

## FITE7405 Assignment 3 - Group Project Report

Ghosh Bratin 3035437692 | Law Vanessa 3035051773 | Zhan Juli 3036029973

### A. List of Contributions

Bratin Ghosh: Backend option pricing functions, frontend UI design and coding, test cases

Law Vanessa: Backend option pricing functions, parameter testing, report compositions

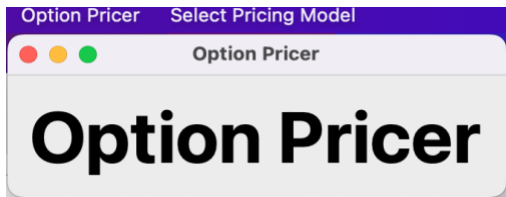
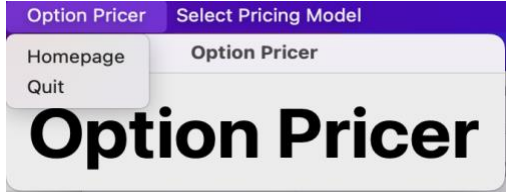
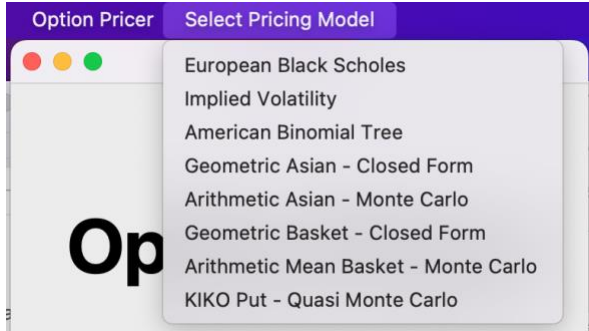
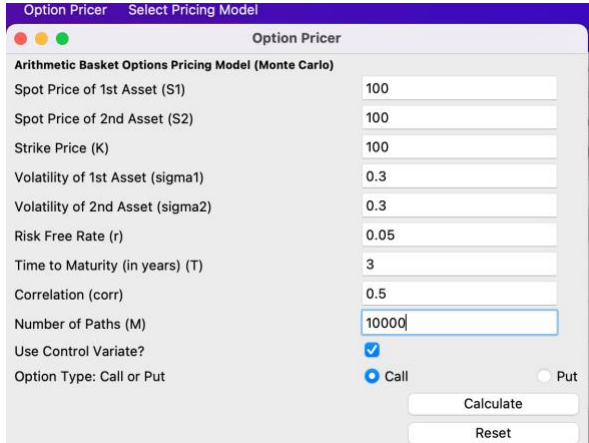
Zhan Juli: Backend option pricing functions, parameter testing, report compositions

### B. Option Pricer User Interface Overview

Install the dependencies: `pip install pandas scipy tk`

Run the application: `python OptionPricerApplication.py`

Interfaces and explanations

Explanations	Interfaces
Once the OptionPricerApplication is launched, there will be a pop-up window with two menu: <ol style="list-style-type: none"> <li>Option Pricer</li> <li>Select Pricing Model</li> </ol>	
To access the first menu, hover over "Option Pricer", under which the options will be: <ol style="list-style-type: none"> <li>Homepage</li> <li>Quit -&gt; to quit the program</li> </ol>	
For the "Select Pricing Model" menu, there are 8 functions you can choose from: <ol style="list-style-type: none"> <li>European Black Scholes</li> <li>Implied Volatility</li> <li>American Binomial Tree</li> <li>Geometric Asian - Closed Form</li> <li>Arithmetic Asian - Monte Carlo</li> <li>Geometric Basket - Closed Form</li> <li>Arithmetic Mean Basket - Monte Carlo</li> <li>KIKO Put - Quasi Monte Carlo</li> </ol>	
As an example, if you select "Arithmetic Mean Basket", the app will bring you to the window where you will need to input parameters such as S1, S2, K, sigma1, sigma2, r, T, rho, M and whether to use control variate or not, and specifying the options type (Call or Put)	

To get the price, click the 'Calculate' button, then the pricer will then print the option price in the box at the bottom, alongside the 95% confidence interval

Hitting "Reset" will clear the inputs and history messages.

The screenshot shows a software window titled "Option Pricer" with a subtitle "Select Pricing Model". Below the title bar, there are three colored circles (red, yellow, green) and the text "Option Pricer". The main area is titled "Arithmetic Basket Options Pricing Model (Monte Carlo)". It contains several input fields with values: Spot Price of 1st Asset (S1) is 100, Spot Price of 2nd Asset (S2) is 100, Strike Price (K) is 100, Volatility of 1st Asset (sigma1) is 0.3, Volatility of 2nd Asset (sigma2) is 0.3, Risk Free Rate (r) is 0.05, Time to Maturity (in years) (T) is 3, Correlation (corr) is 0.5, Number of Paths (M) is 10000, Use Control Variate? is checked, and Option Type is set to Call. At the bottom right, there are "Calculate" and "Reset" buttons. At the bottom left, there is a status bar showing: [SUCCESS] Price: 24.542701796021984 and [SUCCESS] 95% Confidence Interval: [24.443462945716437, 24.64194064632753].

## C. Class & Function Descriptions

### a. American Option

This class is the pricer of American put / call options for single stocks using the Binomial Tree method. Input parameters include Option type (cp\_flag), Spot (S), Strike (K), Maturity date (T), volatility (sigma), risk-free interest rate (r), number of time steps (N). We first compute the risk neutral probability (denoted by  $p$  in the formula), in turn to construct the binomial tree for stock prices, which is then substituted into the corresponding option payoff formula to calculate the respective option prices, and eventually work backward to the first node to find out the option price.

### b. ArithmeticAsianOption

- i. `get_price_closed_form`: This gives the estimate of geometric Asian option price by the closed-form formula. This value is used as the control variate for the arithmetic Asian option pricing later in (ii). For details, please see the explanation for the class `GeometricAsianOption`.
- ii. `get_price_monte_carlo`: This function is the pricer of arithmetic Asian put / call options for single stocks using Monte Carlo simulation. Input parameters include: Option type (cp\_flag), Spot (S), Strike (K), Maturity date (T), pricing date (t), volatility (sigma), risk-free interest rate (r), number of observations (n, pricer will assume  $n=100$  if not specified by the user), number of samples to be generated (M, pricer will assume  $M=10000$  unless specified), and whether to turn on or off the control variate technique.

sPath denotes the path constructing formula, assuming stock follows a stochastic process of Geometric Brownian motion, output is in the shape of (M, n). For each of the stock paths generated, we estimate S by taking the arithmetic mean of stock prices, and hence achieving the respective present value (PV) of arithmetic option payoff.

If control variate is turned off, the function will simply output the average of these M samples of PV of option payoff as the option price and print the range bounds within the 95% confidence level.

If control variate is turned on, the function will first compute the geometric mean of stock prices and the PV of average geometric option payoff for each of the stock paths generated. Adding the adjustment term (theta times the difference between PV of geometric option payoff and the geometric option priced from the closed-form formula) to each of the arithmetic option payoff sampled, the formula then output the sample average of the option payoffs as option price and the range bounds of 95% confidence-level.

### c. ArithmeticBasketOption

The code defines a class `ArithmeticBasketOption` that represents an arithmetic basket option. The code takes in several parameters including the risk-free interest rate ( $r$ ), volatility of the first asset ( $\sigma_1$ ), volatility of the second asset ( $\sigma_2$ ), time to expiration ( $T$ ), initial price of the first asset ( $S_1$ ), initial price of the second asset ( $S_2$ ), strike price ( $K$ ), correlation between the two assets ( $\rho$ ), number of simulations ( $M$ ), option type (`option_type`), and whether to use a control variate (`control_variate`).

The `stock_path_simulations()` function generates Monte Carlo simulations of the stock prices of the two assets using the Black-Scholes model. The `basket_options_payoffs()` function calculates the payoffs for various types of basket options. The `get_price_closed_form()` function calculates the price of the geometric basket option using the closed-form formula. The `geometric_basket()` function calculates the price of the geometric basket option using Monte Carlo simulations. The `get_price_monte_carlo()` function calculates the price of the arithmetic basket option using Monte Carlo simulations.

If the `control_variate` flag is set to `True`, the function uses geometric closed form as control variate to reduce the variance of the Monte Carlo estimate. The function returns the estimated price of the option and a confidence interval.

#### d. `BlackScholesOption`

This class is the plain vanilla put / call option pricer using the classic Black Scholes Merton model for single stocks. Input parameters include Option type (`cp_flag`), Spot ( $S$ ), Strike ( $K$ ), Maturity date ( $T$ ), pricing date ( $t$ ), volatility ( $\sigma$ ), risk-free interest rate ( $r$ ), dividend yield/repo rate ( $q$ ).

#### e. `GeometricAsianOption`

This class is the pricer for geometric Asian put / call options for single stocks using the classic closed form formula. Input parameters include Option type (`cp_flag`), Spot ( $S$ ), Strike ( $K$ ), Maturity date ( $T$ ), pricing date ( $t$ ), volatility ( $\sigma$ ), risk-free interest rate ( $r$ ), number of observations ( $n$ , pricer will assume  $n=100$  if not specified by the user). The pricing function computes the expected values of  $S$  and  $\sigma$  at time  $t$  assuming  $S$  is lognormally distributed, and substituting these new  $S$  and  $\sigma$  estimates into the classic Black Scholes Merton formula to output the option price.

#### f. `GeometricBasketOption`

This class is the pricer for geometric Asian put / call options for a basket of stocks using the classic closed form formula. Input parameters include Option type (`cp_flag`), Spot ( $S$ ), Strike ( $K$ ), Maturity date ( $T$ ), pricing date ( $t$ ), volatility ( $\sigma$ ), risk-free interest rate ( $r$ ), correlation ( $\text{corr}$ ). First, the spot prices and sigmas are converted into an array and taken the geometric average. Then a correlation matrix between the stocks is constructed to estimate the resulting basket sigma. The option price is achieved by substituting the geometric averages of the spot prices in the basket and the basket sigma into the classic Black Scholes formula.

#### g. `ImpliedVolatility`

The code contains a single function, `get_implied_volatility()`, which calculates the implied volatility of an option using the Newton-Raphson method. The function takes in several parameters including the current stock price ( $S$ ), risk-free interest rate ( $r$ ), dividend yield ( $q$ ), time to expiration ( $T$ ), strike price ( $K$ ), option price ( $V$ ), and option type (`option_type`).

The `d1()` and `d2()` functions are helper functions that calculate intermediate values used in the Black-Scholes formula for option pricing. The `bs_price()` function calculates the theoretical option price using the Black-Scholes formula, and the `bs_vega()` function calculates the option's sensitivity to changes in volatility.

The main functionality of the code is to calculate the implied volatility using the Newton-Raphson method. This is done by adjusting an initial guess value for volatility until the difference between the calculated option price and the actual price is within a certain tolerance. The maximum number of iterations is set to 100 to ensure that the function doesn't run indefinitely. The function returns the calculated implied volatility or `None` if it fails to converge within the specified tolerance.

#### h. KikoOption

This class is the pricer for KIKO put options for single stocks using the Quasi-Monte Carlo Simulation. Input parameters include Option type (type, must be 'put' in this case), Spot (S), Strike (K), Maturity date (T), volatility (sigma), discount rate (r), Lower Bound (L), Upper Bound (U), number of observations (n), rebate (R) and number of samples to be generated (M, pricer assumed M=100). This class consists of two functions:

- generate\_simulated\_stock\_prices:** This function generates samples of paths of stock prices based on the quasi-random sampling sequences and stores the stock paths in the shape of (M, n).
- Get\_price\_quasi\_monte\_carlo:** Then for each of the stock price paths generated, the function will check whether at any point of time the stock price is: (1) Above the Upper Bound: Firstly, if the stock price is above the upper bound, the function will find out the knock out time, and calculate the PV of the rebate to be received and include that as the option value. (2) Below the Lower Bound: if the stock price went below the lower bound, the option value will be updated as the PV of the payoff, i.e. K minus stock price at maturity. Hence, the option price is the average of PVs from the samples. The range bounds of the 95% confidence level are also given.

#### i. OptionPricerApplication

This class contains the codes for the graphical user interface (GUI) of the option pricer. The front end is built using the Tkinter package.

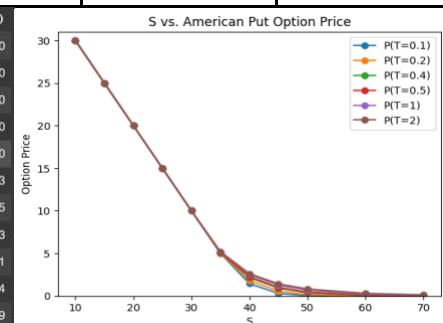
### D. Test Cases and Parameters Analysis

#### a. American Options

Please refer to the AmericanOption.csv for some sample test results

Relationship	S	sigma	r	T	K
Call Option Price	Positive	Positive	Positive	Positive	Inverse
Put Option Price	Inverse	Positive	Inverse	Graph below	Positive

S	Sigma	R	K	N	OptionType	P(T=0.1)	P(T=0.2)	P(T=0.4)	P(T=0.5)	P(T=1)	P(T=2)	P(T=3)
0	10	0.4	40	200	put	30.000000	30.000000	30.000000	30.000000	30.000000	30.000000	30.000000
1	15	0.4	40	200	put	25.000000	25.000000	25.000000	25.000000	25.000000	25.000000	25.000000
2	20	0.4	40	200	put	20.000000	20.000000	20.000000	20.000000	20.000000	20.000000	20.000000
3	25	0.4	40	200	put	15.000000	15.000000	15.000000	15.000000	15.000000	15.000000	15.000000
4	30	0.4	40	200	put	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000
5	35	0.4	40	200	put	5.000000	5.004488	5.058964	5.081311	5.148300	5.200341	5.211793
6	40	0.4	40	200	put	1.453324	1.809854	2.154794	2.254850	2.499123	2.611809	2.643285
7	45	0.4	40	200	put	0.288840	0.572204	0.903108	1.005755	1.274638	1.419389	1.457443
8	50	0.4	40	200	put	0.039877	0.158515	0.369737	0.449217	0.675271	0.816955	0.849251
9	60	0.4	40	200	put	0.000323	0.008933	0.058941	0.088101	0.205471	0.302498	0.331724
10	70	0.4	40	200	put	0.000001	0.000391	0.009060	0.017307	0.067770	0.126900	0.147919

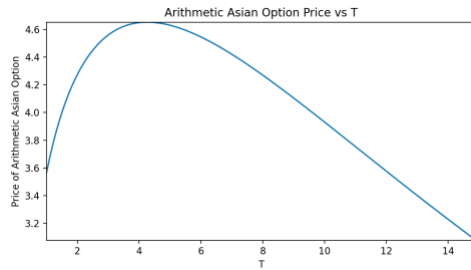


Also: Number of Observations (N): No definite relationship on the price but increases the run time

#### b. Asian Options

Please refer to the AsianOption.csv for some sample test results of the arithmetic Asian options

Relationship	S	K	T	Sigma	r
Call Option Price	Positive	Inverse	Positive	Positive	Positive
Put Option Price	Inverse	Positive	Note	Positive	Inverse

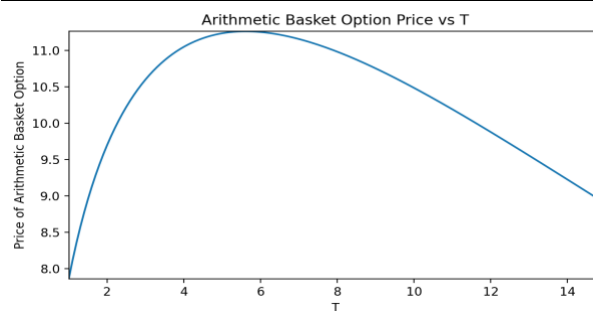


^Note: Arithmetic Asian put when  $S=100$ ,  $K=100$ ,  $r=0.05$ ,  $\sigma=0.2$ ,  $n=10$

c. Basket Options

Please refer to the BasketOption.csv for some sample test results of the arithmetic basket options

Relationship	S	K	T	Sigma	r	rho
Call Option Price	Positive	Inverse	Positive	Positive	Positive	Positive
Put Option Price	Inverse	Positive	Note	Positive	Inverse	Positive



^Note: Arithmetic Put when  $S1=100$ ,  $S2=100$ ,  $K=100$ ,  $r=0.05$ ,  $\sigma_1=0.3$ ,  $\sigma_2=0.3$ ,  $\rho=0.5$

d. Black Scholes Options

Please refer to the BlackScholesOptions.csv for some sample test results.

Relationship	S	K	T	Sigma	r	q
Call Option Price	Positive	Inverse	Positive	Positive	Positive	Inverse
Put Option Price	Inverse	Positive	Note	Positive	Inverse	Positive

Note: The put option value can decrease or increase when getting closer to maturity, depending on the spot level

e. KIKO Put Options

Please refer to the KIKOPutOption.csv for some sample test results

Relationship	S	sigma	r	K
Option Price	Inverse	Positive	Inverse	Positive

	T	Lower Bound	Upper Bound
Option Price	Option price may decrease or increase closer to maturity depending on spot level	Inversely related when Lower Bound is smaller than Strike; When Lower Bound is higher than Strike, no more impact on option price	Positive

	n	Rebate	M
Option Price	No definite relationship	Positively related	No definite relationship