of European puts and calls. The function takes in the parameters of the Black-Scholes Model and returns a surface that solves option values for all prices of underlying assets between $S-min$ and $S-max$ and times to maturity between 0 and T (Figures 2 and 3)

A.2 black-scholes-european.m

This program uses a heat transformation of the Black-Scholes Equation and then solves the Partial Differential Equation. Finally, the heat variables are converted into financial variables and a plot similar to black-scholes-fuction.m is created. (Figures 4 and 5)

A.3 black-scholes-american.m

This program extends the heat equation transformation to evaluate the American puts. (Figure 6)

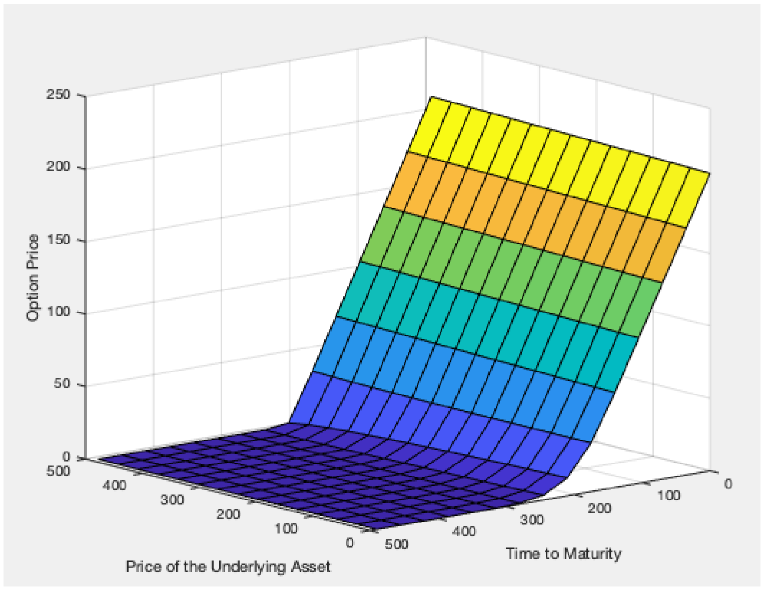

Figure 2: European Call

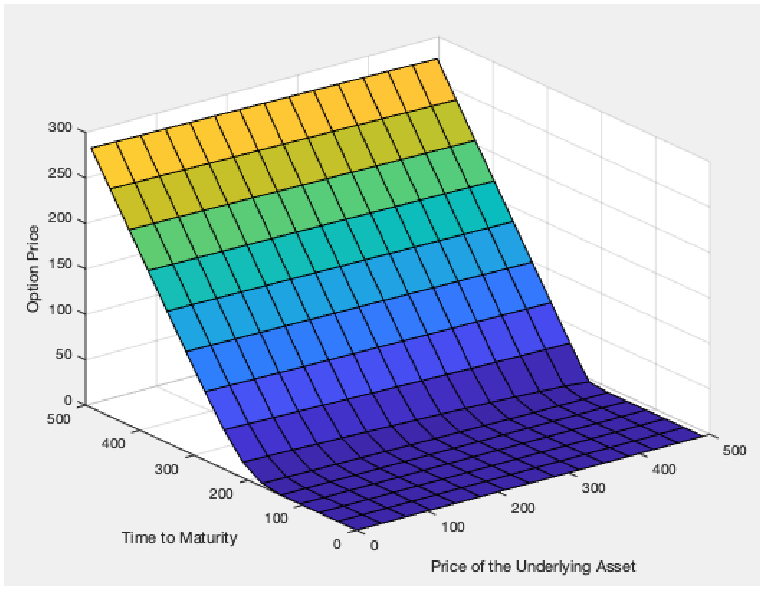

Figure 3: European Put

A.4 greeks.m

This program computes the various Greeks (Partial Derivates) associated with options

1. *Delta*: This computes the change in option price with respect to change

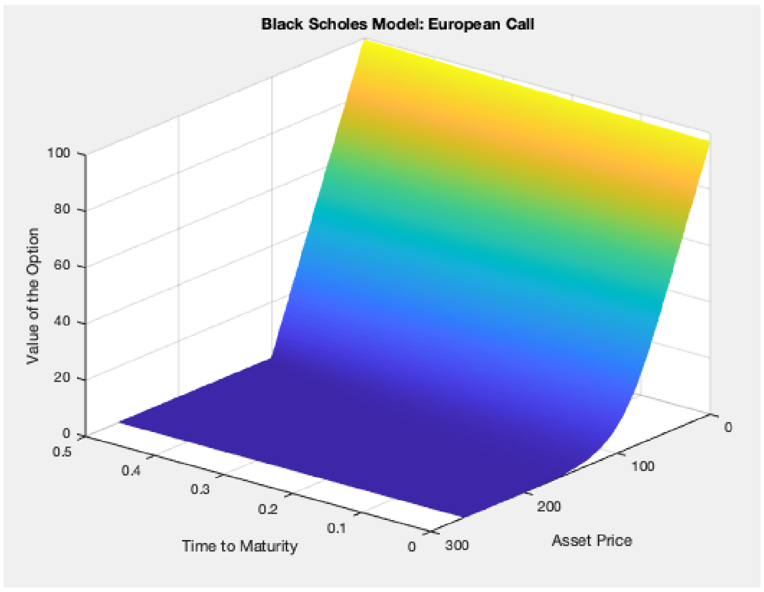

Figure 4: European Call

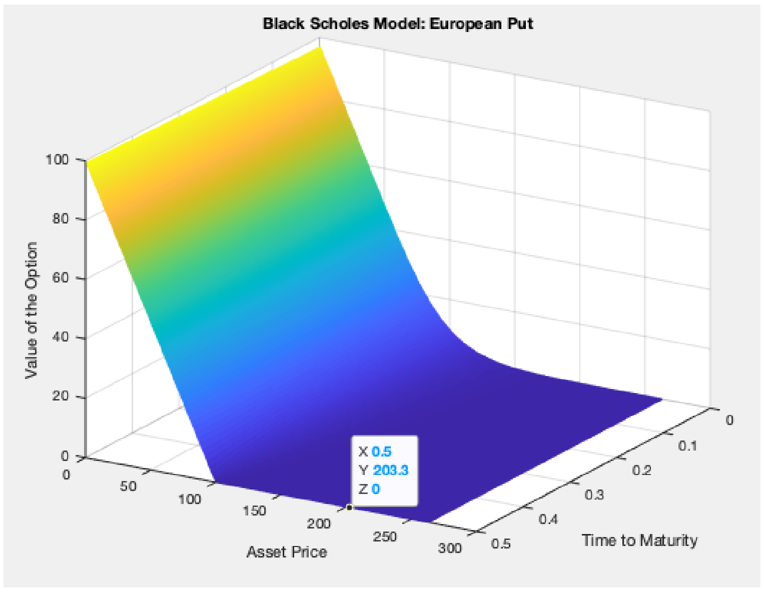

Figure 5: European Put

in price of the underlying stock. (Figure 7)

$$\delta = \frac{\partial V}{\partial S} \tag{3}$$

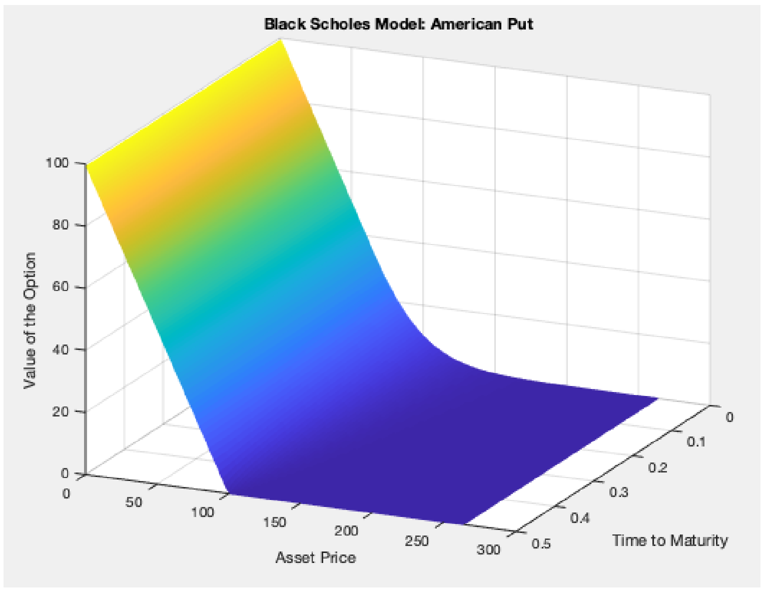

Figure 6: European Put

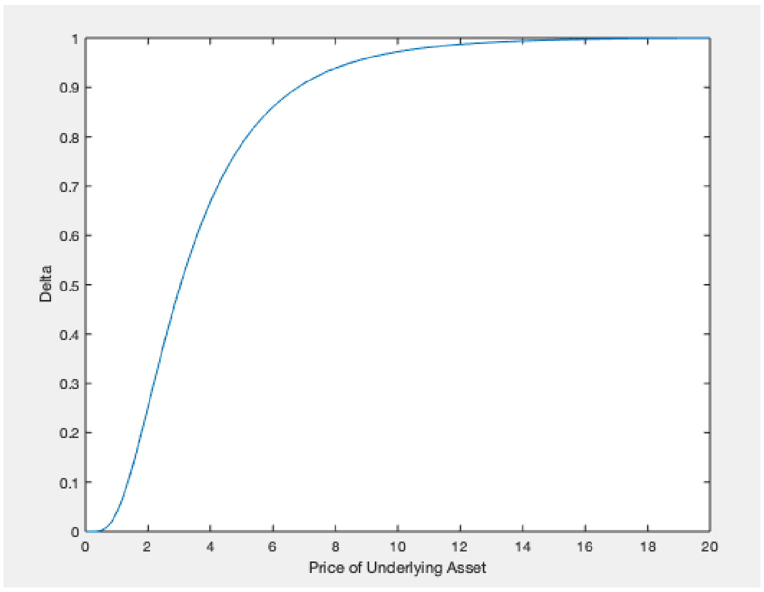

Figure 7: Delta

2. *Gamma*: This computes the change in option price with respect to change in Delta. (Figure 8)

$$\gamma = \frac{\partial^2 V}{\partial S^2} \tag{4}$$

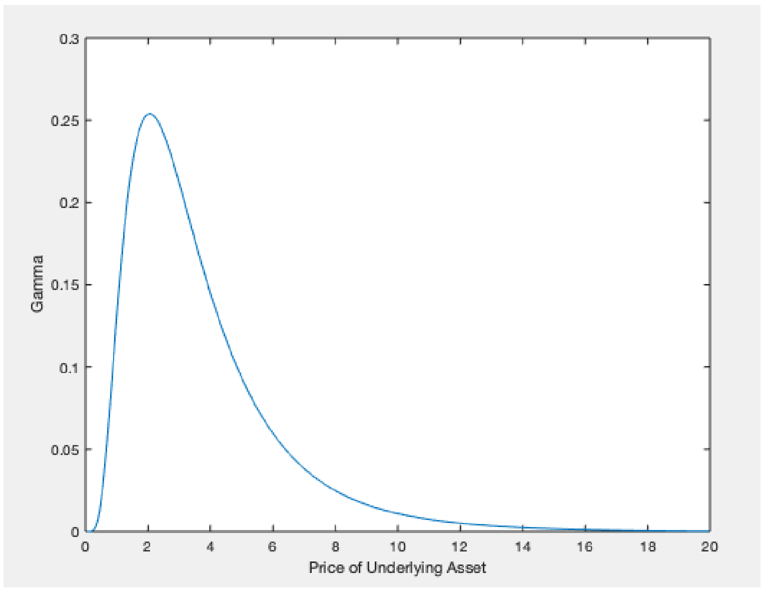

Figure 8: Gamma

3. *Theta*: This computes the change in option price with respect to change in time to maturity. (Figure 9)

$$\theta = \frac{\partial V}{\partial t} \tag{5}$$

4. *Vega*: This computes the change in option price with respect to change in volatility of the underlying stock. (Figure 10)

$$\nu = \frac{\partial V}{\partial \sigma} \tag{6}$$

5. *Rho*: This computes the change in option price with respect to change in risk-free interest rate. (Figure 11)

$$\rho = \frac{\partial V}{\partial r} \tag{7}$$

A.5 implied-volatility.m

This program uses the black-scholes-fucntion.m to get options prices ($v0$) and corresponding asset prices ($s0$) for a list of volatility values ranging

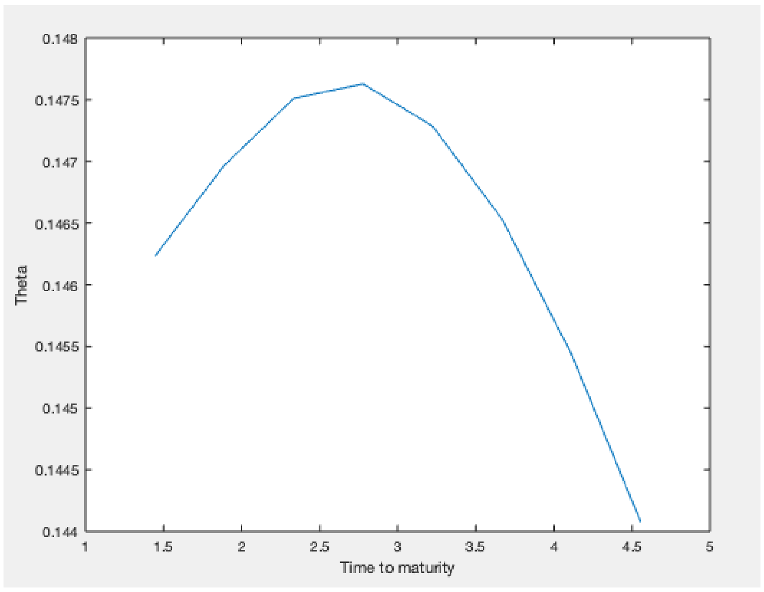

Figure 9: Theta

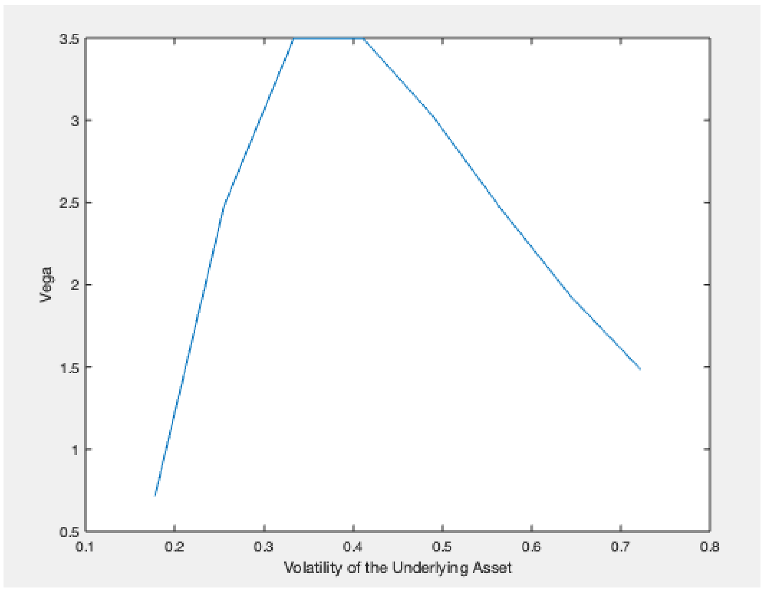

Figure 10: Vega

from 0 to 1 with an interval of 0.05

Next we use linear interpolation using the list of theoretical volatilities and corresponding option prices with the actual option price to get the implied volatility of the asset.

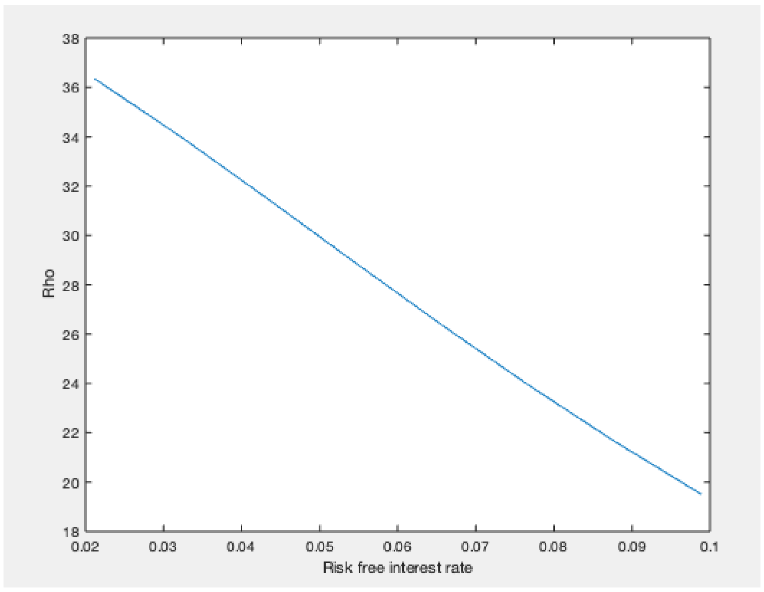

Figure 11: Rho

B Analysis of the Black-Scholes Model

B.1 error-analysis.m

This program uses the black-scholes-function.m to get options prices ($v0$) and corresponding asset prices ($s0$). The first time the function is the called, the grid size is $4X4$. Then we use a loop to get values $v1$ and $s1$ for a $(2 \times v0 - 2)X(2 \times v0 - 2)$ grid.

The two options price vectors v1 and v0 are called in the function error-pde.m which outputs the absolute value of the sum of the error between the two option price vectors multiplied by dx of $s0$.

Once enough values of the error for different grid sizes are computed, the program plots a graph of dx values and error values for each grid size. (Figure 12)

B.2 delta-t.m

This program uses a list of various N values, which are the number of nodes of the time dimension, and then uses the black-scholes-function.m to compute option price for a specific $S0$ on each N value.

Since all other variables except N are constant, the plot of V vs N shows the affect of different values of dt on V. (Figure 13)

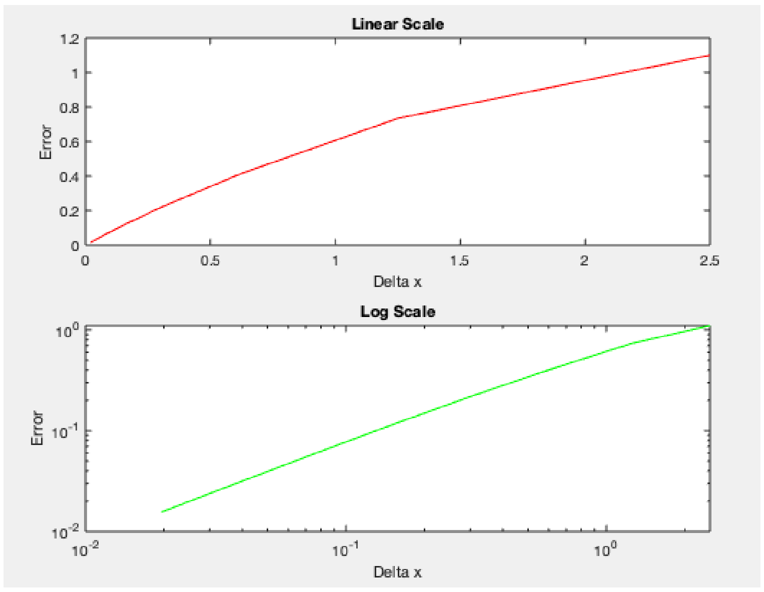

Figure 12: error-analysis

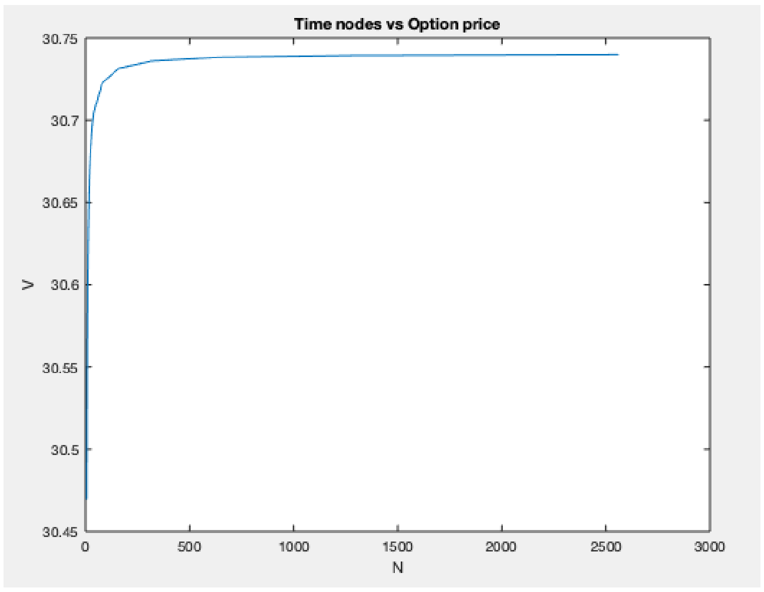

Figure 13: delta-t and option price

$$dt = \frac{T}{N} \tag{8}$$

B.3 S-max.m

This program uses a list of various $S-max$ values, which are the upper limit of the asset price dimension, and then uses the black-scholes-function.m to compute option price for a specific $S0$ on each $S-max$ value.

Since all other variables except $S-max$ are constant, the plot of V vs $S-max$ shows the affect of different values of $S-max$ on V. (Figure. 14)

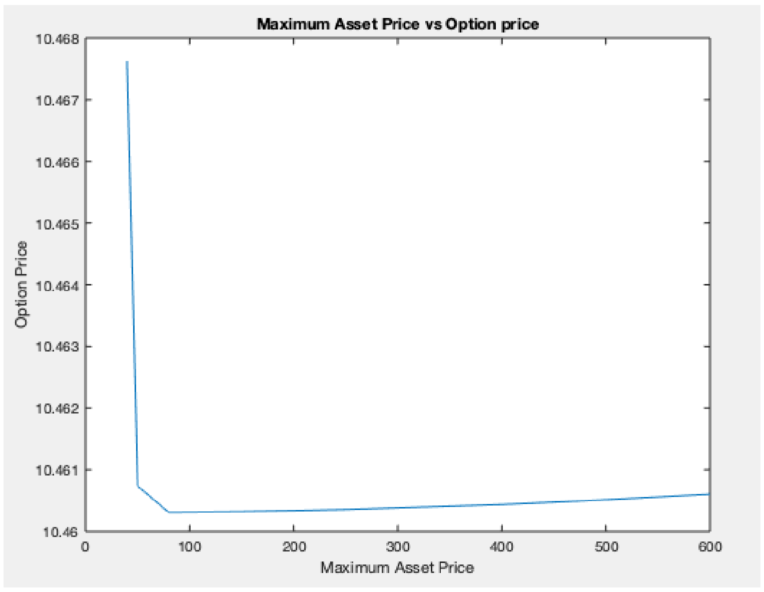

Figure 14: Maximum Asset Price and Option Price

B.4 S0.m

This program uses a list of various $S0$ values, which are the initial price of the underlying asset, and then uses the black-scholes-function.m to compute option price for each $S0$ value.

Since all other variables except $S0$ are constant, the plot of V vs $S0$ shows the affect of different values of $S0$ on V. (Figure 15)

B.5 sig-r.m

This program uses two lists of various r values and sig values, which are the risk free interest rates and volatility of the underlying asset respectively, and

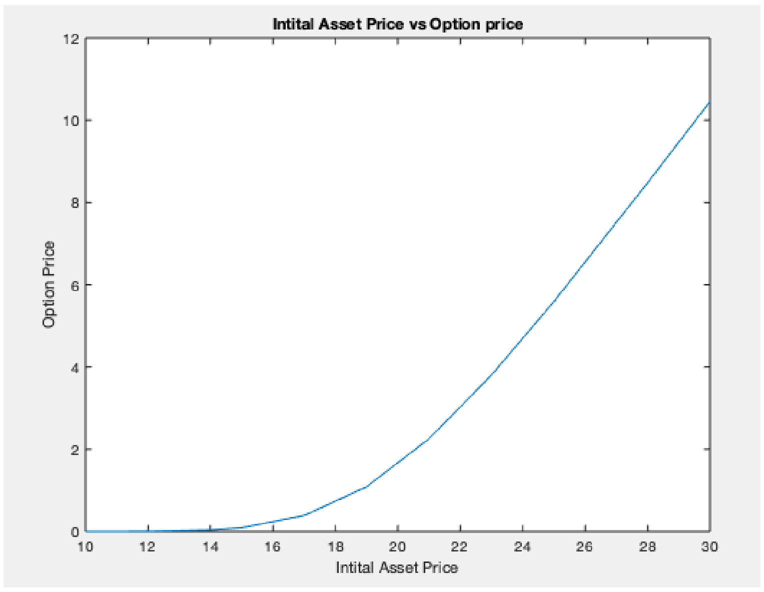

Figure 15: Initial Asset Price and Option Price

then uses the black-scholes-function.m to compute option price for a each possible pair of r and sig values.

Since all other variables except r and sig are constant, the surface plot of V vs r and sig shows the affect of different values of r and sig on V. (Figure 16)

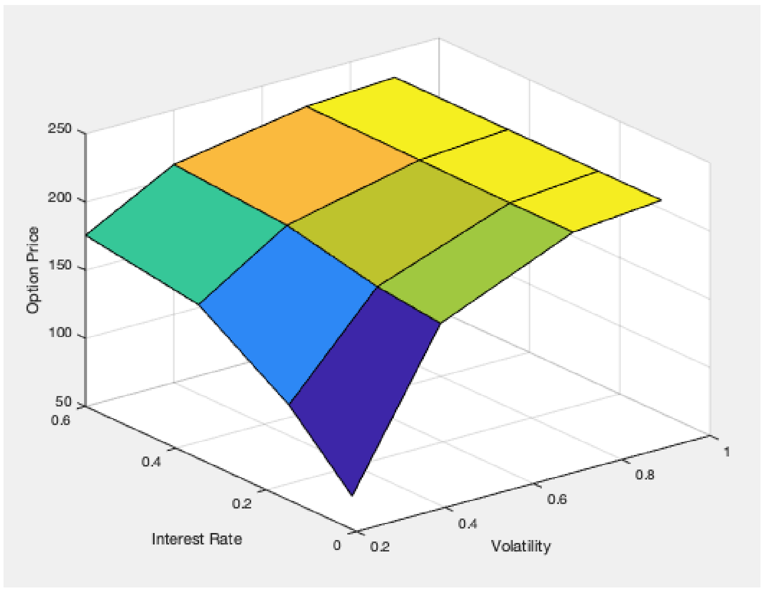

Figure 16: Volatility, Interest Rates and Option Price