4

Monte Carlo Simulation

4.1 INTRODUCTION

MANY complex derivatives exist for which analytical formulae are not possible, Monte Carlo simulation (first used by Boyle, 1977) provides a simple and flexible method for valuing these types of instruments. It can deal easily with multiple random factors; for example options on multiple assets, random volatility or random interest rates. Monte Carlo simulation also allows the incorporation of more realistic asset price processes, such as jumps in asset prices, and more realistic market conditions such as the discrete fixing of exotic path-dependent options. It can also give insights into the effectiveness of a hedge. However, Monte Carlo simulation is computationally inefficient in its basic form. In this chapter, after first introducing the Monte Carlo simulation methods, we show how to improve its efficiency using control variates and quasi-random numbers (deterministic sequences). We also describe in detail how Monte Carlo simulation can be used to value complex path-dependent options. The interested reader is recommended to study Ripley (1987) for a general discussion of stochastic simulation.

4.2 VALUATION BY SIMULATION

In Chapter 1 it was shown that the value of an option is the risk-neutral expectation of its discounted pay-off. We can obtain an estimate of this expectation by computing the average of a large number of discounted payoffs. Consider a European-style option which pays C_T at the maturity date T. Firstly, we simulate the risk-neutral processes for the state variables from their values today, time zero to the maturity date T and compute the pay-off of the contingent claim, $C_{T,j}$ for this simulation (j). Then we discount this pay-off using the simulated short-term interest rate sequence:

$$C_{0,j} = \exp\left(-\int_0^T r_u \, du\right) C_{T,j} \tag{4.1}$$

In the case of constant interest rates equation (4.1) simplifies to

$$C_{0,j} = \exp(-rT) C_{T,j}$$

The simulations are repeated many (say M) times and the average of all the outcomes is taken

$$\hat{C}_0 = \frac{1}{M} \sum_{i=1}^{M} C_{0,j} \tag{4.2}$$

where $\hat{C}_0$ is an estimate of the true value of the option C_0, but with an error due to the fact that it is an average of randomly generated samples and so is itself random. A measure of the error is the standard deviation of $\hat{C}_0$ which is called the standard error $SE(.)$ and can be estimated as the sample standard deviation of $C_{0,j}$ divided by the square root of the number of samples (see Hines and Montgomery, 1980, for an introduction to probability and statistics).

$$\mathrm{SE}(\hat{C}_0) = \frac{\mathrm{SD}(C_{0,j})}{\sqrt{M}} \tag{4.3}$$

where

$$\mathrm{SD}(C_{0,j}) = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}\left(C_{0,j} - \hat{C}_0\right)^2}$$

Let us look at a simple, specific example in detail; a standard European call option in the Black–Scholes world. Here interest rates are constant and so, as noted above, the discounting term in (4.1) becomes $\exp(-rT)$. This is the same for all simulations and so can be taken out of equation (4.1) and applied once to the average obtained using equation (4.2). In order to implement Monte Carlo simulation we need to simulate the geometric Brownian motion (GBM) process for the underlying asset[1]

$$dS_t = (r - \delta)S_t\,dt + \sigma S_t\,dz_t \tag{4.4}$$

The best way to simulate a variable following GBM is via the process for the natural logarithm of the variable which follows arithmetic Brownian motion and is normally distributed. Let $x_t = \ln(S_t)$ then we have

$$dx_t = \nu\,dt + \sigma\,dz_t, \qquad \nu = r - \delta - \tfrac{1}{2}\sigma^2 \tag{4.5}$$

Equation (4.5) can be discretised[2] by changing the infinitesimals dx, dt and dz into small changes Δx, Δt and Δz

$$\Delta x = \nu\Delta t + \sigma\Delta z \tag{4.6}$$

This representation involves no approximation because it is actually the solution of the SDE (4.5) which we can write as

$$x_{t+\Delta t} = x_t + \nu\Delta t + \sigma(z_{t+\Delta t} - z_t) \tag{4.7}$$

In terms of the asset price S we have

$$S_{t+\Delta t} = S_t \exp(\nu\Delta t + \sigma(z_{t+\Delta t} - z_t)) \tag{4.8}$$

where z_t would normally be defined as being equal to zero. The random increment $z_{t+\Delta t} - z_t$ has mean zero and a variance of Δt, it can therefore be simulated by random samples of $\sqrt{\Delta t}\varepsilon$, where ε is a sample from a standard normal distribution.[3] Equation (4.8) therefore provides a way of simulating values of S_t. We divide the time period over which we wish to simulate S_t, in this case $(0, T)$, into N intervals such that $\Delta t = T/N$. We can then generate values of S_t at the end of these intervals, $t_i = i\Delta t$, $i = 1, \ldots, N$ using equation (4.7) as follows:

$$S_{t_i} = \exp(x_{t_i}), \qquad x_{t_i} = x_{t_{i-1}} + \nu\Delta t + \sigma\sqrt{\Delta t}\varepsilon_i \tag{4.9}$$

FIGURE 4.1 Simulated Asset Price Paths in the Black–Scholes World

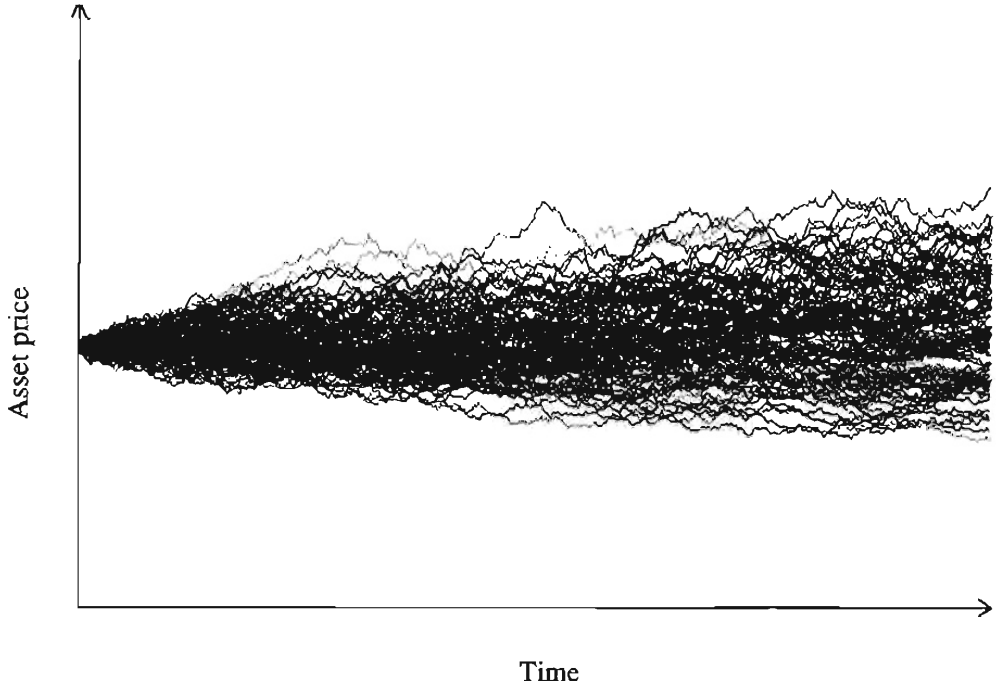


Figure 4.1 illustrates a set of $M = 100$ simulated paths using (4.9) repeatedly with typical parameter values for a stock: $S = 100$, $\sigma = 20$ per cent, $r = 6$ per cent, $T = 1$ year, $N = 365$. For each simulated path we compute the pay-off of the call option $\max(0, S_T - K)$. To obtain the estimate of the call price we simply take the discounted average of these simulated pay-offs

$$\hat{C}_0 = \exp(-rT)\frac{1}{M}\sum_{j=1}^{M}\max(0, S_{T,j} - K) \tag{4.10}$$

Note that for this simple example, since we have the solution of the underlying SDE (equation (4.7)), we can generate the samples of S_T directly without simulating the entire path as shown in Figure 4.1. This is not the case in general, as we will see later; normally we can only obtain an approximate discretisation of the SDE which must be simulated with relatively small time steps. Figure 4.2 gives a pseudo-code implementation of the Monte Carlo valuation of a European call option.

Once again, note that to compute the European call option estimate under GBM we can set $N = 1$, but this is not the case in general.

Example : Pricing a European Call Option by Monte Carlo Simulation

We price a one-year maturity, at-the-money European call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum. The simulation has 10 time steps and 100 simulations; $K = 100$, $T = 1$

FIGURE 4.2 Pseudo-code for Monte Carlo Valuation of a European Call Option in a *Black–Scholes World*

```
initialise_parameters { K, T, S, sig, r, div, N, M }

{ precompute constants }

dt = T/N
nudt = (r-div-0.5*sig^2)*dt
sigsdt = sig*sqrt(dt)
lnS = ln(S)

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

lnSt = lnS

  for i = 1 to N do { for each time step }
    ε = standard_normal_sample
    lnSt = lnSt + nudt + sigsdt*ε { evolve the stock price }
  next i

  ST = exp(lnSt)
  CT = max( 0 , ST - K )
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 10$, $M = 100$. Figure 4.3 illustrates the numerical results, the simulated paths of $\ln(S_i)(i = 1, \ldots, 10)$ are only shown for $j = 1, \ldots 5$ and $j = 95, \ldots, 100$. The corresponding standard normal random numbers ε are shown in the table below the table of $\ln(S_i)$ values in Figure 4.3.

Firstly, the constants; Δt (dt), $\nu\Delta(nudt)$, $\sigma\sqrt{\Delta t}(sigsdt)$, and $\ln(S)(lnS)$ are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{10} = 0.1$$

$$nudt = (r - \delta - \tfrac{1}{2}\sigma^2)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 0.1 = 0.001$$

$$sigsdt = \sigma\sqrt{\Delta t} = 0.2\sqrt{0.1} = 0.0632$$

$$lnS = \ln(S) = 4.6052$$

Then for each simulation $j = 1$ to M, where $M = 100$, $\ln(S_i)$ is initialised to lnS:

$$\ln(S_i) = 4.6052$$

FIGURE 4.3 Numerical Example for Monte Carlo Valuation of a European Call Option in a Black–Scholes World

K	T	S	sig	r	div	N	M	sum_CT	sum_CT2	SD
100	1	100	0.2	0.06	0.03	10	100	996.49	26610.7	12.22457

dt	nudt	sigsdt	lnS	call_value	SE
0.1	0.0010	0.0632	4.6052	9.3846	1.2225

lnSt

j \ i	0	1	2	3	4	5	6	7	8	9	10	ST	CT	CT*CT
1	4.6052	4.6030	4.6257	4.6738	4.6512	4.6666	4.6619	4.5864	4.5521	4.4840	4.6521	104.81	4.8070	23.11
2	4.6052	4.6862	4.6749	4.5386	4.4745	4.4546	4.4901	4.5206	4.4977	4.4618	4.4993	89.95	0.0000	0.00
3	4.6052	4.6430	4.6144	4.6184	4.6770	4.7225	4.7489	4.7402	4.7507	4.7275	4.7983	121.30	21.2996	453.67
4	4.6052	4.6188	4.6297	4.5106	4.5404	4.4814	4.4843	4.4984	4.5072	4.5060	4.4590	86.40	0.0000	0.00
5	4.6052	4.6807	4.7067	4.6915	4.7176	4.7258	4.7810	4.8131	4.8516	4.7885	4.8243	124.49	24.4939	599.95
95	4.6052	4.6121	4.6291	4.6968	4.6099	4.6949	4.5597	4.5229	4.6530	4.6761	4.6349	103.02	3.0211	9.13
96	4.6052	4.6331	4.6620	4.7152	4.7603	4.8091	4.8847	4.8218	4.7674	4.7796	4.8012	121.65	21.6545	468.92
97	4.6052	4.5729	4.7010	4.6682	4.7704	4.8236	4.9556	4.8797	4.9385	4.9607	4.9938	147.49	47.4931	2255.59
98	4.6052	4.5910	4.6111	4.6929	4.6479	4.7125	4.8040	4.9143	4.9915	4.9884	4.9925	147.30	47.3012	2237.41
99	4.6052	4.5634	4.5047	4.4652	4.4984	4.5028	4.5370	4.4913	4.5071	4.4906	4.4065	81.98	0.0000	0.00
100	4.6052	4.5189	4.5183	4.4599	4.5276	4.6812	4.6191	4.5996	4.5985	4.7130	4.7282	113.09	13.0941	171.46

£

j \ i	1	2	3	4	5	6	7	8	9	10
1	−0.0497	0.3425	0.7442	−0.3723	0.2277	−1.6708	0.3709	−0.5581	−1.0924	2.6422
2	1.2660	−0.1948	−2.1717	−1.0290	−0.3296	0.5444	0.4668	−0.3777	−0.5831	0.5763
3	0.5818	−0.4677	0.0476	0.9110	0.7042	0.4014	−0.1541	0.1510	−0.3833	1.1032
4	0.1999	0.1557	−1.8976	0.4551	−0.9486	0.0294	0.2076	0.1225	−0.0347	−0.7592
5	1.1781	0.3955	−0.2564	0.3978	0.1129	0.8569	0.4924	0.5929	−1.0130	0.5489
95	0.0938	0.2533	1.0545	−1.3899	1.3276	−2.1535	−0.5969	2.0413	0.3494	−0.6672
96	0.4258	0.4411	0.8249	0.6971	0.7571	1.1791	−1.0113	−0.8750	0.1762	0.3261
97	−0.5256	2.0098	−0.5352	1.6003	0.8254	2.0709	−1.2152	0.9144	0.3343	0.5075
98	−0.2396	0.3012	1.2779	−0.7271	1.0051	1.4321	1.7281	1.2042	−0.0647	0.0485
99	−0.6757	−0.9440	−0.6416	0.5091	0.0544	0.5256	−0.7383	0.2339	−0.2779	−1.3456
100	−1.3793	−0.0259	−0.9388	1.0538	2.4134	−0.9981	−0.3233	−0.0339	1.7945	0.2252

Then for each time step $i = 1$ to N, where $N = 10$, $\ln(S_t)$ is simulated. For example for $j = 1$ and $i = 1$ (dropping the i and j subscripts):

$$\ln(S_t) = \ln(S_t) + nudt + sigsdt \times \varepsilon$$

$$\ln(S_t) = 4.6052 + 0.001 + 0.0632 \times (-0.0497) = 4.6030$$

At $i = 10$

$$S_T = \exp(\ln(S_t)) = \exp(4.6521) = 104.81$$

$$C_T = \max(0, S_T - K) = \max(0, 104.81 - 100) = 4.8070$$

The sum of the values of C_T and the squares of the values of C_T are accumulated:

$$\sum_{j=1}^{M} C_{T,j} = 996.488 \text{ (sum\_CT)} \quad \text{and} \quad \sum_{j=1}^{M} (C_{T,j})^2 = 26610.7 \text{(sum\_CT2)}$$

The estimate of the option value $\hat{C}_0$ (call_value) is then given by

$$\hat{C}_0 = 996.488/100 \times \exp(-0.06 \times 1) = 9.3846$$

The standard deviation (SD) is given by

$$\text{SD} = \frac{\sqrt{\sum_{j=1}^{M}(C_{T,j})^2 - \frac{1}{M}\left(\sum_{j=1}^{M} C_{T,j}\right)^2}\exp(-2rT)}{M-1}$$

$$= \frac{\sqrt{26610.73 - \frac{1}{100}(996.488)^2}\exp(-2 \times 0.06 \times 1)}{100-1} = 12.2246$$

and so the standard error (SE) is

$$\text{SE} = \frac{SD}{\sqrt{M}} = 12.2246/10 = 1.22246$$

Unfortunately, in order to get an acceptably accurate estimate of the option price a very large number of simulations has to be performed, typically in the order of millions ($M > 1\,000\,000$). This problem can be dealt with by using variance reduction methods. These methods work on exactly the same principle as that of hedging an option position, that is that the pay-off of a hedged portfolio will have a much smaller variability than an unhedged pay-off. This corresponds to the variance (or equivalently standard error) of a simulated hedge portfolio being much smaller than that of the unhedged pay-off. We will stress this interpretation throughout this chapter.

4.3 ANTITHETIC VARIATES AND VARIANCE REDUCTION

Imagine that you have written an option on an asset S_1 and simultaneously are able to write an option on an asset S_2 which is perfectly negatively correlated with S_1, and

which currently has exactly the same price as S_1. That is S_1 and S_2 satisfy the stochastic differential equations

$$dS_{1,t} = rS_{1,t}\,dt + \sigma S_{1,t}\,dz_t \tag{4.11}$$

$$dS_{2,t} = rS_{2,t}\,dt - \sigma S_{2,t}\,dz_t \tag{4.12}$$

The value of these two options are indentical since the price and volatility of the two assets are identical. However, the variance of the pay-off of a portfolio consisting of the two options is much less than the variance of the pay-off of each individual option since, roughly speaking, when one option pays off the other does not and vice versa. It may not be obvious at first why this leads to a smaller variance and so we give some intuition. Figure 4.4 illustrates the pay-off of a written call option on a lognormally distributed asset and the probability distribution of the payoff.

The variance (or variability) of the pay-off is very high because of the large spike of probability which corresponds to all the asset prices below the strike price. The hedge portfolio we have just described removes this spike and so reduces the variance of the pay-off.

This technique of creating a hypothetical asset which is perfectly negatively correlated with the original asset is called antithetic variance reduction and the created asset is called an antithetic variate. Implementation of this technique is very simple, for example, consider pricing a European call option. Our simulated pay-offs are

$$C_{T,j} = \max(0, S\exp(\nu T + \sigma\sqrt{T}(\varepsilon_j)) - K) \tag{4.13}$$

We can simulate the pay-offs to the option on the perfectly negatively correlated asset as

$$\overline{C}_{T,j} = \max(0, S\exp(\nu T + \sigma\sqrt{T}(-\varepsilon_j)) - K) \tag{4.14}$$

In other words we simply replace ε_j by $-\varepsilon_j$ in the equation for the simulation of the asset. We then take the average of the two pay-offs as the pay-off for that simulation. Note that, not only do we obtain a much more accurate estimate from M pairs of $(C_{T,j}, \overline{C}_{T,j})$ than from $2M$ of $C_{T,j}$, but it is also computationally cheaper to generate the pair $(C_{T,j}, \overline{C}_{T,j})$ than two instances of $C_{T,j}$. This method also ensures that the mean of the normally

FIGURE 4.4 Pay-off of Written Call Option and Probability Distribution of the Pay-off

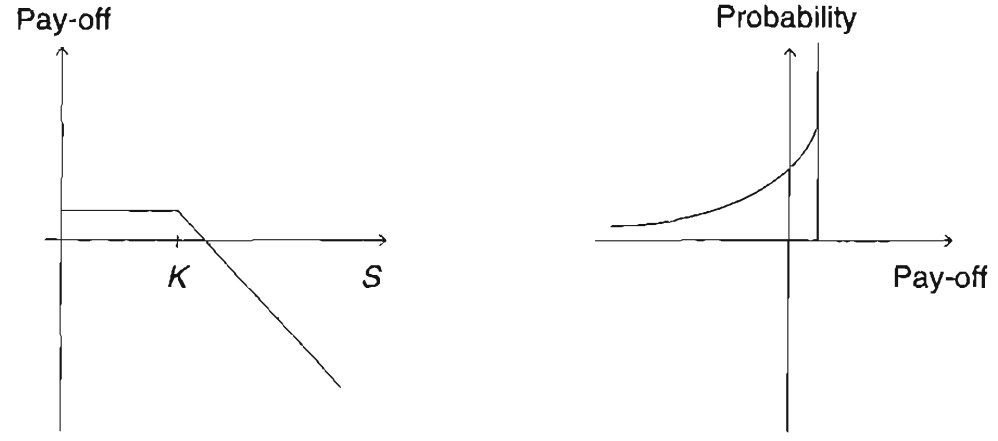


FIGURE 4.5 Pseudo-code for Monte Carlo Valuation of a European Call Option in a Black–Scholes World with Antithetic Variance Reduction

```
initialise_parameters { K, T, S, sig, r, div, N, M }

{ precompute constants }

dt = T/N
nudt = (r-div-0.5*sig^2)*dt
sigsdt = sig*sqrt(dt)
lnS = ln(S)

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

lnSt1 = lnS
lnSt2 = lnS

  for i = 1 to N do { for each time step }
    ε = standard_normal_sample
    lnSt1 = lnSt1 + nudt + sigsdt*(ε)
    lnSt2 = lnSt2 + nudt + sigsdt*(-ε)
  next i

  St1 = exp(lnSt1)
  St2 = exp(lnSt2)
  CT = 0.5*( max( 0 , St1 - K ) + max( 0 , St2 - K ) )
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

distributed samples ε is exactly zero which also helps to improve the simulation. Figure 4.5 gives a pseudo-code implementation of the Monte Carlo valuation of a European call option with antithetic variance reduction. The differences from Figure 4.2 are highlighted in bold.

Example : Pricing a European Call Option by Monte Carlo Simulation with Antithetic Variance Reduction

We price a one-year maturity, at-the-money European call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum. The simulation has one time step and 100 simulations; $K = 100$, $T = 1$ year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 1$, $M = 100$. Figure 4.6 illustrates

FIGURE 4.6 Numerical Example for Monte Carlo Valuation of a European Call Option in a Black–Scholes World with Antithetic Variance Reduction

K	T	S	sig	r	div	N	M	sum_CT	sum_CT2	SD
100	1	100	0.2	0.06	0.03	1	100	1140.37	20790.8	8.3521
dt	**nudt**	**sigsdt**	**lnS**					**call_value**	**SE**	
1	0.0100	0.2000	4.6052					10.7396	0.8352	
j	**ε**	**lnSt1**	**lnSt2**	**ST1**	**ST2**	**CT**	**CT*CT**			
1	–0.8265	4.4499	4.7805	85.62	119.16	9.5807	91.79			
2	–0.6445	4.4863	4.7441	88.79	114.90	7.4508	55.51			
3	–0.9527	4.4246	4.8057	83.48	122.21	11.1033	123.28			
4	–1.8013	4.2549	4.9754	70.45	144.81	22.4057	502.01			
5	2.4056	5.0963	4.1341	163.41	62.43	31.7067	1005.32			
95	2.3200	5.0792	4.1512	160.64	63.51	30.3210	919.36			
96	1.9226	4.9997	4.2306	148.37	68.76	24.1840	584.87			
97	–0.6575	4.4837	4.7467	88.56	115.20	7.5995	57.75			
98	–1.0324	4.4087	4.8217	82.16	124.17	12.0849	146.05			
99	–0.3316	4.5488	4.6815	94.52	107.93	3.9656	15.73			
100	–0.4677	4.5216	4.7087	91.99	110.91	5.4542	29.75			

the numerical results, the simulated asset prices are only shown for $j = 1, \ldots 5$ and $j = 95, \ldots, 100$. Note that in this example there is only one time step ($N = 1$) because we only need to simulate asset prices at the maturity date of the option.

Firstly, the constants: Δt (dt), $\nu\Delta t$ ($nudt$), $\sigma\sqrt{\Delta t}$ ($sigsdt$) and $\ln(S)$(lnS) are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{1} = 1$$

$$nudt = (r - \delta - \tfrac{1}{2}\sigma^2)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 1 = 0.01$$

$$sigsdt = \sigma\sqrt{\Delta t} = 0.2\sqrt{1} = 0.2$$

$$\ln S = \ln(S) = 4.6052$$

Then for each simulation $j = 1$ to M, where $M = 100$, $\ln(S_{1,t})$ and $\ln(S_{2,t})$ are initialised to $\ln(S) = 4.6052$. Then $\ln(S_{1,t})$ and $\ln(S_{2,t})$ are simulated, for example for $j = 1$ and $i = 1$:

$$\ln(S_{1,t}) = 4.6052 + 0.010 + 0.2 \times (-0.8265) = 4.4499$$

$$\ln(S_{2,t}) = 4.6052 + 0.010 + 0.2 \times (0.8265) = 4.7805$$

$$S_{1,T} = \exp(4.4499) = 85.62$$

$$S_{2,T} = \exp(4.7805) = 119.16$$

Computing the pay-off at maturity gives:

$$C_T = 0.5 \times (\max(0, 85.62 - 100) + \max(0, 119.16 - 100)) = 9.5807$$

The sum of the values of C_T and the squares of the values of C_T are accumulated:

$$\sum_{j=1}^{M} C_{T,j} = 1140.37 \quad \text{and} \quad \sum_{j=1}^{M} (C_{T,j})^2 = 20790.8$$

The estimate of the option value $\hat{C}_0$ (call_value) is then given by

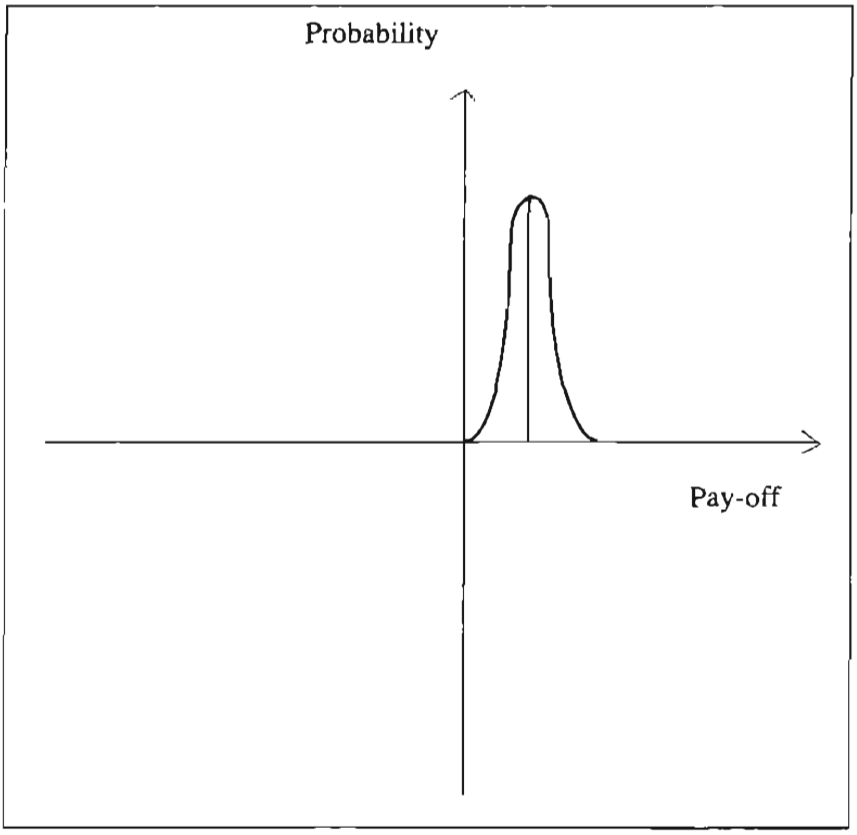
$$\hat{C}_0 = 1140.366/100 \times \exp(-0.06 \times 1) = 10.7396$$

This technique can be easily applied to virtually any Monte Carlo simulation to improve the efficiency. In the next section we describe more advanced variance reduction methods based on the hedging analogy.

4.4 CONTROL VARIATES AND HEDGING

The general approach of using hedges as control variates was first described by Clewlow and Carverhill (1994).[4] Consider the case of writing a European call option. Figure 4.4 illustrated the pay-off and its probability distribution. The distribution of this pay-off has a large standard deviation, and so if we try and estimate the call value as the mean of a number of Monte Carlo simulations then the standard error of the mean will be large.

FIGURE 4.7 Probability Distribution of the Pay-off Written Call Option after Delta Hedging



Consider the effect of delta hedging the call option. Figure 4.7 illustrates the probability distribution of the pay-off after delta hedging

The pay-off of the hedged portfolio has a much smaller standard deviation, this is of course the whole point of the delta hedge. Let us consider the mechanics of a discretely rebalanced delta hedge in detail. The delta hedge consists of a holding of $\partial C/\partial S$ in the asset which is rebalanced at discrete intervals, t_i, $i = 0, \ldots, N$. The changes in the value of the hedge as the asset price changes randomly offset the changes in the option value. Because the hedge is rebalanced at discrete time intervals it is not perfect, but for reasonably frequent rebalancing we expect it to be very good. The hedging procedure consists of selling the option, putting the premium in the bank and rebalancing the holding in the asset at discrete intervals with resultant cash flows into and out of the bank account. At the maturity date the hedge, consisting of the cash account plus the asset, closely replicates the pay-off of the option. We can express this mathematically as follows:

$$C_{t_0}e^{r(T-t_0)} - \left[\sum_{i=0}^{N}\left(\frac{\partial C_{t_i}}{\partial s} - \frac{\partial C_{t_{i-1}}}{\partial s}\right)S_{t_i}e^{r(T-t_i)}\right] = C_T + \eta \qquad (4.15)$$

where $\partial C_{t-1}/\partial S = 0$. The first term in equation (4.15) is the premium received for writing the option, inflated at the riskless rate to the maturity date, the second term represents the cash flows from rebalancing the hedge at each date t_i and the third term is the pay-off of the option C_T and the hedging error η. The expression in square brackets is the delta

hedge. Expanding the summation term in the square brackets in equation (4.15) gives

$$\frac{\partial C_{t_0}}{\partial S}S_{t_0}e^{r(T-t_0)} + \frac{\partial C_{t_1}}{\partial S}S_{t_1}e^{r(T-t_1)} + \cdots + \frac{\partial C_{t_{N-1}}}{\partial S}S_{t_{N-1}}e^{r(T-t_{N-1})} + \frac{\partial C(t_N)}{\partial s}S_{t_N}$$
$$- \frac{\partial C_{t_0}}{\partial S}S_{t_1}e^{r(T-t_1)} - \frac{\partial C_{t_1}}{\partial S}S_{t_2}e^{r(T-t_2)} - \cdots - \frac{\partial C_{t_{N-1}}}{\partial S}S_{t_N} \tag{4.16}$$

Rewriting equation (4.16) grouping terms with $\partial C_{t_i}/\partial S$ at the same time step:

$$-\frac{\partial C_{t_0}}{\partial S}(S_{t_1} - S_{t_0}e^{r\Delta t})e^{r(T-t_1)} - \frac{\partial C_{t_1}}{\partial S}(S_{t_2} - S_{t_1}e^{r\Delta t})e^{r(T-t_2)}\cdots$$
$$-\frac{\partial C_{t_{N-1}}}{\partial S}(S_{t_N} - S_{t_{N-1}}e^{r\Delta t}) + \frac{\partial C_{t_N}}{\partial S}S_{t_N} \tag{4.17}$$

If we assume that the final term in (4.17) is zero, which corresponds to not buying the final delta amount of the asset, but simply liquidating the holding from the previous rebalancing date into cash, then the hedged portfolio becomes

$$C_{t_0}e^{r(T-t_0)} + \left[\sum_{i=0}^{N-1}\frac{\partial C_{t_i}}{\partial S}(S_{t_{i+1}} - S_{t_i}e^{r\Delta t})e^{r(T-t_{i+1})}\right] = C_T + \eta \tag{4.18}$$

The expression in square brackets, which is the delta hedge, we call a delta-based martingale control variate (cv_1). This can be seen by writing it as follows:

$$cv_1 = \sum_{i=0}^{N-1}\frac{\partial C_{t_i}}{\partial S}(S_{t_{i+1}} - E[S_{t_i}])e^{r(T-t_{i+1})} \tag{4.19}$$

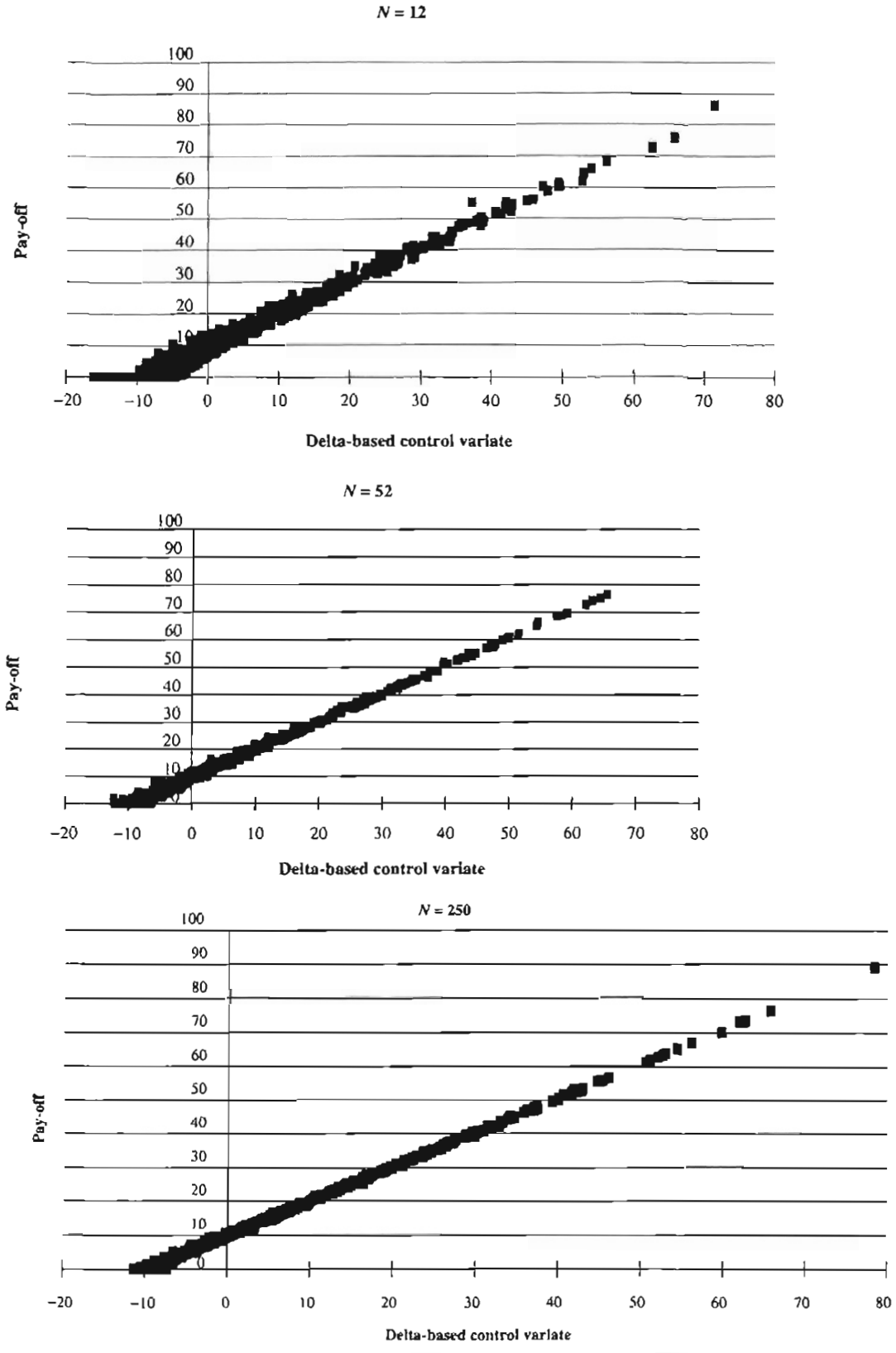
Thus, the expectation or mean of cv_1 will be zero. Rearranging equation (4.18) we have

$$C_{t_0}e^{r(T-t_0)} = C_T - \left[\sum_{i=0}^{N-1}\frac{\partial C_{t_i}}{\partial S}(S_{t_{i+1}} - E[S_{t_i}])e^{r(T-t_{i+1})}\right] + \eta \tag{4.20}$$

and we can interpret equation (4.20) as saying that the expectation of the pay-off plus the hedge is equal to the initial premium inflated to the maturity date at the riskless rate of interest. Therefore if we simulate the payoff and the hedge and compute the mean of these we will obtain an estimate of the option value but with a much smaller variance. This can be more easily visualised by plotting the payoff against the control variate cv_1 resulting from a Monte Carlo simulation. This is done in Figure 4.8 for European call option with parameter values; $K = 100$, $T = 1.0$, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$. The Monte Carlo simulation has been repeated for a number of different rebalancing intervals (or Monte Carlo time steps) corresponding to monthly, weekly and daily time steps ($N = 12$, $N = 52$, and $N = 250$) with $M = 1000$.

Figure 4.8 shows that the combination of the initial premium inflated at the riskless rate of interest plus the control variate cv_1 which is the cash accumulated by the delta hedging process is approximately equal to the pay-off of the option. The hedge gets better as the time step is decreased or equivalently as the hedge is rebalanced more often. Therefore, with this method the key to obtaining accurate prices is to have small time steps rather than a large number of simulations. In the terminology of Monte Carlo simulation, cv_1 is

FIGURE 4.8 Black–Scholes Monte Carlo Simulation with Delta-based Control Variate



a control variate, a random variable, whose expected value we know, which is correlated with the variable we are trying to estimate (in our case the option value). In this case the known mean of cv_1 is zero. In the same way as for cv_1 we can construct other control variates equivalent to other hedges. For example, a *gamma* hedge

$$cv_2 = \sum_{i=0}^{N-1} \frac{\partial^2 C_{t_i}}{\partial S^2}((\Delta S_{t_i})^2 - E[(\Delta S_{t_i})^2])e^{r(T-t_{i+1})} \tag{4.21}$$

where $\mathrm{E}[(\Delta S_{t_i})^2] = S_{t_i}^2(e^{(2r+\sigma^2)\Delta t_i} - 2e^{r\Delta t_i} + 1)$.

For the general case of a European option paying off C_T at time T, setting $t_0 = 0$ and with m control variates, equation (4.20) becomes

$$C_0 e^{rT} = C_T - \sum_{k=1}^{m} \beta_k cv_k + \eta \tag{4.22}$$

where the β factors are included to account for the sign of the hedge, for errors in the hedges due to the discrete rebalancing and only having approximate hedge sensitivities (i.e. *delta*, *gamma*, etc.). This is important for the practical implementation of this method. In reality we will be using Monte Carlo to value an option for which we do not have an analytical expression. Therefore we will not have analytical expressions for the hedge sensitivities; however, we are quite likely to have an analytical formula for a similar option. For example we might be valuing a path-dependent option which is similar to a lookback option where we have analytical expressions for continuously fixed lookback options under the Black–Scholes assumptions. Therefore we can use the hedge sensitivities from the analytical lookback formula in the control variates to value the more complex option. We describe this example in detail in the next section.

We rewrite equation (4.22) as follows:

$$C_T = \beta_0 + \sum_{k=1}^{m} \beta_k cv_k + \eta \tag{4.23}$$

where $\beta_0 = C_0 e^{rT}$ is the forward price of the option. We can interpret equation (4.23) as a linear equation relating the pay-off of the option to the control variates via the β coefficients. If we perform M simulations we can regard the pay-offs and control variates $(C_{T,j}, cv_{1,j}, \ldots, cv_{m,j};\ j = 1, \ldots, M)$ as samples from this linear relationship with noise. The noise comes from the discrete rebalancing and the imperfect sensitivities. We can then obtain an estimate of the "true" relationship by least-squares regression. The least-squares estimate of the β is

$$\beta = (X'X)^{-1}X'Y \tag{4.24}$$

where $\beta = (\beta_0, \beta_1, \ldots, \beta_m)$, X is a matrix whose rows correspond to each simulation and are $(1, cv_{1,j}, \ldots, cv_{m,j})$ and Y is the vector of simulated pay-offs (the "dash" denotes transpose). The matrices $X'X$ and $X'Y$ can be accumulated as the simulation proceeds as follows:

$$(X'X)_{k,l,j+1} = (X'X)_{k,l,j} + cv_{k,j+1}cv_{l,j+1} \tag{4.25}$$

$$(X'Y)_{k,j+1} = (X'Y)_{k,j} + cv_{k,j+1}C_{T,j+1} \tag{4.26}$$

where k and l index the rows and columns of the matrix and j is the time step as usual. It is important to note that since the pay-offs and control variates are not jointly normally distributed then the estimate of β will be biased. This is particularly important for the forward value of the option β_0, as we do not want biased estimates of the option value. This problem is easily overcome by precomputing the β_k; $k = 1, \ldots, m$ by the least-squares regression method or fixing them at some appropriate value for the type of hedge. All options can then be priced, keeping the β's fixed, by simply taking the mean of the hedged portfolio under a different set of simulated paths. This is our recommended method for implementing this technique.

There is one other subtle but important aspect of the hedge control variate idea. If we form a control variate delta hedge for a variable which in the analytical model is stochastic and which in the simulation is following the same process as in the model, then the control variate hedge is simply replicating the model option. In this case it is much more efficient to form a static hedge portfolio which is long the option we want to price and short the analytical model option. We then value the difference between the two options using the Monte Carlo simulation which has much smaller variance than the option we want to price; we use this idea in section 4.9 to value an Asian option. However, a control variate hedge for a variable which is a constant parameter in the analytical model, but which is stochastic in the simulation, cannot be simplified. The analytical model does not price any possible pay-offs due to randomness of this variable. However, using the sensitivity from the analytical model in the simulation will approximately hedge the risk and therefore help reduce the variability of the Monte Carlo estimate. We use this method in section 4.10 to value a lookback option under stochastic volatility.

4.5 MONTE CARLO SIMULATION WITH CONTROL VARIATES

In this section we illustrate the use of control variates with a series of examples based on a European call option. For our first example we consider the Monte Carlo valuation of a European call option with a *delta*-based control variate. Figure 4.9 gives the pseudo-code algorithm.

The code which has been added from the simple Monte Carlo example in Figure 4.2 is highlighted in bold. Notice also that the method of simulating the asset price is slightly different. Since we need the asset price at each time step we simulate this directly rather than its natural logarithm. The variable **erddt** allows us to compute $E[S_i]$ in equation (4.19) efficiently. We set **beta1** $= -1$ which is the appropriate value for this example where we have the exact delta. The **delta** variable is computed as the Black–Scholes delta, by the function Black_ Scholes_delta $(S_i, t, K, T, \sigma, r, \delta)$, at the start of the time step, i.e. before the asset price has been evolved and the control variate is then accumulated after the asset price has been evolved . The pay-off of the hedged portfolio is computed after the end of the time step loop and at the end the mean and standard error of this are computed. Since the mean of the control variate is zero, the mean gives us an estimate of the option price, but the standard error is greatly reduced by the control variate hedge. Figure 4.10 gives a numerical example.

FIGURE 4.9 Pseudo-code for Monte Carlo Valuation of a European Call Option in a Black–Scholes World with a *Delta*-based Control Variate

```
initialise_parameters { K, T, S, sig, r, div, N, M }

{ precompute constants }

dt = T/N
nudt = (r-div-0.5*sig^2)*dt
sigsdt = sig*sqrt(dt)
erddt = exp((r-div)*dt)

beta1 = -1

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

  St = S
  cv = 0

  for i = 1 to N do { for each time step }

   t = (i-1)*dt
   delta = Black_Scholes_delta(St,t;K,T,sig,r,div)
   ε = standard_normal_sample
   Stn = St*exp( nudt + sigsdt*ε )
   cv = cv + delta*(Stn-St*erddt)
   St = Stn
  next i

  CT = max( 0 , St - K ) + beta1*cv
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

Example : Pricing a European Call Option by Monte Carlo Simulation with a Delta-based Control Variate

We price a one-year maturity, at-the-money European call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum. The simulation has 10 time steps and 100 simulations; $K = 100$, $T = 1$

FIGURE 4.10 Monte Carlo Valuation of a European Call Option in a Black–Scholes World with a *Delta*-based Control Variate

K	T	S	sig	r	div	N	M		sum_CT		sum_CT2		SD
100	1	100	0.2	0.06	0.03	10	100		963.1277		9670.3		1.879193
dt	nudt	sigsdt	erddt		beta1				call_value		SE		
0.1	0.0010	0.0632	1.0030		-1				9.070396		0.1879		
j=100													
i	0	1	2	3	4	5	6	7	8	9	10		
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1		
ε	0	0.3766	0.4430	0.3302	0.2286	−0.6488	1.2000	−1.0518	−0.8125	1.0052	−1.1513		
St	100	102.51	105.53	107.87	109.55	105.25	113.66	106.48	101.22	107.97	100.49	CT	CT*CT
delta	0.5810	0.6264	0.6834	0.7304	0.7690	0.6944	0.8686	0.7536	0.5941	0.8991	1.0000	7.0029	49.04
cv	0.0000	1.2853	2.9830	4.3621	5.3525	1.7938	7.4153	0.8576	−3.3255	0.5050	−6.5133		

year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 10$, $M = 100$. Figure 4.10 illustrates the numerical results for the simulation of the path for $j = 100$.

Firstly, the constants; Δt (dt), $\nu\Delta t$ ($nudt$), $\sigma\sqrt{\Delta t}$ ($sigsdt$), $\exp((r-\delta)\Delta t)$ ($erddt$) and β_1 ($beta1$) are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{10} = 0.1$$

$$nudt = \left(r - \delta - \tfrac{1}{2}\sigma^2\right)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 0.1 = 0.001$$

$$sigsdt = \sigma\sqrt{\Delta t} = 0.2\sqrt{0.1} = 0.0632$$

$$erddt = \exp(-(r-\delta)\Delta t) = \exp(-(0.06 - 0.03) \times 0.1) = 1.0030$$

$$lnS = \ln(S) = 4.6052$$

$$\beta_1 = -1$$

Then for each simulation $j = 1$ to M, where $M = 100$, S_t is initialised to $S = 100$ and the control variate $cv = 0$. Then for each time step $i = 1$ to N, where $N = 10$, delta is computed, S_t is simulated and the control variate cv is accumulated. For example for $j = 100$ and $i = 0$, $delta = 0.58101$, this is the $delta$ which is used at $i = 1$ for accumulating the control variate. At $i = 1$ we have:

$$S_t = S \times \exp(nudt + sigsdt \times \varepsilon)$$

$$= 100 \times \exp(0.0010 + 0.06325 \times 0.37656) = 102.513$$

$$cv = cv + \text{delta} \times (S_t - S \times erddt) = 0 + 0.5810 \times (102.513 - 100 \times 1.0030) = 1.2853$$

After the i loop we have

$$C_T = \max(0, S_T - K) + \beta_1 \times cv$$

$$= \max(0, 100.49 - 100) + (-1) \times (-6.5133) = 7.0029$$

The sum of the values of C_T and the squares of the values of C_T are accumulated in sum_CT and sum_CT2, giving sum_CT = 963.128 and sum_CT2 = 9670.3. The estimate of the option value is then given by

$$\hat{C}_0 = \text{sum\_CT/M} \times \exp(-r \times T) = 963.128/100 \times \exp(-0.06 \times 1) = 9.0704$$

It is straightforward to combine the antithetic and control variate methods, we simply accumulate control variates for the standard and antithetic asset paths. Figure 4.11 illustrates the pseudo-code for combining antithetics with delta-based control variates for a European call option. The lines which have been added from Figure 4.9 are highlighted in bold. Note that in the calculation of the pay-off of the hedged portfolio (CT) both cv1 and cv2 are multiplied by beta1 and then added together. Therefore we do not need separate variables for the standard and antithetic control variates, they could both be accumulated in cv1. This method is used in the final example.

Example : Pricing a European Call Option by Monte Carlo Simulation with Antithetic and Delta-based Control Variates

We price a one-year maturity, at-the-money European call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is

FIGURE 4.11 Pseudo-code for Monte Carlo Valuation of a European Call Option in a Black–Scholes World with Antithetic and *Delta*-based Control Variates

```
initialise_parameters { K, T, S, sig, r, div, N, M }

{ precompute constants }

dt = T/N
nudt = (r-div-0.5*sig^2)*dt
sigsdt = sig*sqrt(dt)
erddt = exp((r-div)*dt
beta1 = -1

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

  St1 = S
  St2 = S
  cv1 = 0
  cv2 = 0

  for i = 1 to N do { for each time step }

    t = (i-1)*dt
    delta1 = Black_Scholes_delta(St1,t;K,T,sig,r,div)
    delta2 = Black_Scholes_delta(St2,t;K,T,sig,r,div)
    ε = standard_normal_sample
    Stn1 = St1*exp( nudt + sigsdt*(ε) )
    Stn2 = St2*exp( nudt + sigsdt*(-ε) )
    cv1 = cv1 + delta1*(Stn1-St1*erddt)
    cv2 = cv2 + delta2*(Stn2-St2*erddt)
    St1 = Stn1
    St2 = Stn2

  next i

 CT = 0.5*( max( 0 , St1 - K ) + beta1*cv1 +
            max( 0 , St2 - K ) + beta1*cv2 )
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum. The simulation has 10 time steps and 100 simulations: $K = 100$, $T = 1$ year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 10$, $M = 100$. Figure 4.12 illustrates the numerical results for the simulation of the path for $j = 100$.

FIGURE 4.12 Monte Carlo Valuation of a European Call Option in a Black–Scholes World with Antithetic and *Delta*-based Control Variates

K	T	S	sig	r	div	N	M		sum_CT		sum_CT2		SD
100	1	100	0.2	0.06	0.03	10	100		962.75		9597.3		1.7153
dt	nudt	sigsdt	erddt		beta1				call_value		SE		
0.1	0.0010	0.0632	1.0030		−1				9.0669		0.1715		
j = 100													
i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*		
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1		
ε	0	0.7987	0.0516	0.4174	−0.2495	0.2289	−0.2836	−0.2831	−1.5826	−0.5560	0.0549		
St1	100	105.29	105.74	108.67	107.08	108.75	106.92	105.13	95.21	92.01	92.43	CT	CT*CT
St2	100	95.17	94.95	92.57	94.14	92.88	94.66	96.46	106.72	110.65	110.38	7.3631	54.22
delta1	0.581012	0.6760	0.6871	0.7442	0.7242	0.7677	0.7450	0.7171	0.3291	0.1077	0.0000		
delta2	0.581012	0.4774	0.4625	0.3922	0.4147	0.3594	0.3866	0.4201	0.7946	0.9506	1.0000		
cv1	0	2.8967	2.9868	4.7865	3.3569	4.3332	2.6804	1.1059	−6.2332	−7.3799	−7.3653		
cv2	0	−2.9812	−3.2203	−4.4537	−3.9483	−4.5882	−4.0503	−3.4620	0.7272	3.5954	3.0194		

The calculations are identical for {delta1, Stn1, cv1} and {delta2, Stn2, cv2} as for {delta, Stn, cv} in Figure 4.10. The only other difference is the computation of the pay-off:

$$C_T = 0.5 \times (\max(0, S_{1,t} - K) + \beta_1 \times cv_1 + \max(0, S_{2,t} - K) + \beta_1 \times cv_2)$$

The final example using a European call option combines antithetic, *delta*- and *gamma*-based control variates, the pseudo-code appearing in Figure 4.13.

FIGURE 4.13 Pseudo-code for Monte Carlo Valuation of a European Call Option in a Black–Scholes World with Antithetic, *Delta*- and *Gamma*-based Control Variates

```
initialise_parameters { K, T, S, sig, r, div, N, M }

{ precompute constants }

nudt = (r-div-0.5*sig^2)*dt
sigsdt = sig*sqrt(dt)
erddt = exp((r-div)*dt
egamma = exp((2*(r-div)+sig^2)*dt)-2*erddt+1
beta1 = -1
beta2 = -0.5

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

 St1 = S
 St2 = S
 cv1 = 0
 cv2 = 0

 for i = 1 to N do { for each time step }

  { compute hedge sensitivities }
  t = (i-1)*dt
  delta1 = Black_Scholes_delta(St1,t;K,T,sig,r,div)
  delta2 = Black_Scholes_delta(St2,t;K,T,sig,r,div)
  gamma1 = Black_Scholes_gamma(St1,t;K,T,sig,r,div)
  gamma2 = Black_Scholes_gamma(St2,t;K,T,sig,r,div)

  { evolves asset prices }
  ε = standard_normal_sample
  Stn1 = St1*exp( nudt + sigsdt*(ε) )
  Stn2 = St2*exp( nudt + sigsdt*(-ε) )

  { accumulate control variates }
  cv1 = cv1 + delta1*(Stn1-St1*erddt) +
              delta2*(Stn2-St2*erddt)
  cv2 = cv2 + gamma1*((Stn1-St1)^2-St1^2*egamma) +
              gamma2*((Stn2-St2)^2-St2^2*egamma)
  St1 = Stn1
  St2 = Stn2
next i
```

FIGURE 4.13 *(continued)*

```
CT = 0.5*( max( 0 , St1 - K ) + max( 0 , St2 - K ) +
        beta1*cv1 + beta2*cv2 )

 sum_CT = sum_CT + CT
 sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

Example : Pricing a European Call Option by Monte Carlo Simulation with Antithetic, *Delta*- and *Gamma*-based Control Variates

We price a one-year maturity, at-the-money European call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum. The simulation has 10 time steps and 100 simulations; $K = 100$, $T = 1$ year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 10$, $M = 100$. Figure 4.14 illustrates the numerical results for the simulation of the path for $j = 100$. The calculations are very similar to the previous examples (compare also the pseudo-code implementations).

Table 4.1 illustrates the typical standard errors and computation times which can be achieved for example by the application of antithetic, delta- and gamma-based control variates to the valuation of a standard European call option in a Black–Scholes world.

TABLE 4.1 Typical Standard Errors and Relative Computation Times for the Monte Carlo Valuation of a European Call Option in a Black–Scholes World with Antithetic, *Delta*- and *Gamma*-based Control Variates

Strike price	100	
Time to maturity	1 year	
Initial asset price	100	
Volatility	20%	
Riskless interest rate	6%	
Continuous dividend yield	3%	
Number of time steps	52	
Number of simulations	1000	
Standard European call value	9.1352	
	Standard error	Relative computation time
Simple estimate	0.4348	1.00
With antithetic variate	0.2253	1.29
With control variates	0.0072	3.64
Combined variates	0.0048	6.43

FIGURE 4.14 Monte Carlo Valuation of a European Call Option in a Black–Scholes World with *Delta*- and *Gamma*-based Control Variates

K	T	S	sig	r	div	N	M		sum_CT		sum_CT2		SD
100	1	100	0.2	0.06	0.03	10	100		981.87		9662.2		0.4390
dt	**nudt**	**sigsdt**	**erddt**	**egamma**	**beta1**	**beta2**			**call_value**		**SE**		
0.1	0.0010	0.063246	1.0030	0.004041	−1	−0.5			9.2469		0.043898		
j = 100													
i	0	1	2	3	4	5	6	7	8	9	10		
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1		
ε	0	−0.0944	−1.4005	−1.0658	0.4191	−0.9075	2.9757	0.0462	−0.5343	−1.3023	−0.7867		
St1	100	99.50	91.16	85.30	87.68	82.87	100.14	100.53	97.29	89.68	85.42	CT	CT*CT
St2	100	100.70	110.14	117.93	114.96	121.88	101.07	100.88	104.45	113.53	119.44	10.93887	119.66
delta1	0.581012	0.5680	0.3754	0.2246	0.2517	0.1229	0.5603	0.5683	0.4198	0.0501	0.0000		
delta2	0.581012	0.5917	0.7589	0.8656	0.8476	0.9284	0.5886	0.5805	0.7208	0.9785	1.0000		
cv1	0	-0.2308	0.2653	3.6309	1.2304	5.5206	−12.0463	−12.2874	−12.4048	−9.3998	−4.1767		
gamma1	0.018762	0.0201	0.0229	0.0208	0.0233	0.0173	0.0307	0.0353	0.0447	0.0182	0.0000		
gamma2	0.018762	0.0196	0.0148	0.0097	0.0121	0.0066	0.0299	0.0349	0.0355	0.0063	0.0000		
cv2	0	−1.5026	0.0338	0.2245	−0.7289	−0.9820	6.1436	3.6703	1.6087	3.8441	3.4760		

In this example the total standard error is reduced by a factor of 90. To achieve this order of variance reduction in the simple Monte Carlo method would require increasing the number of simulations by a factor of 8100, that is, 8.1 million simulations with a computation time of approximately 3.15 hours. However, this is a slightly unrealistic example because we have the delta and gamma analytically, and so the hedge works perfectly in the limit as the time step is decreased to zero. In following sections we describe more realistic examples.

4.6 COMPUTING HEDGE SENSITIVITIES

The standard hedge sensitivities, *delta, gamma, vega, theta* and *rho* can be computed by approximating them by finite difference ratios;

$$delta = \frac{\partial C}{\partial S} \approx \frac{C(S + \Delta S) - C(S - \Delta S)}{2\Delta S} \tag{4.27}$$

$$gamma = \frac{\partial^2 C}{\partial S^2} \approx \frac{C(S + \Delta S) - 2C(S) + C(S - \Delta S)}{\Delta S^2} \tag{4.28}$$

$$vega = \frac{\partial C}{\partial \sigma} \approx \frac{C(\sigma + \Delta\sigma) - C(\sigma - \Delta\sigma)}{2\Delta\sigma} \tag{4.29}$$

$$theta = \frac{\partial U}{\partial t} \approx \frac{C(t + \Delta t) - C(t)}{\Delta t} \tag{4.30}$$

$$rho = \frac{\partial C}{\partial r} \approx \frac{C(r + \Delta r) - C(r - \Delta r)}{2\Delta r} \tag{4.31}$$

where $C(S + \Delta S)$ is the Monte Carlo estimate using an initial asset price of $S + \Delta S$, and ΔS is a small fraction of S, e.g. $\Delta S = 0.001S$ and the other $C(.)$'s are defined similarly. Note that every price $C(.)$ in equations (4.27)–(4.31) should be computed using the same set of random numbers. If this is not done then the random error in the prices from the Monte Carlo simulation can be a large proportion of the price differences in the numerator of the finite difference ratios leading to very large errors in the sensitivity estimates. By using the same random numbers the pricing errors will tend to cancel out.

A more efficient way to compute *delta* and from this *gamma* is by applying the discounted expectations approach. We can express the standard European call *delta* as follows:

$$delta = \frac{\partial C}{\partial S} = \frac{\partial}{\partial S}\left(e^{-rT}E\left[(S_T - K)\mathbf{1}_{S_T > K}\right]\right) \tag{4.32}$$

where $S_T = S\exp(\nu T + \sigma z_T)$ and $\mathbf{1}_{S_T > K}$ is the indicator function which is one if $S_T > K$ and zero otherwise. Substituting S_T in equation (4.32) and differentiating we obtain

$$delta = e^{-rT}E[\exp(\nu T + \sigma z_T)\mathbf{1}_{S_T > K}] \tag{4.33}$$

So to compute *delta* by Monte Carlo simulation we simulate the asset price as usual and compute the discounted expectation of an instrument which pays off $\exp(\nu T + \sigma z_T)$ if $S_T > K$ and zero otherwise. Figure 4.15 gives a pseudo-code implementation of this method.

FIGURE 4.15 Pseudo-code for Monte Carlo Calculation of a European Call Option Delta in a Black–Scholes World

```
initialise_parameters ( K, T, S, sig, r, div, M )

{ precompute constants }

dt = T
nudt = (r-div-0.5*sig^2)*dt
sigsdt = sig*sqrt(dt)
lnS = ln(S)

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

  ε = standard_normal_sample
  e = exp( nudt + sigsdt*ε )

  ST = S*e

  if ( ST > K ) then
    CT = e
  else
    CT = 0

  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

delta_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

Example : Computing a European Call Option Delta by Monte Carlo Simulation

We compute the delta of a one year maturity, at-the-money European call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum. The simulation has one time step and 100 simulations; $K = 100$, $T = 1$ year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 1$, $M = 100$. Figure 4.16 illustrates the numerical results for the simulation of paths $j = 1, \ldots, 5$ and $95, \ldots, 100$.

Firstly, the constants; $\Delta t (dt)$, $\nu \Delta t$ $(nudt)$, $\sigma\sqrt{\Delta t}$ $(sigsdt)$ are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{1} = 1$$

$$nudt = (r - \delta - \tfrac{1}{2}\sigma^2)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 1 = 0.01$$

$$sigsdt = \sigma\sqrt{\Delta t} = 0.2\sqrt{1} = 0.2$$

FIGURE 4.16 Monte Carlo Calculation of a European Call Option Delta in a Black–Scholes World

K	T	S	sig	r	div	N	M	sum_CT	sum_CT2	SD
100	1	100	0.2	0.06	0.03	1	100	60.3992	77.98	0.6097
dt	**nudt**	**sigsdt**						**delta_value**	**SE**	
1	0.0100	0.2000						0.5688	0.0610	

j	ε	e	ST	CT	CT*CT
1	–0.8265	0.8562	85.6152	0.0000	0.0000
2	–0.6445	0.8879	88.7892	0.0000	0.0000
3	–0.9527	0.8348	83.4816	0.0000	0.0000
4	–1.8013	0.7045	70.4504	0.0000	0.0000
5	2.4056	1.6341	163.4134	1.6341	2.6704
95	2.3200	1.6064	160.6420	1.6064	2.5806
96	1.9226	1.4837	148.3680	1.4837	2.2013
97	–0.6575	0.8856	88.5599	0.0000	0.0000
98	–1.0324	0.8216	82.1617	0.0000	0.0000
99	–0.3316	0.9452	94.5232	0.0000	0.0000
100	–0.4677	0.9199	91.9860	0.0000	0.0000

Then for each simulation $j = 1$ to M, where $M = 100$, the exponential term e is simulated, S_T and C_T are computed, and the sums sum_CT and sum_CT2 are accumulated. For $j = 1$ we have

$$e = \exp(nudt + sigsdt \times \varepsilon) = \exp(0.010 + 0.2 \times (-0.8265)) = 0.8562$$

$$S_T = S \times e = 100 \times 0.8562 = 85.62$$

$S_T < K$ therefore $C_T = 0$.

For $j = 5$ we have

$$e = \exp(nudt + sigsdt \times \varepsilon) = \exp(0.01 + 0.2 \times (2.4056)) = 1.6341$$

$$S_T = S \times e = 100 \times 1.6341 = 163.41$$

$S_T > K$ therefore $C_T = e = 1.6341$.

The sum of the values of C_T and the squares of the values of C_T are accumulated in sum_CT and sum_CT2, giving sum_CT = 60.399 and sum_CT2 = 78.0. The estimate of the delta value is then given by

$$delta = \text{sum\_CT/M} \times \exp(-r \times T) = 60.399/100 \times \exp(-0.06 \times 1) = 0.5688$$

The antithetic and control variate methods can be applied in the same way as for the Monte Carlo valuation of the option itself.

This technique cannot be used for the calculation of *gamma* because differentiating equation (4.33) again leads to the expectation of a Dirac delta function which cannot easily be evaluated by Monte Carlo simulation. We can, however, use a finite difference ratio in terms of *delta*

$$gamma = \frac{\partial^2 C}{\partial S^2} \approx \frac{delta(S + \Delta S) - delta(S - \Delta S)}{2\Delta S} \tag{4.34}$$

4.7 MULTIPLE STOCHASTIC FACTORS

One of the main uses of Monte Carlo simulation is for pricing options under multiple stochastic factors. For example pricing options whose pay-off depends on multiple asset prices, or with stochastic volatility or interest rates. For example, consider a European spread option on the difference between two assets (e.g. stock indices) S_1 and S_2, which follow GBM:[5]

$$dS_1 = (r - \delta_1)S_1\,dt + \sigma_1 S_1\,dz_1 \tag{4.35}$$

$$dS_2 = (r - \delta_2)S_2\,dt + \sigma_2 S_2\,dz_2 \tag{4.36}$$

The first complication we have is that it is quite likely that we will want S_1 and S_2 to be correlated to some degree ρ. That is, the Brownian motions dz_1 and dz_2 have instantaneous correlation ρ ($dz_1.dz_2 = \rho dt$). In order to price the option by simulation we use the solutions of the SDEs to simulate the asset prices, as in section 4.2:

$$S_{1,T} = S_1 \exp(\nu_1 T + \sigma_1 z_{1,T}) \tag{4.37}$$

where $\nu_1 = r - \delta_1 - \frac{1}{2}\sigma_1^2$ and

$$S_{2,T} = S_2 \exp(\nu_2 T + \sigma_2 z_{2,T}) \tag{4.38}$$

where $\nu_2 = r - \delta_2 - \frac{1}{2}\sigma_2^2$. However, here we need to generate the variates z_1 and z_2 from a standard bivariate normal distribution with correlation ρ. This is easily achieved by generating independent standard normal variates ε_1 and ε_2 and combining them as follows:

$$z_1 = \varepsilon_1 \tag{4.39}$$

$$z_2 = \rho\varepsilon_1 + \sqrt{1-\rho^2}\varepsilon_2 \tag{4.40}$$

The general procedure for generating n correlated normal variates is described in section 4.11.

The Monte Carlo procedure is exactly the same as that for the standard European call in section 4.2 except that we simulate the two asset processes and from this the pay-off of the spread option $(\max(0, S_1 - S_2 - K))$. Figure 4.17 gives the pseudo-code implementation.

Example : Pricing a European Spread Call Option by Monte Carlo Simulation

We price a one-year maturity, European spread call option with a strike price of 1, current asset prices of 100, volatilities of 20 and 30 per cent, continuous dividend yields of 3 and 4 per cent and a correlation of 50 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum and the simulation has one time step and 100 simulations, i.e. $K = 1$, $T = 1$, $S_1 = 100$, $S_2 = 110$, $\sigma_1 = 0.20$, $\sigma_2 = 0.30$, $d_1 = 0.03$, $d_2 = 0.04$, $\rho = 0.50$, $r = 0.06$, $N = 1$, $M = 100$. Figure 4.18 illustrates the results of the calculations for the simulation of paths $j = 1, \ldots, 5$ and $95, \ldots, 100$.

Firstly, the constants; Δt (dt), $\nu_1 \Delta t$ $(nu1\,dt)$, $\nu_2 \Delta t$ $(nu2\,dt)$, $\sigma_1\sqrt{\Delta t}$ $(sig\,1\,sdt)$, $\sigma_2\sqrt{\Delta t}$ $(sig\,2\,sdt)$, $\sqrt{1-\rho^2}$ $(srho)$ are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{1} = 1$$

$$nu1dt = (r - \delta_1 - \tfrac{1}{2}\sigma_1^2)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 1 = 0.01$$

$$nu2dt = (r - \delta_2 - \tfrac{1}{2}\sigma_2^2)\Delta t = (0.06 - 0.04 - 0.5 \times 0.3^2) \times 1 = -0.025$$

$$sig1sdt = \sigma_1\sqrt{\Delta t} = 0.2\sqrt{1} = 0.2$$

$$sig2sdt = \sigma_2\sqrt{\Delta t} = 0.3\sqrt{1} = 0.3$$

$$srho = \sqrt{1-\rho^2} = \sqrt{1-0.5^2} = 0.8660$$

For each simulation $j = 1$ to $M (M = 100)$, S_1 and S_2 are simulated. For example, for $j = 1$ we have

$$\varepsilon_1 = -0.8265, \varepsilon_2 = -0.0833$$

$$z_1 = \varepsilon_1 = -0.8265$$

FIGURE 4.17 Pseudo-code for Monte Carlo Valuation of a European Spread Option in a Black–Scholes World

```
initialise_parameters
  { K, T, S1, S2, sig1, sig2, div1, div2, rho, r, N, M }

{ precompute constants }

N = 1 { no path dependency }

dt = T/N
nu1dt = (r-div1-0.5*sig1^2)*dt
nu2dt = (r-div2-0.5*sig2^2)*dt
sig1sdt = sig1*sqrt(dt)
sig2sdt = sig2*sqrt(dt)
srho = sqrt( 1 - rho^2 )

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

  St1 = S1
  St2 = S2

  for i = 1 to N do { for each time step }
    ε1 = standard_normal_sample
    ε2 = standard_normal_sample
    z1 = ε1
    z2 = rho * ε1 + srho * ε2
    St1 = St1*exp( nu1dt + sig1sdt*z1 )
    St2 = St2*exp( nu2dt + sig2sdt*z2 )
  next i

  CT = max( 0 , St1 - St2 - K )
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

$$z_2 = \rho \times \varepsilon_1 + srho \times \varepsilon_2 = 0.5 \times (-0.8265) + 0.8660 \times (-0.0833) = -0.4854$$

$$S_{1,T} = S_1 \times \exp(nu1dt + sig1sdt \times z_1)$$

$$= 100 \times \exp(0.0100 + 0.2 \times (-0.8265)) = 85.615$$

$$S_{2,T} = S_2 \times \exp(nu2dt + sig2sdt \times z_2)$$

$$= 110 \times \exp(-0.0250 + 0.3 \times (-0.4854)) = 92.746$$

FIGURE 4.18 Monte Carlo Valuation of a European Spread Option in a Black–Scholes World

K	T	S1	S2	sig1	sig2	div1	div2	rho	r	N	M			
1	1	100	110	0.2	0.3	0.03	0.04	0.50	0.06	1	100			
dt	**nu1dt**	**nu2dt**	**sig1sdt**	**sig2sdt**	**srho**					**sum_CT**		**sum_CT2**		**SD**
1.0000	0.0100	−0.0250	0.2000	0.3000	0.8660					742.97		21844.2		12.09319
j	ε1	ε2	z1	z2	St1	St2	CT	CT*CT		**call_value**		**SE**		
1	−0.8265	−0.0833	−0.8265	−0.4854	85.62	92.75	0.0000	0.00		6.9970		1.2093		
2	−0.6445	0.8050	−0.6445	0.3748	88.79	120.05	0.0000	0.00						
3	−0.9527	−1.3859	−0.9527	−1.6766	83.48	64.88	17.6036	309.89						
4	−1.8013	0.9632	−1.8013	−0.0665	70.45	105.16	0.0000	0.00						
5	2.4056	−0.5148	2.4056	0.7569	163.41	134.63	27.7798	771.71						
95	2.3200	−0.4380	2.3200	0.7807	160.64	135.60	24.0443	578.13						
96	1.9226	−0.2514	1.9226	0.7436	148.37	134.10	13.2711	176.12						
97	−0.6575	−0.0533	−0.6575	−0.3748	88.56	95.87	0.0000	0.00						
98	−1.0324	0.2024	−1.0324	−0.3409	82.16	96.85	0.0000	0.00						
99	−0.3316	0.1838	−0.3316	−0.0066	94.52	107.07	0.0000	0.00						
100	−0.4677	0.5573	−0.4677	0.2488	91.99	115.60	0.0000	0.00						

$$\text{CT} = \max(0, S_{1,T} - S_{1,T} - K) = \max(0, 85.615 - 92.746 - 1) = 0.0$$

For $j = 3$ we have

$$\varepsilon_1 = -0.9527, \qquad \varepsilon_2 = -1.3859$$

$$z_1 = \varepsilon_1 = -0.9527$$

$$z_2 = \rho \times \varepsilon_1 + srho \times \varepsilon_2 = 0.5 \times (-0.9527) + 0.8660 \times (-1.3859) = -1.6766$$

$$S_{1,T} = S_1 \times \exp(nu1dt + sig1sdt \times z_1)$$

$$= 100 \times \exp(0.0100 + 0.2 \times (-0.9527)) = 83.482$$

$$S_{2,T} = S_2 \times \exp(nu2dt + sig2sdt \times z_2)$$

$$= 110 \times \exp(-0.0250 + 0.3 \times (-1.6766)) = 64.878$$

$$C_T = \max(0, S_{1,T} - S_{2,T} - K) = \max(0, 83.482 - 64.878 - 1) = 17.604$$

The sum of the values of CT and the squares of the values of CT are accumulated in sum_CT and sum_CT2, giving sum_CT = 742.968 and sum_CT2 = 21844.2. The estimate of the option value is then given by

$$\text{call\_value} = \text{sum\_CT/M} \times \exp(-r \times T) = 742.968/100 \times \exp(-0.06 \times 1) = 6.9970$$

In the same way if we want to price an option under more general stochastic processes such as stochastic volatility and/or stochastic interest rates we simply simulate the required stochastic processes. For example, imagine we want to price the European spread option when the underlying asset prices, S_1 and S_2, follow GBM, but where the variance of returns, V_1 and V_2, of the assets follow mean reverting square root processes (see Hull and White, 1988). The SDE's for the asset prices and variances are given by equations (4.41)–(4.44) respectively

$$dS_1 = rS_1\,dt + \sigma_1 S_1\,dz_1 \tag{4.41}$$

$$dS_2 = rS_2\,dt + \sigma_2 S_2\,dz_2 \tag{4.42}$$

$$dV_1 = \alpha_1(\overline{V}_1 - V_1)\,dt + \xi_1\sqrt{V_1}\,dz_3 \tag{4.43}$$

$$dV_2 = \alpha_2(\overline{V}_2 - V_2)\,dt + \xi_2\sqrt{V_2}\,dz_4 \tag{4.44}$$

where $V_i = \sigma_i^2$, α_i is the rate of mean reversion on the variance, ξ_i is the volatility of the variance and the Wiener processes have the following correlation matrix:

$$\rho_z = \begin{vmatrix} 1 & \rho_{12} & \rho_{13} & \rho_{14} \\ \rho_{12} & 1 & \rho_{23} & \rho_{24} \\ \rho_{13} & \rho_{23} & 1 & \rho_{34} \\ \rho_{14} & \rho_{24} & \rho_{34} & 1 \end{vmatrix}$$

In this case we need to generate four correlated normal variates in order to simulate the four processes (4.41)–(4.44). Then, we simply add the simulation of the variances into the pseudo-code of Figure 4.17. The resulting pseudo-code is shown in Figure 4.19.[6] Figure 4.20 gives a numerical example, the calculations are similar to those for the previous example.

FIGURE 4.19 Pseudo-code for Monte Carlo Valuation of a European Spread Option with Stochastic Volatilities

```
initialise_parameters
  { K, T, S1, S2, sig1, sig2, div1, div2, alpha1, alpha2,
   Vbar1, Vbar2, xi1, xi2, rhoz, r, N, M }

{ precompute constants }

N = 1 { no path dependency }

dt = T/N
alpha1dt = alpha1*dt
alpha2dt = alpha2*dt
xi1sdt = xi1*sqrt(dt)
xi2sdt = xi2*sqrt(dt)
lnS1 = ln(S1)
lnS2 = ln(S2)

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

  lnSt1 = lnS1
  lnSt2 = lnS2

  for i = 1 to N do { for each time step }

   generate_correlated_normals( rhoz, z[] )

   { simulate variances first }

   Vt1 = Vt1 + alpha1dt*(Vbar1-Vt1) + xi1sdt*sqrt(Vt1)*z[3]
   Vt2 = Vt2 + alpha2dt*(Vbar2-Vt2) + xi2sdt*sqrt(Vt2)*z[4]

   { simulate asset prices }

   lnSt1 = lnSt1 + (r-div1-0.5*Vt1)*dt + sqrt(Vt1)*sdt*z[1]
   lnSt2 = lnSt2 + (r-div2-0.5*vt2)*dt + sqrt(Vt2)*sdt*z[2]

 next i

 St1 = exp(lnSt1)
 St2 = exp(lnSt2)
 CT = max( 0 , St1 - St2 - K )
 sum_CT = sum_CT + CT
 sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

FIGURE 4.20 Numerical Example for Monte Carlo Valuation of a European Spread Option with Stochastic Volatilities

K	T	S1	S2	sig1	sig2	div1	div2	alpha1	alpha2	Vbar1	Vbar2			
1	1	100	110	0.2	0.3	0.03	0.04	1.0	2.0	0.04	0.09			
xi1	xi2	rho12	rho13	rho14	rho23	rho24	rho34	r	N	M		sum_CT	sum_CT2	SD
0.05	0.06	0.50	0.20	0.01	0.01	0.30	0.30	0.06	10	100		747.42	22810.0	12.4219
dt	sdt	alpha1dt	alpha2dt	xi1sdt	xi2sdt	lnS1	lnS2					call_value	SE	
0.1000	0.3162	0.1000	0.2000	0.0158	0.0190	4.6052	4.7005					7.0389	1.2422	
j = 100														
i	0	1	2	3	4	5	6	7	8	9	10			
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1			
ε1	0	0.1834	−1.1942	0.0110	−1.3283	−1.6112	−1.1443	−0.6281	2.3196	−0.6359	1.3040			
ε2	0	−2.3608	0.8641	0.3365	−0.6725	−1.1617	0.5131	2.1820	0.8927	1.3425	0.7955			
ε3	0	0.8857	1.4512	0.5038	−0.3170	0.7995	−0.7687	0.3317	−0.8933	1.4908	1.3724			
ε4	0	0.5571	−1.8262	−1.3109	0.5054	0.5144	−1.4457	−0.9238	1.1758	0.3540	−0.9657			
z1	0	0.6911	−1.2425	0.0804	−0.6890	−1.1128	−0.2886	−1.2928	1.2354	−1.7865	0.3465			
z2	0	1.5421	−1.4429	−0.4211	−0.7095	−0.3732	−1.8632	−1.5528	1.6031	−0.4005	0.8264			
z3	0	−1.8217	−1.2013	−0.3317	−0.7780	−1.8556	−0.0723	0.7337	2.4169	−0.2034	0.1552			
z4	0	−0.9626	1.0477	0.7938	−1.4369	−1.3035	−0.3854	1.3337	1.0752	1.1226	2.1615			
Vt1	0.04	0.0112	0.0049	0.0032	0.0054	0.0204	0.0213	0.0347	0.0735	0.0669	0.0667			
Vt2	0.09	0.0717	0.0953	0.1093	0.0782	0.0558	0.0553	0.0876	0.1085	0.1261	0.1599	ST		
lnSt1	4.6052	4.6307	4.6059	4.6102	4.5969	4.5485	4.5372	4.4622	4.5675	4.4210	4.4489	85.53588	CT	CT*CT
lnSt2	4.7005	4.8295	4.6859	4.6384	4.5738	4.5451	4.4058	4.2581	4.4216	4.3724	4.4709	87.43187	0.0000	0.00

4.8 PATH-DEPENDENT OPTIONS

An important application of Monte Carlo simulation is in pricing complex or exotic path-dependent options.[7] Simple analytical formulae exist for certain types of exotic options, these options being classified by the property that the path-dependent condition applies to the continuous path. For example, a popular class of exotic option is the barrier option. These are standard European options except that the option either ceases to exist or only comes into existence if the underlying asset price crosses a predetermined barrier level. If we assume that the underlying asset price is checked continuously for the crossing of the barrier, then simple analytical formulae exist for the price of these options. In contrast, with actual barrier options the underlying asset price is checked (fixed) at most once a day and often much less frequently. This significantly affects the price of the option since the price is much less likely to be observed crossing the barrier if the fixings occur infrequently, and it also complicates the pricing formulae. However, these options can be priced very easily by Monte Carlo simulation.

Consider pricing a daily fixed down-and-out call option. This is a particular type of barrier option which is a normal call option unless the underlying asset price observed once per day crosses the predetermined barrier level H from above, in which case the option ceases to exist. For this option we must simulate the underlying asset price for each fixing date in order to check for the crossing of the barrier. Assuming the asset price follows GBM, the simulation of the asset price takes the usual form:

$$S_{t+\Delta t} = S_t \exp(\nu \Delta t + \sigma \sqrt{\Delta t} z) \tag{4.45}$$

where we assume Δt is one day and z is a standard normal random variate as usual. The Monte Carlo simulation proceeds in exactly the same way as for a standard option, except that at each time step we check whether the asset price has crossed the barrier level H. If so then we terminate the simulation of that path and the pay-off for that path is zero. The pseudo-code is given in Figure 4.21. We can use the analytical formulae for continuously fixed barrier options to construct hedge sensitivity-based control variates as we described in earlier sections.

Example : Pricing a European Down and Out Call Option by Monte Carlo Simulation

We price a one-year maturity, at-the-money European down and out call option with the current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per cent per annum, and the barrier is at 99. The simulation has 10 time steps and 100 simulations; $K = 100$, $T = 1$ year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $H = 99$, $N = 10$, $M = 100$. Figure 4.22 illustrates the numerical results for the simulation of the path for $j = 100$.

Firstly, the constants; Δt (dt), $\nu \Delta t$ ($nudt$), $\sigma \sqrt{\Delta t}$ ($sigsdt$) are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{10} = 0.1$$

$$nudt = (r - \delta - \tfrac{1}{2}\sigma^2)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 0.1 = 0.001$$

$$sigsdt = \sigma \sqrt{\Delta t} = 0.2\sqrt{0.1} = 0.0632$$

FIGURE 4.21 Pseudo-code for Monte Carlo Valuation of a European Down and Out Call Option in a Black–Scholes World

```
initialise_parameters { K, T, S, sig, r, div, H, N, M }
{ N is the number of days in the life of the option T }

{ precompute constants }

dt = T/N
nudt = (r-div-0.5*sig*sig)*dt
sigsdt = sig*sqrt(dt)

sum_CT = 0
sum_CT2 = 0

for j = 1 to M do { for each simulation }

 St = S
 BARRIER_CROSSED = FALSE
 for i = 1 to N do { for each time step }
   ε = standard_normal_sample
   St = St*exp( nudt + sigsdt*ε )
   if ( St <= H ) then
    BARRIER_CROSSED = TRUE
    exit_loop
 next i

 if BARRIER_CROSSED then CT = 0
 else CT = max( 0 , St - K )

 sum_CT = sum_CT + CT
 sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

Then for each simulation $j = 1$ to M where $M = 100$, S_t is initialised to $S = 100$ and BARRIER_CROSSED = FALSE which indicates that the barrier has not yet been crossed. Then for each time step $i = 1$ to N, where $N = 10$, S_t is simulated and the crossing of the barrier is checked. For example for $j = 100$ and $i = 1$ we have

$$S_t = S_t \times \exp(nudt + sigsdt \times \varepsilon)$$
$$= 100 \times \exp(0.001 + 0.0632 \times 0.5087) = 103.37$$

$S_t > H$, therefore BARRIER_CROSSED is FALSE and the loop continues. For $i = 4$ we have

FIGURE 4.22 Monte Carlo Valuation of a European Down and Out Call Option in a Black–Scholes World

K	T	S	sig	r	div	H	N	M	sum_CT	sum_CT2	SD
100	1	100	0.2	0.06	0.03	99	10	100	410.49	13057.7	10.09384
dt	**nudt**	**sigsdt**							**call_value**	**SE**	
0.1	0.0010	0.0632							3.8659	1.0094	

j = 100												
i	0	1	2	3	4	5	6	7	8	9	10	
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1	
ε	0	0.5087	1.2242	–0.3409	–1.8043	–0.3742	1.6139	–0.7529	–0.1701	0.6688	–0.9505	**CT** **CT*CT**
St	100	103.37	111.81	109.53	97.82	95.62	106.01	101.18	100.20	104.63	98.62	0 0
BARRIER_CROSSED		FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	

$$S_t = S_t \times \exp(nudt + sigsdt \times \varepsilon)$$
$$= 109.531 \times \exp(0.001 + 0.0632 \times (-1.8043)) = 97.82$$

$S_t < H$, therefore BARRIER_CROSSED = TRUE and the loop terminates.
For $j = 100$ we have BARRIER_CROSSED = TRUE therefore:

$$C_T = 0$$

The sum of the values of C_T and the squares of the values of C_T are accumulated in sum_CT and sum_CT2, giving sum_CT = 410.493 and sum_CT2 = 13057.7. The estimate of the option value is then given by

call_value = sum_CT/M $\times \exp(-r \times T) = 410.493/100 \times \exp(-0.06 \times 1) = 3.8659$

4.9 AN ARITHMETIC ASIAN OPTION WITH A GEOMETRIC ASIAN OPTION CONTROL VARIATE

In this example we price a European arithmetic Asian (average price) call option.[8] This option pays the difference, if positive, between the arithmetic average of the asset price A_T and the strike price K at the maturity date T. The arithmetic average is taken on a set of observations (fixings) of the asset price S_{t_i} (which we assume follows GBM) at dates t_i; $i = 1, \ldots, N$

$$A_T = \frac{1}{N}\sum_{i=1}^{N} S_{t_i} \tag{4.46}$$

Thus the pay-off at the maturity date is

$$\max(0, A_T - K) \tag{4.47}$$

Figure 4.23 illustrates two typical asset price paths and the fixing dates.

There is no analytical solution for the price of an arithmetic Asian option; however, there is a simple analytical formula for the price of a geometric Asian option. A geometric Asian call option pays the difference if positive, between the geometric average of the asset price G_T and the strike price K at the maturity date T. The geometric average is defined as

$$G_T = \left(\prod_{i=1}^{N} S_{t_i}\right)^{1/N} \tag{4.48}$$

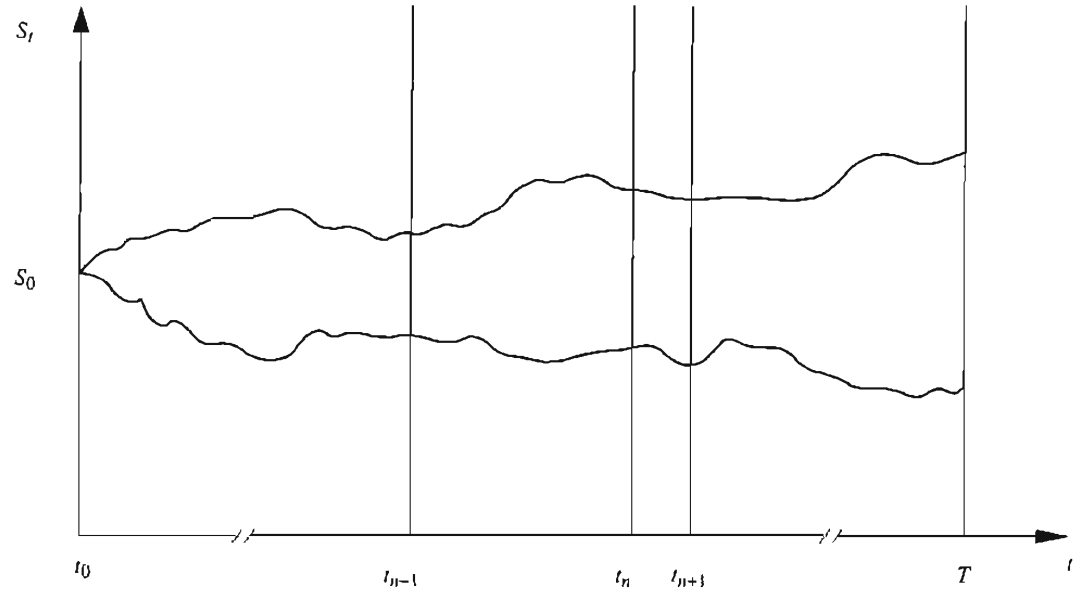
Since the geometric average is essentially the product of lognormally distributed variables then it is also lognormally distributed. Therefore the price of the geometric Asian call option is given by a modified Black–Scholes formula:

$$C_{\text{GEOMETRIC\_ASIAN}} = \exp(-rT)\left(\exp(a + \tfrac{1}{2}b)N(x) - KN(x - \sqrt{b})\right) \tag{4.49}$$

where

$$a = \ln(G_t) + \frac{N-m}{N}(\ln(S) + \nu(t_{m+1} - t) + \tfrac{1}{2}\nu(T - t_{m+1}))$$

FIGURE 4.23 Typical Asset Price Paths and Fixing Dates for an Asian Option



$$b = \frac{(N-m)^2}{N^2}\sigma^2(t_{m+1} - t) + \frac{\sigma^2(T - t_{m+1})}{6N^2}(N-m)(2(N-m)-1)$$

$$\nu = r - \delta - \tfrac{1}{2}\sigma^2, \qquad x = \frac{a - \ln(K) + b}{\sqrt{b}}$$

where G_t is the current geometric average and m is the last known fixing. The geometric Asian option makes a good static hedge style control variate for the arithmetic Asian option. Figure 4.24 shows a pseudo-code implementation of the Monte Carlo valuation of a European Asian call option with a geometric Asian call option control variate. We simulate the difference between the arithmetic and geometric Asian options or a hedged portfolio which is long one arithmetic Asian and short one geometric Asian option. This is much faster than using the delta of the geometric Asian option to generate a delta hedge control variate because we do not have to compute the delta at every time step and it is equivalent to a continuous delta hedge. Note the bold highlighted lines where we precompute the drift and volatility constant expressions required for the simulation of the asset price between the fixing dates. This increases the efficiency of the simulation significantly because these constants are used for every time step for every simulation.

Example : Pricing a European Asian Call Option by Monte Carlo Simulation with Geometric Asian Call Option Control Variate

We price a one-year maturity, European Asian call option with a strike price at 100, current asset price at 100 and volatility of 20 per cent. The continuously compounded interest rate is assumed to be 6 per cent per annum, the asset pays a continuous dividend yield of 3 per

FIGURE 4.24 Pseudo-code for Monte Carlo Valuation of a European Arithmetic Asian Call Option with a Geometric Asian call Option Control Variate

```
initialise_parameters ( K, t[], S, sig, r, div, N, M )
{ t[] is an array containing the fixing times }

{ precompute constants }

for i = 1 to N do { for each fixing }
  nudt[i] = (r-div-0.5*sig*sig)*(t[i]-t[i-1])
  sigsdt[i] = sig*sqrt(t[i]-t[i-1])
next i

sum_CT = 0
sum_CT2 = 0
for j = 1 to M do { for each simulation }

  St = S
  sumSt = 0
  productSt = 1

  for i = 1 to N do { for each fixing }
    ε = standard_normal_sample
    St = St*exp( nudt[i] + sigsdt[i]*ε )
    sumSt = sumSt + St
    productSt = productSt * St
  next i

  A = sumSt/N
  G = productSt^(1/N)
  CT = max( 0 , A - K )- max( 0 , G - K )
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

portfolio_value = sum_CT/M * exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)

{ add back in control variate value }

call_value = portfolio_value +
             geometric_Asian_call( K, t[], S, sig, r, div, N )
```

cent per annum, and their are 10 equally spaced fixing dates. The simulation has 10 time steps and 100 simulations; $K = 100$, $T = 1$ year, $S = 100$, $\sigma = 0.2$, $r = 0.06$, $\delta = 0.03$, $N = 10$, $M = 100$. Figure 4.25 illustrates the numerical results for the simulation of the path for $j = 100$.

FIGURE 4.25 Monte Carlo Valuation of a European Asian Call Option with a Geometric Asian Call Option Control Variate

K	T	S	sig	r	div	N	M	sum_CT	sum_CT2	SD
100	1	100	0.2	0.06	0.03	10	100	22.8411	17.9192	0.3373

dt	nudt	sigsdt	portfolio_value	SE
0.1	0.0010	0.063246	0.2151	0.0337

j=100													
i	0	1	2	3	4	5	6	7	8	9	10		
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1		
ε	0	2.1493	0.0430	0.3294	0.4444	−0.0765	−0.2008	1.6842	−0.0302	0.3285	0.0913	CT	CT*CT
St	100	114.68	115.10	117.64	121.12	120.65	119.25	132.79	132.67	135.59	136.51	0.2730	0.0745
sumSt	0	114.68	229.78	347.42	468.54	589.19	708.44	841.23	973.90	1109.48	1245.99	A	124.60
productSt	1	1.1E+02	1.3E+04	1.6E+06	1.9E+08	2.3E+10	2.7E+12	3.6E+14	4.8E+16	6.5E+18	8.8E+20	G	124.33

Geometric Asian Call Option

GA_a	GA_b	GA_nu	GA_x	call_value
4.6107	0.0154	0.0100	0.1684	5.5577

geometric_Asian_call
5.3426

Firstly, the constants; Δt (dt), $\nu\Delta t$ ($nudt$), $\sigma\sqrt{\Delta t}$ ($sigsdt$) are precomputed:

$$\Delta t = \frac{T}{N} = \frac{1}{10} = 0.1$$

$$nudt = (r - \delta - \tfrac{1}{2}\sigma^2)\Delta t = (0.06 - 0.03 - 0.5 \times 0.2^2) \times 0.1 = 0.001$$

$$sigsdt = \sigma\sqrt{\Delta t} = 0.2\sqrt{0.1} = 0.0632$$

Then for each simulation $j = 1$ to M where M = 100, S_t is initialised to $S = 100$, $sumSt = 0$ and $productSt = 1$. Then for each time step $i = 1$ to N, where $N = 10$, S_t is simulated and the sum and product of the asset prices at the fixing times are accumulated. For example, for $j = 100$ we have, for $i = 1$,

$$S_t = S_t \times \exp(nudt + sigsdt \times \varepsilon)$$

$$= 100 \times \exp(0.0010 + 0.06325 \times 2.14929) = 114.675$$

$$sumSt = sumSt + S_t = 0 + 114.657 = 114.675$$

$$productSt = productSt \times S_t = 1 \times 114.675 = 114.675$$

For $i = 5$:

$$S_t = S_t \times \exp(nudt + sigsdt \times e)$$

$$= 121.118 \times \exp(0.0010 + 0.06325 \times (-0.0765)) = 120.654$$

$$sumSt = sumSt + S_t = 468.539 + 120.654 = 589.193$$

$$productSt = productSt \times S_t = 1.9\text{E}+08 \times 120.654 = 2.3\text{E}+10$$

After the i loop we have

$$A_T = sumSt/N = 1245.99/10 = 124.599$$

$$G_T = productSt^{(1/N)} = (8.8\text{E}+20)^{(1/10)} = 124.326$$

$$C_T = \max(0, A - K) - \max(0, G - K)$$

$$= \max(0, 124.599 - 100) - \max(0, 124.326 - 100)$$

$$= 24.599 - 24.326 = 0.27303$$

The sum of the values of C_T and the squares of the values of CT are accumulated in sum_CT and sum_CT2 giving sum_CT = 22.8411 and sum_CT2 = 17.9192. The estimate of the portfolio value is then given by

$$\text{portfolio\_value} = \text{sum\_CT/M} \times \exp(-r \times T)$$

$$= 22.8411/100 \times \exp(-0.06 \times 1) = 0.21511$$

Finally the estimate of the option value is given by

$$\text{call\_value} = \text{portfolio\_value} + \text{geometric\_Asian\_call}(K, \{t_1, \ldots, t_N\}, S, \sigma, r, \delta, N)$$

$$= 0.21511 + 5.3426 = 5.5577$$

Table 4.2 gives the prices, standard errors and relative computation times with no variance reduction, an antithetic control variate, a geometric Asian control variate and

TABLE 4.2 Results from Pricing a European Arithmetic Asian Option by Monte Carlo Simulation

	Price	Standard error	Relative computation time
Simple Monte Carlo	5.038019	0.248236	1.00
Antithetic	5.156263	0.135463	1.23
Control variate	5.207977	0.010366	1.05
Antithetic and control variate	5.216232	0.006596	1.32

finally both an antithetic control variate and a geometric Asian control variate. The addition of the antithetic and geometric Asian control variate increase the computation time by approximately 30 per cent, but reduces the standard error by approximately 37 times. To achieve this reduction with the simple Monte Carlo would require increasing the number of simulations by $37 \times 37 = 1369$ times with a roughly equivalent increase in the computation time.

4.10 A LOOKBACK CALL OPTION UNDER STOCHASTIC VOLATILITY WITH DELTA, GAMMA AND VEGA CONTROL VARIATES

In this example we price a European fixed strike lookback call option. This option pays *the difference, if positive, between the maximum of a set of observations (fixings)* of the asset price S_{t_i} at dates t_i; $i = 1, \ldots, N$ and the strike price. Thus the pay-off at the maturity date is

$$\max(0, \max(S_{t_i}; i = 1, \ldots, N) - K) \tag{4.50}$$

We will also assume that the asset price and the variance of the asset price returns $V = \sigma^2$ are governed by the following stochastic differential equations:

$$dS = rS\,dt + \sigma S\,dz_1 \tag{4.51}$$

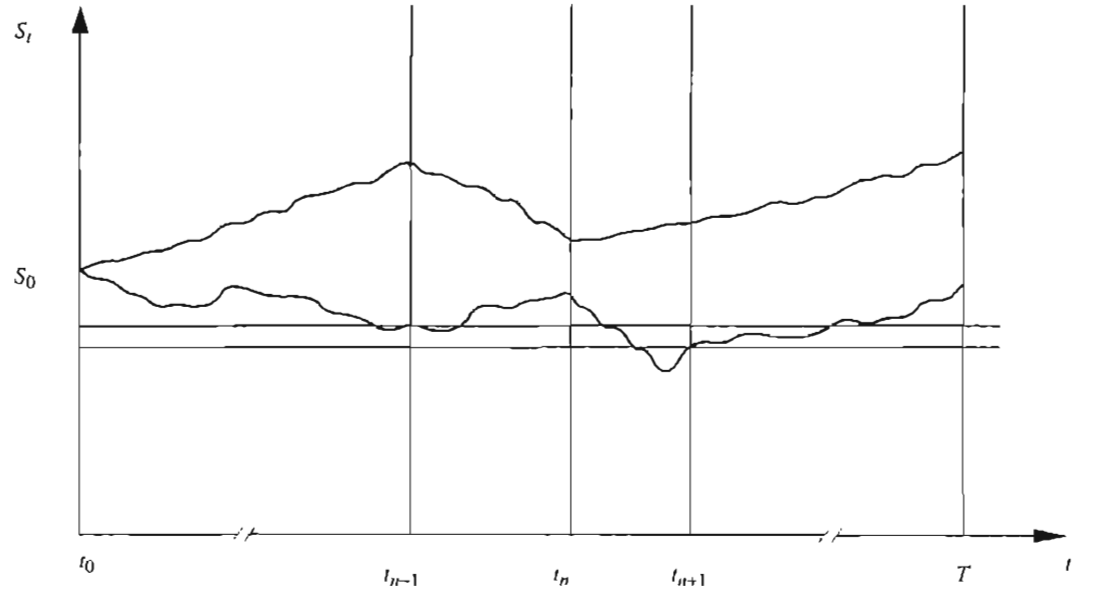
$$dV = \alpha(\overline{V} - V)\,dt + \xi\sqrt{V}\,dz_2 \tag{4.52}$$

and that the Wiener processes dz_1 and dz_2 are uncorrelated, but this is easily generalised as we saw in section 4.7. Figure 4.26 illustrates two typical asset price paths and the fixing dates.

There is no analytical solution for the price of European fixed strike lookback call option with discrete fixings and stochastic volatility. However, there is a simple analytical formula for the price of a continuous fixing fixed strike lookback call with constant volatility:[9]

$$\begin{aligned} C_{\text{FIXED\_STRIKE\_LOOKBACK\_CALL}} = {} & G + Se^{-\delta T}N(x + \sigma\sqrt{T}) - Ke^{-rT}N(x) \\ & - \frac{S}{B}\left(e^{-rT}\left(\frac{E}{S}\right)^{B} N\left(x + (1-B)\sigma\sqrt{T}\right)\right. \\ & \left. - e^{-\delta T}N\left(x + \sigma\sqrt{T}\right)\right) \end{aligned} \tag{4.53}$$

FIGURE 4.26 Typical Asset Price Paths and Fixing for a Lookback Option



where

$$\left.\begin{array}{c} E = K,\ G = 0 \\ E = M,\ G = e^{-rT}\ (\mathrm{M} - \mathrm{K}) \end{array}\right\} \left\{\begin{array}{l} K \geq M \\ K < M \end{array}\right. , \quad B = \frac{2(r-\delta)}{\sigma^2} , \quad x = \frac{\ln\left(\frac{S}{E}\right) + \left((r-\delta) - \frac{1}{2}\sigma^2\right) T}{\sigma\sqrt{T}}$$

and M is the current known maximum. We can therefore use the continuously fixed floating strike lookback call option formula to compute *delta*, *gamma* and *vega* hedge control variates. Rather than differentiate equation (4.52) with respect to the asset price twice and volatility once which would lead to extremely complex expressions it is more efficient to use finite difference approximations to the partial differentials for *gamma* and *vega*.

Figure 4.27 shows a pseudo-code implementation of the Monte Carlo valuation of a European fixed strike lookback call option with continuously fixed lookback call option *delta, gamma* and *vega* hedge control variates.[10]

As discussed in section 4.4, the values of β_1, β_2 and β_3 were obtained by linear regression. Table 4.3 gives typical standard errors and relative computation times for the example of the application of the Monte Carlo valuation of a European fixed strike lookback call option in a Black–Scholes world with stochastic volatility using antithetic and *delta*-, *gamma*- and *vega*-based control variates.

With the combined antithetic, *delta, gamma* and *vega* control variates the standard error is reduced by a factor of 12. To achieve this reduction with the simple Monte Carlo method would require 144 000 simulations with an execution time of roughly 3.5 hours. With careful choice of the most efficient control variates and optimisation of the code the execution time can typically be reduced by a factor of between two and five.

FIGURE 4.27 Pseudo-code for Monte Carlo Valuation of a European Fixed Strike Lookback Call Option with Stochastic Volatility and Continuously Fixed Lookback Call Option *Delta*, *Gamma* and *Vega* Hedge Control Variates

```
initialise_parameters
  { K, t[], S, sig, r, div, alpha, Vbar, xi, N, M }

{ precompute constants }

sig2 = sig^2
alphadt = alpha*dt
xisdt = xi*sqrt(*dt)
erddt = exp((r-div)*dt)
egam1 = exp(2*(r-div)*dt)
egam2 = -2*erddt+1
eveg1 = exp(-alpha*dt)
eveg2 = Vbar - Vbar*eveg1

sum_CT = 0
sum_CT2 = 0

beta1 = -0.88
beta2 = -0.42
beta3 = -0.0003

for j = 1 to M do { for each simulation }

 St1 = S ; St2 = S ; Vt = sig2
 maxSt1 = St ; maxSt2 = St
 cv1 = 0 ; cv2 = 0 ; cv3 = 0

 for i = 1 to N do { for each time step }

   { compute hedge sensitivities }
   t = (i-1)*dt
   delta1 = lookback_delta(St1,t,K,T,Vt,r,div,maxSt1)
   delta2 = lookback_delta(St2,t,K,T,Vt,r,div,maxSt2)
   gamma1 = lookback_gamma(St1,t,K,T,Vt,r,div,maxSt1)
   gamma2 = lookback_gamma(St2,t,K,T,Vt,r,div,maxSt2)
   vega1 = lookback_vegaV(St1,t,K,T,Vt,r,div,maxSt1)
   vega2 = lookback_vegaV(St2,t,K,T,Vt,r,div,maxSt2)

    { evolve variance }
    ε = standard_normal_sample
    Vtn = Vt + alphadt*(Vbar-Vt) + xisdt*sqrt(Vt)*ε

    { evolve asset price }
    ε = standard_normal_sample
    Stn1 = St1*exp( (r-div-0.5*Vt)*dt + sqrt(Vt)*sdt*(ε) )
    Stn2 = St2*exp( (r-div-0.5*Vt)*dt + sqrt(Vt)*sdt*(-ε) )

    { accumulate control variates }
    cv1 = cv1 + delta1*(Stn1-St1*erddt) +
                delta2*(Stn2-St2*erddt)
```

(continues)

FIGURE 4.27 *(continued)*

```
    cv2 = cv2 +
      gamma1*((Stn1-St1)^2-St1^2*(egam1*exp(Vt*dt) + egam2)) +
      gamma2*((Stn2-St2)^2-St2^2*(egam1*exp(Vt*dt)+egam2))
    cv3 = cv3 + vega1*((Vtn-Vt)-(Vt*eveg1+eveg2-Vt)) +
                vega2*((Vtn-Vt)-(Vt*eveg1+eveg2-Vt))

    Vt = Vtn
    St1 = Stn1
    St2 = Stn2

    if ( St1 > maxSt1 ) maxSt1 = St1
    if ( St2 > maxSt2 ) maxSt2 = St2

  next i

  CT = 0.5*( max( 0 , maxSt1 - K ) + max( 0 , maxSt2 - K ) +
             beta1*cv1 + beta2*cv2 + beta3*cv3 )
  sum_CT = sum_CT + CT
  sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( ( sum_CT2 - sum_CT*sum_CT/M )*exp(-2*r*T)/(M-1) )
SE = SD/sqrt(M)
```

TABLE 4.3 Typical Standard Errors and Computation Times for the Monte Carlo Valuation of a European Fixed Strike Lookback Call Option in a Black–Scholes World with Stochastic Volatility Using Antithetic and Delta-, Gamma- and Vega-based Control Variates

Strike price		100
Time to maturity		1 year
Initial asset price		100
Volatility		20%
Riskless interest rate		6%
Continuous dividend yield		3%
Mean reversion rate (α)		5.0
Volatility of volatility (ξ)		0.02
Number of time steps		52
Number of simulations		1000
Continuous fixing fixed strike lookback call value		17.729
	Standard error	Relative computation time
Simple estimate	0.4803	1.00
With antithetic variate	0.2030	1.17
With control variates	0.0485	17.77
Combined variates	0.0378	29.23

4.11 GENERATING STANDARD NORMAL RANDOM NUMBERS

A critical part of Monte Carlo simulation is the simulation of the Brownian motions or the generation of the standard normal random variables. Most programming languages and spreadsheets provide a uniform pseudo-random number generator. This will usually generate a random integer between zero and a specified upper value, each integer in the range being generated with equal probability. Sometimes the generator produces the standard mathematical definition of a uniform random number, that is, a real number between zero and one. The integer version can be converted to the standard by specifying a large upper value and then dividing the generated random numbers by this value. The numbers generated by these routines are designed to appear random when subject to standard statistical tests for randomness. However, some routines are better than others and so the pseudo-random number generator you intend to use should always be subjected to the standard statistical tests (see Press *et al.*, 1992 and Ripley 1987, for excellent discussions of pseudo-random number generation and testing).

Armed with a standard uniform random number generator it is straightforward to convert these to standard normal random numbers. A common, but approximate, way to do this is to generate 12 standard uniform random numbers, add them together and subtract six from the total. The distribution of this combination has the correct zero mean and variance of one, and is a good approximation to a normal distributed random variable. However, if we consider the minimum and maximum possible values of z which are -6 and 6, this is a range of -6 to 6 standard deviations of true standard normal random variable. Thus a true standard normal random variable has a probability of approximately 0.000 000 001 of exceeding these minimum and maximum values. More importantly the kurtosis of z is only 0.15, which means that too many values close to the mean will be generated.

A simple alternative to this method is the Box–Muller transformation. This is an exact transformation of pairs of standard uniform random numbers to pairs of standard normal random variables. Let x_1 and x_2 be standard uniformly distributed random variables, then a pair of standard normally distributed variables z_1 and z_2 can be obtained via

$$z_1 = \sqrt{-2\ln(x_1)}\cos(2\pi x_2)$$

$$z_2 = \sqrt{-2\ln(x_1)}\sin(2\pi x_2)$$

A more efficient implementation is the polar rejection method which can be stated algorithmically as follows:

$$\begin{aligned}
&\text{repeat}\\
&\quad x_1 = \text{standard_uniform_random_number}\\
&\quad x_2 = \text{standard_uniform_random_number}\\
&\quad w = x_1^2 + x_2^2 \qquad (4.54)\\
&\text{until } w < 1\\
&\quad c = \sqrt{-2^* \frac{\ln(w)}{w}}\\
&\quad z_1 = cx_1\\
&\quad z_2 = cx_2
\end{aligned}$$

TABLE 4.4 Relative Execution Speeds of Standard Normal Random Number Generator Methods

Method	Relative execution time
Adding twelve uniform random numbers	1.000
Box–Muller	0.500
Polar rejection	0.375

The only complication with the Box–Muller and polar rejection methods is that they generate two standard normal random variables at a time. Therefore, in order to obtain maximum efficiency the second random number must be saved for use the next time a standard normal random number is required, rather than generating two random numbers every time and throwing one away. If this is done the relative execution times of the 12 uniforms, Box–Muller and polar rejection methods are given in Table 4.4.

Since the polar rejection method is both faster and more accurate it is our preferred method of generating standard normal random numbers. The generation of the random numbers is generally about 30 per cent of the total execution time of a Monte Carlo simulation.

Another problem we encountered earlier is how to combine independent standard normal random variables to obtain correlated variables. The solution is the eigensystem representation of the covariance matrix of the correlated variables. The eigenvectors are the linear combinations of the correlated variables which give independent variables. Therefore, all we have to do is invert this relationship to obtain the linear combinations of independent variables which reproduce the required covariance matrix. In the case of a covariance matrix, which is real and symmetric the matrix of eigenvectors is orthogonal, i.e. the inverse of the matrix is equal to its transpose.

Consider n variables $z_i, i = 1, \ldots, n$ which are jointly normally distributed with mean zero, variance one and correlation/covariance matrix Σ so that they can be considered correlated Brownian motions with infinitesimal increments dz_i.

Principal components analysis (PCA) will give n eigenvectors $\underline{v}_i$ and n associated eigenvalues λ_i such that

$$\Sigma = \Gamma \Lambda \Gamma' \tag{4.55}$$

where

$$\Gamma = \begin{vmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ v_{n1} & v_{n2} & \cdots & v_{nn} \end{vmatrix} \quad \text{and} \quad \Lambda = \begin{vmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \lambda_n \end{vmatrix}$$

the columns of Γ are the eigenvectors.

Since the transpose of Γ is equal to its inverse, the rows of Γ represent the proportions of a set of n independent Brownian motions $dw_i; i = 1, \cdots, n$ which when linearly combined reproduce the original correlated Brownian motions. The eigenvalues represent the variances of the independent Brownian motions. Therefore, the correlated Brownian motions dz_i can be reproduced from linear combinations of the independent Brownian motions dw_i as follows:

$$dz_1 = v_{11}\sqrt{\lambda_1}\, dw_1 + v_{12}\sqrt{\lambda_2}\, dw_2 + \cdots + v_{1n}\sqrt{\lambda_n}\, dw_n$$

$$
\begin{aligned}
dz_2 &= v_{21}\sqrt{\lambda_1}\,dw_1 + v_{21}\sqrt{\lambda_2}\,dw_2 + \cdots + v_{2n}\sqrt{\lambda_n}\,dw_n \\
&\cdots \\
dz_n &= v_{n1}\sqrt{\lambda_1}\,dw_1 + v_{n2}\sqrt{\lambda_2}\,dw_2 + \cdots + v_{nn}\sqrt{\lambda_n}\,dw_n
\end{aligned}
\tag{4.56}
$$

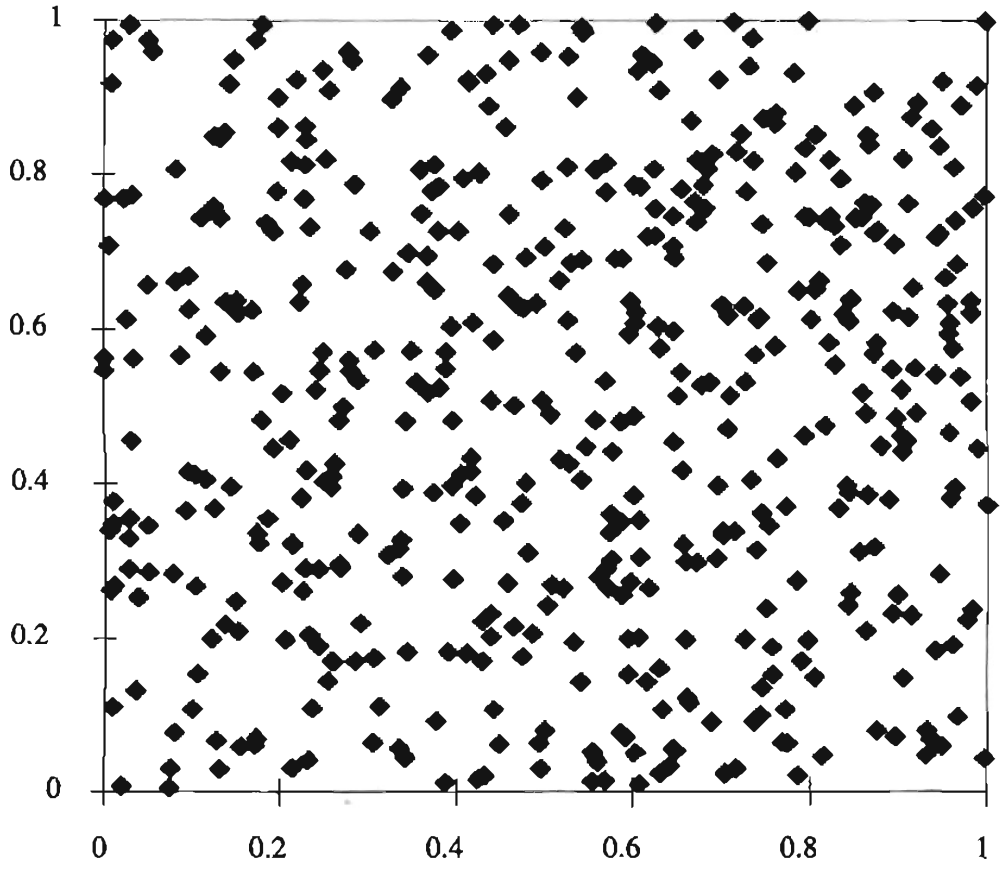
where the dw_i are independent Brownian motions.[11] To use this in a Monte Carlo simulation we simply discretise equation (4.56) in the usual way.

4.12 QUASI-RANDOM NUMBERS

It turns out that pseudo random numbers, which are designed to be as random as possible, are poor choices for use in Monte Carlo simulation. To illustrate the reason for this consider two uniform independent random numbers between zero interpreted as the horizontal and vertical position on a graph. Figure 4.28 illustrates a graph with 500 points obtained in this way.

Since the numbers are uniform and independent then every point on the graph is equally likely to appear. However, every time we sample a point, all points are still equally likely and so we get "clumps" of points close together and empty spaces (see Figure 4.28). Eventually, as we sample more and more points we will obtain a distribution of points which appears smooth, the initial clumpiness will have been swamped by the large number

FIGURE 4.28 Plot of 500 Pairs of Uniform Random Numbers

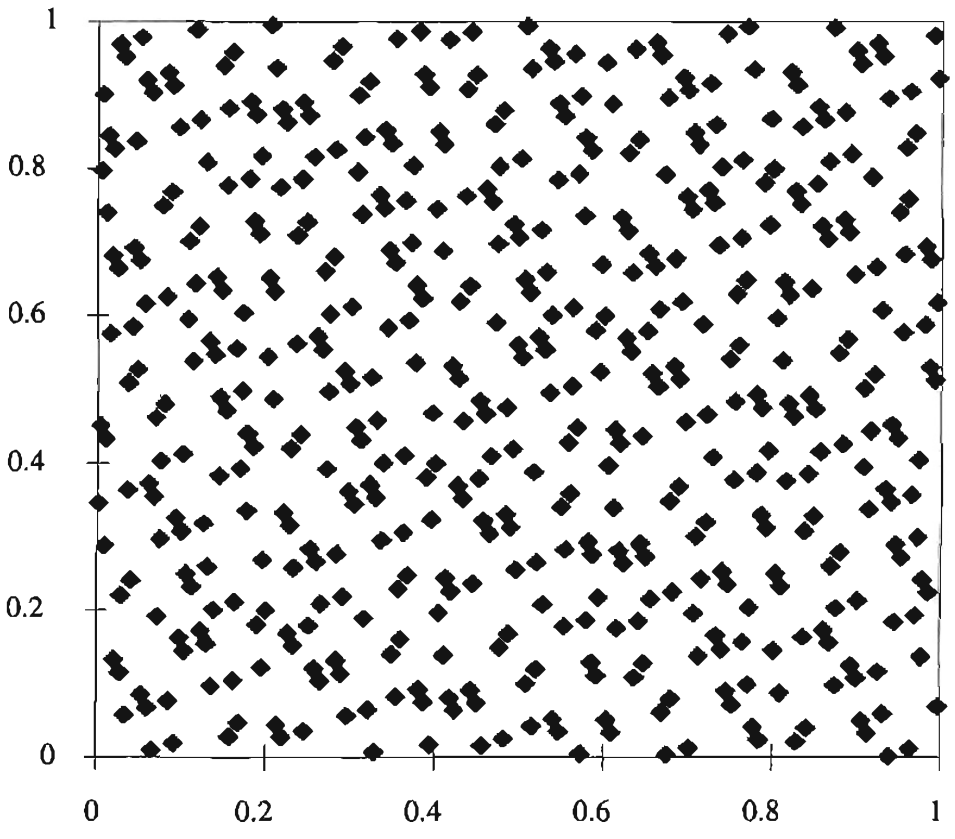


of points spread evenly across the graph. The problem with pseudo-random numbers for Monte Carlo simulation, is that for small samples, which has been the aim of the control variates, the clumpiness biases the results. A very large number of samples are needed to make the bias negligible.

Quasi-random numbers or low-discrepancy sequences are designed to appear random but not clumpy. In other words a quasi-random sample is not independent from the previous one, it "remembers" the previous samples and tries to position itself away from all the previous samples. There are many ways of producing quasi-random numbers (see Niederreiter, 1992, Niederreiter and Shiue, 1995), but we will just describe one, the Faure method, which we have found to work well for option valuation problems and which is reasonably efficient to compute. Figure 4.29 plots 500 pairs of Faure numbers, compare this with Figure 4.28 and notice how the Faure points are much more evenly distributed but still appear somewhat random. This behaviour is ideal for obtaining fast convergence in a Monte Carlo simulation and we illustrate this with an example shortly.

Consider generating the quasi-random numbers for the Monte Carlo simulation of the path of an asset price with N steps as usual. In the context of quasi-random numbers N is the dimension of the sequence, that is, the number of independent quasi-random numbers to be generated simultaneously. We must generate all the N random numbers simultaneously so that they are independent and the increments along the paths have the correct statistical properties. Let the N quasi-random numbers be $x_k; k = 1, \ldots, N$ then

FIGURE 4.29 Plot of 500 Pairs of Faure Quasi-random Numbers



a Faure sequence of length M is defined by

$$x_k = \sum_{l=0}^{m} \frac{a_{k,l}}{p^{l+1}} \tag{4.57}$$

where m is the number of digits in the base p representation of M (m = integer part of $\ln(M)/\ln(p)$), $a_{0,l}$; $j = 0, \ldots, m(a_{0,l}$ = integer part of $(M\%p^{l+1}/p^l))$,[12]

$$a_{k,l} = \sum_{q=l}^{m} \frac{q!}{l!(q-l)!} a_{k-1,q}\%p \tag{4.58}$$

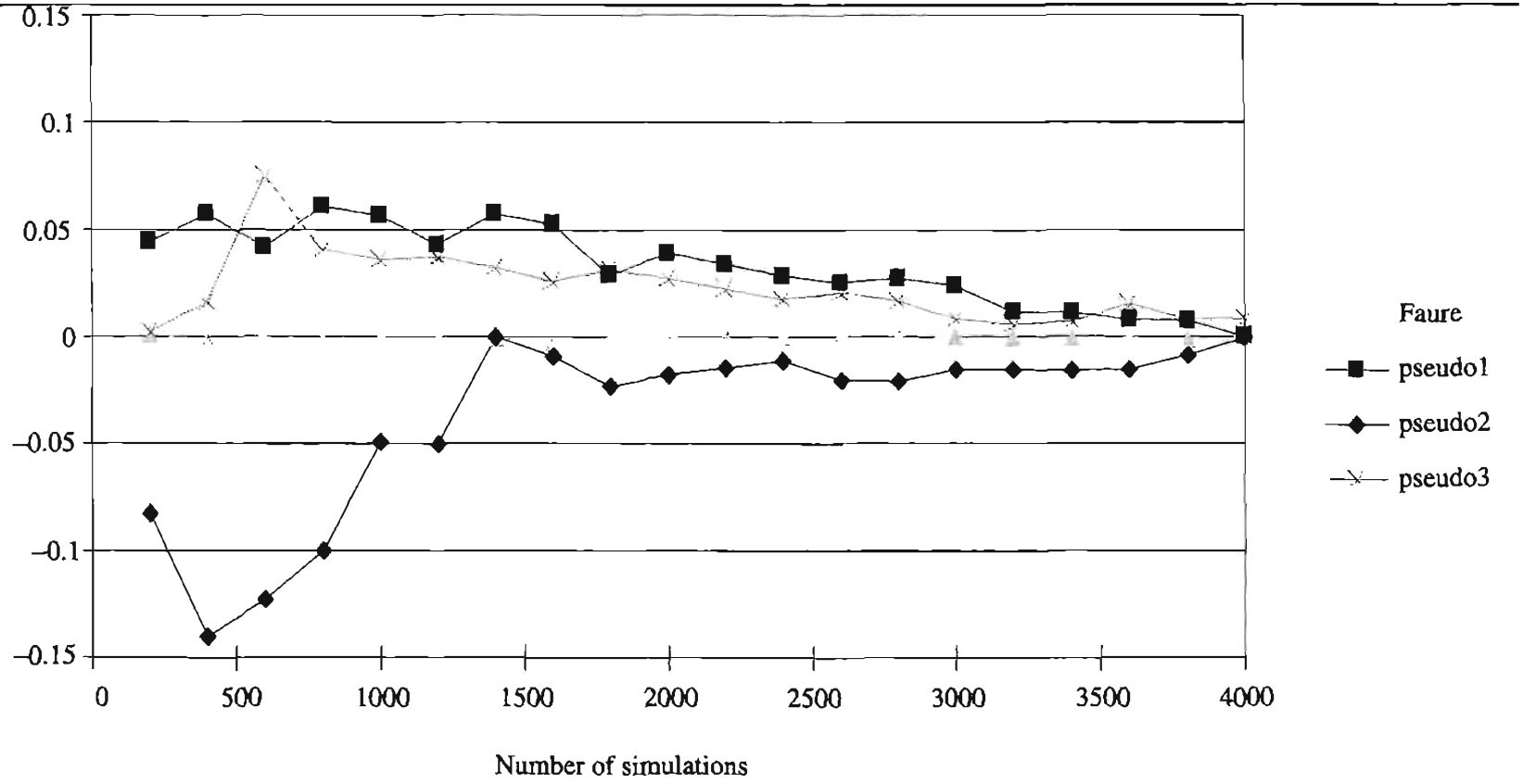
and p is the smallest prime number greater than or equal to N. Figure 4.30 gives the pseudo-code algorithm for the Monte Carlo valuation of a European call option using a Faure sequence.

The only difference between Figure 4.30 and Figure 4.2 are the two lines highlighted in bold where the Faure numbers for the current path are generated and standard normal random number is generated by Box–Muller from the appropriate Faure numbers (see section 4.11). Figure 4.31 illustrates the prices obtained from pricing a European call

FIGURE 4.30 Pseudo-code for Monte Carlo Valuation of a European Call Option in a Black–Scholes World using a Faure Sequence

```
initialise_parameters { K, T, S, sig, r, div, N, M }
{ precompute constants }

dt = T/N
nudt = (r-div-0.5*sig*sig)*dt
sigsdt = sig*sqrt(dt)
sum_CT = 0
sum_CT2 = 0
lnS = ln(S)

for j = 1 to M do { for each simulation }
 generate_Faure_sequence( x[], N, M )

 lnSt = lnS
 for i = 1 to N do { for each time step }
  ε = standard_normal_by_Box_Muller( x[i] )
  lnSt = lnSt + nudt + sigsdt*ε
 next i

 St = exp(lnSt)
 CT = max( 0 , St - K )
 sum_CT = sum_CT + CT
 sum_CT2 = sum_CT2 + CT*CT

next j

call_value = sum_CT/M*exp(-r*T)
SD = sqrt( sum_CT2/M*exp(-2*r*T) - call_value*call_value )
SE = SD/sqrt(M-1)
```

FIGURE 4.31 Relative Pricing Error for a European Call Option Using Pseudo-random Numbers and Faure Quasi-random Numbers



option using three different sets of pseudo-random numbers and a set of Faure quasi-random numbers.

It is clear that the prices using the Faure quasi-random numbers converge much faster as a function of the number of simulations than the prices using pseudo-random numbers. The Faure quasi-random numbers take approximately the same computation time to generate as pseudo random numbers.

4.13 SUMMARY

In this chapter we have described the use of Monte Carlo simulation for pricing and hedging derivatives. We discussed efficient implementation techniques using antithetic and control variates based on the concept of hedging and methods for computing hedge sensitivities. It was shown how the Monte Carlo method could be used to handle multiple stochastic variables and path-dependent options with detailed examples using Asian and lookback options. Finally, we discussed important aspects involved in the generation of the independent standard normal samples via pseudo-random and quasi-random methods.

ENDNOTES

1. In this section we explicitly show the time dependence of variables with a subscirpt because we will often need to refer to the time the variable is observed.
2. See Kloeden and Platen (1992,1994) for advanced methods of discretising stochastic differential equations.
3. We discuss the generation of standard normal random variates in section 4.11.
4. See also Hull and White (1988) on the use of control variates for tree or lattice methods.
5. It is more efficient to value this option by Gaussian quadrature (see Ravindran, 1993), but when we add path-dependent features or use more general stochastic processes, applying Gaussian quadrature becomes very difficult if not impossible.
6. Here we have applied a simple (Euler) discretisation to the SDE for the variance for clarity, see Kloeden and Platen (1992, 1994) for more advanced methods.
7. See Nelken (1996) and Clewlow and Strickland (1997) for more details on the specification of exotic options contracts, analytical pricing formulae and hedging techniques.
8. See Levy (1997) for a survey of Asian options and a review of published work on them.
9. see Heynen and Kat (1997b) for a survey of lookback options contracts, analytical results and a comparison of numerical methods.
10. Here we have applied a simple (Euler) discretisation to the SDE for the variance for clarity (see Kloeden and Platen, 1992, 1994 for more advanced methods). See Hull and White (1988) for details of the stochastic volatility process (in particular the expression for the expectation of the variance used in the vega-based control variate).
11. Standard libraries are available for performing PCA analysis, for example "Numerical recipes in C", Press *et al.* (1992).
12. % indicates the remainder after integer division,