Advanced Mathematical Approaches for Pricing Asian Options

By

Rishik Kumar

Supervised by Dr Larbi Alili

Submitted to the University of Warwick for the degree of MSc Mathematical Finance

September 2024

Abstract

This thesis investigates the pricing of Asian options, a class of path-dependent derivatives whose payoff depends on the average price of the underlying asset. Due to the complexity of their path dependency, a traditional closed form expression does not exist, leading to significant mathematical and computational challenges. This study explores various approaches, starting with Numerical methods, before moving onto superior alternatives using the Laplace Transform and moment matching approaches, offering greater accuracy and computational efficiency. The results of this study has potential to reshape traditional quantitative finance strategies that involve standard numerical methods.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor, Dr. Larbi Alili, for his invaluable time and support throughout the development of this thesis. I am especially grateful to him for suggesting this research topic, which I found both engaging and rewarding to explore. His expertise and insightful feedback were crucial to the successful completion of this thesis.

Contents

1 Introduction

Asian options are a type of path-dependent derivatives whose popularity is growing due to their use in hedging within the commodity market. These options are particularly useful for companies that depend heavily on commodities, such as airlines that require large quantities of fuel. By using Asian options to hedge against price fluctuations, these companies can secure more stable pricing and reduce their exposure to sudden market shocks or price spikes that are prone in commodity markets. There are two types of averaging for Asian options, arithmetic and geometric. The geometric average has a closed form expression, which I derive in this paper, and therefore does not require any additional study. The arithmetic average however, despite being more frequently traded, does not have a traditional closed form expression and therefore will be the primary focus of this study.

The first section of this paper looks at pricing an Asian option through numerical methods. Starting with a standard Monte Carlo scheme before moving onto multiple variance reduction methods. In particular, looking at how a geometric average can be used as a control variate to greatly increase the accuracy of pricing. Then, looking at how an antithetic approach can be used for an Asian option.

The second section looks at pricing an Asian option using the Laplace Transform. Making use of both Bessel processes and Laplace Transform properties, we obtain the derivation of the pricing formula. Then comparing its results to the Monte Carlo method, we find that the Laplace Transform method offers a superior alternative.

The third section looks at pricing an Asian option by approximating its distribution. The fundamental idea of this section is match the mean and variance to a Log-Normal distribution, therefore obtaining an approximate closed form expression, greatly reducing the computational time. Then comparing its results with the those of the previous techniques, focusing on the trade off between speed and accuracy. We find that this approach could potentially be preferred for options with small parameter values.

The fourth sections looks at pricing an Asian option through Martingale and PDE approach. I derive the standard partial differential equation (PDE) for an Asian option, before looking at alternative substitutions to simply the PDE. In particular, the use of Martingales to derive Rogers and Shi's PDE. Then looking at numerical methods to solve this PDE without running into oscillatory errors and numerical diffusion.

The final two sections, after the conclusion, contain two appendices. The first appendix contains key fundamental mathematics used throughout this paper while the second appendix contains all the Python code used.

2 Numerical Methods

The Monte-Carlo method was first introduced by Boyle [1] and is by far the most widely used method to price an Asian option due to it's flexibility and easy implementation. That being said, the simulation is computationally expensive to obtain accurate results, therefore specific variance reduction methods are needed to control this. In this section, I will examine two key variance reduction methods: control variates and antithetic variates. These techniques are not only applicable to Asian options but also to other path-dependent derivatives, therefore lookback and barrier options can also be priced using this method.

2.1 Naive Monte Carlo

Naive Monte Carlo is the standard approach to pricing options using simulation. The technique involves simulating a large number of price paths, of the underlying asset S, over a specified time period T. These simulated paths are designed to mimic and capture a range of possible scenarios the asset could take. For each simulated path, the average price is calculated and the options payoff can be determined. These payoffs are then averaged across all paths to estimate the fair value of the option at time T. However, in order to satisfy the risk neutral condition, that all assets are expected to grow at the risk free rate, we discount the expected future price at the risk free rate r.

Recall from **Theorem A7.6** that we assume the underlying stock price follows Geometric Brownian Motion:

$$\mathrm{d}S_t = \mu S_t \,\mathrm{d}t + \sigma S_t dW_t$$

and under a risk neutral measure:

$$\mathrm{d}S_t = rS_t\,\mathrm{d}t + \sigma S_t dW_t \tag{1}$$

In order to solve this Stochastic Differential Equation (SDE), we can use properties of the stochastic exponential $\mathcal{E}(X)$, shown in **Theorem A7.4**.

$$\begin{aligned} S_t &= S_0 \times \exp\left\{ rt + \sigma W_t - 0 - \frac{1}{2}\langle rt + \sigma W_t \rangle \right\} \\ &= S_0 \times \exp\left\{ \left(r - \frac{1}{2}\sigma^2 \right) t + \sigma W_t \right\} \end{aligned}$$

Which leads to the recursive relation:

$$S_{t+\Delta t} = S_t \times \exp\left\{ \left(r - \frac{1}{2}\sigma^2 \right) t + \sigma\sqrt{\Delta t}\, Z \right\} \tag{2}$$

where $Z \sim N(0,1)$ is a standard Normal Distribution and Δt is the increment size $= T/n_{steps}$. Note here that the use of Δt is based on the Euler time-stepping method, which is a standard approach to discretise SDE's that model underlying asset prices (Glasserman, 2004) [2].

Using this recursive relation, we can simulate skeleton paths of the stock price up to S_T and calculate the path payoff using the following formula:

$$(\frac{1}{N}\sum_{i=1}^{N} S_i - K)^+$$

Repeating these steps several times will generate a large number of payoffs. Averaging these payoffs and discounting accordingly, as mentioned before, we can obtain the present fair value of the derivative using:

$$e^{-r(T-t)}E[(\frac{1}{N}\sum_{i=1}^{N} S_i - K)^+]$$

The Monte Carlo estimator can be considered as the sample average of the payoffs, denoted as x_i, formally expressed as:

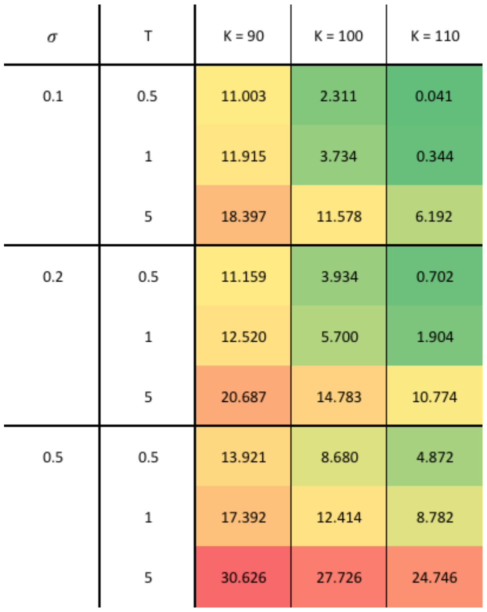
$$\hat{M} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

Its quite easy to see that the Monte Carlo estimator $\hat{M}$ is unbiased, that is $E[\hat{M}] \to M$, as $n \to \infty$. However, we are more interested in the variance, which determines the accuracy of the estimate.

$$Var(\hat{M}) = Var(\frac{1}{N} \sum_{i=1}^{n} x_i) = \frac{Var(x)}{N}$$

This will be used as a benchmark when comparing other methods.

2.1.1 Results



σ	T	K = 90	K = 100	K = 110
0.1	0.5	11.003	2.311	0.041
	1	11.915	3.734	0.344
	5	18.397	11.578	6.192
0.2	0.5	11.159	3.934	0.702
	1	12.520	5.700	1.904
	5	20.687	14.783	10.774
0.5	0.5	13.921	8.680	4.872
	1	17.392	12.414	8.782
	5	30.626	27.726	24.746

Figure 1: *Asian Call Option Prices with* $S_0 = 100$ *and* $r = 0.05$

This table illustrates the price of Asian options across different levels of volatility σ, time to maturity T and strike prices K. Straight away, it's clear to see that option prices increase as both volatility and time-to maturity increase, which is consistent with standard option characteristics. Specifically, the combined effect, as the volatility increases to $\sigma = 0.5$, the option prices become more sensitive to increases in time to maturity T. Options in-the-money clearly have a higher price than those out-the-money reflecting the intrinsic value component in addition to the time value.

With a Monte-Carlo simulation, we are primarily concerned with the computation time to achieve accurate results.

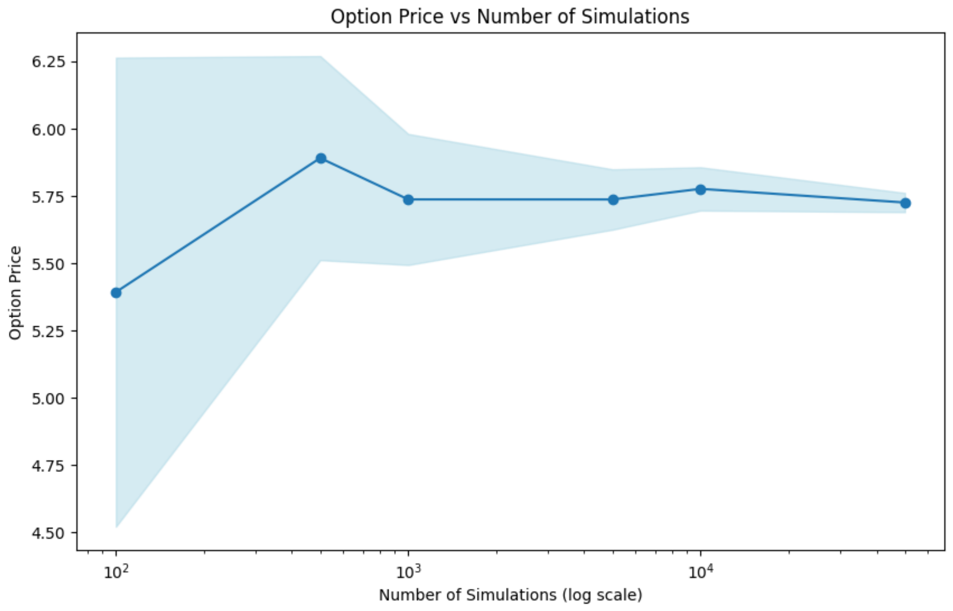


Figure 2: *Convergence Plot with Increasing Number of Simulations N, with $S_0 = 100,\ K = 100,\ T = 1,\ r = 0.05,\ \sigma = 0.2$ and $q = 0$*

This plot demonstrates how the Monte Carlo simulation converges to the 'correct' price as the number of simulations N increases. Where the shaded region represents the standard error around the estimated price.

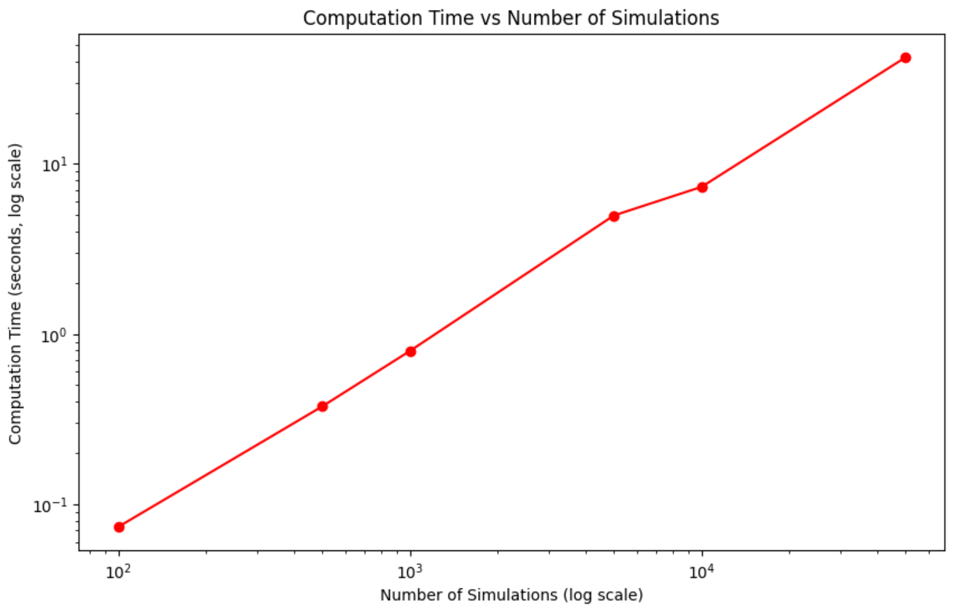


Figure 3: Complexity Plot

This second plot demonstrates how computation time increases (on a Log-scale) with the number of simulations. It is a common result that a Monte Carlo simulation has complexity $\mathcal{O}(N)$, shown by the linear relationship on this log-log plot. Specifically, if you increase the number of simulations N by a factor of 10, the computational time should increase by the same factor of 10.

These two plots combined, clearly demonstrate that there is a clear trade-off between accuracy and computational time with the Monte Carlo approach.

2.2 Monte-Carlo with Geometric Control Variate

Control variates is a variance reduction technique used to improve the accuracy of pricing Asian options when using Monte Carlo simulation. The method leverages on a positively correlated auxiliary variable whose true value is known. The geometric Asian option as been shown by many authors to be a very effective control variate due to its high correlation with the arithmetic Asian option and its true value is known, which I later derive in Section 2.5 to be:

$$e^{-rT}[S_0 e^{r-\frac{1}{2}\sigma^2\frac{T}{2}+\sigma^2\frac{T}{6}}\Phi(d_1) - K\Phi(d_2)] \tag{3}$$

where

$$d_1 = \frac{\ln|\frac{S_0}{K}| + (r-\frac{1}{2}\sigma^2)\frac{T}{2} + \sigma^2\frac{T}{3}}{\sigma\sqrt{\frac{T}{3}}}$$

$$d_2 = d_1 - \sigma\sqrt{\frac{T}{3}}$$

2.2.1 Variance Calculation

The general Control Variate formula is:

$$X^{CV} = \hat{X} - c(\hat{Y} - Y^*)$$

where $\hat{X}$ and $\hat{Y}$ are Monte Carlo estimates, and $\hat{Y} \to Y^*$
As Y^* is the true value of Y, the second term $\to 0$ therefore not affecting the final result.

$$Var(X^{CV}) = Var(\hat{X}) + c^2 Var(\hat{Y}) - 2cCov(\hat{X}, \hat{Y})$$

where $Var(Y^*) = 0$

We want to choose c to minimise the variance, by taking the derivative of $Var(X^{CV})$ with respect to c and set $= 0$:

$$\frac{d}{dc}Var(X^{CV}) = 2cVar(\hat{Y}) - 2Cov(\hat{X}, \hat{Y}) = 0$$

solving for c:

$$c = \frac{Cov(\hat{X}, \hat{Y})}{Var(\hat{Y})}$$

Now substituting this optimal c into the variance formula:

$$Var(X^{CV}) = Var(\hat{X}) + \left(\frac{Cov(\hat{X}, \hat{Y})}{Var(\hat{Y})}\right)^2 Var(\hat{Y}) - 2\left(\frac{Cov(\hat{X}, \hat{Y})}{Var(\hat{Y})}\right) Cov(\hat{X}, \hat{Y})$$

$$= Var(\hat{X}) + \left(\frac{Cov(\hat{X}, \hat{Y})^2}{Var(\hat{Y})}\right) - 2\left(\frac{Cov(\hat{X}, \hat{Y})^2}{Var(\hat{Y})}\right)$$

$$= Var(\hat{X}) - \frac{Cov(\hat{X}, \hat{Y})^2}{Var(\hat{Y})}$$

This clearly shows that the stronger X is correlated with Y, the more significant the variance reduction. More importantly noting that:

$$Var(X^{CV}) \leq Var(\hat{X})$$

where $Var(\hat{X})$ is the same variance as the naive Monte-Carlo estimator. Therefore, using the geometric Asian option as our control variate, we have;

$$A^{CV} = \hat{A} - c(\hat{G} - G^*)$$

where $\hat{A}$ and $\hat{G}$ are Monte-Carlo estimates of Arithmetic and Geometric Asian options respectively and G^* is the true value of $\hat{G}$ using equation 3.

2.2.2 Results

The benefit of using a control variate is to significantly reduce the number of simulations required to obtain accurate results.

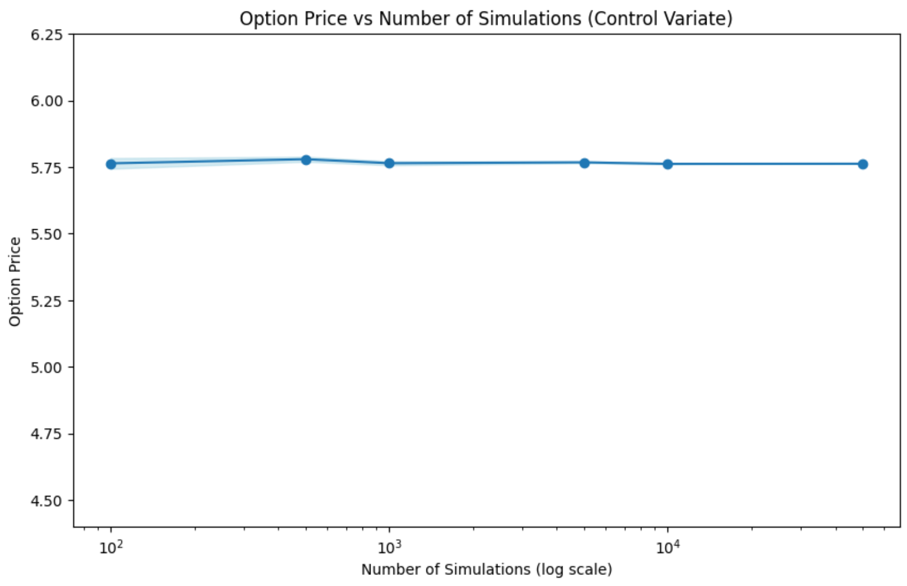


Figure 4: *Control Variate Convergence Plot with Increasing Number of Simulations with* $S_0 = 100,\ K = 100,\ T = 1,\ r = 0.05,\ \sigma = 0.2$ *and* $q = 0$

Plotting on the same axis limits as before, we can clearly see that using the geometric control variate, the prices converges to the 'correct' price significantly faster than the naive Monte Carlo scheme. We also notice that the shaded region, representing the standard error, is also significantly smaller.

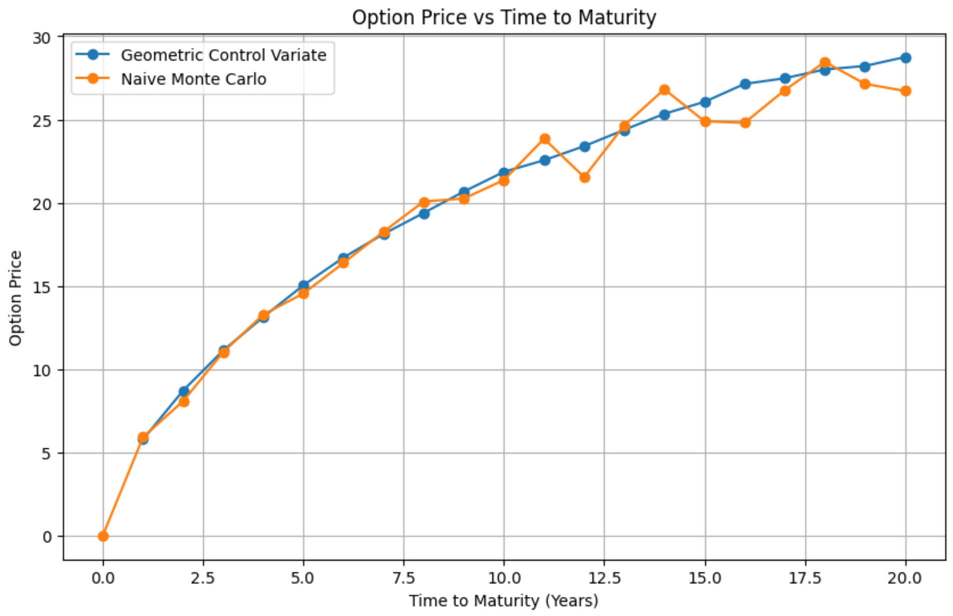


Figure 5: *Monte Carlo and Control Variate Estimates of Asian Option Prices with* $S_0 = 100,\ K = 100,\ r = 0.05,\ \sigma = 0.2$ *and* $q = 0$

This second plot clearly shows how the Monte Carlo results deviate above and below the control variate results. This happens because the Monte Carlo method has a higher variance and further supports how the reduced variance of the control variate approach gives more consistent and accurate prices. Specifically, as T increases, the longer time period allows for more potential outcomes and therefore, more variability in the prices.

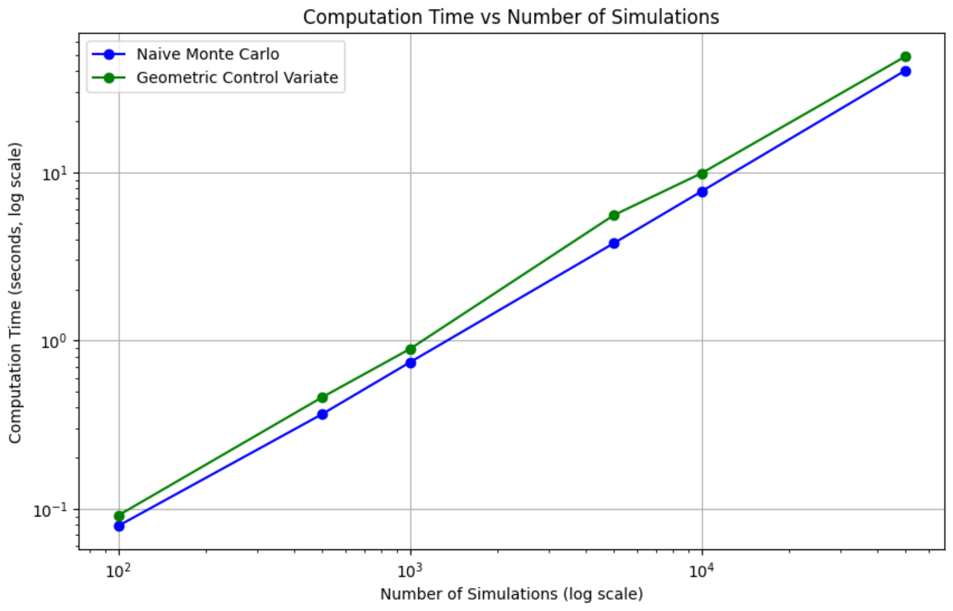


Figure 6: *Complexity Plot of Monte Carlo and Control Variate*

This next plot demonstrates that the control variate scheme also follows complexity $\mathcal{O}(N)$. We can see that when using the control variate approach, each simulation takes slightly longer. However this extra computational time is negligible in practice, as the number of simulations required is significantly lower, as shown by figure 4.

This supports that the control variate methodology is significantly superior to the naive Monte Carlo.

2.3 Monte-Carlo with Antithetic Variance Reduction

Antithetic variance reduction is a common technique that utilizes pairs of negatively correlated variables to reduce variance in simulations. In order to generate a pair of negatively correlated variables we can use the symmetric property of the Normal distribution. Instead of just simulating a path using a positive standard Normal variable Z in equation 2, the antithetic method simultaneously simulates another path using a negative standard Normal $-Z$.

i.e.

$$S_{t+\Delta t} = S_t \times \exp\left\{\left(r - \frac{1}{2}\sigma^2\right)\Delta t + \sigma\sqrt{\Delta t}\,Z\right\}$$

and

$$S^{AV}_{t+\Delta t} = S_t \times \exp\left\{\left(r - \frac{1}{2}\sigma^2\right)\Delta t + \sigma\sqrt{\Delta t}\,(-Z)\right\}$$

Due to the symmetric property of the normal distribution, where Z and $-Z$ are identically distributed, the expected value of the simulation remains unchanged. However, by using these negatively correlated variables, the variance of the estimate is reduced, leading to more precise and reliable results.

2.3.1 Variance Calculation

The Antithetic estimator is

$$\frac{1}{2N}\left(\sum_{i=1}^{N}(X_{1,i} + X_{2,i})\right)$$

where $X_{1,i}$ and $X_{2,i}$ are the payoffs from paths generated using Z and $-Z$, respectively, and N is the number of simulations.

Then the variance can be expressed as:

$$Var(X^{AV}) = Var\left(\frac{1}{2N}\sum_{i=1}^{n}(X_{1,i}+X_{2,i})\right)$$

$$= \frac{1}{4N^2}\sum_{i=1}^{n} Var\left(X_{1,i}+X_{2,i}\right)$$

$$= \frac{1}{4N}\left(Var(X_{1,i}) + Var(X_{2,i}) + 2Cov(X_1, X_2)\right)$$

Notice that X_1 and X_2 have the same distribution due to the symmetric property of the Normal Distribution, therefore:

$$Var(X^{AV}) = \frac{1}{4N}\left(2Var(X) + 2Cov(X_1, X_2)\right)$$

$$= \frac{1}{2N}(Var(X) + Cov(X_1, X_2))$$

As we constructed X_1 and X_2 to be negatively correlated, the second term will always be negative.

Therefore:

$$Var(X^{AV}) \leq \frac{1}{2N}Var(X) \leq \frac{Var(X)}{N}$$

where $\frac{Var(X)}{N}$ is the same variance as the naive Monte-Carlo estimator.

2.3.2 Results

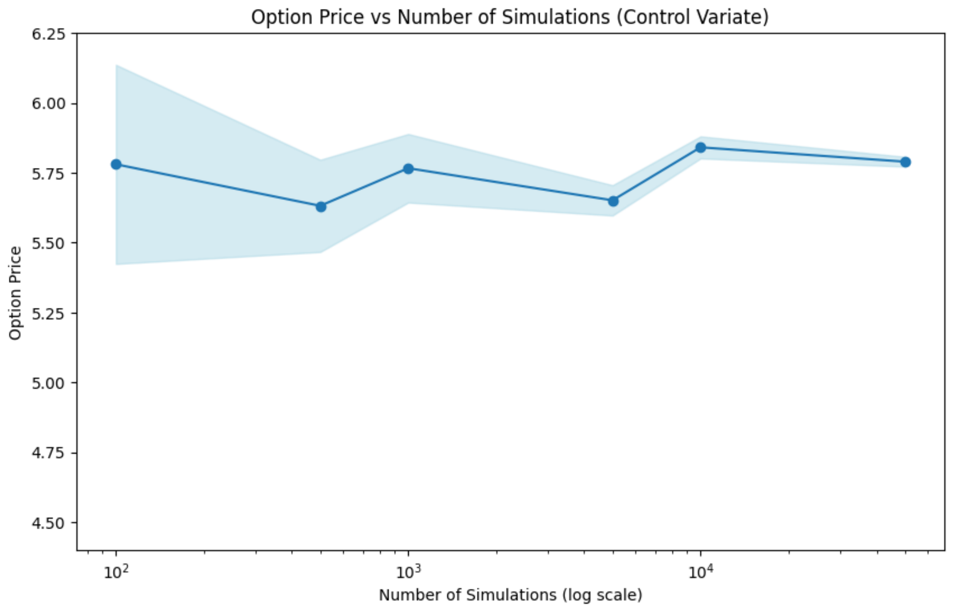


Figure 7: *Convergence Plot using Antithetic*

Whilst this plot does show an improvement over the naive Monte Carlo approach, the prices do not converge fast enough when compared to the control variate approach. Therefore, the control variate approach remains preferable due to its superior efficiency and faster convergence in providing accurate option price estimates.

2.4 Monte-Carlo with Dividends

Recall the dynamics of the underlying asset S_t under a risk-neutral measure from 1:

$$dS_t = rS_t dt + \sigma S_t dW_t$$

When we introduce dividends, the dynamics change to:

$$dS_t = (r - q)S_t dt + \sigma S_t dW_t$$

$q =$ dividend yield

The term $(r - q)$ reflects the expected return on the stock reduced by the dividend yield. This is because a dividend payment reduces the value of the company by the amount of the dividend, as cash (or other assets) are leaving the company and going to shareholders.

To solve this SDE we can use the stochastic exponential $\mathcal{E}(X)$ again.

In a similar way to before:

$$dS_t = S_t[\underbrace{(r - q)dt + \sigma dW_t}_{dX_t}]$$

$$\begin{aligned}\mathcal{E}(X) &= exp\left\{(r - q)t + \sigma W_t - 0 - \frac{1}{2}\sigma^2 t\right\} \\ &= exp\left\{[(r - q) - \frac{1}{2}\sigma^2]t + \sigma W_t]\right\}\end{aligned}$$

Therefore:

$$S_t = S_0 \times exp\left\{[(r - q) - \frac{1}{2}\sigma^2]t + \sigma W_t\right\}$$

In a similar way we can construct the recursive relation:

$$S_{t+\Delta t} = S_t \times exp\left\{[(r - q) - \frac{1}{2}\sigma^2]t + \sigma\sqrt{\Delta t}Z\right\}$$

Using this new recursive relation combined with Euler time-stepping, we can simulate new skeleton paths which take into account dividend yield. Calculating the payoff of each path as before, then averaging and discounting, we can obtain the present value of the derivative, incorporating dividends.

2.4.1 Results

The main aim of this section is to show how prices of the Asian option are affected by dividends. Particularly, across different values of volatility σ and time to maturity T.

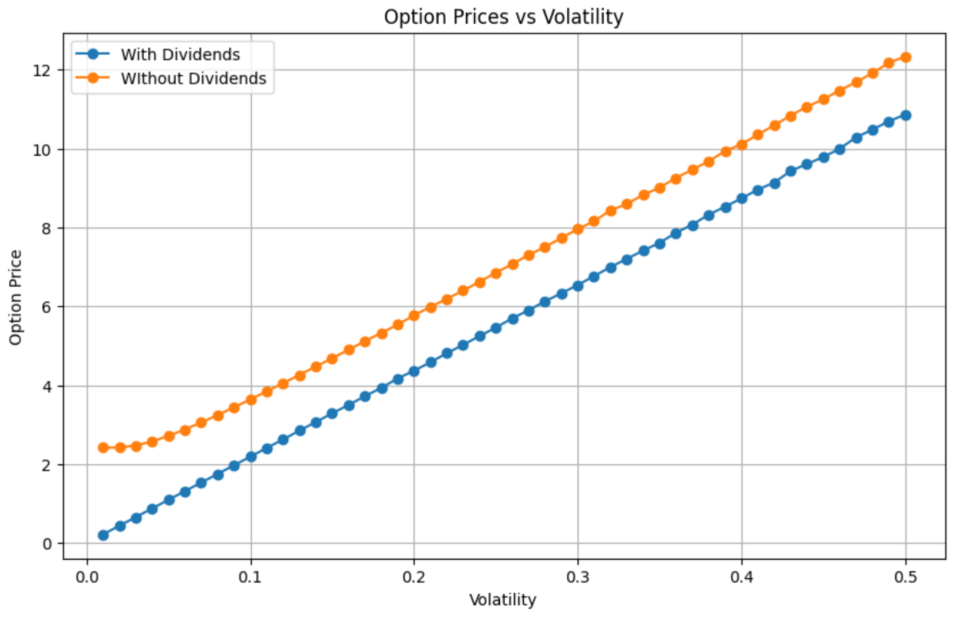


Figure 8: *Estimates of Asian Option Price With and Without Dividends, with* $S_0 = 100,\ K = 100,\ T = 1$ *and* $r = 0.05$

This plot shows a consistent gap between the option prices, clearly illustrating that the presence of dividends reduces the option price by a constant amount, against volatility. This is because the new drift term $(r - q)$, representing the expected return on the stock, is expected to drop by the dividend amount on the ex-dividend date. As dividends only affects the expected return, there is no direct impact on volatility, hence a constant gap. However, we will see how this behavior changes when we examine the relationship between dividends and time to maturity.

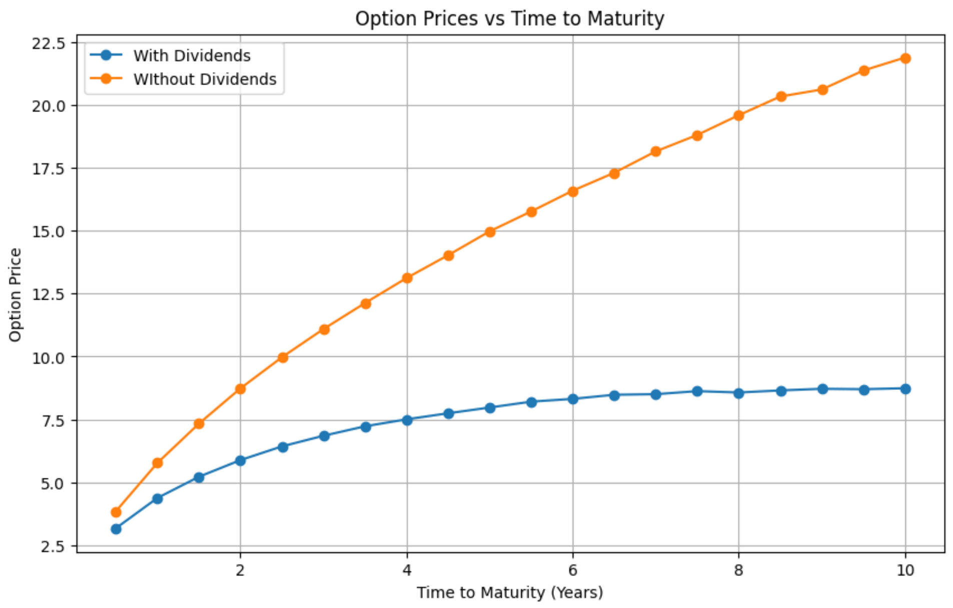


Figure 9: *Estimates of Asian Option Prices With and Without Dividends, with* $S_0 = 100,\ K = 100,\ r = 0.05$ *and* $\sigma = 0.2$

In contrast to the previous plot, we observe that the gap between the option prices increases with time to maturity T. This is due to the cumulative compounding effect of dividends over time. As the time to maturity extends, there are more opportunities for dividends to be paid out, which significantly lowers the expected return of the underlying asset. Consequently, the difference between the prices with and without dividends grows larger as time to maturity increases.

2.5 Geometric Asian Option

In this section, we derive an exact closed-form solution for pricing a Geometric Asian option, specifically, for its use as a control variate when pricing its arithmetic counterpart. Shown in Section 2.2. The key advantage of the Geometric Asian option is that the product of prices follows a Log-Normal distribution, enabling a closed-form solution to exist. By employing this derivative as a control variate, we can significantly increase the accuracy of a standard Monte-Carlo scheme.

A geometric Asian option is very similar to it's arithmetic counter part, except it utilises geometric averaging instead of arithmetic.
That is:

$$G_t = \left(\prod_i^N S_i \right)^{\frac{1}{N}}$$

With payoff:

$$(G_T - K)^+$$

To derive the closed form expression, we can re-write the payoff in log form.

$$\ln G_t = \frac{1}{N} \sum_i^N \ln S_i$$

$$G_t = exp \left\{ \frac{1}{N} \sum_i^N \ln S_i \right\}$$

Which in continuous form:

$$G_t = exp \left\{ \frac{1}{T} \int_0^T \ln S_u \; du \right\}$$

Which we can write as $G = e^Y$, where $Y = \frac{1}{T} \int_0^T S_u \; du$

Recall from **Theorem A7.6** that we can obtain an expression for $\ln S_t$. Leading to:

$$G_t = \exp\left\{ \underbrace{\frac{1}{T}\int_0^T \ln S_0 \, du}_{1} + \underbrace{\frac{1}{T}\int_0^T \left(r - \frac{1}{2}\sigma^2\right) u \, du}_{2} + \underbrace{\frac{1}{T}\int_0^T \sigma W_u \, du}_{3} \right\}$$

$$1 : \frac{1}{T}\int_0^T \ln S_0 \, du = \ln S_0$$

$$2 : \frac{1}{T}\int_0^T \left(r - \frac{1}{2}\sigma^2\right) u \, du = \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2}$$

For the third term, we are more directly interested in its distribution. Specifically its variance, as through standard properties of Brownain Motion its mean is quite clearly zero.

Through Itô isometry (**Theorem A7.5**), Fubini's theorem (**Theorem A7.13**) and properties of Brownian motion, we can write:

$$Var\left(\frac{1}{T}\int_0^T \sigma W_u \, du\right) = E\left[\left(\frac{1}{T}\int_0^T \sigma W_u \, du\right)^2\right]$$

$$= \frac{\sigma^2}{T^2} E\left[\left(\int_0^T W_u^2 \, du\right)\right]$$

$$= \frac{\sigma^2}{T^2} E\left[\int_0^T W_s \; ds \int_0^T W_u \; du\right] = \frac{\sigma^2}{T^2}\int_0^T E[W_s W_u] \; ds \; du$$

$$= \frac{\sigma^2}{T^2}\left[\int_0^s \int_0^T I_{\{}s > u\} E[W_s W_u] \; ds \; du + \int_0^s \int_0^T I_{\{}u > s\} E[W_u W_s] \; ds \; du\right]$$

$$= \frac{2\sigma^2}{T^2}\int_0^s \int_0^T I_{\{}s > u\} E[W_u W_s] \; ds \; du$$

$$= \frac{2\sigma^2}{T^2}\int_0^T \int_0^s E[(W_s - W_u + W_u)W_u] \; du \; ds$$

$$\frac{2\sigma^2}{T^2}\int_0^T \int_0^s u \; du \; ds = \frac{2\sigma^2}{T^2}\int_0^T \frac{s^2}{2} ds$$

$$= \frac{2\sigma^2 T^3}{T^2 6} = \frac{\sigma^2 T}{3}$$

This now gives us the complete distribution for Y:

$$\ln G = Y \sim N\left(\ln S_0 + \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2},\ \frac{\sigma^2 T}{3}\right)$$

Therefore:

$$G \sim LN\left(\ln S_0 + \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2},\ \frac{\sigma^2 T}{3}\right)$$

Recall the present value of a Geometric Asian option is given by:

$$e^{-r(T-t)}E[(G_T - K)^+]$$

Focusing on the expectation we can write:

$$E[(G_T - K)^+] = E[G\, I_{\{G_T > K\}} - K\, I_{\{G_T > K\}}] = E[G\, I_{\{G_T > K\}}] - KE[I_{\{G_T > K\}}]$$

Focusing on the first expectation:

$$E[G\ I_{\{G_T > K\}}] = E[G\ Pr(G_T > K)]$$

$$= E[e^Y\ Pr(\ln G_T > \ln K)]$$

Now, as we know the distribution of $\ln G$ we can express this expectation as a probability density function where $\ln G = Y \sim N(\mu_Y, \sigma_Y^2)$

$$= e^Y \int_{\ln K}^{\infty} \frac{1}{\sqrt{2\pi\sigma_Y^2}} exp\left\{-\frac{(y-\mu_Y)^2}{2\sigma_Y^2}\right\} dy$$

$$= \int_{\ln K}^{\infty} \frac{1}{\sqrt{2\pi\sigma_Y^2}} exp\left\{-\frac{1}{2\sigma_Y^2}(y^2 - 2\mu_Y y + \mu_Y^2 - 2y\sigma_Y^2)\right\} dy$$

$$= \int_{\ln K}^{\infty} \frac{1}{\sqrt{2\pi\sigma_Y^2}} exp\left\{-\frac{1}{2\sigma_Y^2}(y^2 - 2y(\mu_Y + \sigma_Y^2) + \mu_Y^2)\right\} dy$$

$$= \int_{\ln K}^{\infty} \frac{1}{\sqrt{2\pi\sigma_Y^2}} exp\left\{ -\frac{1}{2\sigma_Y^2}[(y - (\mu_Y + \sigma_Y^2))^2 + \mu_Y^2 - (\mu_Y + \sigma_Y^2)^2] \right\} dy$$

$$= exp\left\{ \frac{-\mu_Y^2 + (\mu_Y + \sigma_Y^2)^2}{2\sigma_Y^2} \right\} \int_{\ln K}^{\infty} \frac{1}{\sqrt{2\pi\sigma_Y^2}} exp\left\{ \frac{Y - (\mu_Y + \sigma_Y^2)^2}{2\sigma_Y^2} \right\} dy$$

$$= \underbrace{\exp\left\{ \frac{-\mu_Y^2 + (\mu_Y + \sigma_Y^2)^2}{2\sigma_Y^2} \right\}}_{1} \underbrace{Pr\left(N(\mu_Y + \sigma_Y^2, \sigma_Y^2) > \ln K \right)}_{2}$$

Evaluating 1:

$$exp\left\{ \frac{-\mu_Y^2 + (\mu_Y + \sigma_Y^2)^2}{2\sigma_Y^2} \right\} = exp\left\{ \frac{2\mu_Y\sigma_Y^2 + \sigma_Y^4}{2\sigma_Y^2} \right\}$$
$$= exp\left\{ \mu_Y + \frac{\sigma_Y^2}{2} \right\}$$

Which we can evaluate with known μ_Y and σ_Y^2 as.

$$exp\left\{ \ln S_0 + \left(r - \frac{1}{2}\sigma^2 \right) \frac{T}{2} + \frac{\sigma^2 T}{6} \right\}$$

Now looking at the second term 2:

From properties of the Normal distribution, we can re-write this expression as:

$$Pr\left(N(\mu_Y + \sigma_2, \sigma_Y^2) > \ln K \right) = Pr(\sigma_Y \times N(0,1) + \mu_Y + \sigma_Y^2 > \ln K)$$
$$= Pr\left(Z > \frac{\ln K - \mu_y - \sigma_Y^2}{\sigma_Y} \right)$$
$$= Pr\left(Z > \frac{\ln K - \ln S_0 - \left(r - \frac{1}{2}\sigma^2 \right) \frac{T}{2} - \frac{\sigma^2 T}{3}}{\sigma\sqrt{\frac{T}{3}}} \right)$$
$$= Pr\left(Z < \frac{\ln \frac{S_0}{K} + \left(r - \frac{1}{2}\sigma^2 \right) \frac{T}{2} + \frac{\sigma^2 T}{3}}{\sigma\sqrt{\frac{T}{3}}} \right)$$
$$= \Phi(d_1)$$

Now focusing on the second expectation:

$$KE[I_{\{G_T>K\}}] = KPr(G_T > K)$$

$$= KPr(\ln G_T > \ln K)$$

$$= K\ Pr\left(Z > \frac{\ln K - \ln S_0 - \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2}}{\sigma\sqrt{\frac{T}{3}}}\right)$$

$$= K\ Pr\left(Z < \frac{\ln\frac{S_0}{K} + \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2}}{\sigma\sqrt{\frac{T}{3}}}\right)$$

$$= K\ \Phi(d_2)$$

Note that: $d_2 = d_1 - \sigma_Y$

This gives the final exact solution for a geometric Asian call option:

$$V(S_T, G_T, T) = e^{-r(T-t)}\left[e^{\left\{\ln S_0 + \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2} + \frac{\sigma^2 T}{6}\right\}}\Phi(d_1) - K\Phi(d_2)\right]$$

where

$$d_1 = \frac{\ln\frac{S_0}{K} + \left(r - \frac{1}{2}\sigma^2\right)\frac{T}{2} + \frac{\sigma^2 T}{3}}{\sigma\sqrt{\frac{T}{3}}}$$

$$d_2 = d_1 - \sigma_Y$$

Please note; this solution is in agreement with the results derived by Kemna and Vorst [3].

3 Laplace Transform Approach

As mentioned previously, the complication in pricing the arithmetic Asian option is due to the unknown distribution of the average price. This section addresses this issue by deriving an analytical expression for the Asian option using the properties of Bessel processes and Laplace transforms, as originally introduced by German and Yor [6]. However, the methodology employed here is based on the approach developed by Fusai [4] and Dufresne [5], who provide a review and extension of the original German and Yor framework.

Theorem 7.10 The Laplace transform is an integral transform used to convert a function of time $f(t)$ into a function of a complex variable s. For a function $f(t)$ defined for $t \geq 0$, the Laplace transform $\mathcal{L}\{f(t)\}(s)$ is given by:

$$\mathcal{L}\{f(t)\}(s) = \int_0^\infty e^{-st} f(t)\, dt,$$

where s is a complex number with $\mathrm{Re}(s) > \sigma$, and σ is a real number such that the integral converges.

Please note the full definition can be found in Appendix A ***Theorem A7.10****.*

3.0.1 Re-writing the Payoff

As a first step, we will re write the payoff substituting in our expression for S_t, derived in 1.

$$S_t = S_0 \exp\{(r - \frac{1}{2}\sigma^2)t + \sigma W_t\}$$

Substituting this into the payoff of an Asian option:

$$\left(\frac{1}{T}\int_0^T S_u du - K\right)^+$$

becomes:

$$\left(\frac{S_0}{T}\int_0^T \exp\{(r-\frac{1}{2}\sigma^2)u + \sigma W_u\}du - K\right)^+$$

As suggested by Yor [7], we can apply the following substitutions to make the algebra easier;

$t = \frac{\sigma^2 T}{4}$

$\mu = r - \frac{1}{2}\sigma^2$

$v = \frac{2\mu}{\sigma^2}$

$\int_0^T exp\{\mu u + \sigma W_u\} = \frac{4}{\sigma^2}A_t$

Where we can define $A_t = \int_0^t exp\{2(vu + \sigma W_u)\}du$

Now, applying these substitutions into the payoff:

$$e^{-r(T-t)}E\left[\frac{S_0}{T}\int_0^T \exp\{(r-\frac{1}{2}\sigma^2)u + \sigma W_u\}du - K\right]^+$$

$$= e^{-r(T-t)}E\left[\frac{4S_0}{\sigma^2 T}\int_0^T \frac{\sigma^2}{4}\exp\{(r-\frac{1}{2}\sigma^2)u + \sigma W_u\}du - K\right]^+$$

$$= e^{-r(T-t)}E\left[\frac{4S_0}{\sigma^2 T}A_t - K\right]^+$$

$$= e^{-r(T-t)}\frac{4S_0}{\sigma^2 T}E\left[A_t - K\frac{\sigma^2 T}{4S_0}\right]^+$$

$$= e^{-r(T-t)}\frac{4S_0}{\sigma^2 T}E\left[A_t - k\right]^+ \qquad (4)$$

where $k = K\frac{\sigma^2 T}{4S_0}$ is a new constant variable.

3.0.2 Applying the Laplace Transform

Applying the Laplace transform to the expectation:

$$E[(A_t - k)^+]$$

becomes

$$\int_0^\infty e^{-\lambda s} E[(A_s - k)^+] ds$$

Then using Fubini's theorem (**Theorem A7.13**)

$$\begin{aligned}
\int_0^\infty e^{-\lambda s} E[(A_s - k)^+] ds &= \int_0^\infty e^{-\lambda s} \int_{-\infty}^\infty (A_t - k)^+ \; dads \\
= \int_0^\infty e^{-\lambda s} E[(A_s - k)^+] ds &= \int_0^\infty e^{-\lambda s} \int_k^\infty (A_t - k)^+ \; dads \\
&= \frac{1}{\lambda} \int_0^\infty \int_k^\infty \lambda e^{-\lambda s} A_t \; dads - \frac{1}{\lambda} \int_0^\infty \int_k^\infty \lambda e^{-\lambda s} k \; dads \\
&= \frac{1}{\lambda} \int_k^\infty \int_0^\infty \lambda e^{-\lambda s} A_t \; dsda - \frac{1}{\lambda} \int_k^\infty \int_0^\infty \lambda e^{-\lambda s} k \; dsda \\
&= \frac{1}{\lambda} \int_k^\infty A_t \; da - \frac{1}{\lambda} \int_k^\infty k \; da \\
&= \frac{1}{\lambda} \int_0^\infty (A_t - k)^+ \; da \\
&= \frac{1}{2\lambda} E[(2A_t - 2k)^+]
\end{aligned}$$

In order to solve this expectation, we can to use Yor's theorem (**Theorem A7.11 & A7.12**).

$$2A_t \sim \frac{B(1, \alpha)}{G(\beta, 1)}$$

where:

$B \sim Beta(1, \alpha)$

$G \sim Gamma(\beta, 1)$

$\gamma = \sqrt{\lambda + v^2}$

$\alpha = \frac{\nu+\gamma}{2}$

$\beta = \frac{\nu-\gamma}{2}$

We can then write:

$$E[(2A_t - 2k)^+] = E[(\frac{B(1,\alpha)}{G(\beta,1)} - 2k)^+]$$

Where:

$$u \sim B(1,\alpha) = \frac{u^0(1-u)^{\alpha-1}}{\beta(1,\alpha)}$$
$$x \sim G(\beta,1) = \frac{1^\beta x^{\beta-1}e^{-x}}{\Gamma(\beta)}$$

Note that:

$$\beta(1,\alpha) = \frac{\Gamma(1+\alpha)}{\Gamma(1)\Gamma(\alpha)} = \frac{\alpha\Gamma(\alpha)}{\Gamma(1)\Gamma(\alpha)} = \alpha$$

Using the Beta and Gamma distribution we can then write:

$$E[(\frac{B(1,\alpha)}{G(\beta,1)} - 2k)^+] = \int_0^\infty \int_0^1 \left(\frac{u}{x} - 2k\right)\left(\frac{\alpha(1-u)^{\alpha-1}x^{\beta-1}e^{-x}}{\Gamma(\beta)}\right) dudx \, I_{\{\frac{u}{x}>2k\}}$$
$$= \int_0^{\frac{u}{2k}} \int_0^1 \left(\frac{u}{x} - 2k\right)\left(\frac{\alpha(1-u)^{\alpha-1}x^{\beta-1}e^{-x}}{\Gamma(\beta)}\right) dudx$$
$$= \int_0^{\frac{1}{2k}} \frac{x^{\beta-1}}{\Gamma(\beta)}e^{-x}\, dx \int_{2kx}^1 \left(\frac{u}{x} - 2k\right)\alpha\,(1-u)^{\alpha-1}\, du$$

A finite solution to the second integral is provided by Fusai in [4], giving us:

$$= \int_0^{\frac{1}{2k}} \frac{x^{\beta-1}}{\Gamma(\beta)}e^{-x}\, dx \left[\frac{(1-2kx)^{\alpha+1}}{(1+\alpha)x}\right]$$

Now, substituting γ, α and β in:

$$= \frac{1}{\left(1+\frac{\nu+\gamma}{2}\right)} \int_0^{1/2k} \frac{x^{\frac{\gamma-\nu}{2}-2}(1-2kx)^{\frac{\nu+\gamma}{2}+1}}{\Gamma\left(\frac{\gamma-\nu}{2}\right)} e^{-x}\, dx$$

Now, making use of the recursive property of the Gamma function $\Gamma(...)$, that is:

$$\Gamma(b) = (b-1)\Gamma(b-1)$$

We can then re-write the gamma function:

$$\Gamma\left(\frac{\gamma-\nu}{2}\right) = \left(\frac{\gamma-\nu}{2}-1\right)\Gamma\left(\frac{\gamma-\nu}{2}-1\right)$$

Then, looking at the denominator:

$$\begin{aligned}\left(1+\frac{\nu+\gamma}{2}\right)\times\left(\frac{\gamma-\nu}{2}-1\right) &= \left(-1-\frac{\nu+\gamma}{2}+\frac{\gamma-\nu}{2}+\frac{-\nu^2+\gamma^2}{2}\right)\\ &= \left(-1-\nu+\frac{-\nu^2+\lambda+\nu^2}{2}\right)\\ &= \left(\frac{\lambda-2\nu-2}{2}\right)\end{aligned}$$

Therefore, the expectation can be written as:

$$E[(2A_t-2k)^+] = \frac{1}{\frac{\lambda-2\nu-2}{2}}\int_0^{1/2k}\frac{x^{\frac{\gamma-\nu}{2}-2}(1-2kx)^{\frac{\nu+\gamma}{2}+1}}{\Gamma\left(\frac{\gamma-\nu}{2}-1\right)}e^{-x}\,dx$$

$$\frac{1}{2\lambda}E[(2A_t-2k)^+] = \frac{1}{\lambda(\lambda-2\nu-2)}\int_0^{1/2k}\frac{x^{\frac{\gamma-\nu}{2}-2}(1-2kx)^{\frac{\nu+\gamma}{2}+1}}{\Gamma\left(\frac{\gamma-\nu}{2}-1\right)}e^{-x}\,dx$$

3.0.3 Pricing Formula

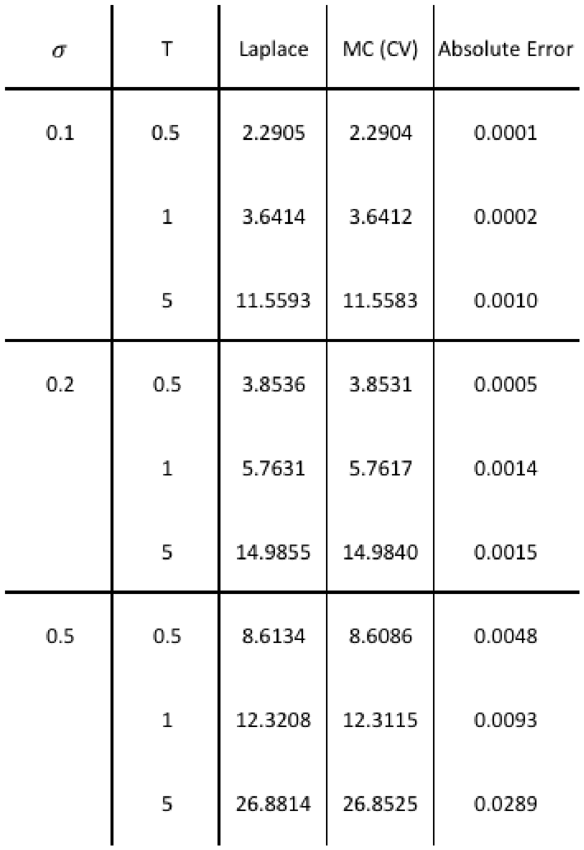
Finally substituting into the original expression 4; we obtain the final analytical solution for an arithmetic Asian option:

$$V(S_T,A_T,T) = e^{-rT}\frac{4S_0}{\sigma^2 T}\,\mathcal{L}^{-1}\left(\frac{1}{\lambda(\lambda-2\nu-2)}\int_0^{1/2q}\frac{x^{\frac{\gamma-\nu}{2}-2}(1-2qx)^{\frac{\nu+\gamma}{2}+1}}{\Gamma\left(\frac{\gamma-\nu}{2}-1\right)}e^{-x}\right)dx$$

where $\mathcal{L}^{-1}$ represents the inverse Laplace transform.

3.0.4 Results

This Laplace approach provides a more analytical / closed-form solution, leading to computationally faster and more accurate results, by not relying on numerical simulations.



σ	T	Laplace	MC (CV)	Absolute Error
0.1	0.5	2.2905	2.2904	0.0001
	1	3.6414	3.6412	0.0002
	5	11.5593	11.5583	0.0010
0.2	0.5	3.8536	3.8531	0.0005
	1	5.7631	5.7617	0.0014
	5	14.9855	14.9840	0.0015
0.5	0.5	8.6134	8.6086	0.0048
	1	12.3208	12.3115	0.0093
	5	26.8814	26.8525	0.0289

Figure 10: *Laplace Transform Results*

Looking at this table comparing the Laplace results to the Control Variate Monte Carlo results, we can see that the Laplace method is a highly accurate alternative, with very small absolute errors. This is better shown in the next plot.

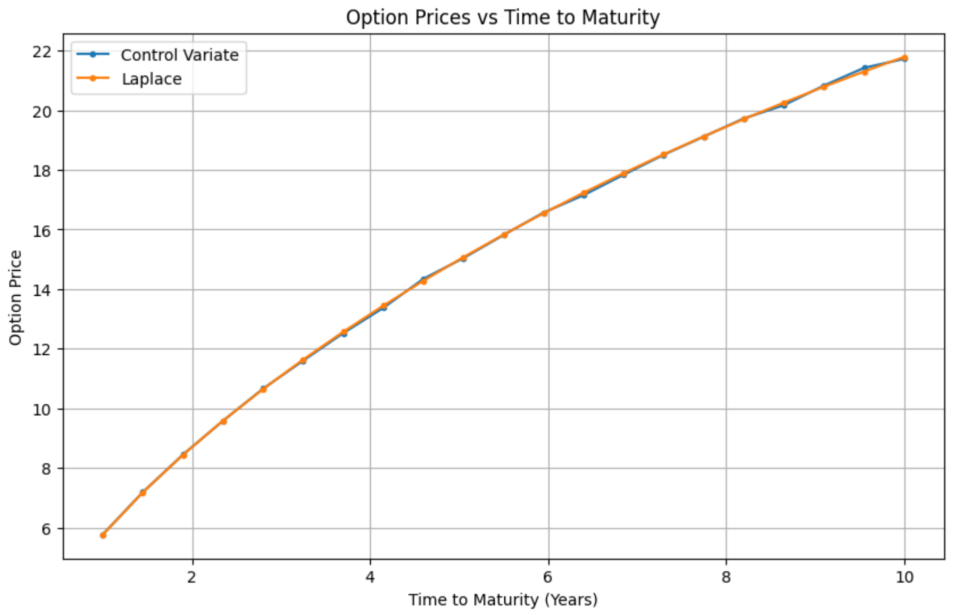


Figure 11: *Laplace Transform and Control Variate Estimates of Asian Option Prices, with* $S_0 = 100,\ K = 100,\ r = 0.05,\ \sigma = 0.2$ *and* $q = 0$

This plot further illustrates the consistent accuracy of the Laplace approach, reinforcing its suitability to price arithmetic Asian options.

The method maintains a very close approximation across different scenarios, however, the error slightly increases with higher volatility and longer time to maturities. This is due to the increased variation in the control variate approach with a limited simulations N.

Whilst this Laplace approach is very quick, certain parameter values can give slowly converging results, particularly for small volatilities and short maturities.

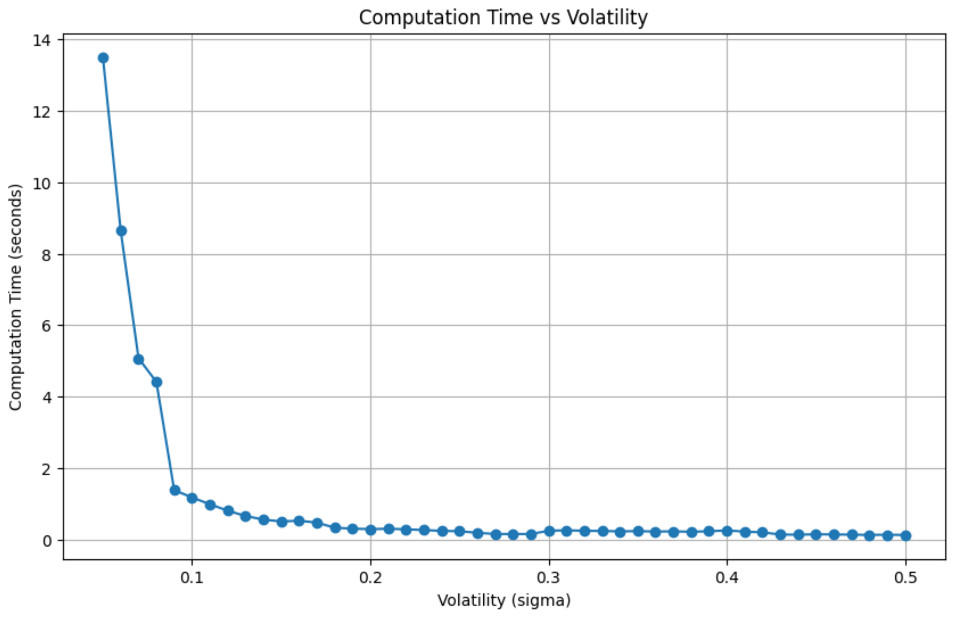


Figure 12: *Laplace Transform Computational Times vs Volatility*

This plot clearly shows that for small volatility values, inverting the Laplace transform becomes more challenging and time consuming.

When volatility is low, the distribution of the underlying asset's returns becomes narrower, reducing the likelihood of extreme values. As a result, the option's payoff tends to be more concentrated around a specific value, leading to sharper peaks in the Laplace domain. To accurately capture these sharp peaks and correctly invert the Laplace transform, the numerical integration must be more precise, often resulting in more computational time per discretization point N.

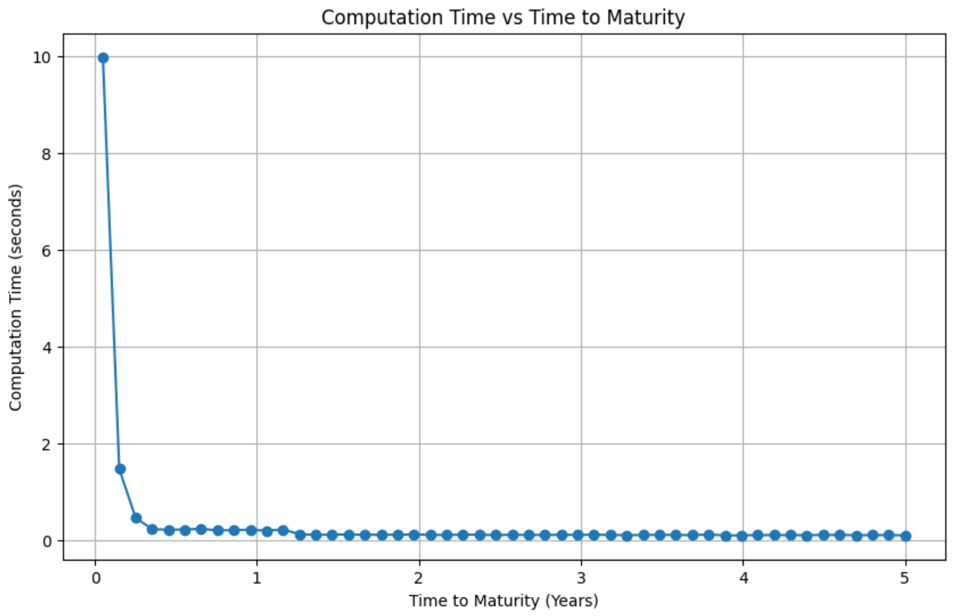


Figure 13: *Laplace Transform Computational Times vs Time to Maturity*

A similar situation can be seen in this plot against time to maturities. When the time to maturity T is short, the option has less time to react to changes in the underlying asset's price. This leads to sharper and more concentrated peaks around S_0 in the Laplace domain. Similar to the case with low volatility, these sharp peaks require more precise numerical integration to accurately invert. Again, resulting in more computational time per discretization point N.

N also plays a significant role in determining computation time and therefore it's value needs to be chosen carefully to balance both accuracy and computational time. This new variable, not to be confused with number of simulations, is the number of discrretization points used in the Laplace inversion. We can think of N as determining the resolution of the numerical integration. A higher N results in a finer grid and more accurate results, but, at the cost of increased computation time. I therefore created a function `find_optimal_N` (found in Appendix B) which acts as a convergence test to find the optimal value for N. In order to ensure high accuracy, the tolerance level was set to 10 decimal points.

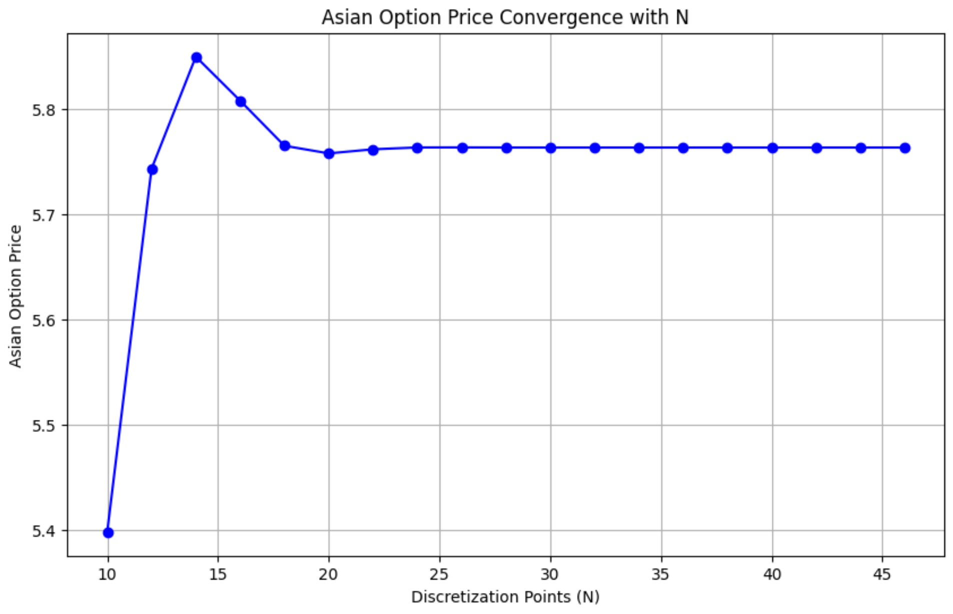


Figure 14: *Convergence Plot with Number of Discretization Points N, with $S_0 = 100,\ K = 100,\ T = 1,\ r = 0.05,\ \sigma = 0.2$ and $q = 0$*

This first plot, set with 'normal' parameter values, shows that only $N = 46$ number of discretization points are required to achieve an option price accurate to 10 decimal points. Even using $N = 100$ for 'normal' parameter valued options greatly reduces computational time while maintaining high accuracy.

Looking at the worst case scenario, when we set $\sigma = 0.10$ and $T = 0.5$:

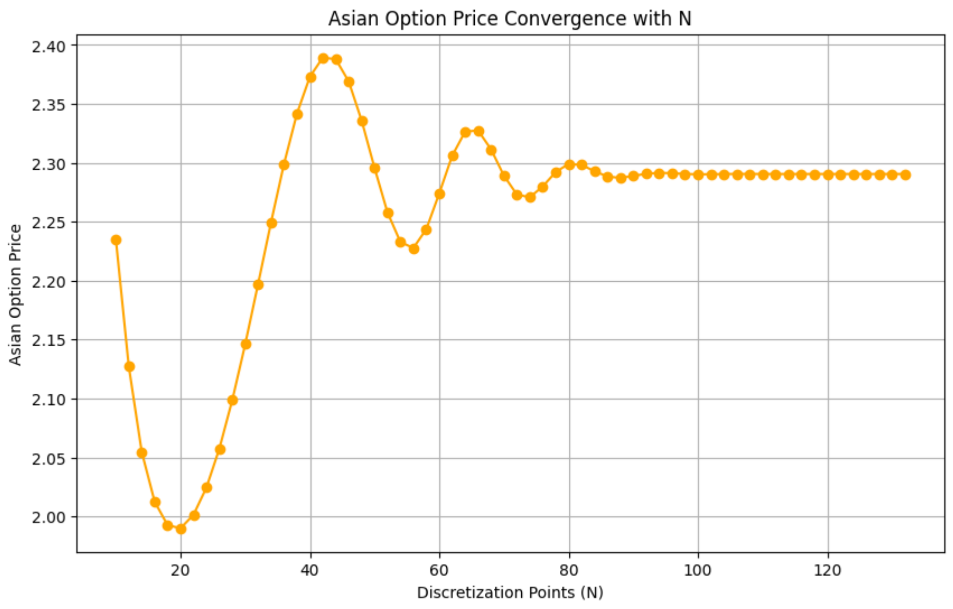
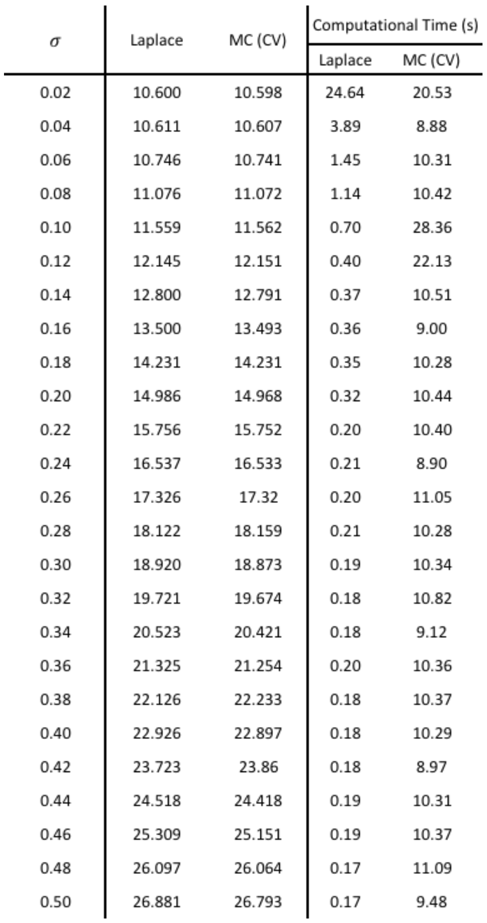


Figure 15: *Convergence Plot with Number of Discretization Points N, with $S_0 = 100,\ K = 100,\ T = 0.5,\ r = 0.05,\ \sigma = 0.1$ and $q = 0$*

We can see that $N = 132$ number of discreteization points are required to achieve accuracy up to 10 decimal points. Therefore, choosing $N = 150$ seems like an appropriate number of discretization points to balance computational time and accuracy across all parameter values.



σ	Laplace	MC (CV)	Computational Time (s)	
			Laplace	MC (CV)
0.02	10.600	10.598	24.64	20.53
0.04	10.611	10.607	3.89	8.88
0.06	10.746	10.741	1.45	10.31
0.08	11.076	11.072	1.14	10.42
0.10	11.559	11.562	0.70	28.36
0.12	12.145	12.151	0.40	22.13
0.14	12.800	12.791	0.37	10.51
0.16	13.500	13.493	0.36	9.00
0.18	14.231	14.231	0.35	10.28
0.20	14.986	14.968	0.32	10.44
0.22	15.756	15.752	0.20	10.40
0.24	16.537	16.533	0.21	8.90
0.26	17.326	17.32	0.20	11.05
0.28	18.122	18.159	0.21	10.28
0.30	18.920	18.873	0.19	10.34
0.32	19.721	19.674	0.18	10.82
0.34	20.523	20.421	0.18	9.12
0.36	21.325	21.254	0.20	10.36
0.38	22.126	22.233	0.18	10.37
0.40	22.926	22.897	0.18	10.29
0.42	23.723	23.86	0.18	8.97
0.44	24.518	24.418	0.19	10.31
0.46	25.309	25.151	0.19	10.37
0.48	26.097	26.064	0.17	11.09
0.50	26.881	26.793	0.17	9.48

Figure 16: *Laplace Transform Results across Volatilities σ, with $S_0 = 100,\ K = 100,\ T = 5,\ r = 0.05$ and $q = 0$*

This table clearly demonstrates that the Laplace method is vastly superior to the Monte Carlo method in terms of computational efficiency, highlighting its dominance in speed and effectiveness for pricing an Asian option. However, we must note that for cases $\sigma < 0.1$ the inversion does struggle with convergence, while still maintaining accurate answers. Therefore, these extreme cases should be priced using a Monte Carlo (CV) approach.

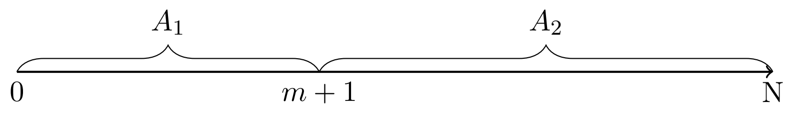
4 Moment Matching Approach

While the density function of the arithmetic average is unknown, more specifically not log-normal. We can adopt an approach, introduced by Levy [8], using the so-called 'Wilkinson Approximation' which is used to approximate the arithmetic density function by matching the first two moments to a Log-Normal distribution.

We can do this by introducing a new variable $M(t)$ which represents the sum of Log-Normal random variables.

4.0.1 Defining the new Variable

Consider an Asian option timeline:



Where:
$m+1$ is a time between $0 \leq m+1 \leq N$
A_1 = the average share price up to $m+1 = \frac{1}{m+1}\sum_{i=0}^{m} S_i$
A_2 = the average share price from $m+1$ up to $N = \frac{1}{N-m}\sum_{i=m+1}^{N} S_i$
Clearly, $m = 0$ represents the initial time t_0.
From this notation we can then formulate an expression for the total average share price $A(t_N)$.

$$A(t_N) = A_1 \times \left(\frac{m+1}{N}\right) + \underbrace{A_2 \times \left(1 - \frac{m+1}{N}\right)}_{M(t)}$$

Where $M(t)$ represents the average share price after time $t = m+1$. Giving us:

$$M(t) = A(t_N) - A_1 \times \left(\frac{m+1}{N+1}\right)$$

4.0.2 Moments Calculation

Using the approximation that the sum of Log-Normals is a Log-Normal distribution with unknown mean $\alpha(t)$ and unknown variance $v(t)$

$$M(t) \sim LN(\alpha(t), v(t))$$

$$\ln M(t) \sim N(\alpha(t), v(t))$$

Then using the Moment Generating Function defined in **Theorem A7.14**:

$$E\left[e^{\ln Mk}\right] = E\left[M^k\right] = exp\left\{k\alpha(t) + \frac{1}{2}k^2 v(t)^2\right\}$$

We can now find $\alpha(t)$ and $v(t)$:

$$E[M] = exp\left\{\alpha(t) + \frac{1}{2}v(t)^2\right\}$$

$$E[M^2] = exp\left\{2\alpha(t) + \frac{1}{2}4v(t)^2\right\}$$

$$\ln E[M] = \alpha(t) + \frac{1}{2}v(t)^2$$

$$\ln E[M^2] = 2\alpha(t) + 2v(t)^2$$

$$\alpha(t) = \ln E[M] - \frac{1}{2}\left[\frac{1}{2}\ln E[M^2] - \alpha(t)\right]$$

$$\alpha(t) = 2\ln E[M] - \frac{1}{2}\ln E[M^2]$$

$$v(t)^2 = 2\ln E[M] - 2\alpha(t)$$

$$v(t) = \sqrt{\ln E[M^2] - 2\ln E[M]}$$

By assuming $M(t) \sim LN(\alpha(t), v(t))$, we can write the price of the option as:

$$V(S_T, A_T, T) = e^{-r(T-t)} E\left[(A(t_N) - K)^+\right]$$

$$= e^{-r(T-t)} E\left[\left(M(t) - \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right)\right)^+\right]$$

$$= e^{-r(T-t)} E\left[M(t) I_{\{M(t) > K - A(t)\frac{m+1}{N+1}\}}\right] - e^{-r(T-t)}\left(K - A(t)\left(\frac{m+1}{N+1}\right)\right) E\left[I_{\{M(t) > K - A(t)\frac{m+1}{N+1}\}}\right]$$

4.0.3 Solving Expectations

Simarly to the derivation in Section 2.5 using $\ln M(t) = Y \sim N(\alpha(t), v(t))$, we can evaluate the first expectation as:

$$= e^Y \int_{\ln\left|K - A(t)\frac{m+1}{N+1}\right|}^{\infty} \frac{1}{\sqrt{2\pi v(t)}} exp\left\{-\frac{(y-\alpha)^2}{2v(t)}\right\} dy$$

$$= \int_{\ln\left|K - A(t)\frac{m+1}{N+1}\right|}^{\infty} \frac{1}{\sqrt{2\pi v(t)}} exp\left\{-\frac{1}{2v(t)}(y^2 - 2\alpha y + \alpha^2 - 2yv(t))\right\} dy$$

$$= \int_{\ln\left|K - A(t)\frac{m+1}{N+1}\right|}^{\infty} \frac{1}{\sqrt{2\pi v(t)}} exp\left\{-\frac{1}{2v(t)}(y^2 - 2y(\alpha + v(t)) + \alpha^2)\right\} dy$$

$$= \int_{\ln\left|K - A(t)\frac{m+1}{N+1}\right|}^{\infty} \frac{1}{\sqrt{2\pi v(t)}} exp\left\{-\frac{1}{2v(t)}[(y - (\alpha + v(t)))^2 + \alpha^2 - (\alpha + v(t))^2]\right\} dy$$

$$= exp\left\{\frac{-\alpha^2 + (\alpha + v(t))^2}{2v(t)}\right\} \int_{\ln\left|K - A(t)\frac{m+1}{N+1}\right|}^{\infty} \frac{1}{\sqrt{2\pi v(t)}} exp\left\{\frac{Y - (\alpha + v(t))^2}{2v(t)}\right\} dy$$

$$= \underbrace{\exp\left\{\frac{-\alpha^2 + (\alpha + v(t))^2}{2v(t)}\right\}}_{1} \underbrace{Pr\left(N(\alpha + v(t), v(t)) > ln\left|K - A(t)\frac{m+1}{N+1}\right|\right)}_{2}$$

We can show that the first term $\rightarrow E[M(t)]$:

$$exp\left\{\frac{-\alpha^2 + (\alpha + v(t))^2}{2v(t)}\right\} = exp\left\{\frac{2\alpha v(t) + v(t)^2}{2v(t)}\right\}$$

$$= exp\left\{\alpha + \frac{v(t)}{2}\right\}$$

$$= exp\left\{2\ln E[M(t)] - \frac{1}{2}\ln E[M(t)^2] - \frac{1}{2}\left(\ln E[M(t)^2] - 2\ln E[M(t)]\right)^{\frac{1}{2}}\right\}$$

This can be hard to recognise, but using simple Logarithm properties, we can write:

$$= exp\left\{lnE[M(t)] - 0\right\}$$

$$= E[M(t)]$$

Now evaluating the second term:

$$Pr\left(N(\alpha + v(t), v(t)) > \ln\left|K - A(t)\frac{m+1}{N+1}\right|\right)$$

$$= Pr\left(Z > \frac{\ln\left|K - A(t)\frac{m+1}{N+1}\right| - \alpha(t) - v(t)}{\sqrt{v(t)}}\right)$$

$$= Pr\left(Z > \frac{\ln\left|K - A(t)\left(\frac{m+1}{N+1}\right)\right| - 2\ln E[M(t)] + \frac{1}{2}\ln E[M(t)^2] - \ln E[M(t)^2] + 2\ln E[M(t)]}{\left(\ln E[M(t)^2] - 2\ln E[M(t)]\right)^{\frac{1}{4}}}\right)$$

$$= Pr\left(Z < \frac{\frac{1}{2}\ln E[M(t)^2] - \ln\left|K - A(t)\left(\frac{m+1}{N+1}\right)\right|}{\left(\ln E[M(t)^2] - 2\ln E[M(t)]\right)^{\frac{1}{4}}}\right)$$

$$= \Phi(d_1)$$

Now focusing on the second expectation:

$$\left(K - A(t)\left(\frac{m+1}{N+1}\right)\right) Pr\left(M(t) > \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right)\right)$$

$$= \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right) Pr\left(\ln M(t) > \ln\left|K - A(t)\left(\frac{m+1}{N+1}\right)\right|\right)$$

$$= \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right) Pr\left(Z > \frac{\ln\left|K - A(t)\left(\frac{m+1}{N+1}\right)\right| - 2\ln E[M(t)] + \frac{1}{2}\ln E[M(t)^2]}{\left(\ln E[M(t)^2] - 2\ln E[M(t)]\right)^{\frac{1}{4}}}\right)$$

$$= \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right) Pr\left(Z < \frac{-\ln\left|K - A(t)\left(\frac{m+1}{N+1}\right)\right| + 2\ln E[M(t)] - \frac{1}{2}\ln E[M(t)^2]}{(\ln E[M(t)^2] - 2\ln E[M(t)])^{\frac{1}{4}}}\right)$$

$$= \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right)\Phi(d_2)$$

Where:

$$d_1 = d_2 + \sqrt{v(t)}$$

$$d_1 = \frac{\frac{1}{2}\ln E[M(t)^2] - \ln\left|K - A(t)\left(\frac{m+1}{N+1}\right)\right|}{(\ln E[M(t)^2] - 2\ln E[M(t)])^{\frac{1}{4}}}$$

Giving us the following pricing formula:

$$e^{-r(T-t)}\left[E[M(t)]\Phi(d_1) - \left(K - A(t)\left(\frac{m+1}{N+1}\right)\right)\Phi(d_2)\right]$$

Now, in order to actually price this derivative, we need to first solve the values of $E[M(t)]$ and $E[M(t)^2]$.

By the definition of $M(t)$:

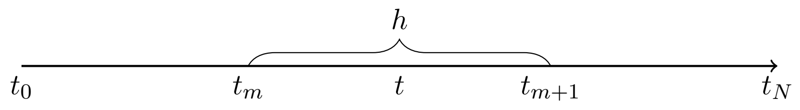
$$E[M(t)] = E[A(t_N)] - E[A_1]\left(\frac{m+1}{N+1}\right)$$

$$= \frac{1}{N+1}\sum_{i=0}^{N} E[S_i] - \frac{1}{m+1}\sum_{j=0}^{m} E[S_j]\frac{m+1}{N+1}$$

$$= \frac{1}{N+1}\sum_{i=0}^{N} E[S_i] - \frac{1}{N+1}\sum_{j=0}^{m} E[S_j]$$

$$= \frac{1}{N+1}\sum_{i=m+1}^{N} E[S_i]$$

Now, by using our expression for S_t and by defining a new timeline:



Where: $Nh = t_N - t_0$

and define: $\psi = \frac{t - t_m}{h}$

We can then write:

$$E[M(t)] = \frac{1}{N+1} \sum_{i=m+1}^{N} S(t) e^{r(i-m-\psi)h}$$

Which we can recognise as the sum of a finite geometric series:

$$= \frac{S(t)}{N+1} e^{r(1-\psi)h} \left[\frac{1 - e^{r(N-m)h}}{1 - e^{rh}} \right]$$

Now by taking $N \to \infty$, $h \to 0$ and $Nh = t_N - t_0$, we can write:

$$= \frac{S(t)}{N+1} e^{0} \left[\frac{1 - e^{rNh - gmh}}{1 - e^{rh}} \right]$$

$$= \frac{S(t)}{N+1} \left[\frac{1 - e^{r(t_N - t_0)}}{1 - e^{rh}} \right]$$

Now, if we apply a Taylor series expansion on the denominator:

$$(N+1)(1 - e^{rh}) = (N+1) \left[1 - \left(1 + rh + \frac{(rh)^2}{2!} + \frac{(rh)^3}{3!} + \cdots \right) \right]$$

$$= -Nrh = -r(t_n - t_0)$$

Therefore we have an expression for $\lim_{n \to \infty} E[M(t)]$

$$\lim_{n \to \infty} E[M(t)] = \frac{S(t)}{r(t_N - t_0)} (e^{r(t_N - t_0)} - 1)$$

In a similar manner, we can derive an expression for $\lim_{n\to\infty} E[M(t)^2]$ as well. The proof can be found in [8].

$$\lim_{n\to\infty} \mathbb{E}\left[M(t)^2\right] = \frac{2S(t)^2}{(t_N - t_0)^2(r+\sigma^2)}\left[\frac{e^{(2r+\sigma^2)(t_N-t_0)}-1}{2r+\sigma^2} - \frac{e^{r(t_N-t_0)}-1}{r}\right]$$

Now we have expressions for these terms, we can perform certain substitutions to formulate an analytical approximate pricing formula.

For better notation we can change our timeline to be in terms of $0 \to T$ where T_1 and T_2 represent the first and second averaging period respectively.

We can then write:

$K^* = K - S_{AVG}\frac{T-T_2}{T}$ from $K - A(t)\left(\frac{m+1}{N+1}\right)$

$d_1 = \frac{\frac{1}{2}\ln L - \ln K^*}{\sqrt{v}}$, where $L = E[M(t)^2]$ and $d_2 = d_1 - \sqrt{v}$

By defining $S_Z = e^{-rT_2}E[M(t)]$, we get:

$$= e^{-rT_2}\frac{S}{rT}\left(e^{rT_2}-1\right)$$

$$= \frac{S}{rT}\left(1-e^{-rT_2}\right)$$

Then we can write:

$v = \ln L - 2(rT_2 + \ln S_Z)$

Where:

S = Spot price

S_{AVG} = Average asset price (known)

K = Strike price

T = Time to maturity

T_2 = Time remaining until maturity

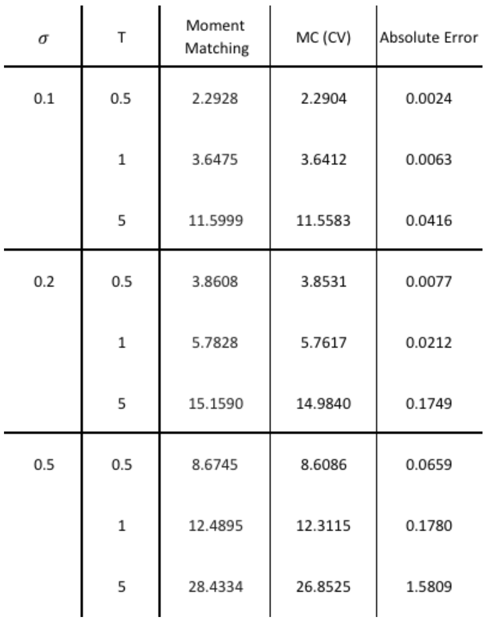
σ = Observed volatility

4.0.4 Pricing Formula

Giving us the final expression for the analytical approximation of the Arithmetic Asian option price:

$$V(S_T, A_T, T) = S_Z\Phi(d_1) - e^{-rT_2}K^*\Phi(d_2)$$

4.0.5 Results



σ	T	Moment Matching	MC (CV)	Absolute Error
0.1	0.5	2.2928	2.2904	0.0024
	1	3.6475	3.6412	0.0063
	5	11.5999	11.5583	0.0416
0.2	0.5	3.8608	3.8531	0.0077
	1	5.7828	5.7617	0.0212
	5	15.1590	14.9840	0.1749
0.5	0.5	8.6745	8.6086	0.0659
	1	12.4895	12.3115	0.1780
	5	28.4334	26.8525	1.5809

Figure 17: *Moment Matching Results, with* $S_0 = 100,\ K = 100,\ r = 0.05$ *and* $q = 0$

Looking at this table comparing the Moment Matching results to the Control Variate Monte Carlo results, we can see that the Moment Matching method is fairly accurate for smaller parameter values. However, as σ and T increases, the absolute errors do increase slightly.

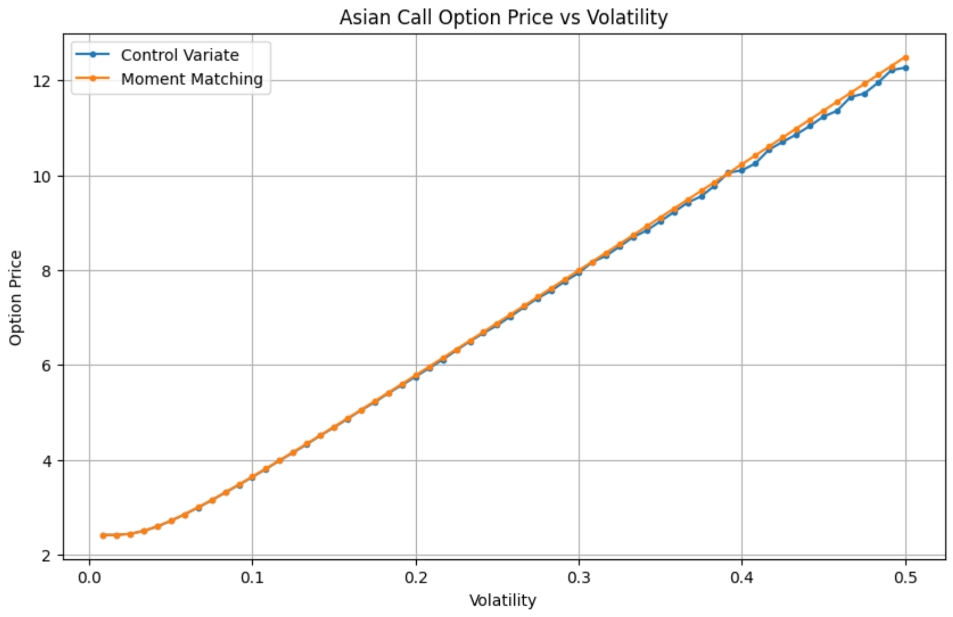


Figure 18: *Moment Matching and Control Variate Estimates, with* $S_0 = 100,\ K = 100,\ T = 1,\ r = 0.05$ *and* $q = 0$

This plot further supports that the Moment Matching approach does return accurate prices for the most part, especially for smaller volatility values, trumping the Laplace Transform approach. However, this plot also illustrates the increased error as volatility σ increases. *Please note: only 1000 simulations were used for the Control Variate approach to clearly illustrate the gap in prices.*

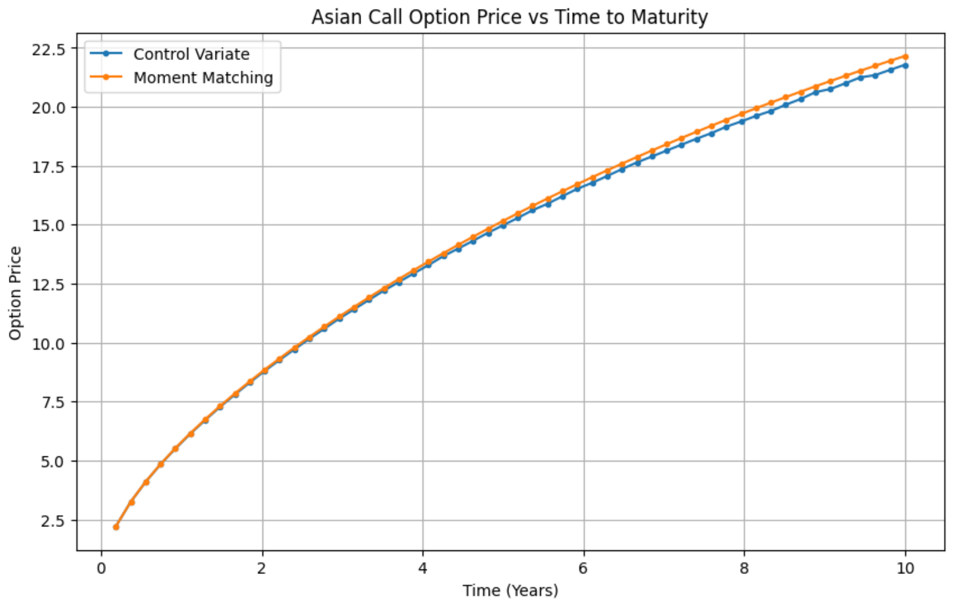
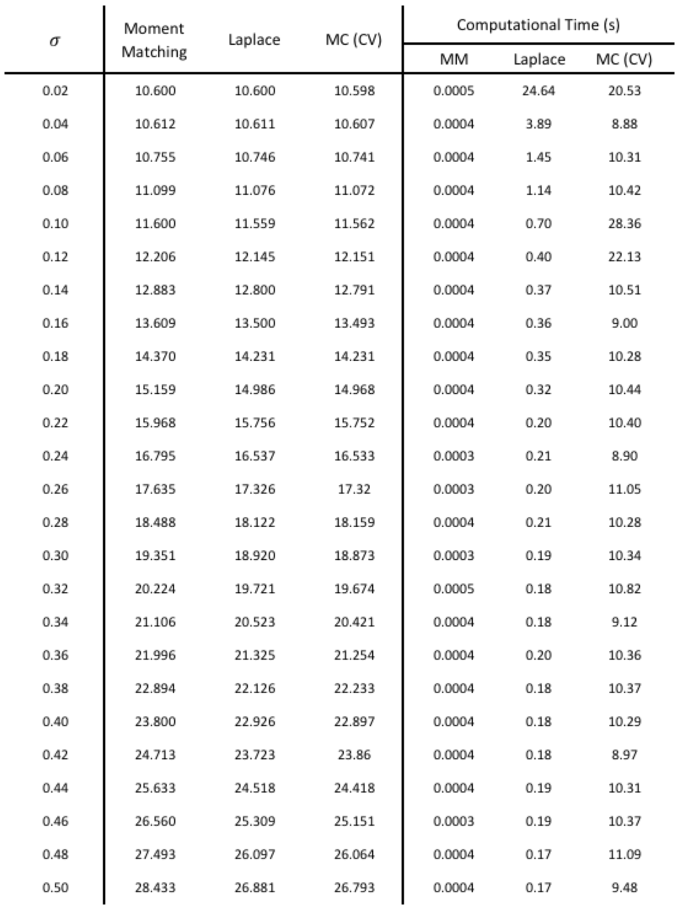


Figure 19: *Moment Matching and Control Variate Estimates, with* $S_0 = 100,\ K = 100,\ r = 0.05,\ \sigma = 0.2$ *and* $q = 0$

A similar situation can be seen in this plot against time to maturity as well. Highlighting that the Moment Matching approach does return accurate prices for the most part and again trumping the Laplace Transform approach for small T values.

The reason why this approximation method becomes less accurate as both σ and T increase is due to the higher order moments. The fundamental approach of this method is to match the Mean and Variance of the payoffs to a Log-Normal distirbution. However, for options with high volatility and longer time to maturities, the higher-order moments (such as skewness and kurtosis) play a role in determining the shape of the payoff distribution. Whereas, just the mean and variance will not be able to capture these changes. When the maturity T is large, even small errors in the approximation of the distribution can compound over time, thereby, increasing the absolute error as T increases.



σ	Moment Matching	Laplace	MC (CV)	Computational Time (s)		
				MM	Laplace	MC (CV)
0.02	10.600	10.600	10.598	0.0005	24.64	20.53
0.04	10.612	10.611	10.607	0.0004	3.89	8.88
0.06	10.755	10.746	10.741	0.0004	1.45	10.31
0.08	11.099	11.076	11.072	0.0004	1.14	10.42
0.10	11.600	11.559	11.562	0.0004	0.70	28.36
0.12	12.206	12.145	12.151	0.0004	0.40	22.13
0.14	12.883	12.800	12.791	0.0004	0.37	10.51
0.16	13.609	13.500	13.493	0.0004	0.36	9.00
0.18	14.370	14.231	14.231	0.0004	0.35	10.28
0.20	15.159	14.986	14.968	0.0004	0.32	10.44
0.22	15.968	15.756	15.752	0.0004	0.20	10.40
0.24	16.795	16.537	16.533	0.0003	0.21	8.90
0.26	17.635	17.326	17.32	0.0003	0.20	11.05
0.28	18.488	18.122	18.159	0.0004	0.21	10.28
0.30	19.351	18.920	18.873	0.0003	0.19	10.34
0.32	20.224	19.721	19.674	0.0005	0.18	10.82
0.34	21.106	20.523	20.421	0.0004	0.18	9.12
0.36	21.996	21.325	21.254	0.0004	0.20	10.36
0.38	22.894	22.126	22.233	0.0004	0.18	10.37
0.40	23.800	22.926	22.897	0.0004	0.18	10.29
0.42	24.713	23.723	23.86	0.0004	0.18	8.97
0.44	25.633	24.518	24.418	0.0004	0.19	10.31
0.46	26.560	25.309	25.151	0.0003	0.19	10.37
0.48	27.493	26.097	26.064	0.0004	0.17	11.09
0.50	28.433	26.881	26.793	0.0004	0.17	9.48

Figure 20: *Moment Matching, Laplace Transform and Control Variate Estimates*

This table shows that the Moment Matching approach significantly outperforms the Laplace method in terms of computational time, demonstrating a clear advantage in efficiency, especially for small σ values. The errors and misprices are hard to spot in this table and will be further examined in the next plot.

Having shown that the Moment Matching approach significantly outperforms the Laplace in computational time, we now need to investigate its accuracy in order to evaluate this method entirely.

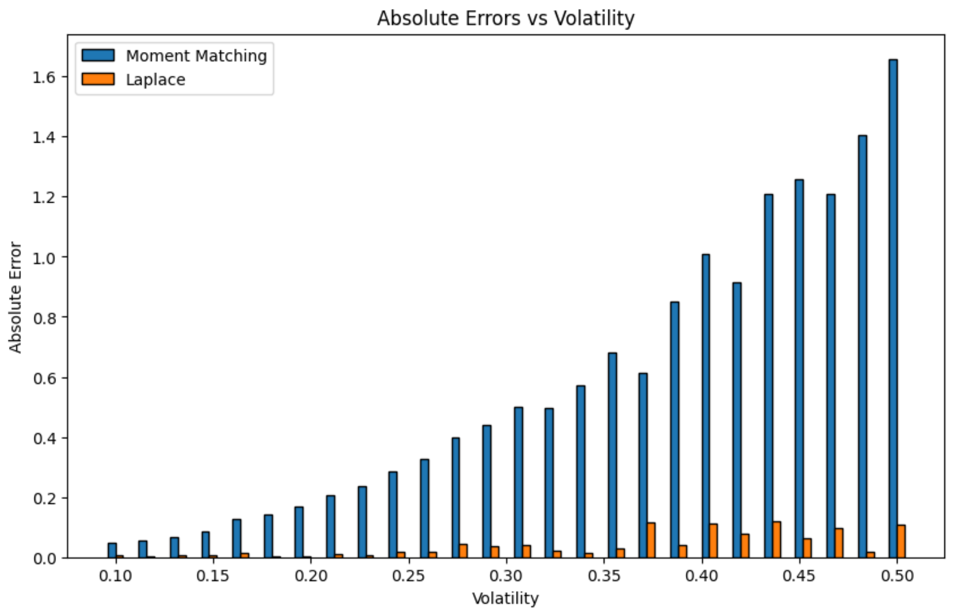
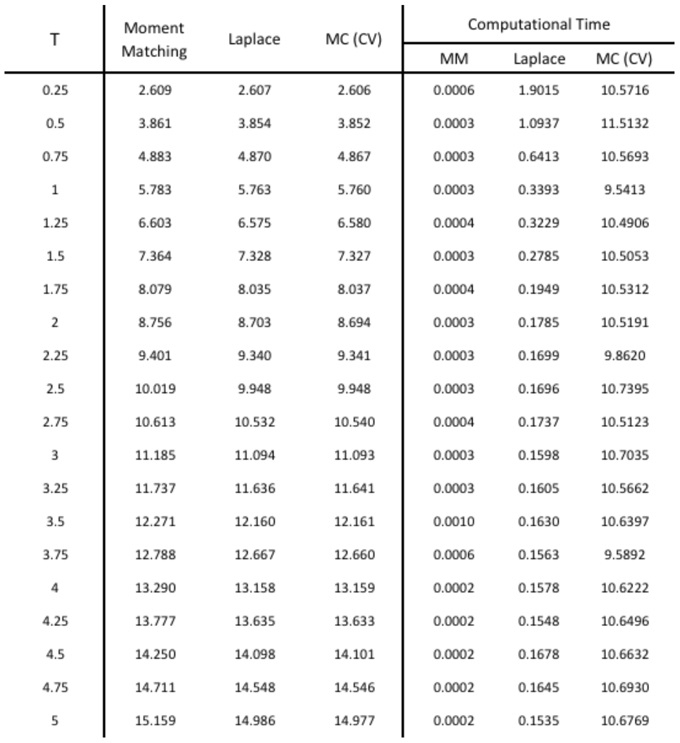


Figure 21: *Moment Matching and Laplace Transform Errors using Control Variate as a Benchmark*

This histogram of absolute errors versus volatility illustrates that, while the Moment Matching approach is faster, it is consistently outperformed by the Laplace transform approach in terms of accuracy.

It's important to note that the last two figures were calculated using $T = 5$, as recall that the Laplace transform method struggles with small values of σ and T.

Therefore, fixing $\sigma = 0.2$ and varying T, we have the following table and histogram:



T	Moment Matching	Laplace	MC (CV)	Computational Time		
				MM	Laplace	MC (CV)
0.25	2.609	2.607	2.606	0.0006	1.9015	10.5716
0.5	3.861	3.854	3.852	0.0003	1.0937	11.5132
0.75	4.883	4.870	4.867	0.0003	0.6413	10.5693
1	5.783	5.763	5.760	0.0003	0.3393	9.5413
1.25	6.603	6.575	6.580	0.0004	0.3229	10.4906
1.5	7.364	7.328	7.327	0.0003	0.2785	10.5053
1.75	8.079	8.035	8.037	0.0004	0.1949	10.5312
2	8.756	8.703	8.694	0.0003	0.1785	10.5191
2.25	9.401	9.340	9.341	0.0003	0.1699	9.8620
2.5	10.019	9.948	9.948	0.0003	0.1696	10.7395
2.75	10.613	10.532	10.540	0.0004	0.1737	10.5123
3	11.185	11.094	11.093	0.0003	0.1598	10.7035
3.25	11.737	11.636	11.641	0.0003	0.1605	10.5662
3.5	12.271	12.160	12.161	0.0010	0.1630	10.6397
3.75	12.788	12.667	12.660	0.0006	0.1563	9.5892
4	13.290	13.158	13.159	0.0002	0.1578	10.6222
4.25	13.777	13.635	13.633	0.0002	0.1548	10.6496
4.5	14.250	14.098	14.101	0.0002	0.1678	10.6632
4.75	14.711	14.548	14.546	0.0002	0.1645	10.6930
5	15.159	14.986	14.977	0.0002	0.1535	10.6769

Figure 22: *Moment Matching, Laplace Transform and Control Variate Estimates with Computation Time*

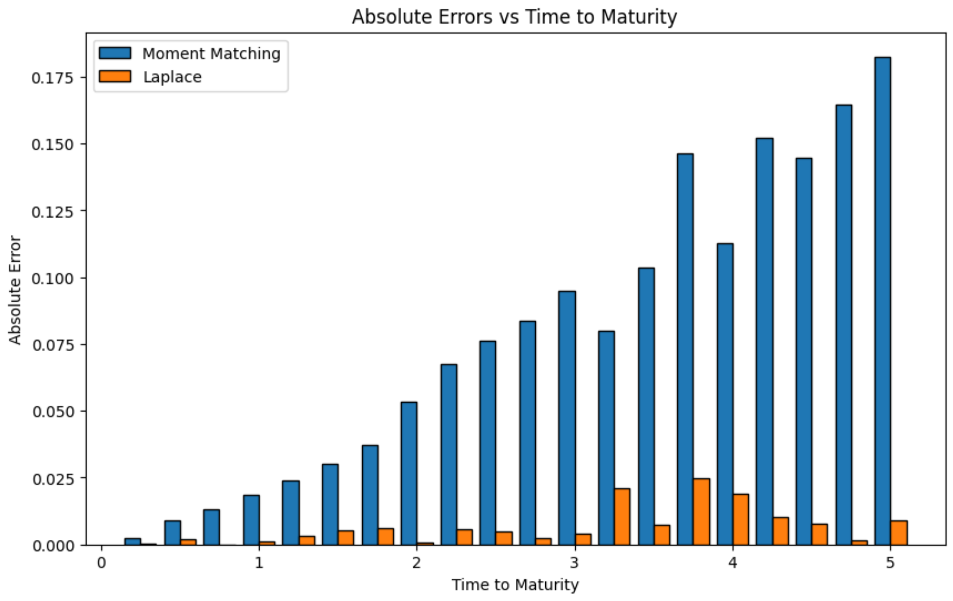


Figure 23: *Moment Matching and Laplace Transform Errors using Control Variate as a Benchmark*

In a similar way to before, the histogram still illustrates that the Laplace transform approach is more accurate than the Moment Matching method, however the Moment Matching errors are much smaller for low T values. Implying that for 'small' parameter valued options, $\sigma < 0.1$ and $T < 0.5$ the Moment Matching method could be superior due to its incredibly quick computational time while maintaining a level of accuracy.

4.1 Moment matching with dividends

Previously, we derived an expression to price an Asian option under the assumption that no dividends are paid during the life of the option. However, in many real-world scenarios, the underlying asset may pay dividends, which can significantly impact the option's value.

To account for this, we can extend our previous methodology to incorporate the effects of dividends. Specifically, we will adjust the underlying asset price dynamics to reflect the continuous dividend yield, as shown in Section 2.4.

If we replace r in our previous derivation with g which reflects the drift term of the price dynamics.

$$dS_t = gS_t dt + \sigma S_t dW_t$$

We can let $g = r - q$, reflecting the expected return on the stock reduced by the dividend yield q, and perform the same steps as before:

$$\lim_{n\to\infty} E[M(t)] = \frac{S(t)}{g(t_N - t_0)}(e^{g(t_N - t_0)} - 1)$$

and

$$\lim_{n\to\infty} \mathbb{E}\left[M(t)^2\right] = \frac{2S(t)^2}{(t_N - t_0)^2(g + \sigma^2)}\left[\frac{e^{(2g+\sigma^2)(t_N - t_0)} - 1}{2g + \sigma^2} - \frac{e^{g(t_N - t_0)} - 1}{g}\right]$$

Now using the substitution $g = r - q$:

$$S_Z = \frac{S}{(r-q)T}\left(e^{-qT_2} - e^{-rT_2}\right)$$

$$L = \frac{2S(t)^2}{(t_N - t_0)^2((r-q) + \sigma^2)}\left[\frac{e^{(2(r-q)+\sigma^2)T_2} - 1}{2(r-q) + \sigma^2} - \frac{e^{(r-q)T_2} - 1}{(r-q)}\right]$$

Which gives the same analytical expression for the price of the Arithmetic Asian option:

$$V(S_T, A_T, T) = S_Z\Phi(d_1) - e^{-rT_2}K^*\Phi(d_2)$$

but with different substitutions for S_Z and L.

5 Martingale & PDE Approach

This section focuses on pricing an Asian option through solving partial differential equations (PDE's). Starting with a standard finite difference Crank Nicolson scheme, before using a Martingale approach to derive a new PDE.

5.0.1 Deriving the PDE

I will first derive the PDE for a standard European Call option before deriving for the Arithmetic Asian option.

5.0.2 European Call

Consider the price of the underlying asset S_t under a risk-neutral measure, which follows a Geometric Brownian Motion:

$$dS_t = rS_t\,dt + \sigma S_t\,dW_t$$

where r is the risk-free interest rate, σ is the volatility and W_t is a Brownian Motion (Wiener process).
The Bank account follows:

$$dB_t = rB_t dt$$

with constant interest rate r.
For a standard European option, the payoff depends only one the final stock price S_T with payoff:

$$(S_T - K)^+$$

Let $V(S_t, t)$ represent the price of a European option as a function of the current asset price S_t and time t.

Applying Itô's lemma formula (**Theorem A7.1**) to $V(S_t, t)$:

$$dV = \frac{\partial V}{\partial t} dt + \frac{\partial V}{\partial S} dS_t + \frac{1}{2}\frac{\partial^2 V}{\partial S^2} dS_t^2$$
$$= \frac{\partial V}{\partial t} dt + \frac{\partial V}{\partial S} [rS_t dt + \sigma S_t dW_t] + \frac{1}{2}\frac{\partial^2 V}{\partial S^2} \sigma^2 S_t^2 dt$$

Then grouping dt and dW_t terms:

$$\left(\frac{\partial V}{\partial t} + \frac{\partial V}{\partial S} rS_t + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2}\right) dt + \left(\sigma S_t \frac{\partial V}{\partial S}\right) dW_t$$

Through a Delta Hedging strategy, where can we can hedge the risk of the option by trading the underlying S_t, we can eliminate the exposure from the Brownian term.

Consider a portfolio Π which is formed to replicate the behaviour of the option.

$$\Pi = \Delta S_t + \alpha B_t$$

$\Delta =$ number of units of stock

$\alpha =$ number of units borrowed form the bank

with the following dynamics:

$$d\Pi = \Delta dS_t + \alpha dB_t$$

$$= \Delta(rdS_t + \sigma S_t dW_t) + \alpha r B_t dt$$

$$= (\Delta r S_t + \alpha r B_t)\, dt + \Delta \sigma S_t dW_t$$

The total portfolio is now $dV + d\Pi$ consisting of an option, stocks and the bank account.

We can choose Δ to eliminate the Brownian:

Grouping the dW_t terms

$$\Delta\sigma S_t + \sigma S_t \frac{\partial V}{\partial S} = 0$$

$$\Delta = -\frac{\partial V}{\partial S}$$

Therefore, substituting Δ into the the total portfolio:

$$\begin{aligned} dV + d\Pi &= \left(\frac{\partial V}{\partial t} + rS_t \frac{\partial V}{\partial S} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} - rS_t \frac{\partial V}{\partial S} + \alpha r B_t \right) dt \\ &= \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + \alpha r B_t \right) dt \end{aligned}$$

For the portfolio to be arbitrage-free, it must grow at the risk-free rate r. As shown in [9]

That is:

$$d(V + \Pi) = (V + \Pi) r dt$$

$$\left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + \alpha r B_t \right) dt = \left(V - \frac{\partial V}{\partial S} S + \alpha B_t \right) r dt$$

Which gives the final PDE for a European Option

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = 0$$

5.0.3 Asian Call

However this PDE will be slightly different for an Asian option as the value $V(S_t, A_t, t)$ depends on the averaging property A_t as well.

Define:

$$A_t = \frac{1}{N} \sum_{i=1}^{N} S_i$$

The continuous definition can be written as:

$$A_t = \frac{1}{t}\int_0^t S_u du$$

Therefore applying Itô's lemma formula to $V(S_t, A_t, t)$:

$$dV = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}dS_t + \frac{\partial V}{\partial A}dA_t + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}dS_t^2$$

Note $dA^2 = 0$ and $\langle A, S\rangle_t = 0$ as A_t is a deterministic average with no stochastic term.

Note by product rule (**Theorem A7.3**):

$$\begin{aligned} dA_t &= \frac{\partial}{\partial t}\left(\frac{1}{t}\right)\int_0^t S_u\,du + \frac{1}{t}\cdot\frac{\partial}{\partial t}\left(\int_0^t S_u\,du\right). \\ &-\frac{1}{t^2}\int_0^t S_u\,du\ + \frac{S_t}{t}\,dt. \\ &= -\frac{A_t}{t}\,dt + \frac{S_t}{t}\,dt \\ &= \frac{(S_t - A_t)}{t}dt \end{aligned}$$

Therefore:

$$dV = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}[rS_t dt + \sigma S_t dW_t] + \frac{\partial V}{\partial A}\frac{(S_t - A_t)}{t}dt + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 V}{\partial S^2}dt$$

$$\left(\frac{\partial V}{\partial t} + rS_t\frac{\partial V}{\partial S} + \frac{(S_t - A_t)}{t}\frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 V}{\partial S^2}\right)dt + \left(\sigma S_t\frac{\partial V}{\partial S}\right)dW_t$$

Note: as the Brownian term is the same as before we can construct the same delta hedge strategy with $\Delta = -\frac{\partial V}{\partial S}$.

Therefore we arrive at the following:

$$dV + d\Pi = \left(\frac{\partial V}{\partial t} + rS_t\frac{\partial V}{\partial S} + \frac{(S_t - A_t)}{t}\frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 V}{\partial S^2} - rS_t\frac{\partial V}{\partial S} + \alpha r B_t\right)dt$$

$$= \left(\frac{\partial V}{\partial t} + \frac{(S_t - A_t)}{t} \frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + \alpha r B_t \right) dt$$

Applying the same arbitrage-free condition as before.

$$\left(\frac{\partial V}{\partial t} + \frac{(S_t - A_t)}{t} \frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + \alpha r B_t \right) dt = \left(V - \frac{\partial V}{\partial S} S_t + \alpha B_t \right) r dt$$

Which gives us the final PDE for an Asian option:

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

For an easier calculation it is also possible to use the running sum of the stock price I_t instead of A_t.

$$I_t = \int_0^t S_u du$$

$$dI_t = S_t dt$$

However, this would change the payoff from:

$$(A_T - K)^+$$

to

$$\left(\frac{I_T}{T} - K \right)^+$$

Giving us the PDE for $V(S_t, I_t, t)$

$$\frac{\partial V}{\partial t} + S \frac{\partial V}{\partial I} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} - rV = 0$$

5.0.4 Asian Call with Dividends

Working with the new dynamics for the underlying asset S_t, incorporating dividends q, and the Bank account, B_t

$$dS_t = (r - q) S_t dt + \sigma S_t dW_t$$

$$dB_t = rB_t dt$$

We can construct the new PDE of the Asian option.

Recall from Itô's lemma:

$$dV = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}dS_t + \frac{\partial V}{\partial A}dA_t + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}dS_t^2$$

and substituting the new dynamics dS_t:

$$dV = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}[(r-q)S_t dt + \sigma S_t dW_t] + \frac{\partial V}{\partial A}\frac{(S_t - A_t)}{t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\sigma^2 S_t^2 dt$$

$$\left(\frac{\partial V}{\partial t} + (r-q)S_t\frac{\partial V}{\partial S} + \frac{(S_t - A_t)}{t}\frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 V}{\partial S^2}\right)dt + \left(\sigma S_t\frac{\partial V}{\partial S}\right)dW_t$$

We construct the replicating portfolio slightly differently:

$$\Pi = \Delta S_t + \alpha B_t + \Delta q$$

where Δq represents the total amount received from Δ shares paying q dividends.

$$d\Pi = \Delta dS_t + \alpha dB_t + \Delta q dt$$

$$= \Delta[(r-q)S_t + \sigma S_t dW_t] + \alpha r B_t dt + \Delta q dt$$

$$= (\Delta(r-q)S_t + \alpha r B_t + \Delta q)\,dt + \Delta\sigma S_t dW_t$$

Here, $\Delta = -\frac{\partial V}{\partial S}$ again.

Substituting Δ into the total portfolio:

$$dV + d\Pi = \left(\frac{\partial V}{\partial t} + (r-q)S_t\frac{\partial V}{\partial S} + \frac{(S_t - A_t)}{t}\frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 V}{\partial S^2} - (r-q)S_t\frac{\partial V}{\partial S} + \alpha r B_t + q\frac{\partial V}{\partial S}\right)dt$$

$$\left(\frac{\partial V}{\partial t} + \frac{(S_t - A_t)}{t}\frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + \alpha r B_t + q\frac{\partial V}{\partial S}\right) dt$$

Using the same arbitrage-free condition:

$$d(V + \Pi) = (V + \Pi) r dt$$

We obtain:

$$\left(\frac{\partial V}{\partial t} + \frac{(S_t - A_t)}{t}\frac{\partial V}{\partial A} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + \alpha r B_t + q\frac{\partial V}{\partial S}\right) dt = \left(V - \frac{\partial V}{\partial S} S_t + \alpha B_t\right) r dt$$

Which gives the final PDE for an Asian option with dividends:

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

It is also possible to adopt another change of variables $\tau = T - t$ which allows the PDE to be solved forward in time, rather than backwards recursively. However, this doesn't enhance the performance of the scheme and therefore will not be used.

5.1 Solving the PDE - Finite Difference

We aim to solve the following partial differential equation (PDE) using the Crank-Nicholson approach. The PDE is given by:

$$\frac{\partial V}{\partial t} + S\frac{\partial V}{\partial I} + \frac{1}{2}\sigma^2 S_t^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0,$$

where $V(S_t, I_t, t)$ is the option value, S_t is the stock price at time t, I_t is the running sum of the stock price, r is the risk-free interest rate, and σ is the volatility of the stock price.

To implement the Crank-Nicholson scheme, we discretize the time, stock price, and the running sum variables. Let:

$$t_n = n\Delta t, \quad S_j = j\Delta S, \quad I_k = k\Delta I,$$

where n, j, and k are indices corresponding to the time, stock price, and running sum, respectively.

The Crank-Nicholson method involves averaging the explicit and implicit finite difference methods, which gives us the following approximation:

$$\frac{\partial V}{\partial t} \approx \frac{V_j^{n+1} - V_j^n}{\Delta t},$$

$$\frac{\partial V}{\partial S} \approx \frac{V_{j+1}^n - V_{j-1}^n}{2\Delta S},$$

$$\frac{\partial^2 V}{\partial S^2} \approx \frac{V_{j+1}^n - 2V_j^n + V_{j-1}^n}{\Delta S^2},$$

$$\frac{\partial V}{\partial I} \approx \frac{V_{k+1}^n - V_{k-1}^n}{2\Delta I}.$$

Applying the Crank-Nicholson method to the PDE, we get:

$$\frac{V_j^{n+1} - V_j^n}{\Delta t} + \frac{S_j}{2}\left(\frac{V_{j+1}^{n+1} - V_{j-1}^{n+1}}{2\Delta S} + \frac{V_{j+1}^n - V_{j-1}^n}{2\Delta S}\right)$$

$$+\frac{\sigma^2 S_j^2}{4}\left(\frac{V_{j+1}^{n+1} - 2V_j^{n+1} + V_{j-1}^{n+1}}{\Delta S^2} + \frac{V_{j+1}^n - 2V_j^n + V_{j-1}^n}{\Delta S^2}\right)$$

$$+\frac{rS_j}{2}\left(\frac{V_{j+1}^{n+1} - V_{j-1}^{n+1}}{2\Delta S} + \frac{V_{j+1}^n - V_{j-1}^n}{2\Delta S}\right) - r\frac{V_j^{n+1} + V_j^n}{2} = 0.$$

Note: this is an adaptation of the normal finite difference scheme utilising a weight parameter $\theta = 0.5$ to balance explicit and implicit components.

This equation can be rearranged into a tridiagonal system of equations that can be solved iteratively at each time step to determine the value of the option at each grid point (S_j, I_k, t_n).

Initial condition: At maturity $t = T$, the option payoff is given by

$$V(S_T, I_T, T) = \left(\frac{I_T}{T} - K\right)^+.$$

The first step to solve this system is to set up a grid of S, I and t values. Then initialise the option value through the initial conditions, iterating backwards using the Crank Nicolson scheme, returning the numerical solution for $V(S_t, I_t, t)$.

However, there are a lot of complications with this method:

To apply the initial condition, we require the value of the running sum I_T, which of course is unknown at time $t = 0$. One way to address this uncertainty is to approximate I_T using the current stock price S_0. However, this is only an approximation and may introduce bias into the model.

An alternative approach is to simulate multiple potential paths of the underlying asset price and calculate the expected value of I_T. This method allows us to estimate more accurately by capturing the range of possible future behaviors of the asset. However, this involves a numerical Monte Carlo scheme, which we are trying to avoid in favour of a PDE approach.

That being said, the main issue with the finite difference scheme is that it is vulnerable to severe oscillations, which is touched on in [10].

Another, more common, approach is to use various substitutions to simplify the PDE further. According to current literature, the main substitutions are from Ingersoll, (Wilmott, Dewynne & Howison) [11] and Rogers & Shi's [12]. However, the first two are only effective for floating strike options, therefore we will focus on Rogers & Shi's martingale approach.

5.2 Martingale approach

This section focuses on deriving a PDE to price Asian options, utilizing a Martingale approach.

Recall that under standard assumptions; assets are traded continuously, no transactions costs and no taxes alongside the underlying risk-neutral price dynamics:

$$dS_t = rS_t dt + \sigma S_t dW_t$$

We can express the payoff as follows:

$$\left(\int_0^T S_u \mu \; (du) - K \right)^+$$

where μ is a probability measure with density ρ that determines how the prices are weighted throughout the averaging period. In the case of an Arithmetic Asian Call with fixed strike price K, we define μ as a uniform probability measure on $[0, T]$ giving $\rho_t = \frac{1}{T}$.

5.2.1 Constructing the Martingale

We define the function $\phi(t, x)$ as the expected payoff and construct a Martingale M_t on all the information up to time t.

$$\phi(t, K) = \mathbb{E}\left[\left(\int_t^T S_u \, \mu(du) - K \right)^+ \right]$$

$$M_t = \mathbb{E}\left[\left(\int_0^T S_u \, \mu(du) - K \right)^+ \Big| \mathcal{F}_t \right]$$

Now, developing the Martingale:

$$M_t = \mathbb{E}\left[\left(\int_t^T S_u\mu\ (du) + \int_0^t S_u\,\mu(du) - K\right)^+ \Big|\mathcal{F}_t\right]$$

$$= \mathbb{E}\left[\left(\int_t^T S_u\,\mu(du) - \left(K - \int_0^t S_u\mu\ (du)\right)\right)^+ \Big|\mathcal{F}_t\right]$$

Using the change of Numeraire technique where we use the underlying asset S_t as the Numeraire, we can write:

$$= S_t\mathbb{E}\left[\left(\int_t^T \frac{S_u}{S_t}\,\mu(du) - \left(\frac{K - \int_0^t S_u\mu\ (du)}{S_t}\right)\right)^+ \Big|\mathcal{F}_t\right]$$

Which we recognise as:

$$M(t) = S_t\phi(t,x)$$

where x is defined by:

$$x = \frac{K - \int_0^t S_u\mu\ (du)}{S_t}$$

To derive the dynamics of x_t, we apply Itô's Lemma:

First, we find the dynamics $\frac{1}{S_t}$

$$d\left(\frac{1}{S_t}\right) = \left(-\frac{1}{S_t^2}\right)dS_t + \left(\frac{1}{S_t^3}\right)d\langle S\rangle_t$$

$$= \left(-\frac{1}{S_t^2}\right)\left[rS_tdt + \sigma S_tdW_t\right] + \left(\frac{1}{S_t^3}\right)\sigma^2S_t^2dt$$

$$= \left(-\frac{1}{S_t}\right)\left[rdt + \sigma dW_t\right] + \left(\frac{1}{S_t}\right)\sigma^2dt$$

Now using Itô's product rule:

$$dx_t = \left(K - \int_0^t S_u\mu\ (du)\right)d\left(\frac{1}{S_t}\right) + \left(\frac{1}{S_t}\right)d\left(K - \int_0^t S_u\mu\ (du)\right) + \langle\left(K - \int_0^t S_u\mu\ (du)\right), \left(\frac{1}{S}\right)\rangle_t$$

Note here that K is a constant and therefore of finite variation and $\int_0^t S_u\mu(du)$ is also of finite variation, therefore reducing the quadratic co-variation $\rightarrow 0$.

$$dx_t = \left(K - \int_0^t S_u\mu\ (du)\right)\left(-\frac{1}{S_t}[rdt + \sigma dW_t - \sigma^2 dt]\right) + \left(\frac{1}{S_t}\right)(-S_t\rho_t)\,dt$$
$$= -\rho_t dt + x[-rdt - \sigma dW_t + \sigma^2 dt]$$

Now applying Itô's Lemma :

$$dM_t = S_t d(\phi(t,x)) + \phi(t,x)d(S_t) + \langle S, \phi(...,x)\rangle_t$$
$$= S_t\left[\dot{\phi} + \phi' dx + \frac{1}{2}\phi'' d\langle x\rangle_t\right] + \phi(t,x)[rS_t dt + \sigma S_t dW_t] - \sigma S_t \phi' \sigma x dt$$
$$= \phi r S_t dt + S_t\left[\dot{\phi} + \phi'(-\rho_t - xr + x\sigma^2) + \frac{1}{2}\sigma^2 x^2\right] dt - \sigma^2 S_t \phi' x dt$$
$$= S_t\left[r\phi + \dot{\phi} - (\rho_t + rx)\phi' + \frac{1}{2}\sigma^2 x^2\right] dt$$

Since M_t is a martingale, we have the following PDE:

$$0 = \dot{\phi} + r\phi + \frac{1}{2}\sigma^2\xi^2\phi'' - (\rho_t + r\xi)\phi$$

5.2.2 Transforming the PDE

To help solve the PDE, we apply the following transformation:

$$f(t,x) = e^{-r(T-t)}\phi(t,x).$$

we find that f solves:

$$\dot{f} + Gf = 0$$

where G is the differential operator:

$$G = \frac{1}{2}\sigma^2 x^2 \frac{\partial^2}{\partial x^2} - (\rho_t + rx)\frac{\partial}{\partial x}$$

The boundary conditions are relatively simple to derive for a fixed strike option.

$$f(T,x) = e^0 \phi(T,x)$$

$$= (-x)^+$$

$$f(T,x) = max(-x,0)$$

The PDE derived above is relatively simple and can be solved numerically. Now looking at the price of the Asian option:

$$e^{-rT}\mathbb{E}\left[\frac{1}{T}\int_0^T (S_u - K)\,du\right] = S_0 f(0, K/S_0).$$

Note here that we have substituted $\rho = \frac{1}{T}$

5.3 Solving the PDE - Martingale

Solving this PDE using a finite difference method encounters severe oscillations. Which we will treat by introducing the Van Leer flux limiter.
As mentioned previously, we can convert the PDE to a forward equation by substituting t with $\tau = T - t$. We obtain the following general form equation:

$$\frac{V_i^{n+1} - V_i^n}{\Delta\tau} = \theta F_{i-\frac{1}{2}}^{n+1} - \theta F_{i+\frac{1}{2}}^{n+1} + (1-\theta)F_{i-\frac{1}{2}}^{n} - (1-\theta)F_{i+\frac{1}{2}}^{n} + (1-\theta)f_i^n,$$

Note here that $\theta = 0.5$ is the Crank Nicolson scheme as before.
The flux functions are defined as:

$$F_{i-\frac{1}{2}}^{n+1} = \frac{1}{\Delta S_i}\left[\left(-\frac{1}{2}\sigma^2 S_i^2\right)\frac{V_i^{n+1} - V_{i-1}^{n+1}}{\Delta S_{i-\frac{1}{2}}} + (-rS_i)V_{i-\frac{1}{2}}^{n+1}\right],$$

$$F_{i+\frac{1}{2}}^{n+1} = \frac{1}{\Delta S_i}\left[\left(-\frac{1}{2}\sigma^2 S_i^2\right)\frac{V_{i+1}^{n+1} - V_i^{n+1}}{\Delta S_{i+\frac{1}{2}}} + (\rho^{n+1} + rS_i)V_{i+\frac{1}{2}}^{n+1}\right],$$

and the source term is:

$$f_i^{n+1} = 0.$$

The flux functions allow for non-uniform grid spacing, which enables efficient numerical computation by using finer grids near the exercise price and coarser grids farther away.
To handle the convective term $V_{i+\frac{1}{2}}^{n+1}$, we use a central weighting scheme:

$$V_{i+\frac{1}{2}}^{n+1} = \frac{V_{i+1}^{n+1} + V_i^{n+1}}{2},$$

To prevent severe oscillations from occurring, we must satisfy the Peclet conditions:

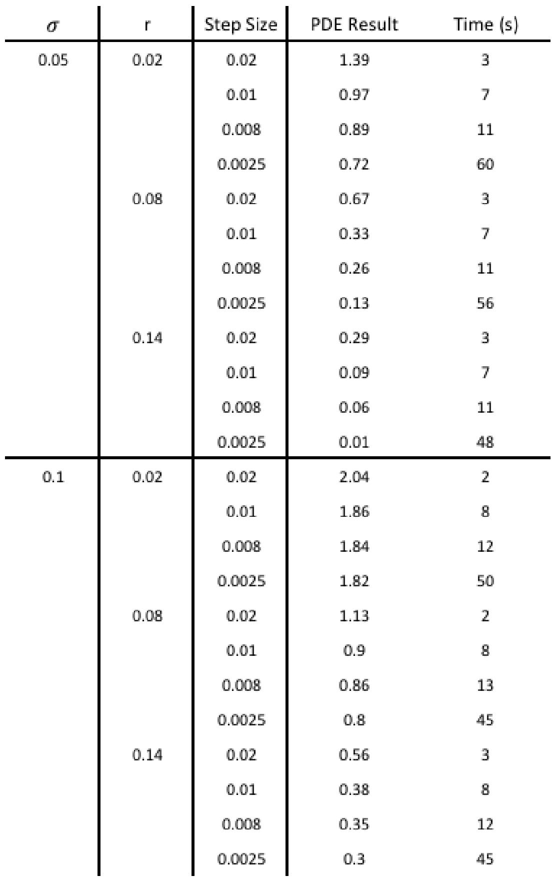
$$\frac{1}{\Delta S_{i-\frac{1}{2}}} > \frac{r}{\sigma^2 S_i},$$

and the additional condition:

$$\frac{1}{(1-\theta)\Delta\tau} > \frac{\sigma^2 S_i^2}{2}\left(\frac{1}{\Delta S_{i-\frac{1}{2}}\Delta S_i} + \frac{1}{\Delta S_{i+\frac{1}{2}}\Delta S_i}\right) + r,$$

5.3.1 Results

From Rogers and Shi's, we can obtain the following results:



σ	r	Step Size	PDE Result	Time (s)
0.05	0.02	0.02	1.39	3
		0.01	0.97	7
		0.008	0.89	11
		0.0025	0.72	60
	0.08	0.02	0.67	3
		0.01	0.33	7
		0.008	0.26	11
		0.0025	0.13	56
	0.14	0.02	0.29	3
		0.01	0.09	7
		0.008	0.06	11
		0.0025	0.01	48
0.1	0.02	0.02	2.04	2
		0.01	1.86	8
		0.008	1.84	12
		0.0025	1.82	50
	0.08	0.02	1.13	2
		0.01	0.9	8
		0.008	0.86	13
		0.0025	0.8	45
	0.14	0.02	0.56	3
		0.01	0.38	8
		0.008	0.35	12
		0.0025	0.3	45

Figure 24: *Rogers and Shi's PDE Results*

From this table, it's clear to see that the time taken to return an accurate result is far longer than both the Laplace and Moment Matching approaches. Although it is faster than the Monte Carlo method, the time required to achieve a respectable level of accuracy makes it less favorable than both the Laplace and Moment Matching approaches. Therefore, we will not investigate this further

6 Conclusion

Throughout this paper, I have derived and presented multiple methods for pricing an arithmetic Asian option, presenting their results and both advantages and disadvantages.

For numerical methods, we conclude that using a geometric Asian option as a control variate significantly increases the accuracy, thus reducing both the number of simulations and the time required to price the option. This approach still is a great method to price an Asian option and can be seen as a good starting point to price path-dependant options.

The Laplace Transform is a much more complicated approach to pricing this derivative, however, it does return accurate and extremely efficient prices when compared to the Monte Carlo approach, for the most part. This approach however, does struggle pricing options with $\sigma < 0.1$ and $T < 0.5$ efficiently. In conclusion this approach offers a superior alternative for pricing Asian options in both accuracy and speed and should be used indefinitely apart from cases with $\sigma < 0.1$ and $T < 0.5$, where a Monte Carlo simulation could be used. However, when looking at the moment matching approach, we find that this method returns accurate prices incredibly quick, especially for options with small volatility and short time to maturities. Although this method does not achieve the same level of accuracy as the Laplace Transform, it still offers a respectable accuracy for these specific options and is very easy to understand and implement. Consequently, it can be effectively used as an alternative to the Laplace Transform and Monte Carlo methods for options with low volatility and short time to maturities.

The PDE approach contains several challenges from severe oscillations to convergence issues and requires a high level of expertise in numerical methods to price effectively. To start with, the PDE involves multiple dimensions which requires certain substitutions to reduce dimensionality. These substitutions can affect the boundary conditions and often require approximations of the running sum. Ensuring both accuracy and numerical stability in solving the transformed PDE still demands sophisticated techniques and increased computational time. Furthermore, the majority of methods in the literature focus on floating strike options. While these methods are easier to implement and price, floating strike options are traded far less frequently compared to arithmetic options, limiting their practical application.

In conclusion, the Laplace Transform approach proves to be a superior alternative for pricing arithmetic Asian options whilst using the moment matching approach for options with small volatility and short time to maturities.

References

[1] Boyle, P., 1977. Options: A Monte Carlo Approach. *Journal of Financial Economics*, 4(3), pp.323-338.

[2] Glasserman, P., 2004. *Monte Carlo Methods in Financial Engineering.* Springer.

[3] Kemn, M. and Vorst, T., 1990. Option pricing and hedging using the simulation method. *Journal of Financial and Quantitative Analysis*, 25(3), pp. 323–338.

[4] Fusai, G., 2000. *Applications of Laplace Transform for Evaluating Occupation Time Options and Other Derivatives.*

[5] Dufresne, D., 2004. *Bessel Processes and a Functional of Brownian Motion.*

[6] Hélyette Geman, M. Y., 1993. Bessel Processes, Asian Options, and Perpetuities. *Mathematical Finance*, 3(4), pp. 349–375.

[7] Yor, M., 2001. *Exponential Functionals of Brownian Motion and Related Processes.* New York: Springer-Verlag.

[8] Levy, E., 1992. Pricing European Average Rate Currency Options. *Journal of International Money and Finance*, 14, pp. 474–491.

[9] Rouah, F. D., 2009. Four Derivations of the Black-Scholes PDE.

[10] Rehurek, A., 2011. *Stable Numerical Methods for PDE Models of Asian Options.*

[11] Wilmott, P., Dewynne, J., and Howison, S., 1993. *Option Pricing: Mathematical Models and Computation.* Oxford Financial Press, Oxford, UK.

[12] Rogers, L. C. G. and Shi, Z., 1995. The Value of an Asian Option. *Journal of Applied Probability*, 32, pp. 1077–1088.

[13] Ruggeri, R., 2015. *The Laplace Transform in Option Pricing.*

[14] Mkhize, N. G. Z., 2007. *The Pricing Theory of Asian Options.*

7 Appendix A - Key Mathematics

Whilst this chapter may not be necessary for all readers, it contains some important fundamental mathematical concepts which are referenced and used throughout this paper.

7.1 Itô's Lemma (One Dimensional)

Theorem A7.1 Let $X = (X_t)_{t\geq 0}$ be a continuous semimartingale on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F} = (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ satisfying the usual conditions, and let $f : \mathbb{R} \to \mathbb{R}$ be twice continuously differentiable. Then $f(X) = (f(X_t))_{t\geq 0}$ is again a continuous semimartingale and we have Itô's formula:

$$f(X_t) = f(X_0) + \int_0^t f'(X_s)\, dX_s + \frac{1}{2}\int_0^t f''(X_s)\, d\langle X\rangle_s \quad \mathbb{P}\text{-a.s. for each } t \geq 0.$$

Itô's formula is often written more compactly in differential form as:

$$df(X_t) = f'(X_t)\, dX_t + \frac{1}{2} f''(X_t)\, d\langle X\rangle_t$$

7.2 Itô's Lemma (Several Dimensions)

Theorem A7.2 Let $X = (X_t^1, \ldots, X_t^d)_{t\geq 0}$ be an $\mathbb{R}^d$-valued continuous semimartingale on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F} = (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ satisfying the usual conditions, and let $f : \mathbb{R}^d \to \mathbb{R}$ be twice continuously differentiable. Then $f(X) = (f(X_t))_{t\geq 0}$ is again a continuous semi-martingale and we have Itô's formula:

$$f(X_t) = f(X_0) + \sum_{i=1}^d \int_0^t \frac{\partial f}{\partial x_i}(X_s)\, dX_s^i + \frac{1}{2}\sum_{j=1}^d \sum_{k=1}^d \int_0^t \frac{\partial^2 f}{\partial x_j \partial x_k}(X_s)\, d\langle X^j, X^k\rangle_s$$

7.3 Product Rule

Theorem A7.3 As an extension of Itô's multi-dimensional formula, in two dimensions, we ca write the so-called product rule.

$$X_tY_t = X_0Y_0 + \int_0^t Y_u dX_u + \int_0^t X_u dY_u + \langle X, Y\rangle_t$$

More commonly written in differential form:

$$d(X_tY_t) = X_t dY_t + Y_t dX_t + d\langle X, Y\rangle_t$$

7.4 Stochastic Exponential

Theorem A7.4 Let $X = (X_t)_{t\geq 0}$ be a continuous semimartingale on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{F} = (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ satisfying the usual conditions. Then the stochastic differential equation (SDE)

$$dZ_t = Z_t\, dX_t$$

has a unique solution Z given by

$$Z_t := Z_0 \exp\left(X_t - X_0 - \frac{1}{2}\langle X\rangle_t\right) \quad t \geq 0$$

This solution is called the *stochastic exponential* of X and is denoted by $\mathcal{E}(X) = (\mathcal{E}(X)_t)_{t\geq 0}$.

7.5 Itô Isometry

Theorem A7.5 Let W_t be a standard Brownian motion on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, and let $H = (H_t)_{t\geq 0}$ be an adapted process where $\mathbb{E}\left[\int_0^T H_t^2\, dt\right] < \infty$ for some $T > 0$. Then the Itô integral $I_T = \int_0^T H_t\, dW_t$ satisfies the isometry property:

$$\mathbb{E}\left[\left(\int_0^T H_t\, dW_t\right)^2\right] = \mathbb{E}\left[\int_0^T H_t^2\, dt\right].$$

7.6 Stock Dynamics (GBM)

Theorem A7.6 Let S_t denote the price of a stock at time t. We model S_t using Geometric Brownian Motion (GBM), which is described by the following stochastic differential equation (SDE):

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t,$$

$\mu =$ drift term representing the rate of return
$\sigma =$ volatility term
However, under a risk neutral measure, the SDE becomes:

$$dS_t = rS_t dt + \sigma S_t dW_t^Q$$

Where the solution to this SDE can be found using the stochastic exponential $\mathcal{E}(X)$:

$$S_t = S_0 \times exp\left\{\left(r - \frac{1}{2}\sigma^2\right) t + \sigma W_t\right\}$$

7.7 Martingale

Theorem A7.7 Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, and let $\mathbb{F} = (\mathcal{F}_t)_{t \geq 0}$ be a filtration, which is a family of nested σ-algebras $\mathcal{F}_t$ where $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \cdots$. A stochastic process $(M_t)_{t \geq 0}$ is called a *martingale* with respect to the filtration $\mathbb{F}$ and the probability measure $\mathbb{P}$ if the following conditions are satisfied:

(i) M_t is $\mathcal{F}_t$-adapted for each $t \geq 0$.

(ii) M_t is integrable, i.e., $\mathbb{E}[|M_t|] < \infty$ for each $t \geq 0$.

(iii) The martingale property: For all $0 \leq s \leq t$, we have

$$\mathbb{E}[M_t \mid \mathcal{F}_s] = M_s \quad \text{almost surely.}$$

In other words, the conditional expectation of M_t given the past information up to time s (i.e., $\mathcal{F}_s$) is equal to the value of M_s, reflecting that the process M_t has no tendency to drift upwards or downwards over time, given the past information.

7.8 Quadratic Variation

Theorem A7.8 Let $X = (X_t)_{t \geq 0}$ be a continuous process on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ with filtration $\mathbb{F} = (\mathcal{F}_t)_{t \geq 0}$. The *quadratic variation* of X over the interval $[0, t]$ is denoted by $\langle X \rangle_t$ and is defined as the unique increasing, continuous process such that:

$$\langle X \rangle_t = \lim_{|\pi| \to 0} \sum_{i=1}^{n} \left(X_{t_i} - X_{t_{i-1}} \right)^2$$

where the limit is taken over all partitions π of the interval $[0, t]$ into n subintervals, and $|\pi|$ denotes the mesh of the partition (i.e., the length of the largest subinterval).

Key Properties:

(i) If the process X has finite variation over $[0, t]$, then its quadratic variation is zero:

$$\langle X \rangle_t = 0 \quad \text{for all } t \geq 0.$$

This is because a process of finite variation can be decomposed into a difference of two increasing processes, and thus, it does not have any "jumps" that contribute to the quadratic variation.

(ii) If X is a continuous semimartingale, then its quadratic variation provides a measure of the "roughness" or "volatility" of the process. For example, for a Brownian motion W_t, we have $\langle W \rangle_t = t$.

7.9 Quadratic Covariation

Theorem A7.9 Let $X = (X_t)_{t\geq 0}$ and $Y = (Y_t)_{t\geq 0}$ be two continuous processes on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ with filtration $\mathbb{F} = (\mathcal{F}_t)_{t\geq 0}$. The *covariation* of X and Y over the interval $[0, t]$ is denoted by $\langle X, Y\rangle_t$ and is defined as the unique process such that:

$$\langle X, Y\rangle_t = \lim_{|\pi|\to 0} \sum_{i=1}^{n} \left(X_{t_i} - X_{t_{i-1}}\right)\left(Y_{t_i} - Y_{t_{i-1}}\right),$$

where the limit is taken over all partitions π of the interval $[0, t]$ into n subintervals, and $|\pi|$ denotes the mesh of the partition (i.e., the length of the largest subinterval).

If X and Y are processes of finite variation, then their covariation is zero:

$$\langle X, Y\rangle_t = 0 \quad \text{for all } t \geq 0.$$

This follows from the fact that processes of finite variation can be decomposed into differences of increasing processes, and thus the product of the increments of two such processes has a vanishing limit.

7.10 Laplace Transform

Theorem A7.10 The Laplace transform is an integral transform used to convert a function of time $f(t)$ into a function of a complex variable s. For a function $f(t)$ defined for $t \geq 0$, the Laplace transform $\mathcal{L}\{f(t)\}(s)$ is given by:

$$\mathcal{L}\{f(t)\}(s) = \int_0^\infty e^{-st} f(t)\, dt,$$

where s is a complex number with $\mathrm{Re}(s) > \sigma$, and σ is a real number such that the integral converges.

Properties:

(i) **Linearity:** The Laplace transform is a linear operator. If $f(t)$ and $g(t)$ are functions with Laplace transforms $F(s)$ and $G(s)$, respectively, then for any constants a and b:

$$\mathcal{L}\{af(t) + bg(t)\}(s) = a\mathcal{L}\{f(t)\}(s) + b\mathcal{L}\{g(t)\}(s).$$

(ii) **Derivative:** If $f(t)$ is differentiable with $f'(t)$ having a Laplace transform, then:

$$\mathcal{L}\{f'(t)\}(s) = s\mathcal{L}\{f(t)\}(s) - f(0).$$

(iii) **Integration:** If $f(t)$ is integrable, then:

$$\mathcal{L}\left\{\int_0^t f(u)\,du\right\}(s) = \frac{1}{s}\mathcal{L}\{f(t)\}(s).$$

(iv) **Inverse Laplace Transform:** The inverse Laplace transform is given by:

$$f(t) = \mathcal{L}^{-1}\{F(s)\}(t) = \frac{1}{2\pi i}\int_{\sigma-i\infty}^{\sigma+i\infty} e^{st}F(s)\,ds,$$

where the integral is taken along a vertical line in the complex plane.

7.11 The Law of $A_t(v)$ at an Independent Exponential Time

We start by introducing a key definition necessary for demonstrating Yor's theorem.

Theorem A7.11: Resolvent of a Markov Process

Consider a time-homogeneous Markov process X under the objective measure $\mathbb{P}$, defined by the stochastic differential equation:

$$dX(t) = \mu X(t)\,dt + \sigma X(t)\,dW(t),$$

where μ and σ are given functions, and $W(t)$ is a standard Brownian motion. Let g be a real-valued function that is sufficiently integrable. For $\alpha \geq 0$, the resolvent operator is defined by:

$$[R_\alpha g](x) = \mathbb{E}_x^{\mathbb{P}} \left[\int_0^\infty e^{-\alpha s} g(X_s)\, ds \right],$$

where the subscript x denotes the initial condition $X_0 = x$.

We recognize that the expression inside the brackets is equivalent to the definition of the Laplace Transform given in Chapter 2 (see Definition 5.2). In our case, $g(X_s) = A_t(v)$. Furthermore, the definition of the resolvent can be rewritten as:

$$\mathbb{E}_x^{\mathbb{P}} \left[\int_0^\infty e^{-\alpha s} g(X_s)\, ds \right] = \frac{1}{\lambda} \mathbb{E}_x^{\mathbb{P}} \left[\int_0^\infty \lambda e^{-\alpha s} g(X_s)\, ds \right] = \frac{1}{\lambda} \mathbb{E}_x^{\mathbb{P}} [X_{S_\lambda}],$$

where S_λ and the process X_t are independent.

The goal is to determine the density of the resolvent, which is equivalent to finding the distribution of X_{S_λ}, and specifically $A_{S_\lambda}(v)$.

In Yor2001, Yor proved that $A_{S_\lambda}(v)$ is distributed as a combination of two well-known functions: the gamma and beta distributions. This result is stated more formally in the following theorem.

7.12 Yor's Theorem

Theorem A7.12 The distribution of $A_t(v)$ is characterized by:

$$2A_t(v) \sim \frac{Beta(1, \alpha)}{Gamma(\beta, 1)},$$

where Beta $\sim$ Beta$(1, \alpha)$ and Gamma $\sim$ Gamma$(\beta, 1)$ are independent, with the parameters:

$$\gamma = \sqrt{2\lambda + v^2},$$

$$\alpha = \frac{v + \gamma}{2},$$

$$\beta = \frac{\gamma - v}{2}.$$

A complete proof of this theorem can be found in . This proof is a simplified version of the one provided by Yor , benefiting from the probability density function and properties of the Bessel process.

7.13 Fubini Theorem

Theorem A7.13 Let $(X, \mathcal{A}, \mu)$ and $(Y, \mathcal{B}, \nu)$ be two σ-finite measure spaces. Suppose $f : X \times Y \to \mathbb{R}$ is a measurable function such that

$$\int_X \left(\int_Y |f(x, y)| \, d\nu(y) \right) d\mu(x) < \infty.$$

Then f is integrable over $X \times Y$, and

$$\int_{X\times Y} f(x, y) \, d(\mu\times\nu)(x, y) = \int_X \left(\int_Y f(x, y) \, d\nu(y) \right) d\mu(x) = \int_Y \left(\int_X f(x, y) \, d\mu(x) \right) d\nu(y).$$

In other words, if the integral of the absolute value of f is finite, the order of integration can be interchanged.

7.14 Moment Matching

Theorem A7.14 Moment matching is a technique used to approximate a probability distribution by matching its moments (mean, variance, etc.)

with those of another distribution.

To match the moments, we aim to find a distribution that has the same mean and variance as the original distribution. For example, if we have a random variable X with mean μ and variance σ^2, we can approximate X with a normal distribution Y that also has mean μ and variance σ^2.

Moment Generating Function (MGF):
The moment generating function (MGF) of a normal distribution $\mathcal{N}(\mu, \sigma^2)$ is:

$$M_Y(s) = \exp\left(\mu s + \frac{1}{2}\sigma^2 s^2\right).$$

The MGF provides a way to generate moments of the distribution, and matching MGFs can help ensure that the moments of the approximating distribution match those of the original distribution.

8 Appendix B - Python Code

8.1 Geometric Asian Option Closed Form

```
def geometric_asian_call_price(S0, K, T, r, sigma,q):

    d1 = (np.log(S0 / K) + (T / 2) * ((r-q) - (sigma**2 / 2) )
        + (sigma**2 * T / 3)) / (np.sqrt(sigma**2 * T / 3))
    d2 = d1 - (sigma * np.sqrt(T / 3))

    price = np.exp(-r * T) * (S0 * np.exp(((r-q) - (0.5 * sigma
        **2)) * T / 2 + (sigma**2 * T / 6)) * norm.cdf(d1) - (K
         * norm.cdf(d2)))


    return price
```

8.2 Geometric Asian Option Monte Carlo

```
    def geometric_asian_call_monte_carlo(S0, K, T, r, sigma,
        n_simulations, n_steps):



    dt = T / n_steps
    discount_factor = np.exp(-r * T)
    payoffs = np.zeros(n_simulations)

    for i in range(n_simulations):
        prices = np.zeros(n_steps + 1)
        prices[0] = S0
        for t in range(1, n_steps + 1):
            z = np.random.standard_normal()
            prices[t] = prices[t-1] * np.exp((r - 0.5 * sigma
                **2) * dt + sigma * np.sqrt(dt) * z)

        geometric_mean = np.exp(np.mean(np.log(prices[1:])))

        payoffs[i] = max(0, geometric_mean - K)


    option_price = discount_factor * np.mean(payoffs)
    se = np.std(payoffs*discount_factor, ddof=1) / np.sqrt(
        n_simulations)


    return option_price, se
```

8.3 Arithmetic Asian Option Monte Carlo

```
    def arithmetic_asian_call_monte_carlo_with_div(S0, K, T, r,
         sigma,q, n_simulations, n_steps):

    dt = T / n_steps
    discount_factor = np.exp(-r * T)
    payoffs = np.zeros(n_simulations)

    for i in range(n_simulations):
        prices = np.zeros(n_steps + 1)
        prices[0] = S0
        for t in range(1, n_steps + 1):
            z = np.random.standard_normal()
            prices[t] = prices[t-1] * np.exp(((r-q) - 0.5 *
                sigma**2) * dt + sigma * np.sqrt(dt) * z)

        arithmetic_mean = np.mean(prices[1:])
        payoffs[i] = max(0, arithmetic_mean - K)

    option_price = discount_factor * np.mean(payoffs)
    se = np.std(payoffs*discount_factor, ddof=1) / np.sqrt(
        n_simulations)

    return option_price, se
```

8.4 Arithmetic Asian Option with Control Variate

```
def arithmetic_approx_from_geometric(S0, K, T, r, sigma,q,
        n_simulations, n_steps):
    dt = T / n_steps
    discount_factor = np.exp(-r * T)
    diff = np.zeros(n_simulations)
    arith_payoff = np.zeros(n_simulations)
    geo_payoff = np.zeros(n_simulations)

    for i in range(n_simulations):
        prices = np.zeros(n_steps + 1)
        prices[0] = S0
        for t in range(1, n_steps + 1):
            z = np.random.standard_normal()
            prices[t] = prices[t-1] * np.exp(((r-q) - 0.5 *
                sigma**2) * dt + sigma * np.sqrt(dt) * z)

        geometric_mean = np.exp(np.mean(np.log(prices[1:])))
        arithmetic_mean = np.mean(prices[1:])

        arith_payoff[i] = max(0, arithmetic_mean - K)
        geo_payoff[i] = max(0, geometric_mean - K)
        diff[i] = arith_payoff[i] - geo_payoff[i]
```

```
av_diff = np.mean(diff)
geometric_asian_price = geometric_asian_call_price(S0, K, T
    , r, sigma,q)

cov_xy = np.cov(arith_payoff, geo_payoff)[0, 1]
var_y = np.var(geo_payoff)
b = cov_xy / var_y

adjusted_arith_price = (discount_factor * (np.mean(
    arith_payoff))) - b * ((discount_factor * np.mean(
    geo_payoff)) - geometric_asian_price)

se = np.std(discount_factor * (arith_payoff) - b * ((
    discount_factor * (geo_payoff)) - geometric_asian_price
    ), ddof=1) / np.sqrt(n_simulations)
return adjusted_arith_price, se
```

8.5 Arithmetic Asian Option with Antithetic Reduction

```
def arithmetic_antithetic(S0, K, T, r, sigma, n_simulations
    , n_steps):
dt = T / n_steps
discount_factor = np.exp(-r * T)
arith_payoff = np.zeros(n_simulations)

for i in range(n_simulations):
    prices1 = np.zeros(n_steps + 1)
    prices2 = np.zeros(n_steps + 1)
    prices1[0] = prices2[0] = S0

    for t in range(1, n_steps + 1):
        z = np.random.standard_normal()
```

```
            prices1[t] = prices1[t-1] * np.exp((r - 0.5 * sigma
                **2) * dt + sigma * np.sqrt(dt) * z)
            prices2[t] = prices2[t-1] * np.exp((r - 0.5 * sigma
                **2) * dt - sigma * np.sqrt(dt) * z)

        arithmetic_mean1 = np.mean(prices1[1:])
        arithmetic_mean2 = np.mean(prices2[1:])

        arith_payoff1 = max(0, arithmetic_mean1 - K)
        arith_payoff2 = max(0, arithmetic_mean2 - K)

        arith_payoff[i] = (arith_payoff1 + arith_payoff2) / 2

    adjusted_arith_price = discount_factor * np.mean(
        arith_payoff)
    se = np.std(discount_factor * arith_payoff, ddof=1) / np.
        sqrt(n_simulations)

    return adjusted_arith_price, se
```

8.6 Code for Tables

```
    T_values = [0.5, 1, 5]
sigma_values = [0.1, 0.2,0.5]



results = {}

for T in T_values:
    for sigma in sigma_values:



        start_time = time.time()
        option_value,_ = arithmetic_asian_call_monte_carlo(S0,
            K, T, r, sigma, n_simulations, n_steps)

        end_time = time.time()
        elapsed_time = end_time - start_time

        key = (T, sigma, r)
        results[key] = (option_value, elapsed_time)

for params, (value, elapsed_time) in results.items():
    T, sigma, r = params
    print(f"sigma={sigma},␣T={T},␣␣->␣Option␣Value={value},␣
        Time={elapsed_time:.4f}␣seconds")
```

8.7 Convergence Plots

```
    simulations_range = [100, 500, 1000, 5000, 10000, 50000]
prices = []
errors = []
times = []

for n_simulations in simulations_range:
    start_time = time.time()
    price, se = arithmetic_asian_call_monte_carlo(S0, K, T, r,
        sigma, n_simulations, n_steps)
    end_time = time.time()

    prices.append(price)
    errors.append(se)
    times.append(end_time - start_time)

plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
plt.plot(simulations_range, prices, marker='o')
plt.fill_between(simulations_range, np.array(prices) - np.array
    (errors),
                 np.array(prices) + np.array(errors), color='
                     lightblue', alpha=0.5)
plt.xscale('log')
plt.xlabel('Number␣of␣Simulations␣(log␣scale)')
plt.ylabel('Option␣Price')
plt.title('Option␣Price␣vs␣Number␣of␣Simulations')

plt.subplot(1, 2, 2)
plt.plot(simulations_range, times, marker='o', color='red')
plt.xscale('log')
plt.yscale('log')
plt.xlabel('Number␣of␣Simulations␣(log␣scale)')
plt.ylabel('Computation␣Time␣(seconds,␣log␣scale)')
plt.title('Computation␣Time␣vs␣Number␣of␣Simulations')

plt.tight_layout()
plt.show()
```

8.8 Plots to Compare Dividends

```
    T_values = np.linspace(0, 10, 21)
prices = []
prices1 = []

for T in T_values:
    price,_ = arithmetic_approx_from_geometric(S0 = 100, K =
        100, T = T, r = 0.05, sigma = 0.2, q = 0.05,
        n_simulations = 1000, n_steps = 252)
    prices.append(price)

for T in T_values:
    price1,_ = arithmetic_approx_from_geometric(S0 = 100, K =
        100, T = T, r = 0.05, sigma = 0.2, q = 0.00,
        n_simulations = 1000, n_steps = 252)
    prices1.append(price1)

plt.figure(figsize=(10, 6))

plt.plot(T_values, prices, marker='o', label = 'With␣Dividends'
    )
plt.plot(T_values, prices1, marker='o', label = 'Without␣
    Dividends')
plt.title('Option␣Prices␣vs␣Time␣to␣Maturity')
plt.xlabel('Time␣to␣Maturity␣(Years)')
plt.ylabel('Option␣Price')
plt.grid(True)
plt.legend()
plt.show()
```

8.9 Comparative Plots

```
    T_values = np.linspace(1, 20, 21)
n_simulations = 100
prices = []
prices1 = []

for T in T_values:
    price,_ = arithmetic_asian_call_monte_carlo(S0, K, T, r,
        sigma, n_simulations, n_steps)
    prices.append(price)

```

```
for T in T_values:
    price1,_ = arithmetic_approx_from_geometric(S0, K, T, r,
        sigma, n_simulations, n_steps)
    prices1.append(price1)



plt.plot(T_values, prices, marker='o')
plt.plot(T_values, prices1, marker='o')
plt.title('Geometric␣Asian␣Call␣Option␣Price␣vs␣Time␣to␣
    Maturity')
plt.xlabel('Time␣to␣Maturity␣(Years)')
plt.ylabel('Option␣Price')
plt.grid(True)
plt.show()
```

8.10 Moment Matching

```
    def levy_approx(S0, S_avg, K, r, q, sigma, T, t):
    K_z = K - S_avg * (t / T)
    S_z = S0 / ((r - q)*T) * (np.exp(-q*(T-t)) - np.exp(-r*(T-t
        )))
    M = 2*S0**2 / ((r - q) + sigma**2) * ( (np.exp((2 * (r - q)
         + sigma**2) * (T-t)) - 1) / (2 * (r - q) + sigma**2) -
         (np.exp((r - q) * (T-t)) - 1) / (r-q))
    L = M / T**2
    v = np.log(L) -2*(r*(T-t) + np.log(S_z))
    d1 = 1/np.sqrt(v) * (np.log(L)/2 - np.log(K_z))
    d2 = d1 - np.sqrt(v)

    price = S_z * norm.cdf(d1) - K_z * np.exp(-r*(T-t)) * norm.
        cdf(d2)
    return price
```

8.11 Error Histograms

```
    K_values = np.arange(0.25,5.25,0.25)
N = 150
q = 0

abs_errors_method1 = []
abs_errors_method2 = []

for T in K_values:
    price, _ = arithmetic_approx_from_geometric(S0, K, T, r,
        sigma, q, n_simulations, n_steps)
    absolute_error1 = abs(levy_approx(S0, S_avg, K, r, q, sigma
        , T, t) - price)
    absolute_error2 = abs(Asian(S0, K, T, t, sigma, r, N) -
        price)

    abs_errors_method1.append(absolute_error1)
    abs_errors_method2.append(absolute_error2)

bar_width = 0.1
K_values_offset = K_values - bar_width / 2

plt.figure(figsize=(10, 6))

plt.bar(K_values_offset, abs_errors_method1, width=bar_width,
    edgecolor='black', label = 'Moment␣Matching' )
plt.bar(K_values_offset + bar_width, abs_errors_method2, width=
    bar_width, edgecolor='black', label = 'Laplace')

plt.xlabel('Time␣to␣Maturity')
plt.ylabel('Absolute␣Error')
plt.title('Absolute␣Errors␣vs␣Time␣to␣Maturity')
plt.legend()
plt.show()
```

8.12 Laplace Transform

```
    from mpmath import mp, mpf, mpc, pi, sin, tan, exp, gamma,
        hyp1f1, sqrt

def Asian(S0, K, T, t, sigma, r, N):
    S0    = mpf(S0)
    K    = mpf(K)
    sigma = mpf(sigma)
    T    = mpf(T)
    t    = mpf(t)
    r    = mpf(r)

    tau  = mpf((sigma**2) / 4 * (T - t))
    v    = mpf(2 * r / (sigma**2) - 1)
    alp  = mpf(sigma**2 / (4 * S0) * K * T)
    beta = mpf(-1 / (2 * alp))
    N    = mpf(N)

    h = 2 * pi / N
    mp.dps = 100

    c1 = mpf('0.5017')
    c2 = mpf('0.6407')
    c3 = mpf('0.6122')
    c4 = mpc('0', '0.2645')

    ans = mpf('0.0')

    for k in range(int(N / 2)):
        theta = -pi + (k + 0.5) * h
        z     = 2 * v + 2 + N / tau * (c1 * theta / tan(c2 *
            theta) - c3 + c4 * theta)
        dz    = N / tau * (-c1 * c2 * theta / sin(c2 * theta)
            **2 + c1 / tan(c2 * theta) + c4)
        zz    = N / tau * (c1 * theta / tan(c2 * theta) - c3 +
            c4 * theta)
        mu    = sqrt(v**2 + 2 * z)
        a     = mu / 2 - v / 2 - 1
        b     = mu / 2 + v / 2 + 2
        G     = (2 * alp)**(-a) * gamma(b) / gamma(mu + 1) *
            hyp1f1(a, mu + 1, beta) / (z * (z - 2 * (1 + v)))
        ans  += exp(zz * tau) * G * dz

    return (2 * exp(tau * (2 * v + 2)) * exp(-r * (T - t)) * 4
        * S0 / (T * sigma**2) * h / (2j * pi) * ans).real
```

8.13 Computational Time Plots

```
    K_values = np.linspace(0.02,0.04,1)
n_simulations = 10000
n_steps = 252
t = 0
q = 0.00
computation_times = []
for sigma in K_values:
        start_time = datetime.now()
        price = Asian(S0,K,T,t,sigma,r,N)
        price = round(price,3)
        end_time = datetime.now()
        computation_time = (end_time - start_time).
            total_seconds()
        computation_times.append(computation_time)
        print(f"Volatility:␣{sigma:.2f},␣Asian␣option␣price:␣{
            price},␣Computation␣time:␣{computation_time}␣
            seconds")

print(f"Computation␣times:␣{computation_times}")
```

8.14 Optimal N for Laplace Transform

```
    def find_optimal_N(S0, K, T, t, sigma, r, initial_N=10,
        tolerance=1e-8, max_N=5000):
    N = initial_N
    price_prev = Asian(S0, K, T, t, sigma, r, N)

    Ns = [N]
    prices = [price_prev]

    while N < max_N:
        N += 2
        price_curr = Asian(S0, K, T, t, sigma, r, N)
        Ns.append(N)
        prices.append(price_curr)

        if abs(price_curr - price_prev) < tolerance:
            optimal_N = N
            final_price = price_curr
            break

        price_prev = price_curr
    else:
        optimal_N = N
        final_price = price_curr
```