IEOR E4603: Monte-Carlo Simulation
Columbia University

Simulating Stochastic Differential Equations

In these lecture notes we discuss the simulation of stochastic differential equations (SDEs), focusing mainly on the Euler scheme and some simple improvements to it. We discuss the concepts of weak and strong convergence and note that in financial applications it is typically only weak convergence that is required. We also briefly discuss variance reduction for SDE's, the simulation of SDE's for jump-diffusion processes, and the optimal allocation of a fixed computational budget to minimize the mean-squared error of discretized SDE estimators. Most of the development in these notes follows Chapter 6 of Glasserman (2004) and this reference can be consulted for further details. Finally, in Appendix 6 we present a brief overview of *multilevel* Monte-Carlo, a new and recent technique that also focuses on the optimal allocation of computational resources.

1 The Euler Scheme for Diffusions

Suppose we have an SDE of the form

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t \tag{1}$$

and that we wish to simulate values of X_T without knowing[1] its distribution. In this event we can simulate a *discretized* version of the SDE. In particular, we simulate a discretized process, $\{\widehat{X}_0, \widehat{X}_h, \widehat{X}_{2h}, \ \ldots \ , \widehat{X}_{mh}\}$, where m is the number of time steps, h is a constant and $m = \lfloor T/h \rfloor$. The smaller the value of h, the closer our discretized path will be to the continuous-time path of (1) that we wish to simulate. Of course this will be at the expense of greater computational effort. While there are a number of discretization *schemes* available, the simplest and most common scheme is the *Euler* scheme. This scheme is intuitive, easy to implement and satisfies

$$\widehat{X}_{kh} = \widehat{X}_{(k-1)h} + \mu\left((k-1)h, \widehat{X}_{(k-1)h}\right)h + \sigma\left((k-1)h, \widehat{X}_{(k-1)h}\right)\sqrt{h}Z_k \tag{2}$$

where the Z_k's are IID $N(0,1)$. If we want to estimate $\theta := \mathbb{E}[f(X_T)]$ using the Euler scheme, then for a fixed number of paths, n, and discretization interval, h, we have the following algorithm.

Using the Euler Scheme to Estimate $\theta = \mathbb{E}[f(X_T)]$ When X_t Follows a 1-Dimensional SDE

```
for $j = 1$ to $n$
    $t = 0$; $\widehat{X} = X_0$
    for $k = 1$ to $\lfloor T/h \rfloor =: m$
        generate $Z \sim N(0,1)$
        set $\widehat{X} = \widehat{X} + \mu(t, \widehat{X})h + \sigma(t, \widehat{X})\sqrt{h}\ Z$
        set $t = t + h$
    end for
    set $f_j = f(\widehat{X})$
end for
set $\widehat{\theta}_n = (f_1 + \ldots + f_n))/n$
set $\widehat{\sigma}_n^2 = \sum_{j=1}^{n}(f_j - \widehat{\theta}_n)^2/(n-1)$
set approx. $100(1-\alpha)$ % CI $= \widehat{\theta}_n \pm z_{1-\alpha/2}\frac{\widehat{\sigma}_n}{\sqrt{n}}$
```

[1] This could be due to the fact that we cannot solve (1) to obtain an explicit solution for X_T, or because we simply cannot determine the distribution of X_T even though we do know how to solve (1).

Remark 1 *Observe that even though we only care about X_T, we still need to generate intermediate values, X_{ih}, if we are to minimize the* discretization error. *Because of this discretization error, $\widehat{\theta}_n$ is no longer an unbiased estimator of θ.*

Remark 2 *If we wished to estimate $\theta = \mathbb{E}[f(X_{t_1}, \ldots, X_{t_p})]$ then in general we would need to keep track of $(X_{t_1}, \ldots, X_{t_p})$ inside the inner for-loop of the algorithm.*

Exercise 1 *Can you think of a derivative where the payoff depends on $(X_{t_1}, \ldots, X_{t_p})$, but where it would* not *be necessary to keep track of $(X_{t_1}, \ldots, X_{t_p})$ on each sample path?*

1.1 The Euler Scheme for Multidimensional Diffusions

In the multidimensional case, $\mathbf{X}_t \in \mathbb{R}^d$, $\mathbf{W}_t \in \mathbb{R}^p$ and $\boldsymbol{\mu}(t, \mathbf{X}_t) \in \mathbb{R}^d$ in (1) are now vectors, and $\boldsymbol{\sigma}(t, \mathbf{X_t}) \in \mathbb{R}^{d \times p}$ is a matrix. This situation arises when we have a *series* of SDE's in our model. This could occur in a number of financial engineering contexts. Some examples include:

(1) Modeling the evolution of multiple stocks. This might be necessary if we are trying to price derivatives whose values depend on multiple stocks or state variables, or if we are studying the properties of some portfolio strategy with multiple assets.

(2) Modeling the evolution of a single stock where we assume that the volatility of the stock is itself stochastic. Such a model is termed a *stochastic volatility* model.

(3) Modeling the evolution of interest rates. For example, if we assume that the short rate, r_t, is driven by a number of factors which themselves are stochastic and satisfy SDE's, then simulating r_t amounts to simulating the SDE's that drive the factors. Examples include the multi-factor Gaussian and CIR models. Such models also occur in HJM and LIBOR market models.

In all of these cases, whether or not we will have to simulate the SDE's will depend on the model in question and on the particular quantity that we wish to compute. If we do need to discretize the SDE's and simulate their discretized versions, then it is very straightforward. If there are p correlated Brownian motions, $\mathbf{W}_t$, driving the SDE's, then at each time step, t_i, we must generate p IID $N(0,1)$ random variables. We would then use the Cholesky Decomposition to generate $X_{t_{i+1}}$. This is exactly analogous to our method of generating correlated geometric Brownian motions. In the context of simulating multidimensional SDE's, however, it is more common to use independent Brownian motions as any correlations between components of the vector, $\mathbf{X}_t$, can be induced through the matrix, $\boldsymbol{\sigma}(t, \mathbf{X}_t)$.

1.2 Weak and Strong Convergence of Discretization Schemes

There are two approaches for measuring the error in a discretization scheme $\{\widehat{X}_0, \widehat{X}_h, \widehat{X}_{2h}, \ \ldots \ , \widehat{X}_{mh}\}$ with $m = \lfloor T/h \rfloor$. A *strong* error criterion might take the form

$$\mathrm{E}\left[||\widehat{X}_{mh} - X_T||^q \right] \tag{3}$$

$$\mathrm{E}\left[\sup_{0 \leq t \leq T} ||\widehat{X}_{\lfloor t/h \rfloor h} - X_t|| \right]$$

for some vector norm $|| \cdot ||$ and with $q = 1$ or $q = 2$ in (3). In contrast, a *weak* error criterion takes the form

$$\left| \mathrm{E}[f(\widehat{X}_{mh})] - \mathrm{E}[f(X_T)] \right| \tag{4}$$

where f ranges over "smooth" functions from $\mathbb{R}^d$ to $\mathbb{R}$. Note that with a weak error criterion, all that matters is the distribution of $\widehat{X}_{mh}$ and how it compares to the distribution of X_T and so it's possible to have a very small weak error even if $\widehat{X}_{mh}$ and X_T live on different probability spaces. In finance applications we generally care about derivatives prices which are (risk-neutral) expectations and so the weak criterion of (4) is more appropriate. Given an error criterion, we can assess the performance of the Euler scheme (and others) via its *order of convergence*. We have the following definitions.

Definition 1 *We say the discretization $\widehat{X}$ has a* **strong order of convergence** *of $\beta > 0$ if*

$$\mathrm{E}\left[||\widehat{X}_{mh} - X_T||\right] \leq ch^\beta \tag{5}$$

for some constant c and all sufficiently small h.

Definition 2 *We say the discretization $\widehat{X}$ has a* **weak order of convergence** *of $\beta > 0$ if*

$$\left|\mathrm{E}[f(\widehat{X}_{mh})] - \mathrm{E}[f(X_T)]\right| \leq ch^\beta \tag{6}$$

for some constant c (possibly depending on f), all sufficiently small h, and all $f \in C_P^{2\beta+2}$ where $C_P^{2\beta+2}$ consists of functions whose derivatives of all orders up to $2\beta + 2$ are polynomially[2] bounded.

We note that a larger value of β in (5) and (6) is better in that it implies a faster convergence of the discretization error to 0. In practice, it is often the case that a given discretization scheme will have a smaller strong order of convergence than its weak order of convergence. The Euler scheme, for example, has a strong order of $\beta = 1/2$ whereas[3] its weak order is $\beta = 1$.

It is also worth noting that the conditions on f in Definition 2 are often not met in practice. For example, if f represents the payoff of a simple European call option, then f will not be differentiable and certainly we will not have $f \in C_P^{2\beta+2}$. Similarly, technical conditions on $\boldsymbol{\mu}(t, \mathbf{X}_t)$ and $\boldsymbol{\sigma}$ are also sometimes violated in practice. This means, for example, that if we are using an Euler scheme for such an SDE then there is no theoretical guarantee that it will have a weak order of convergence of $\beta = 1$. As a result, experimentation is often required to understand which schemes perform better, i.e. have a superior order (of weak) convergence for a given payoff f and / or SDE X_t.

2 Other Discretization Schemes

There are several other discretization schemes that (typically) improve on the Euler scheme. We briefly discuss them here.

2.1 The Milstein Scheme

Consider a scalar SDE of the form $dX_t = \mu(X_t)dt + \sigma(X_t)dW_t$ with corresponding Euler scheme

$$\widehat{X}_{kh} = \widehat{X}_{(k-1)h} + \mu(\widehat{X}_{(k-1)h})h + \sigma(\widehat{X}_{(k-1)h})\sqrt{h}Z_k.$$

Without going into the specific details, we can apply Itô's Lemma to $\sigma(X_t)$ to construct a superior approximation for the diffusion term over the interval $[(k-1)h, kh]$. This leads to "the" Milstein scheme

$$\widehat{X}_{kh} = \widehat{X}_{(k-1)h} + \mu(\widehat{X}_{(k-1)h})\, h + \sigma(\widehat{X}_{(k-1)h})\, \sqrt{h}\, Z_k + \frac{1}{2}\sigma'(\widehat{X}_{(k-1)h})\sigma(\widehat{X}_{(k-1)h})h(Z_k^2 - 1) \tag{7}$$

where $\sigma'(x)$ denotes the derivative of σ w.r.t. to x. The approximation in (7) means that both the drift and diffusion terms have both been expanded to $O(h)$. In contrast, the Euler scheme expands the drift to $O(h)$ but only the diffusion term to $O(\sqrt{h})$. Under various smoothness conditions (which again often do not hold in practice) it can be shown that the Milstein scheme has a weak and strong order of convergence of $\beta = 1$. While the Milstein scheme is easy to implement for scalar diffusions it is much more challenging in the multidimensional case because the $O(h)$ approximation to the diffusion term results in "off-diagonal" terms of the form

$$\int_t^{t+h} [W_u^{(k)} - W_t^{(k)}]\, dW_u^{(j)}$$

for $k \neq j$ and simulating such terms is difficult. As a result, the Milstein scheme is typically only ever applied in the scalar case.

[2] A function $g : \mathbb{R}^d \mapsto \mathbb{R}$ is polynomially bounded if there exists constants k and q such that $g(x) \leq k(1 + ||x||^q)$ for all $x \in \mathbb{R}^d$.

[3] These orders of convergence for the Euler scheme require additional smoothness conditions on the coefficients $\boldsymbol{\mu}(t, \mathbf{X}_t)$ and $\boldsymbol{\sigma}$. See Section 6.1.2 of Glasserman for further details.

2.2 Second Order Schemes

It is possible to refine the Euler scheme beyond the Milstein refinement of (7) to obtain schemes of weak order 2. Again, these schemes are generally only applicable in the scalar case but under certain "commutativity" conditions they can be implemented in the multi-dimensional case. See Section 6.2 of Glasserman for further details.

2.3 The Euler Scheme With Richardson Extrapolation

An alternative to second order schemes is the Euler scheme with Richardson extrapolation. This is easy to implement and often has superior performance to second order schemes, especially in high dimensions. As a result, the Euler scheme with Richardson extrapolation is often considered to be a benchmark scheme for reducing discretization error. In order to simplify notation, we will write $\widehat{X}_T^h$ for $\widehat{X}_{\lfloor T/h \rfloor h}$ with the superscript h in $\widehat{X}_T^h$ used to explicitly denote the length of the time step in the scheme.

We now recall that the Euler (often) scheme has weak order 1 so that

$$\left| \mathrm{E}[f(\widehat{X}_T^h)] - \mathrm{E}[f(X_T)] \right| \leq Ch \tag{8}$$

for some constant C, all sufficiently small h and suitably smooth f. Talay and various colleagues have shown that (8) can sometimes be strengthened to

$$\mathrm{E}[f(\widehat{X}_T^h)] = \mathrm{E}[f(X_T)] + ch + o(h) \tag{9}$$

where c depends on f. In this case we can apply (9) with discretization step $2h$ to obtain

$$\mathrm{E}[f(\widehat{X}_T^{2h})] = \mathrm{E}[f(X_T)] + 2ch + o(h). \tag{10}$$

We can then combine the two estimators in (9) and (10) to eliminate the leading $O(h)$ error term. Specifically, we have

$$2\mathrm{E}[f(\widehat{X}_T^h)] - \mathrm{E}[f(\widehat{X}_T^{2h})] = \mathrm{E}[f(X_T)] + o(h). \tag{11}$$

This suggests an obvious improvement to the basic Euler scheme (and alternative to second order schemes):

1. Simulate with time step h to estimate $\mathrm{E}[f(\widehat{X}_T^h)]$
2. Simulate with time step $2h$ to estimate $\mathrm{E}[f(\widehat{X}_T^{2h})]$
3. Double the first estimate and subtract the second to obtain an estimate of $\mathrm{E}[f(X_T)]$

In a similar spirit to the use of common random numbers[4], it makes sense to use *consistent* Brownian increments in simulating the paths of $\widehat{X}^h$ and $\widehat{X}^{2h}$ as doing so will typically result in an often substantial reduction in variance. More specifically, each Brownian increment driving $\widehat{X}^h$ is the sum of two of the increments driving $\widehat{X}^{2h}$. This means that if we use $\sqrt{h}Z_1, \sqrt{h}Z_2, \ldots$ as the Brownian increments for $\widehat{X}^h$ then we can use $\sqrt{h}(Z_1 + Z_2), \sqrt{h}(Z_3 + Z_4), \ldots$ as the Brownian increments for $\widehat{X}^{2h}$. Using such a construction amounts to rewriting (11) as

$$\mathrm{E}[2f(\widehat{X}_T^h) - f(\widehat{X}_T^{2h})] = \mathrm{E}[f(X_T)] + o(h). \tag{12}$$

and then computing $2f(\widehat{X}_T^h) - f(\widehat{X}_T^{2h})$ along each sample path. The variance of this estimator is

$$\mathrm{Var}\left(2f(\widehat{X}_T^h) - f(\widehat{X}_T^{2h})\right) = 4\mathrm{Var}\left(f(\widehat{X}_T^h)\right) + \mathrm{Var}\left(f(\widehat{X}_T^{2h})\right) - 4\mathrm{Cov}\left(f(\widehat{X}_T^h),\, f(\widehat{X}_T^{2h}\right).$$

A variance reduction will therefore be obtained if the covariance term is positive. This is not always the case but can be guaranteed under certain monotonicity conditions.

[4] We will discuss them when we study Monte-Carlo methods for estimating the Greeks.

3 Some Examples From Finance

Example 1 (Option Pricing Under GBM)
We consider the pricing of a European call option in the Black-Scholes framework by simulating the SDE

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

with parameters $S_0 = K = 100$, $T = .5$ years, $r = .01$ and $\sigma = .4$. Of course we can price such an option using the Black-Scholes formula (and obtain a value of 11.469) but it is of interest to see how well our discretization schemes perform here. In Figure 1 we have plotted the *mean* absolute error of the Euler scheme with and without Richardson extrapolation as a function of the number of time steps. The results were obtained by simulating 16 million sample paths. We see that the absolute pricing error generally decreases as h decreases (which corresponds to the number of time steps increasing). But occasionally we see the error increase and this can largely be explained by the (unreported) *statistical error*. Even with 16 million samples, the approximate 95% confidence intervals had a width of approximately 2 cents (for both schemes and all step sizes) so this statistical error can sometimes dominate the discretization error and cause the mean error to increase occasionally. But the general trend in the error is clear and as we expect. Moreover, we also see the superior performance of the Euler scheme with Richardson extrapolation kick in at about 500 time steps but due to the statistical noise this superiority may not be so clear on a different set of simulated paths. ▮

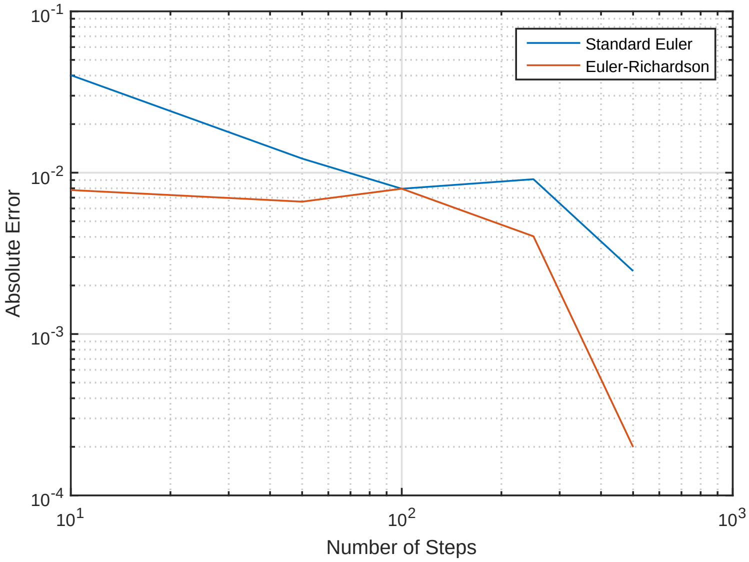

Figure 1: Convergence of the Euler scheme with and without Richardson extrapolation for pricing a European call option under geometric Brownian motion. Both axes are on a log-scale.

Example 2 (Option Pricing Under Heston's Stochastic Volatility Model)
Consider Heston's stochastic volatility model where the evolution of the stock price, S_t, under the risk-neutral probability measure satisfies

$$dS_t = rS_t dt + \sqrt{V_t} S_t dW_t^{(1)} \tag{13}$$
$$dV_t = \kappa (\theta - V_t)\, dt + \sigma \sqrt{V_t} dW_t^{(2)}. \tag{14}$$

with $dW_t^{(1)} \times dW_t^{(2)} = \rho\, dt$. We again wish to price a European call option on the stock and use the same parameters as those in Example 6.2.2 of Glasserman. We therefore take $T = 1$, $S_0 = K = 100$ and $r = 0.5$ for the call option parameters. Our process parameters are $V_0 = 0.04$, $\kappa = 1.2$, $\theta = 0.04$, $\rho = -0.5$ and $\sigma = 0.3$. While an explicit formula for the call option price is not available, we can price it extremely accurately using Fourier inversion methods and we find it's price to be 10.3009. We can use this price to compare the absolute error of various discretization schemes as a function of the number of time steps. As in Example 1, we consider the Euler scheme with and without Richardson extrapolation and also consider a second order scheme whose details we do not provide here. Results are plotted in Figure 2 with each point based on 8 million sample paths.

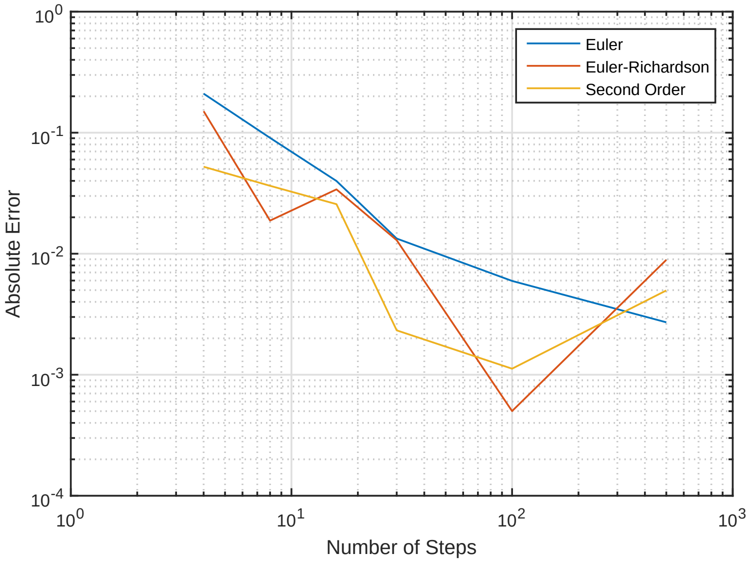

Figure 2: Convergence of various schemes for pricing a European call option under Heston's stochastic volatility model. Both axes are on a log-scale.

We again see the general decrease in the mean absolute error of all three schemes as the number of time steps increases. As discussed at the end of Section 1.2, the various conditions (on both the option payoff and the SDE) that are required to guarantee a given order of convergence of the schemes are often not satisfied in financial applications and that is also the case here. Moreover, even if the conditions were satisfied it may be the case that a very small value of the time-step h would be necessary before the stated order of convergence actually became apparent. These observations and in particular the (unreported) statistical error help explain the somewhat erratic convergence of the schemes and the apparently superior performance of the Euler scheme when 500 time-steps are employed. This apparent "superior" performance can easily switch to an "inferior" performance with an alternative set of simulated sample paths.

It is also worth noting that the Euler scheme (which is often the default scheme for practitioners) can perform extremely poorly in practice with Heston's stochastic volatility model. For example, Andersen reports the following results for pricing an at-the-money 10-year call option when $r = q = 0$. He takes $\kappa = .5$, $V_0 = \theta = .04$, $\sigma = .1$, $S_0 = K = 100$ and $\rho = -0.9$. Using one million sample paths and a "sticky zero" or "reflection" assumption[5], he obtains the estimates displayed in Table 2 for the option price as a function of m, the number of discretization points. Note that the true price (calculated via Fourier inversion) is 23.69 and so it's clear that the Euler scheme with the reflection assumption converges very slowly and that using as many as 1,000 time steps results in an estimated option price that is off by more than 40%. One therefore needs to be very careful when applying an Euler scheme to this SDE. Note that these convergence problems would be easily identified if we followed the procedure outlined immediately following (21) in Section 4.4. But for this particular process, one should use a better scheme such as that proposed by Andersen (2007) or perhaps the second-order or Euler with Richardson extrapolation schemes. In general then, experimentation with step size, sample size and

[5] The sticky zero assumption simply means that anytime the variance process, V_t, goes negative in the Monte-Carlo it is replaced by 0. The reflection assumption replaces V_t with $|V_t|$. In the limit as $m \to \infty$, the variance will stay non-negative with probability 1 so both assumptions are unnecessary in the limit.

Table 1: Call Option Price Estimates Using Euler Scheme in Heston's Stochastic Volatility Model

Time Steps	Sticky Zero	Reflection
100	28.3	45.1
200	27.1	41.3
500	25.6	37.1
1000	24.8	34.6

discretization scheme is often required for a given application. ▮

Example 3 (The CIR Model with Time-Dependent Parameters)
Consider a generalized CIR model for the short-rate, r_t. We assume its risk-neutral dynamics satisfy

$$dr_t \;=\; \alpha[\mu(t) - r_t]\; dt \;+\; \sigma\sqrt{r_t}\; dW_t \tag{15}$$

where $\mu(t)$ is a deterministic function of time. This CIR model is used when we want to fit a CIR-type model to the initial term-structure. Suppose now that we wish to price a derivative security maturing at time T with payoff $C_T(r_T)$. Then its time 0 price, C_0, is given by

$$C_0 = \mathbb{E}_0\left[e^{-\int_0^T r_s\; ds}\; C_T(r_T)\right]. \tag{16}$$

The distribution of r_t is not available in an easy-to-use closed form so perhaps the easiest way to estimate C_0 is by simulating the dynamics of r_t. Towards this end, we could either use (15) and simulate r_t directly or alternatively, we could simulate $X_t := f(r_t)$ where $f(\cdot)$ is an invertible transformation. Note that because of the discount factor in (16), it is also necessary to simulate the process, Y_t, given by

$$Y_t = \exp\left(-\int_0^t r_s\; ds\right).$$

▮

Exercise 2 *Describe in detail how you would you would estimate C_0 in Example 3. Note that there are alternative ways to do this. What way do you prefer?*

Exercise 3 *Have you ever implemented a discrete-time delta hedging strategy in the Black-Schole framework. If so, what discretization scheme did you use?*

---

4 Improvements and Extensions

4.1 Change of Variables

Once we have fixed a discretization scheme, we still have considerable flexibility[6] in choosing what process we apply it to. More specifically, if we wish to simulate a discretized version of $\mathbf{X}_t \in \mathbb{R}^d$ then we can apply our scheme to $\mathbf{X}_t$ or to $\mathbf{Y}_t := g(\mathbf{X}_t)$ where $g : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a smooth invertible function. If we choose to apply it to $\mathbf{Y}_t$ then $\widehat{\mathbf{X}}_{kh} := g^{-1}(\widehat{\mathbf{Y}}_{kh})$ is the corresponding discretized scheme for $\mathbf{X}_t$.

It is often the case that a particular transformation seems intuitively appealing. In financial applications, for example, the SDE often describes security price dynamics and so it is desirable for the discretized scheme to maintain the property of non-negative prices. This can be accomplished by applying the scheme to $\mathbf{Y}_t := \log(\mathbf{X}_t)$ with $g^{-1}(\widehat{\mathbf{Y}}_{kh}) = \exp(\widehat{\mathbf{Y}}_{kh})$ which is always non-negative.

[6] Note that this flexibility is what we had in mind in Exercise 2 above when we mentioned "alternative ways" to estimate C_0.

Exercise 4 *Characterize the discretization error that results from applying an Euler scheme to* $\log(S_t)$ *when* S_t *follows a geometric Brownian motion.*

Exercise 5 *Suppose we wish to simulate the known dynamics of a zero-coupon bond. How would you ensure that the simulated process satisfies* $0 < Z_t^T < 1$ *?*

An important advantage of this flexibility in that we can seek a g with a view to minimizing discretization error. A common strategy is to choose a g (if possible) so that the dynamics of $\mathbf{Y}_t := g(\mathbf{X}_t)$ have a constant volatility coefficient. (This is what we do when we take $Y_t := \log(X_t)$ when $X_t \sim$ GBM.)

4.2 Simulating Jump-Diffusion Processes

Consider a jump-diffusion process of the form

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t + c(X_{t-}, Y_{N_{t-}+1})dN_t \tag{17}$$

where N_t is a Poisson process (independent of W_t) with parameter λ and the Y_i's are IID random variables independent of the Brownian motion W_t. The notation X_{t-} refers to $\lim_{u \uparrow t} X_u$, i.e. the limit of X_u as $u \to t$ from the left. If t is a jump time then X_{t-} is the value of the process *immediately before* t. Note that if the n^{th} jump in the Poisson process occurs at time t, then

$$X_t - X_{t-} = c(X_{t-}, Y_n)$$

If a jump does not occur at time t then $X_{t-} = X_t$. An obvious approach to simulating a discretized version of (17) on the interval $[0, T]$ is:

1. First simulate the arrival times in the Poisson process up to time T.
2. Use a pure diffusion discretization between the jump times.
3. At the n^{th} jump time τ_n, simulate the jump size $c(\widehat{X}_{\tau_n -}, Y_n)$ conditional on the value of the discretized process, $\widehat{X}_{\tau_n -}$, immediately before τ_n.

Exercise 6 *Suppose the process* N_t *in (17) is a more general* jump process *with stochastic intensity* $\lambda(X_t)$. *If the intensity is bounded above by some constant* $\bar{\lambda}$, *how would you extend the scheme outlined above to this new process?*

4.3 Variance Reduction Techniques for Simulating SDE's

Simulating SDE's is a computationally intensive task as we need to do a lot of work for each sample that we generate. Naturally, variance reduction techniques can be very useful in such contexts. We give one example based on stratified sampling and the Brownian bridge. Note that these ideas could be applied very generally to many different models. A further example will be discussed in Exercise 9 of Section 5.1.

Example 4 (The Brownian Bridge and Stratified Sampling)
Consider a short rate model of the form

$$dr_t = \mu(t, r_t)dt + \sigma(t, r_t)dW_t.$$

When pricing a derivative that matures at time T using an Euler scheme it is necessary to generate the path $(W_h, W_{2h}, \ldots, W_{mh} = W_T)$. It will often be the case, however, that the value of W_T will be particularly significant in determining the payoff. As a result, we might want to stratify using the random variable, W_T. This is easy to do for the following two reasons.

(i) $W_T \sim \mathsf{N}(0, T)$ so we can easily generate a sample of W_T and

(ii) We can easily generate $(W_h, W_{2h}, \ldots, W_{T-h} \mid W_T)$ by computing the relevant conditional distributions and then simulating from them. For example, it is straightforward to see that

$$(W_t \mid W_s = x, W_v = y) \ \sim \ \mathsf{N}\left(\frac{(v-t)x + (t-s)y}{v-s}, \ \frac{(v-t)(t-s)}{v-s}\right) \quad \text{for } s < t < v \tag{18}$$

and we can use this result to generate $(W_h | W_0, W_T)$. More generally, we can use (18) to successively simulate $(W_h|W_0, W_T)$, $(W_{2h}|W_h, W_T)$, $\ldots$, $(W_{T-h}|W_{T-2h}, W_T)$.

We can in fact simulate the points on the sample path in any order we like. In particular, to simulate W_v we use (18) and condition on the two closest sample points before and after v, respectively, that have already been sampled. This method of pinning the beginning and end points of the Brownian motion is known as a *Brownian bridge* construction. ▮

Exercise 7 *If we are working with a multi-dimensional correlated Brownian motion, W_t, (e.g. in the context of a multi-factor model of the short rate) is it still easy to use the Brownian bridge construction where we first generate the random vector, W_T?*

4.4 Allocation of Computational Resources

An important issue that arises when simulating SDE's is the allocation of computational resources. In particular, we need to determine how many sample paths, n, to generate and how many time steps, m, to simulate on each sample path. A smaller value of m will result in greater bias and numerical error, whereas a smaller value of n will result in greater statistical noise. Indeed numerical and statistical error were both discussed in Examples 1 and 2 but we did not discuss the optimal tradeoff between the two in those examples. That is the problem we now discuss: how to choose n and m in an optimal manner given a fixed computational budget.

Suppose then $dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t$ and that we wish to estimate $\theta := \mathrm{E}[f(X_T)]$ using a discretization scheme with weak order β. The bias then satisfies

$$\text{Bias} \approx am^{-\beta}$$

for some constant a and all sufficiently large m.

Suppose now that we have a fixed computational budget, C, and that each simulation step costs c. We must therefore have $n = C/mc$. We would like to choose the optimal values of m (and therefore n) as a function of C. We do this by minimizing the *mean squared error* (MSE), which is the sum of the bias squared and the variance, v/n. In particular, we have

$$\text{MSE} \approx \frac{a^2}{m^{2\beta}} + \frac{v}{n} \tag{19}$$

for sufficiently large m. Substituting for n in (19), it is easy to see that it is optimal (for sufficiently large C) to take

$$m \ \propto \ C^{1/(2\beta+1)} \tag{20}$$

$$n \ \propto \ C^{2\beta/(2\beta+1)} \tag{21}$$

with the optimal MSE $\approx C^{-\frac{2\beta}{2\beta+1}}$. Note that the RMSE ($= \sqrt{\text{MSE}} \approx C^{-\frac{\beta}{2\beta+1}}$) approaches $C^{-1/2}$ as $\beta \to \infty$, which is (why?) as expected.

When it comes to estimating θ, (20) and (21) provide guidance as follows. Suppose we are using an Euler scheme with $\beta = 1$. We begin by using n_0 paths and m_0 discretization points per path to compute an initial estimate, $\widehat{\theta}_0$, of θ. If we then compute a new estimate, $\widehat{\theta}_1$, by setting $m_1 = 2m_0$, then (20) and (21) suggest we should set $n_1 = 4n_0$. We may then continue to compute new estimates, $\widehat{\theta}_i$, in this manner until the estimates and their associated confidence intervals converge. In general, if we increase m by a factor of 2 then we should increase n by a factor of 4. Although estimating θ in this way requires additional computational resources, it is not usually necessary to perform more than two or three iterations, provided we begin with sufficiently large values of m_0 and n_0. Note that *Multilevel Monte-Carlo* (which is discussed in Appendix 6) is a more recently developed technique and sophisticated approach to determining an optimal allocation of computational resources.

5 Extremes and Barrier Crossings

Example 3 showed how certain forms of path dependence can be handled by including additional state variables. But other types of dependence can be more problematic, even when the inclusion of additional state variables is appropriate. We begin by handling the extremes of a process.

5.1 Extremes

Suppose X_t is a standard Brownian motion and let $M_t := \max_{0 \leq u \leq t} X_t$ denote the running maximum of the process with

$$\widehat{M}_m^h := \max\left\{X_0, \widehat{X}_h, \widehat{X}_{2h}, \ldots, \widehat{X}_{mh}\right\} \tag{22}$$

denoting the maximum of the corresponding Euler process up to time $T = mh$. It can be shown that the weak order of convergence of this discretization scheme for M_t cannot be better than $1/2$. Note that the Euler scheme for X_t is exact (since it's a Brownian motion) and has a weak order of convergence equal to 1. The apparent discrepancy between the two orders of convergence is that the max process M_t is singular – note we can't use Itô's Lemma to write dynamics for M_t and therefore don't have a direct Euler scheme for M_t. The upshot of this is that simulating discretized schemes for the extremes of a process is inherently more difficult.

There are ways around this problem, however. Again in the case where X_t is a Brownian motion we can simulate M_T directly for any value of T. We do this by:

1. Simulating $X_T \sim \mathsf{N}(0, T)$
2. Simulating $M_T \,|\, X_T$. This amounts to simulating from the maximum of a Brownian bridge with its endpoints fixed at $X_0 = 0$ and X_T fixed at its simulated value in step 1. This can be done because it is known that

$$M_T \,|\, X_T \;\sim\; \frac{X_T + \sqrt{X_T^2 - 2T\log U}}{2} \tag{23}$$

   where $U \sim \mathsf{U}(0, 1)$ (independent of X_T).

This procedure can easily be adapted to handle more general processes like X_t defined by (1). As before we use a discretization scheme to obtain $\widehat{X}_{kh}$ for $k = 0, 1, \ldots$. Rather than using (22) (which amounts to using a piecewise linear interpolation of the $\widehat{X}_{kh}$'s to approximate X_t for any t), we can instead interpolate over each interval $[kh, (k+1)h]$ by using a Brownian bridge with fixed parameters $\mu(kh, \widehat{X}_{kh})$ and $\sigma_k := \sigma(kh, \widehat{X}_{kh})$. That is given the endpoints $\widehat{X}_{kh}$ and $\widehat{X}_{(k+1)h}$ the maximum of the process on $[kh, (k+1)h]$ can be simulated as

$$\widehat{M}_k = \frac{\widehat{X}_{(k+1)h} + \widehat{X}_{kh} + \sqrt{\left(\widehat{X}_{(k+1)h} - \widehat{X}_{kh}\right)^2 - 2h\sigma_k^2 \log U_k}}{2} \tag{24}$$

where the U_k's are IID $\mathsf{U}(0, 1)$ random variables. The maximum of X over $[0, T]$ can then approximated using

$$\max\left\{\widehat{M}_0, \widehat{M}_h, \widehat{M}_2, \ldots, \widehat{M}_{m-1}\right\}$$

and this scheme can be very effective.

5.2 Barrier Crossings

The same technology we discussed for extremes in Section 5.1 can be immediately applied to the pricing of barrier options when we have to simulate an SDE. Suppose, for example, that we wish to price a knock-out put option with time T payoff of the form

$$(K - X_T)^+ 1_{\{\tau > T\}}$$

where $\tau = \inf\{t \geq 0 \, : \, X_t > B\}$ and with $X_0 < B$. The simplest approach, analogous to (22), would be to approximate τ with $\widehat{\tau}$ where

$$\widehat{\tau} := \inf\{k \, : \, \widehat{X}_{kh} > B\}.$$

But we can do much better by using the construction in (24). We note that barrier is crossed in the interval $[kh, (k+1)h]$ if the maximum of the process in that interval exceeds B. We can thus approximate the option payoff with

$$(K - \widehat{X}_{mh})^+ \prod_{k=0}^{m-1} 1_{\left\{\widehat{M}_k \leq B\right\}} \tag{25}$$

with $\widehat{M}_k$ generated as in (24) and $nh = T$.

Exercise 8 *We can simplify the approximation of the survival indicator $1_{\{\tau > T\}}$ in (25) with*

$$\prod_{k=0}^{m-1} 1_{\{U_k \leq \widehat{p}_k\}} \tag{26}$$

where the U_k's are as defined in (24). Provide an expression for $\widehat{p}_k$ in terms of B, $\widehat{X}_{kh}$ and $\widehat{X}_{(k+1)h}$.

Exercise 9 *Following on from the previous exercise, explain how this leads to a superior estimator (of the option payoff) of the form*

$$(K - \widehat{X}_{mh})^+ \prod_{k=0}^{m-1} \widehat{p}_k.$$

In what sense is this estimator superior to the estimator in (25)? Is there any sense in which the estimator might be inferior?

---

6 Appendix: Multilevel Monte-Carlo

Multilevel Monte-Carlo is a recently developed[7] approach that looks to optimize the allocation of computational resources in the simulation of the SDE with the goal of minimizing the estimator's MSE. We can motivate the technique by considering the *Paley-Wiener representation* of Brownian motion on the interval $[0, 2\pi]$. This representation has the form

$$W_t = Z_0 \frac{t}{\sqrt{2\pi}} + \frac{2}{\sqrt{\pi}} \sum_{n=1}^{\infty} Z_n \frac{\sin\left(\frac{nt}{2}\right)}{n} \tag{27}$$

where the Z_i's are IID $\mathsf{N}(0,1)$. This representation suggests an obvious approximation to W_t based on truncating the infinite sum in (27). Specifically we can take

$$W_t^{(m)} = Z_0 \frac{t}{\sqrt{2\pi}} + \frac{2}{\sqrt{\pi}} \sum_{n=1}^{m} Z_n \frac{\sin\left(\frac{nt}{2}\right)}{n} \tag{28}$$

as an approximation to W_t and it should be clear that the approximations becomes increasingly accurate as we increase m. In Figure 3 we have plotted these approximations for a series of m values on a given Brownian path.

[7] The multilevel approach was developed by Giles (2008, *Operations Research*) but our approach here follows an expository paper by Higham (2015, *International Journal of Computer Mathematics*). This latter paper should be consulted for further details on the approach.

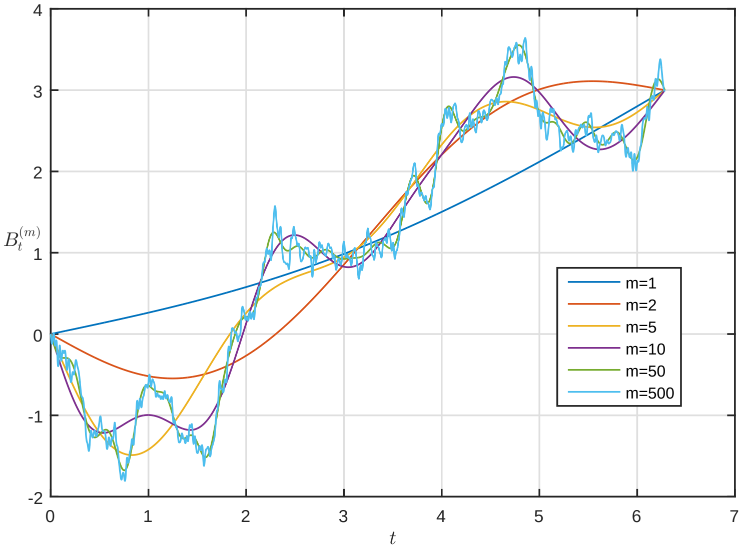

Figure 3: Paley-Wiener Representation of Brownian Motion

It should be clear that earlier terms in the series determine the overall shape of the Brownian path while the later terms add the finer details and improve the resolution of the approximation. Since it is typically the case that the earlier terms (which determine the overall shape of the path) are more important for determining the quantity of interest, e.g. the payoff of an option, it makes sense that we could construct a superior estimator by focusing more effort on simulating the Z_i's for small values of i rather than large values of i. This is essentially the insight used by the multilevel method.

The multilevel approach for simulating an SDE and estimating $\mathrm{E}\left[h(X_T)\right]$ uses a range of step-sizes. In particular it uses step-sizes of the form

$$h_l := \frac{T}{M^l}, \qquad l = 0, \ldots, L \tag{29}$$

where $M > 1$ is a fixed quantity (that is often set equal to 2) and

$$L := \frac{\log \epsilon^{-1}}{\log M}.$$

Note that when $l = L$ in (29) we have $h_l = O(\epsilon)$, the step-size needed by an Euler scheme to achieve a weak error of $O(\epsilon)$. The multilevel scheme works by applying an Euler scheme to the SDE in (1) for each step-size h_l. If we let $\widehat{P}_l$ to denote the estimate of $h(X_T)$ on the discretized path with step-size h_l then we have

$$\mathrm{E}\left[\widehat{P}_L\right] = \mathrm{E}\left[\widehat{P}_0\right] + \sum_{l=1}^{L} \mathrm{E}\left[\widehat{P}_l - \widehat{P}_{l-1}\right] \tag{30}$$

We estimate $\mathrm{E}\left[\widehat{P}_L\right]$ by estimating each of the terms on the right-hand-side of (30) with each such term estimated *independently* of the other terms. However, it is important that $\widehat{P}_l$ and $\widehat{P}_{l-1}$ in (30) are computed on the *same* (discretized) paths (as is the case with Richardson extrapolation). Let N_0 and N_l be the number of paths used to estimate $\mathrm{E}\left[\widehat{P}_0\right]$ and $\mathrm{E}\left[\widehat{P}_l - \widehat{P}_{l-1}\right]$, respectively for $l = 1, \ldots, L$. For a fixed computational budget, the multilevel approach optimizes over the N_l's with the objective of minimizing the mean-squared error of the estimator in (30). Subject to technical conditions on h and the coefficients of the SDE, the multilevel algorithm achieves a weak error of $O(\epsilon)$ (as with the Euler scheme) but with a computational speedup of almost $O(\epsilon^{-1})$. This means, for example, that when an accuracy of 2 decimal places is required, i.e. $\epsilon = .01$, the computations using the multilevel approach will run approximately $1/.01 = 100$ times faster than the regular Euler scheme.