Pricing American Options Using Monte Carlo Simulation

Hammond Mason*

ABSTRACT

Monte Carlo simulation is a numerical method used to value options that have no closed-form, analytical solution. The method was first applied to value options in 1977 and has since been used to value increasingly complex option structures. Despite its successful adoption by the financial community, monte carlo simulation remains a computationally intensive method and the growth in increasingly complex financial products has resulted in the need for innovative techniques to reduce this computational burden. This dissertation analyses two of the current techniques as they apply to valuing American options. One of these techniques proves to be complex and time consuming and suffers from a restriction on its practical application. The other displays simplicity, speed and flexibility.

1 Introduction

As financial markets evolve, they continue to produce complex financial products for which mathematics cannot yet provide analytical valuation solutions. Despite the lack of closed-form pricing formulae, the continued demand for new product from financial market participants has led both buyers and sellers to arrive at agreed price valuations using what are termed 'numerical methods'. Monte Carlo simulation is one such numerical method.

Since Boyle [1977] first applied the method to option valuation, research into Monte Carlo simulation techniques has expanded. Monte Carlo simulation of American options has been one such area of research. This dissertation reviews the results of this research with an emphasis on the comparative performance of two of the most recently published algorithms. The dissertation is organised as follows. First, Section 2 discusses a few basics of Monte Carlo simulation and how it can be used to approximate the price of a *European* option. The discussion is then extended to the key feature of *American* options – the free boundary problem – and how this complicates the Monte Carlo method. In Section 3, a brief review of the research literature is made. This continues with Section 4 providing greater discussion of two recent algorithms. Section 5 contains some numerical results and Section 6 the concluding remarks.

2 Background

Monte Carlo Simulation

Pricing options using Monte Carlo simulation relies on two key mathematical concepts: the Law of Large Numbers and the Martingale Approach to asset pricing.

If $X_1, X_2, X_3, \ldots, X_M$ is a sequence of independent and identically distributed random variables having mean μ, then the Weak Law of Large Numbers states:

$$P\left(\left|\frac{X_1+X_2+\ldots+X_M}{M}\right|>\varepsilon\right)\rightarrow 0 \quad as\ M\rightarrow\infty \tag{1}$$

which can be generalised to the Strong Law of Large Numbers:

$$\lim_{M\rightarrow\infty}\frac{X_1+X_2+\ldots+X_M}{M}=\mu \tag{2}$$

ie.:

$$\frac{1}{M}.\sum_{i=1}^{M}X_i\rightarrow\mu \quad as \quad M\rightarrow\infty \tag{3}$$

* MSc Financial Engineering student, Birkbeck College, University of London.

For example, if $X_i = PV\left\{max\left(0, K-S_i^T\right)\right\}$ (the present value of the payoff from a European put option expiring at time T and having a strike price of K) is distributed with mean, P, then:

$$\frac{1}{M}.\sum_{i=1}^{M} PV\left\{max\left(0, K-S_i^T\right)\right\} \rightarrow P \quad as \quad M\rightarrow\infty \qquad (4)$$

and since the mean is its expected value, ie:

$$P = E\left[PV\left\{max\left(0, K-S_i^T\right)\right\}\right] \qquad (5)$$

we have the result:

$$\lim_{M\rightarrow\infty} \frac{1}{M}.\sum_{i=1}^{M} PV\left\{max\left(0, K-S_i^T\right)\right\} \rightarrow E\left[PV\left\{max\left(0, K-S_i^T\right)\right\}\right] \qquad (6)$$

which can be simulated under the Monte Carlo method using an algorithm such as follows:

1. Divide the chosen time horizon (T-t) into a discrete set of N individual time periods ('time steps'), each of Δt in duration[1].
2. Working forward in time, for each random variable (eg. stock price) at each Δt, randomly sample its value from the population of possible values, thereby constructing a 'price path' for the asset over the N number of Δt time steps.
3. Determine the set of discounted (ie. present-valued) cash flows from the option according to the option's payoff function (eg. European call, American put, Asian strike call, etc.).
4. Repeat the above process for a desired number, M, of times.
5. Take the average of step 4 to arrive at an estimate of the option's price.
6. Repeat at different levels of M in order to achieve the desired level of precision (ε in equation (1) above).

Having performed the simulation and obtained an estimate, $\hat{P}$, for the value P, the obvious question is "what does this estimate represent?"

The Martingale Approach to asset pricing states that under the assumptions of (i) no arbitrage opportunities, and (ii) market completeness, the price of an option is the risk-neutral expectation[2] of the present value of its possible future cash flows (ie. payoffs):

$$P = E\left[PV\left\{max\left(0, K-S_i^T\right)\right\}\right] \qquad (7)$$

Therefore, Monte Carlo simulation provides us with an estimate of an option's price.

$$\hat{P} = \frac{1}{M}.\sum_{i=1}^{M} PV\left\{max\left(0, K-S_i^T\right)\right\} \qquad (8)$$

which, being an estimate is subject to some error, ε:

$$\hat{P} = P+\varepsilon \ , \ |\varepsilon| > 0 \qquad (9)$$

Having obtained an estimate we can invoke the Central Limit Theorem to allows us to set confidence intervals for P.

Under the Central Limit Theorem, if X_1, X_2, X_3, ... is a sequence of independent and identically distributed random variables having finite mean, μ, and finite variance, σ^2, then:

$$\lim_{M\rightarrow\infty} P\left(\left|\frac{X_1+X_2+...+X_{Mn}-M.\mu}{\sigma.\sqrt{M}}\right| \leqslant x\right) = \Phi(x) \qquad (10)$$

where $\Phi(x)$ denotes the cumulative distribution function of the standard normal random variable. Put another way, this theorem states that for large M:

$$\frac{1}{M}.\sum_{i=1}^{M} X_i \Rightarrow N\left(\mu, \frac{\sigma^2}{M}\right) \qquad (11)$$

Hence, for the European put option example, above:

$$\hat{P} \Rightarrow N\left(P, \frac{\sigma^2}{M}\right) \quad for\ large\ M \qquad (12)$$

ie. when M is large:

$$\frac{\hat{P}-P}{\frac{\sigma}{\sqrt{M}}} \sim N(0,1) \qquad (13)$$

and the (1-α)% confidence interval for P is:

$$\left[\hat{P}-\frac{\sigma}{\sqrt{M}}.z_{(1-\alpha)}\ ,\ \hat{P}+\frac{\sigma}{\sqrt{M}}.z_{(1-\alpha)}\right] \qquad (14)$$

1 For the case of a vanilla European option N=1 as we are only interested in the exercise at maturity. However, exotic options (including American options) require N > 1.

2 This is alternatively expressed as taking the expectation with respect to the Q-measure.

As σ^2 is unknown, it is common to us its unbiased estimate, s^2, instead:

$$s^2 = \frac{1}{M-1} . \sum_{i=1}^{M} \left(X_i - \hat{P} \right)^2 \tag{15}$$

so that the confidence interval becomes:

$$\left[\hat{P} - \frac{s}{\sqrt{M}} . z_{(1-\alpha)} \, , \; \hat{P} + \frac{s}{\sqrt{M}} . z_{(1-\alpha)} \right] \tag{16}$$

So far, nothing has been said of the present value operator, *PV{...}*. Throughout this dissertation a constant interest rate environment will be assumed, unless stated otherwise. This allows all cash flows to be present valued at a single, riskless rate of interest, r, using the function:

$$PV_t \left\{ X_T \right\} = e^{r.(T-t)} . X_T \tag{17}$$

As r and T are both known at time t, this allows all present value operations to be performed outside of the expectation operator, ie:

$$P_t = e^{r.(T-t)} . E\left[max\left(0, K - S_i^T\right)\right] \tag{18}$$

This greatly simplifies the mathematics, preventing unnecessary complication from clouding the concepts discussed throughout this dissertation.

The Free Boundary Problem in American Options

While a European option allows the option holder to exercise its rights under the option at the option's maturity date, an American option allows the holder to exercise at any time[3]. It is this 'early-exercise' feature that poses problems for an analytical solution to an American option.

While Black and Scholes [1973] solved the problem of the European option, there is no general solution to their partial differential equation (PDE) where the exercise boundary cannot be explicitly determined. To illustrate this, consider the price of an American put option on a non-dividend paying stock[4].

The value of the put option, P, at time t, tends to the option's exercise price, K as the stock price, S, tends to zero – because the option holder is unlikely to sell in the market and increasingly likely to exercise its put, in order to maximise their gain. In other words:

$$P \rightarrow K \quad as \quad S \rightarrow 0 \tag{19}$$

Similarly, the value of the put option tends to zero as the stock price tends to infinity – the option holder being more inclined to sell and receive *S* than exercise and receive *K*, ie:

$$P \rightarrow 0 \quad as \quad S \rightarrow \infty \tag{20}$$

Interpolating between these two extremes it can be seen that, at any point in time, there will exit a particular stock price, *S**, such that if *S* is below *S**, it is optimal to exercise. Additionally, if *S* is above *S**, it is optimal to continue holding the option. As *t* changes so does *S** such that it forms what is known as 'the optimal exercise boundary' as demonstrated in Figure 1:

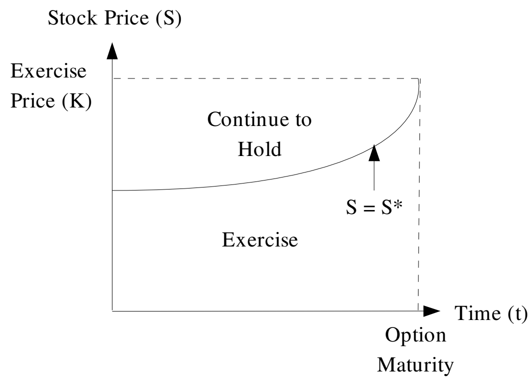

Figure 1: Optimal Exercise Boundary, S = S, for an American put option.*

The problem in formulating a closed form solution to the Black-Scholes PDE in this case, is that the free boundary cannot be defined ex-ante, thereby frustrating any attempt to solve the PDE.

The free boundary problem is also referred to as the "optimal stopping time" problem. Taking equation (18) above and expressing it for the American case by using τ to represent the time at which exercise occurs (the optimal stopping time) we have:

$$P_t = e^{r.(\tau-t)} . E\left[max\left(0, K - S_i^{\tau}\right)\right] \tag{21}$$

3 Monte Carlo simulation works with discrete time steps, Δt. Hence, it can only approximate the value of an American option – when Δt is small. Hence, an American option is approximated by its Bermudan counterpart. Despite this technicality, I will continue to use the term 'American options' when discussing Monte Carlo techniques.

4 An American *call* option on a non-dividend paying stock has the same value as the equivalent European call option. This is because it is never optimal to exercise such an American call option prior to its maturity date. The reader is referred to Higham [2005] p.174 for an explanation.

which, we can see, requires knowledge of exactly when τ occurs[5]. In theory, the holder of an American option will be constantly comparing the value of his option if exercised now ("Intrinsic Value") versus its value if held for a fraction of a second more ("Continuation Value"). In the case of an American put option, this decision rule can be expressed mathematically as:

$$P_t = max\left(K - S_t,\ e^{r.dt}.P_{t+dt}\right) \quad (22)$$

where P_t is the option's value at time t and dt is the infinitesimally small increment in time. While K-S_t is readily estimated using Monte Carlo simulation, $e^{r.dt}.P_{t+dt}$ (the Continuation Value) is not so easily obtained. Since the Continuation Value is essentially the option's value as determined at t plus a fraction of a second more, obtaining this value would require knowing the Continuation Value at a fraction of a second after the first fraction of a second (ie 2 x dt) and so on, recursively, until the maturity date. Such an algorithm would appear to require a whole series of Monte Carlo simulations to be performed – one for each Continuation Value – thus giving rise to an infeasibly large computational task. Indeed, this was the thinking up until the early 1990s and pricing American options was confined to other numerical methods such as the binomial lattice method of Cox, Ross and Rubenstein [1979] and the method of finite differences first applied to option pricing by Brennan and Schwartz [1978].

The problem with these methods, however, is that they do not scale well with options involving high dimensions (ie. many state variables rather than a single underlying asset). Boyle, Evnine and Gibbs [1989] did extend the binomial lattice approach to American options with two state variables and to *European* options involving three assets. However, it is acknowledged by many that although it is possible to take this further to valuing multivariate *American* options, Barraquand and Martineau [1995] point out that such approaches become impractical for valuing higher order American options (ie. >3 state variable), as the amount of computational effort expands exponentially with the number of dimensions. A similar problem also occurs with the finite difference approach to option valuation.

Consequently, Monte Carlo simulation remained (and still remains) an attractive area of research for pricing high dimension options. Additionally, the attraction of Monte Carlo simulation as a numerical method also lies in the generality of the types of assets it can handle and the ease with which it copes with complex payoff functions such as those exhibiting path dependencies.

5 τ is actually the set of all stopping times that occur when S=S* so, technically, equation (21) should bring the present value operator back inside the expectation operator. However, equation (21) has been present as above as it is believed it makes the concept easier to understand for the reader.

3 Monte Carlo Methods For American Options

Boyle [1977] was the first to apply Monte Carlo simulation to pricing options. While the analytical solution to a *European* call option on a non-dividend paying stock had been solved some four years earlier by Black and Scholes [1973], Boyle focussed his efforts on the price of a European call option on a *dividend paying* stock. Early attempts to use Monte Carlo simulation to price *American* options had been limited to estimating the explicit functional form of the optimal exercise boundary. For example, Omberg [1987] uses constrained optimisation of exponential functions to specify this boundary. Indeed, as stated previously, it was common knowledge at the time that Monte Carlo simulation could not be used to value American options[6].

It was some 16 years after Boyle's paper that Tilley [1993] proposed the first algorithm for pricing American options using Monte Carlo simulation. Tilley's 'bundling' algorithm was a regression method using crude kernel smoothing techniques. Carriere [1996] Subsequently proved this algorithm gave rise to biased estimates and, in doing so, offered an alternative method employing a sequential regression algorithm that resulted in unbiased estimates.

Barraquand and Martineau [1995] proposed a stratification method for pricing high-dimensional American options by partitioning the payoff space instead of the state space (Tilley's 'bundling'). However, this too encountered subsequent problems when Boyle, Broadie and Glasserman [1997] showed that the method did not necessarily converge to the correct value and often lead to a significant underestimate of an option's value.

Broadie and Glasserman [1997] argued there could, in fact, be no general method for producing an unbiased simulation estimator of American option values[7]. Instead of focussing on a point estimate, their 'Random Tree' method produced two asymptotically unbiased estimates: one biased high, the other biased low. By taking the upper confidence limit from the 'high' estimator and the lower confidence limit from the 'low' estimator and combining them, they produce a conservative confidence interval for the true option price. Their numerical results are encouraging, exhibiting very tight bands for this interval. However, their Random Tree approach is not without its problems. While it lends itself well to options with many state variables, it suffered from the same "curse of dimensionality" found in many tree-based algorithms (including lattice methods). Considerable computational effort is required when the

6 See, for example, footnote 8 of Hull and White [1988]: "... the [Binomial lattice and Monte Carlo simulation] approaches are not direct substitutes for each other. Monte Carlo simulation can be used only for European options, whereas [Binomial lattice] approaches can be used for European or American options."

7 See Broadie and Glasserman [1997] at p. 1326.

number of exercise opportunities is large[8], rendering it unsuitable for American option valuation. The authors acknowledge this and suggest using extrapolation procedures when implementing the method in practice. However, reliance on extrapolation techniques is likely to be a less than satisfactory solution for many a financial risk manager. Consequently, Broadie and Glasserman [2004][9] sought to improve on this with their "Stochastic Mesh" method, discussed in Section 4 below.

In a significant development, Longstaff and Schwartz [2001] published an algorithm to estimate Continuation Values using simple, ordinary least squares regression techniques. Their algorithm has been widely adopted and is considered by many to be the market standard in American option valuation[10]. Longstaff-Schwartz's Least Squares Method ("LSM") is also discussed further in Section 4, below.

Although Monte Carlo techniques for pricing American options have come a long way since Tilley's first algorithm, they all still suffer from the same problem inherent in Monte Carlo simulation: the need to generate *and keep* the forward price paths and in order to work recursively backwards to determine stopping times. For a large number of paths each with many time steps, such as long-dated American options and/or options with high-dimensions (>3 state variables), this can lead to significant computer memory requirements with additional strain on processing power in managing that memory. In a recent article, Dutt and Welke [2008] show that this problem can be overcome in many instances by eliminating the need to store the forward paths. Rather than the traditional forward generation of price paths with subsequent backward recursion of stopping times, used by researchers in the past, Dutt and Welke show that geometric brownian motion (and some other stochastic processes[11]) can be generated backwards in time, thereby dispensing with the need for generating and storing forward price paths. For example, a standard brownian motion can be specified by the following formula:

$$\tilde{W}_{i-1} = \left(1-\frac{1}{i}\right).\tilde{W}_i + \sqrt{1-\frac{1}{i}}.\tilde{z}_i, \qquad i=1,2,\ldots,n \qquad (23)$$

where $\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_n$ are mutually independent, standard normal random variables.

They label this algorithm the "just-in-time" ("JIT") Monte Carlo procedure. While JIT Monte Carlo offers benefits when using certain stochastic processes, the authors acknowledge that it will be of little or no benefit in pricing some types of path-dependent American options, such as American-Asian options ("Amerasians"). Nonetheless, the JIT procedure is easy to implement and useful for high dimension options. Just where the boundaries are in terms of stochastic processes that can be described by similar backward-recursive processes, remains an area of continuing research (the article having only been published recently).

4 The Stochastic Mesh and LSM Algorithms

The Stochastic Mesh Method

As mentioned previously, the Broadie and Glasserman [2004] Stochastic Mesh method is an improvement to their 1997 Random Tree. Under the latter approach, at each time step on a particular asset path (ie. a node), the node is 'branched' into b subsequent nodes, where b is a user-specified number of branches per node. This forms a tree that looks similar to a non-recombining binomial lattice[12]. This tree quickly 'explodes' with branches as both b and the number of time steps increases. Their Stochastic Mesh method, however, remains linear in the number of branches. It does this by constraining each node's branches to the set connecting that node with all of the nodes appearing in the next time step. A simple example is shown in Figure 2:

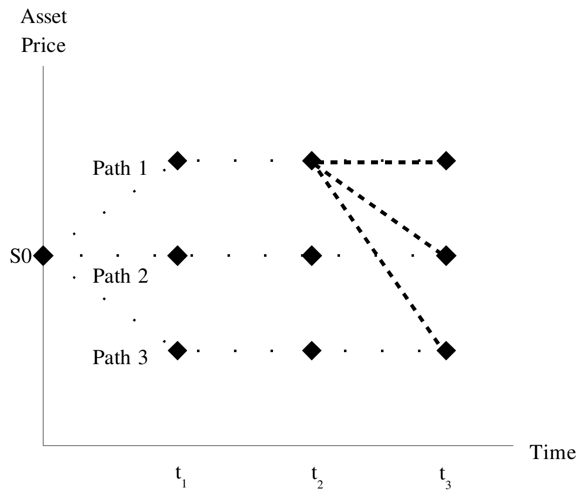

Figure 2: Beginnings of a simple Stochastic Mesh with 3 price paths and 3 time steps.

Here, three independent price paths for the asset (dotted lines) are set out over three time steps (t_1 to t_3). In practice, these are generated no differently from a normal Monte Carlo price path simulation. Also shown are three dashed lines connecting the node on Path 1 at time t_2 to the nodes on all price paths at time t_3. If this is repeated for Path 2 and Path 3 and then for each time step, then we end up with the full mesh:

8 "the computational requirements of the method grow exponentially in the number of exercise opportunities": Broadie and Glasserman [1997] at p. 1327.

9 Although published in 2004, the original working paper was in wide circulation from 1997 and is frequently referenced as such in many of published articles.

10 Wilmott [2006] p. 1279.

11 Namely the Ornstein-Uhlenbeck and Cox-Ingersoll-Ross processes.

12 Although the authors do point out that, unlike the binomial lattice, the nodes appear according to the order in which they are generated and *not* according to their node values.

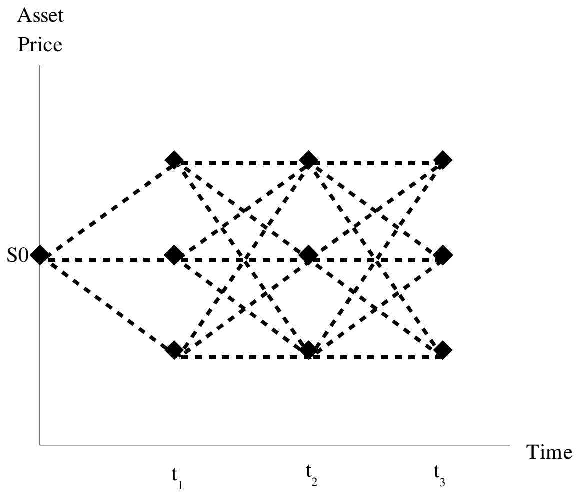

Figure 3: The full mesh with 3 price paths and 3 time steps.

These dashed lines hold a fundamental importance to the Stochastic Mesh method, for they represent the weighting scheme necessary for the calculation of Continuation Values at any particular node. Expanding on the notation from Equation (22) above, Broadie and Glasserman's estimator for the Continuation value at time *t* on asset path *i* is:

$$\hat{P}_{i,t} = max\left[K - S_{i,t},\ e^{r.\Delta t}.\left(\frac{1}{b}.\sum_{j=1}^{b}\hat{P}_{j,t+\Delta t}.w_{i,j}\right)\right] \quad (24)$$

where *j* is the index for the *b* nodes[13] appearing at time $t+\Delta t$ and $w_{i,j}$ is the weight attaching to the (dashed) line joining node *i* at time *t* with node *j* at time $t+\Delta t$. Broadie and Glasserman call equation 23 their Mesh Estimator.

Another estimator, the Path Estimator, is then calculated as part of the Broadie and Glasserman approach. The "Path Estimator" is determined by first simulating a separate set of price paths independent from those used to construct the mesh. For each of these paths and at each time step (starting at *t*=1 and moving forward in time) the exercise value (ie. the intrinsic value) is compared to a "Continuation Value" determined by a weighted average of all the Continuation Values contained in the mesh as at the next time step and as originally calculated in accordance with equation (24). In other words, the Path Estimator virtually ignores the previously specified mesh except when it comes to determining a suitable Continuation Value in order to compare the intrinsic values (as determined from the Path Estimator's simulation paths) and determine whether to stop (exercise) or not. The end result is a completely separate matrix of exercise cash flows (intrinsic values) which can be present valued and averaged to give the Path Estimator. For an American put option the Path Estimator is:

$$\hat{p}_t = \frac{1}{k}.\sum_{l=1}^{k} e^{r.(\hat{\tau}-t)}.\left(K - S_{l,\hat{\tau}}\right) \quad (25)$$

where $\hat{\tau}$ is the stopping time determined in according with the rule:

$$\hat{\tau}_k = min\left(t\ :\ K - S_{k,t} \geq \frac{1}{b}.\sum_{j=1}^{b}\hat{P}_{j,t+dt}.w_{k,j}\right) \quad (26)$$

where *k* is the asset path index for the paths generated in the Path Estimator simulation run.

From these two estimators, the importance of the weighting scheme (the $w_{i,j}$ and $w_{k,j}$) becomes readily apparent. However, while their importance to the method cannot be underestimated, what is not obvious from Broadie and Glasserman's 2004 paper is just how these weights are actually determined in order for the method to be coded into VBA, MatLab or C++ for practical application[14]. While Broadie and Glasserman do discuss the choice of weights in terms of marginal densities, (full) densities and transition densities, they acknowledge that it will often be the case that these density functions are either unknown or fail to exist[15]. Despite this practical limitation of the method the authors do prove that:

1. the Mesh Estimator is biased high, but consistent:

$$E\left(\hat{P}_0\right) \geq P_0 \quad but \quad E\left(\hat{P}_0\right) \rightarrow P_0 \quad as \quad b \rightarrow \infty \quad (27)$$

2. the Path Estimator is biased low but, consistent:

$$E\left(\hat{p}_0\right) \leq P_0 \quad but \quad E\left(\hat{p}_0\right) \rightarrow P_0 \quad as \quad b \rightarrow \infty \quad (28)$$

3. if the weights for the Mesh Estimator are not chosen carefully, there is a risk of exponential growth in the variance of the Mesh Estimator.

This last point has particular significance. The Stochastic Mesh method is fundamentally reliant on producing a tight confidence interval for both estimators. By taking the upper confidence bound of the Mesh Estimator and combining it with the lower confidence bound of the Path Estimator, a valid interval containing

13 Although it is not explicit in Broadie and Glasserman [2004] it is useful to keep in mind that *b* actually corresponds to the number of (simulated) asset paths.

14 The author contacted Mr P. Glasserman by e-mail requesting sample code used in the 2004 paper but received the somewhat helpful response "I do not have code to provide you. You may be able to get something from..." and pointed to a web site containing a library of C++ routines for a variety of options. As the author's knowledge of C++ is still in its infancy, it was not possible to identify the relevant code fragments from the mass of code, let alone translate them into into MatLab format.

15 Such as when the covariance matrix in a multivariate model does not have full rank.

the 'true' price of an American option can be obtained for a given level of confidence. Therefore, if the standard error of the Mesh Estimator is large (due to a poor choice of weights), then the desired 'tight' interval will, itself, be large thereby potentially rendering the method ineffective in its practical application.

As stated previously, practical application of the method requires knowledge of the transition density(s) of the underlying process(es) of the states variables, which can be problematic. In a subsequent paper, Broadie and Glasserman [2000] report using constrained optimisation techniques as an alternative way to determine mesh weights. They use two separate optimisation criteria: maximum entropy (“ME”) and ordinary least squares (“OLS”). Despite the apparent promise of these methods, their superficial treatment in the paper combined with mathematical errors, casts uncertainty over the practicality of these alternatives methods. As an illustration of the former, the authors acknowledge that the OLS approach offers an improvement in computational speed over ME, but it suffers from the drawback “that the weights produced [by OLS] are not guaranteed to be non-negative.”[16] Given the importance of the weighting scheme to the method, the prospect of generating negative weights renders OLS dead on arrival as a potential solution. Furthermore, salvaging OLS by constraining the weights to be non-negative removes any speed advantage it may have had over ME. The only saving grace of this conclusion is that the reader can safely ignore the fact that the matrix of OLS regression coefficients is $(B^TB)^{-1}B^Tb$ and not $B^T(BB^T)^{-1}b$ as their equation 15 states. It is also worth noting that their specified constraint on the weighting scheme:

$$\sum_{j=1}^{N} B_{kj}.w_{ij} = b_k \tag{29}$$

is expressed as:

$$\sum_{k,j} \lambda_k.\left(B_{kj}.w_{ij} - b_k\right) \tag{30}$$

in a Lagrange equation and not:

$$\sum_{k,j} \lambda_k.\left(B_{kj} - b_k\right).w_{ij} \tag{31}$$

as shown in their equation 9.

Despite these apparent typographical errors the ME approach offers some hope for those situations where transition densities cannot be determined in advance. What practical advantage it offers, though, remains to be seen. The introduction of optimisation algorithms adds further to computational effort that is already reportedly time consuming: “Broadie and Glasserman report that computing [the high and low interval for an American option on the maximum of five risky assets] requires slightly more than 20 hours a piece using a 266MHz Pentium II processor...while [Longstaff and Schwartz's own algorithm (discussed further below)] takes less than three minutes using the same CPU architecture (Pentium II) with a slightly higher clock speed (300MHz).[17]

While computer processing power has increased significantly since 1997, a crude application of Moore's Law[18] suggests a time of approximately 37 minutes would be needed to perform the same calculations (20 hours * 2^{-5}). Such high processing times may help explain the lack of widespread adoption of the Stochastic Mesh method by the financial community – a fact not shared by the least squares method of Longstaff and Schwartz [2001].

The Least Squares Method (“LSM”)

Longstaff and Schwartz [2001] use least squares regression to estimate Continuation Values. Formally, they assume the existence of a probability space (Ω, f, P) over the finite time horizon $(0, T)$ – where Ω is the set of all possible realisations of the stochastic economy during that time horizon (with ω representing a sample path), f is the sigma field of distinguishable events at time T and P is the probability measure defined on the elements of f. Consistent with assuming no-arbitrage opportunities, they also assume the existence of an equivalent martingale measure, Q, in the economy.

By taking expectations (with respect to this Q-measure) of all possible future cash flows and their present values, for a given f, the Continuation Value an be represented by the following conditional expectation function:

$$F\left(\omega;t_k\right) = E_Q\left[\sum_{j=k+1}^{K} e^{\left(-\int_{t_k}^{t_j} r(\omega,s)\,ds\right)}.C\left(\omega,t_j;t_k,T\right) \mid f\left(t_k\right)\right] \tag{32}$$

where:

$F(\omega;t_k)$ is the Continuation Value at time t_k;

E_Q is the expectation taken with respect to the Q-measure (see the discussion around footnote 3, above, on risk-neutral expectations);

16 Broadie and Glasserman [2000] at p. 42.

17 Longstaff and Schwartz [2001] at p. 142.

18 Moore's Law describes the phenomenon that computer processing power doubles approximately every 2 years. The law is attributed to the comments of Gordon E. Moore, the co-founder of Intel Corporation. While still a rule of thumb, it has shown remarkable resilience since 1965 when it was first postulated.

$e^{\left(-\int_{t_k}^{t_j} r(\omega,s)ds\right)}$ is the present value factor which, for this case, has deterministic or stochastic interest rate $r(\omega,s)$[19];

$C(\omega,t_j;t_k,T)$ is the cash flow path generated by the option conditional on the option not being exercised at or prior to time t and on the holder of the option following the optimal stopping strategy for all t_j, with $t_k < t_j \leq T$; and

$f(t_k)$ is the filtration at t_k.

The LSM uses least square regression to approximate $F(\omega;t_k)$ by assuming it can be represented as a linear combination of a countable set of basis functions measurable on the filtration, $f(t_k)$. Therefore, using the fitted value from this regression as the estimate of the Continuation Value, the LSM allows the 'exercise or continue' decision rule of equation (22) to be specified. By performing this regression algorithm in a recursive fashion (beginning at the option's expiry date and working backwards in time) a complete specification of the optimal exercise strategy can be obtained. The following algorithm demonstrates this for the one-asset case:

1. Generate an M x (N+1) matrix of M asset paths and N time steps (the first column being filled with the starting value (time t=0) of the asset, S_0);

2. Generate an additional M x (N+1) matrix of zero values which will be used to hold the entire simulation's cash flow profile, ie. the $C(\omega,t_j;t_k,T)$ values, as determined by the LSM algorithm (steps below);

3. For each of the M asset paths, generate the European payoffs and place these in column N+1 of the matrix generated in step 2;

4. At time T-Δt determine which of the M asset paths are showing a positive payoff if exercised at T-Δt, ie. determine which paths are "in-the-money" at time T-Δt, (Δt being the size of a single time step[20]);

5. For each of the in-the-money paths identified in step 4, present value their corresponding values from step 3 (ie. present value the step 3 value at time T to time T-Δt) and regress these present values on a user defined set of functions of the in-the-money values to generate a vector of regression coefficients;

6. Using the coefficients from step 5 estimate the Continuation Values [$F(\omega;t_k)$] at T-Δt for each in-the-money path using the same set of functions and in-the-money values as used in step 5;

7. For each in-the-money path, compare the Continuation Value with the intrinsic value (ie. at T-Δt) and use the early exercise rule of equation (22) to determine if the option is exercised at T-Δt or not;

8. If the option is exercised then set the corresponding cash flow amount in step 2 equal to the intrinsic value and set all subsequent cash flows for that path to zero;

9. Repeat steps 4 to 8 for the previous time period, ie. determine the in-the-money paths, Continuation Values and any exercise cash flows for time T-2Δt using the information in the payoff matrix (step 2) from all time periods after T-2Δt; and

10. Once column two of the payoff matrix (step 2) has been filled we have the complete cash flow matrix for the simulation – being a single cash flow along each of the M asset paths which occurs at a time period as determined by the above algorithm. These cash flows can then be present valued to time t=0 and averaged to give the LSM estimate of the option's value at t=0: this average being the Monte Carlo estimate $\hat{F}(\omega;t_k)$ of $F(\omega;t_k)$ (the true value).

GNU Octave[21] code for the above algorithm is contained in Appendix 1.

The reader will notice the code at Appendix 1 uses the polynomials [x^0, x^1, x^2, x^3] as the countable set of basis functions when implementing step 5 of the above algorithm. Appendix 2 contains the same code but without the comments and modified to incorporate the variance reduction technique of antithetic variates[22]. Unless otherwise stated, the Appendix 2 code has been used in the following Numerical Analysis.

5 Numerical Analysis

19 As stated previously, this dissertation assumes a fixed interest environment, hence, this formula becomes $e^{-r.(tj-tk)}$ (see the discussion around equation (17), above).

20 If we were valuing a Bermudan option, Δt would be the time between option expiry (T) and the exercise date closest to expiry.

21 The GNU Octave high level, matrix programming language has been used in this dissertation as (a) it is freely available to download and install, (b) it is faster than Microsoft's proprietary VBA language, and (b) it is largely MatLab compatible. In fact, the code as specified in the Appendices can be run in MatLab with minimal changes required eg. change 'endfor' to 'end' in each of the for...next loops.

22 Perusing the code, the reader will realise just how simple this variance reduction technique is to employ. Although figures are not reproduced in this dissertation, the antithetic variate technique results in standard errors of the LSM estimate being between 40% and 80% of non-antithetic (ie. 'raw') Monte Carlo estimates.

Longstaff and Schwartz produce numerical results for an American put option on a non-dividend paying stock that follows the risk-neutral diffusion process:

$$\frac{dS}{S} = r_f . dt + \sigma . \sqrt{dt} . dZ \qquad (33)$$

where S is the stock price, dS is the infinitesimally small change in S, r_f is the risk-free rate of interest, dt is an infinitesimally small increment of time, σ is the volatility of the rate of return of S, and dZ is the standard normal distribution function.

Table 1, below, contains their results under the columns headed “LS2001”. Additionally, the table employs their estimate of the exact American put option value obtained by the explicit finite difference method (“FDM”). The column headed “Current” contains the results from the LSM code contained in Appendix 2 and based on the following Monte Carlo simulation parameters (as originally used in LS2001):

M = 100,000 asset price paths (50,000 plus 50,000 antithetic);
N = 50 times steps per annum;
T = 1 or 2 year option term (ie. until maturity);
K = option strike price of 40;
S_0 = starting stock price of 36, 38, 40, 42 or 44;
r_f = the fixed, risk free interest rate of 6%pa; and
σ = 20%pa. or 40%pa.

Table 1: LSM Results for an American Put Option

Option #	*S_0*	*FDM LS2001*	*LSM Estimate LS2001*	*LSM Estimate Current*	*Standard Error LS2001*	*Standard Error Current*
T=1, σ=0.2						
1	36	4.478	4.472	4.471	0.010	0.009
2	38	3.250	3.244	3.256	0.009	0.009
3	40	2.314	2.313	2.302	0.009	0.009
4	42	1.617	1.607	1.628	0.007	0.008
5	44	1.110	1.118	1.109	0.007	0.007
T=1, σ=0.4						
6	36	7.101	7.091	7.113	0.020	0.019
7	38	6.148	6.139	6.144	0.019	0.019
8	40	5.312	5.308	5.313	0.018	0.018
9	42	4.582	4.588	4.588	0.017	0.017
10	44	3.948	3.957	3.934	0.017	0.016
T=2, σ=0.2						
11	36	4.840	4.821	4.828	0.012	0.011
12	38	3.745	3.735	3.744	0.011	0.011
13	40	2.885	2.879	2.877	0.010	0.011
14	42	2.212	2.206	2.206	0.010	0.010
15	44	1.690	1.675	1.681	0.009	0.009
T=2, σ=0.4						
16	36	8.508	8.488	8.499	0.024	0.023
17	38	7.670	7.669	7.639	0.022	0.022
18	40	6.920	6.921	6.905	0.022	0.022
19	42	6.248	6.243	6.252	0.021	0.021
20	44	5.647	5.622	5.615	0.021	0.021

An importance difference between LS2001 and Current is the choice of basis functions used in the regression. Longstaff and Schwartz used a constant and three Laguerre polynomials[23] whereas the Current results, as shown by the code in Appendix 2, use simple powers: a constant ($x^0 = 1$) and the first three powers in the series.

The reader can see that the choice of functions does not have a big impact on the LSM estimate. Differences between the LS2001 values and the Current values are small and both are very close to the FDM put option value. In fact, the FDM amount is within – at most – 1.54 standard errors of the Current LSM estimate (1.67 in the case of FS2001). These results are in line with Longstaff and Schwartz who reported “results virtually identical” to their Laguerre polynomials when they regressed on a constant and the first three powers (2004, p. 142).

Longstaff and Schwartz also claim “using more than three basis functions... does not change the numerical results” (2001, p.126). To test this, variations of the Appendix 2 code were run by increasing the number of basis functions incrementally up to x^9. The same assumptions were kept as used in Table 1 save for holding σ=40%, T=1 and using S_0=36. The FDM value under these assumptions (per Longstaff and Schwartz) is 7.101. The results are shown in Table 2:

Table 2: Effect of Increasing the Number of Basis Functions

Basis Functions	Estimate	Std Error	Process Time*
$1\ x^1$	7.016	0.020	6.9
$1\ x^1 x^2$	7.080	0.019	8.4
$1\ x^1 x^2 x^3$	7.112	0.019	12.3
$1\ x^1 x^2 x^3 x^4$	7.104	0.019	12.8
$1\ x^1 x^2 x^3 x^4 x^5$	7.095	0.019	13.1
$1\ x^1 x^2 x^3 x^4 x^5 x^6$	7.098	0.019	15.1
$1\ x^1 x^2 x^3 x^4 x^5 x^6 x^7$	7.109	0.019	17.2
$1\ x^1 x^2 x^3 x^4 x^5 x^6 x^7 x^8$	7.114	0.019	19.3
$1\ x^1 x^2 x^3 x^4 x^5 x^6 x^7 x^8 x^9$	7.090	0.019	21.8

* Time (in seconds) to perform the simulation using a 1.8GHz AMD Opteron processor with 4GB of DDR2-RAM running Octave 3.0.0 on the Ubuntu 8.04 GNU/Linux distribution.

A number of points can be made about the results in Table 2. Firstly, they are in accordance with Longstaff and Schwartz's claim: save for the first result ($1\ x^1$), there is little difference in the LSM estimates when higher powers are included. Secondly, the third result ($1\ x^1 x^2 x^3$) remains close to the estimate from Table 1's simulation run. Thirdly, the standard errors of the estimates remain

23 The first three in the Laguerre series.

similar[24] irrespective of the number of basis functions employed in the regression. Finally, the computational time almost doubles as we move from three basis functions to nine, removing any remaining doubt the reader may have that higher order basis functions are an apparent waste of time – for single-asset American option valuation, at least.

As a final demonstration of the LSM, the case of a deeply-out-of-the-money option can be examined. As stated earlier, the LSM algorithm regresses only those paths where the option is in-the-money. This raises an interesting question as to the algorithm's accuracy when pricing American options that are deeply out-of-the-money. In such a scenario, the number of independent variables may be few and far between such that the resulting regression provides estimates of Continuation Values which are equal, or close, to the actual dependent variables. This 'perfect foresight' results in systematic overestimation of an option's value. This point is not lost on Dutt and Welke [2008] who demonstrate that the number of paths in a Monte Carlo simulation needs to be close to 10^6 for the LSM to give an accurate approximation of the price of a deeply out-of-the-money American put option. Anything less and the LSM shows consistent overestimation, suggesting the LSM may not be appropriate for valuing deeply-put-of-the-money options irrespective of their complexity. Unfortunately, Octave showed its limitations when trying to reproduce Dutt and Welke's results. It consistently refused to price Dutt and Welke's deeply-out-of-the-money option ($K=20$, $S_0=100$ and $\sigma=0.4$) complaining that it was being asked to regress non-numeric matrices. This is a disappointing result as up until now Octave had shown no signs of any problems. At first it was thought there was some problem with the code in Appendix 2, however, the MatLab code supplied by Dutt and Welke in their paper was adjusted for Octave and run. This is, perhaps, an area for future research for the Octave project.

6 Conclusion

This dissertation has examined two recent algorithms for applying Monte Carlo simulation techniques to the problem of pricing American-style options.

The Stochastic Mesh algorithm of Broadie and Glasserman [2004] remains an interesting area for future research but practical implementation of the approach remains unclear. It requires prior knowledge of transition densities for the underlying state variables used which may or may not be known depending upon the type of processes chosen for those state variables. Alternatives have been proposed when knowledge of transition densities are not known, but these are poorly documented and contain glaring mathematical errors.

The Least Squares Method of Longstaff and Schwartz [2001], however, is very different. It is well documented and relatively simple to implement. It produces estimates that are very close to other numerical methods (finite differences and binomial lattices) and requires only simple linear regression techniques to implement. It does, however, suffer from a requirement for increased processing time when valuing deeply-out-of-the-money options, as the number of paths in the simulation must be of the order 10^6 or more. In this respect, an interesting area for future research is using parallel processing techniques to reduce simulation times. Monte Carlo simulation lends itself well to such techniques and there are many instances in the literature on American option pricing where reference is made to the potential promise of parallel processing to speed up simulation times. As Longstaff and Schwartz state "from the perspective of the LSM algorithm, the only constraint on parallel computation is that the regression needs to use cross-sectional information in the simulation... [and] there are many ways in which regressions could be estimated using individual CPUs and then aggregated across CPUs to form a composite estimate of [the Continuation Value]" at p. 144.

While the author is not aware of any 'parallelized' version of MatLab, there are many GNU/Linux distributions available, such as PelicanHPC and ParallelKnoppix, that incorporate parallel processing techniques into their distribution of Octave. However, the author's experience with Octave suggests it may still not be appropriate to use Octave for valuing deeply-out-of-the-money options. However, Octave is an open source project and, as such, has a fast development cycle. It may be that the problem experienced in this dissertation will be overcome in the near future and parallel processing using Octave will open a Pandora's Box to American option valuation.

24 To three decimal places, that is. There is variability at more than three decimal places.

Appendix 1

Octave 3.0.0 Code Listing: Longstaff-Schwartz Least Squares Method ("LSM")

```
function [Output] = lsm2 (M, N, T, rf, K, S0, vol)

start1=time();      % start the clock!
dt=T/N;             % dt  = size (in years) of each time step

%% generate MxN matrix of (standard normal) random numbers
Z=randn(M,N);

% generate Mx(N+1) asset price path matrix (M=path, N=time)
S=[S0*ones(M,1), S0*cumprod( exp( (rf-0.5*vol^2)*dt + vol*sqrt(dt)*Z ), 2)];

% create Mx1 vectors to hold cash flows at exercise (IV) and PV of cash flows (EV)
IV=zeros(M,1);
EV=zeros(M,1);

% set CV to European exercise values (needed to 'start the ball rolling')
EV=max( 0, K-S(:,N+1) );

% loop through each column (ie. time step) but in reverse, starting at 2nd last column (N)
for t=N:-1:2

        % PV the exercise values
        EV=EV.*exp(-rf*dt);

        % set IV to intrinsic value for this pass through the loop
        IV=max(0,K-S(:,t));

        % create vector of binaries (0=path is out-of-the-money, 1=path is in-the-money)
        itm=IV>0;

        % find asset price associated with the in-the-money paths, ie. x is the independent variable
        x=S(itm,t);

        % define the set of basis functions we're going to use for the LSM algorithm
        X=[ones(length(x),1), x, x.^2, x.^3];

        % set up the vector for the Y dependent variable (ie. pv of next period's IV)
        Y=EV(itm);

        % find regression coefficients 'b'
        b=regress(Y,X);

        % set up Mx1 colum vector of zeros (not itm) and Continuation Values (itm)
        CV=zeros(M,1); CV(itm)=X*b;

        % create vector of binaries: 0=no early exercise (IV<CV), 1=early exercise (IV>CV)
        EE=IV>CV;

        % clear current IV column of IVs where there is no early exercise
        IV(~EE)=0;

        % if there is early exercise clear all 'subsequent' exercise values for that path
        EV(EE)=0;

        % finally, add current intrinsic values to remaining PV'd exercise values in preparation for next loop
```

```
        EV=EV+IV;

endfor

V=mean(EV)*exp(-rf*dt);          % calculate LSM estimator
s=std(EV)/sqrt(M);               % calculate standard error of LSM estimator
end1=time();                     % stop the clock!
Time=end1-start1;                % calculate how long it took to compute
CI=1.96*s/sqrt(M);               % determine confidence interval width
Output=[M, N, S0, vol, T, Time, V, s, V-CI, V+CI];

endfunction
```

Appendix 2

Octave 3.0.0 Code Listing: LSM with Antithetic Variates

```
function [Output] = lsm2anti (M, N, T, rf, K, S0, vol)

start1=time();      % start the clock!

dt=T/N;             % dT  = size (in years) of each time step

%% generate MxN matrix of (standard normal) random numbers (M/2 + M/2 antithetic)
Z1=randn(M/2,N); Z=[Z1;-Z1];

S=[S0*ones(M,1), S0*cumprod( exp( (rf-0.5*vol^2)*dt + vol*sqrt(dt)*Z ), 2)];
IV=zeros(M,1);
EV=zeros(M,1);
EV=max( 0, K-S(:,N+1) );
for t=N:-1:2
        EV=EV.*exp(-rf*dt);
        IV=max(0,K-S(:,t));
        itm=IV>0;
        x=S(itm,t);
        X=[x.^0, x.^1, x.^2, x.^3];
        Y=EV(itm);
        b=regress(Y,X);
        CV=zeros(M,1); CV(itm)=X*b;
        EE=IV>CV;
        IV(~EE)=0;
        EV(EE)=0;
        EV=EV+IV;
endfor
V=mean(EV)*exp(-rf*dt);
s=std(EV);
end1=time();
Time=end1-start1;
CI=1.96*s/sqrt(M);
Output=[M, N, S0, vol, T, Time, V, s/sqrt(M), V-CI, V+CI];

endfunction
```

Bibliography

Barraquand, J. and D. Martineau — 1995, "Numerical Valuation of High Dimensional Multivariate American Securities", *The Journal of Financial and Quantitative Analysis*, Vol. 30(3), pp. 383-405.

Black, F and M. J. Scholes — 1973, "The Pricing of Options and Corporate Liabilities", *Journal of Political Economy*, Vol. 81, pp. 637-659.

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

Boyle, P. P. — 1988, "A Lattice Framework for Option Pricing with Two State Variables", *Journal of Financial and Quantitative Analysis*, Vol. 23, pp. 1-12.

Boyle, P. P., Evnine, J. and S. Gibbs — 1989, "Numerical Evaluation of Multivariate Contingent Claims", *The Review of Financial Studies*, Vol. 2(2), pp. 241-250.

Boyle, P. P., Broadie, M. and P. Glasserman — 1997, "Monte Carlo Methods for Security Pricing", *Journal of Economic Dynamics and Control*, Vol. 21, pp. 1267-1321.

Brennan, M. J. and E. S. Schwartz — 1978, “Finite Difference Methods and Jump Processes Arising in the Pricing of Contingent Claims: A Synthesis”, *Journal of Financial and Quantitative Analysis*, Vol. 12(3), pp.461-474.

Broadie, M. and P. Glasserman — 1997, "Pricing American-Style Securities Using Simulation", *Journal of Economic Dynamics and Control*, Vol. 21, pp. 1323-1252.

Broadie, M. and P. Glasserman — 2000, “Pricing American Options by Simulation Using a Stochastic Mesh with Optimized Weights”, in *Probabilistic Constrained Optimization: Methodology and Applications*, Kluwer Academic Publishers, Editor: SP Uryasev, pp. 32-50.

Broadie, M. and P. Glasserman — 2004, "A Stochastic Mesh Method for Pricing High-Dimensional American Options", *The Journal of Computational Finance*, Vol. 7(4).

Carriere, J. F. — 1996, "Valuation of the Early-Exercise Price for Derivative Securities Using Simulations and Non-parametric Regression", *Insurance: Mathematics and Economics*, Vol. 19, pp. 19-30.

Cox, J. C., Ross, S. A. and M. Rubenstein — 1979, "Option Pricing: A Simplified Approach", *Journal of Financial Economics*, Vol. 7, pp. 229-263.

Dutt, S. K. and G. M. Welke — 2008, "Just-In-Time Monte Carlo for Path-Dependent American Options", *The Journal of Derivatives*, Summer, pp. 29-47.

Hull, J and A. White — 1988, "The Use of the Control Variate Technique in Option Pricing", *The Journal of Financial and Quantitative Analysis*, Vol. 23(3), pp. 237-251.

Longstaff, F. A. and E. S. Schwartz — 2001, "Valuing American Options by Simulation: A Simple Least-Squares Approach", *The Review of Financial Studies*, Vol. 14(1), pp. 113-147.

Omberg, E. 1987, "The Valuation of American Put Options with Exponential Exercise Policies", *Advances in Futures and Options Research*, Vol. 2, pp. 117-142.

Tilley, J. A. 1993, "Valuing American Options in a Path Simulation Model", *Transactions of the Society of Actuaries*, Vol. 45, pp. 83-104.

Wilmott, P. 2006, *Paul Wilmott on Quantitative Finance*, , 2nd Edition, John Wiley & Sons Ltd, Chichester.