Monte Carlo Methods to Price Asian and European Options
Spring 2009 GE 598 Course Project

Siva K. Gorantla and Sundar Ram Srinivasan

April 30, 2009

Contents

Chapter 1

Introduction

The risk-neutral derivative pricing technique of derivatives involves determining the expected value of scalar functions of the underlying geometric Brownian motion that models the stock prices. For complex derivatives that depend on the stock prices at different times, closed form expressions for the expectations are not available. This report concerns the use of Monte Carlo methods to evaluate the price of two specific derivatives: European and Asian options.

To make the report self sufficient, we provide a brief introduction to Monte Carlo methods and derivative pricing.

1.1 Monte Carlo Methods

Monte Carlo methods are powerful techniques to numerically evaluate expected values of random variables, and more generally, integrals. They rely on generating multiple samples of the random variables and then estimating the expected value using these. Formally, the aim is to evaluate the expected value $\mu = E(f(X))$ of a function of random variable X. The *standard Monte Carlo method* generates i.i.d $\{X_i : i \geq 1\}$, and approximates $E(f(X))$ by

$$\hat{\mu}_n = \frac{(f(X_1) + \cdots + f(X_n))}{n}.$$

The asymptotic properties follow from the Strong Law of large numbers,

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} f(X_i) \to 1, \qquad \text{with probability 1.}$$

In addition, from the Central limit theorem we can conclude that the error $\hat{\mu}_n - \mu_n$ in the Monte Carlo estimate is approximately normally distributed with mean 0 and standard deviation $\frac{\sigma_f}{n}$, where σ_f is the standard deviation of $f(X)$. An important feature of the error is that the quality of this approximation improving with increasing n. Typically, σ_f is also not know, but it can be estimated using the sample standard deviation

$$s_f = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(f(X_i) - \hat{\mu}_n)^2}.$$

The form of the standard error $\frac{\sigma_f}{\sqrt{n}}$ is a central feature of the Monte Carlo method. In higher dimensions, the value of Monte Carlo as a computational tool lies in the fact that its $O(n^{-1/2})$ convergence rate.

1.2 Derivatives Pricing

A derivative is a financial instrument that derives its value from a more basic underlying asset. Typically, the derivative value is a function of the stock price over a specified time interval, say $[0, T]$. Of course the value of future stock prices are not known at time 0 and hence to price the derivative one needs to model the movement of the stock price. The standard model for the stock price $S(t)$ is a geometric Brownian motion. Mathematically, he Black-Scholes model describes the evolution of the stock price through the stochastic differential equation (SDE)

$$dS(t) = \mu S(t)dt + \sigma S(t)dB(t),$$

where μ is the expected return and σ is the volatility of the stock price.

The risk neutral pricing technique obtains the price of a derivative as the the present value of the expected future payoff in a risk neutral world (i.e., μ is the risk free interest rate) discounted at the risk free interest rate. Mathematically, suppose the payoff at time T from the derivative is $V(\{S(t)\}_{0,T})$ then the price of the derivative at time $T = 0$ is fixed at $e^{-rT}E_Q\left[V\left(\{S(t)\}_{[0,T]}\right)\right]$. Here E_Q is the expectation calculated in a risk neutral world.

Chapter 2

European Options in the BSM Model

Let $S(t)$ denote the price of the stock at time t. Consider a call option granting the holder the right to buy the stock at a fixed price K at a fixed time T in the future; the current time is $t = 0$. If at time T the stock price $S(T)$ exceeds the strike price K, the holder exercises the option for a profit of $S(T) - K$; if, on the other hand, $S(T) \leq K$, the option expires worthless. The payoff to the option holder at time T is thus

$$(S(T) - K)^+ = \max\{0, S(T) - K\}.$$

Similarly for a put option, the payoff to the option holder at time T is $(K - S(T))^+$. Hence, the value of the European option at time T ($V_T(S(T))$) is

$$V_T(S(T)) = \begin{cases} (S(T) - K)^+ & \text{for call option} \\ (K - S(T))^+ & \text{for put option} \end{cases}$$

2.1 Solution for the value of European options

In the risk neutral world the stock price follows

$$\frac{dS(t)}{S(t)} = (r - q)dt + \sigma dB(t) \tag{2.1}$$

with B a standard Brownian motion and q is the continuous yield of the asset. The solution of the stochastic differential equation is

$$S(T) = S(0)\exp([r - q - \frac{1}{2}\sigma^2]T + \sigma B^*(T)) \tag{2.2}$$

The random variable $B(T)$ is normally distributed with mean 0 and variance T ; this is also the distribution of $\sqrt{T}Z$ if Z is a standard normal random variable (mean 0, variance 1).

$$S(T) = S(0)\exp([r - q - \frac{1}{2}\sigma^2]T + \sigma\sqrt{T}Z) \tag{2.3}$$

The logarithm of the stock price is thus normally distributed, and the stock price itself has a lognormal distribution.

The expectation $E[e^{-rT}V_T(S(T))]$ is an integral with respect to the lognormal density of $S(T)$. This integral can be evaluated in terms of the standard normal cumulative distribution function Φ as
$BS(Option, K, T, S(0), \sigma, r, q)$ with

$$BS(Call, K, T, S, \sigma, r, q) = S\Phi(d_1) - e^{-rT}K\Phi(d_2) \tag{2.4}$$

$$BS(Put, K, T, S, \sigma, r, q) = e^{-rT}K\Phi(-d_2) - S\Phi(-d_1) \tag{2.5}$$

where $d_1 = \frac{log(S/K)+(r-q+\frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}$ and $d_2 = \frac{log(S/K)+(r-q-\frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}$

2.2 Algorithm for the value of European option

As a precurser to more complex derivatives, we compute the price of the European option using Monte carlo simulation Methods. Comparing the Monte carlo proce to the actual price in (2.4) will provide some useful insights on the quality of the Monte Carlo estimates.

From (2.3), we see that to draw samples of the terminal stock price $S(T)$ it suffices to have a mechanism for drawing samples from the standard normal distribution. Given a mechanism for generating the Z_i, we can estimate $E[e^{-rT}V_T(S(T))]$ using the following algorithm:

Algorithm 1 European option using BSM model

```
for i = 1, ..., n do
    generate Z_i
    set S_i(T) = exp([r - q - (1/2)σ^2]T + σ√T Z_i)
    set C_i = e^{-rT}(V_T(S(T)))^+
    set Ĉ_n = (C_1 + ... + C_n)/n
end for
```

2.3 Consistency of the Estimate

The estimated standard error is $s_n/\sqrt{n}$ where

$$s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(C_i - \hat{C}_n)^2$$

For finite but at least moderately large n, we can supplement the point estimate $\hat{C}_n$ with a confidence interval. The $1-\alpha$ confidence interval is

$$\hat{C}_n \pm z_{\alpha/2}\frac{s_n}{\sqrt{n}}.$$

2.4 Variance Reduction Techniques

To reduce the variance of the estimate $s_n/\sqrt{n}$, we use variance reduction techniques. Two of the techniques - antithetic and control variate techniques are discussed here.

- **Antithetic Variates** The method of *antithetic variates* attempts to reduce variance by introducing negative dependence between pairs of replications. In particular, in a simulation driven by independent standard normal random variables, antithetic variates can be implemented by pairing a sequence $Z_1, Z_2, \cdots$ of i.i.d $N(0,1)$ variables with the sequence $-Z_1, -Z_2, \cdots$ of i.i.d. $N(0,1)$ variables. To analyze this approach more precisely, suppose our objective is to estimate an expectation E[Y] and that using some implementation of antithetic sampling produces a sequence of pairs of observations $(Y_1, \tilde{Y}_1), (Y_2, \tilde{Y}_2), \cdots, (Y_n, \tilde{Y}_n)$. The key features of the antithetic variates method are the following:

  - the pairs $(Y_1, \tilde{Y}_1), (Y_2, \tilde{Y}_2), \cdots, (Y_n, \tilde{Y}_n)$ are i.i.d.;
  - for each i, Y_i and $\tilde{Y}_i$ have the same distribution, though ordinarily they are not independent.

  The antithetic variates estimator is simply the average of all 2n observations,

  $$\hat{Y}_{2n} = \frac{1}{2n}\left(\sum_{i=1}^{n} Y_i + \sum_{i=1}^{n}\tilde{Y}_i\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i + \tilde{Y}_i}{2}\right)$$

- **Control Variate** This method exploits information about the errors in estimates of known quantities to reduce the error in an estimate of an unknown quantity.

- produce $X_i \tilde{X}, 1 \leq i \leq n$. E(X) is known and hence the error $E(X) - X_i$.
- (X_i, Y_i) are i.i.d. such that X and Y are highly correlated.
- $E(Y) - Y_i \approx b(E(X) - X_i)$

Hence the new replication of Y_i is

$$Y_i^b = Y_i + b(E(X) - X_i)$$

and the estimate is

$$\hat{Y}_n^b = \frac{1}{n} \sum_{i=1}^{n} (Y_i + b(E(X) - X_i))$$

where

$$b^* = \frac{Cov(X, Y)}{Var(X)}$$

and is estimated as

$$\hat{b}_n = \frac{\sum_{i=1}^{n} (X_i - \overline{X}_n)(Y_i - \overline{Y}_n)}{\sum_{i=1}^{n} (X_i - \overline{X}_n)^2}$$

For a European option with payoff $V_T(S(T))$,

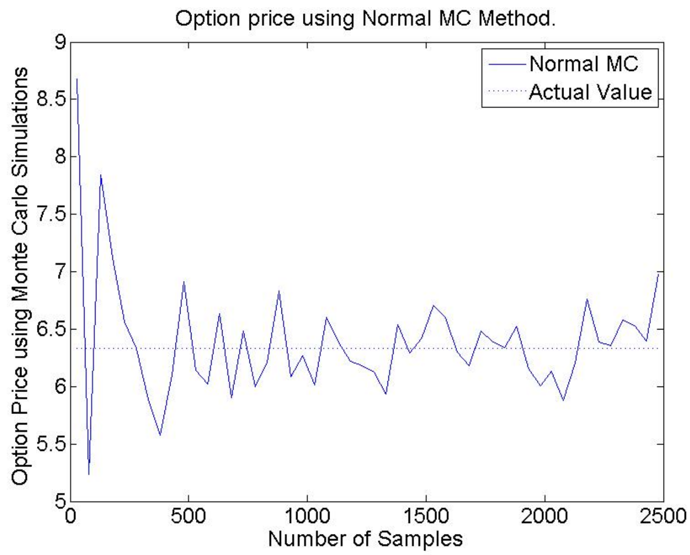
$$Y_i = e^{-rT} V_T(S(T)^i) \quad X_i = e^{-(r-q)T} S(T)^i \quad Y_i^b = Y_i + b(S_0 - X_i)$$

2.5 Numerical Example

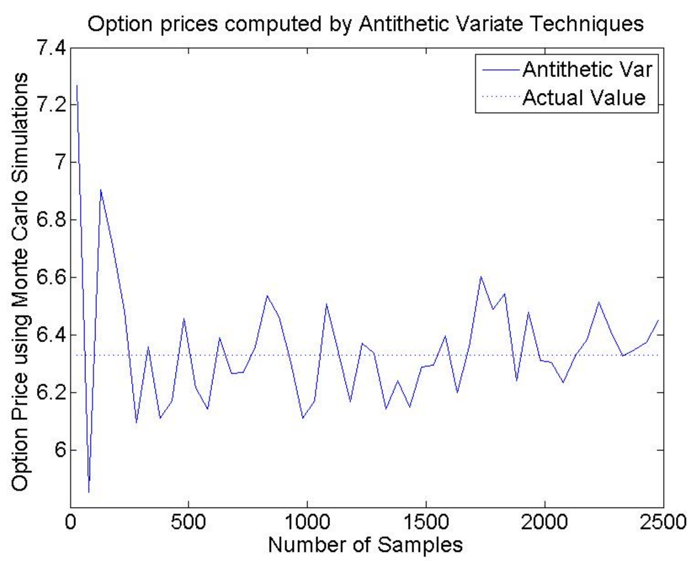
For Put Option with $S = K = 100, T = 1, \sigma = 0.2, r = 0.05, q = 0.02$, Black-Scholes price from (2.5) is $**6.3301**.

	Normal MC methods			Antithetic variate		
n	put price	Error	Standard error	put price	Error	Standard error
80	5.23227	-1.0978	0.972075	5.85295	-0.4771	0.526998
230	6.56452	0.2344	0.59607	6.48255	0.1525	0.297884
530	6.14474	-0.1853	0.384063	6.21359	-0.1165	0.197273
1030	6.01002	-0.3201	0.269132	6.16987	-0.1602	0.141586
2480	6.97682	0.6467	0.189561	6.45108	0.1210	0.094278
30000	6.33266	.00258	0.0528868	6.30663	-0.2645	0.0324475

The results from the code for n=30000 are given as follows:
****
The price using standard Monte Carlo technique is 6.33266. The standard error is 0.0528868
The price using antithetic variate Monte Carlo technique is 6.30663. The standard error is 0.0324475
The price using BSM formula is 6.33008
Time taken to price the option is 0.0309999 seconds
****
More numerical values for n=50-2500 are provided in the **prob2_project.xls** file attached.
The Figure 2.1 shows the prices computed using Normal and antithetic variate techniques for the variables given in the example above.
The standard error as a function of n is plotted in Figure 2.2. It can be seen that the standard error is less for the simulations using Antithetic variate techniques than the normal Monte Carlo techniques.



(a)



(b)

Figure 2.1: Put Option price for $S = K = 100, \sigma = 0.2, r = 0.5, q = .02, T = 1$ using normal and antithetic variate techniques in Monte Carlo Simulations as a function of number of smaples

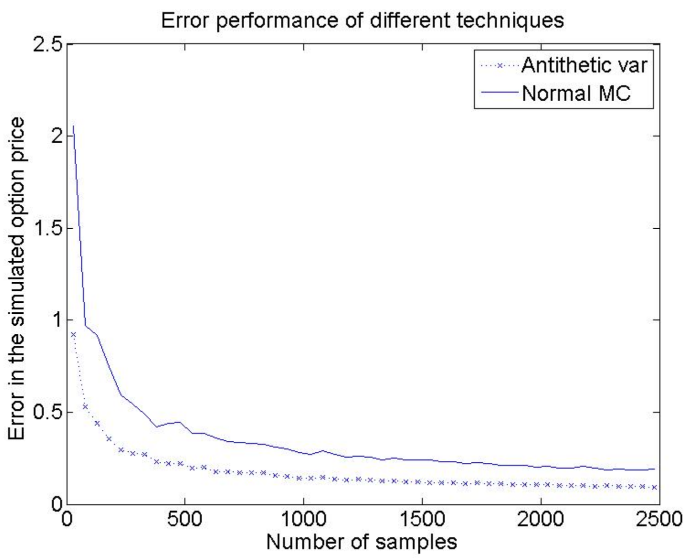


Figure 2.2: Standard error for $S = K = 100, \sigma = 0.2, r = 0.05, q = .02, T = 1$ using normal and antithetic variate techniques for Monte Carlo Simulations as a function of number of samples.

Chapter 3

Asian Options

The payoff of a standard European call option is determined by the terminal stock price S(T) and does not otherwise depend on the evolution of S(t) between times 0 and T . In Asian Options, it is necessary to simulate paths over multiple intermediate dates and not just at the initial and terminal dates. These are options with payoffs that depend on the average level of the underlying asset. This includes, for example, the payoff $(\overline{S} - K)^+$ with

$$\overline{S} = \frac{1}{m}\sum_{j=1}^{m} S(t_j)$$

for some fixed set of dates $0 = t_0 < t_1 < \;\; < t_m = T$, with T the date at which the payoff is received. To calculate the expected discounted payoff $E[e^{-rT}V_T(\overline{S})]$, we need to be able to generate samples of the average $\overline{S}$. The simplest way to do this is to simulate the path $S(t_1), \cdots, S(t_m)$ and then compute the average along the path. We saw in (2.3) how to simulate S(T) given S(0); simulating $S(t_{j+1})$ from $S(t_j)$ works the same way:

$$S(t_{j+1}) = S(t_j)\exp\left([r - q + \frac{1}{2}\sigma^2](t_{j+1} - t_j) + \sigma\sqrt{t_{j+1} - t_j}Z_{j+1}\right)$$

where $Z_1, \cdots, Z_m$ are independent standard normal random variables. Let Z_{ij} denote the j-th draw from the normal distribution along the i-th path. The algorithm for pricing Asian Call Options is

3.1 Variance Reduction Techniques

To reduce the variance of the estimate $s_n/\sqrt{n}$, we use variance reduction techniques. Two of the techniques - antithetic and control variate techniques

Algorithm 2 Asian option using BSM model

```
1: for i = 1, ··· , n do
2:   for j = 1, ··· , m do
3:     generate Z_ij
4:     set S_i(t_j) = S(t_{j-1}) exp([r - q + (1/2)σ^2](t_j - t_{j-1}) + σ sqrt(t_j - t_{j-1}) Z_ij)
5:     set S̄ = (S_i(t_1) + ··· + S_i(t_m)) / m
6:     set C_i = e^{-rT}(V_T(S̄))
7:     set Ĉ_n = (C_1 + ... + C_n) / n
8:   end for
9: end for
```

are discussed here.

- **Antithetic Variates** The antithetic variates monte carlo technique for the Asian options is similar to those for the European options.

- **Control Variate**

$$\hat{b}_n = \frac{\sum_{i=1}^{n}(X_i - \overline{X}_n)(Y_i - \overline{Y}_n)}{\sum_{i=1}^{n}(X_i - \overline{X}_n)^2}$$

  For an Asian option,

$$Y_i = e^{-rT}V_T(S(T)^i) \qquad X_i = e^{-(r-q)T}S(T)^i \qquad Y_i^b = Y_i + b(S_0 - X_i)$$

3.2 Numerical Example

For asian Option with $S = K = 100, T = 1, \sigma = 0.2, r = 0.10, q = 0, m = 50$, the output using the code for n=30000 samples is

```
************************************************************************
The price using standard Monte Carlo technique is 5.82299. The standard error is 0.0466805
The price using antithetic variate Monte Carlo technique is 5.69454. The standard error is 0.0483065
The price using control variate Monte Carlo technique is 5.85497. The standard error is 0.00128007
Time taken to price the option is 0.826 seconds
************************************************************************
```

Hence, the correct value of the Asian Call option is $5.855 \pm .001$ and is correct upto 2 digits. This price is estimated from the Control Variate Monte Carlo Technique since it has the least standard error.

	Normal MC methods		Antithetic variate		Control Variate	
n	put price	Standard error	put price	Standard error	put price	Standard error
80	4.6322	0.7288	4.5030	0.7104	5.8425	0.0191
530	5.7850	0.3526	5.6562	0.3425	5.8738	0.0105
1030	5.9847	0.2535	5.8482	0.2466	5.8649	0.0069
10000	5.96842	0.0816785	5.83507	0.0846407	5.85516	0.00224795
30000	5.82299	0.0466805	5.69454	0.0483065	5.85497	0.00128007

More numerical values for n=50-2500 are provided in the **prob4_project.xls** file attached.

The Figure 3.1 shows the prices computed using Normal and antithetic variate techniques for the variables given in the example above.

The standard error as a function of n is plotted in Figure 3.2. It can be seen that the standard error is comparably same for antithetic variate and normal Monte Carlo techniques for m=50. But the standard error for the control variate is significantly less compared to any of the above mentioned techniques. Hence, the control variate estimate of the price provides a more accurate estimate of the Asian option price.

3.3 Continuous Average Asian Option

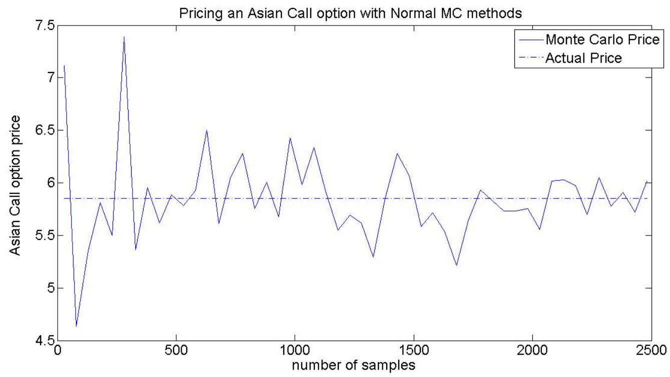
Suppose the discrete average of an Asian option is replaced by a continuous average

$$\overline{S} = \frac{1}{T}\int_{u=0}^{T} S(u)du$$

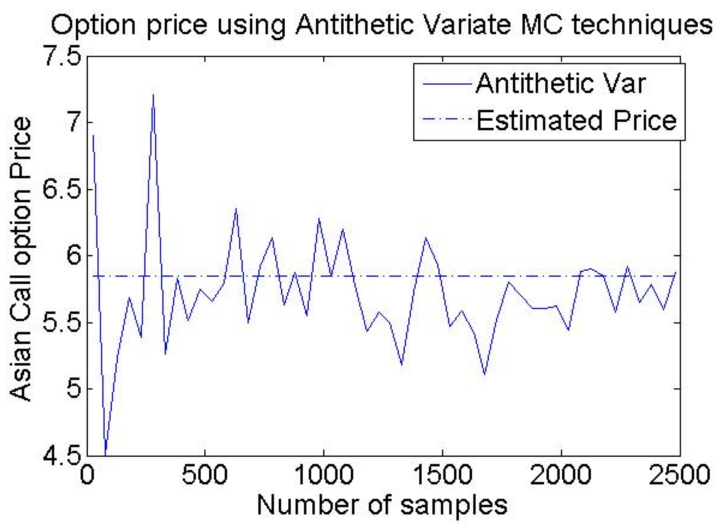
In this case, even if we generate values of $S(t_i)$ at a discrete set of dates t_i, we cannot calculate $\overline{S}$ exactly we need to use a discrete approximation to the continuous average. The bias can be made arbitrarily small by using a sufficiently small simulation time step, at the expense of increasing the computational cost per path.

- **Fixed n=1000, increasing m**

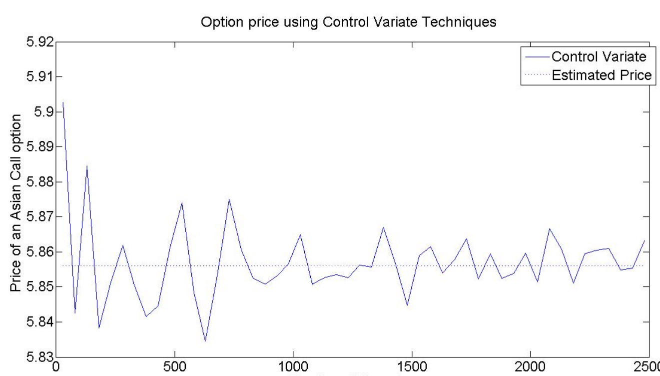
  Here, we fix the number of sample paths and evaluate the performance of the estimate of the Asian option price as a function of discrete time sample size. We see that as m increases, the error in estimate becomes less. The performance of Antithetic variate Monte Carlo techniques is better than the normal methods, and the control variate technique outperforms both antithetic variate and normal methods. This can be seen in the Figure 3.3.



(a)



(b)



(c)

Figure 3.1: Asian Call Option price for $S = K = 100, \sigma = 0.2, r = 0.1, q = 0, T = 1, m = 50$ using normal, antithetic variate and control variate techniques in Monte Carlo Simulations as a function of number of samples

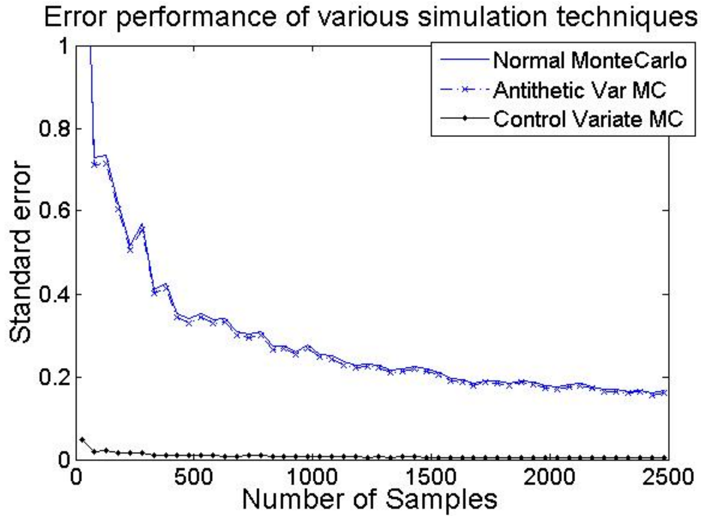


Figure 3.2: Standard error for $S = K = 100, \sigma = 0.2, r = 0.05, q = 0, T = 1, m = 1$ using normal, antithetic variate and control variate techniques for Monte Carlo Simulations as a function of number of samples.

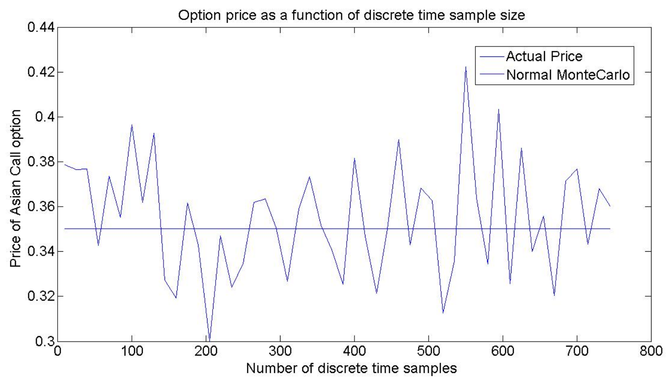
The standard error estimated using various techniques is summarized in Figure 3.4. Its clear that the control variate technique performs significantly better. Also observe that, for lower values of m, the standard error from normal methods is more than the error in antithetic methods. As m increases, the standard error of both normal and antithetic methods tend to be the same.

More numerical values for m=10-750 are provided in the **prob5b_project.xls** file attached.

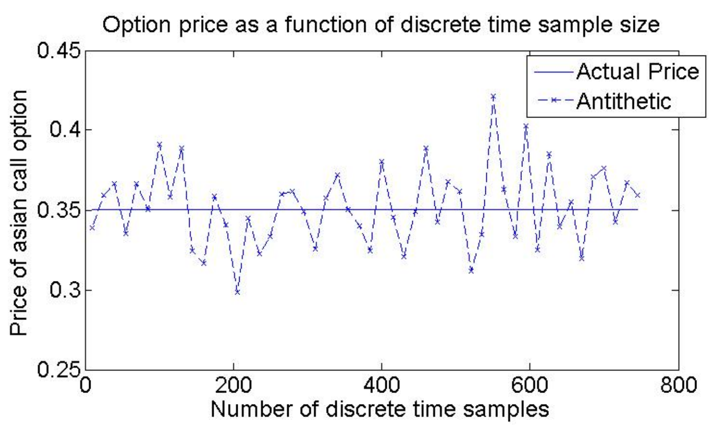
- **Fixed m=500, increasing n**

  If we fix the number of time instants for averaging and evaluate the performance of the estimate the Asian option price as a function of the number of sample paths, we see that as n increases, the error in estimate becomes less. The performance of Antithetic variate Monte Carlo techniques is better than the normal methods, and the control variate technique outperforms both antithetic variate and normal methods. This can be seen in the Figure 3.5 and 3.6.

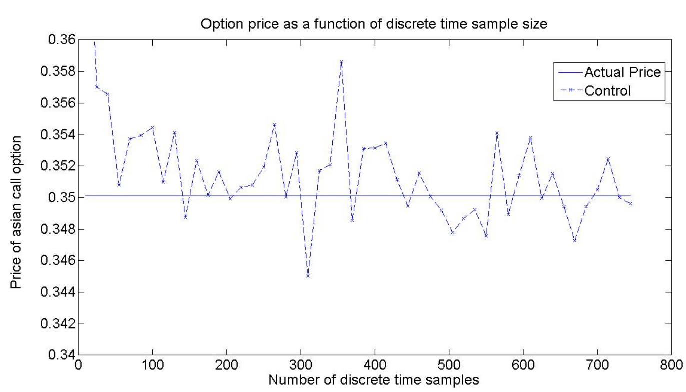
  The standard error estimated using various techniques is summarized in Figure 3.7. Its clear that the control variate technique performs significantly better.



(a)



(b)



(c)

Figure 3.3: Asian Call Option price for $T = 2, r = 0.05, \sigma = 0.5, S = K = 2, q = 0.$ using normal, antithetic variate and control variate techniques in Monte Carlo Simulations as a function of number of discrete time samples for n=1000

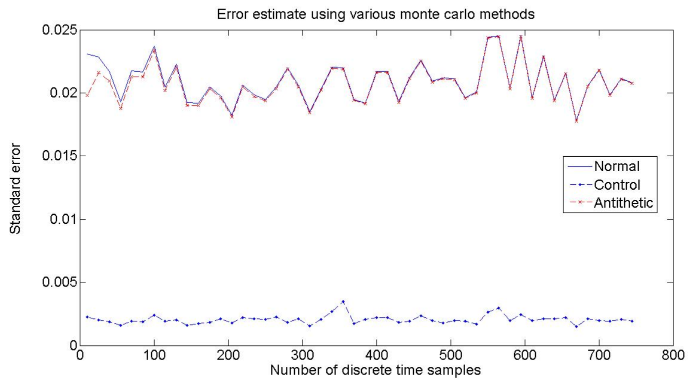
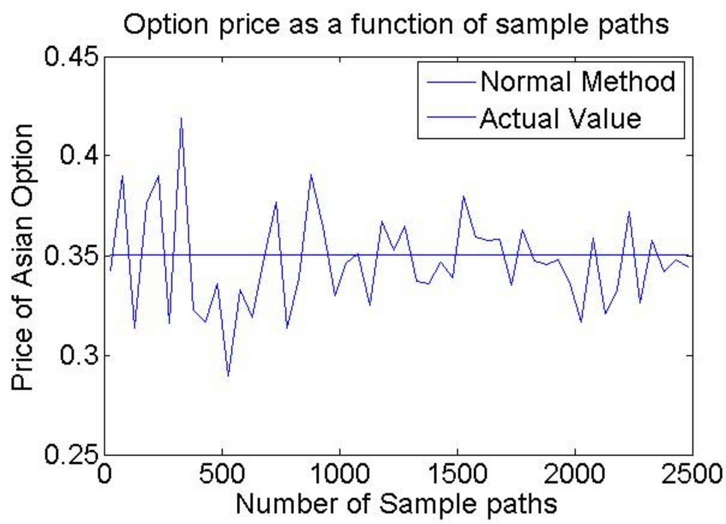


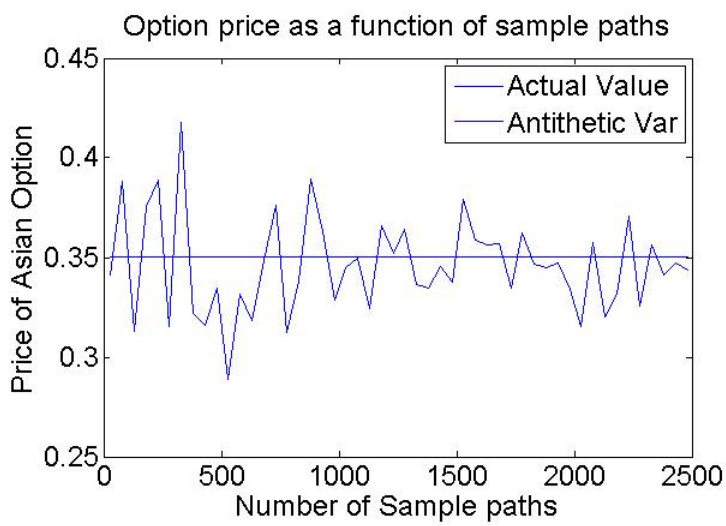
Figure 3.4: Standard error for $T = 2, r = 0.05, \sigma = 0.5, S = K = 2, q = 0$ using normal, antithetic variate and control variate techniques for Monte Carlo Simulations as a function of number of discrete time samples. for n=1000

	Normal MC methods		Antithetic variate		Control Variate	
n	put price	Standard error	put price	Standard error	put price	Standard erro
30	0.342573	0.154763	0.341099	0.1538785	0.387628	0.0217337
230	0.389823	0.0483921	0.388692	0.04822345	0.352539	0.00440638
530	0.289458	0.0239567	0.2888375	0.02389935	0.347266	0.00217793
980.00	0.3297	0.0195	0.3288	0.0195	0.3520	0.0019
2480	0.344353	0.013974	0.3435365	0.01393205	0.349372	0.001384

More numerical values for n=50-2500 are provided in the **prob5a_project.xls** file attached.



(a)



(b)

Figure 3.5: Asian Call Option price for $T = 2, r = 0.05, \sigma = 0.5, S = K = 2, q = 0$. using normal, antithetic variate and control variate techniques in Monte Carlo Simulations as a function of number of sample paths for m=500

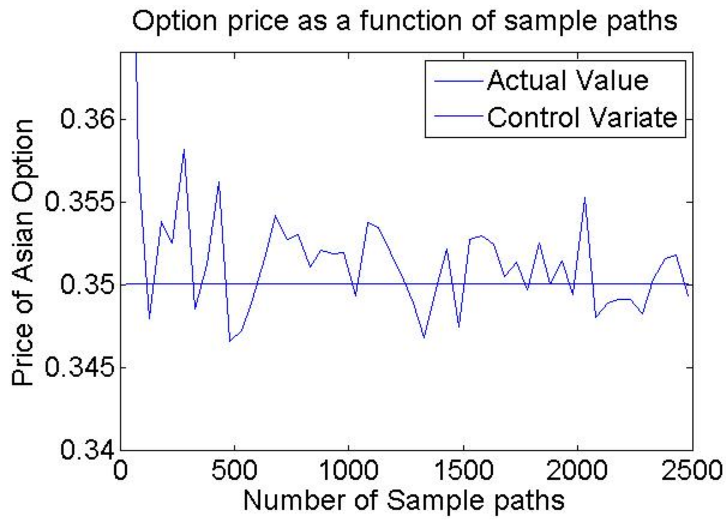


Figure 3.6: Asian Call Option price for $T = 2, r = 0.05, \sigma = 0.5, S = K = 2, q = 0$. using control variate techniques in Monte Carlo Simulations as a function of number of sample paths for m=500

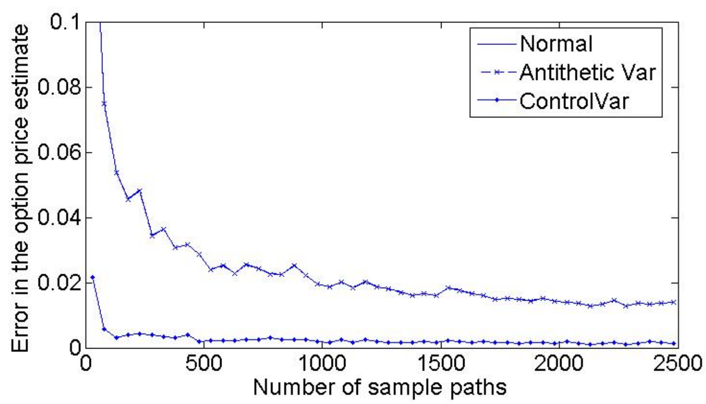


Figure 3.7: Standard error for $T = 2, r = 0.05, \sigma = 0.5, S = K = 2, q = 0$ using normal, antithetic variate and control variate techniques for Monte Carlo Simulations as a function of number of samples.

Chapter 4

Programming Issues

In this chapter we briefly discuss the organization and the key issues of the program attache to the end of the report.

1. *Object Oriented Structure: Class Options* is used to build options on the underlying asset. Class options includes an object of *class stock* as a data member. *Class Stock* is used to build objects of the underlying asset on which the option is based. The main function of interest is *price-op* that prices the function. Since the function needs to access the private members of class Stock it is declared to be a *friend* of class stock.

2. *Memory Issues:* Observe that from step 5 of Algorithm 1, in order to calculate $\hat{C}_n$, we need to store $C_1, \ldots, C_n$ and the storage size increases with n. In order to avoid this, we can write the estimate in terms of the previous estimate and the generated value at n-th time instant.

$$\hat{C}_n = \frac{(C_1 + \ldots + C_n)}{n} = \frac{(n-1)\hat{C}_{n-1} + C_n}{n}$$

   Hence we use only one storage unit for the estimate and one for the option price generated at n-th time instant. Thus the step 5 from Algorithm 1 will now be

$$\text{5: set}\hat{C}_n = \frac{(n-1)\hat{C}_{n-1} + C_n}{n}$$

3. *Code Optimization and Structuring:* A key feature of the program is that it treats European option as a special case of Asian option with exactly 1 monitoring intervals. This allows us to develop a single integrated user interface and pricing function.

4. *Time Elapsed:* To determine the time elapsed, we use the function $gettimeofday()$ available in the library. $sys/time.h$ The time elapsed is calculated as the difference between the *time since epoch* determined at the start and end of the pricing function.

5. *Generating Normal random variables:* This corresponds to step 2 in the Algorithm 1 and step 3 of Algorithm 2 discussed above. Gaussian random variables can be generated by Box-Muller method. The Box-Muller is implemented in *gasdev*.

6. *Generating the Gaussian cdf:* We generate the Gaussian cdf $\phi(x)$ as $\frac{erf\left(\frac{x}{\sqrt{2}}\right)}{2}$. Here erf is the standard error function and is available in the library *cmath*.

7. *Scope of the program:* We have designed the program for a maximum of $30,000$ simulation runs. This limitation is because of the data structures (essentially arrays) that we have used. It is possible to extend this to higher values by using data structures with better scalability properties like linked lists.