Lecture 8

Finite differences: further considerations

Here we will complete our look at finite-difference methods by looking at how to deal with American options in a more sophisticated fashion. There are two popular ways of dealing with American options, we can either use penalty methods, policy iteration, or body-fitted co-ordinates. Penalty methods and policy iteration are improvements to the PSOR method, allowing direct solver to be used speeding up calculations. Body fitted coordinates allow us to fit the grid to avoid any non-linearity error even when there is early exercise, the speed and efficiency of these methods cannot be matched but they are harder to code and require a deep understanding of the problem you are solving (i.e. they are hard to generalise).

8.1 Policy Iteration and the Penalty Method

American options are options which can be exercised at any time to receive $S - X$ or $X - S$ for call and put options respectively. Unfortunately this gives rise to a **non-linear** problem and as such it is not possible in general to derive explicit formulae like those for European options.

American put options

The first problem is to decide at which values of S and t it is optimal to exercise. To consider the problem, treat the American put option as a European put option with the extra early exercise feature. At expiry the early exercise condition has no effect, as the value of the American put, $P(S, t)$, is given by

$$P(S, T) = \max(X - S, 0).$$

Moving back from expiry there will, however, be certain values of S for which

$$X - S > P_{BS}(S,t)$$

where $P_{BS}(S,t)$ is the value of the European put option derived from the Black-Scholes PDE. In this case the holder of the option would exercise their right and receive $X - S$. The major problem is to locate the value of S at which it becomes optimal to exercise the option, if we call this value $S_f(t)$ then we have

$$P(S,t) = \begin{cases} X - S & \text{for} \quad S \leq S_f(t) \\ P_{BS}(S,t) & \text{for} \quad S > S_f(t). \end{cases}$$

This is known as a free boundary problem and they are very difficult to solve. More formally when pricing American options the Black-Scholes equation becomes an inequality, which is an equality when it is optimal to hold the option:

$$S_f(t) < S < \infty : \qquad P > X - S, \quad \frac{\partial P}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + rS\frac{\partial P}{\partial S} - rP = 0, \tag{8.1}$$

and an inequality when it is optimal to exercise

$$0 \leq S < S_f(t) : \qquad P = X - S, \quad \frac{\partial P}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + rS\frac{\partial P}{\partial S} - rP < 0. \tag{8.2}$$

Example 8.1 (Variational Inequality)**.** A problem of this type is known as a variational inequality. Explain using no arbitrage arguments why the inequality on the Black-Scholes PDE holds.

Solution 8.1.

We have already seen how to adapt Explicit methods and SOR method to solve American Options in Section 7.5. Both methods take advantage of (8.1) and (8.2) by taking the maximum of the continuation value and exercise value. Unfortunately both methods have serious drawbacks when it comes to efficiency. For the explicit method, we have

$$V_j^i = \max[\frac{1}{1+r\Delta t}(AV_{j+1}^{i+1} + BV_j^{i+1} + CV_{j-1}^{i+1}), X - j\Delta S]$$

which still suffers from the timestep restriction making your code run very slowly for large values of $jMax$. For PSOR, we have

$$y_j^{i,k+1} = \frac{1}{b_j}(d_j^i - a_j V_{j-1}^{i,k+1} - c_j V_{j+1}^{i,k})$$

$$V_j^{i,k+1} = \max(V_j^{i,k} + \omega(y_j^{i,k+1} - V_j^{i,k}), X - j\Delta S)$$

which doesn't have timestep restrictions. However, the number of iterations required for convergence can be huge (in the thousands) which can be mitigated by choosing ω but it is not

always possible to know what ω is going to be optimal *a priori*. What we would like to do is take advantage of fast direct solvers (such as LU decomposition) but somehow iterate to get the American Option value. Two possible methods are **Policy Iteration** and **Penalty Method**.

8.1.1 Policy Iteration

This method is a slight adaptation of the PSOR method, which tries to combine both direct and iteration methods. In order to implement the scheme, we need some way of *guessing* whether a particular point should be exercised or not. This scheme builds in (8.2) into the matrix equations.

The algorithm can be written as:

- If $V_j^i > X - j\ dS$ then

$$\begin{aligned} a_j &= \tfrac{1}{4}(\sigma^2 j^2 - rj) \\ b_j &= -\frac{\sigma^2 j^2}{2} - \frac{r}{2} - \frac{1}{\Delta t} \\ c_j &= \tfrac{1}{4}(\sigma^2 j^2 + rj) \\ d_j &= -\tfrac{1}{4}(\sigma^2 j^2 - rj)V_{j-1}^{i+1} - (-\frac{\sigma^2 j^2}{2} - \frac{r}{2} + \frac{1}{\Delta t})V_j^{i+1} - \tfrac{1}{4}(\sigma^2 j^2 + rj)V_{j+1}^{i+1} \end{aligned}$$

- Else if $V_j^i \leq X - j\ dS$ then

$$\begin{aligned} a_j &= 0 \\ b_j &= 1 \\ c_j &= 0 \\ d_j &= X - j\ dS \end{aligned}$$

The matrix can then be solved directly using an appropriate method. Obviously the difficulty here is that V_j^i is not known beforehand, so some form of iteration has to be carried out.

Example 8.2. Outline how this scheme might be implemented.

Solution 8.2.

8.1.2 Penalty Method

Consider the problem:-

$$0 < S < \infty : \qquad \frac{\partial P}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + rS\frac{\partial P}{\partial S} - rP + \kappa \max((X - S) - P, 0) = 0, \tag{8.3}$$

where κ is a large number that penalises deviations away from the condition

$$P \geq X - S.$$

If it were the case that $P < X - S$, the penalty term would dominate the equation and

$$\lim_{\kappa \to \infty} P \to X - S.$$

So this term pushes the value back toward the exercise value if it is crossed. If $P \geq S - X$, we just solve the standard BS PDE. Again we can include the penalty term by adapting the matrix equation.

The algorithm can be written as:

- If $V_j^i > X - j\ dS$ then

$$\begin{aligned} a_j &= \tfrac{1}{4}(\sigma^2 j^2 - rj) \\ b_j &= -\frac{\sigma^2 j^2}{2} - \frac{r}{2} - \frac{1}{\Delta t} \\ c_j &= \tfrac{1}{4}(\sigma^2 j^2 + rj) \\ d_j &= -\tfrac{1}{4}(\sigma^2 j^2 - rj)V_{j-1}^{i+1} - (-\frac{\sigma^2 j^2}{2} - \frac{r}{2} + \frac{1}{\Delta t})V_j^{i+1} - \tfrac{1}{4}(\sigma^2 j^2 + rj)V_{j+1}^{i+1} \end{aligned}$$

- Else if $V_j^i \leq X - j\ dS$ then

$$\begin{aligned} a_j &= \tfrac{1}{4}(\sigma^2 j^2 - rj) \\ b_j &= -\frac{\sigma^2 j^2}{2} - \frac{r}{2} - \frac{1}{\Delta t} - \kappa \\ c_j &= \tfrac{1}{4}(\sigma^2 j^2 + rj) \\ d_j &= -\tfrac{1}{4}(\sigma^2 j^2 - rj)V_{j-1}^{i+1} - (-\frac{\sigma^2 j^2}{2} - \frac{r}{2} + \frac{1}{\Delta t})V_j^{i+1} - \tfrac{1}{4}(\sigma^2 j^2 + rj)V_{j+1}^{i+1} - \kappa(X - jdS) \end{aligned}$$

The matrix can then be solved directly using an appropriate method. Iteration will be required, although typically 3 or 4 iterations per timestep are sufficient.

Example 8.3. Outline how this scheme might be implemented.

Solution 8.3.

8.2 Body-fitted coordinates

As discussed, the main problem when applying numerical schemes to problems involving early exercise is that the position of the early exercise boundary is not known a priori. This often means that when constructing a grid this boundary does not often fall on a grid point, or node, causing a degree of non-linearity error. The usual numerical schemes: lattice methods and basic finite difference schemes are unable to fix the grid correctly. The method described here involves keeping the time axis fixed but transforming the S to an $\hat{S}$ axis that moves with the free boundary.

The American put option

The American put option price, $P(S,t)$ is described by the Black-Scholes equation for the following boundary conditions:

$$P(S,T) = \max(X - S, 0)$$

$$P(S \leq S_f(t), t) = X - S$$

$$P(S,t) \to 0 \quad \text{as} \quad S \to \infty$$

where, as usual, $S_f(t)$ is the position of the free boundary at time t. To incorporate the body-fitted co-ordinate system simply introduce the transformation

$$\hat{S} = S - S_f(t)$$

and so the range of $\hat{S}$ is from 0 to ∞. This will also ensure that, at every timestep, the S-dimension of the grid will range from the exact position of the free-boundary to a sufficiently large value of S.

This removes the nonlinearity error present in the previous schemes. This desired effect does come at a cost though. As the position of the early exercise boundary is not known a priori, then the position of all the $\hat{S}$ values are also not known at each point in time. It is, thus, necessary to perform a technique analogous to a multivalued Newton-Raphson iteration. This is undertaken in combination with the usual Crank Nicolson scheme. The first point to note is that the above transformation leads to a slight change in the governing equation, since

$$\frac{\partial P}{\partial t} \to \frac{\partial P}{\partial t} + \frac{\partial P}{\partial \hat{S}}\frac{\partial \hat{S}}{\partial t} = \frac{\partial P}{\partial t} - \frac{\partial P}{\partial \hat{S}}\frac{dS_f}{dt}$$

and so the Black-Scholes equation becomes:

$$\tfrac{1}{2}\sigma^2(\hat{S}+S_f)^2\frac{\partial^2 P}{\partial \hat{S}^2} + r(\hat{S}+S_f)\frac{\partial P}{\partial \hat{S}} + \frac{\partial P}{\partial t} - \frac{\partial P}{\partial \hat{S}}\frac{dS_f}{dt} - rP = 0$$

Fortunately the option value, P and the position of the free boundary $S_f(t=T)$ are both known at expiry (i.e. $S_f = X$ and $P = \max(-\hat{S}, 0)$) and so to calculate the values at $T - \Delta t$ we simply perform an iterative scheme using these values as a starting point.

$$P^{(m+1)}(i\Delta\hat{S}, j\Delta\tau) = P^{(m)}(i\Delta\hat{S}, j\Delta\tau) + \delta P_i \quad \text{for} \quad 0 \le i \le n$$

At a given timestep $j\Delta\tau$, after m iterations write the values at the n distinct P values in the finite difference scheme and the value of S_f as

$$S_f^{(m+1)}(j\Delta\tau) = S_f^{(m)}(j\Delta\tau) + \delta S_f$$

The value of $P_i^{(0)}(i\Delta\hat{S}, j\Delta\tau)$ and $S_f^{(0)}(j\Delta\tau)$ are taken to be the converged values of $P_i(i\Delta\hat{S}, (j+1)\Delta\tau)$ and $S_f((j+1)\Delta\tau)$ respectively. All that remains is to calculate the successive values of the δP_i's and δS_f. In order to do this a Crank-Nicolson scheme is used.

Crank-Nicolson

The principal differences from the basic scheme are, first, that the unknown values are the δP_i's and δS_f. Secondly, there is also be an extra column in the matrix to cope with the new extra unknown, δS_f. There are, now, $N+1$ equations in $N+2$ unknowns, this is easily overcome by using both the boundary conditions at $\hat{S} = 0$ ($S = S_f$) including the condition from before and the smooth pasting condition

$$\frac{\partial P}{\partial \hat{S}} = -1$$

The next step requires arranging the difference equations into a system with the unknown values are on one side and the known ones on the other. The approximation for the derivatives is as

usual. In general $P(i\Delta\hat{S}, j\Delta t)$ is denoted $P_{i,j}$ and the resulting set of equations, for $0 \leq i \leq N$, is

$$a_i\delta P_{i-1} + b_i\delta P_i + c_i\delta P_{i+1} + d_i\delta S_f = e_i$$

where, as before, the values of a_i, b_i, c_i, d_i and e_i need to be calculated.

Boundary conditions

There are two boundary conditions at $\hat{S} = 0$ and one at $\hat{S} = \hat{S}_{max}$. For these the values of a, b, c, d and e are known and are as follows:

- At $\hat{S} = 0$:

  $$P^m_{0,j} + \delta P_0 = X - (S^{(m)}_{f,j} + \delta S_f)$$

  and using a one-sided difference scheme we can generate an equation for the derivative condition

  $$\frac{-3P^{(m)}_{0,j} + 4P^{(m)}_{1,j} - P^{(m)}_{2,j} - 3\delta P_0 + 4\delta P_1 - \delta P_2}{2\Delta\hat{S}} = -1$$

which rearrange to give

$$\delta P_0 + \delta S_f = X - P^{(m)}_{0,j} - S^{(m)}_{f,j},$$
$$3\delta P_0 + 4\delta P_1 - \delta P_2 = -2\Delta\hat{S} + 3P^{(m)}_{0,j} - 4P^{(m)}_{1,j} + P^{(m)}_{2,j}$$

and at $\hat{S}_{max}$:

$$\delta P_N = -P^{(m)}_{N,j}.$$

Method

There are now a series of $N + 2$ equations in $N + 2$ unknowns which can be solved using linear algebra techniques. In matrix form the problem can be displayed as

$$\begin{pmatrix} 1 & 0 & 0 & . & . & . & . & . & 1 \\ -3 & 4 & -1 & 0 & . & . & . & . & 0 \\ a_1 & b_1 & c_1 & 0 & . & . & . & . & d_1 \\ 0 & a_2 & b_2 & c_2 & . & . & . & . & d_2 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & a_i & b_i & c_i & . & d_i \\ . & . & . & . & . & . & . & . & . \\ 0 & . & . & . & . & . & 0 & b_N & 0 \end{pmatrix} \begin{pmatrix} \delta P_0 \\ \delta P_1 \\ \delta P_2 \\ . \\ . \\ \delta P_{N-1} \\ \delta P_N \\ \delta S_f \end{pmatrix} = \begin{pmatrix} X - P^{(m)}_{0,j} - S^{(m)}_{f,j} \\ . \\ e_1 \\ . \\ . \\ . \\ . \\ -P^{(m)}_{N,j} \end{pmatrix}$$

A reduction technique is then applied to solve the system. First, remove all the c_i terms, starting from c_{N-1} by using

$$c_i^{'} = c_i - b_{i+1}\frac{c_i}{b_{i+1}}$$

and as a result the other terms are affected, namely

$$b_i^{'} = b_i - a_{i+1}\frac{c_i}{b_{i+1}}$$

$$d_i^{'} = d_i - d_{i+1}\frac{c_i}{b_{i+1}}$$

$$e_i^{'} = e_i - e_{i+1}\frac{c_i}{b_{i+1}}$$

There is a slight anomaly in the reduction technique in that the second row in the matrix has an extra column. This is easily overcome as this last term is removed using b_2 and the other terms adjusted accordingly and then the c_0 term is removed by using c_1 as before. The matrix is now in the form:

$$\begin{pmatrix} 0 & 0 & 0 & . & . & . & . & . & d_{-1}^{'} \\ b_0^{'} & 0 & 0 & 0 & . & . & . & . & d_0^{'} \\ a_1 & b_1 & 0 & 0 & . & . & . & . & d_1^{'} \\ 0 & a_2^{'} & b_2^{'} & 0 & . & . & . & . & d_2^{'} \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & a_i^{'} & b_i^{'} & 0 & . & d_i^{'} \\ . & . & . & . & . & . & . & . & . \\ 0 & . & . & . & . & . & 0 & b_N^{'} & 0 \end{pmatrix} \begin{pmatrix} \delta P_0 \\ \delta P_1 \\ \delta P_2 \\ . \\ . \\ \delta P_{N-1} \\ \delta P_N \\ \delta S_f \end{pmatrix} = \begin{pmatrix} X - P_{0,j}^{(m)} - S_{f,j}^{(m)} \\ . \\ e_1^{'} \\ . \\ . \\ . \\ . \\ e_N^{'} \end{pmatrix}$$

but from the top row it is possible to calculate the value of δS_f as

$$\delta S_f = \frac{e_{-1}^{'}}{d_{-1}^{'}}$$

and from the second row the value of δP_0 can be deduced

$$\delta P_0 = \frac{e_0^{'} - d_0^{'}\delta S_f}{b_0^{'}}$$

then continue this regime to calculate the values of δP_i for $1 \leq i \leq N$ using

$$\delta P_i = \frac{e_i^{'} - d_i^{'}\delta S_f - a_i^{'}\delta P_{i-1}}{b_i^{'}}$$

This scheme is equivalent to a Newton-Raphson iteration and, as such, must have converged before moving forward to the next timestep.

Depending on the accuracy required, we select a value for which all the δP_i's or δS_f need to be less than to have assumed convergence - a typical value is $1 * 10^{-8}$.

Time adjustment

In order to try and predict the movement of the early exercise boundary, a somewhat unrigorous scaling of the timesteps is carried out. It is known that the early exercise boundary does not move exactly as $\sqrt{\tau}$ (This is because there are also log-terms present in the exact representation of the early exercise boundary as $\tau \to 0$.) However, it is a good enough approximation to ensure that there are no stepsize constraints in the body-fitted Crank Nicolson scheme.

This scaling is very easy to implement. Simply divide time into N steps of size $\Delta t = T/N$ as usual, however when using Δt in the numerical calculation instead use $\Delta t = T/N^2$ Often, when using body-fitted co-ordinates, it can be difficult to correctly determine the position of the boundary (in this case, near to $\tau = 0$) but this ensures that the early exercise boundary follows, approximately, its proper form.

Now that any nonlinearity error has been removed it would be expected that the convergence to the correct option price is at the rate $O((\Delta t)^2, (\Delta S)^2)$. If so, it should be possible to extrapolate results, assuming that there is also no non-linearity error occurring at the strike price at maturity. At expiry the early exercise boundary is at the strike price and, thus, extrapolation can be performed. As the results converge at the same rate in both the t and the S directions it is possible to perform just one extrapolation.

Lecture 9

Monte Carlo methods for American options the Longstaff and Schwartz method

We have looked at using Monte Carlo methods for most types of options, but they struggle with early exercise. This is the subject of lots of research at the moment and we have seen the basic idea of some of the methods. What we present here is one of the easiest methods to understand and implement. There are still issues with how it performs in practise which we will also deal with here.

The key calculations when valuing an American option are determining the early exercise value and the expected continuation value. The option value at any point in time will be the larger of these two values. The early exercise value is simple to calculate at any point in time. The continuation value is typically determined by a backward iteration approach, where the value now is the discounted expected option value at the next instance in time. The key information we need to know is given the share price at time t, what is the expected option value at time $t + \Delta t$. This is simple to determine in binomial models but less so in Monte Carlo approaches.

9.1 The least squares method

This is a technique for fitting a set of functions to (given) data. Here we describe the procedure for an mth degree polynomial - it is straightforward to extend the idea to a general class of polynomial When using an mth degree polynomial

$$y = a_0 + a_1 x + a_2 x^2 + ... + a_m x^m$$

to approximate the given set of data, (x_1, y_1), (x_2, y_2), ..., (x_n, y_n), where $n \geq 3$, the best fitting curve $f(x)$ has the least square error, i.e.,

$$\Pi = \sum_{i=1}^{n} [y_i - f(x_i)]^2 = \sum_{i=1}^{n} [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m)]^2 = \min$$

Note that a_0, a_1, a_2, ..., and a_m are unknown coefficients while all x_i and y_i are given. To obtain the least square error, the unknown coefficients a_0, a_1, a_2, ..., and a_m must yield zero first derivatives.

$$\frac{\partial \Pi}{\partial a_0} = 2 \sum_{i=1}^{n} [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m)] = 0$$

$$\frac{\partial \Pi}{\partial a_1} = 2 \sum_{i=1}^{n} x_i [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m)] = 0$$

$$\frac{\partial \Pi}{\partial a_2} = 2 \sum_{i=1}^{n} x_i^2 [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m)] = 0$$

.....

$$\frac{\partial \Pi}{\partial a_m} = 2 \sum_{i=1}^{n} x_i^m [y_i - (a_0 + a_1 x_i + a_2 x_i^2 + ... + a_m x_i^m)] = 0$$

Expanding the above equations, we have

$$\sum_{i=1}^{n} y_i = a_0 \sum_{i=1}^{n} 1 + a_1 \sum_{i=1}^{n} x_i + \alpha_2 \sum_{i=1}^{n} x_i^2 + \ldots + a_m \sum_{i=1}^{n} x_i^m$$

$$\sum_{i=1}^{n} x_i y_i = a_0 \sum_{i=1}^{n} x_i + a_1 \sum_{i=1}^{n} x_i^2 + \alpha_2 \sum_{i=1}^{n} x_i^3 + \ldots + a_m \sum_{i=1}^{n} x_i^{m+1}$$

$$\sum_{i=1}^{n} x_i^2 y_i = a_0 \sum_{i=1}^{n} x_i^2 + a_1 \sum_{i=1}^{n} x_i^3 + \alpha_2 \sum_{i=1}^{n} x_i^4 + \ldots + a_m \sum_{i=1}^{n} x_i^{m+2}$$

.....

$$\sum_{i=1}^{n} x_i^m y_i = a_0 \sum_{i=1}^{n} x_i^m + a_1 \sum_{i=1}^{n} x_i^{m+1} + \alpha_2 \sum_{i=1}^{n} x_i^{m+2} + \ldots + a_m \sum_{i=1}^{n} x_i^{2m}$$

The unknown coefficients a_0, a_1, a_2, ..., and a_m can hence be obtained by solving the above linear equations. There are also many library routines available to do the job.

9.2 Longstaff and Schwartz (2001)

The Longstaff and Schwartz method, essentially estimates the conditional expected option value at the next time step by simulating lots of paths and then carrying out a regression of the future realised option value as a function of the current value of the underlying asset. This gives an

approximation for the continuation value that can then be compared to the early exercise value and then we know the option value at each point in time on each path. In terms of Monte Carlo pricing, all we actually need to know is the rule for early exercising, so we know when we receive the cash flows and the value of the option is the average of the discounted payoffs for each path. We will explain the method via an example and then describe the general method.

9.3 Example

We will attempt to value a Bermudan put option where exercise is possible now and at three future dates. $S_0 = 1$, $X = 1.1$, $r = 0.06$. The first step is to simulate some paths, the table below denotes the results:

Stock price paths

Path	$t = 0$	$t = 1$	$t = 2$	$t = 3$
1	1.00	1.09	1.08	1.34
2	1.00	1.16	1.26	1.54
3	1.00	1.22	1.07	1.03
4	1.00	.93	.97	.92
5	1.00	1.11	1.56	1.52
6	1.00	.76	.77	.90
7	1.00	.92	.84	1.01
8	1.00	.88	1.22	1.34

We need to use this information to determine the continuation value at each point in time for each path. To do this we will construct a "Cash Flow Matrix" at each point in time.

Continuation value at $t = 2$

The table below denotes the cash flows at $t = 3$ assuming that we held the option that far:

Cash-flow matrix at $t = 3$

Path	$t = 1$	$t = 2$	$t = 3$
1	–	–	.00
2	–	–	.00
3	–	–	.07
4	–	–	.18
5	–	–	.00
6	–	–	.20
7	–	–	.09
8	–	–	.00

The next step is to attempt to find a function that describes the continuation value at time 2 as a function of the value of S at time 2.

To do this we use a regression technique, that takes the values at time 2 as the "x" values and the discounted payoff at time 3 as the "y" values.

Continuation value at $t = 2$

Path	y	x
1	$.00 \times .94176$	1.08
2	–	–
3	$.07 \times .94176$	1.07
4	$.18 \times .94176$	.97
5	–	–
6	$.20 \times .94176$	.77
7	$.09 \times .94176$	.84
8	–	–

Note that the regression is only carried out on paths that are in the money at time 2.

The regression here is simple where y is regressed upon x and x^2 (the actual scheme is slightly more sophisticated). In this particular example (using least squares): $y = -1.070 + 2.938x - 1.813x^2$, and so we can use this to estimate the continuation value for each of the current share prices (x in the regression). For example for path 1, where $S = 1.08$, the regression formula gives y (the continuation value) to be 0.0369.

Option value at $t = 2$

Optimal early exercise decision at time 2

Path	Exercise	Continuation
1	.02	.0369
2	–	–
3	.03	.0461
4	.13	.1176
5	–	–
6	.33	.1520
7	.26	.1565
8	–	–

This then allows you to decide at which points in time you would exercise and thus determine the cash flows at $t = 2$ (below). Notice that for each path, if you exercise at $t = 2$ then you do not also exercise at $t = 3$

Cash-flow matrix at time 2

Path	$t=1$	$t=2$	$t=3$
1	–	.00	.00
2	–	.00	.00
3	–	.00	.07
4	–	.13	.00
5	–	.00	.00
6	–	.33	.00
7	–	.26	.00
8	–	.00	.00

Continuation value at $t = 1$

We can apply the same process to $t = 1$, for each of the paths that are in the money we regress the discounted future cash flows (y) on the current value of the underlying asset (x), where x and y are as given below:

Regression at time 1

Path	y	x
1	$.00 \times .94176$	1.09
2	–	–
3	–	–
4	$.13 \times .94176$	.93
5	–	–
6	$.33 \times .94176$	.76
7	$.26 \times .94176$	.92
8	$.00 \times .94176$	.88

The regression equation here is $y = 2.038 - 3.335x + 1.356x^2$ and again we use this to estimate the continuation value and decide on an early exercise strategy. The next table compares the two values and the final table denotes the early exercise or stopping rule.

Optimal early exercise decision at time 1

Path	Exercise	Continuation
1	.01	.0139
2	–	–
3	–	–
4	.17	.1092
5	–	–
6	.34	.2866
7	.18	.1175
8	.22	.1533

The early exercise strategy is as follows:

Stopping rule

Path	$t=1$	$t=2$	$t=3$
1	0	0	0
2	0	0	0
3	0	0	1
4	1	0	0
5	0	0	0
6	1	0	0
7	1	0	0
8	1	0	0

From this we can then value the option, by forming the final cash flow matrix from this rule.

Option cash flow matrix

Path	$t=1$	$t=2$	$t=3$
1	.00	.00	.00
2	.00	.00	.00
3	.00	.00	.07
4	.17	.00	.00
5	.00	.00	.00
6	.34	.00	.00
7	.18	.00	.00
8	.22	.00	.00

So the option value is the average of the discounted cash flows, so in this case:

$$V_0 = \tfrac{1}{8}(0+0+0.07e^{-3r}+0.127e^{-r}+0.34e^{-r}+0.18e^{-r}+0.22e^{-r}) = 0.1144$$

9.4 More sophisticated regression

In general the regression here $y = a_1 + a_2x + a_3x^2$ is not going to be satisfactory, especially as we will have far more than 8 paths when attempting to find the functional form of the continuation values. In fact the general form of y is:

$$y = \sum_{j=0}^{M} a_j F_j(x)$$

The user decides upon M the number and the functional form $F_j(x)$ (in our example $F_j(x) = x^j$) where $F_j(x)$ is called a basis function. Although Longstaff and Schwartz suggest Laguerre

polynomials provide the best fit, my favourite is Chebyshev polynomials. However, you are free to choice whichever basis functions you want (e.g. Polynomial as in the example, Chebyshev, Hermite, Laguerre, etc.)

The Laguerre polynomials are given by:

$$F_0(x) = exp(-x/2)$$
$$F_1(x) = (1-x)exp(-x/2)$$
$$F_n(x) = exp(-x/2)\frac{e^x}{n}\frac{d^n}{dx^n}(x^n e^{-x})$$

Chebyshev polynomials are given by:

$$F_n(x) = \cos(n(\cos^{-1} x))$$

In particular $T_0(x) = 1$, $T_1(x) = 1$, $T_2(x) = 2x^2 - 1$, ...

Then the least squares approach approximates the constants a_j, and when we have these values we can use them to predict the continuation value for each value of S at every point in time.

9.5 Longstaff and Schwartz - general procedure

1. Decide on the number of sample paths N, the number of basis functions for the regression, M, and the type of basis functions $F_j(x)$ and the number of observation dates d.
2. Draw Nd Normally distributed random numbers and simulate the sample paths for the underlying asset at each point in time $S^n_{t_i}$ $1 \leq i \leq d$, $1 \leq n \leq N$
3. At expiry $t = t_d$, record the cash flow values $CF^n(t_d)$ which for a put are $\max(X - S^n_{t_d}, 0)$
4. Move back to $t = t_{d-1}$ for each path where $S^n_{t_{d-1}} < X$ calculate the continuation value as $CV^n(t_{d-1}) = e^{-r(t_d - t_{d-1})}CF^n(t_d)$. From these values perform the regression to determine the functional form of the continuation value, $y(S)$ where S is the value of the underlying asset
5. Recalculate the continuation value as $CV^n(t_{d-1}) = y(S^n_{td-1})$
6. For every path calculate the cash flow value where if the continuation value $CV^n(t_{d-1}) < X - S^n_{t_{d-1}}$ then $CF^n(t_{d-1}) = X - S^n_{t_{d-1}}$ and $CF^n(t_i) = 0$ for $i > d-1$ otherwise $CF^n(td-1) = 0$
7. Repeat this process for the previous time step until you have $CF^n(t_i)$ for all i and n. Note that in general to calculate $CV^n(t_i)$ before the regression

$$CV^n(t_i) = \sum_{n=i+1}^{d} e^{-r(t_n - t_i)}CF^n(t_n)$$

8. The option value V_0 is then

$$V_0 = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} e^{-rt_j} CF^i(t_j)$$

9. When you have more than one underlying asset in order to perform the regression you need to have basis functions in all of the underlying assets as well as in the cross terms between them (i.e. in S_1, S_2 and $S_1 S_2$).

10. This means that the number of basis functions will increase exponentially as you increase the number of underlying assets, although it is not necessary in practise to have too large a number of basis functions.

9.6 How well does it perform?

Longstaff and Schwartz provide proofs that as $M \to \infty$ and $N \to \infty$ the option value obtained from their scheme converges to the theoretical value. This isn't much use for practical considerations as you will be limited by how many basis functions you can calculate and how many simulations you can perform. There have been a few investigations into the method and the view of the method's performance are mixed: Broadie and Detemple (2004) say regression methods 'often incur unknown approximation errors and are limited by a lack of error bounds'. A detailed appraisal of this technique by Moreno and Navas (2003) investigates the use of various polynomial fits and numbers of basis functions.

It is not clear that increasing the number of basis functions actually increases the accuracy of the method and there is no real difference from using different types of basis functions (e.g. Chebyshev rather than Laguerre) For more complicated derivative pricing problems the trend is even less clear, sometimes errors can increase as you add more basis functions (too many essentially fits the stochastic values of S exactly) In general, the method will provide good estimates but will be difficult to assess exactly how accurate it is.

See Duck et al (2005) for some improvements on the basic Longstaff and Schwartz scheme (the latter provides superoptimal estimates, but these can be exploited using extrapolation).

We have introduced a method for valuing options with early exercise features using simulation. The main idea is to estimate the continuation value (as a function of the current underlying asset price) by performing a least squares regression. The method converges to the correct option price but research shows that it is unclear how well the method performs and how to estimate the error when using a finite number of paths/ basis functions.

Lecture 10

The QUAD method

In this chapter we consider a simple, widely applicable numerical approach for option pricing based on quadrature methods. Though in some ways similar to lattice or finite-difference schemes, it possesses exceptional accuracy and speed. Discretely monitored options are valued with only one time step between observations, and nodes can be perfectly placed in relation to discontinuities. Convergence is improved greatly; in the extrapolated scheme, a doubling of points can reduce error by a factor of two-hundred and fifty-six. Complex problems (e.g fixed-strike lookback discrete barrier options) can be evaluated accurately and orders of magnitude faster than by existing methods.

10.0.1 Introduction

With QUAD, a full range of final positions can be evaluated using just one time-step and the improvement in efficiency is quite remarkable. For example, using a lattice method a doubling of accuracy requires approximately four times the computational effort, whereas using QUAD with Simpson's rule implementation (see later), a four-fold increase in computational time improves accuracy by a factor of sixteen rising to a factor of two-hundred and fifty-six with extrapolation.

With QUAD, nodes can be positioned exactly as required, thus avoiding all non-linearity error, even when pricing such exotic products as moving discrete barrier options and lookback options with barriers. A welcome consequence of this eradication of non-linearity error is that extrapolation can be used to reduce distribution error even further.

QUAD can be seen as analogous to a multinomial lattice in that the option prices are calculated from values at nodes later in time, working backwards from maturity. However, it also has the added flexibility that nodes can be placed wherever desired and only one time step is required between observations of exotic features. In many ways, it could be considered 'the perfect tree' method.

Figure 10.1 shows schematically how QUAD evaluates a down-and-out moving discrete barrier call option; a much more detailed summary of the method follows later. This option has a pay-off at maturity identical to a vanilla-call option, but this pay-off is contingent on the underlying asset remaining above the level B_m at times T_m. If at these observation times the underlying is below the barrier level then the option is 'knocked out' and becomes worthless.

Between barriers, the option value behaves precisely as a vanilla European option, and as a consequence the well-known exact solution for such options can be employed to relate values, backwards in time, between successive barriers. It is only necessary to calculate option values when the special feature, the knock out, applies. In contrast with lattice and finite-difference methods there is no need to consider intermediate timesteps and it is possible to move to any value of the underlying at each time-step. The nodes at each significant time-step may be simply positioned so as to cope with non-linearity, in this case the barriers and the strike price at maturity.

It is well known that the majority of options can be written as either a finite or an infinite set of nested (multiple) integrals.

10.0.2 Building blocks of the QUAD method

Adapting the Black-Scholes equation

As a starting point, consider the well-known Black and Scholes (1973) partial differential equation for an option with an underlying asset following geometric Brownian motion:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + (r - D_c) S \frac{\partial V}{\partial S} - rV = 0, \tag{10.1}$$

where $V(S,t)$ is the price of the derivative product, S the current value of the underlying asset, t is time, T is the time to maturity r the risk-free interest rate, σ the volatility of the underlying asset and X is the exercise price of the option. D_c is a continuous dividend yield which could, for example, be the foreign interest rate in a foreign exchange option.

Next make the following standard transformations in Eq. 10.1

$$x = log(S_t/X), \tag{10.2}$$

$$y = log(S_{t+\Delta t}/X). \tag{10.3}$$

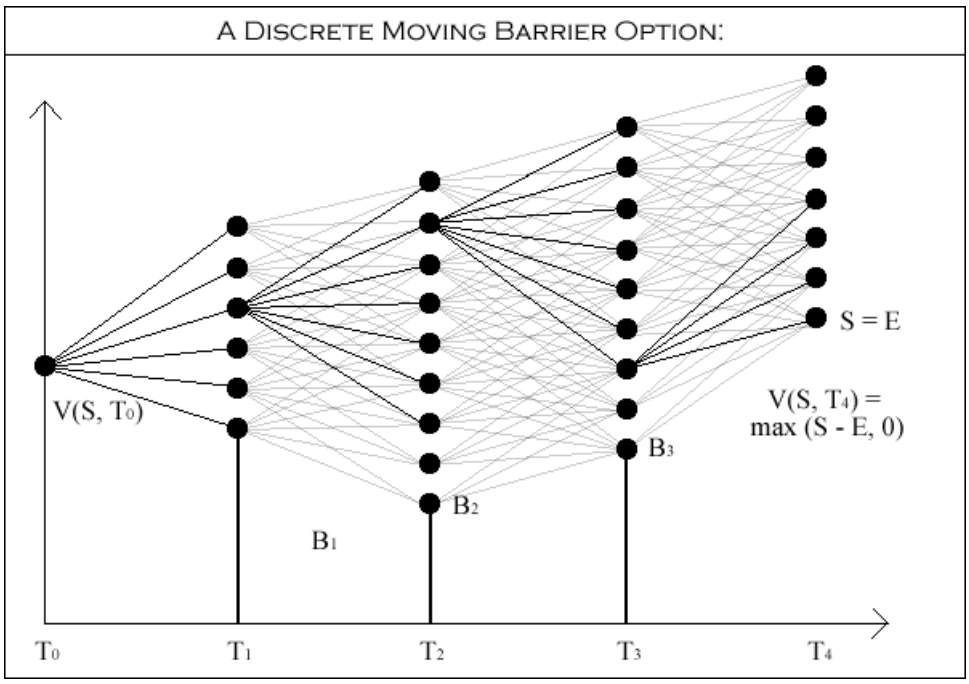

Figure 10.1: A schematic diagram demonstrating the evaluation of a discrete moving barrier option with barriers at B_1, B_2 and B_3, strike price X, time to maturity T_4 and current underlying price of S. QUAD integrates across a set of points at each observation time (reduced in number and shown in bold for only one point in each set before maturity

It is important to note in what follows that Δt is not restricted to small time periods. The final conditions for a European option expiring at time $T = t + \Delta t$ with $V(y, t + \Delta t)$ are transformed in straightforward fashion; e.g. the payoff for a call option $\max(S_{t+\Delta t} - X, 0)$ becomes $\max(Xe^y - X, 0)$. The value of the option at time t on an underlying asset S_t has an exact solution given by

$$V(x,t) = A(x) \int_{-\infty}^{\infty} B(x,y) V(y, t + \Delta t)\, dy, \tag{10.4}$$

where,

$$A(x) = \frac{1}{\sqrt{2\sigma^2 \pi \Delta t}} e^{-\frac{1}{2}kx - \frac{1}{8}\sigma^2 k^2 \Delta t - r\Delta t}, \tag{10.5}$$

and

$$B(x,y) = e^{-\frac{(x-y)^2}{2\sigma^2 \Delta t} + \frac{1}{2}ky} \tag{10.6}$$

and

$$k = \frac{2(r - D_c)}{\sigma^2} - 1. \tag{10.7}$$

Also, let x_0 be the x corresponding to the value of S at the outset, s_t i.e.

$$x_0 = log(S_t/X). \tag{10.8}$$

Henceforth the integrand here will be denoted as $f(x, y)$, so that Eq. (10.4) becomes

$$V(x,t) = A(x) \int_{-\infty}^{\infty} f(x,y) dy \tag{10.9}$$

where the integrand is given by

$$f(x,y) = B(x,y) V(y, t + \Delta t). \tag{10.10}$$

The solution (10.4) contains an integral which, in general, cannot be evaluated analytically in terms of simple functions, although for European options it is easily converted to the probability density function for the Normal distribution. For more complicated options, numerical techniques are required to evaluate the integral in Eq. (10.4). The intuition behind QUAD is that although Eq. (10.4) is not valid across all points in time for anything other than a plain European option, the valuation problem can be sliced into consecutive time intervals, during which Eq. (10.4) is locally applicable. Imposition of the appropriate option features at their corresponding 'observation' times provides a link between these consecutive intervals, and solution of complex problems becomes possible.

10.0.3 Quadrature methods

There are many different methods of numerical integral evaluation. Simpson's rule is used as a starting point because it is robust, simple, nimble, well-known and has fast convergence: of order $(\delta y)^4$, where δy is the size of the gap between points (or of order N^{-4} where N is the number of steps). By comparison a trinomial tree converges merely with N^{-1} or in some cases only with $N^{-\frac{1}{2}}$. Even the more sophisticated finite difference methods at best converge at the rate of $(\Delta S^2, \Delta t^2)$ where ΔS and Δt are the step sizes in the S and t directions respectively. Use of the trapezoidal rule will also be discussed here, although virtually any quadrature method is applicable, including higher-order schemes and Gaussian quadrature.

Simpson's rule

Although Simpson's rule was devised as long ago as 1743, it remains one of the most accurate and popular methods for approximating integrals. The idea is conceptually simple: for a function of y, $f(y)$, plotted against y divide the desired range, $[a_1, a_2]$ into intervals of a fixed length and then approximate the area under the curve by summing the area of the individual regions. This yields the following expression, for a step size of δy,

$$\int_{a_1}^{a_2} f(y)\,dy \approx \frac{\delta y}{6}\{f(a_1) + 4f(a_1 + \frac{1}{2}\delta y) + 2f(a_1 + \delta y) + 4f(a_1 + \frac{3}{2}\delta y)$$

$$+ 2f(a_1 + 2\delta y) + \cdots + 2f(a_2 - \delta y) + 4f(a_2 - \frac{1}{2}\delta y) + f(a_2)\}. \qquad (10.11)$$

It is straightforward to show that for smooth functions the error term associated with this method decreases at a rate of order $(\delta y)^4$, i.e. a doubling of the number of steps reduces the error by a factor of sixteen.

Trapezoidal rule

The trapezoidal rule is an even simpler quadrature method but is slower to converge, at a rate of $(\delta y)^2$. It is included here because the procedure can save computational time when pricing options for which the overall valuation technique puts

other, more stringent, limitations on convergence, thus making the use of more accurate quadrature schemes superfluous.

$$\int_{a_1}^{a_2} f(y)\,dy \approx \frac{\delta y}{2}\{f(a_1) + 2f(a_1+\delta y) + 2f(a_1+2\delta y)\cdots + 2f(a_2-\delta y) + f(a_2)\}. \tag{10.12}$$

The trapezoidal rule will find a reasonably accurate value more quickly than will Simpson's rule and so when speed is required rather than accuracy, the trapezoidal rule is sometimes more appropriate.

10.0.4 Method and general application

In order to demonstrate how the quadrature component is incorporated into the QUAD method, this section first considers the simple case of a vanilla call option. Here there is only one observation time, at maturity, and calculations are complete in a single time step. The closed form solution for this option is then used to verify the accuracy of the quadrature scheme. More interesting, challenging and complex problems will follow later.

Illustration using vanilla calls

In the case of a vanilla call valued now, time t, the payoff at maturity time, $t+\Delta t$, is $\max(S_{t+\Delta t} - X, 0)$. The integrand, Eq. (10.10) becomes

$$f(x,y) = B(x,y) \times X\max(e^y - 1, 0). \tag{10.13}$$

A numerical scheme is then implemented to evaluate the integral in Eq. (10.9) using Simpson's Rule, Eq. (10.11)

Simpson's rule converges with $(\delta y)^4$ but only when evaluating functions with continuous derivatives. The first derivative of the payoff to a call option is clearly discontinuous at the strike price (since for $S < X$, $\frac{\partial V}{\partial S} = 0$, for $S > X$, $\frac{\partial V}{\partial S} = 1$). In order remove this 'non-linearity error' and to conserve the overall accuracy of the quadrature scheme it is necessary to split the integral into two parts, with the boundary precisely at $S = X$ (or $y = 0$). The payoff function in the range $-\infty < S < X$ is continuous, as is the function in the range $X < S < \infty$. In fact,

for a call option, since the value at maturity in the range $-\infty < S \leq X$ is equal to zero, there is no contribution to the integral from this range (correspondingly, for a put there is no contribution from $X \leq S < \infty$). The range of integration in y in Eq. (10.9) is doubly infinite and so this must be truncated; however for $|y| \gg |x|$, the $B(x, y)$ term in Eq. (10.4) will tend to zero very quickly, and consequently leads to insignificant contribution to the overall integral outside a computationally quite modest range of y. Practically, a range corresponding to asset price movements, up or down, within ten standard deviations during a time step proved more than adequate.

To value the call option at time t with asset price S_t, the method is as follows (see also figure 2). The value of y at the discontinuity (here, the strike price) is defined to be b, which in the case of a vanilla option is clearly zero. The truncated range of integration is taken to be $[y_{\min}, y_{\max}]$, and the integers N^+ and N^- are the number of steps taken in the half ranges $[b, y_{\max}]$ and $[y_{\min}, b]$ respectively. Since the value of the function below b for a vanilla call option is zero, this region does not contribute to the option valuation. For a call option, then,

$$N^+ = \left[\frac{y_{\max} - b}{\delta y}\right] \qquad \text{and} \qquad N^- = 0, \tag{10.14}$$

where $[a]$ denotes the integer closest to a and δy is the step size in y. An implementation of Simpson's rule would have the steps, i, say, with $0 \leq i \leq N^+$ with $y = b + i\delta y$ and use values at these points and half way in between.

Using these new limits along with Eq. (10.4) the expression for the option price $V(x, t)$ is

$$V(x, t) \approx A(x) \int_0^{N^+ \delta y} f(x, y)\, dy \tag{10.15}$$

which, on implementing Simpson's rule, Eq. (10.11), is

$$\approx \frac{A(x)\delta y}{6}\left(f(x,0)+4f(x,\frac{1}{2}\delta y)+\left(\sum_{i=1}^{N^+-1} 2f(x,i\delta y)+4f(x,(i+\frac{1}{2})\delta y)\right)+f(x,N^+\delta y)\right) \tag{10.16}$$

and, by the definition of $f(x, y)$ in Eqs. (10.9) and (10.13), this is

$$\approx \frac{A(x)\delta y}{6}\Bigg(B(x,0)V(0,t+\Delta t) + 4B(x,\frac{1}{2}\delta y)V(\frac{1}{2}\delta y, t+\Delta t) + \ldots$$

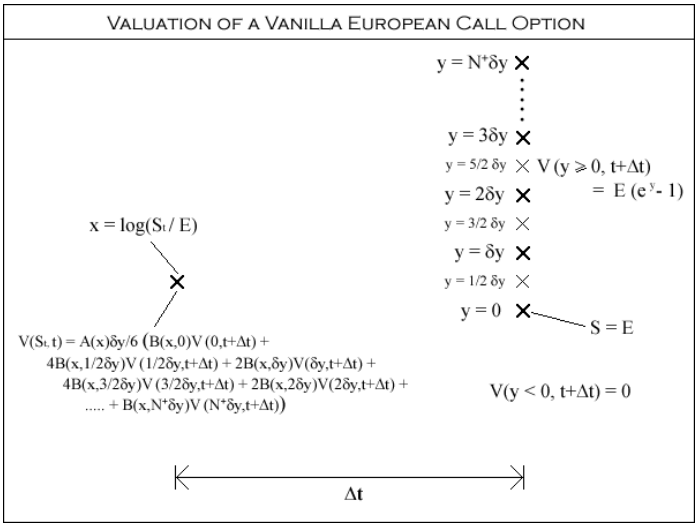

Figure 10.2: A demonstration of QUAD used to evaluate a European call option ($V(S_t, t)$). y is the transformed value of the underlying, S, δy is the step size, X is the strike price, Δt is the time to maturity and functions $A(x)$ and $B(x, y)$ are as described in Eqs. 10.5 and 10.6 respectively.

$$+ B(x, N^+\delta y)V(N^+\delta y, t + \Delta t)\Big). \tag{10.17}$$

This calculation is no more complicated than calculating the option value using a lattice method, however in QUAD the contribution at the previous timestep comes from many nodes representing different levels of the underlying asset not just two or three.

Figure 10.2 depicts the valuation process. A range of y values at maturity is chosen, increasing in steps of δy from $y = 0$ (the discontinuity) where the asset price, S, is equal to the exercise price, X see Eq. (10.3). The option value at time t is then found by integrating Eq. (10.4) using Simpson's rule via Eq. (10.17). Since a plain European call is being used as a simple example, valuation is achieved in a single time step and, in this case, is directly comparable with valuation using the known analytical solution commonly available in finance texts, such as Hull (2000).

The valuation can be quickly repeated for other possible current asset prices (transformed as x in Eq. (10.2)). In more complex option valuations these values

one step back in time from maturity to another discontinuity become, in their turn, the basis for further integration backwards in time. This is reminiscent of backward valuation using a tree or finite difference lattice but with the important difference that large time steps are used, set by the period between discontinuities.

Table 10.1 shows the results of applying this method to the pricing of a European call option using Simpson's rule. To reduce the distribution error still further, a simple Richardson-type extrapolation procedure was adopted. This was undertaken by performing calculations with two stepsizes δy_1 and δy_2, producing option values V_1 and V_2, respectively. An improved value V_{ext} may be obtained through extrapolation, viz.

$$V_{\text{ext}} = \frac{(\delta y_1)^d V_2 - (\delta y_2)^d V_1}{(\delta y_1)^d - (\delta y_2)^d}, \tag{10.18}$$

where $(\delta y)^d$ is the rate at which the error decreases for the unextrapolated data. Assuming that the position of the discontinuity has been accurately gauged, then for Simpson's rule $d = 4$.

Absolute RMS errors have been calculated using sixteen different options, both with and without the extrapolation described by Eq. (10.18) (denoted by QUAD and QUAD_{ext} respectively). Results are benchmarked against those from a trinomial method (denoted by TRIN). Timings were obtained using a Pentium III 550 MHz computer and the NAG Fortran 95 v4.0 optimized compiler, and indicate excellent performance of the present scheme when compared with the standard trinomial tree method. However, a more demanding test of the scheme is its application to more complex options for which no analytic, closed form solutions are known; these are considered next.

10.0.5 Application to multiply-observed options

A discrete path-dependent option is one where the price of the underlying asset before maturity is important only at discrete points in time, referred to here as observation times. The valuation of the option in between these times is described by the Black-Scholes partial differential equation Eq. (10.1).

ABSOLUTE ROOT MEAN SQUARED ERRORS				
N	TRIN (time/seconds)	K	QUAD_{ext} (time/seconds)	QUAD (time/seconds)
European call options:				
100	0.013879152 (0.00)	10	0.000008539 (0.00)	0.000246595 (0.00)
200	0.010018265 (0.01)	20	0.000000087 (0.00)	0.000015034 (0.00)
500	0.001833449 (0.03)	30	0.000000007 (0.00)	0.000002956 (0.00)
1000	0.000600399 (0.14)	40	0.000000001 (0.00)	0.000000934 (0.00)
2000	0.000405737 (0.56)	50	0.000000000 (0.00)	0.000000382 (0.00)

Table 10.1: Parameters are as follows: $S = 100, X = 95/105, \sigma = 0.2/0.4$, $r = 0.06/0.2, \Delta t = 0.5/1, D_c = 0$. TRIN is the basic trinomial model described in Hull (2000) with N steps in the lattice. QUAD, QUAD_{ext} are results using Simpson's rule, the latter extrapolated. K is such that $\delta y = \sqrt{\Delta t}/K$.

Consider a path-dependent option observed M times during its lifetime. Conceptually this is divided for valuation purposes into M separate options. These options, denoted V_m, have maturities T_m where $m = 1, 2, \ldots, M$. Let Δt_m be equal to $T_m - T_{m-1}$. Given the current time, t, the maturity of the complete option is at time T_M given by

$$T_M = t + \sum_{m=1}^{M} \Delta t_m. \tag{10.19}$$

Moving backwards from this maturity, whenever an observation time is encountered , Eq. (10.4), is evaluated for the period between that time and the next observation later in time. For example, with a Bermudan option the process could begin with integration of Eq. (10.4) covering the time period from the final possibility of early exercise until the maturity of the option.

At each observation time T_m the option is priced for a full range of x. These x values then become the y values used in the integration to find the x values at the previous time step, T_{m-1} and so on until the complete option is valued. The values of $y_{\max}$ and $y_{\min}$ for a truncated range of integration will change at each T_m and as a consequence will be denoted by $y_{\max_m}$ and $y_{\min_m}$ respectively. The corresponding integer values of N^+ and N^- for the steps in the integration will be termed N_m^+ and N_m^- respectively.

The probability of an asset following geometric Brownian motion moving more than 10 standard deviations within a time period is so small as to be insignificant. For this reason, curtailment of the range is possible such that, if $T = T_m - t$ is the time before maturity, then it is suggested that $y_{\max_m} = x_0 + q$ and $y_{\min_m} = x_0 - q$ are sufficient where

$$q = 10\sigma\sqrt{T}. \tag{10.20}$$

This assumes that $(r - D_c)$ is not unrealistically large, although adjustment for such values would be routine. For the same reason, it is not actually necessary to integrate over the entire range of y for each value of x at every time step, but instead, letting

$$q^* = 10\sigma\sqrt{\Delta t_m}, \tag{10.21}$$

the integral for each individual x only needs to extend over the range $[x-q^*, x+q^*]$. A point of discontinuity in V_m will be denoted by b_m. If the integer values of i at time T_m corresponding to this range are i^+ and i^-, where

$$i^+ = \left[\frac{x - b_m + q^*}{\delta y}\right], \tag{10.22}$$

and

$$i^- = \left[\frac{x - b_m - q^*}{\delta y}\right], \tag{10.23}$$

then Simpson's rule should be employed for i between $\max(i^-, N^-)$ and $\min(i^+, N^+)$. If the δy chosen is too large, then results will clearly be inaccurate; fortuitously, it is generally very obvious when this is the case. As a rough guideline, δy should always be smaller than $\frac{\sqrt{\Delta t_m}}{4}$.

The location of discontinuities in the payoff function will be known a priori for some classes of options. For example, for a vanilla call or put option with exercise price, X, a discontinuity occurs at $y = 0$; for a discrete barrier option, with a barrier position B_m at time T_m, a discontinuity occurs where $y = \log(\frac{B_m}{X})$. For other classes of option the position of the discontinuity must be calculated at every observation time. For a Bermudan put option, for example, the location of the discontinuity, b_m, at time T_m is where $X - S = P_{m+1}(b_m, T_m)$. Taking this class of option as a prototype, the location is determined as follows. It is convenient

to define a function, $g(x)$ say, which is zero at the desired value of x, b_m. For a Bermudan put is

$$g(x) = X(1 - e^x) - P_m(x, T_m). \tag{10.24}$$

The equation $g(b_m) = 0$ may then be solved using Newton-Raphson iteration, which is generally very quick to converge (rarely are more than ten iterations necessary to obtain machine precision). In brief, initial values for b_m and Δb_m are guessed (e.g 0 and -0.01, respectively) and a new value for b_m is then:

$$b_m^* = b_m + \Delta b_m. \tag{10.25}$$

$g(b_m)$ and $g(b_m^*)$ must then be calculated, using Simpson's rule for example, and then a new Δb_m is given by

$$\Delta b_m^* = \frac{-\Delta b_m g(b_m^*)}{g(b_m^*) - g(b_m)}. \tag{10.26}$$

The iterative process is repeated until $g(b_m)$ is sufficiently small, with the programming caveat that iterations cease if the discontinuity is out of the range of integration, $[y_{\min_m}, y_{\max_m}]$ and hence, cannot be found.

Working back through time, from known final conditions, the first individual option $V_M(x, T_{M-1})$ can be priced for the entire range of underlying prices, in the manner described previously. The condition at the observation time (e.g. an early exercise condition) is imposed on these values and then these adjusted values provide the new final conditions for the next option, $V_{M-1}(y, T_{M-1})$. The next option is priced using these final conditions and the process is repeated until the value is found for $V_1(x, T_0)$. Values for $V_m(x_i, T_{m-1})$ must be calculated for all values of x_i, where

$$x_i = b_m + i\delta y, \tag{10.27}$$

and also, in the case of Simpson's rule, at the points half way between. The i will run from N_m^- to N_m^+, with one of these sometimes equal to zero (as in figure 10.2, for a European vanilla call option). For the final option, V_1, if the option value is just required at a single asset value, it is only necessary to calculate the value of V corresponding to S_t, i.e. $x = \log(S_t/X)$. Figure 10.3 shows how QUAD is used to price a discrete down-and-out call option with one barrier, B_1 at T_1.

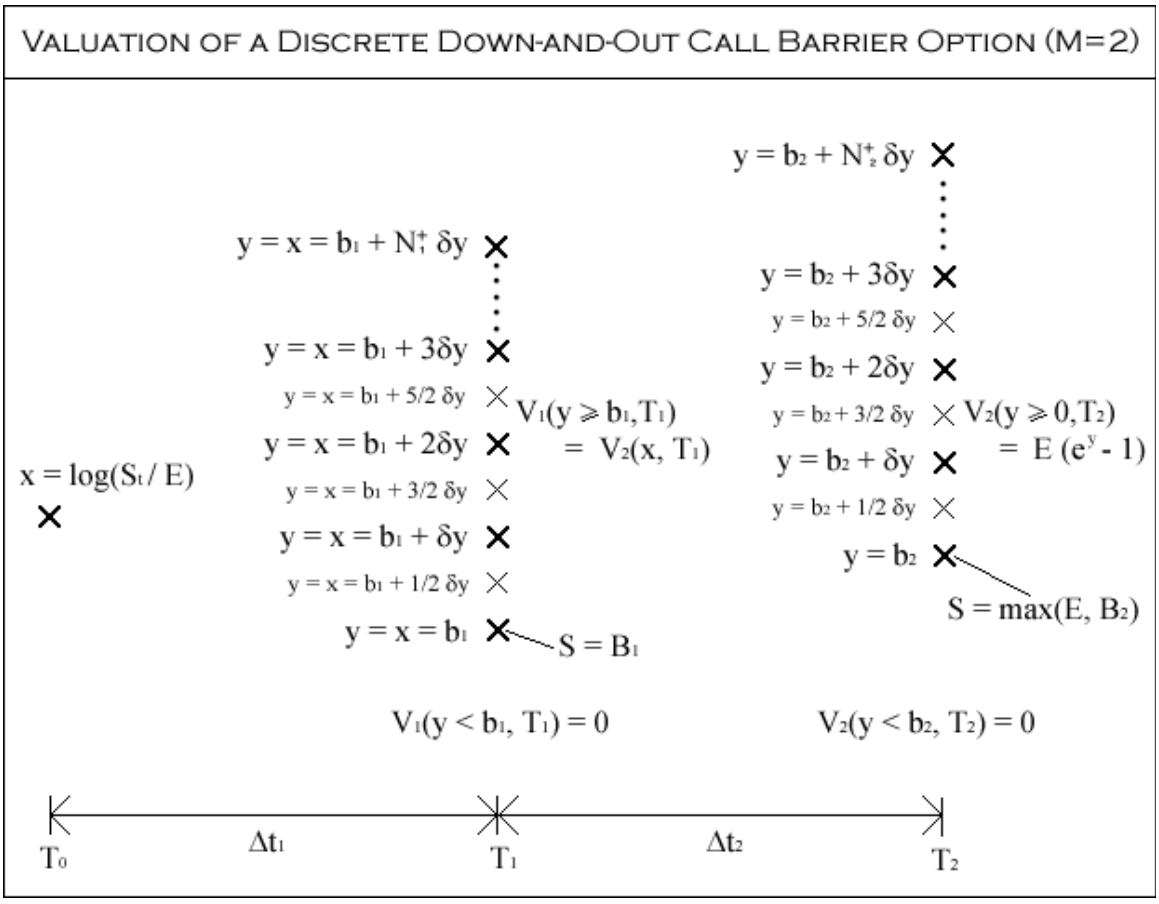

Figure 10.3: A diagram depicting how QUAD is used to evaluate a discrete down and out call option with barrier at $S = B_1$ at time T_1. x and y are the transformed values of the underlying S, b_1 is the transformed value of B_1, δy is the step size, the time to maturity is T_2. The option value at T_0 is calculated using an implementation of Simpson's rule on the $V_1(y \geq b_1, T_1)$ values.

High frequency observations and continuous American features

Extrapolation is not only useful in removing 'distribution error' caused by the finiteness of the quadrature stepsize δy (using Eq. (10.18)) but can also be used to value options with more observations, or even with continuous, American style exercise. For continuous observation, the formula Eq. (10.18) may be utilized, but with δy replaced by T/M, where M is the number of observations. Although QUAD is tailored to be exceptionally powerful in valuing options with discrete observations, for those with continuous observation, its performance is at least comparable to all standard techniques, and, if the option is a particularly complex one with American features, then QUAD may prove very useful. To value options with many observations, where computing time is an issue (for example, three-dimensional lookbacks), extrapolation to many multiple observations (taken at moments equally distributed in time) can be performed using:

$$V_{M_3} = \frac{M_1^d(M_2^d - M_3^d)V_{M_1} + M_2^d(M_3^d - M_1^d)V_{M_2}}{M_3^d(M_2^d - M_1^d)}, \tag{10.28}$$

where M_1, $M_2 < M_3$ are the number of observations and V_M is the respective option value. Note that there are certain cases (e.g. random moving barriers) where such extrapolation is not applicable. A value of d of either $\frac{1}{2}$ or 1 should be used depending on the particular problem. Discretely monitored American put options (Bermudans), for example, converge to the continuous solution at the rate of $\frac{1}{M}$, where M is the number of observations, hence $d = 1$; discrete barrier options converge with $\frac{1}{\sqrt{M}}$ and so $d = \frac{1}{2}$. For a given type option it is straightforward to gauge the appropriate value for d simply by running a few trial calculations.

10.0.6 Bermudan put options

A Bermudan put option is one which has an early exercise feature, similar to an American put, but only at certain prescribed dates. For a Bermudan put option the value of P_m (replacing V by P for a put option) at time T_m is

$$\max(P_{m+1}(y = x, T_m), X - S), \tag{10.29}$$

where $P_{M+1} = \max(X - S, 0)$.

There will be a discontinuity in the second derivative of the option at the point where $P_{m+1}(T_m)$ is equal to $(X - S)$. Above this point the value of the option is given by $P_{m+1}(T_m)$, below it is equal to $(X - S)$. Therefore the integration is split into two components to avoid any non-linearity error.

For the first option period evaluated, b_M is the point where $X = S$, and so $b_M = 0$. P_{M+1} above this point is equal to zero. Newton-Raphson may be used to find subsequent values of b_m, as described in Section 3.2.

Above b_m, the value of $P_m(y \geq b_m, T_m)$ is $P_{m+1}(y = x, T_m)$, below the boundary the value is simply the payoff, namely $X - S$, and so $P_m(y \leq b_m, T_m) = X(1 - e^y)$. It is possible to continue using quadrature as before using Eq. (10.4)

$$P_m(x, T_{m-1}) \approx A(x) \int_{b_m}^{y_{\max_m}} B(x, y) P_m(y \geq b_m, T_m)\, dy$$

$$+ A(x) \int_{y_{\min_m}}^{b_m} B(x, y) \times X(1 - e^y)\, dy \; = I_1 + I_2. \tag{10.30}$$

However I_2 has a simple analytic form, related to the Black-Scholes solution for a put option within the bound $(-\infty, b_m]$. In this region, in order to save computational time, the procedure is then to use this analytic (Black-Scholes) form for $I_2(x)$, namely

$$I_2(x) = Xe^{-r\Delta t_m} N(-d_2) - Xe^{x - D_c \Delta t_m} N(-d_1), \tag{10.31}$$

where $N(.)$ is the cumulative probability distribution for a function that is normally distributed with a mean of zero and standard deviation of 1, and d_1 and d_2 are given by:

$$d_1 = \frac{x - b_m + (r - D_c + \frac{\sigma^2}{2})(\Delta t_m)}{\sigma\sqrt{\Delta t_m}},$$

$$d_2 = d_1 - \sigma\sqrt{\Delta t_m}. \tag{10.32}$$

10.0.7 Results

The accuracy and efficiency of QUAD are now compared with more familiar numerical techniques for valuing the different types of options, described in the previous section. Tables 10.2 through to 10.5 show absolute root mean squared errors

for a set of eight options. Results (QUAD) were generally extrapolated once (labelled QUAD_{ext}) using Eq. (10.18) to further enhance accuracy. In the case of calculations performed using trapezoidal quadrature (in particular lookback options in three dimensions), extrapolation using Eq. (10.18) was carried out twice ($\text{QUAD}_{\text{ext2}}$), first with $d = 2$, and then the resulting values (which may be regarded as having an error of $O((\delta y)^4)$) were the subject of a further application of Eq. (10.18) with $d = 4$. Choosing $\delta y = \sqrt{\Delta t_m}/K$, extrapolation was performed with values obtained from K and $K-2$, except in the case of lookback puts where K and $K-1$ were used. For the lookback call options it was performed twice using $K, K-1$, and $K-2$. Also, D_c is considered to be 0 for all options valued here.

For multiple compounds, Bermudan put, moving barrier, and American call options, the results of the present method are compared with the basic trinomial model (Hull, 2000) denoted by TRIN. When valuing stationary discrete barrier options Ritchken's (1995) approach (TRIN) is used but adapted, as suggested by Cheuk and Vorst (1996) in a straightforward manner to treat discrete rather than continuous barriers. To value lookback options with a similarity reduction, Cheuk and Vorst's (1997) two-dimensional binomial method (BINC), incorporating the placing of extra time steps between observations, (and adapted for $\overline{A}_M$ different from S_t) is used for comparison. Finally, for the barrier lookback option, the benchmark is a basic three-dimensional trinomial model adapted for multiple steps between observations (TRIN3). For each case the exact solutions were obtained by using the current scheme when the δy was small enough for the price to have converged to the given degree of accuracy ($\sim 1 \times 10^{-10}$). As an indication of the efficiency of the QUAD, average computational times are also displayed. As before, timings were taken on a Pentium III 550 MHz system, using the NAG Fortran 95 v4.0 compiler with optimization.

Although the basic trinomial tree approach was used as a benchmark for moving barrier options, improved results can be obtained using the AMM method of Ahn, Figlewski, and Gao (1999), though at a cost of considerably more complex programming. For comparison, AMM8 with eighty time steps between observa-

tions gives absolute RMS of 0.00034 and takes an average time of 8.65 seconds whereas QUAD_{ext} achieves better accuracy in a mere 0.28 of a second.

The results are shown in Table 10.5 and are excellent; for the two-dimensional models results in a given time are consistently orders of magnitude better than the benchmarks for all options considered. In the case of fixed-strike lookback discrete barrier options penny accuracy is achieved in 1.16 seconds whereas the trinomial method is, on average, 40 cents out after 23 minutes. The superiority of QUAD over present methods is clear.

10.0.8 Overview

Numerical methods based on quadrature are a fast and extremely accurate means for option pricing. In effect, between observations it is analogous to a multinomial tree method, but uses only a single time step. The assumption that in between observations the option behaves as a European option enables the problem to be broken down into a series of integrals which, coupled with numerical quadrature, are used to value the option over the whole required period. The method benefits from the simple and exact placement of 'nodes' on boundaries/discontinuities (whether stationary or moving), thus removing any non-linearity error, a problem that plagues other numerical techniques; indeed, much of the success of the method can be attributed to this ability; the results are extremely accurate. As an example of its effectiveness, an option that would take hours, if not days, to price with penny accuracy using a trinomial method (such as the lookback barrier in Table 10.5) can now be priced correct to four decimal places in just seconds. The simplicity and universality for all discretely monitored options shows this method to be considerably superior to the traditional lattice or finite-difference methods, making it a powerful addition to the mathematical financier's toolkit.

ABSOLUTE ROOT MEAN SQUARED ERRORS				
N between obs.	TRIN (time/seconds)	K	QUAD_{ext} (time/seconds)	QUAD (time/seconds)
Discrete down and out barrier call options:				
40	0.009969399 (1.68)	6	0.000121107 (0.41)	0.000460985 (0.28)
50	0.009670104 (2.59)	8	0.000008423 (0.75)	0.000140402 (0.47)
60	0.008317414 (3.86)	10	0.000001673 (1.19)	0.000056567 (0.72)
70	0.008159162 (5.93)	12	0.000000481 (1.74)	0.000027042 (1.02)
80	0.008055604 (9.23)	20	0.000000035 (5.02)	0.000003461 (2.77)
Moving discrete down and out barrier call options:				
40	0.147934594 (1.84)	6	0.000143546 (0.41)	0.000283021 (0.28)
50	0.091196513 (3.11)	8	0.000012456 (0.75)	0.000082115 (0.47)
60	0.074112820 (4.06)	10	0.000002472 (1.19)	0.000032372 (0.72)
70	0.040418332 (6.89)	12	0.000000706 (1.74)	0.000015296 (1.03)
80	0.062995092 (8.58)	20	0.000000025 (5.04)	0.000001925 (2.78)
Multiply-compounded call options:				
50	0.007175522 (0.01)	6	0.000183337 (0.01)	0.000485888 (0.01)
100	0.002157462 (0.04)	8	0.000014998 (0.01)	0.000144786 (0.01)
200	0.001524073 (0.14)	10	0.000002991 (0.02)	0.000057748 (0.01)
300	0.000782219 (0.31)	12	0.000000855 (0.03)	0.000027458 (0.02)
500	0.000356063 (0.87)	20	0.000000030 (0.08)	0.000003487 (0.05)

Table 10.2: Parameters in each case are as follows. For discrete barriers: $M = 6/125, B = 90/99, S = 100, X = 95/105, \sigma = 0.2/0.4, r = 0.06$, $T_M - t = 0.5$. For moving barriers: $M = 5/125; B_4, B_9, B_{14}, \ldots, B_{124} = 94$; $B_3, B_8, \ldots, B_{123} = 93; B_2, \ldots, B_{122} = 92$; $B_1, \ldots, B_{121} = 91$; $B_0, \ldots, B_{125} = 90; S = 100, X = 95/105, \sigma = 0.2/0.4, r = 0.06, T_M - t = 0.5/1$. For multiply compounded call options: $M = 4/6, S = 100, X_M = 95/105$, $X_{1,2,\ldots,M-1} = 2/3, \sigma = 0.2/0.4, r = 0.06, T_M - t = 1$.

ABSOLUTE ROOT MEAN SQUARED ERRORS				
N between obs.	TRIN (time/seconds)	K	QUAD_{ext} (time/seconds)	QUAD (time/seconds)
Bermudan put options:				
10	0.017733020 (0.15)	6	0.000042132 (1.27)	0.000220897 (0.84)
25	0.002844881 (0.84)	8	0.000004260 (2.29)	0.000067584 (1.42)
50	0.001249349 (4.18)	10	0.000000900 (3.59)	0.000027261 (2.16)
75	0.001662824 (8.04)	12	0.000000263 (5.32)	0.000013039 (3.07)
100	0.000841146 (22.85)	20	0.000000010 (15.13)	0.000001670 (8.20)
American call options with changing strike price:				
100	0.001232661 (0.13)	6	0.000024459 (0.15)	0.000073288 (0.09)
200	0.000720084 (0.51)	8	0.000001241 (0.25)	0.000023066 (0.15)
300	0.000634989 (1.14)	10	0.000000232 (0.38)	0.000009425 (0.23)
400	0.000311496 (2.07)	12	0.000000065 (0.54)	0.000004539 (0.32)
500	0.000280925 (3.15)	20	0.000000002 (1.50)	0.000000587 (0.83)
American call options with dividend payments:				
100	0.005009045 (0.12)	6	0.000086041 (0.03)	0.000261641 (0.02)
200	0.003474849 (0.51)	8	0.000006501 (0.05)	0.000079463 (0.03)
300	0.000561382 (1.13)	10	0.000001284 (0.06)	0.000031983 (0.04)
400	0.001450507 (2.14)	12	0.000000366 (0.08)	0.000015283 (0.05)
500	0.000573460 (3.22)	20	0.000000013 (0.20)	0.000001955 (0.11)

Table 10.3: Parameters in each case are as follows. For Bermudan puts: $M = 6/125, S = 100, X = 95/105, \sigma = 0.2/0.4, r = 0.06, T_M - t = 0.5/1$. For American calls with changing strike price: $M = 6/12, X_M = 95/105$, $X_m = X_{m+1} - \Delta X$ (where $\Delta X = 1/0.5$ for $M = 6/12$ respectively), $\sigma = 0.2/0.4, r = 0.06, T_M - t = 0.5/1$. For American calls with dividend payments: $M = 3/6, S = 100, X = 95, D = 2/3, \sigma = 0.2/0.4, r = 0.06$, $T_M - t = 1.5/3$.

ABSOLUTE ROOT MEAN SQUARED ERRORS				
N between obs.	BINV (time/seconds)	K	QUAD_{ext} (time/seconds)	QUAD (time/seconds)
Lookback put options with payoff $\overline{A} - S$:				
40	0.067937970 (0.97)	2	0.000038073 (1.78)	0.000205447 (1.08)
60	0.088310529 (2.18)	3	0.000001070 (2.81)	0.000039762 (1.73)
80	0.061800587 (3.86)	4	0.000000134 (4.38)	0.000027261 (2.65)
100	0.041746916 (6.21)	5	0.000000029 (6.42)	0.000005101 (3.77)
120	0.056115193 (8.70)	6	0.000000009 (8.98)	0.000002456 (5.21)

Table 10.4: Parameters in each case are as follows. For lookback puts: $M = 6/125, S = 100, \overline{A_0} = 100/105, X = 95, \sigma = 0.2/0.4, r = 0.06$, $T_M - t = 0.5/1$. For lookback calls with barrier: $M = 6/26, B = 90/100$, $S = 100, \overline{A_0} = 100/105, X = 95, \sigma = 0.2/0.4, r = 0.06, T_M - t = 0.5$.

N between obs.	TRIN3 (time/seconds)	Approx. K	$\text{QUAD}_{\text{ext2}}$ (time/seconds)	QUAD (time/seconds)
Fixed strike lookback call options with barrier:				
10	1.071426815 (1.92)	10	0.007466455 (1.16)	0.119313542 (0.68)
20	0.798811819 (15.39)	14	0.000458519 (2.78)	0.061134855 (1.52)
30	0.712570741 (63.75)	16	0.000182039 (5.41)	0.039947736 (2.82)
60	0.482999391 (445.80)	20	0.000078422 (9.55)	0.026971794 (4.85)
90	0.412661235 (1427.48)	28	0.000016221 (31.00)	0.013740692 (15.33)

Table 10.5: The M barriers or observations are equally spaced throughout the time period. TRIN indicates the relevant trinomial model for the option; BINC is Cheuk and Vorst's (1997) binomial model; TRIN3 is a three-dimensional trinomial model for lookbacks. QUAD are the results obtained using the given method with Simpson's rule except in the case of the lookback with barrier which uses the trapezium rule. QUAD_{ext} and $\text{QUAD}_{\text{ext2}}$ are the results extrapolated once and twice respectively. In the case of lookback barrier options $q = 7.5\sigma\sqrt{T}$ and $q^* = 7.5\sigma\sqrt{\Delta t_m}$. N is the number of steps in between observations in the trinomial tree (i.e. the total number of time steps is $N \times M$), and K is such that $\delta y = \sqrt{T_{\Delta t_m}/K}$.'Exact' solutions were obtained using QUAD_{ext} with $K > 150$.

Lecture 11

Numerical methods in general

We have seen the four fundamental numerical methods used in CF: Monte Carlo, Binomial lattices, Finite difference and Quadrature methods. Each one will be useful for different types of derivative pricing problems and it is important that you choose the right method for the right problem.

The main issues are:

- Accuracy/Speed/Ease of programming
- Hedging estimation
- Different processes
- Exotic features
- More than one Brownian motion

11.1 Accuracy

For a basic option pricing problem in one underlying asset the most accurate methods are the Crank-Nicolson finite difference method and QUAD. However, for these basic options either analytic solutions are available or any of the other methods will provide perfectly acceptable levels of accuracy given the available computing power. The real test comes with different types of problems.

11.2 Hedging estimation

We have not really discussed calculation of the 'greeks' or implied volatility but as one would expect these are easy to estimate using binomial lattices and especially finite difference methods as the greeks involve estimating derivatives. It is less easy with Monte Carlo methods, as you typically have to perturb the parameter in question and estimate the derivative from the answer. To avoid error from random numbers you also have to use the same random numbers. This can still cause problems for options with discontinuities - like barrier options.

11.3 Different processes

If we assume a process other than GBM then we things become a little more difficult. Such problems often arrive when pricing interest rate derivatives where the interest rate follows a mean reverting process (e.g. Vasicek; CIR etc.) Here it is simple to adapt the Monte Carlo method to deal with this as we typically have an SDE formulation and we can use the Euler approximation. In certain cases it is also possible to adapt the binomial tree but here we require the tree to match the new distribution. More problematic are finite difference methods where you need to derive an entire new PDE (not actually possible in the more sophisticated interest rate model, HJM).

11.4 Exotic features

Exotic features are often path dependent features, such as payoffs as a function of the average or maximum option value. They may also be knock-out features such as barriers. Typically path dependent options require another dimension for lattice and finite difference methods and some level of programming difficulty. However, the Monte Carlo method is easy to adapt for path dependent options, although you now need to include more time steps. Here there is thus a trade off between the accurate, slow lattice methods and the inaccurate fast Monte Carlo method. By comparison if we consider the convergence as a function of the computational effort

(or work) then we typically see Monte Carlo being $O(w^{-1/4})$ as the computation is time steps x sample paths. For one underlying and one path dependent feature a lattice is $O(w^{-1/3})$ and Crank-Nicolson is $O(w^{-2/3})$.

The problem becomes more pronounced when there are early exercise features too as in this case it is most inefficient to use Monte Carlo as the convergence now is uncertain and it is far more difficult to program. However, if the number of underlying assets are also increased then this becomes a very difficult problem indeed as the lattice/f-d methods will struggle with the dimensionality but the Monte-Carlo method will struggle with the early exercise.

11.5 More than one Brownian motion

This could be either more than one underlying asset, stochastic volatility or interest rates or a multifactor interest rate model. In each of these if you have d Brownian motions the effort convergence as a function of the computational effort w is as follows: Monte-Carlo: $O(w^{-1/2})$, Lattice and Explicit finite-difference: $O(w^{-1/d+1})$ and Crank-Nicolson $O(w^{-2/d+1})$. Thus for $d > 3$, Monte Carlo methods should be preferred. Again there is a large problem when there are also American features as it is very difficult to estimate the convergence from these Monte-Carlo schemes and it may well be that for $d < 6$, you could attempt to use finite-difference methods or a binomial lattice.

Quadature methods:

These are very accurate, often with $1/N^8$ convergence, this means that even as you add underlying assets the convergence as a function of work is only $O(w^{-8/d+1})$. These methods will still have memory requirements for large d, but require less nodes due to their excellent accuracy. They also cope well with early exercise features. The only problem occurs when there are non-GBM underlying processes as the method relies on having analytic transition density functions. However, there are functional approximations that may overcome these difficulties... See Andricopoulos et al., 2003

11.6 Overview

We have seen the final adaptation to the finite difference method: body-fitted coordinates that enable you to even remove non-linearity error for American options. There was then a brief discussion about the advantages of each of the numerical methods