FINA4140A Computational Finance
Course Project
Group I

Lo Chun Tung	1155009121
Ng Wing Yung	1155009148
Yiu Kit Ho	1155015374
Yu Cheuk Hin Anthony	1155009047

1. Geometric Brownian Motion Model

For the analysis of various derivatives, data on underlying assets are required. Since real market data only represents one trajectory of the price movements, future stock price trajectories have to be generated by Monte Carlo simulation.

This requires assumptions on asset price model. Unless otherwise stated, we assume the underlying asset price follows the Geometric Brownian Motion (GBM) model:

$$\frac{dS_t}{S_t} = \mu dt + \sigma dW_t, \text{ or } dS_t = S_t(\mu dt + \sigma dW_t)$$

where S_t is the stock price at time t, μ is the expected rate of return of the stock, σ is the volatility of the stock, and W_t is a standard Brownian motion (or Wiener process).

Discretizing (for use in the simulation part), we have

$$\Delta S_t = S_t(\mu \Delta t + \sigma\sqrt{\Delta t}\varepsilon_t)$$

where $\Delta S_t = S_{t+\Delta t} - S_t$, Δt is the time interval (which should be small), $\varepsilon_t \sim \mathcal{N}(0,1)$

1.1 Use Market Data to Estimate Model Parameters μ and σ

Before generating stock price trajectories using the discretized GBM model, we have to obtain an estimate of the parameters, μ and σ. The underlying assets and market data are specified in Table 1:

Underlying Asset	Hang Seng Index (HSI)
Observation Time Period	1 April 2013 – 31 March 2014
Number of Trading Days	246 Since 1 April 2013 is a public holiday, the actual data starts from 2 April 2013.
Data Source	Yahoo! Finance URL: http://finance.yahoo.com/q/hp?s=%5EHSI&a=03&b=1&c=2013&d=02&e=31&f=2014&g=d

Table 1: Specifications of Underlying Asset and Market Data

(Raw market data is attached in the file HSI_Price.csv)

With the raw market data, we obtain Figure 1:

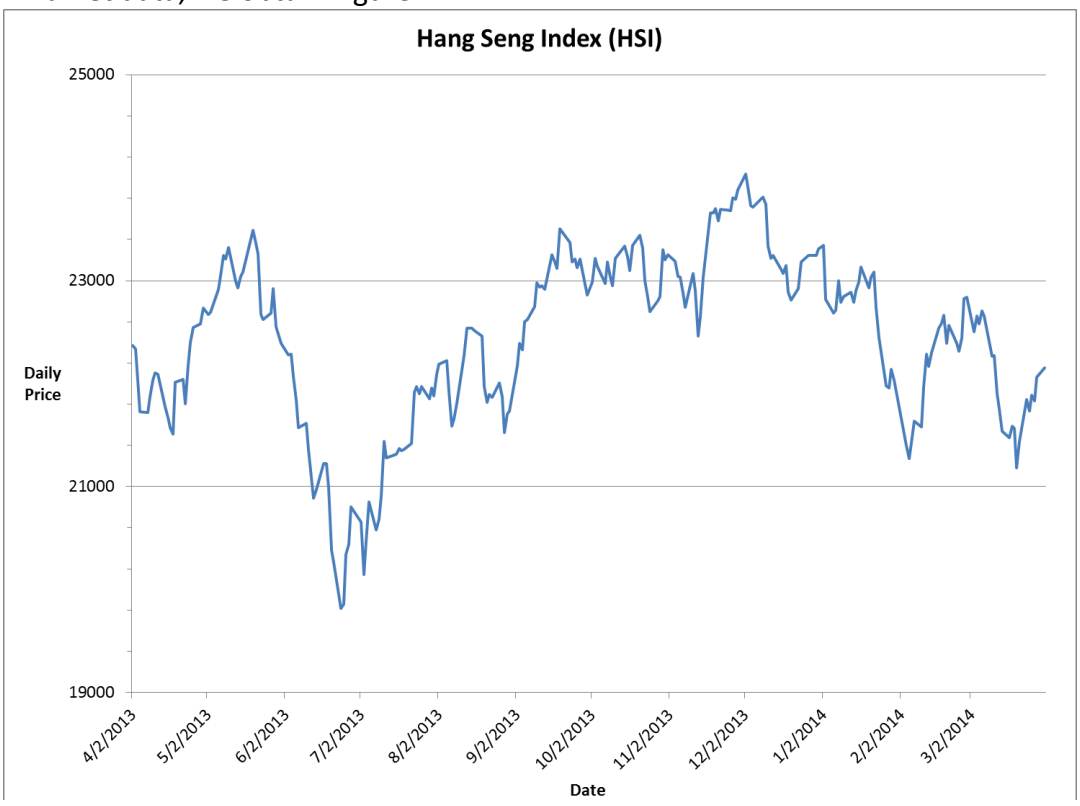


Figure 1: Asset Price in Observation Time Period

Assuming $\ln\frac{S_T}{S_t} \sim N\left(\left(\mu - \frac{1}{2}\sigma^2\right)(T-t), \sigma^2(T-t)\right)$ for trading day t (t = 1, 2, …, 246), we have

$$\text{Log return} = \ln\left(\frac{S_t}{S_{t-1}}\right) \sim N\left(\left(\mu - \frac{1}{2}\sigma^2\right), \sigma^2\right)$$

where μ and σ are the daily expected rate of return and volatility rate

Then, we find the sample mean $\bar{X}$ and sample variance S^2 of the log returns.

Daily expected rate of return = $\bar{X} + \frac{S^2}{2}$

Daily volatility rate = S

Annualized expected rate of return μ = Daily expected rate of return × 246

Annualized volatility rate σ = Daily volatility rate × $\sqrt{246}$

The results are summarized in Table 2:

Sample Mean $\bar{X}$	Sample S.D. S^2	Daily μ	Daily σ	Annualized μ	Annualized σ
-0.0040%	1.0143%	0.0012%	1.0143%	0.2877%	15.9087%

Table 2: Calculation Results of Annualized Expected Rate of Return μ and the Volatility Rate σ

The calculations in this part are listed in the sheet "Historical Data (1.1)" in Calculations.xlsm.

1.2 Generate Future Stock Prices Using Monte Carlo Simulation

With the estimated values of μ and σ, we can now generate future stock price using Monte Carlo simulation. The details are as follows:

- The index value on 31 March 2014, which is used as initial price S_0 in simulation, is 22151.06.
- Δt = 1 day. Hence, 250 random prices are generated for each trajectory.
- For simulation purpose, daily expected rate of return is $\frac{\mu}{250}$ and daily volatility rate is $\frac{\sigma}{\sqrt{250}}$.
- Since $\Delta S_t = S_t(\mu\Delta t + \sigma\sqrt{\Delta t}Z)$ for each trading day t, where t = 1, 2, …, 250, the price is generated by
  $$S_t = S_{t-1} \times \left(1 + \frac{\mu}{250} + \frac{\sigma}{\sqrt{250}} \times Z\right)$$
  where Z is a Gaussian random number with mean 0 and variance 1.
- The Gaussian random numbers are generated by Excel built-in functions with the following formula:
  `NORM.S.INV(RAND())`
  I.e. A Uniform(0,1) random variable is first generated, then converted back to Gaussian by inverse function of standard Gaussian distribution.
- 1000 independent trajectories were generated. The first 10 trajectories are displayed in Figure 2.

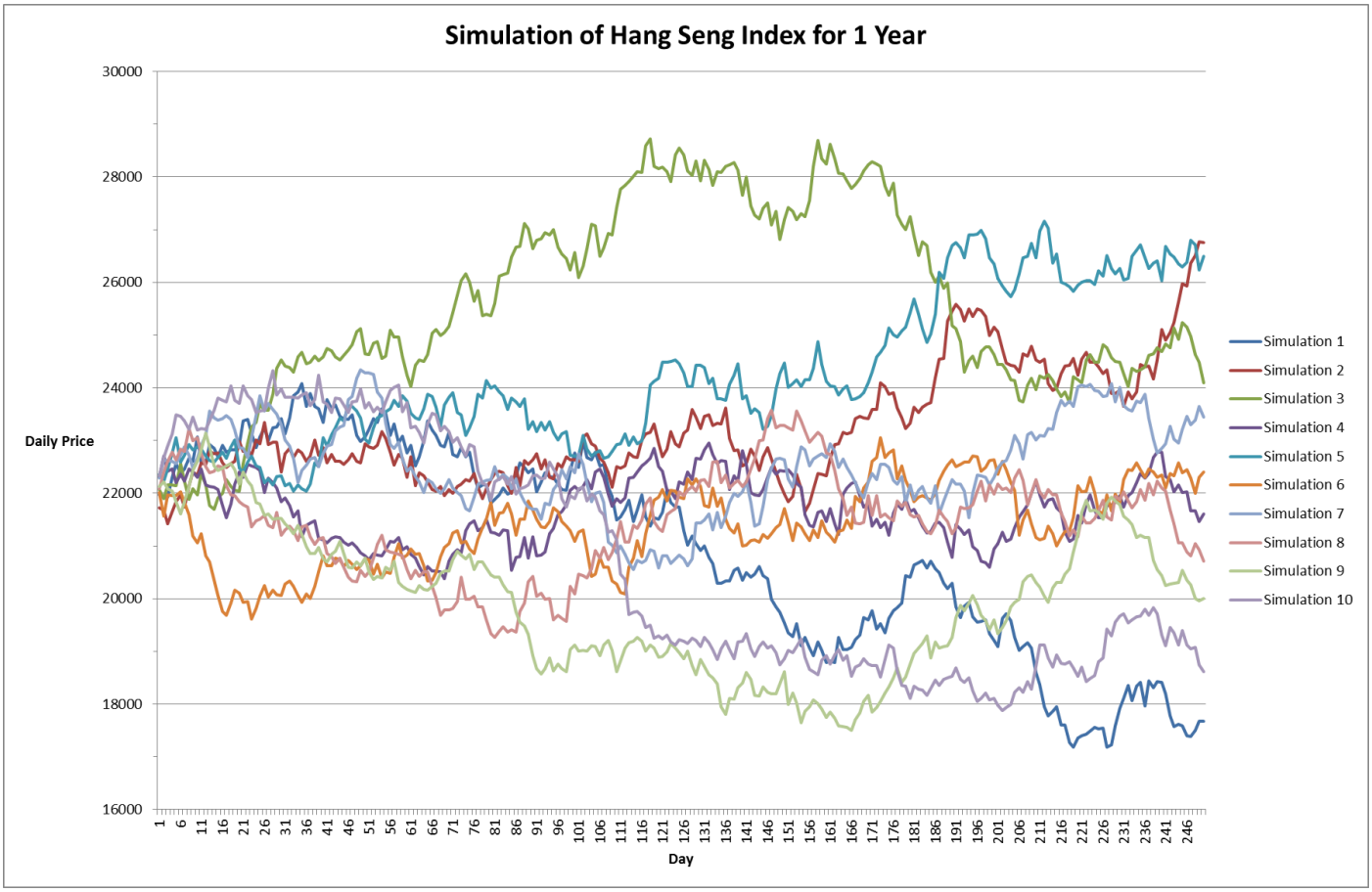


Figure 2: The First 10 Trajectories Generated

The formulae used to calculate the required statistical quantities are as follows, using the first trajectory (in column H) as an example:
Maximum price = **`MAX(H2:H251)`**
Minimum price = **`MIN(H2:H251)`**
Average price for the one year period = **`AVERAGE(H2:H251)`**
Daily log returns are calculated in H258 to H507.
Average rate of return = **`AVERAGE(H258:H507)`**
Standard deviation of the rate of return = **`STDEV.S(H258:H507)`**
Average daily log returns for each trajectory are calculated in H256 to ALS256 and standard deviations of daily log returns for each trajectory are calculated in H257 to ALS257.

For all 1000 trajectories,
Sample average of average rate of return = **`AVERAGE(H256:ALS256)`**
Sample average of standard deviation of the rate of return = **`AVERAGE(H257:ALS257)`**

The results are summarized in Table 3:

Sample mean of average rate of return	Sample mean of standard deviation of the rate of return	μ calculated from trajectories	σ calculated from trajectories
-0.0043%	1.0032%	0.1922%	15.8627%

Table 3: Results from Simulations

The simulated trajectories do not look similar to the historical realization in part 1.1.
The historical prices ranged from around 20000 to 24000 while the simulated trajectories range from 18000 to 28000.
The annualized μ calculated from trajectories is 0.1922%, compared with 0.2877% in part 1.1.
μ from simulations is significantly different from the estimated value in part 1.1.
The annualized σ from trajectories is 15.8627%, compared with 15.9087% in part 1.1.
σ from simulations is close to the estimated value in part 1.1.
A possible explanation to these observations is that the underlying assumption for stock returns is incorrect, i.e. log-returns do not follow normal distribution. Indeed, if we let

$$H_0: \ln\left(\frac{S_t}{S_{t-1}}\right) \sim N\left(\left(\mu - \frac{1}{2}\sigma^2\right), \sigma^2\right), H_1: \ln\left(\frac{S_t}{S_{t-1}}\right) \text{ does not follow normal distribution}$$

and carry out chi-square goodness-of-fit test, we have $\chi^2 = 13.73 > \chi^2_{0.10}(7) = 12.02$.
Hence we reject H_0 and conclude that log-return does not follow normal distribution.

Calculation formulae in this part are listed in the sheet "Simulation with Formula (1.2)" in Calculations.xlsm; to keep the generated trajectories for use in later parts, this sheet is copied and pasted by value into the sheet "Simulation Result Values (1.2)" in the same file

2. Black-Scholes-Merton Option Pricing Model

With the model specified in part 1, we now explore various risk-neutral pricing methods for standard options, including the Black-Scholes-Merton (BSM) formula, Monte-Carlo simulations and binomial trees.
To begin with, 8 European options are selected for investigation, with the specifications listed in Table 4:

Option Type	Spot Price	Strike Price	Maturity in Years	Dividend Yield	Annualized Volatility	Risk-free Interest Rate
Call	22151.06	17720.85	0.25	0%	15.9087%	5%
Call	22151.06	24366.17	0.50	0%	15.9087%	5%
Call	22151.06	22151.06	0.75	0%	15.9087%	5%
Call	22151.06	19935.95	1.00	1%	15.9087%	5%
Put	22151.06	26581.27	2.00	0%	15.9087%	5%
Put	22151.06	17720.85	0.50	0%	15.9087%	5%
Put	22151.06	22151.06	1.00	0%	15.9087%	5%
Put	22151.06	24366.17	0.75	1%	15.9087%	5%

Table 4: Specifications of the 8 Selected European Options

All the above 8 options are evaluated using the BSM formula; only the first call and the first put are evaluated using Monte Carlo simulation; the first call and the first put and their American counterparts are evaluated using binomial trees.

The option parameters are chosen in the following way:

- Spot price is chosen to be $S_0 = 22151.06$.
- Strike prices are chosen from K = S = 22151.06, K = 1.1S = 24366.17, K = 1.2S = 26581.27, K = 0.9S = 19935.95, K = 0.8S = 17720.85.
- Maturities are chosen from 3 months (0.25 years), 6 months (0.50 years), 9 months (0.75 years), 1 year and 2 years.
- Dividend yield is chosen from q = 0% (no dividend) and q = 1%.
- The annualized volatility from part 1.1 is chosen to be the volatility for these options.

2.1 Calculate Option Price Using the BSM Formula

As a standard for comparison, the prices of the 8 options are first evaluated by the BSM formula, which is

$$c(S,t) = S\mathcal{N}(d_1) - Ke^{-r(T-t)}\mathcal{N}(d_2)$$
$$p(S,t) = Ke^{-r(T-t)}\mathcal{N}(-d_2) - S\mathcal{N}(-d_1)$$
$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}}$$
$$d_2 = d_1 - \sigma\sqrt{T-t}$$

The results are shown in Table 5:

Option Type	Spot Price	Strike Price	Maturity in Years	Dividend Yield	Annualized Volatility	Risk-free Interest Rate	Calculated Option Value
Call	22151.06	17720.85	0.25	0%	15.9087%	5%	4651.03
Call	22151.06	24366.17	0.50	0%	15.9087%	5%	417.53
Call	22151.06	22151.06	0.75	0%	15.9087%	5%	1645.68
Call	22151.06	19935.95	1.00	1%	15.9087%	5%	3284.09
Put	22151.06	26581.27	2.00	0%	15.9087%	5%	3155.75
Put	22151.06	17720.85	0.50	0%	15.9087%	5%	10.56
Put	22151.06	22151.06	1.00	0%	15.9087%	5%	896.92
Put	22151.06	24366.17	0.75	1%	15.9087%	5%	2127.49

Table 5: BSM Formula Value of the 8 Selected European Options

The calculations in this part are listed in the sheet "BSM Pricing (2.1)" in Calculations.xlsm.

2.2 Calculate Option Price Using Monte Carlo Simulation

Using VBA, depending on the sample size n, S_T (index price at maturity) is generated for n times by

$$S_T = S_0 \exp\left(\left(r - \frac{1}{2}\sigma^2\right)T + \sigma\sqrt{T}Z\right), \text{where } Z \sim N(0,1)$$

Then, the payoff $\max(S_T - K, 0)$ is found for each S_T generated. The average of the n payoffs is calculated and discounted to value at time 0 with the discount factor e^{-rT}.

$$\text{European call price} = e^{-rT} \times \frac{1}{n}\sum_{i=1}^{n} \max((S_T)_i - K, 0)$$

The European call prices estimated with different number of samples are listed in Table 6:

Initial price: S	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06
Interest rate: r	5%	5%	5%	5%	5%	5%	5%	5%
Volatility: σ	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%
Maturity: T	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
Strike: K	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85
Number of samples: n	100	1000	5000	10000	50000	100000	500000	1000000
Price	**4445.062**	**4659.532**	**4630.617**	**4662.585**	**4645.116**	**4657.483**	**4650.469**	**4649.387**
Difference from BSM price	-205.964	8.506009	-20.4092	11.55841	-5.91013	6.456819	-0.55742	-1.63916

Table 6: Monte Carlo Approximations of European Call Price with Varying n

Similarly,

$$\text{European put price} = e^{-rT} \times \frac{1}{n}\sum_{i=1}^{n} \max(K - (S_T)_i, 0)$$

The European put prices estimated with different number of samples are listed in Table 7:

Initial price: S	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06
Interest rate: r	5%	5%	5%	5%	5%	5%	5%	5%
Volatility: σ	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%
Maturity: T	2	2	2	2	2	2	2	2
Strike: K	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27
Number of samples: n	100	1000	5000	10000	50000	100000	500000	1000000
Price	**3484.147**	**3163.169**	**3165.229**	**3119.199**	**3124.536**	**3165.884**	**3149.727**	**3154.058**
Difference from BSM price	328.3992	7.422138	9.481728	-36.5482	-31.2117	10.13685	-6.02007	-1.68917

Table 7: Monte Carlo Approximations of European Put Price with Varying n

As n increases, the Monte Carlo approximations of both options become closer and closer to the BSM value. A plot of the absolute error is shown in Figure 3.

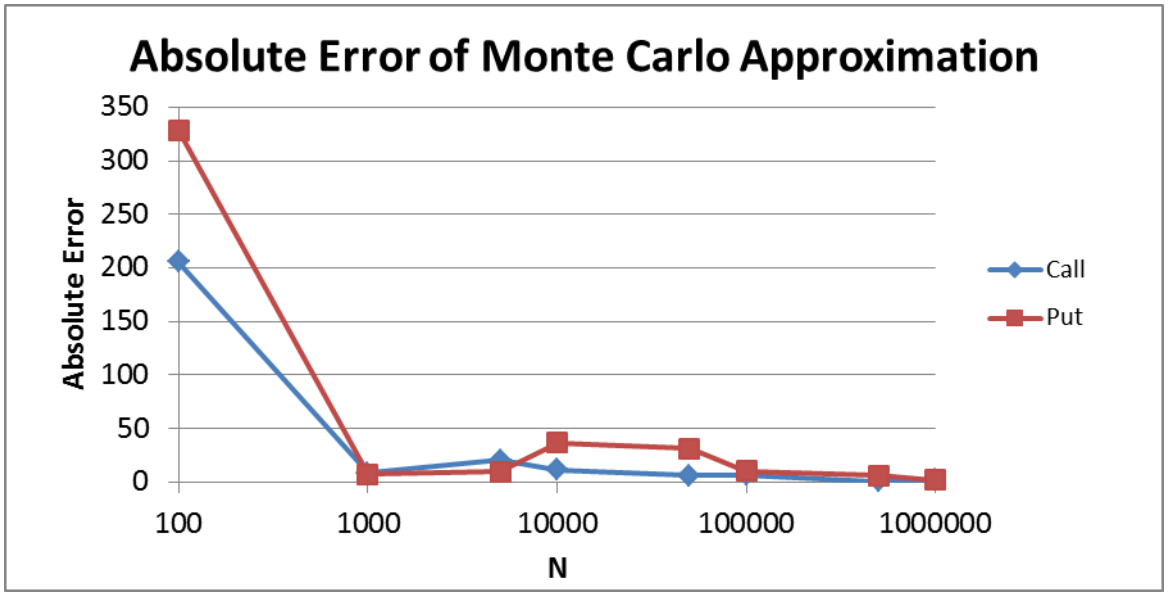


Figure 3: Plot of Absolute Error of Monte Carlo Approximation as Number of Samples Increases

The calculations in this part are listed in the sheet "Monte Carlo Pricing (2.2)" in Calculations.xlsm.

2.3 Calculate Option Price Using Binomial Tree Algorithm

In this part, the options in part 2.2 are priced again using binomial tree algorithm (multi-period recombining tree).

Assuming in each period, the asset price can only go up by a factor u or down by a factor d, where $ud = 1$, a binomial tree can be generated with different number of periods. For each period, $\Delta t = \frac{T}{M}$.

Using Cox-Ross-Rubinstein parameterization, we can choose $u = e^{\sigma\sqrt{\Delta t}}, d = e^{-\sigma\sqrt{\Delta t}}, p = \frac{e^{r\sqrt{\Delta t}} - d}{u - d}$.

For European options,

$$V_n^i = \max\left(e^{-r\sqrt{\Delta t}}\left(pV_{n+1}^{i+1} + (1-p)V_n^{i+1}\right), 0\right)$$

where V_n^i denotes the value of the option at time i and the asset price has gone up by n times.

Using VBA, the European call price V_0^0 is calculated and shown in Table 8 with different number of lattice steps.

Initial price: S	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06
Maturity: T	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
Strike: K	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85	17720.85
Volatility: σ	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%
Interest rate: r	5%	5%	5%	5%	5%	5%	5%	5%	5%
Number of steps: m	4	8	16	32	100	500	800	1000	2000
Δt	0.0625	0.03125	0.015625	0.007813	0.0025	0.0005	0.000313	0.00025	0.000125
u	1.0406	1.0285	1.0201	1.0142	1.0080	1.0036	1.0028	1.0025	1.0018
d	0.9610	0.9723	0.9803	0.9860	0.9921	0.9964	0.9972	0.9975	0.9982
p	0.5294	0.5208	0.5147	0.5104	0.5059	0.5026	0.5021	0.5019	0.5013
European Call Price	**4650.34**	**4650.43**	**4650.85**	**4650.87**	**4650.97**	**4651.02**	**4651.02**	**4651.02**	**4651.02**
Difference from BSM price	-0.6825	-0.5930	-0.1724	-0.1591	-0.0611	-0.0060	-0.0038	-0.0030	-0.0017

Table 8: Binomial Approximations of European Call Price with Varying m

Keeping the parameters the same and changing the option type to American, the option is priced using

$$V_n^i = \max\left(S_t - K, e^{-r\sqrt{\Delta t}}\left(pV_{n+1}^{i+1} + (1-p)V_n^{i+1}\right), 0\right)$$

The results are displayed in Table 9:

American Call Price	**4650.34**	**4650.43**	**4650.85**	**4650.87**	**4650.97**	**4651.02**	**4651.02**	**4651.02**	**4651.02**

Table 9: Binomial Approximations of American Call Price with Varying m

The American call prices are exactly the same as the European call prices. It is because the selected option does not pay dividend and the American call would not be exercised early.

Similarly, the results for European put are displayed in Table 10:

Initial price: S	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06
Maturity: T	2	2	2	2	2	2	2	2	2
Strike: K	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27	26581.27
Volatility: σ	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%
Interest rate: r	5%	5%	5%	5%	5%	5%	5%	5%	5%
Number of steps: m	4	8	16	32	100	500	800	1000	2000
Δt	0.5	0.25	0.125	0.0625	0.02	0.004	0.0025	0.002	0.001
u	1.1191	1.0828	1.0579	1.0406	1.0228	1.0101	1.0080	1.0071	1.0050
d	0.8936	0.9235	0.9453	0.9610	0.9778	0.9900	0.9921	0.9929	0.9950
p	0.5842	0.5591	0.5416	0.5294	0.5166	0.5074	0.5059	0.5052	0.5037
European Put Price	**3107.95**	**3111.58**	**3163.51**	**3155.25**	**3149.59**	**3154.58**	**3155.94**	**3155.54**	**3155.55**
Difference from BSM price	-47.7970	-44.1722	7.7676	-0.5012	-6.1541	-1.1669	0.1959	-0.2118	-0.1963

Table 10: Binomial Approximations of European Put Price with Varying m

The American counterpart of this option is priced using

$$V_n^i = \max\left(K - S_t, e^{-r\sqrt{\Delta t}}\left(pV_{n+1}^{i+1} + (1-p)V_n^{i+1}\right), 0\right)$$

The results are displayed in Table 11:

American Put Price	**4430.21**	**4430.21**	**4430.21**	**4430.21**	**4430.21**	**4430.21**	**4430.21**	**4430.21**	**4430.21**

Table 11: Binomial Approximations of American Put Price with Varying m

The American put prices are the same for different number of steps. It is because the option is being exercised at a very early stage. The price is much higher than that of European put because the holder can enjoy a higher payoff from being able to exercise early.

As m increases, the binomial approximations of both European call and put become closer to the BSM value. Also, the differences using binomial method are smaller than those from Monte Carlo approximations. A plot of the absolute error as number of lattice steps increases is shown in Figure 4. It is clear that as number of lattice steps increases, absolute error decreases and approaches zero.

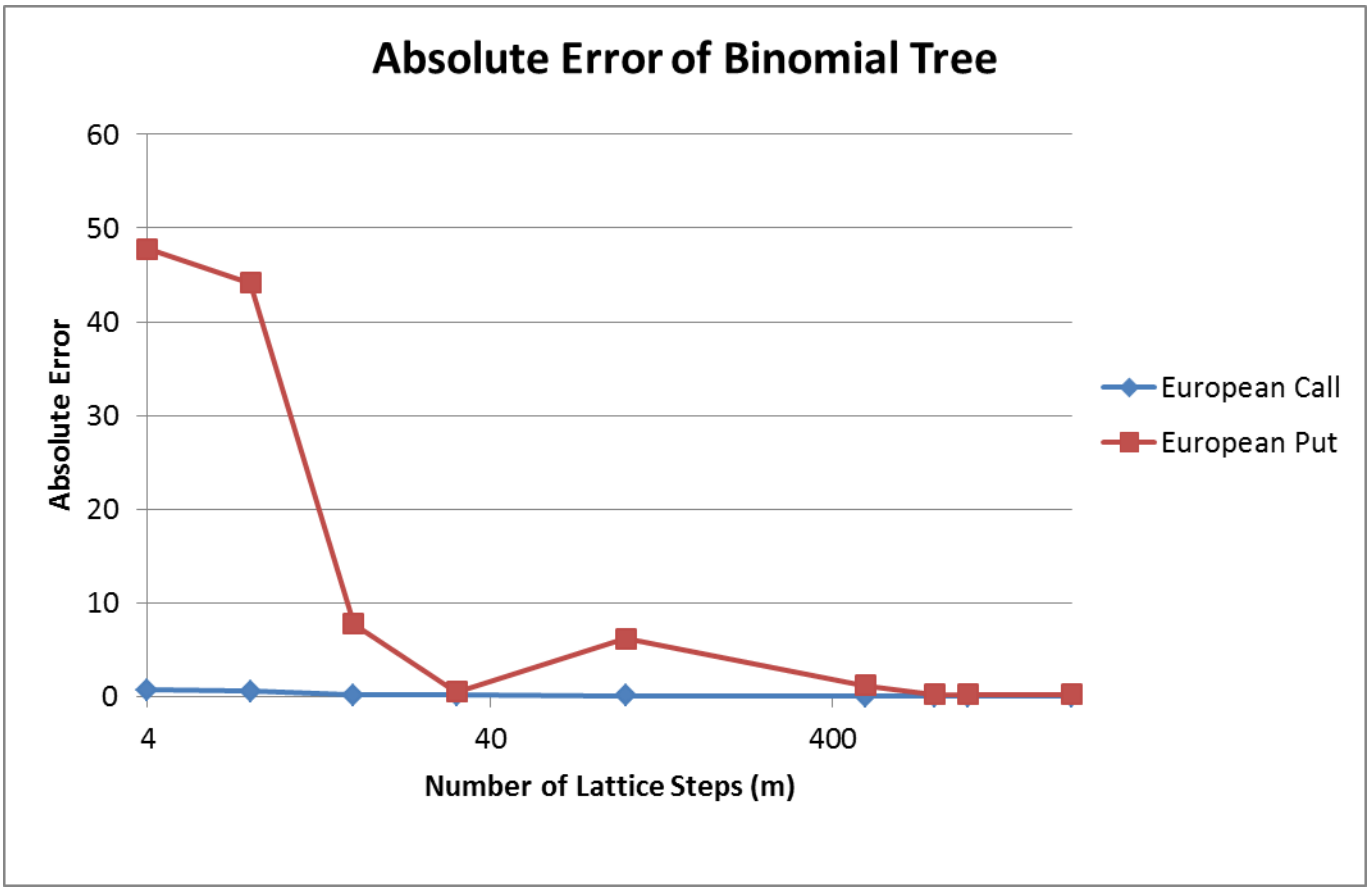


Figure 4: Plot of Absolute Error of Binomial Model as Number of Lattice Steps Increases

The calculations in this part are listed in the sheet "Binomial Pricing (2.3)" in Calculations.xlsm.

3. Option Pricing with Finite Difference Method

In this part, we consider pricing of European options using finite difference method (in particular, the Crank-Nicolson finite difference method), and various considerations in the model, including the choice of PDE, boundary conditions, initial/finite conditions, underlying asset price models and discretization mechanisms.

3.1 Boundary Conditions of European Options

For European call options, when $S \to 0$, $C \to 0$. As $\mathbb{P}(S_T > K) \to 0$, the probability of the call option generating a positive payoff at maturity is negligible.

When $S \to \infty$, i.e. $S \gg K$, we have $C \to S - Ke^{-rT}$. Since $\mathbb{P}(S_T > K) \to 1$, it is almost certain that the option will be in-the-money at maturity. Consider selling one unit of the underlying asset for S, buying a call option and depositing amount Ke^{-rT} at risk-free rate r. At maturity, the option is very likely to be exercised, so the underlying asset can be bought at K, which equals to the deposit plus interest. By no-arbitrage principle we have the value of call option $C \to S - Ke^{-rT}$.

For European put options, when $S \to 0$, $P \to Ke^{-rT}$. As $\mathbb{P}(S_T < K) \to 1$, the option will almost definitely be in-the-money at maturity and exercised. Consider buying one unit of the underlying asset at S, buying a put option and borrowing amount Ke^{-rT} at risk-free rate r. At maturity, when the option is exercised, the asset can be sold for K, which is used to repay the debt plus interest. By no-arbitrage principle we have the value of put option $P \to Ke^{-rT} - S$.
When $S \to \infty$, i.e $S \gg K$, the probability $\mathbb{P}(S_T < K) \to 0$, which implies that the probability of the put option generating a positive payoff at maturity is negligible. Therefore, $P \to 0$.

Based on the above discussion, we obtain Table 12:

	European Call	European Put
Final Condition (t = T)	max(S – K, 0)	max(K – S, 0)
Boundary Condition ($S = S^1$)	$V^1 = 0$	$V^1 = Ke^{-rT} - S$
Boundary Condition ($S = S^{N+1}$)	$V^{N+1} = S - Ke^{-rT}$	$V^{N+1} = 0$

Table 12: Final Conditions and Boundary Conditions for the Backward Parabolic PDE

3.2 Local Volatility Function (LVF) Models

In the Local Volatility Function (LVF) model, the underlying asset price is assumed to follow the stochastic differential equation:

$$\frac{dS}{S} = (\mu - q)dt + \sigma(S,t)dW$$

Where μ is the expected rate of return, q is continuous dividend yield, W denotes a standard Brownian motion and $\sigma(S,t)$ is the local volatility function. Assume interest rate r is a positive constant and denote the initial price of underlying asset as S_0, the value $V(S,t)$ of the European option satisfies the backward parabolic PDE:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma(S,t)^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r - q)S\frac{\partial V}{\partial S} - rV = 0$$

Here, t denotes time from the spot. For simulation purposes, it would be more convenient to define t as the time until maturity instead. The PDE can be obtained just by changing a few signs in the above equation:

$$\frac{\partial V}{\partial t} - \frac{1}{2}\sigma(S,t)^2 S^2 \frac{\partial^2 V}{\partial S^2} - (r - q)S\frac{\partial V}{\partial S} + rV = 0$$

3.2.1 Constant Elasticity of Variance (CEV) model

The Constant Elasticity of Variance (CEV) model is a commonly used LVF model which assumes:

$$\frac{\frac{\partial \mathrm{Var}(\tilde{R})}{\mathrm{Var}(\tilde{R})}}{\frac{\partial S}{S}} = 2(1 - \beta), \beta \geq 1$$

When β = 1, we have the classical Black-Scholes-Merton (BSM) model, and the asset price follows Geometric Brownian Motion (GBM).
When β > 1, the variance of asset return is a decreasing function of the asset price. For example, β = 1.5 gives rise to the Square Root Process, and β = 2 generates the Absolute Diffusion Process.

Based on the assumption, we have

$$\sigma(S,t) = \alpha S^{1-\beta}, \alpha > 0, \beta \geq 1$$

Underlying asset price thus follows the stochastic differential equation:

$$\frac{dS}{S} = (\mu - q)dt + \alpha S^{1-\beta}dW$$

Under CEV, value of European option value satisfies the backward PDE:

$$\frac{\partial V}{\partial t} - \frac{1}{2}\alpha^2 S^{4-2\beta}\frac{\partial^2 V}{\partial S^2} - (r-q)S\frac{\partial V}{\partial S} + rV = 0$$

Since the coefficients of $\frac{\partial^2 V}{\partial t^2}$ and $\frac{\partial^2 V}{\partial S\,\partial t}$ are both 0, this PDE is parabolic.

The PDE usually has no closed-form solution, but its value can be approximated with finite difference methods.

3.2.2 Crank-Nicolson Finite Difference Method with Non-Uniform Discretization

To compute the PDE numerically, the region $S \in [0, L], t \in [0, T]$ is discretized. L denotes a suitably large upper bound for the discretization.

Let $0 = S^0 < S^1 < \cdots < S^N = L$ denote the discretized values for S, and $0 = t_0 < t_1 < \cdots < t_M = T$ the discretized values for t.

A simple uniform discretization can be done by taking $S^i = i\Delta S, t_j = j\Delta t$, where $\Delta S = \frac{L}{N}, \Delta t = \frac{T}{M}$.

However, for computational efficiency, the number of discretization points should not be too large. Therefore, a more accurate and efficient method would be to focus the discretization points around the strike price, where option prices fluctuates most significantly, and fewer discretization points near the boundary where option prices tend not to change significantly. This gives rise to a non-uniform discretization in S.

A simple non-uniform discretization can be generated by the function:

$$S^i = \left(1 + \left(\frac{4i}{N} - 1\right)^{\frac{x}{y}}\right)E$$

Here, x and y are odd integers. The discretization has the nice property that it is the most concentrated around S = E. It allows flexibility on how concentrated the discretization points around E should be. If we take x = y = 1, we get the uniform discretization over the region [0,4E]. The greater the value x/y is, the more non-uniform the discretization gets. In the project, x = 7, y = 5 is chosen, so x/y = 1.4. The upper bound of the region covered by this discretization is $(1 + 3^{1.4})E = 5.66E$. The histogram in Figure 5 shows the distribution of the discretization points when N = 800 and strike price E = 100.

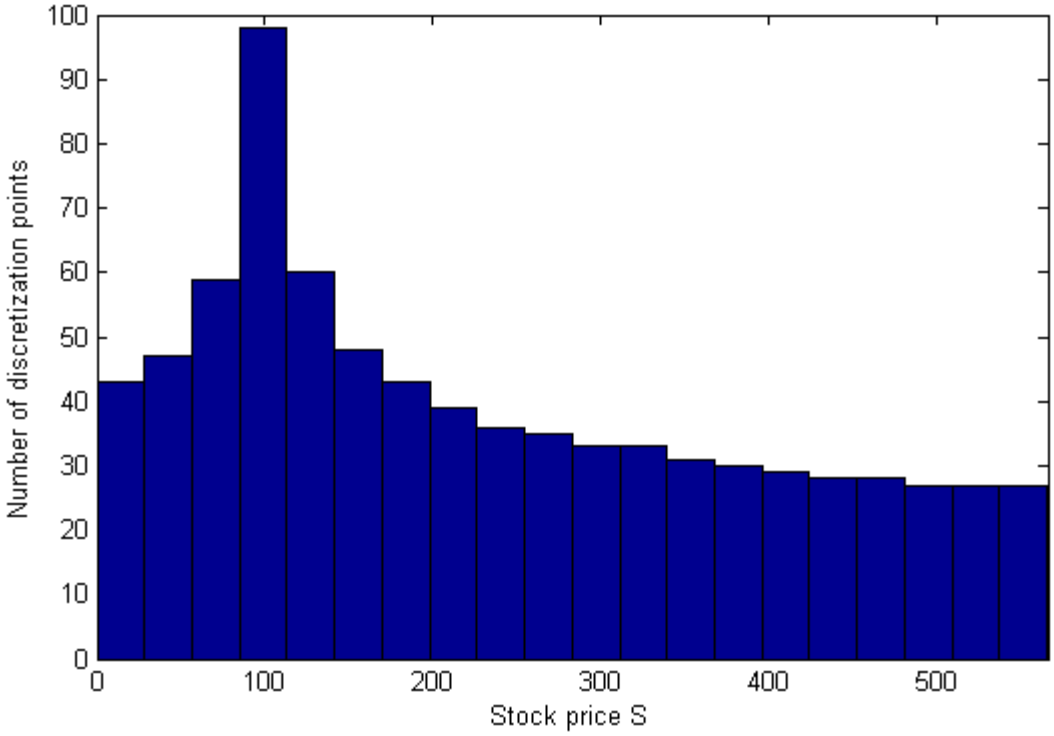


Figure 5: Distribution of Discretization Points, E=100, N=800

Let $h_i = S^{i+1} - S^i$, V_j^i = value of the option when $S = S^i, t = t_j$. Then, we have

$$V_j^{i+1} = V_j^i + \frac{\partial V_j^i}{\partial S} h_i + \frac{1}{2}\frac{\partial^2 V_j^i}{\partial S^2} h_i^2 + \frac{1}{6}\frac{\partial^3 V_j^i}{\partial S^3} h_i^3 + O(h_i^4)$$

$$V_j^{i-1} = V_j^i - \frac{\partial V_j^i}{\partial S} h_{i-1} + \frac{1}{2}\frac{\partial^2 V_j^i}{\partial S^2} h_{i-1}^2 - \frac{1}{6}\frac{\partial^3 V_j^i}{\partial S^3} h_{i-1}^3 + O(h_{i-1}^4)$$

Define $a_i = \frac{2}{h_{i-1}(h_{i-1}+h_i)}, b_i = \frac{2}{h_i(h_{i-1}+h_i)}$, then

$$\begin{aligned} & a_i V_j^{i-1} - (a_i + b_i) V_j^i + b_i V_j^{i+1} \\ &= (-a_i h_{i-1} + b_i h_i)\frac{\partial V_j^i}{\partial S} + \frac{1}{2}(a_i h_{i-1}^2 + b_i h_i^2)\frac{\partial^2 V_j^i}{\partial S^2} + \frac{1}{6}(-a_i h_{i-1}^3 + b_i h_i^3)\frac{\partial^3 V_j^i}{\partial S^3} + O(a_i h_{i-1}^4 + b_i h_i^4) \\ &= \frac{\partial^2 V_j^i}{\partial S^2} + \frac{1}{3}\frac{\partial^3 V_j^i}{\partial S^3}(h_i - h_{i-1}) + O(h_{i-1}^2 + h_i^2) \end{aligned}$$

Hence $a_i V_j^{i-1} - (a_i + b_i) V_j^i + b_i V_j^{i+1}$ converges to $\frac{\partial^2 V_j^i}{\partial S^2}$ at rate O(h). However, when S is close to E where the discretization points are dense and close to uniform, the term $\frac{1}{3}\frac{\partial^3 V_j^i}{\partial S^3}(h_i - h_{i-1})$ vanishes and the expression converges at rate $O(h^2)$, which is the same as in uniform discretization.

Also,

$$V_j^{i+1} - V_j^{i-1} = \frac{\partial V_j^i}{\partial S}(h_{i-1} + h_i) + \frac{1}{2}\frac{\partial^2 V_j^i}{\partial S^2}(h_i^2 - h_{i-1}^2) + O(h_i^3 + h_{i-1}^3)$$

$$\frac{V_j^{i+1} - V_j^{i-1}}{h_{i-1} + h_i} = \frac{\partial V_j^i}{\partial S} + \frac{1}{2}\frac{\partial^2 V_j^i}{\partial S^2}(h_i - h_{i-1}) + O(h_i^2 + h_{i-1}^2)$$

Therefore, $\frac{V_j^{i+1} - V_j^{i-1}}{h_{i-1}+h_i}$ converges to $\frac{\partial V_j^i}{\partial S}$ at rate O(h), and the convergence rate becomes $O(h^2)$ when S is close to E, where the discretization is approximately uniform as the term $\frac{1}{2}\frac{\partial^2 V_j^i}{\partial S^2}(h_i - h_{i-1})$ vanishes.

As for the time dimension, the discretization is kept uniform.

Denote $k = \Delta t = \frac{T}{M}$, then

$$V_{j+1}^i = V_j^i + \frac{\partial V_j^i}{\partial t} k + O(k^2)$$

$$\frac{V_{j+1}^i - V_j^i}{k} = \frac{\partial V_j^i}{\partial t} + O(k)$$

The forward difference in time, central difference in space (FTCS) scheme can be written as:

$$\frac{V_{j+1}^i - V_j^i}{k} - \frac{1}{2}\alpha^2 (S^i)^{4-2\beta}\left(a_i V_j^{i-1} - (a_i + b_i) V_j^i + b_i V_j^{i+1}\right) - (r - q) S^i \left(\frac{V_j^{i+1} - V_j^{i-1}}{h_{i-1} + h_i}\right) + r V_j^i = 0$$

Similarly, the backward difference in time, central difference in space scheme is:

$$\frac{V_{j+1}^i - V_j^i}{k} - \frac{1}{2}\alpha^2 (S^i)^{4-2\beta}\left(a_i V_{j+1}^{i-1} - (a_i + b_i) V_{j+1}^i + b_i V_{j+1}^{i+1}\right) - (r - q) S^i \left(\frac{V_{j+1}^{i+1} - V_{j+1}^{i-1}}{h_{i-1} + h_i}\right) + r V_{j+1}^i = 0$$

Using matrix notations, denote:

$$U^j = \begin{pmatrix} V_j^1 \\ \vdots \\ V_j^{N-1} \end{pmatrix}, D_1 = \begin{pmatrix} S^1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & S^{N-1} \end{pmatrix}, D_2 = D_1^{4-2\beta}, H = \begin{pmatrix} \frac{1}{h_0 + h_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{h_{N-2} + h_{N-1}} \end{pmatrix}$$

$$T_1 = \begin{pmatrix} 0 & 1 & 0 & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & & 0 & 1 \\ & & & -1 & 0 \end{pmatrix}$$

$$T_2 = \begin{pmatrix} -(a_1+b_1) & b_1 & & & \\ a_2 & -(a_2+b_2) & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & -(a_{N-2}+b_{N-2}) & b_{N-2} \\ & & & a_{N-1} & -(a_{N-1}+b_{N-1}) \end{pmatrix}$$

$$F = (1-rk)I + \frac{1}{2}k\alpha^2 D_2 T_2 + k(r-q)D_1 H T_1$$

Then the FTCS scheme can be written as $U^{j+1} = FU^j + p^j$, where

$$p^j = \begin{pmatrix} k\left(\frac{\alpha^2 (S^1)^{4-2\beta}}{2} a_1 - \frac{(r-q)S^1}{h_0+h_1}\right) V_j^0 \\ 0 \\ \vdots \\ 0 \\ k\left(\frac{\alpha^2 (S^{N-1})^{4-2\beta}}{2} b_{N-1} + \frac{(r-q)S^{N-1}}{h_{N-2}+h_{N-1}}\right) V_j^N \end{pmatrix}$$

Similarly, using the same notations, denote

$$B = (1+rk)I - \frac{1}{2}k\alpha^2 D_2 T_2 - k(r-q)D_1 H T_1$$

Then the BTCS scheme can be written as $BU^{j+1} = U^j + q^j$, where

$$q^j = \begin{pmatrix} k\left(\frac{\alpha^2 (S^1)^{4-2\beta}}{2} a_1 - \frac{(r-q)S^1}{h_0+h_1}\right) V_{j+1}^0 \\ 0 \\ \vdots \\ 0 \\ k\left(\frac{\alpha^2 (S^{N-1})^{4-2\beta}}{2} b_{N-1} + \frac{(r-q)S^{N-1}}{h_{N-2}+h_{N-1}}\right) V_{j+1}^N \end{pmatrix}$$

Crank-Nicolson method is the average of FTCS and BTCS schemes, which gives

$$\frac{1}{2}(I+B)U^{j+1} = \frac{1}{2}(I+F)U^j + \frac{1}{2}(p^j + q^j)$$

For European call options, set $V_j^0 = 0, V_j^N = S^N - Ee^{-rt_j}, U^0 = \begin{pmatrix} \max(S^1 - E, 0) \\ \vdots \\ \max(S^{N-1} - E, 0) \end{pmatrix}$

For European put options, set $V_j^0 = Ee^{-rt_j} - S^N, V_j^N = 0, U^0 = \begin{pmatrix} \max(E - S^1, 0) \\ \vdots \\ \max(E - S^{N-1}, 0) \end{pmatrix}$

Then the initial option prices are given by U^M.

An implementation of the algorithm as a function that accepts different values of S_0, r, q, T, E, α, β, M, N, and type of option (call or put) is available at MyCallPutExp.m. A simplified version which sets α = 20, β = 2, M = N = 1600 is available at MyCallPut.m. Both of them output the initial option value and a matrix of option values for different underlying prices at different times. Also, both of them require LBidiSol.m, TriDiLU.m and UBidiSol.m.
In parts 3.2.3, 3.2.4 and 3.3, we make use of these functions to carry out a series of analysis.

3.2.3 Accuracy and Performance Analysis

There are two ways to evaluate the performance of the method. The first method is to set β = 1, where the CEV model becomes the BSM model and European option values can be calculated explicitly by the Black-Scholes formula. Thus, the value from the numerical PDE can be compared to the analytic value. For the second method, the numbers of discretization points M, N are increased, so as to study how the computed values converge. The results from a uniform discretization over the same range are also included to show the relative performance.

3.2.3.1 Performance Analysis – BSM Model

We consider the price of at-the-money European options with the parameters given in Table 13:

S_0	E	r	q	T	σ
100	100	4%	2%	1 year	15%

Table 13: Parameters for BSM Simulation

The Black-Scholes formula gives the call value as 6.8240 and put value as 4.8831.

Running the algorithm with different number of discretization points, we obtain the results in Table 14 and Figures 6 to 9:

Number of Discretization Points (N)	50	100	200	400	800	1600	3200	6400
Call (Non-Uniform)	6.755045	6.798766	6.817669	6.822407	6.823593	6.823889	6.823963	6.823978
Absolute Error	0.068943	0.025222	0.006319	0.001581	0.000395	0.000099	0.000025	0.000010
Call (Uniform)	6.850098	6.900183	6.845239	6.827688	6.825535	6.823850	6.824006	6.824009
Absolute Error	0.026110	0.076195	0.021251	0.003700	0.001547	0.000138	0.000018	0.000021
Put (Non-Uniform)	4.814121	4.857843	4.876745	4.881484	4.882669	4.882966	4.883040	4.883055
Absolute Error	0.068944	0.025222	0.006319	0.001581	0.000395	0.000099	0.000025	0.000010
Put (Uniform)	4.909175	4.959260	4.904316	4.886764	4.884612	4.882926	4.883083	4.883086
Absolute Error	0.026110	0.076195	0.021251	0.003700	0.001547	0.000138	0.000018	0.000021

Table 14: Simulated Call and Put Values and Absolute Error under BSM Model

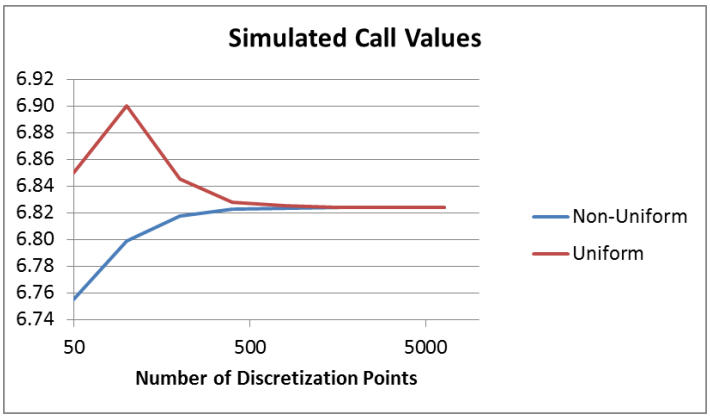


Figure 6: Simulated Call Values Under BSM Model

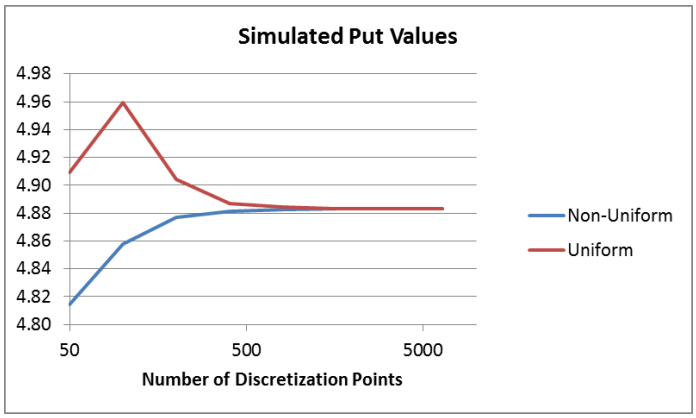


Figure 7: Simulated Put Values Under BSM Model

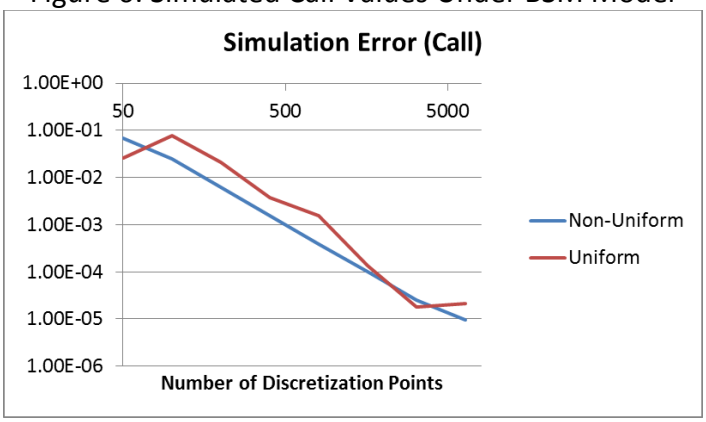


Figure 8: Simulation Error of Call Values Under BSM Model

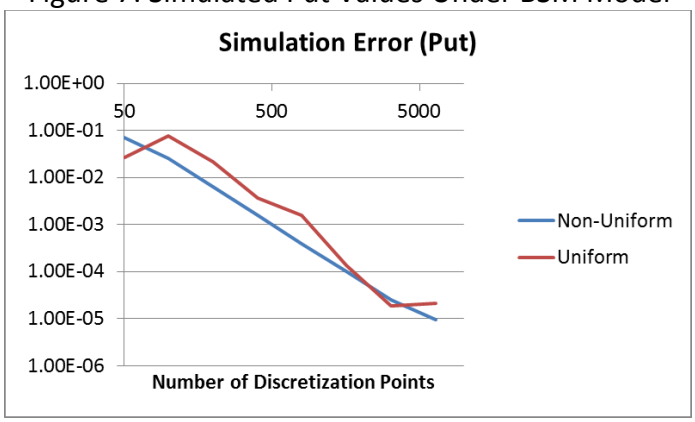


Figure 9: Simulation Error of Put Values Under BSM Model

As seen from the results, the simulated option values from the Crank-Nicolson finite difference method converge to the analytic values from the Black-Scholes formula. Furthermore, the convergence by non-uniform discretization is quicker and more stable compared to uniform discretization, hence its performance is better.

Table 14 and Figures 6 to 9 are also available at sheet "Crank-Nicolson - BSM (3.2.3)" of Calculations.xlsm.

3.2.3.2 Performance Analysis – CEV model

We now consider another case in the CEV model with β = 2, where the underlying asset price follows Absolute Diffusion Process. The option values have no closed-form solution, so we vary the step sizes for conducting convergence analysis.
The parameters for the option and the model are listed in Table 15:

S_0	E	r	q	T	α
100	100	4%	2%	1 year	20

Table 15: Parameters for CEV Simulation

By running the algorithm with different number of discretization points, we get the results in Table 16 and Figures 10 to 11:

Number of discretization points (N)	50	100	200	400	800	1600	3200	6400	12800
Call (Non-Uniform)	8.6879	8.7303	8.7469	8.7511	8.7521	8.7524	8.7524	8.7524	8.7524
Call (Uniform)	8.7636	8.8086	8.7687	8.7552	8.7536	8.7523	8.7525	8.7525	8.7524
Put (Non-Uniform)	6.7469	6.7894	6.8060	6.8101	6.8112	6.8114	6.8115	6.8115	6.8115
Put (Uniform)	6.8227	6.8676	6.8277	6.8143	6.8127	6.8114	6.8115	6.8115	6.8115

Table 16: Simulated Call and Put Values Under CEV Model

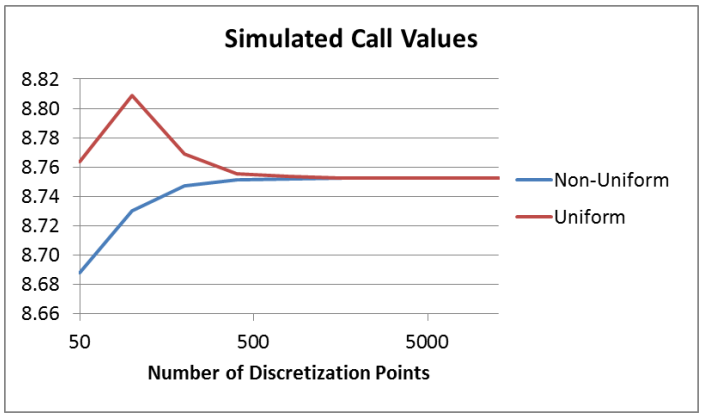


Figure 10: Simulated Call Values Under CEV Model

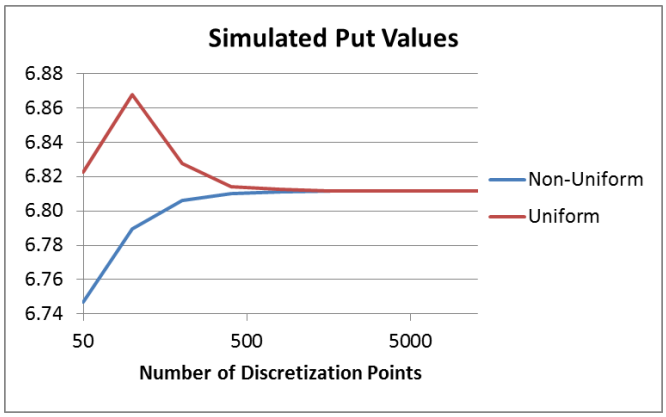


Figure 11: Simulated Put Values Under CEV Model

We see that both call and put values converge as number of discretization points increase. Thus, we take the simulated values at N = 12800 as the true value of the option, and calculate the simulation errors.
The results are shown in Table 17 and Figures 12 to 13:

N	50	100	200	400	800	1600	3200	6400
Call (Non-Uniform)	0.064530	0.022083	0.005491	0.001336	0.000297	0.000037	0.000018	0.000009
Call (Uniform)	0.011194	0.056123	0.016216	0.002778	0.001169	0.000107	0.000011	0.000013
Put (Non-Uniform)	0.064530	0.022083	0.005491	0.001336	0.000297	0.000037	0.000018	0.000009
Put (Uniform)	0.011194	0.056123	0.016216	0.002778	0.001169	0.000107	0.000011	0.000013

Table 17: Simulation Errors of Call and Put Values Under CEV Model

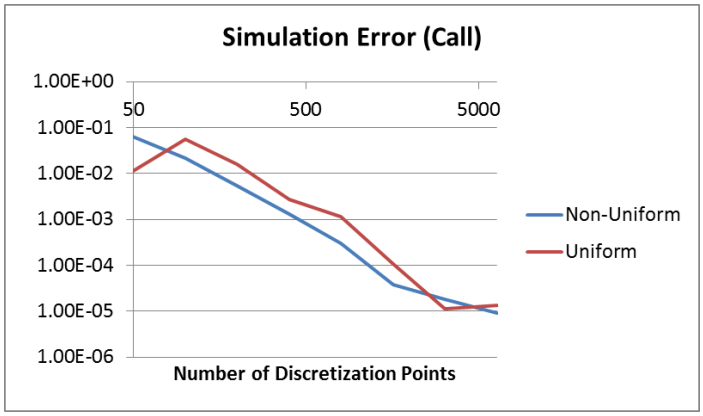


Figure 12: Simulation Error of Call Values Under CEV Model

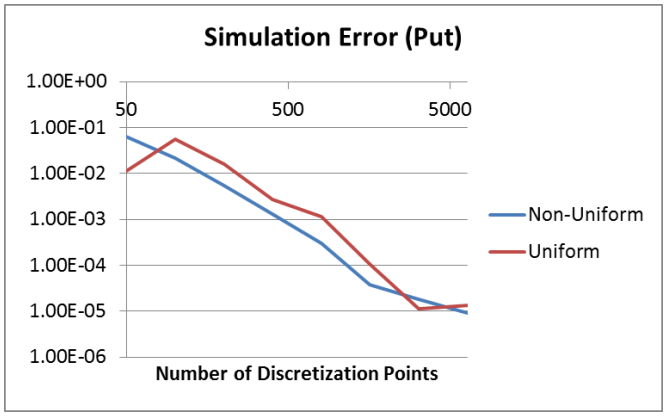


Figure 13: Simulation Error of Put Values Under CEV Model

Again, non-uniform discretization generates less error and its convergence is more stable, compared to uniform discretization.

Thus, the Crank-Nicolson finite difference method we implemented is accurate. Also, using the non-uniform discretization we specified above gives slightly better performance over uniform discretization.

Tables 16 to 17 and Figures 10 to 13 are also available at sheet "Crank-Nicolson - CEV (3.2.3)" of Calculations.xlsm.

3.2.4 Simulation Results and Discussions

With the accuracy of the Crank-Nicolson finite difference method we implemented being verified, we now investigate in the effects of changes in parameters on the option value, including time to maturity, underlying price, strike price and the variance at S = 0 under CEV model.

3.2.4.1 Variance at S = 0 under CEV model

For a CEV model, $\sigma(S) = \alpha S^{1-\beta}$. Then, when $\beta > 1$, we would have $\lim_{S \to 0} \sigma(S) = \infty$, which means that variance is undefined when S = 0. In our simulation, S = 0 is used as the boundary and the option values are explicitly given (C(0, t) = 0, $P(0,t) = Ke^{-rT} - S$), so the problem is avoided. However, we would like to see how the sharp increase in variance near 0 affects the simulation. Therefore, when doing discretization we set S^1 to be very close to 0, such that $\sigma(S^1)$ would be huge.

Using β = 2, N = 20 and the same parameters as in Table 15, we get Table 18:

S^1	0.1	0.05	0.01	0.005	0.001	0.0005	0.0001	0.00005	0.00001
$\sigma(S^1)$	200	400	2000	4000	20000	40000	200000	400000	2000000
Call	8.2498	8.2498	8.2498	8.2498	8.2498	8.2498	8.2498	8.2498	8.2498
Put	6.3089	6.3089	6.3089	6.3089	6.3089	6.3089	6.3089	6.3089	6.3089
Difference (Call)	6.72E-10	3.36E-10	6.73E-11	3.37E-11	6.72E-12	3.36E-12	6.71E-13	3.30E-13	7.11E-14
Difference (Put)	6.71E-10	3.36E-10	6.72E-11	3.36E-11	6.72E-12	3.36E-12	6.71E-13	3.30E-13	6.04E-14

Table 18: Effect of Close-to-0 Discretization Point on Simulated Option Values, S_0 = 100, N = 20

The difference when inserting a close-to-0 discretization point is insignificant, mainly due to the fact that the underlying asset price rarely decrease so much to be close to 0, hence the variance at 0 does not play an important role. Notice that the difference decreases as $\sigma(S^1)$ increases. This indicates that the theoretical value $\sigma(0) = \infty$ truly fits the model.

Then we run the simulations again except we take a smaller initial underlying price $S_0 = 80$, getting Table 19:

S^1	0.1	0.05	0.01	0.005	0.001	0.0005	0.0001	0.00005	0.00001
$\sigma(S^1)$	200	400	2000	4000	20000	40000	200000	400000	2000000
Call	2.1269	2.1269	2.1269	2.1269	2.1269	2.1269	2.1269	2.1269	2.1269
Put	19.7900	19.7900	19.7900	19.7900	19.7900	19.7900	19.7900	19.7900	19.7900
Difference (Call)	2.81E-08	1.41E-08	2.81E-09	1.41E-09	2.81E-10	1.41E-10	2.81E-11	1.41E-11	2.82E-12
Difference (Put)	2.81E-08	1.40E-08	2.81E-09	1.41E-09	2.81E-10	1.40E-10	2.81E-11	1.40E-11	2.80E-12

Table 19: Effect of Close-to-0 Discretization Point on Simulated Option Values, $S_0 = 80$, N = 20

Because the initial underlying asset price is lower, its probability of decreasing to close to 0 is relatively higher, however it is still insignificant. As shown in Table 19, the difference is greater compared to the previous scenario, but it is still not noticeable. Also note that the difference decreases as $\sigma(S^1)$ increases, similar to the previous case.

Finally we increase the number of discretization points. Using $\beta = 2$, N = 40 and parameters as in Table 15, we get Table 20:

S^1	0.1	0.05	0.01	0.005	0.001	0.0005	0.0001	0.00005	0.00001
$\sigma(S^1)$	200	400	2000	4000	20000	40000	200000	400000	2000000
Call	8.6157	8.6157	8.6157	8.6157	8.6157	8.6157	8.6157	8.6157	8.6157
Put	6.6747	6.6747	6.6747	6.6747	6.6747	6.6747	6.6747	6.6747	6.6747
Difference (Call)	3.91E-14	1.95E-14	0	0	0	0	0	0	0
Difference (Put)	4.00E-14	1.95E-14	9.77E-15	0	0	0	0	0	0

Table 20: Effect of Close-to-0 Discretization Point on Simulated Option Values, $S_0 = 100$, N = 40

The difference becomes even smaller. This is because the simulation becomes more accurate as number of discretization points increases. The effect of extreme σ at boundary diminishes.

Tables 18 to 20 are also available in sheet “Crank-Nicolson - Vol (3.2.4)” of Calculations.xlsm.

3.2.4.2 Value of Option with Respect to Time and Underlying Asset Price

We simulate the values of option with the parameters in Table 15, except for varying the underlying asset price S from 60 to 140 and time-to-maturity t from 0 to 1, using non-uniform discretization with N = M = 1600.

The simulated call values are presented in Figures 14 to 16, while the simulated put values are presented in Figures 17 to 19:

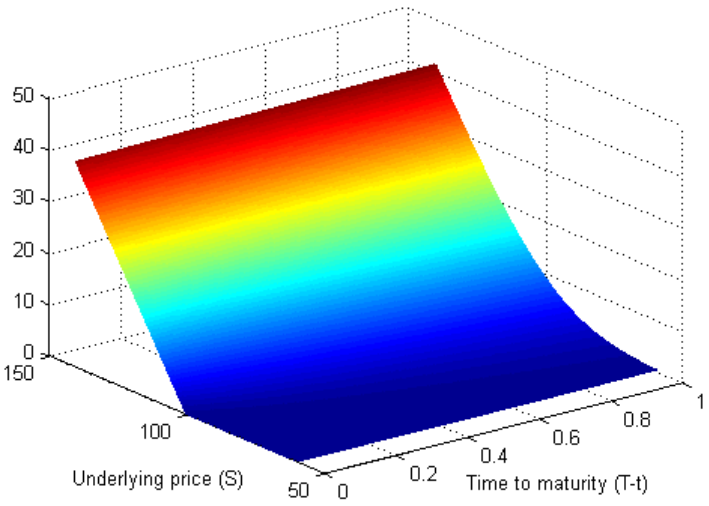


Figure 14: Simulated Call Values

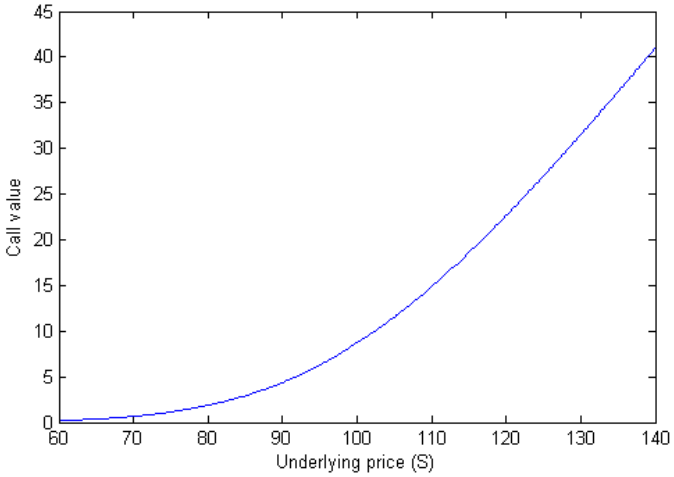


Figure 15: Simulated Call Values at t = 0

From the graph, it is evident that the value of call option increases with the underlying price. The value only slightly increases for out-of-the-money call options, and it increases at about the same rate as underlying for deep in-the-money calls.

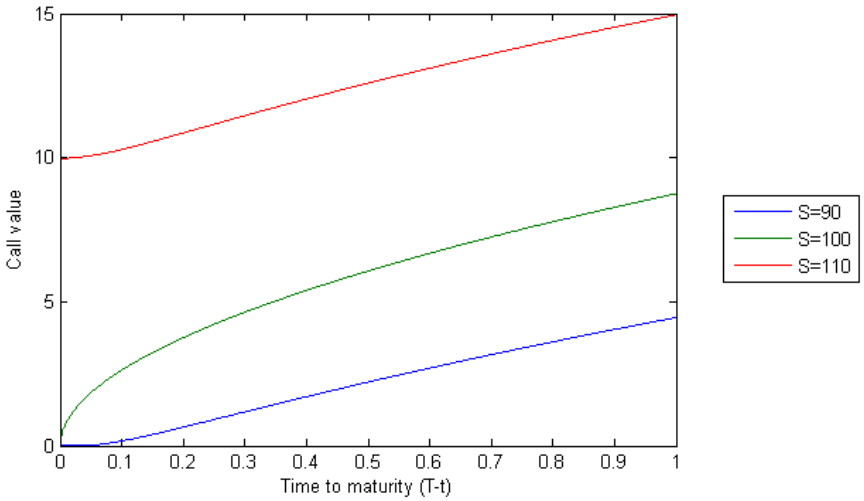


Figure 16: Simulated Call Values, S = 90, 100, 110

The above graph shows the values of three calls as a function of time to maturity. The underlying prices for the three calls are 90, 100, and 110 respectively, representing out-of-the-money, at-the-money and in-the-money call options.

For out-of-the-money call option, its intrinsic value is 0, so it only has time value which gradually decreases over time. For in-the-money call option, a large part of its value is its intrinsic value, while its time value gradually decreases over time, hence it also gradually loses value over time. As for the at-the-money call option, its value consists of time value only. In the beginning, its value decreases gradually just like the other two options. However, when it is near maturity, time value decreases rapidly, so the option value also decreases quickly to 0.

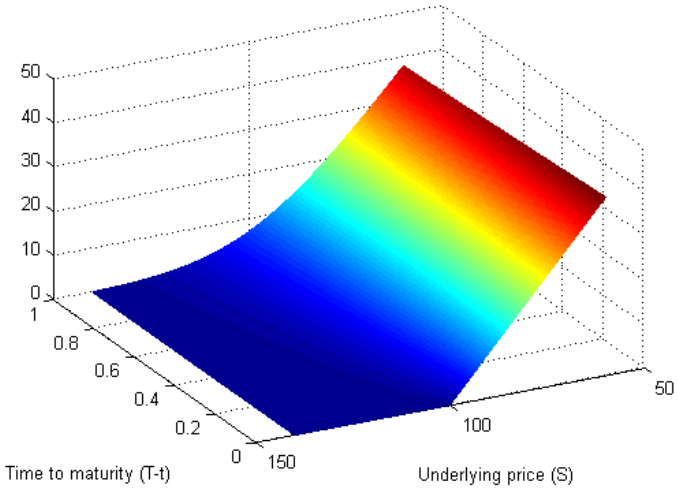


Figure 17: Simulated Put Values

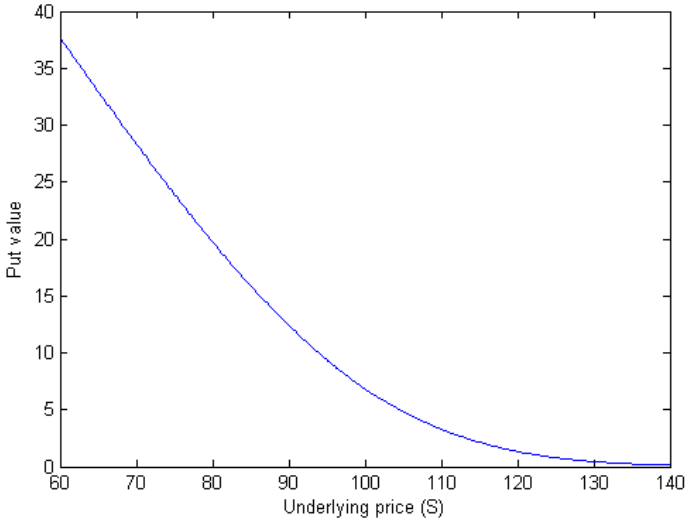


Figure 18: Simulated Put Values at t = 0

The graph shows that the value of put option decreases with the underlying price. The value decreases at about the same rate as underlying for in-the-money puts, but only slightly decreases for out-of-the-money put options.

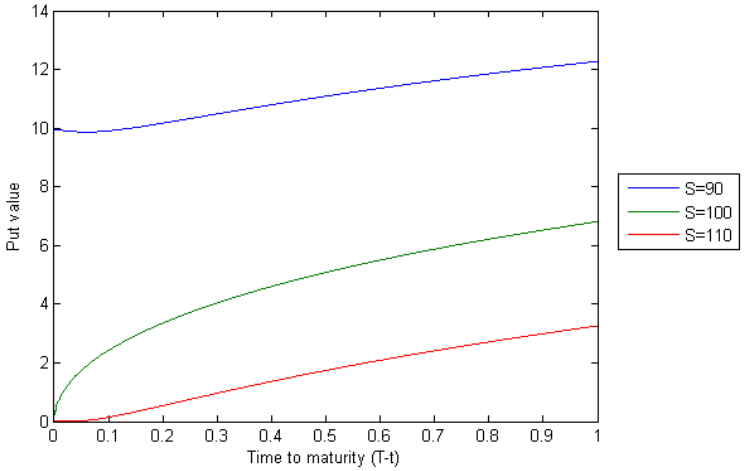


Figure 19: Simulated Put Values, S = 90, 100, 110

Again, the graph shows the values of the puts at S=90 (in-the-money), S=100 (at-the-money) and S=110 (out-of-the-money) as functions of time to maturity. Both the values of the out-of-the-money and the in-the-money put options gradually decrease to intrinsic value as time value decreases over time. Notice that the put value of the in-the-money option is slightly below intrinsic value just before maturity, due to time value of money and the fact that European puts cannot be exercised early. Similar to the case for call options, at-the-money put option loses its value gradually in the beginning but quickly near maturity.

3.2.4.3 Value of Option with Respect to Strike Price

For call options, as strike price increases, the probability of the underlying price at maturity to be above strike price decreases, which means that the option is less likely to be in-the-money at maturity. Hence, the call option value decreases with strike price.

Based on the parameters in Table 15, except for varying strike price (E), simulation with N = M = 1600 gives Table 21:

Strike Price	90	92	94	96	98	100	102	104	106	108	110
Call value	14.8493	13.4903	12.1984	10.9766	9.8273	8.7524	7.7529	6.8292	5.9811	5.2073	4.5061

Table 21: Call Option Values, Strike Price from 90 to 110

As for put options, its value increases with strike price due to a higher probability for the option to be in-the-money at maturity.
Based on the same parameters, simulation gives Table 22:

Strike Price	90	92	94	96	98	100	102	104	106	108	110
Put value	3.3005	3.8631	4.4927	5.1925	5.9648	6.8114	7.7335	8.7315	9.8049	10.9527	12.1731

Table 22: Put Option Values, Strike Price from 90 to 110

Tables 21 to 22 are also available in sheet "Crank-Nicolson - Strike (3.2.4)" of Calculations.xlsm.
The implementation for this part in MATLAB is available at part3bStrike.m.

3.3 Implied Volatility

From the Black-Scholes formula, it can be shown that given all other factors (S, E, r, q, T) fixed, option price is an increasing function of volatility (σ). Therefore, by numerical methods, volatility can be adjusted to a level that causes the option value given by the Black-Scholes formula to be equal to the actual price. This is known as the implied volatility, and it serves as a proxy for the future volatility of the underlying asset.
The implied volatility for put options with parameters as in Table 15, except for varying strike prices and maturities, are calculated with the assumption that the price of an option equals to its simulated value with size of simulation N = 1600 and M = 1600. The results are shown in Table 23 and Figure 20:

Strike Price	80	90	100	110	120
T=3 months	0.22325	0.21081	0.20008	0.19070	0.18241
T=6 months	0.22334	0.21090	0.20016	0.19078	0.18248
T=9 months	0.22343	0.21099	0.20025	0.19086	0.18256
T=1 year	0.22353	0.21108	0.20034	0.19094	0.18263

Table 23: Implied Volatilities of Put options

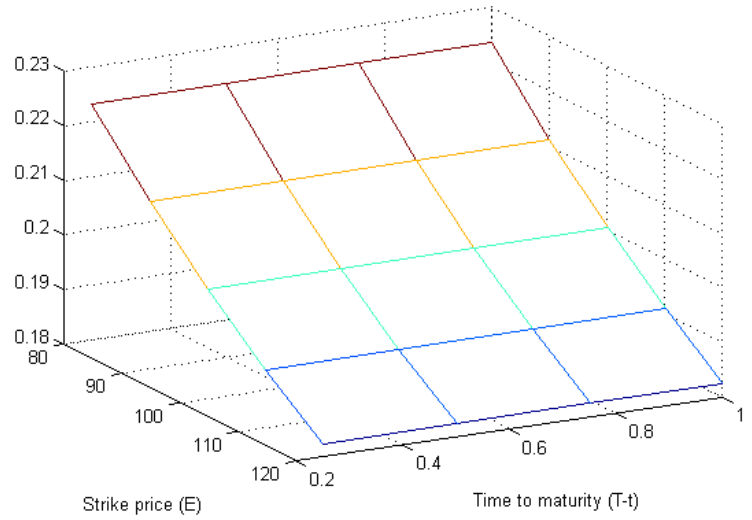


Figure 20: Implied Volatilities of Put options

From the above, it can be observed that implied volatility decreases significantly as strike price increases. Also, it increases with time to maturity, but the scale of increase is very small compared to the changes in strike price.

Table 23 is also available in sheet "Crank-Nicolson - ImpVol (3.2.4)" of Calculations.xlsm.
Implementation of this part in MATLAB is available at part3cImpVol.m.

3.4 Alternative Method Using Forward PDE

Instead of the backward PDE used above, we can regard the initial European option value V(K, T; S_0, 0) (i.e. S = S_0, t = 0) as a function of strike price K and expiry T. It can be shown that this function satisfies the following forward PDE:

$$\frac{\partial V}{\partial T} - \frac{1}{2}\sigma(K,T)^2K^2\frac{\partial^2 V}{\partial K^2} + (r-q)K\frac{\partial V}{\partial K} + qV = 0$$

Since the coefficients of $\frac{\partial^2 V}{\partial T^2}$ and $\frac{\partial^2 V}{\partial K\,\partial T}$ are both 0, this PDE is parabolic.

Then, the initial condition is the value of V when T = 0 (i.e. value of V(K, 0; S_0, 0)).

The boundary conditions are the value of V when K = 0 and K = L, where L > S_0 is the upper bound of K. (I.e. values of V(0, T; S_0, 0) and V(L, T; S_0, 0))

Clearly, when T = 0, the value of V is just the payoff; when K = 0, European put is worthless while European call simply worth the same as the current stock price; when K = L, European call is worthless while European put has the value of $Le^{-rT} - S_0$.

The conditions are summarized in Table 24:

	European Call	European Put
Initial Condition	$\max(S_0 - K, 0)$	$\max(K - S_0, 0)$
Boundary Condition (Lower)	S_0	0
Boundary Condition (Upper)	0	$Le^{-rT} - S_0$

Table 24: Initial Conditions and Boundary Conditions for the Forward Parabolic PDE

Using this forward PDE, we can produce another MATLAB function, with the signature

`V = MyCallPut2(S0, r, q, flag)`

which returns an $(N+1) \times (M+1)$ matrix of option values for different strike prices and maturities at time 0, keeping initial underlying price constant.

On the other hand, the MATLAB function created in previous part returns an $(N+1) \times (M+1)$ matrix of option values for different initial underlying price and time to maturity, keeping strike price constant.

Hence, when the initial values of options for many different strike prices and maturities are needed, only 1 (or 2, if both call and put are needed) call of MyCallPut2 is required, but X (or 2X, if both call and put are needed) calls of MyCallPut2 are required, where X is the number of different strike prices. Hence, the forward PDE method is computationally more efficient.

On the other hand, when the initial as well as future option values for a single option are needed, only 1 or 2 calls of MyCallPut is required, while Y or 2Y calls of MyCallPut2 are required, where Y is the number of different maturities. Hence, the backward PDE method is computationally more efficient.

4. Option Pricing with Monte Carlo Method

To investigate the pricing ability of Monte Carlo method further, we consider the valuation of various exotic options by Monte Carlo simulation, including pricing of barrier options, accumulators, and basket options.

In addition, we will analyze the performance of dynamic hedging strategy by Monte Carlo simulation and the finite difference method used in part 3.

4.1 Barrier Options

For this part, we consider a barrier option, with parameters as shown in Table 25:

Option Type	European Up-And-Out Call
Number of Assets: L	2
Strike Price: K	90
Barrier Price: S_u	120
Time to Expiry (in Years): T	1
Current Time: t	0
Volatility: σ	20%
Expected Rate of Return: μ	15%
Risk-free Interest Rate: r	5%
Note	The barrier has not been reached at time 0.

Table 25: Specification of the Barrier Option

I.e. From t = 0 to T, if the stock price ever passes above the barrier, payoff = 0;
If the stock price never passed above the barrier, payoff is the same as the corresponding European call.

4.1.1 Analytic Formula of European Up-And-Out Call Option

The analytic solution of European up-and-out call option value is

$$V_{out}(S,t) = S\left(\mathcal{N}(d_1) - \mathcal{N}(d_3) - \left(\frac{S_u}{S}\right)^{1+\frac{2r}{\sigma^2}}\left(\mathcal{N}(d_6) - \mathcal{N}(d_8)\right)\right)$$
$$- Ke^{-r(T-t)}\left(\mathcal{N}(d_2) - \mathcal{N}(d_4) - \left(\frac{S_u}{S}\right)^{-1+\frac{2r}{\sigma^2}}\left(\mathcal{N}(d_5) - \mathcal{N}(d_7)\right)\right)$$

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}, d_2 = \frac{\ln\left(\frac{S}{K}\right) + \left(r - \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

$$d_3 = \frac{\ln\left(\frac{S}{S_u}\right) + \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}, d_4 = \frac{\ln\left(\frac{S}{S_u}\right) + \left(r - \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

$$d_5 = \frac{\ln\left(\frac{S}{S_u}\right) - \left(r - \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}, d_6 = \frac{\ln\left(\frac{S}{S_u}\right) - \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

$$d_7 = \frac{\ln\left(\frac{SK}{S_u^2}\right) - \left(r - \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}, d_8 = \frac{\ln\left(\frac{SK}{S_u^2}\right) - \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$$

Then, implementing the analytic formula in MATLAB, we obtain Table 26 and Figure 21:

Initial Underlying Price: S0	Analytic Value
80	2.445796
82	2.762320
84	3.060266
86	3.328128
88	3.555146
90	3.731873
92	3.850638
94	3.905885
96	3.894375
98	3.815244
100	3.669940
102	3.462038
104	3.196967
106	2.881680
108	2.524279
110	2.133625
112	1.718963
114	1.289570
116	0.854439
118	0.422018
120	0.000000

Table 26: Analytic Value of the European Up-and-Out Call with Different Initial Underlying Price (Selected)

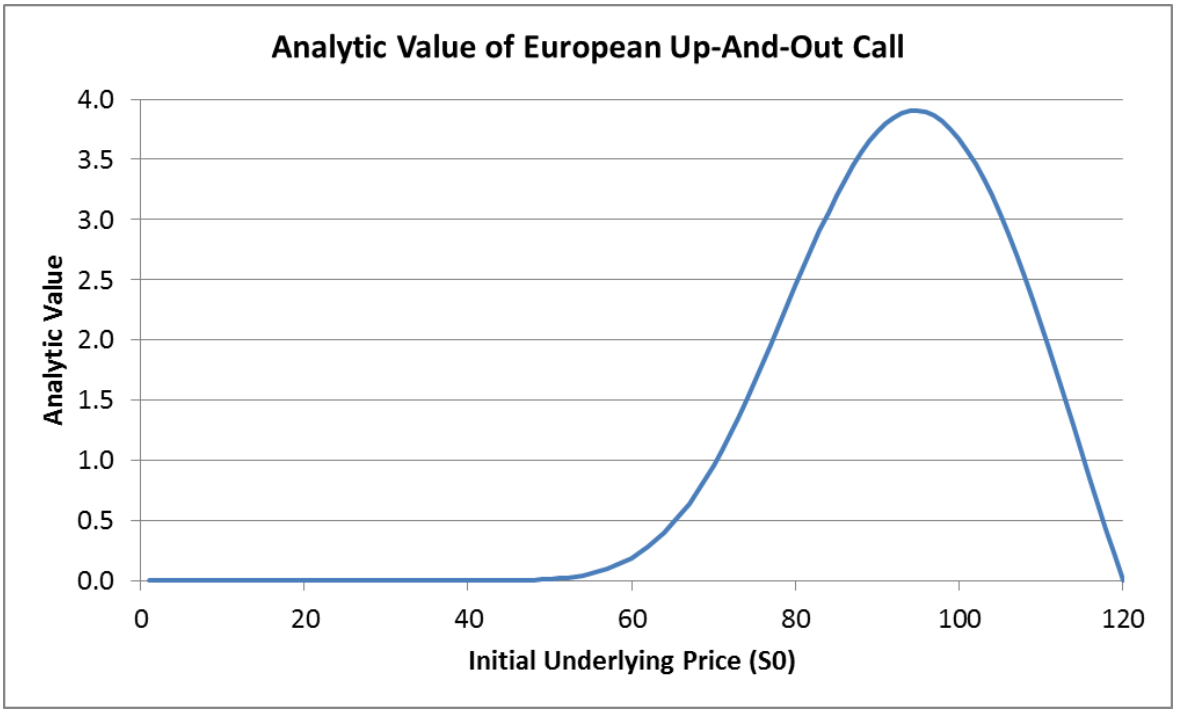


Figure 21: Plot of Initial Up-and-Out European Call Option Value V(S,t) against S

Using the fminbnd function in MATLAB, we find that the maximum value of the option is 3.909558, attained at $S_0 = 94.66$.

The implementation of the analytic solution in MATLAB is available at part4aUpAndOutAnalytic.m, and the full Table 26 and Figure 21 is available in the sheet "Up&Out Call-Analytic (4.1.1)" of Calculations.xlsm.

4.1.2 Monte Carlo Simulation of European Up-And-Out Call Option

For illustration purpose, we only consider the case where initial underlying price is $S_0 = 100$.

Setting $\Delta t = 0.001$ and $M = \frac{T}{\Delta t} = 1000$, we have

$$\Delta S_t = S_t(\mu \Delta t + \sigma\sqrt{\Delta t} Z)$$
$$S_{t+1} = S_t(1 + \mu \Delta t + \sigma\sqrt{\Delta t} Z)$$

where Z is a Gaussian random number with mean 0 and variance 1, t = 1, 2, 3, …, M.

We generate N = 10000 paths, and then the discounted payoff for each path is calculated.

Mean of the discounted payoff is the value of the option.

Then, we repeat the above Monte Carlo simulation process for 200 times to obtain 200 estimates of the option value, and find the mean and standard deviation of the estimates:

Mean of the estimates of initial call value = 3.6516

SD of the estimates of initial call value = 0.0680

Comparing with the value using the analytic formula, which is 3.6699, we can see that the difference is within one standard deviation, which means the estimation is very close to the analytic value.

The implementation in MATLAB is available at part4aUpAndOutMonte.m. It is a function that accepts S_0 as input, and outputs the mean of the estimates of initial call value, and the standard deviation of the estimates.

4.1.3 Application: Accumulator

Accumulator is a derivative where, while the end-of-day underlying asset price does not rise to or above the knock-out price S_u, the buyer is obliged to buy a pre-specified quantity of the underlying security from the seller at a predetermined reference price K every trading day, settled by cash at the end of each month, until maturity T. The pre-specified quantity is usually different when the asset price is above or below the reference price, and is usually higher when asset price is below reference price. Below we assume the quantity purchased per day is 1 when the asset price is at or above the reference price, and 3 when the asset price is below the reference price.

When the end-of-day underlying asset price is at or above the knock-out price, the accumulator contract early terminates.

At the end of every month, the payoff is

$$\text{Payoff} = \text{Quantity purchased} \times (\text{Asset price at end of month} - K)$$

Note that there are two types of accumulators:

- Normal accumulator – Simply the description above
- Honeymoon accumulator – Similar to normal accumulator, but:
  If the contract is early terminated in the first month, then quantity purchased equals the number of trading days of the first month.
  When the underlying asset price falls below reference price in the first month, quantity purchased is still the same as when the asset price is above reference price.

In this way, honeymoon accumulator has the same payoff with normal accumulator after the first month, and higher payoff than normal accumulator (unless the end-of-month price is between reference price and knock-out price). Hence, it should have a higher value when volatility is high.

In this part, we replicate the analysis in Yen and Xiang (2008), which is on the relationship between accumulator value and volatility, and accumulator payoff and days until termination.

For the first part, we consider an accumulator with parameters specified in Table 27:

Initial price: S	22151.06
Maturity: T	1.00
Interest Rate: r	5%
Expected Return: μ	0.2877%
Reference Price: K	19935.95
Knock-Out Price: S_u	23258.61
Dividend yield: q	0%

Table 27: Specification of the Accumulator

To estimate the value of the accumulator, we generated 500000 future price trajectories by Monte Carlo simulation, with number of random prices M = 250, time interval $\Delta t = \frac{T}{M} = \frac{1}{250}$, and calculated the discounted payoff for each trajectory. The mean discounted payoff of the trajectories is the estimated value of the accumulator.
Then, varying volatility from 0% to 80%, we obtain Figure 22:

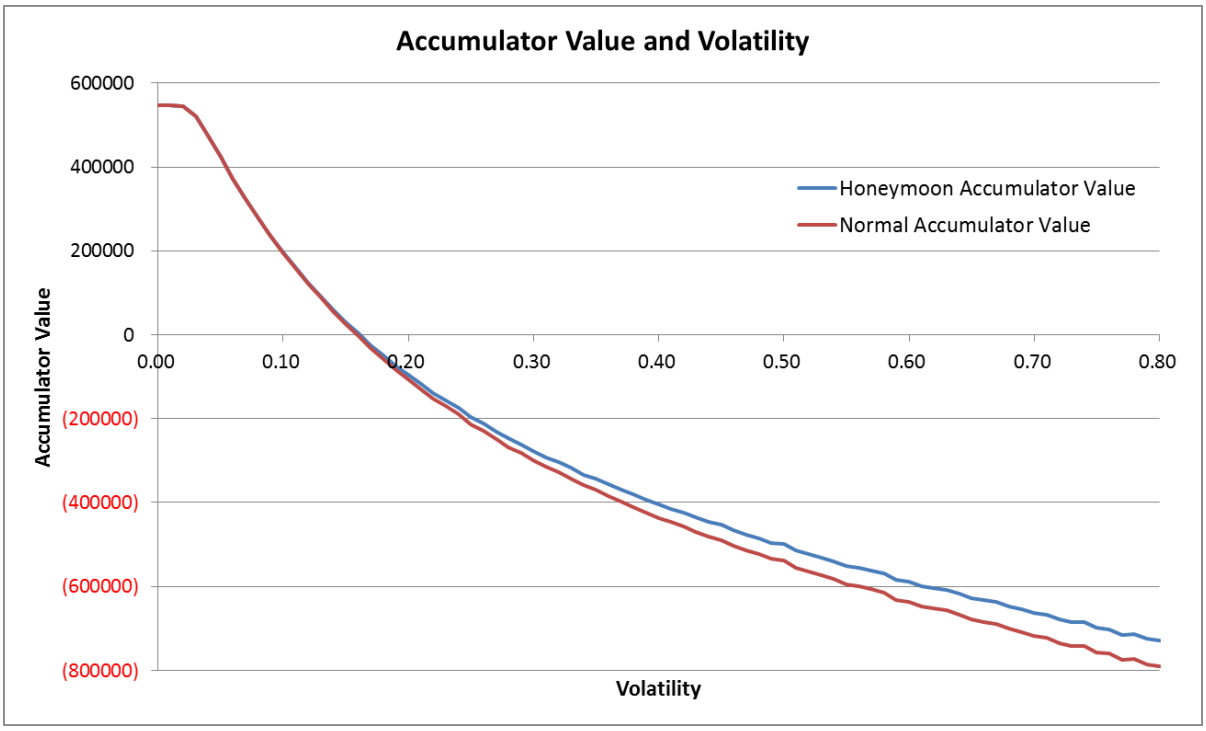


Figure 22: Accumulator Value with Different Volatilities

From the graph, we can conclude that both honeymoon and normal accumulator value decreases when volatility increases, and eventually falls to negative; also, the value of honeymoon accumulator is always the same or above the value of normal accumulator, and the difference increases as volatility increases.

Using the same method, this time fixing volatility at 15.0987% and varying expected return μ, we obtain Figure 23:

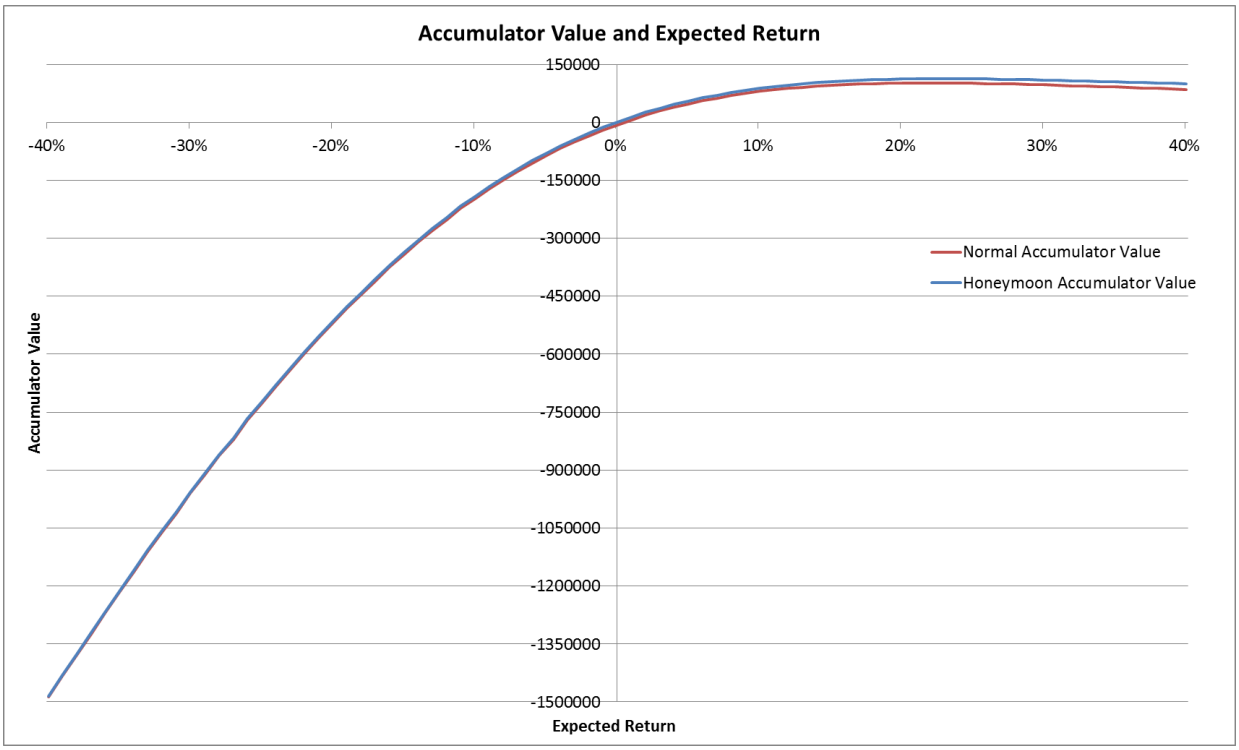


Figure 23: Accumulator Value with Different Expected Return

From the graph, we can conclude that both values of honeymoon and normal accumulator increases when expected return increases, hence making it attractive in bull markets. However, it gradually levels off and decreases when expected return is very high, since the probability of early termination is high, adversely affecting the payoff.
Also, the maximum value of the accumulator is about 100000 for normal accumulator and 110000 for honeymoon accumulator, but the loss in a market downturn can be much higher than this value, reaching more than a million in extreme cases, hence the riskiness of accumulators.
The value of honeymoon accumulator is always the same or above the value of normal accumulator, and the difference increases as expected return increases.

For the second part, we consider 8 accumulators with parameters given in Table 28:

Accumulator	1	2	3	4	5	6	7	8
Initial price: S	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06	22151.06
Maturity: T	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Volatility: σ	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%	15.9087%
Interest Rate: r	5%	5%	5%	5%	5%	5%	5%	5%
Expected Return: μ	0.2877%	0.2877%	0.2877%	0.2877%	0.2877%	0.2877%	0.2877%	0.2877%
Reference Price: K	19935.95	17720.85	19935.95	17720.85	19935.95	17720.85	19935.95	17720.85
Knock-Out Price: S_u	22594.08	22594.08	23258.61	23258.61	22594.08	22594.08	23258.61	23258.61
Dividend yield: q	0%	0%	0%	0%	1%	1%	1%	1%

Table 28: Specification of the Accumulators

Using the same method as above, we generated 500000 future price trajectories by Monte Carlo simulation, with number of random prices M = 250, time interval $\Delta t = \frac{T}{M} = \frac{1}{250}$. Then the undiscounted payoff for each trajectory is calculated, and a scatter plot of payoff against date of termination for each accumulator is shown in Figure 24:

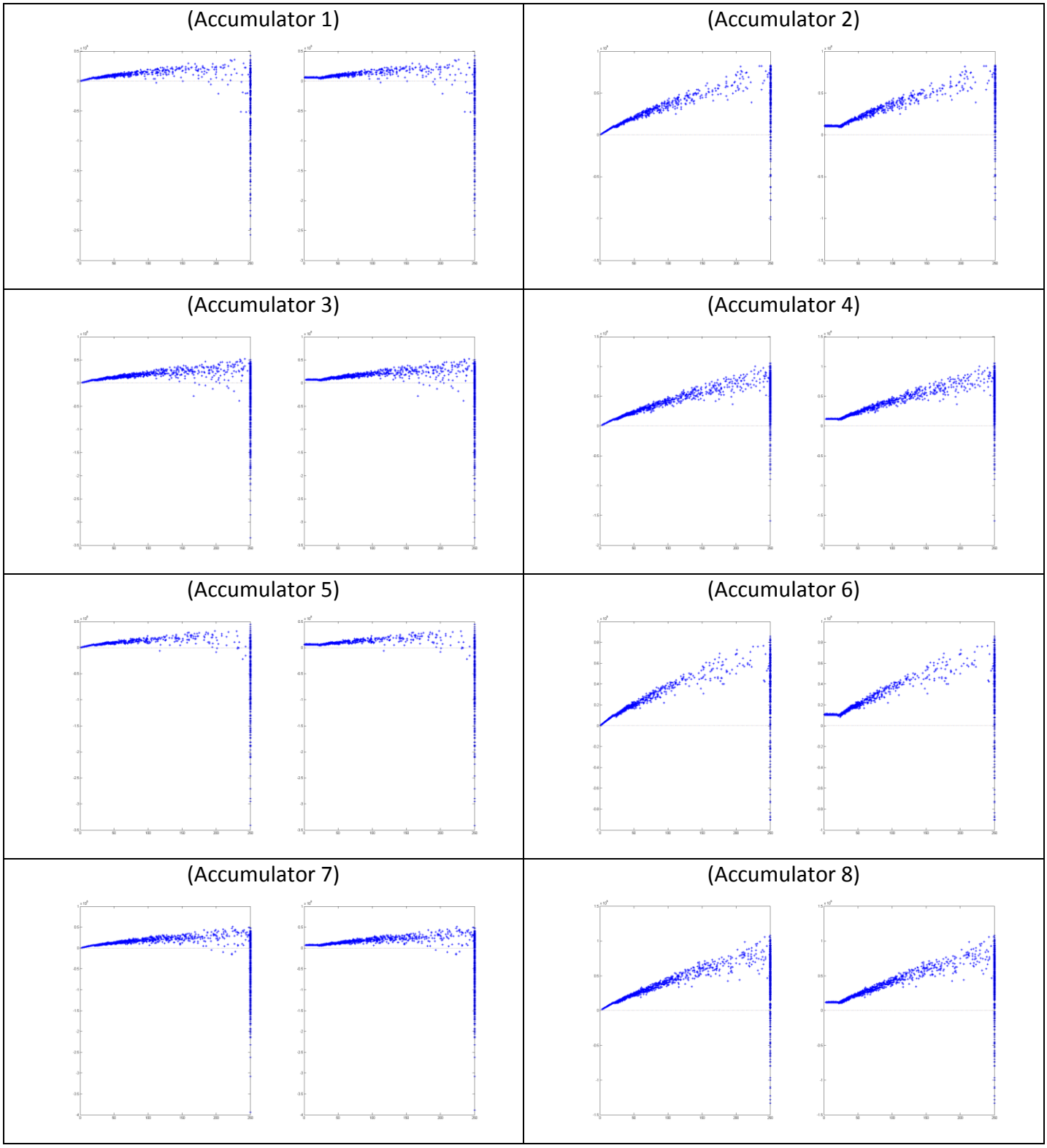


Figure 24: Accumulator Payoff and Date of Contract Termination
(For each accumulator, the graph on the left corresponds to normal accumulator, while the graph on the right corresponds to honeymoon accumulator)

From Figure 24, we can see that when the contract is early terminated, the payoff tends to be positive but small; when the contract is not early terminated, the loss can be huge. This is consistent with the results of Yen and Xiang (2008).

MATLAB Functions that output honeymoon and normal accumulator value and accepting different types of input are available in part4aAccVol.m and part4aAccMu.m, with the following signatures:

```
[V1,V2] = part4aAccVol(trials,sigma)
[V1,V2] = part4aAccMu(trials,mu)
```

where trials = number of trajectories, sigma = volatility, mu = expected return,
V1 = normal accumulator value, V2 = honeymoon accumulator value.
An implementation in MATLAB for the plots of payoff against date of contract termination is available at part4aAccPay.m.

A table of the accumulator values as volatility varies (i.e. source data for Figure 22) is available in the sheet "Accumulator-Sigma (4.1.3)" of Calculations.xlsm; a table of the accumulator values as expected return varies (i.e. source data for Figure 23) is available in the sheet "Accumulator-Mu (4.1.3)" of Calculations.xlsm.

4.2 Hedging Analysis

Under the Black-Scholes-Merton option pricing model, we assume that hedging is continuous so option risk is completely eliminated. However, in practice, this is not possible due to transaction costs.
Therefore, in this part, we determine the hedging error of a delta hedging portfolio with different rebalancing frequencies.

4.2.1 Finding the Hedging Error

Assume

- Dividend yield q = 0%
- Continuously compounding risk-free interest rate r is constant at 4%
- Initial asset price $S_0 = 100$
- Rate of return $\mu = 15\%$
- Assume the underlying price follows the CEV model below:
$$\frac{dS}{S} = (\mu - q)dt + \sigma(S,t)dW, \sigma(S,t) = \min\left(100, \alpha S^{1-\beta}\right)$$
where $\alpha = 20$, $\beta = 2$.
- Discretizing, it becomes
$$\Delta S = S \times \left((\mu - q)\Delta t + \sigma(S,t)\sqrt{\Delta t}\varepsilon_i\right)$$
where ε_i is a random number following N(0,1).
- Assume that an at-the-money European option with expiry T is sold at time 0.
- Assume there is no transaction cost.

Now our strategy is delta-hedging, which involves trading the (risky) underlying asset and a riskless bond with maturity T and face value 1, at a set of fixed times $0 = t_0 < t_1 < \dots < t_M = T$
where M is the number of hedging times.
Let S_k be the underlying asset value at time t_k, V_k be the option value at time t_k.
Let ξ_k be the number of underlying asset held, and η_k be the number of bonds held, at time t_k, k = 0, 1, 2, …, M. Denote the strategy at time t_k as $\{\xi_k, \eta_k\}$.
Then, value of hedging portfolio at time t_k is $P_k = \xi_k S_j e^{-rt_k} + \eta_k$, and the cumulative gain of strategy $\{\xi_k, \eta_k\}$ at time t_k is given by

$$G_k = \sum_{j=0}^{k-1} \xi_j\left(S_{j+1}e^{-rt_{j+1}} - S_j e^{-rt_j}\right), G_0 = 0$$

Cumulative cost of the strategy $\{\xi_k, \eta_k\}$ at time t_k is given by

$$C_k = P_k - G_k$$

We would like to make the strategy self-financing (no in or out cash flows at rebalancing times), i.e.

$$C_0 = C_1 = \cdots = C_M$$

It results in

$$P_k = P_0 + G_k = P_0 + \sum_{j=0}^{k-1} \xi_j \left(S_{j+1} e^{-rt_{j+1}} - S_j e^{-rt_j} \right)$$

$$\eta_k = \eta_{k-1} + \xi_k S_k e^{-rt_k} - \xi_k S_k e^{-rt_k}$$

For the dynamic delta hedging strategy, we have

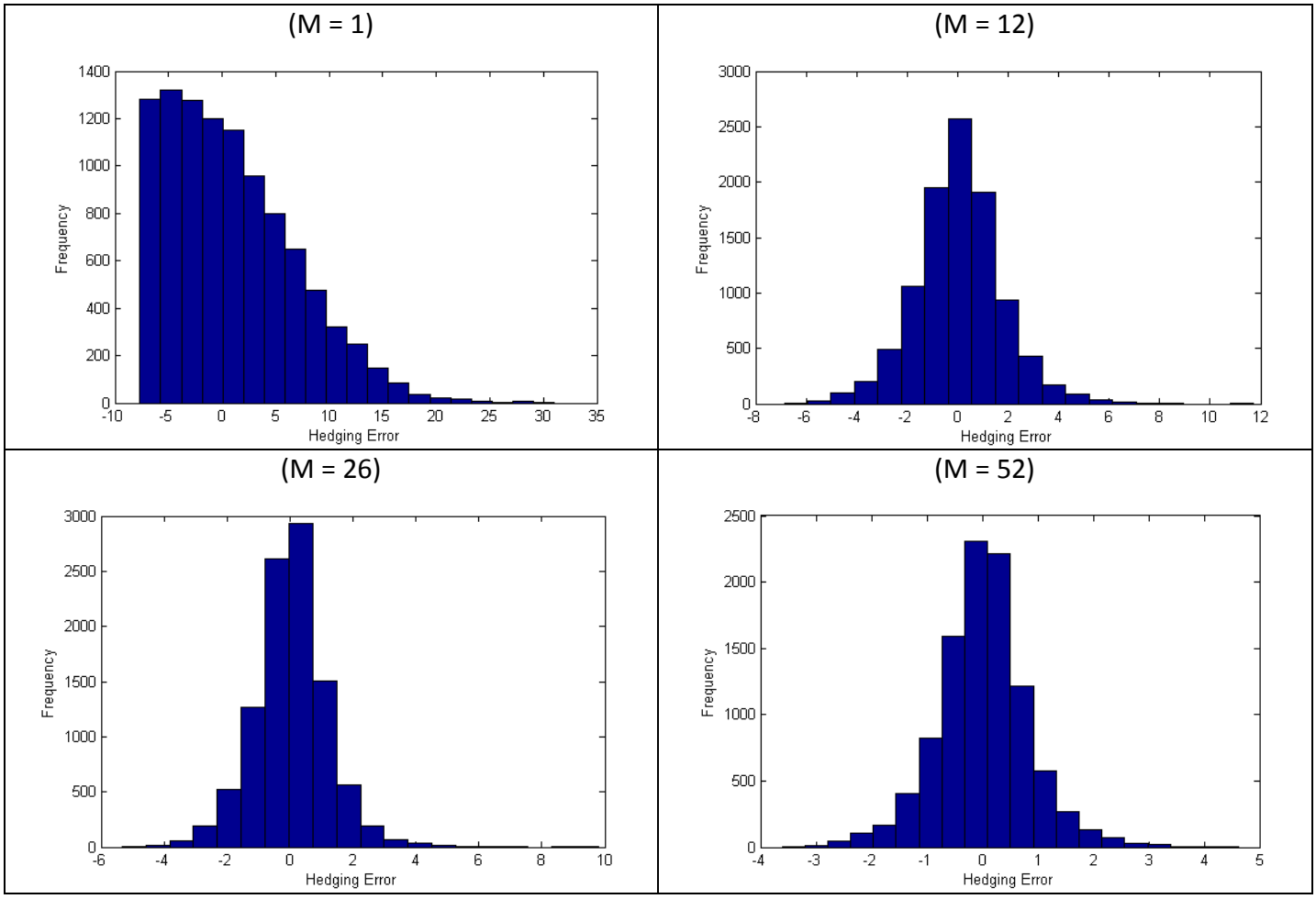
$$\xi_k = \left(\frac{\partial V}{\partial S} \right)_{t=t_k, S=S_k} \approx \frac{V_k^{i+1} - V_k^{i-1}}{S^{i+1} - S^{i-1}}, \eta_0 = V_0 - \xi_0 S_0$$

Now, by modifying the code in part 3, we can produce a matrix of approximate delta values for the ATM put at discretization points using the formula above. For other points, the respective delta values can be approximated through linear interpolation with surrounding discretization points.
Thus, at time T, hedging error of a dynamic delta hedging strategy is

$$\text{Error} = e^{-rT} V_T - \left(\xi_0 S_0 + \eta_0 + \sum_{j=0}^{M-1} \xi_j \left(S_{j+1} e^{-rt_{j+1}} - S_j e^{-rt_j} \right) \right)$$

And we can use Monte Carlo simulations to investigate the distribution of hedging error.

Generating 10000 stock price paths by Monte Carlo simulation, we get the distribution of hedging error as in Figure 25, and the statistics as in Table 29:



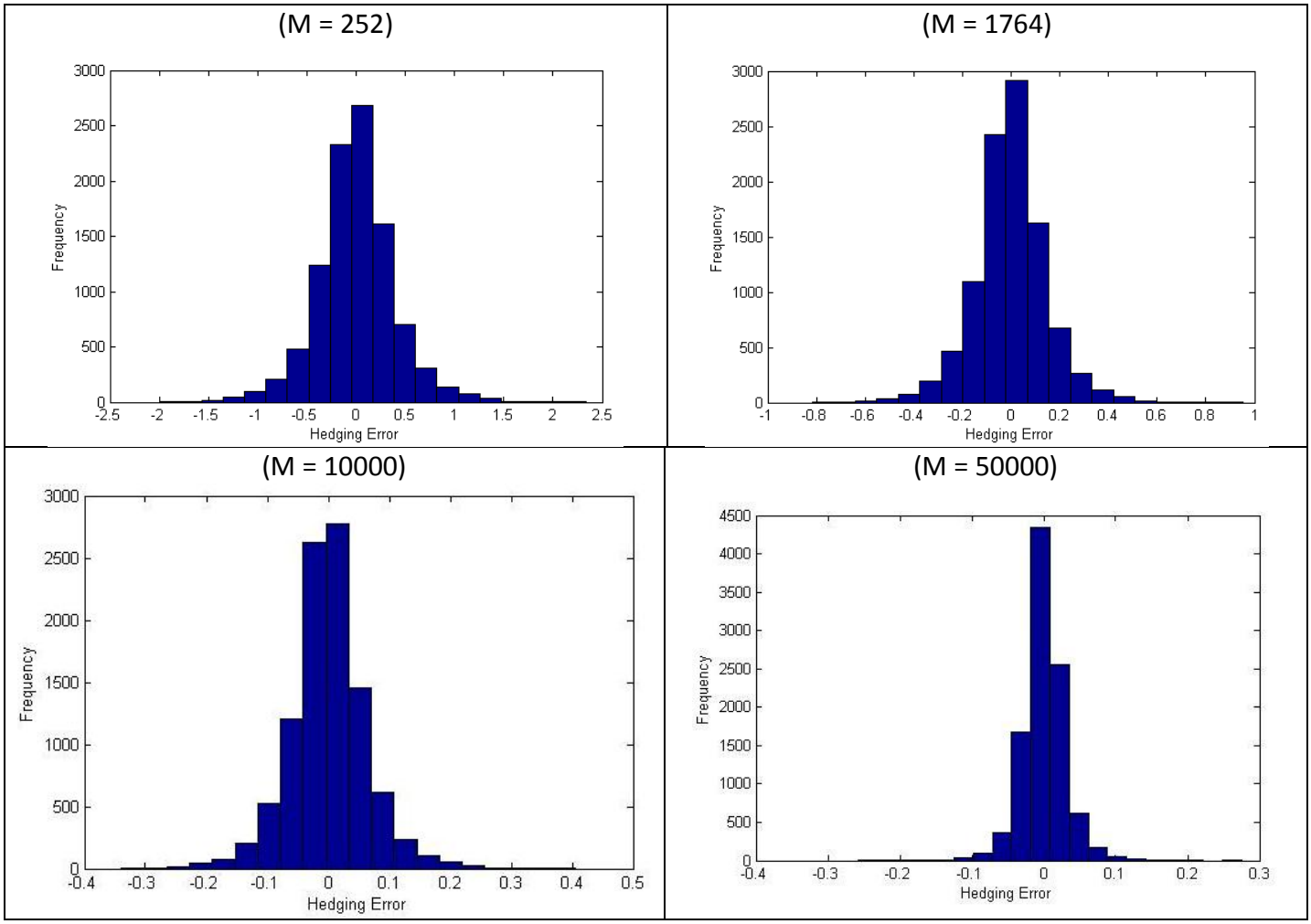


Figure 25: Distribution of Hedging Error

M	1	12	26	52	252	1764	10000	50000
Mean	0.8873	0.0656	0.0403	0.0127	0.0030	-0.0017	-0.0004	-0.0001
Standard Deviation	5.9909	1.7075	1.1927	0.8367	0.3902	0.1488	0.0640	0.0317
95% Quantile	12.1500	2.8874	1.9212	1.3826	0.6440	0.2377	0.1016	0.0475
95% Shortfall	-6.9331	-2.7594	-1.9107	-1.3546	-0.6156	-0.2469	-0.1022	-0.0462

Table 29: Statistics of Hedging Error

For the first case, we consider the delta hedging portfolio to be rebalanced weekly, hence M = 52. The mean of the error is 0.0127, which is roughly 0.2% of the option value (V_0 = 6.0172). Standard deviation of hedging error is 0.8367, while the 95% quantile and 95% shortfall are 1.3826 and -1.3546 respectively. The mean is relatively small, while standard deviation is still considerable.

When rebalancing is done less frequently, such as once every two weeks (M = 26), once every month (M = 12), or just once per year (M = 1), the distribution becomes more dispersed. In the extreme case of only once per year, the 95% quantile and 95% shortfall are 12.1500 and -6.9332 respectively. Meanwhile, by increasing the rebalancing frequency, hedging error becomes more concentrated around 0. For example, when rebalancing is done daily (M = 252), the mean becomes 0.0030 and standard deviation becomes 0.3902. If rebalancing is done hourly, assuming 7 trading hours per trading day (M = 1764), the mean becomes -0.0017, standard deviation reduces to 0.1488, and the 95% quantile and 95% shortfall are only 0.2377 and -0.2469. The more frequently rebalancing is done, the more closely the portfolio follows the option value, reducing the mean and volatility of the hedging error. Indeed, when M = 50000 (which is close to continuous), the mean becomes -0.0001, which is very close to zero.

While the simulation shows that hedging error can be minimized by ultra-high frequency rebalancing, in real life this is not feasible due to the presence of transaction costs, which makes frequent rebalancing prohibitively costly.

Table 29 is also available in sheet "Hedging Error (4.2)" of Calculations.xlsm.

For the implementation in MATLAB, MyCallPutExp.m is modified into MyCallPutDelta.m, which accepts the same input but this time outputs the initial option value, a matrix of option delta at different underlying asset price and time to maturity, and an array of the grid points. getDelta.m is a function which makes use of these output to find the delta value at a specific point, using linear interpolation or extrapolation when the point is not a discretization point. SimulateHedgingError.m is the program for generating price paths, determining the corresponding delta at each rebalancing time (by calling getDelta), determining the hedge error for each price path, displaying time T hedge error distribution graphically, as well as calculating the related statistics.

4.3 Multi-Asset Options

Consider a basket option with option parameters specified in Table 30:

Option Type	European Call	
Number of Assets: L	2	
Strike Price: K	Equals S_0, i.e. 22151.06	
Time to Expiry (in Years): T	1	
Correlation: ρ	0.5	
Risk-free Interest Rate: r	5%	
Asset Number: i	1	2
Asset Weight: w_i	0.5	0.5
Initial Price: $S_i(0)$	22151.06	22151.06
Volatility: σ_i	0.3	0.3

Table 30: Specification of the Basket Option

Payoff of the above function at maturity is specified by

$$f(S_1, S_2, T) = \max\left(\sum_{i=1}^{L} w_i S_i(T), 0\right) = \max\left(\frac{S_1(T) + S_2(T)}{2}, 0\right)$$

For simplicity purpose, assume the asset price dynamics in the risk-neutral world follow

$$\frac{dS_t^{(1)}}{S_t^{(1)}} = rdt + \sigma_1 dW_t^{(1)}, \dots, \frac{dS_t^{(L)}}{S_t^{(L)}} = rdt + \sigma_L dW_t^{(L)}$$

$$\hat{\mathbb{E}}\left(dW_t^{(j)} dW_t^{(k)}\right) = \rho_{j,k} dt, \rho_{j,j} = 1 \text{ for all } j, k \text{ in } 1,2, \dots, L$$

4.3.1 Establish a Lower Bound of Basket Option Value Using Geometric Basket Option

By AM-GM inequality, $f(S_1, S_2, T) \geq \max\left(\sqrt{S_1(T)S_2(T)}, 0\right)$ which is the payoff of the corresponding geometric basket option. Then, we can establish a lower bound for the basket option value since European geometric basket call option has an analytic solution as follows:

$$\text{Option value} = e^{-rT}\hat{\mathbb{E}}(G(T) - K)^+ = e^{-rT}\hat{\mathbb{E}}\big(G(T)\big)\Phi(d_1) - Ke^{-rT}\Phi(d_2)$$

where

$$G(t) = \sqrt{S_1(t)S_2(t)}$$

$$\hat{\mathbb{E}}\big(G(T)\big) = G(t)\exp\left(\left(r - \frac{1}{2L}\sum_{i=1}^{L}\sigma_i^2\right)(T-t) + \frac{1}{2}\sigma^2(T-t)\right)$$

$$\sigma^2 = \frac{1}{L^2}\sum_{i,j=1}^{L}\rho_{i,j}\sigma_i\sigma_j$$

$$d_1 = \frac{\ln\left(\frac{G(t)}{K}\right) + \left(r + \sigma^2 - \frac{1}{2L}\sum_{i=1}^{L}\sigma_i^2\right)(T-t)}{\sigma\sqrt{T-t}}$$
$$d_2 = d_1 - \sigma\sqrt{T-t}$$

Using the above formula, the geometric basket call option value is 2661.61, which acts as a lower bound for the basket option value.

The implementation in MATLAB is available in part4cGeometric.m.

4.3.2 Calculate Basket Option Price Using Binomial Model

To price the basket option in binomial model, we assume that in each period with $\Delta t = \frac{T}{M}$, the price of asset i (i = 1, 2, …, L) can only go up by a factor u_i or down by a factor d_i, where $u_i d_i = 1$. Then a binomial "pyramid" can be generated with different number of periods.

The boundary condition is $V\left(S_1 u_1^i d_1^{M-i}, S_2 u_2^j d_2^{M-j}, T\right) = f\left(S_1 u_1^i d_1^{M-i}, S_2 u_2^j d_2^{M-j}, T\right)$.

We choose $u_i = e^{\sigma_i\sqrt{\Delta t}}, d_i = e^{-\sigma_i\sqrt{\Delta t}}$.

Considering correlation and identifying the first two moments,

$$p_1 = \frac{1}{4}\left(1 + \rho + \left(\frac{r - \frac{\sigma_1^2}{2}}{\sigma_1} + \frac{r - \frac{\sigma_2^2}{2}}{\sigma_2}\right)\sqrt{\Delta t}\right), p_2 = \frac{1}{4}\left(1 - \rho + \left(\frac{r - \frac{\sigma_1^2}{2}}{\sigma_1} - \frac{r - \frac{\sigma_2^2}{2}}{\sigma_2}\right)\sqrt{\Delta t}\right)$$
$$p_3 = \frac{1}{4}\left(1 - \rho + \left(-\frac{r - \frac{\sigma_1^2}{2}}{\sigma_1} + \frac{r - \frac{\sigma_2^2}{2}}{\sigma_2}\right)\sqrt{\Delta t}\right), p_4 = \frac{1}{4}\left(1 + \rho - \left(\frac{r - \frac{\sigma_1^2}{2}}{\sigma_1} + \frac{r - \frac{\sigma_2^2}{2}}{\sigma_2}\right)\sqrt{\Delta t}\right)$$

Then, the binomial model for the basket option is

$$V(S_1, S_2, t) = e^{-r\Delta t} \times$$
$$\left(p_1 V(S_1 u_1, S_2 u_2, t + \Delta t) + p_2 V(S_1 u_1, S_2 d_2, t + \Delta t) + p_3 V(S_1 d_1, S_2 u_2, t + \Delta t) + p_4 V(S_1 d_1, S_2 d_2, t + \Delta t)\right)$$

Substituting the option parameters specified above, we have

$$u_1 = 1.0135, d_1 = 0.9867, u_2 = 1.0135, d_2 = 0.9867$$
$$p_1 = 0.3754, p_2 = 0.1250, p_3 = 0.1250, p_4 = 0.3746$$

Implementing the binomial model in MATLAB, and varying the number of lattice steps, we obtain Table 31 and Figure 26:

Number of Lattice Steps: M	Basket Call Value
5	2819.53
10	2818.61
50	2819.39
100	2819.33
500	2818.91
1000	2818.83
5000	2818.76
10000	2818.75

Table 31: Basket Call Value from Binomial Model with Varying M (Selected)

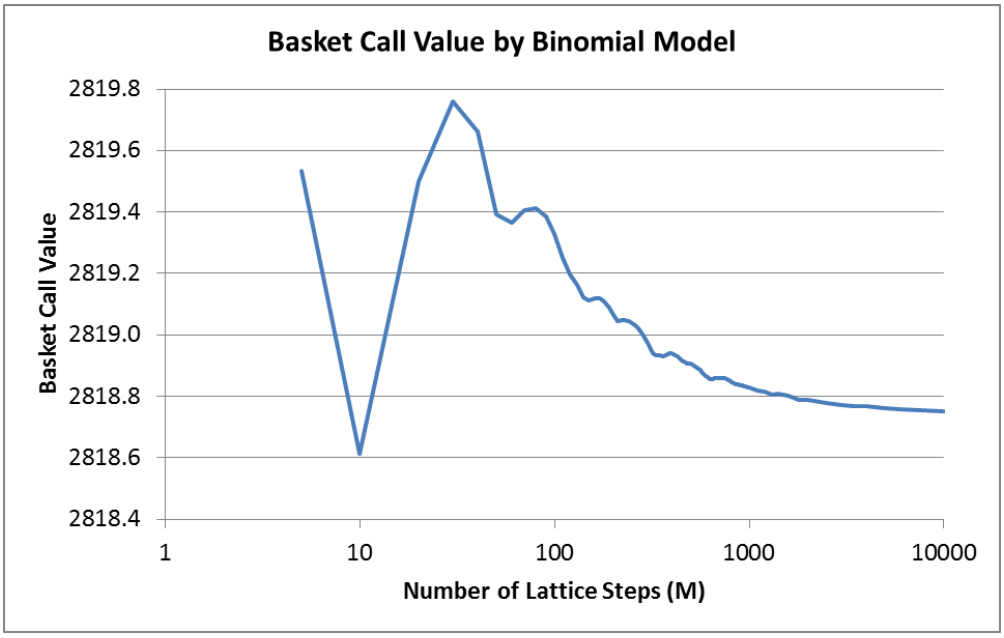


Figure 26: Basket Call Value from Binomial Model with Varying M

From Table 31 and Figure 26, we can conclude that the call value converges as number of lattice steps increases, and a small number of lattice steps (e.g. 5) already gives an estimate with an absolute error of less than 1.

The implementation in MATLAB is available at part4cBinomial.m; Figure 26 and the full Table 31 are available in sheet "Basket Call-Binomial (4.3.2)" of Calculations.xlsm.

4.3.3 Calculate Basket Option Price Using Monte Carlo Simulation

Since the assets in the basket are correlated, we have to generate correlated Gaussian random variables in order to value the option by Monte Carlo simulation.
To do so, we first generate L independent standard normally distributed random variables stored in the vector Z.
Then, we find the covariance matrix, which is given by

$$\Sigma = (\sigma_{i,j}) = \sigma_i \sigma_j \rho_{i,j}$$

Next, the Cholesky decomposition of covariance matrix, $\Sigma = AA^T$, is given by the following algorithm:

Input: $\Sigma = (\sigma_{i,j})$ which must be symmetric and positive definite. Its size is $L \times L$.
$A = (0)_{L \times L}$, i.e. a zero matrix with the same size as covariance matrix.
(Let $a_{i,j}$ denote the element on i-th row and j-th column of A)
For k = 1, 2, …, L **do**
$a_{k,k} = \sqrt{\sigma_{k,k} - \sum_{s=1}^{k-1} a_{k,s}^2}$ (Diagonal entries)
For i = k + 1, k + 2, …, L **do**
$a_{i,k} = \frac{1}{a_{k,k}} \left(\sigma_{i,k} - \sum_{s=1}^{k-1} a_{i,s} a_{k,s}\right)$ (Off-diagonal entries)
End for
End for
Output: A

Finally, we have $X = LZ \sim \mathcal{N}(\mathbf{0}, \Sigma)$.
Therefore, for each run, we have $S_T^{(i)} = S_0^{(i)} \exp\left(\left(r - \frac{1}{2}\sigma_i^2\right)T + \sqrt{T}X\right)$ for i = 1, 2, …, L, and we can find the corresponding basket option payoff. The average of the n payoffs is calculated and discounted to value at time 0 with the discount factor e^{-rT}. This is the estimated basket call value (without antithetic variate).

For variance reduction purpose, the method of antithetic variate is utilized. To evaluate the performance of this method, the same set of random numbers is used. The corresponding antithetic variate is $S_T^{(i)} = S_0^{(i)} \exp\left(\left(r - \frac{1}{2}\sigma_i^2\right)T - \sqrt{T}X\right)$, and we find the corresponding payoff discounted to zero.
Then, for each run, we average out the payoffs calculated with and without antithetic variate. The average of the n averages is calculated and discounted to value at time 0 with the discount factor e^{-rT}. This is the estimated basket call value with antithetic variate.

Implementing the simulation in MATLAB, and varying the number of lattice steps, we obtain Table 32 and Figure 27:

Number of Samples: N	Basket Call Value Without Antithetic Variate	Basket Call Value With Antithetic Variate
100	2658.52	2982.47
500	3029.60	2872.26
1000	2894.74	2800.30
5000	2871.59	2817.01
10000	2880.20	2825.79
50000	2786.82	2806.27
100000	2812.46	2822.41
500000	2820.51	2817.53
1000000	2808.24	2815.59
5000000	2817.74	2817.74
10000000	2820.48	2819.76
50000000	2818.05	2818.40
100000000	2818.12	2818.81
500000000	2818.57	2818.66
1000000000	2818.55	2818.82

Table 32: Basket Call Value from Monte Carlo Simulation with Varying N (Selected)

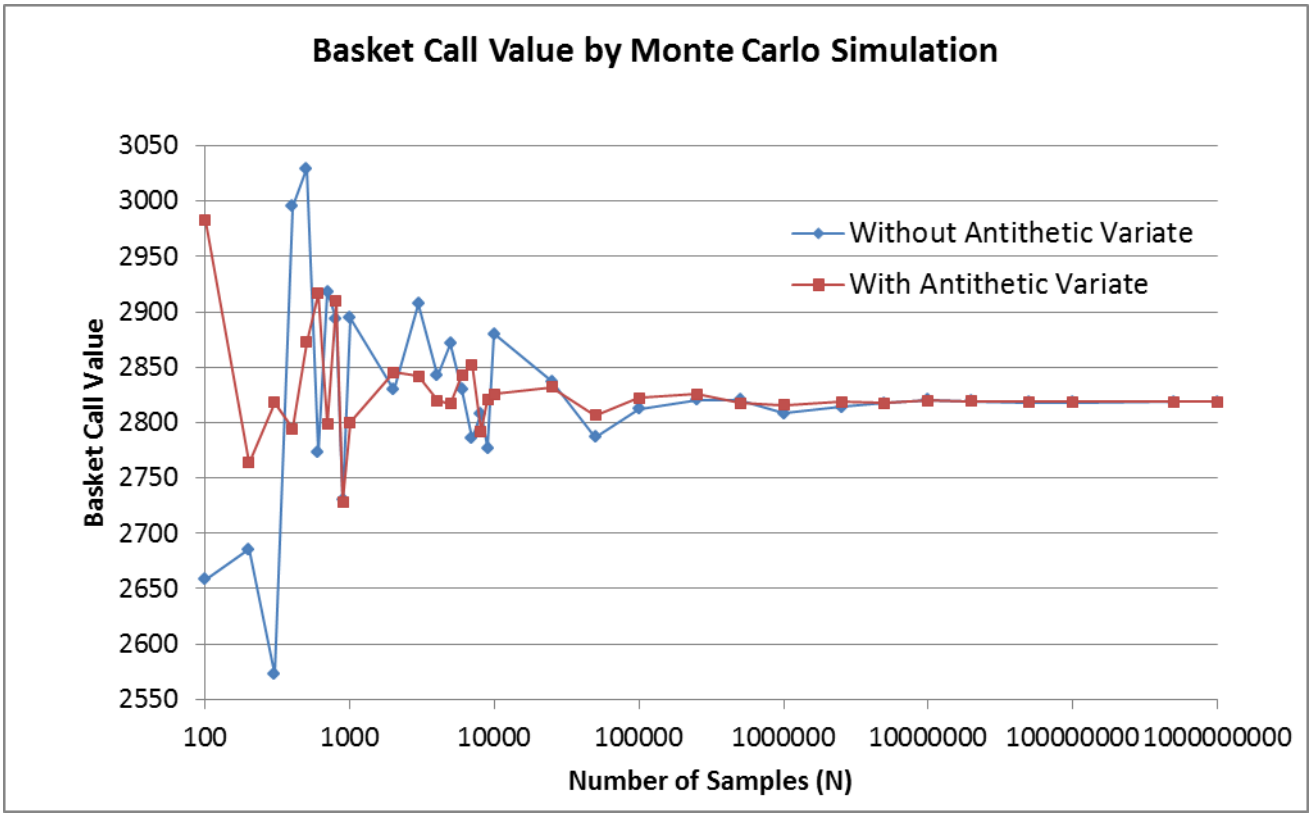


Figure 27: Basket Call Value from Monte Carlo Simulation with Varying N

From Table 32 and Figure 27, we can conclude that the call value converges as number of samples increases, and the variance is smaller (the value converges faster) when antithetic value is used.
Also, we find that even a small number of lattice steps (e.g. 5), the value calculated is already very close to the true value (which is estimated to be 2818.7), but at least 5 million samples are required for the Monte Carlo estimate to have an absolute error of less than 1.

The implementation in MATLAB is available at part4cMonte.m; Figure 27 and the full Table 32 are available in sheet “Basket Call-MonteCarlo (4.3.3)” of Calculations.xlsm.

References

- Yen, Jerome, and Yi Xiang, 2008, Empirical analysis of accumulator, Hong Kong Economic Journal Monthly 378 (September), 20—26. (in Chinese)
- Higham, Desmond, 2004, An Introduction to Financial Option Valuation: Mathematics, Stochastics and Computation (Cambridge University Press).