A Mixed PDE/Monte Carlo Approach as An Efficient Way to Price under High-Dimensional Systems

---

Xing Xian Ang

Christ Church
University of Oxford



A thesis submitted for the degree of
Master of Science
Trinity 2013

Abstract

We propose to price derivatives modelled by multi-dimensional systems of stochastic differential equations using a mixed PDE/Monte Carlo approach. We derive a stochastic PDE where some of the coefficients are conditional on stochastic ancillary factors. The stochastic PDE is solved with either analytical or finite difference methods, where we simulate all the ancillary processes using Monte Carlo. The multilevel technique has also been introduced to further reduce the variance. The combined method showed over 80% cost reduction for the same accuracy, in pricing a barrier option in an FX market with stochastic interest rate and volatility (which is usually expensive to work with) , when compared to the pure Monte Carlo simulation.

Contents

List of figures **i**

Introduction **2**

1 Introduction to Hybrid Methods **4**

1.1 Pricing with Monte-Carlo / Analytic Method 8

1.1.1 Numerical Experiments 10

2 Multi-Dimensional Systems **14**

2.1 Problem Formulation 14

2.1.1 Decomposing Brownian Motion 15

2.1.2 Numerical Experiments 18

3 Multi-Level Simulation **23**

3.1 Numerical Experiments 25

4 Final Words **28**

5 Appendix **29**

Forewords

Acknowledgements

I would like to thank my supervisor, Dr. Chistopher Reisinger for his guidance, patience, valuation suggestions as well as the lectures given on Finite Differences. Without his help this thesis could not be of this quality.

I would also like to thank Dr. Jeff Dewynne and Prof. Mike Giles who have taught me on Monte Carlo methods. Many of the ideas in the paper are extension and application of the materials from the lectures.

On top of that, I would like to take this opportuinity to say thanks to my coursemates Vincent Tan and Hendrick Brackmann whom I have learnt a lot from throughout the durcation of this MSc course.

Introduction

Consider a Heston stochastic volatility model for a stock price under its risk-neutral dynamics.

$$\mathrm{d}S_t = S_t(r\mathrm{d}t + \sqrt{v_t}\mathrm{d}W_t^1) \tag{1}$$

$$\mathrm{d}v_t = k(\theta - v_t)\mathrm{d}t + \lambda\sqrt{v_t}\mathrm{d}W_t^2) \tag{2}$$

To run a conventional full Monte-Carlo simulation, we would split the time to maturity T into N steps with step size δt (i.e. $T = N\delta t$). Then we have a timestepping scheme, for example, the following Euler-Maruyama time-stepping with the initial value S_0, v_0 of

$$S_{i+1} = S_i(1 + r\delta t + \sqrt{v_i}(\rho N_{0,1}^1 + \sqrt{1-\rho^2}N_{0,1}^2)\sqrt{\delta t} \tag{3}$$

$$v_{i+1} = v_i + k(\theta - v_i)\delta t + \sqrt{v_i}\lambda N_{0,1}^2\sqrt{\delta t} \tag{4}$$

where $N_{0,1}^i$'s are realizations of two independent $\mathcal{N}(0,1)$ variables. And then M realizations of the stock price paths $\{S_i^m\}_{i=0,m=1}^{N,M}$ and the variance paths $\{v_i^m\}_{i=0,m=1}^{N,M}$ are simulated. Finally the estimate of the value of a particular option $P(0)$ with payoff $G(S)$, at time $t = 0$ is set to be $P(0) = \frac{e^{-rT}}{M}\sum G(S^m)$. While this method is simple and straightforward, it is computationally expensive. For non path-dependent options, the payoff will only depend on the final value of S^m at time T.

Alternatively, one can try to solve this using a finite-difference approach. Using Ito's calculus or many other methods, we can derive the Black-Scholes PDE that the option price $u(S_t, t; v_t)$ satisfies. For a simple vanilla put with maturity T and strike K, we have

$$\partial_t u + rS\partial_S u + \frac{1}{2}vS^2\partial_{SS}u + \rho\lambda v\partial_{Sv}u + k(\theta - v)\partial_v u + \frac{1}{2}\lambda^2 v\partial_{vv}u - ru = 0$$
$$u(S, v, T) = (K - S)^+$$

This is computational expensive again because we are working with two-spatial dimensions. In the case where more than 2 stochastic processes are involved, the number of grid-points required increase exponentially and we face the *curse of dimensionality*.

Of course there are many variance reduction methods which may help to reduce the number of samples required for a particular required precision. In the next section we will introduce a different approach to deal with such situations.

Chapter 1

Introduction to Hybrid Methods

The idea of combining Monte-Carlo and Finite Difference methods was first introduced by Lipp, Loeper and Pironneau ([10]) where the standard Heston model was solved to price a standard vanilla option and a knock-out option. However, we aim to demonstrate in this paper that this technique can be extended and applied to solve a practical problems in the financial industry. High dimensional models with stochastic interest rates and stochastic volatility have often been used to price long-dated FX options. Unfortunately, the high dimensionality often makes simulation extremely expensive. We believe that the mixed method is a good candidate as a scheme to solve the problem efficiently.

The idea is that, for a standard Heston model, instead of generating a large number of 2-dimensional samples $(N^1_{0,1}, N^2_{0,1})$, one can first simulate a large number of paths for v_t (and hence the volatility process $\sigma_t = \sqrt{v_t}$), then derive the PDE conditional on the simulated v_t path in the S space. Ultimately this PDE is either solved analytically or numerically.

To illustrate this, let us rewrite the SDE (1) by decomposing the pair of correlated Brownian paths (W^1_t, W^2_t) into two uncorrelated Brownian paths $(\tilde{W_t}^1, \tilde{W_t}^2)$, where we maintain the correlation structure by having $\mathrm{d}W^1_t = \sqrt{1-\rho^2}\mathrm{d}\tilde{W_t}^1 + \rho\mathrm{d}\tilde{W_t}^2$.

$$\mathrm{d}S_t = S_t(r\mathrm{d}t + \sqrt{v_t}(\sqrt{1-\rho^2}\mathrm{d}\tilde{W_t}^1 + \rho\mathrm{d}\tilde{W_t}^2) \tag{1.1}$$

$$\mathrm{d}v_t = k(\theta - v_t)\mathrm{d}t + \lambda\sqrt{v_t}\mathrm{d}\tilde{W_t}^2)) \tag{1.2}$$

After simulating the path for v_t, the only stochasticity in S_t comes from $\tilde{W_t}^1$. Hence we can also write the dynamics of S_t in terms of its conditional drift μ_t.

$$\mathrm{d}S_t = S_t(\mu_t\mathrm{d}t + \sigma_t\sqrt{1-\rho^2}\mathrm{d}\tilde{W^1_t}) \tag{1.3}$$

To derive the conditional drift μ_t we need to make sure that the solution to (1.3) tends

to the solution of (1.1) as $\delta t \to 0$.

$$\hat{S}_t \to S_0 \exp\left(\int_0^t r - \frac{\sigma_s^2}{2}\ \mathrm{d}s + \rho\int_0^t \sigma_s \mathrm{d}\tilde{W}_s^2 + \sqrt{1-\rho^2}\int_0^t \sigma_s \mathrm{d}\tilde{W}_s^1\right) \tag{1.4}$$

We split our time-to-maturity into N intervals, with $t_{i+1} - t_i = \frac{T}{N}$. We can prove that, if we set, $\forall i = 1 \dots N$

$$\mu_i = r - \frac{\rho^2}{2}\sigma_i^2 + \rho\sigma_i \frac{\delta\tilde{W}_i^2}{\delta t} \tag{1.5}$$

then (1.3) will converge to the solution of the Heston model (1).

From (1.3), we can write down the dynamics of $\log S_t$

$$\mathrm{d}(\log S_t) = (\mu_t - \frac{1}{2}\sigma_t^2(1-\rho^2))\mathrm{d}t + \sigma_t\sqrt{1-\rho^2}\mathrm{d}\tilde{W}_t^1$$

which can then be easily solved to obtain

$$S_t = S_0 \exp\left(\int_0^t \mu_s - \frac{1}{2}(1-\rho^2)\sigma_s^2\ \mathrm{d}s + \sqrt{1-\rho^2}\int_0^t \sigma_s\ \mathrm{d}\tilde{W}_t^1\right) \tag{1.6}$$

Then compare (1.6) with (1.4) and we see that μ_t must satisfy

$$\int_0^t \mu_s \mathrm{d}s = \int_0^t r - \frac{\sigma_s^2}{2}\ \mathrm{d}s + \rho\int_0^t \sigma_s\ \mathrm{d}\tilde{W}_t^2 + \int_0^t \frac{1}{2}(1-\rho^2)\sigma_s^2\ \mathrm{d}s \tag{1.7}$$

$$= \int_0^t r - \frac{1}{2}\rho^2\sigma_s^2\ \mathrm{d}s + \rho\int_0^t \sigma_s\ \mathrm{d}\tilde{W}_t^2 \tag{1.8}$$

Hence if we re-write the stochastic integral as approximating discrete sums

$$\sum_{i=1}^N \mu_i \delta t = \sum_{i=1}^N r\delta t - \frac{\rho^2}{2}\sum_{i=1}^N \sigma_i^2\delta t + \rho\sum_{i=1}^N \sigma_i\delta\tilde{W}_i^2$$

$$\Rightarrow \qquad \mu_i = r - \frac{\rho^2}{2}\sigma_i^2 + \rho\sigma_i\frac{\delta\tilde{W}_i^2}{\delta t}$$

we have the expression for μ_i as in (1.5). Of course this is a rather heuristic argument and is nowhere rigorous. Essentially we discretize the stochastic integrals so that a simple comparison can be made. Then we find the discretized conditional drifts for simulation, similar to what was done in ([10]). To derive the continuous conditional drift would require advance analysis on stochastic PDE, which is not the focus of this paper.

With the expression for μ_t derived, we can write down the conditional Black-Scholes PDE for u,

$$\partial_t u + \mu_{t_i} S_t \partial_S u + \frac{1}{2}(1-\rho^2)\sigma_t^2 S_t^2 \partial_S S u - ru = 0 \tag{1.9}$$

Note that, in (1.9), the process μ_t is a piece-wise constant process, with jumps at each time t_i. While (1.10) is a more accurate representation, (1.9) allow us to write down the discretized PDE later. One problem about the representation in (1.9) is that it implies μ_t will tend to the time derivative of a Brownian motion. This obviously does not make sense as the Brownian path is not differentiable.

The precise result should be the stochastic PDE below, which describes the conditional dynamics of $u(S_t, t; \sigma_t)$, the value of the option price at time t.

$$du = -\left[\left(r - \frac{\rho^2}{2}\sigma_t^2\right) S_t \partial_S u + \frac{1}{2}(1-\rho^2)\sigma_t^2 S^2 \partial_{SS} u - ru\right] dt - \left(\rho \sigma_t S_t \partial_S u\right) d\tilde{W}_t^2 \tag{1.10}$$

In this case the boundary conditions and terminal conditions depend on the payoff of the option. For example, in the case where $u(S_t, t)$ is the price of the put option with strike K and maturity T, then the terminal condition is just $u(S, T) = (K - S)^+$, defined on the range of S. Strictly speaking, when solving the PDE numerically we always truncate the range of S and solve the PDE on $[0, T] \times [0, S_{max}]$ and assume that the *gamma* of the option vanishes at S_{max}.

If we write out fully the explicitly discretized equation we have the following scheme:

$$\begin{aligned} u_m^n = {} & u_{m-1}^{n+1}\left(\frac{1-\rho^2}{2\Delta S^2}\sigma_n^2 S_m^2 - \frac{\mu_n S_m}{2\Delta S}\right)\Delta t \\ & + u_m^{n+1}\left[1 - \left(\frac{1-\rho^2}{\Delta S^2}\sigma_n^2 S_m^2 + r\right)\Delta t\right] \\ & + u_{m+1}^{n+1}\left(\frac{1-\rho^2}{2\Delta S^2}\sigma_n^2 S_m^2 + \frac{\mu_n S_m}{2\Delta S}\right)\Delta t \end{aligned}$$

where $u_m^n = u(n\Delta t, m\Delta S)$. As we are discretizing the stochastic integral via μ_t, one has to be very careful in defining the Euler discretization carefully to ensure the scheme converges to the correct Ito integral. Since in our case the integrand σ_t of the Ito integral is time dependent, the coefficients of the Brownian increment over (t_i, t_{i+1}) must be evaluated at the start of the time interval. That is, we should choose σ_{t_i} to be the coefficient of the

Brownian increment.

The explicit scheme above is only conditionally stable. There is an upper bound (which is a function of the spatial step size) for the timestep size. Hence if the time step got too huge the scheme will be unstable. One may opt to use an implicit scheme instead which is unconditionally stable although slightly more expensive to implement. This is because usually a large linear system will have to be solved at each time step.

To find out the stability of this scheme and get the bound for the time step size Δt, we carry out a von Neumann analysis. We will not cover the theory and details here, for more information please refer to ([13]).

If we let $R^{-n}e^{-ikm} = u_m^n$ be the ansatz to the discretized equation, then we need $|R| < 1$ for the scheme to be stable. To simplify the notation, we denote A_m^n and B_m^n as the following:

$$A_m^n = \frac{1-\rho^2}{2\Delta S^2}\sigma_n^2 S_m^2$$
$$B_m^n = \frac{\mu_n S_m}{2\Delta S}$$

This give us

$$u_m^n = u_{m-1}^{n+1}\left(A-B\right)\Delta t + u_m^{n+1}\left[1-\left(2A+r\right)\Delta t\right] + u_{m+1}^{n+1}\left(A+B\right)\Delta t$$

By substituting the ansatz into the discretized equation above, we find that we have the following upper bound on Δt

$$\Delta t < \frac{A\left(1-\cos k\right)+r/2}{\left[A(1-\cos k)-r/2\right]^2+\left[B\sin k\right]^2}$$

A_m^n is a known bounded set of numbers of $O(\Delta S^{-2})$ but B_m^n is only known after we simulate the volatility path. According to the expression derived in (1.5), $\mu \sim \mathcal{N}(r_i - \frac{\rho^2\sigma_i^2}{2}, \frac{\rho^2\sigma_i^2}{\delta t})$. We should expect μ to be roughly of $O(\Delta t^{-1/2})$ and hence $B^2 \sim O(\Delta t^{-1}\Delta S^{-2})$. Given the fact that μ is itself random, there is a tiny chance that we can be extremely unlucky to have really large μ. A more rigorous way is actually to analyze the mean-square stability as what was done in ([8]) and ([6]). For more details, one should refer to ([6]) where a mean-square staibility analysis has been done on Milstein finite difference where the stochastic PDE is very similar to our scheme here.

The numerator is of $O(\Delta S^{-2})$ and the denominator is of order $O(\Delta S^{-4})+O(\Delta S^{-2}\Delta t^{-1})$. Hence we find that the upper bound for Δt for a stable scheme is of $O(\Delta S^2)$. This is similar

to the explicit Euler discretization of a standard Black-Scholes PDE.

In fact this is optimal because the central explicit scheme used in our discretized PDE has the convergence order of $O(\Delta t, \Delta S^2)$. Hence even if the implicit scheme is used, one may ultimately stick to this ratio as well.

1.1 Pricing with Monte-Carlo / Analytic Method

Considering a simple European option, we can simplify this pricing exercise further as there is an analytical solution.

Using the fact that $\int_0^t \sigma_s d\tilde{W}_s^1 \sim \mathcal{N}(0, \int_0^t \sigma_s^2 ds)$ by Ito's isometry, we can rewrite the expression (1.4) with the following, where $Z \sim \mathcal{N}(0,1)$:

$$S_T(Z) = S_0 \exp\left(\int_0^T r - \frac{\sigma_s^2}{2} \, \mathrm{d}s + \rho \int_0^T \sigma_s \mathrm{d}\tilde{W}_s^2 + \sqrt{(1-\rho^2)\int_0^T \sigma_s^2 \, \mathrm{d}s} \; Z \right) \tag{1.11}$$

and the discrete path

$$\hat{S_T}(Z) = S_0 \exp\left(\left(r - \frac{1}{2}\bar{\sigma}^2 \right) T + \rho \sum_{i=1}^{N} \sigma_i \delta \tilde{W}_i^2 + \bar{\sigma}\sqrt{(1-\rho^2)T} \; Z \right) \tag{1.12}$$

where $\bar{\sigma}^2 = \sum_{i=1}^N \frac{\sigma_i^2}{N}$. Hence we can think of $\hat{S_T}$ as a function of Z.

For any European Option (or non path-dependent option) with payoff $G(S_T)$, the solution can be expressed in the following iterated expectation:

$$u(0, s_0) = \mathbb{E}^{\mathbb{Q}^2}\left[\mathbb{E}^{\mathbb{Q}^1}\left[e^{-rT} G\left(S_T(Z)\right) \middle| \mathcal{F}_T \right] \middle| S_0 = s_0 \right] \tag{1.13}$$

$$= \mathbb{E}^{\mathbb{Q}^2}\left[\int_{\mathbb{R}+} G(S_T(z)) \phi_{0,1}(z) dz \;\middle|\; S_0 = s_0 \right] \tag{1.14}$$

where $\mathcal{F}$ is the filtration for the process σ_t, $\mathbb{E}^{\mathbb{Q}^2}[\,.\,]$ is the expectation taken over all $\tilde{W}_t^2$ paths (or the σ_t paths), $\mathbb{E}^{\mathbb{Q}^1}[\,.\,]$ is the expectation taken over all $\tilde{W}_t^1$ paths (or the S_t paths), and $\phi_{0,1}(.)$ is the density of $\mathcal{N}(0,1)$. Then one has to evaluate the integral. There may be an analytical solution, as with the case of a vanilla call or vanilla put. Or else, it can always be numerically evaluated via a quadrature formula or even via a simulation for

S_T, in which case this is just a full Monte Carlo exercise.

If $G(S) = (K - S)^+$ then the value of the integral can be obtained directly from the Black-Scholes formula. To do this we compare (1.12) with the following standard Black-Scholes solution with interest rate r, dividend rate q and volatility $a\tilde{\sigma}$.

$$S(Z) = S_0 \exp\left(\left(r - q - \frac{a^2\tilde{\sigma}^2}{2}\right)T + a\tilde{\sigma}\sqrt{T}Z\right) \tag{1.15}$$

and a is a constant in $\mathbb{R}$. Hence we obtain that

$$\tilde{\sigma} = \bar{\sigma} \tag{1.16}$$

$$a = \sqrt{1 - \rho^2} \tag{1.17}$$

$$q = \frac{\rho^2\bar{\sigma}^2}{2} - \frac{\rho}{T}\sum_{i=1}^{N}\sigma_i\delta\tilde{W}_i^2 \tag{1.18}$$

We can then use the Black-Scholes formula for a vanilla put option to obtain the price conditional on a particular realization of σ_t path.

Note : This contradicts the results provided by the original authors Loeper and Pironneau in their paper ([10]). However it can be verified with numerical experiments that the results derived in this paper serves as the correction required for the method to work. As $\Delta t \to 0$, we indeed obtained convergence for this particular European put example.

$$\begin{aligned} u(0, S_0) &= K\Phi_{0,1}(-d_2) - S_0 e^{-qT}\Phi_{0,1}(-d_1) \\ d_1 &= \frac{\ln\frac{S_0}{K} + \left(r - q + \frac{1}{2}a^2\tilde{\sigma}^2\right)T}{a\tilde{\sigma}\sqrt{T}} \\ d_2 &= d_1 - a\tilde{\sigma}\sqrt{T} \end{aligned}$$

Of course as this is only the analytical value conditional on a particular path of σ_t. The final estimation should be the average of all $u(0, S_0; \mathcal{F}_T)$, conditional on each realization of the volatility path.

1.1.1 Numerical Experiments

We now conduct some numerical tests to verify the method and check if we get consistent results with pure Monte-Carlo method. We also compare the mean and standard deviation of the solution given by both methods.

The option to be priced here is a put option with parameters: $S_0 = 100$, $K = 90$, $r = 0.05$, $\sigma_0 = 0.4$, $\theta = 0.16$, $k = 5$, $\lambda = 0.2$, $T = 0.5$, $\rho = -0.25$. We also use M_{mixed} samples for the mixed algorithm and M_{mc} samples for each of S_t and σ_t in the full Monte-Carlo (or MC) algoritms. For the path generation, and we use the exact solution for S_t conditional on σ_t, with $\delta t = \frac{T}{N}$, $N = 500$. Whereas for σ_t the Euler-Maruyama time-stepping is used.

Based on the particular algorithm used (attached in the appendix) in Matlab, the cost per sample is roughly 25% higher for the mixed method compared to the pure MC method.

	Full MC: $M_{mc} =$			MC+A: $M_{mixed} =$		
	20,000	40,000	80,000	20,000	40,000	80,000
Mean $P = \frac{1}{M}\sum P_i$	5.449	5.600	5.580	5.562	5.552	5.570
SD $= \sqrt{\frac{1}{M}\sum (P_i - P)^2}$	0.070	0.050	0.035	0.017	0.012	0.009

Table 1.1: The standard deviation of the mixed method is approximately 4 times lower

We see that for both of the method, the standard deviation decreases by a factor of $\sqrt{2}$ when the number of samples are doubled. Also across different sample size, we observe a standard deviation savings of about 4 times, translating into more than 95% of variance reduction. However the variance reduction is not consistent across the range of ρ. It has been observed that as $\rho \to 0$, the variance reduction gets more significant. The table below shows the reduction in standard deviation across different values of ρ, with 80,000 samples and the rest of the parameters left unchanged.

ρ	-0.75	-0.5	-0.25	0	0.25	0.5	0.75
Full MC SD	0.0364	0.0358	0.0354	0.0350	0.0349	0.0337	0.0348
Mixed SD	0.0252	0.0162	0.0085	0.0016	0.0069	0.0127	0.0231
Speed-Ups	1.4	2.2	4.2	21.9	5.1	2.7	1.5

Table 1.2: The SD reduction is the most significant when $\rho = 0$

To explain this, notice that in the exponential in (1.15), $\bar{\sigma}^2 = \sum v_i/N$ should be very

close to θ for each simulation due to its mean-reverting property. Hence the main source of stochasticity comes from the discrete stochastic integral $\frac{\rho}{T}\sum \sigma_i \delta \tilde{W}_i^2$ which is part of q. As $\rho \to 0$, we also have $q \to 0$ in (1.15). In which case there will be little variance among different σ_t paths. Since we are using the Black-Scholes formula there will be no variance coming from the $\tilde{W}_t^1$. Hence overall we get such a significant variance reduction.

Next we also investigate the relationship between the speed-ups and different parameters including the long term average variance θ, mean-reverting coefficient k, and the volatility of volatility λ. The default parameters are: $S_0 = 100$, $K = 90$, $r = 0.05$, $\sigma_0 = 0.4$, $k = 5$, $T = 0.5$, $\rho = -0.25$, with 80,000 samples. The value of λ is set to be low at 0.075. This is to ensure that the variance will not go negative. **Note:** In the continuous case, as long as we have $2k\theta > \lambda^2$ the process v_t will never hit 0. However in the discrete case there is still a chance, albeit extremely small, for the process to go negative. For the purpose of this paper, we do not explicitly bound the process v_t from below.

θ	0.15^2	0.25^2	0.75^2	1.00^2
FMC Price	0.550	2.211	14.050	20.325
FMC SD	0.0070	0.0178	0.0649	0.0818
FMC SD/Price	1.27%	0.81%	0.46%	0.40%
Mixed SD	0.548	2.234	14.082	20.251
Mixed SD	0.0013	0.0038	0.0149	0.0191
Mixed SD/Price	0.25%	0.17%	0.11%	0.09%
Speed-Ups	5.2	4.8	4.4	4.3

Table 1.3: The speed-up decays slowly as the average volatility increases

Again this is because q is approximately distributed as $\mathcal{N}(\frac{\rho^2\bar{\sigma}^2}{2}, \frac{\rho^2\bar{\sigma}^2}{T})$. As $\theta \to 0$, q, which is the main source of stochasticity in (1.15) will behave almost like a constant. This explains the more significant speed-ups at low average volatility. Between 10% to 15% volatility which is the range for FX options generally, the method works quite well with 5 to 7 times standard deviation reduction.

Next we investigate the effect of the speed-up with a range of mean-reversion coefficient k and the volatility of volatility λ. Both of the parameters will decide how much the v_t paths will resemble the constant function $v_t = \theta$. To study the combine effect of both, we study the effects by changing the ratio between k and λ^2. Again to ensure that the process v_t stays positive, we need $2k\theta > \lambda^2 \Rightarrow \frac{k}{\lambda^2} > \frac{1}{2\times 0.4^2} = 3.125$. In this part of the experiment, we modify k and keep λ constant to achieve the desired ratio and vice versa. Again we keep all the other parameters unchanged, and the sample size remains at 80,000 too.

k	5	5	5	5	10	50	200
λ	0.650	0.500	0.300	0.200	0.200	0.200	0.200
$k/\lambda^2 \sim$	12	20	56	125	250	1250	6000
FMC Price	5.464	5.527	5.524	5.580	5.574	5.537	5.542
FMC SD	0.0364	0.0361	0.0354	0.0350	0.0352	0.0349	0.0340
FMC Price/SD	0.67%	0.65%	0.64%	0.63%	0.63%	0.63%	0.61%
Mixed Price	5.544	5.564	5.579	5.570	5.550	5.539	5.546
Mixed SD	0.0118	0.0108	0.0093	0.0090	0.0079	0.0071	0.0070
Mixed SD/Price	0.21%	0.19%	0.17%	0.16%	0.14%	0.13%	0.12%
Speed-Ups	3.1	3.4	3.8	3.9	4.4	4.9	5.0

Table 1.4: The speed-ups improves with the ratio of k/λ^2 and levels off around 5

From the results, it can be seen that the speed-up improves with the k/λ^2 ratio, but level off at 5. The improvement is expected as high value of k will cause the process v_t to return to the mean level θ very quickly; and lower values of λ would also reduce the effect of the Brownian increment on the process v_t, making it harder for it to deviate away from θ. As a result we have a fairly constant volatility process for the Heston model, resembling the Black-Scholes model. Hence using the Black-Scholes analytical formula should give us better results. However against our intuition the improvement levels off for very high value of k/λ^2. This is because we still have the variance contributed by q which contains the discrete stochastic integral.

Furthermore, for an extremely high value of k, the simulation will be unstable. This is because we are using the Euler-Maruyama time-stepping scheme for v_i. A very high value of k would cause the discrete correction to overshoot. A good solution to this would be to using an implicit time stepping scheme for v_i as the following

$$v_{i+1} - v_i = k(\theta - v_{i+1})\delta t + \lambda\sqrt{v_i}\delta W_i$$

The coefficient of the Brownian increment is chosen to be at the start of δW_i so that this expression converges to the right stochastic integral. Solving the equation for v_{i+1} to finally obtain

$$v_{i+1} = \frac{v_i + k\theta\delta t + \lambda\sqrt{v_i}\delta W_i^2}{1 + k\delta t}$$

which is also 'explicit'. As $k \to \infty$, v_{i+1} in the expression above will converge nicely to

θ. Since this is not the focus of this paper, we stick to the explicit method for simplicity.

Chapter 2

Multi-Dimensional Systems

There are cases where we may have to work with multi-dimensional systems. For example, a stochastic volatility model with multiple stochastic processes for the volatility, or a stochastic volatility model with stochastic interest rate processes. For example, Grasselli et al ([3]) introduced the multi-Heston model which is able to reproduce consistently the usual multi-dimensional FX vanilla markets. In the situation where we model the currency exchange rate, we may even have a 4 dimensional system like this following example:

$$dS = S_t \left[(r_t^d - r_t^f)dt + \sigma_t dW_t^1 \right] \tag{2.1}$$

$$dv_t = k(\theta - v_t)dt + \lambda\sqrt{v_t}dW_t^2 \qquad v_t = \sigma_t^2 \tag{2.2}$$

$$dr_t^d = l^d(\delta^d - r_t^d)dt + \omega^d\sqrt{r_t^d}dW_t^3 \tag{2.3}$$

$$dr_t^f = l^f(\delta^f - r_t^f)dt + \omega^f\sqrt{r_t^f}dW_t^4 \tag{2.4}$$

where the 4-dimensional Brownian process is described by the correlation matrix Σ. Some practitioners have expressed interests in looking for ways to price barrier options under this models efficiently.

Of course we can still try to write down the PDE and solve it but this is again prohibitively expensive given the high dimensionality. Monte-Carlo may be faster but we think that we can do better by adapting the methods developed by ([10]), as introduced in the previous chapter.

2.1 Problem Formulation

To illustrate our method, we choose to solve the problem above, which often is used to price a long-dated currency option. The r_d and r_f stand for the domestic interest rate and

interest rate of the foreign currency respectively. One can think of the foreign interest rate as the dividend yield which is stochastic. The 4-dimensional Brownian process is described by the correlation matrix Σ. Hence again we want to simplify the system by decoupling the Brownian motion. In other words, we want to rewrite $(W_t^1, W_t^2, W_t^3, W_t^4)$ as a linear combination of independent Brownian motions $(\tilde{W}_t^1, \tilde{W}_t^2, \tilde{W}_t^3, \tilde{W}_t^4)$ or vice versa.

2.1.1 Decomposing Brownian Motion

To decompose the Brownian motion, we find the coefficients a_{ij} such that $W_i = \sum_{j=1}^{3} a_{ij}\tilde{W}_t^j$. In matrix notation, we want to find matrix in $\mathbb{R}^3$, $A = a_{ij}$ such that

$$\begin{bmatrix} W_t^1 \\ W_t^2 \\ W_t^3 \\ W_t^4 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} \tilde{W}_t^1 \\ \tilde{W}_t^2 \\ \tilde{W}_t^3 \\ \tilde{W}_t^4 \end{bmatrix}$$

with the constraints $\mathbb{E}[\mathbf{W_t}\mathbf{W_t}^\top] = \mathbf{\Sigma}$. Each the diagonal entries of Σ is equal to 1 and the off-diagonal entries $(\Sigma)_{ij} = \rho_{ij}$ where $\rho_{ij} = \rho_{ji}$. Due to the symmetry of Σ, there are 16 unknowns and 10 equations, giving us flexibilities with *Degree of Freedom* equals to $16 - 10 = 6$. For convenience sake, we set all the entries in $\{a_{ij} : i > j\}$ to be 0, in other words, we find A such that it is an upper triangular matrix. Given that the correlation matrix Σ is Symetric-Positive-Definite, we can always apply *Cholesky Decomposition* to obtain the matrix A. The related 10 equations are:

$$\mathbb{E}[(W_t^i)^2] = t \qquad \Rightarrow \sum_{i \le j} a_{ij}^2 = 1 \qquad \forall i = 1..4 \tag{2.5}$$

$$\mathbb{E}[W_t^i W_t^j] = \rho_{ij} t \qquad \Rightarrow \sum_{k=j}^{4} a_{ik} a_{jk} = \rho_{ij} \qquad \forall i < j \tag{2.6}$$

Solving the system above gives us a rather complicated expression if written out explicitly. However the idea is to solve row-by-row from the bottom row, and then term-by-term from the right, in the following order: $(4,4) \to (3,4) \to (3,3) \to (2,4) \to ... \to (1,2) \to (1,1)$. The algorithm and formulae are attached as appendix.

Note: One can also use the Cholesky Decomposition directly, provided that the algorithm starts from the bottom-right corner by setting $a_{44} = 1$, just to keep the notation

consistent with the rest of the paper. However, the computation time involved has doubled when matrix algebra is involved for the particular program and machine used in this paper. Hence throughout all the implementations, matrix operations are avoided if possible.

The objective now is to , through this decomposition, find the theoretical price of an European option with payoff $G(S_T)$ at maturity T.

Similar to the 2-D case, we can simulate the paths of v_t, r_t^d and r_t^f, and then derive conditional distribution of S_t and hence the conditional Black-Scholes PDE. We start with

$$dS_t = S_t(\mu_t dt + a_{11}\sigma_t d\tilde{W}_t^1) \tag{2.7}$$

which gives us the conditional solution

$$S_t = S_0 \exp\left(\int_0^t \mu_s - \frac{1}{2}a_{11}^2\sigma_s^2\, ds + \int_0^t a_{11}\sigma_s d\tilde{W}_t^1\right) \tag{2.8}$$

Compare this with the true solution

$$S_t = S_0 \exp\left(\int_0^t r_s^d - r_s^f - \frac{1}{2}\sigma_s^2\, ds + \sum_{j=1}^{4} a_{1j}\int_0^t \sigma_s d\tilde{W}_t^j\right) \tag{2.9}$$

We know that μ_t must satisfy

$$\int_0^t \mu_s\, ds = \int_0^t r_s^d - r_s^f + \frac{\sigma_s^2}{2}(a_{11}^2 - 1)\, ds + \sum_{j=2}^{4} a_{1j}\int_0^t \sigma_s d\tilde{W}_t^j \tag{2.10}$$

Hence for the discrete path $\forall t \in [t_i, t_{i+1})$,

$$\mu_i = r_i^d - r_i^f + \frac{\sigma_i^2}{2}(a_{11}^2 - 1) + \sum_{j=2}^{4} a_{1j}\sigma_i \frac{\tilde{W}_{i+1}^j - \tilde{W}_i^j}{\delta t} \tag{2.11}$$

As we mentioned before, this is only heuristic and is nowhere rigorous due to the discretization of the stochastic integral to allow a simple comparison to be made. As $\delta t \to 0$ we have the following conditional PDE:

$$\partial_t u + \mu_t S_t \partial_S u + \frac{1}{2} a_{11}^2 \sigma_t^2 S_t^2 \partial_{SS} u - r_t^d u = 0 \tag{2.12}$$

$$u(S_T, T) = G(S_T) \tag{2.13}$$

And the discretized Black-Scholes PDE (explicit) is

$$\frac{u_m^{n+1} - u_m^n}{\Delta t} + \mu_n S_m \frac{u_{m+1}^{n+1} - u_{m-1}^{n+1}}{2\Delta S} + a_{11}^2 \sigma_n^2 S_m^2 \frac{u_{m+1}^{n+1} - 2u_m^{n+1} + u_{m-1}^{n+1}}{2\Delta S^2} - r_n^d u_m^{n+1} = 0 \tag{2.14}$$

$$u_m^N = G(S_m) \tag{2.15}$$

It would be confusing to write down the conditional PDE (2.12) as μ_t is not well defined because $\delta_t \to 0$. The precise representation should be as the following:

$$du = \left[\left(r_t^d + r_t^f + (a_{11}^2 - 1) \frac{\sigma_t^2}{2} \right) S_t \partial_S u + \frac{1}{2} a_{11}^2 \sigma_t^2 S_t^2 \partial_{SS} u - r_t^d u \right] dt + S_t \partial_S u \sum_{j=2}^{4} a_{1j} \sigma_t d\tilde{W}_t^j \tag{2.16}$$

To make the conditional expression of S_T comparable to the standard Black-Scholes problems with deterministic time-dependent interest rate and volatility, we write down the discrete version of (2.9) and compare with (1.15). This would then allow us to use to price a European option if the analytical expression for its price under Black-Scholes is known. Otherwise we have to price the option via finite difference method (see later). The discrete version of (2.9) is as the following:

$$S_t = S_0 \exp\left(\left(\bar{r}^d - \bar{r}^f - \frac{\bar{\sigma}^2}{2} \right) T + \sum_{j=2}^{4} a_{1j} \sum_{i=1}^{N} \sigma_i \delta \tilde{W}_i^j + a_{11} \bar{\sigma} \sqrt{T} N_{0,1} \right) \tag{2.17}$$

where $\bar{r}^d = \sum_{i=1}^N \frac{r_i^d}{N}$, $\bar{r}^f = \sum_{i=1}^N \frac{r_i^f}{N}$ and $\bar{\sigma}$ is as defined as in previous chapter. Compared with (1.15) we can obtain the required Black-Scholes parameters:

$$r = \bar{r}^d; \qquad \tilde{\sigma} = \bar{\sigma}; \qquad a = a_{11}$$
$$q = \bar{r}^f + \frac{1}{2}(1 - a_{11}^2)\bar{\sigma}^2 - \frac{1}{T}\sum_{i=1}^{N} \sigma_i \left(a_{12}\delta\tilde{W}_i^2 + a_{13}\delta\tilde{W}_i^3 + a_{14}\delta\tilde{W}_i^4 \right)$$

2.1.2 Numerical Experiments

In this subsection we look at implementing the model above, to price a simple European vanilla put, and also another Double-No-Touch (DNT) with all the other parameters remain unchanged. The barrier is (almost) a constantly monitored barrier which is active throughout the life of the option. Being the most liquid asset class, the currency market has many more sophisticated models and more exotic options have been developed as opposed to equities markets.

Vanilla Put with Mixed MC/Analytic Method

For the vanilla put, we look at the standard deviation reduction of the mixed method.

The particular model used is as the following, which is not unusual for currency pairs such as USD/JPY :

$$\begin{aligned}
dS_t &= S_t \left[(r_t^d - r_t^f)dt + \sqrt{v_t} dW_t^1 \right] && S_0 = && 77 \\
dv_t &= 15(0.12^2 - v_t)dt + 0.15\sqrt{v_t} dW_t^2 && v_0 = && 0.12^2 \\
dr_t^d &= 2(0.025 - r_t^d)dt + 0.05\sqrt{r_t^d} dW_t^3 && S_0 = && 0.05 \\
dr_t^f &= 2(0.025 - r_t^f)dt + 0.05\sqrt{r_t^f} dW_t^4 && S_0 = && 0.05
\end{aligned}$$

with the following correlation parameters: $\rho_{12} = \rho_{13} = \rho_{24} = -0.15$, $\rho_{14} = \rho_{23} = \rho_{34} = 0.15$. As for the timestepping scheme, again we use the exact solution for S_t and Euler-Maruyama for v_t, r_t^d and r_t^f across $N = 400$ steps. The option has strike $K = 75$ and will expire in 3 months, that is, $T = 0.25$.

	Full MC: $M_{mc} =$			MC+A: $M_{mixed} =$		
	20,000	40,000	80,000	20,000	40,000	80,000
Mean $P = \frac{1}{M}\sum P_i$	1.338	1.340	1.345	1.336	1.340	1.342
SD $= \sqrt{\frac{1}{M}\sum (P_i - P)^2}$	0.0139	0.0098	0.0069	0.0031	0.0022	0.0016

Table 2.1: The standard deviation is approximately 4.5 times lower for the mixed method

Based on the algorithm used (as in appendix), the running time per sample is approximately 35% higher for the mixed method. However in the table below, it can be observed that the standard deviation is reduced by approximately 4.5 times (or 20 times for variance).

Double-No-Touch with Mixed MC/PDE Method

Barrier options are very common for FX compared to equities. However, some of the models make pricing extremely expensive. The conventional basic methods such as the basic Monte Carlo and Finite Differences usually have to be improvised with tricks before they can be used.

In pricing path-dependent options, Monte Carlo faces the problems of low order of strong errors ($\sim 1/2$ for Euler-Maruyama) and often need a huge number of samples. To restore the first order convergence for path dependent options, various methods have been developed. For example, one can derive the probability of barrier crossing between two points on a discrete Brownian path, if both of the points were on the same side of the barrier. The final price can then be adjusted using the conditional probability of barrier crossing at each step. Alternatively one can also use the probability density function of barrier crossing to sample the minimum or maximum point between the small time interval between the two points.

As for the PDE approach, one can use a barrier correction method as what was done in ([1]) by Broadie and Glasserman. Howison and Steinberg also did an analysis on correcting the bias due to discrete sampling via a matched asymtoptic approach ([9]). However the PDE approach faces more problems as models get more sophiscated and complicated, which often result in higher dimensionality. Hence we need to develop ways to deal with high dimensional problems.

We will illustrate, through an example, how the mixed methods can address the cost issue of the Monte Carlo methods as well as the ability to handle high dimensional problems.

In this section we will use the 1-month Double-No-Touch (DNT) with continuous barrier

as an example. While this model traditionally is more widely used for a long term contracts, the short maturity has been chosen to ease the overall computational effort involved. It will be explained later how the result is expected to generalize to long-term contract easily.

A DNT is an option which pays 1 dollar upon maturity if the stock price $S_{t_i} \in (S_{min}, S_{max})$, $\forall i \in \{1..N\}$. Else there will be no payout. In this case we choose $(S_{min}, S_{max}) = (73, 78)$, with maturity 1/12 which is approximately a month. For the Monte Carlo part, 400 timesteps will be used, whereas for the mixed method, there will be 21 spatial mesh points and 401 time mesh points. Since we are focusing on the weak error and variance in this case, the exact solution for the stock path S_t will be used to reduce the strong error for the Monte Carlo. As Euler-Maruyama timestepping is still used for σ_t, r_t^d, and r_t^f, there are still some strong errors in the system. But given the mean reverting nature of these 3 quantities, the error distribution among the samples are expected to be symmetrical and would cancel each other out. Other parameters are stated in the model below:

$$
\begin{aligned}
dS_t &= S_t\left[(r_t^d - r_t^f)dt + \sqrt{v_t}dW_t^1\right] && S_0 = && 76\\
dv_t &= 15(0.12^2 - v_t)dt + 0.15\sqrt{v_t}dW_t^2 && v_0 = && 0.12^2\\
dr_t^d &= 2(0.025 - r_t^d)dt + 0.05\sqrt{r_t^d}dW_t^3 && S_0 = && 0.025\\
dr_t^f &= 2(0.025 - r_t^f)dt + 0.05\sqrt{r_t^f}dW_t^4 && S_0 = && 0.025
\end{aligned}
$$

By implementing the schemes, we have the following results:

	Full MC: $M_{mc} =$			MC+A: $M_{mixed} =$		
	60,000	120,000	240,000	600	1200	2400
Mean $P = \frac{1}{M}\sum P_i$	0.344	0.343	0.342	0.343	0.343	0.341
SD $= \sqrt{\frac{1}{M}\sum(P_i - P)^2}$	0.0019	0.0014	0.0010	0.0020	0.0014	0.0010
Time Taken (sec)	8.0	17.3	31.2	5.3	10.3	21.5

Table 2.2: The standard deviation is 100 times lower on a per sample basis, and the total costs is approximately 40% lower

It can be observed that in this case, by switching to the mixed algorithm, we have managed to reduce the standard deviation by a huge 100 times. For the options with

tighter barriers, we actually expect an even better improvement as many of the samples end up not contributing to the final estimation for the full Monte-Carlo. Given that it is not uncommon in the FX market to write an option with less than 10% premium, this can be extremely useful in practice. This is because the Monte Carlo is essentially producing a Binomial variable. Hence the only direct factor which affects its variance should be the barrier hitting probability. Another point to note is also that, in practice this model would be used to price a contract with much longer maturity. Hence we would expect the probability of hitting the barrier to be even higher, making the variance of the pure Monte Carlo even larger. If in the long term contract, a wider pair of barriers are used, we should expect the performance to generalize to the case where the corresponding survival zone of the driving Brownian process W_t^1, is scaled such that the square of it is proportionate the maturity. This is due to the scaling invariance of the Brownian path , which states the fact that $W'_t = \frac{1}{c}W_{c^2t}$, is also a Brownian motion $\quad \forall c > 0$.

However one caveat is that, finite difference implementation is more expensive than pure Monte Carlo in terms of costs per sample. In our implementation, the cost per sample for the mixed algorithm is approximately 50 - 60 times more expensive than the full Monte-Carlo algorithm. Hence the effective cost reduction to achieve the same variance is about 40% - 50%. In theory we do not expect this, given that only 21 spatial mesh points are used for the pricing. Given the limited resources available, we decided to prioritize the rest of the thesis over this.

Note that these are based on the codes written as in the appendix. Different code structures, different machines will produce different results generally in terms of computational costs.

Notes: The readers should also be reminded again that, in this case, the exact solution for the stock price path has been used and essentially the strong error of the path has been greatly reduced. Or else this is actually expected to contribute much of the error to a very path dependent option like DNT under the pure Monte Carlo pricer.

There are various methods that allow us to reduce the variance for the Monte Carlo and PDE approaches. A good example is the Multi-Level method, which has been widely used to speed up Monte Carlo simulations.

One may argue that using the Multi-level technique does not make the mixed method better, because the same technique can be applied to the full Monte Carlo case too. However, for the mixed method, we are able to reduce the number of time steps, as well as reducing the number of spatial steps. If we compare the costs to simulate a correction of level l and a correction of level $l-1$ in Monte Carlo, the cost will decrease by 3/4, assuming that

the number of timesteps for level $l-1$ is only $1/4$ as many as that of level l. For the mixed method however, if we keep $\Delta t \propto \Delta S^2$, the cost should in fact decrease by $7/8$ as the number of time mesh points will be reduce by 4 times and the number of spatial mesh points will be reduced by half.

In this next chapter we introduce the Multi-Level method for the mixed algorithm.

Chapter 3

Multi-Level Simulation

The Multi-Level Monte Carlo method was first introduced by ([4]) in 2008 as an efficient method to simulate path-dependent payoffs valued on the solution of SDEs.

Up until now, we have been implementing the simplest forms of simulations. To calculate the expectation of the payoff $\mathbb{E}(P)$, we simulate a lot of sample paths S_t^i according to the model specification, which gives us $\hat{P}_i = f(S_t^i)$ where $f(.)$ is the payoff function. And then by the *Law of Large Number* we know that the average of these samples $\bar{P} = \sum \hat{P}_i/M$ converges to $\mathbb{E}(P)$ where M is the number of samples simulated.

Hence we have the mean square error as the following, which can further be decomposed into two components.

$$\begin{aligned}
\mathbb{E}\left[(\bar{P} - \mathbb{E}\left[P\right])^2\right] &= \mathbb{E}\left[(\bar{P} - \mathbb{E}[\hat{P}] + \mathbb{E}[\hat{P}] - \mathbb{E}[P])^2\right] \\
&= \mathbb{E}\left[(\bar{P} - \mathbb{E}[\hat{P}])^2\right] + \Big(\mathbb{E}[\hat{P}] - \mathbb{E}[P]\Big)^2 \\
&= \mathbb{V}[\hat{P}]/M + \Big(\mathbb{E}[\hat{P}] - \mathbb{E}[P]\Big)^2
\end{aligned}$$

The first term $\mathbb{V}[\hat{P}]/M$ is due to the variance from the simulation samples which scales like M^{-1}, whereas the second term $\Big(\mathbb{E}[\hat{P}] - \mathbb{E}[P]\Big)^2$ is due to the bias of the estimator - in our case it is because finite number of timesteps is used. Hence if we need the root mean square (or RMS) error to be of order $O(\epsilon)$, we need both terms to be of $O(\epsilon^2)$. This would mean that $M \sim O(\epsilon^{-2})$. Also, for the second terms, we know that the standard implicit or explicit Euler scheme would have errors scaled like $O(\Delta t, \Delta x^2)$. Hence we also need $\Delta t \sim O(\epsilon)$ and $\Delta x \sim O(\epsilon^{1/2})$. This would mean that the costs to achieve such a RMS error of $O(\epsilon)$ to be $O(\epsilon^{-7/2}) \propto M\Delta t^{-1}\Delta x^{-1}$. The aim of implementing the multi-level Monte Carlo is to reduce the complexities to $O(\epsilon^{-2})$ ([6]).

Consider multiple sets of simulations with different timesteps $\Delta t = 2^{-l}T, \quad l = 0, 1..., L$

and the respective payoff $\hat{P}_l$. We can express the estimation based on the number of mesh points in level L as the following:

$$\mathbb{E}[\hat{P}_L] = \mathbb{E}[\hat{P}_0] + \sum_{i=1}^{L} \mathbb{E}[\hat{P}_l - \hat{P}_{l-1}]$$

The terms $\mathbb{E}[\hat{P}_l - \hat{P}_{l-1}]$ are the correction terms and will be approximated by $\hat{Y}_l = M_l^{-1} \sum_{i=1}^{M_l} (\hat{P}_l^{(i)} - \hat{P}_{l-1}^{(i)})$. Due to the linearity of the operation $\mathbb{E}[.]$ the expected value stays the same. The idea is that, if we estimate $\hat{P}_L$ and $\hat{P}_{l-1}$ using the same Brownian path to obtain $\hat{Y}_l$, we can actually reduce the variance of estimator for a fixed computational cost ([5]).

This cost savings is mainly due to the decaying variance of the correction estimators $\hat{Y}_l$ as we move from $l = 0$ to $l = L$. Hence a lot of samples will be required to compute the based estimate for $\mathbb{E}[\hat{P}_0]$ which is cheap to do, and then only relatively few samples are required to compute the high-level corrections $\hat{Y}_l$ which are more expensive.

The final variance of the corrected estimator $\sum \hat{Y}_l$ is $\sum N_l^{-1} V_l$ where V_l is the variance for each sample of Y_l, which is a function of the number of samples chosen at each level. To choose the optimal number of samples at each level, we do the following Lagrangian optimization exercise.

Define the variance function as the objective function and the constraint.

$$\text{Total Variance}, V = \sum_{i=0}^{L} \frac{V_l}{N_l}$$
$$\text{Total Costs}, C = \sum_{i=0}^{L} N_l C_l$$

We want to minimize the variance V while keeping the costs C constant at $\hat{C}$. Define the following Lagrangian

$$L = V - \lambda(C - \hat{C})$$

Solving this optimization problem and we find that, optimally, the number of samples at each level should be proportionate to the quantities $\sqrt{V_l/C_l}$ where C_l is the cost per sample at level l.

One can easily see that, if the variance V_l decays faster than the cost C_l increases with level, then there will be a lot more samples simulated at level 0. Hence the cost savings

relative to standard Monte Carlo will be approximately C_0/C_l. On the other hand, if the variance V_l decays slower than the cost C_l increases, then it will be optimal to simulate more samples at the finest level L. In this case the cost savings will be approximately V_L/V_0 ([6]). In our exercise it is hoped that the former case will apply and we can end up simulating most of the samples on a fairly coarse grid and slowly adjust the estimate.

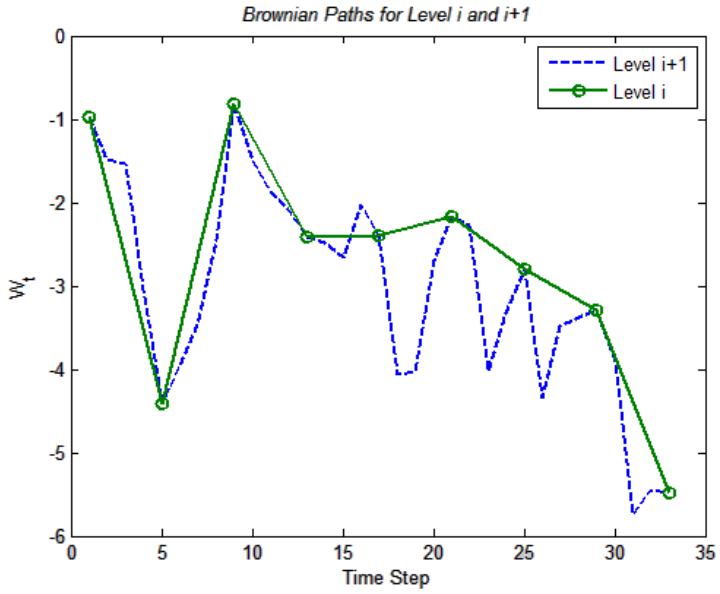
3.1 Numerical Experiments

We now implement the multi-level scheme to price the options. Previously we used 21 mesh points in the S-direction (with step size 0.2) and 400 mesh points in the t-direction. For the multi-level mixed method, we split the simulation into 3 levels (that is, $L = 2$) , keeping $\Delta t \propto \Delta x^2$ with the followings:

Level	N=Num(Mesh - Time)	J=Num(Mesh - Spatial)
$l = 0$	$25 \times 4^0 = 25$	$5 \times 2^0 + 1 = 6$
$l = 1$	$25 \times 4^1 = 100$	$5 \times 2^1 + 1 = 11$
$l = 2$	$25 \times 4^2 = 400$	$5 \times 2^2 + 1 = 21$

Table 3.1: Number of mesh points used at each level

To simulate the correction terms going across from level i to level $i + 1$, Y_{i+1}, we first simulate the Brownian path according to the mesh defined by level $i + 1$. And then at the Brownian path for the coarser grid at level i is constructed from it by extracting only the $(4j + 1)$'th points, $j = 0...N/4$ like the following example.



The constructed Brownian Path is then used to price the option, along with the coarser spatial grid. A pilot run is first implemeted to find out the variance of Y_l, which then can help us decide on the optimal number of samples to use at each level.

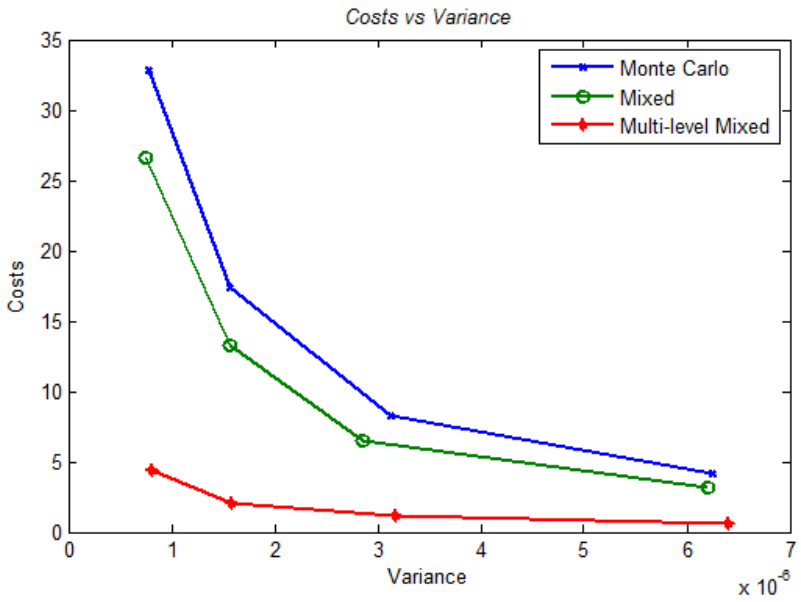
	Estimate	SD	Costs	$\sqrt{V_l/C_l}$	Optimal N_l Ratio
Y_0	0.3337	0.00174	0.36	0.002900	284
Y_1	0.0067	0.00023	2.04	0.000161	16
Y_2	0.0016	0.00012	7.95	0.000043	4
Y_3	0.0004	0.00006	40.25	0.000010	1

Table 3.2: Results from the pilot run shows that the correction decreases by 3/4 as we refine the time mesh by 4 times and spatial mesh by 2 times

The results shows that as we move from one level of correction to another, the standard deviation decreases by half (or the variance decreases by 4 times) while the costs increases by approximately 4 times. Hence in this exercise the majority of the costs come from simulating Y_0. Each level of the correction costs the same as the reducing number of samples required and the increasing costs cancel each other.

Now if we re-run the simulation, and compare it with the standard Monte Carlo and standard mixed methods, we see the following variance reduction. Note: To allow a consistent comparison, Y_3 is omitted. This is because for both the standard Monte Carlo and mixed methods, the barrier conditions were only applied 400 times. The Y_3 correction would

apply the barrier knock-out conditions 1600 times. The graph below shows the relationships between the variance and the costs required for each method.



To further describe the relationship between the costs and the variance, we fit the data points into the following inverse relationship to model the costs.

$$C = \frac{100,000c}{V}$$

Where C and V are defined as before. The coefficient c is the costs coefficient, and its value is 2.65, 1.90, and 0.42 for the standard Monte Carlo, standard mixed, and the multi-level mixed methods respectively. Hence compared to the standard Monte Carlo method, the mixed method would only cost 2/3 as much, and for the multilevel method, 1/6.

Over here we are just comparing the costs with the variance. In fact a more relevant relationship should be one between the costs and the root-mean-square error, which take the bias into consideration. In which case some techniques have to be used (as we previously mentioned) and that will introduce additional costs.

Chapter 4

Final Words

Overall we have demonstrated that the mixed method can be a good candidate as a practical, efficient way to price barrier options.

We believe that the work can be improvised and extended further with the following suggestions:

- In previous chapters we briefly discussed about how the multi-level method would benefit the mixed methods more than the Monte Carlo method. However the modification was only applied onto the mixed method. It would be great to see the empirical comparison to verify our reasonings.
- We previously discussed that we did not expect the MC/Finite Difference methods to be so much more expensive than the MC/Analytic or pure MC methods. This is because the explicit Euler stepping has been used and we only have at most 21 spatial mesh points at one time. A possible improvement would be to find out the source of the costs and perhaps we can see a more significant cost-savings with the mixed method.
- The investigation uses very specific examples. So one may wonder whether this results generalizes to other options. In practice the FX model with stochastic volatility and interest rates is used for options term maturity rather than 1-month as in the example used in this paper. While we previously argued that the result should generalize to other cases easily due to the invariant scaling properties of the Brownian path, it would be good to have empiral verification.
- This paper mainly focuses on investigation and application of the methods developed by ([10]) and ([4]). While some related analysis have been done in ([11]) and ([6]) in terms of convergence and stability, it would be much better to have the method review throughout the paper in terms of its convergence, stability and consistency further.

Chapter 5

Appendix

The followings are the algorithms used in this paper

Algo 1: Mixed MC-Analytical - Heston Model

Define model parameters $S_0, K, r, \sigma_0, v_0, T, \theta, \lambda, \rho$

Define N(# of timeteps), M(# of paths)

$\tilde{a} \leftarrow \sqrt{1-\rho^2}$

for $j = 1..M$

$V_T \leftarrow 0$

for $i = 1..N$

$R_i \leftarrow N_{0,1}$

$v_{i+1} = v_i + k(\theta - v_i)\delta t + R_i \lambda \sqrt{v_i \delta t}$

end **for**

$\bar{\sigma}^2 \leftarrow \frac{1}{N} \sum v_i$

$q \leftarrow \frac{1}{2}\rho^2 \bar{\sigma}^2 - \frac{\rho}{T} \sum \sigma_i R_i \sqrt{\delta t}$

$d1 \leftarrow [\log(S_0/K) + (r - q + a^2\bar{\sigma}^2)T]/\tilde{a}\bar{\sigma}\sqrt{T}$

$d2 \leftarrow d1 - \tilde{a}\bar{\sigma}\sqrt{T}$

$P \leftarrow Ke^{-rT}N(-d2) - S_0 e^{-qT} N(-d1)$

$V_T = V_T + P/M$

end **for**

Price $\leftarrow V_T$

Algo 2: Algorithm to find Matrix A

$a_{i4} \leftarrow \rho_{i4} \quad i = 1..4$

$a_{33} \leftarrow \sqrt{1 - a_{34}^2}$

$a_{23} \leftarrow (\rho_{23} - a_{24}a_{34})/a_{33}$

$a_{22} \leftarrow \sqrt{1 - a_{23}^2 - a_{24}^2}$

$a_{13} \leftarrow (\rho_{13} - a_{14}a_{24})/a_{23}$

$a_{12} \leftarrow (\rho_{12} - a_{14}a_{24} - a_{13}a_{23})/a_{22}$

$a_{11} \leftarrow \sqrt{1 - a_{12}^2 - a_{13}^2 - a_{14}^2}$

Algo 3: Mixed MC-Analytical - FX Model

Define the following model parameters

S_0, K, r, T, Σ

$\sigma_0, v_0, \theta, \lambda$

$r_0^d, l^d, \delta^d, \omega^d$

$r_0^f, l^f, \delta^f, \omega^f$

Define the matrix A

$\tilde{a} \leftarrow a_{11}$

Define N(# of timeteps), M(# of paths)

for $j = 1..M$

$V_T \leftarrow 0$

for $i = 1..N$

$$\tilde{R}_i^k \leftarrow N_{0,1} \qquad k = 2,3,4$$
$$R_i^k \leftarrow \sum_{p=k}^{4} a_{kp}\tilde{R}_i^p \qquad k = 2,3,4$$

$$v_{i+1} = v_i + k(\theta - v_i)\delta t + R_i^2\lambda\sqrt{v_i\delta t}$$
$$r_{i+1}^d = r_i^d + l^d(\delta^d - r_i^d)\delta t + R_i^3\omega^d\sqrt{v_i\delta t}$$
$$r_{i+1}^f = r_i^f + l^f(\delta^f - r_i^f)\delta t + R_i^4\omega^f\sqrt{v_i\delta t}$$

end **for**

$$\bar{\sigma}^2 \leftarrow \frac{1}{N}\sum v_i; \quad \bar{r}^d \leftarrow \frac{1}{N}\sum r_i^d; \quad \bar{r}^f \leftarrow \frac{1}{N}\sum r_i^f$$
$$q \leftarrow \bar{r}^f\tfrac{1}{2}\bar{\sigma}^2(1-\tilde{a}^2) - \tfrac{1}{T}\sum\sigma_i(a_{12}R_i^2 + a_{13}R_i^3 + a_{14}R_i^4)\sqrt{\delta t}$$

$$d1 \leftarrow [\log(S_0/K) + (\bar{r}^d - q + a^2\bar{\sigma}^2)T]/\tilde{a}\bar{\sigma}\sqrt{T}$$
$$d2 \leftarrow d1 - \tilde{a}\bar{\sigma}\sqrt{T}$$

$$P \leftarrow Ke^{-rT}N(-d2) - S_0e^{-qT}N(-d1)$$
$$V_T = V_T + P/M$$

end **for**

Price $\leftarrow V_T$

Algo 4: Mixed MC-PDE - FX Model to Price DNT

Define the model parameters

Define the matrix A

$\tilde{a} \leftarrow a_{11}$

Define N(# of timeteps), M(# of paths), J(# of spatial steps)

$S \leftarrow (S_{min} = S_0, S_1, ..., S_J = S_{max})^\top$

for $j = 1..M$

$V_T \leftarrow 0$

for $i = 1..N$

$\tilde{R}_i^k \leftarrow N_{0,1} \qquad k = 2, 3, 4$
$R_i^k \leftarrow \sum_{p=k}^{4} a_{kp} \tilde{R}_i^p \qquad k = 2, 3, 4$

$v_{i+1} = v_i + k(\theta - v_i)\delta t + R_i^2 \lambda \sqrt{v_i \delta t}$
$r_{i+1}^d = r_i^d + l^d(\delta^d - r_i^d)\delta t + R_i^3 \omega^d \sqrt{v_i \delta t}$
$r_{i+1}^f = r_i^f + l^f(\delta^f - r_i^f)\delta t + R_i^4 \omega^f \sqrt{v_i \delta t}$
$\mu_{i+1} = r_i^d - r_i^f + v_i(\tilde{a}^2 - 1)/2 + \sqrt{v_i/\delta t} \sum_{p=2}^{4} a_{1p} \tilde{R}_i^p$

end **for**

$u \leftarrow (0, 1, ..., 1, 0)^\top \in \mathbb{R}^{J+1}$

for $j = 1..N$

$A \leftarrow (v_{N-n+1} \tilde{a}^2 S^2 / 2\Delta S^2)\delta t$
$B \leftarrow (\mu_{N-n+1} S / 2\Delta S)\delta t$

$x \leftarrow A - B$
$y \leftarrow 1 - 2A - r_{N-n+1}^d \delta t$
$z \leftarrow A + B$

$Z \leftarrow \text{tridiag}(x, y, z, -1, 0, 1)$
$u \leftarrow Zu$

end **for**

$V_T = V_T + u[1 + (S_0 - S_{min})/\Delta S]/M$

end **for**

Price $\leftarrow V_T$

Algo 5a: Multilevel MC-PDE - FX Model to Price DNT
Correction at Level l, Y_l

Define the parameters, Matrix A, M

$J \leftarrow 2^l(S_{max} - S_{min})$
$N \leftarrow J^2$

$$\begin{aligned} &\tilde{R}_i^k \leftarrow N_{0,1} && k = 2,3,4 \\ &W_i^{k,fine} \leftarrow (\textstyle\sum_{p=k}^{4} a_{kp}\tilde{R}_i^p)\delta t && k = 2,3,4 \\ &W_i^{k,coarse} \leftarrow \textstyle\sum_{j=1}^{4} W_{4i+j}^{k,fine} && k = 2,3,4 \end{aligned}$$

1. Run **Algo4** with $W_i^{k,fine}$ as the random input with $J_l \leftarrow 2^l(S_{max} - S_{min})$, $N_l \leftarrow J_l^2$
2. Run **Algo4** with $W_i^{k,coarse}$ as the random input with $J_{l-1} \leftarrow 2^{l-1}(S_{max} - S_{min})$, $N_{l-1} \leftarrow J_{l-1}^2$
3. $Y_l \leftarrow V_l - V_{l-1}$, the difference of the results from 1 and 2 respectively.

Algo 5b: Multilevel MC-PDE - FX Model to Price DNT
Combining Y_l's

Define the parameters, Matrix A, M
Define the finest level L

1. Run **Algo4**, with $J_0 \leftarrow 2^0(S_{max} - S_{min})$, $N_l \leftarrow J_0^2$ to obtain Y_0
2. Run **Algo5a** with $l = 1, 2, ..., L$ to obtain $Y_1, Y_2, ..., Y_L$
3. Price $\leftarrow \sum_{i=0}^{L} Y_i$

Bibliography

[1] M. Broadie, P. Glasserman, 1997, *A Continuity Correction for Discrete Barrier Options*

[2] J.C. Butcher, 2008, *Numerical Methods for Ordinary Differential Equations*, Second Edition, John Wiley and Sons Ltd

[3] A.D. Col, A. Gnoatto, M. Grasselli, 2013, *Smiles all around: FX Joint Calibration in A Multi-Heston Model*

[4] M. B. Giles, 2008, *Multi-Level Monte Carlo Path Simulation*

[5] M. B. Giles, 2013, *Multilevel MC Approach*, Lecture delivered for the course Numerical Methods - Monte Carlo, University of Oxford, Unpublished.

[6] M. B. Giles, C. Reisinger, 2012, *Stochastic Finite Differences and Multilevel Monte Carlo for a Class of SPDEs in Finance*

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

[8] D. J. Highmam, 2000, *A-Stability and Stochastic Mean-Square Stability*

[9] S. Howison, M. Steinberg, 2007, *A Matched Asymtoptic Expansions Approach to Continuity Corrections for Discretely Sampled Options*, p.g. 63 - 89

[10] T. Lipp, G. Loeper, O. Pironneau, 2012, *Mixing Monte-Carlo and Partial Differential Equations for Pricing Options*

[11] G. Loeper, O. Pironneau, 2008, *A Mixed PDE/Monte-Carlo Method for Stochastic Volatility Models*

[12] J. Norgaard, 2011, *Pricing and Hedging of FX Plain Vanilla Options*, Aarhus School of Business

[13] Johnson, 2008, *Von-Neumann Stability Analysis*, Lecture Notes, *http://www.maths.manchester.ac.uk/∼pjohnson/CFD/Lectures/von-neumann-analysis.pdf*