IMPLEMENTING NUMERICAL OPTION PRICING MODELS

by

Simon Benninga
Raz Steinmetz
John Stroughair

11-93

RODNEY L. WHITE CENTER FOR FINANCIAL RESEARCH
The Wharton School
University of Pennsylvania
Philadelphia, PA 19104-6367

IMPLEMENTING NUMERICAL OPTION PRICING MODELS

by

Simon Benninga
Raz Steinmetz
John Stroughair

May 10, 1993

Address all communications to:
Simon Benninga
Finance Department, Wharton School
University of Pennsylvania
Philadelphia, PA 19104
Telephone: 215-898-9466
E-Mail: BENNINGA@WILMA.WHARTON.UPENN.EDU

Abstract: We develop routines in *Mathematica* for pricing various European and American options using the binary option model and Monte Carlo methods. As might be expected, *Mathematica* permits parsimonious programming of the option pricing expressions.

Simon Benninga is on the faculty of the School of Business at the Hebrew University of Jerusalem, Israel; this paper was written while he was a Visiting Professor at the Wharton School. Raz Steinmetz and John Stroughair are graduate students at the Wharton School of the University of Pennsylvania.

IMPLEMENTING BINOMIAL AND BINARY OPTION PRICING MODELS

1. Introduction

In a previous paper in the *Mathematica Journal*, Miller (1990) showed how the Black-Scholes option pricing model could be implemented in *Mathematica*. While the Black-Scholes model gives a closed-form solution for pricing European options, there are many other options which cannot be priced by this model. Such options include American options and options whose terminal payoffs are dependent on the path followed by the stock price.

In this paper we show how *Mathematica* may be used to price such options, and we discuss some of the convergence and timing properties of the *Mathematica* pricing. The structure of the paper is as follows: In Section 2 we review definitions and some classic option pricing results. Section 3 discusses the mechanics of the binary and binomial option pricing model and gives a simple, four-state example. In Section 4 we discuss *Mathematica* programs for pricing various kinds of options. In section 5 we discuss timing and convergence issues. Section 6 implements the Monte Carlo method and discusses its properties.

2. A review of some option pricing definitions and results[1]

A *call option on a stock* is a security which gives the holder the right but not the obligation to buy one share of the stock on or before date T for a price K, the strike price. A *European call option* gives the holder the right to purchase the share of stock *only* on the terminal date T, whereas an *American call option* gives the holder the right to purchase the share at any date up to and including the terminal date T. While it would seem that the *early exercise* feature of an American call option should give it more value than a European option, a well-known theorem states that early exercise of an American call option is not optimal if the stock on which the option is written does not pay a dividend before the option's exercise date T. It follows, therefore, that:

> If the stock on which the option is written does not pay a dividend before the maturity date T, then an American call option written on the stock has the same value as a European option.[2]

A *put option on a stock* is a security which gives the holder the right to *sell* one share of the stock on or before date T for a price K. As for calls, one may distinguish between European put options, which do not allow early exercise, and American put options. The above result is no

---

[1] This review is extremely cursory. The interested reader is referred to a standard options textbook, such as Jarrow and Rudd (1983), Cox and Rubinstein (1985) or Hull (1993).

[2] For a proof, see Merton (1973).

longer true, however; it may be optimal to exercise an American put before expiration, even if the underlying stock pays no dividends, so that in general an American put option will be more valuable than the corresponding European put. This dichotomy between puts and calls has its roots in the fact that the stock price is bounded below by zero but is unbounded above.

In addition to the standard European and American options discussed above various other "exotic" options are traded. In this paper we consider two of these options: *lookback options* and *Asian options*. A lookback call gives the holder the right to purchase the underlying stock for the minimum price at which the stock traded during the life of the option and a lookback put the right to sell for the maximum price reached during the life of the option. For an Asian option, either the payoff or the strike price is related to the average price of the underlying asset. Both lookback and Asian options come in European and American versions. The algorithm presented can be easily adapted to price various other exotic options.

Throughout this paper we shall consider only options on stocks which do not pay dividends before the expiration date of the option.

In general, most of the models that price options assume a certain stochastic movement of the stock prices. The most common assumption is that the stock price at time T will be governed by the following stochastic diffusion process:

(1) $$dS = S\mu dt + S\sigma dz$$

where dS is the change in the stock price, S is the stock price, μ is the expected logarithmic return on the stock (which can be replaced by the risk free rate for option valuation, see below), σ is the standard deviation of the return on the stock, and z is the standard Wiener process.

Using results from stochastic calculus, and the fact that an option may be perfectly hedged, three important observations can be made:

a. Any contingent claim, that depend on the underlying asset price, satisfies the partial differential equation of:

(2) $$rS\frac{\partial F}{\partial S} + \frac{\sigma^2 S^2}{2}\frac{\partial^2 F}{\partial S^2} - \frac{\partial F}{\partial t} = r_f F$$

where F is the claim value, r is the continuously compounded interest rate and S, t, and σ are the same as in (1). One should note that there are many solution to the above equation, and that each solution depends on the boundary condition given. For example, a European call has boundary conditions:

(3) $$F(0,t) = 0, \quad F(S,T) = Max(0, S-K),$$

and a European put has boundary conditions:

(4) $$F(0,t) = 0, \quad F(S,T) = Max(0, K - S).$$

b. Since the option can be perfectly hedged, the value of the option is the same regardless of the risk aversion of investors, and it can thus be valued using the assumption that all investors are risk neutral. By making suitable substitutions into the stochastic process (1), risk neutrality allows us to express the value of the option today as the discounted expected value of the payoffs from the option using the formula:

(5) $$F = E\left\{e^{-rt} F_t\right\}.$$

c. Under the stochastic equation (1) the terminal stock price follows a lognormal distribution, and thus the logarithm of the stock final price follows normal distribution with the following parameters:

(6) $$\mathrm{Ln}(S_T) \sim \mathrm{N}\left[S_0(r - \sigma^2/2)t,\ \sigma\sqrt{t}\right]$$

In a well known paper, , Black and Scholes (1973) prove that an European call option price is given by:

(7) $$C = S_o N(d_1) - Ke^{-rT} N(d_2)$$

where S_0 is the stock price today, T is the time to maturity, N( ) is the cumulative standard normal distribution and d_1 and d_2 are defined by[3]:

(8) $$d_1 = \frac{\mathrm{Ln}(S_o / K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}$$

$$d_2 = d_1 - \sigma\sqrt{T}$$

Since for European options the *put-call parity theorem* holds:

(9) $$P = C - S_0 + Ke^{-rT}$$

it follows that the Black-Scholes theorem allows us to price both European puts and calls.

[3] Notice that evaluation of the Black and Scholes formula on a computer is one application of numerical integration of the normal distribution.

In practice, many existing options do not have closed-form solutions. In those cases we try to estimate the value of the option by numerically solving the differential equation. In this paper we illustrate two methods:

a. **Binary option pricing models**: These methods approximate the expected price of the option by using the lattice method, which simulates the Wiener process that the stock price follows. Binary models make strong use of the fact that options may be perfectly hedged by combinations of other securities. The prices of these securities (or derivatives of these prices) may then be used to price the option.

b. **Monte Carlo methods**: Here we simulate possible price paths the stock might take, and again, using the fact that options may be valued in risk neutral world we average the possible option outcomes. Only European options can be priced directly with Monte Carlo techniques, and they are useful as a check on the results from other techniques (although much less accurate). The Monte Carlo technique is also easily extended to handle various other distributions and multiple stochastic variables.

3. The binary (lattice) option pricing model

A wide variety of options cannot be priced analytically. An approach to pricing such options is the *binary option pricing model*, developed by Cox, Ross, Rubinstein (1979) and Rendleman and Bartter (1979).[4] The binary option pricing model assumes that at any point in time there are two possible future states of the world which define security returns. It follows that two basic securities may used to price all other financial assets in the economy. We refer to these basic securities as a *stock* and a *risk-free asset*. The stock's return over a short period of time is either *up* or *down*, the risk-free asset's return over the same period of time is r.[5] Given this binary uncertainty, all other risky assets may be priced by defining the spanning *state prices*. Assuming, with no loss in generality, that the initial stock price is S = 1, these prices, denoted by p and q are given by the solution to:

$$\begin{cases} p \cdot up + q \cdot down = 1 \\ p \cdot r + q \cdot r = 1 \end{cases} \tag{10}$$

It follows that

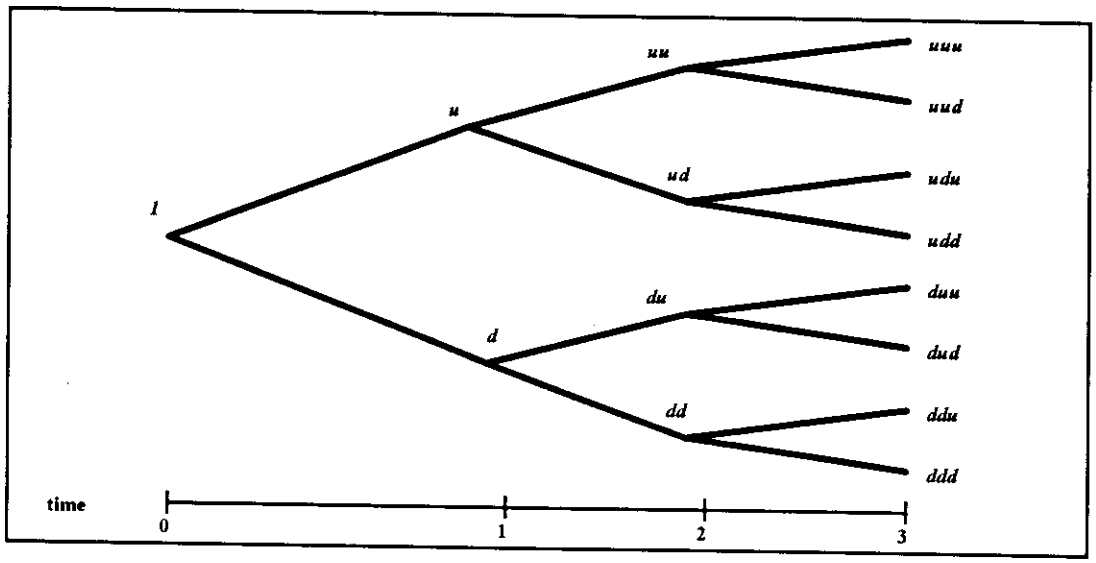
$$p = \frac{r-d}{r(u-d)}, \qquad q = \frac{u-r}{r(u-d)} \tag{11}$$

---

[4] In the option literature, this model is usually referred to as the *binomial option pricing model*. We use a more general terminology in order to price path dependent options. We will use the term binomial for the case when the binary tree of stock prices recombines.

[5] We use *return* to indicate one-plus-the percentage return. Thus, if the stock price at date t is p_t the stock price at date $t+1$ will be either $p_t \cdot up$ or $p_t \cdot down$ Similarly, we let r denote one plus the percentage risk-free rate of interest.

When an option's price is not path dependent, we may use the *binomial option pricing model* to price it. When it is path dependent, we use the *binary option pricing model*. The binary model is much more computationally intensive than the binomial option pricing. The figure below shows a four-date binary tree.



For clarity, we present a four-date example of both binary and binomial option pricing model (See Table 1). In Table 1 the stock price over a single period can change by $u = 1.04$ or $d = 0.98$; when the risk-free interest rate is r = 2% (per period), this gives *one-period* prices for "up" and "down" states of $p = 0.6536$ and $q = 0.3268$ respectively. The column labeled "State Prices" shows the value at time zero (today) of a future dollar (in any state $u^i d^j$ the state price is $p^i q^j$).

All of the options in Table 1 have an exercise price $K = 50$ and an initial stock price $S_0 = 50$. The value of any option is the dot product of the state prices for states $\{1, u, d, uu, \ldots, dddd\}$ and the vector of security payoffs in these states. Securities that show *NA* in some states means that those states are never reached because of early exercise of the option. Security prices in *italics* indicate that this is an early exercise value.

Security A is a European put, which is exercised only in states *ddd*, *ddu*, *dud*, *udd*, and consequently has value today of 0.115. Security B is an American put, which is exercised early in state *d*. The test for early exercise in this (or any other) state is that immediate exercise is more valuable than the value of the next period's put price (i.e., in state *u*, $K - price(u) > p \cdot Max[price(du) - K, 0] + q \cdot Max[price(dd) - K, 0]$). Option C is a European lookback put, in which the owner receives the difference between the strike price and the lowest price the security reached on the path to the exercise point. Option D is a corresponding American lookback put. Finally, we use Options E and F to illustrate that an American call option will never be exercised early.

Table I - Four Period Example

Parameter	Value	Security	Description	Intrinsic Value Formula
K	50.00	A	European Put	$Max[0, K - S_3]$
u	1.04	B	American Put	$Max[0, K - S_t]$
d	0.98	C	European Lookback Put	$Max[0, K - Min(S_0, S_1, S_2, S_3)]$
r	1.02	D	American Lookback Put	$Max[0, K - Min(S_0, .., S_t)]$
p	0.6535	E	European Call	$Max[0, S_3 - K]$
q	0.3268	F	American Call	$Max[0, S_t - K]$

State	State Prices	A	B	C	D	E	F
today							
u	0.654						
d	0.327		*1.000*				
uu	0.427						
ud	0.214						
du	0.214		NA		*1.000*		
dd	0.107		NA				
uuu	0.279					6.243	6.243
uud	0.140					2.998	2.998
udu	0.140					2.998	2.998
udd	0.070	0.059	0.059	0.059	0.059		
duu	0.140		NA	1.000	NA	2.998	2.998
dud	0.070	0.059	NA	1.000	NA		
ddu	0.070	0.059	NA	1.980	1.980		
ddd	0.035	2.940	NA	2.940	2.940		
Price Today		**0.115**	**0.331**	**0.454**	**0.458**	**2.998**	**2.998**

The binomial option pricing model is an extension of the Black-Scholes results in the following sense: For appropriate choices of *up* and *down*, the binomial process underlying the stock price development in the binary option pricing model converges to a lognormal distribution (for specific parameters see the implementation section). It can be shown that the option price determined from the binomial option pricing model converges to the Black-Scholes option pricing formula.

There are two components to any option's value: the value associated with exercising the option (also called the intrinsic value), which we will denote by the term exercise value or EV, and the value associated with letting the option run on unexercised, denoted by option value or OV. (Note that a European option can only be exercised at maturity, so that it does not have an exercise value except at maturity). At any time, an option will trade for a price equal to the higher of these two components. Algebraically, the option price, Op is given by:

$$Op = \max[EV,\ OV] \tag{12}$$

At maturity, the option has only exercise value. Thus on the terminal nodes of the tree of stock prices, a call option will have a value equal to the maximum of zero or $S-K$ and a put will have a value equal to the greater of zero or $K-S$; where K is the strike price.

We can now price an option by proceeding recursively down through the tree of stock prices starting with the terminal nodes. The key result from the binomial model is that for one time step, when the stock can only tick up or down by a definite amount, we can construct a perfect hedge portfolio using the stock and a risk-free bond. The option can then be priced in the risk neutral world. The recursive formula for valuing any option would be:

$$OV(S_t,t) = Max\left[p \cdot OV(uS_t, t-\Delta t) + q \cdot OV(uS_t, t-\Delta t),\ EV(S_t,t)\right] \qquad t > 0$$

$$OV(S_0,0) = EV(S_0,0)$$

$$p = \frac{(r-d)}{r(u-d)}$$

$$q = \frac{1}{r} - p$$

$$r = e^{r\,\Delta t}$$

(13)

We will assume that the stock price follows the stochastic process defined in section 2, thus the distribution of stock prices at any time is lognormal[6]. If the stock price distribution generated by the process described earlier is to approximate the lognormal distribution then u and d should be related to the observed stock price volatility and to r, the risk-free, continuously compounded, interest rate.[7] While the parameter choice converging to the lognormal distribution is not unique (see Omber 1987), one convenient set of parameters is given by:

---

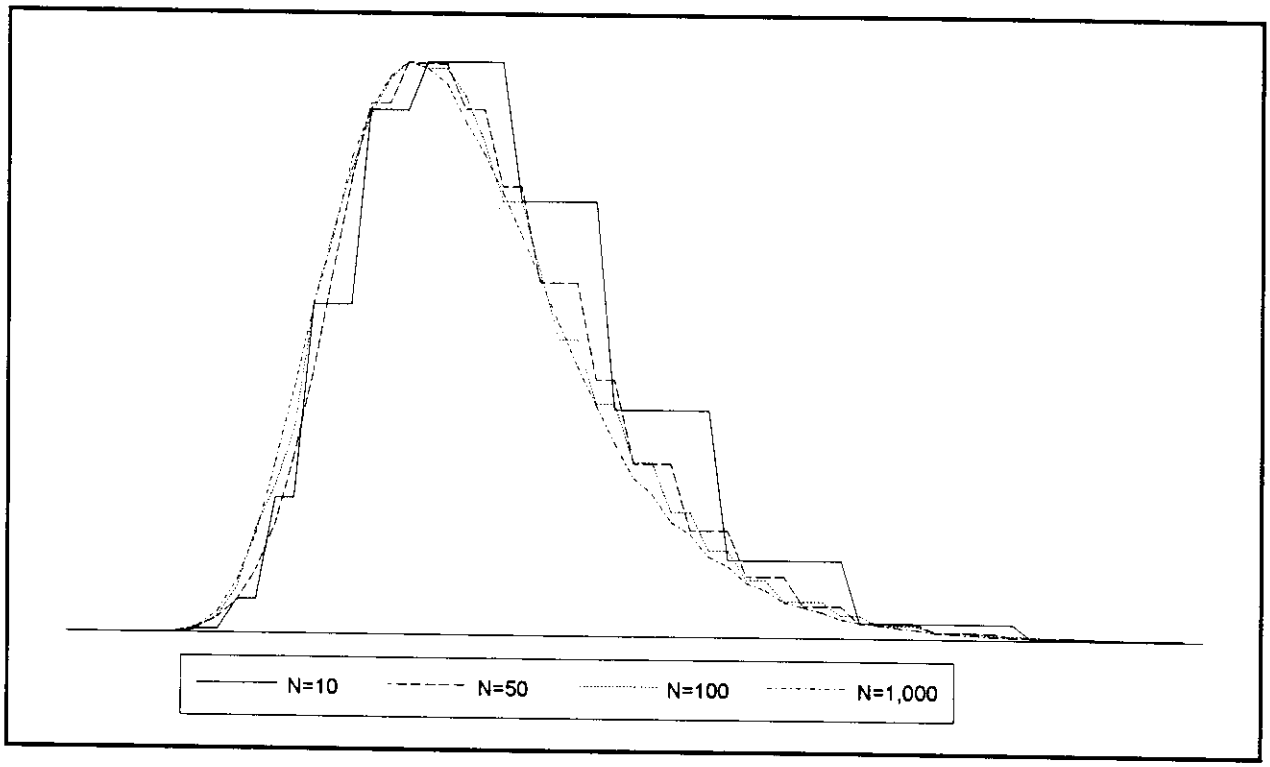
[6] For a discussion of the properties of the lognormal distribution, see J. Aitchison and J. A. C. Brown (1966).
[7] J.C. Hull (1993).

$$u = e^{\sigma\sqrt{\Delta t}}$$

(14) $$d = e^{-\sigma\sqrt{\Delta t}}$$

$$r = e^{r\,\Delta t}$$

Here Δt is the step time in the tree. The following graph illustrates the convergence of the distribution of the final nodes for various tree sizes:



4. Implementing binomial option pricing in *Mathematica*

In this section we present *Mathematica* solutions that can be used to value both American and European options of various kinds.

The first program will value any European style option that is not path dependent. Such options include the European calls and puts. The algorithm is non-recursive and uses the expectation of the option value method which is represented in the equation:

$$OV = e^{-rt}\sum_{i=0}^{n}\left\{\frac{n!}{i!(n-i)!}p^i q^{n-i} OE(u^i d^{n-i} S)\right\} \tag{15}$$

This equation simplifies the solution considerably, and can be easily implemented in *Mathematica*. We decided to separate the *OE* function from the main calculation routine in order to create a "black box" effect, and allow the user to value any option, without modifying the actual valuation. The most efficient way both in time and in programming style was to notice that the equation can be written as the dot product of two vectors, one consisting of the binomial expansion in p and q and the other is the option exercise value at each leaf (terminal node). The following *Mathematica* module does the valuation:

```
OV[S_,n_,sigma_,T_,Rf_,OEF_]:=Module[{p,q,up,down,r},
u=N[Exp[ Sqrt[T / n] sigma]];
d=N[1/up];
r=N[Exp[ Rf T / n]];
p=N[(r-down)/(up-down)/r];
q=N[1/r-p];

N[Map[OEF[#,True]&,Table[S up^j down^(n-j), {j,0,n}]] .
Table[Binomial[n,j]  p^j q^(n-j), {j,0,n}]]  ]
```

The parameters of the function are the current stock price **S**, the number of periods **n** to use, the standard deviation **sigma** of the stock prices, the time **T** to maturity (in years), the annualized continuously-compounded risk free interest rate **Rf**, and the option exercise function **OEF**.

The *OEF* is a function which takes two parameters: The stock price S, and a second parameter which is *True* if we are at the option expiration date, and *False* if the option is still alive (i.e., we are at a node on the tree prior to the option's expiration date). This definition of the *OEF* enables us to use the same definition for both the European option and American options (the following module).

Here is an example of valuation of an European put and call options with the same parameters as before:

```
In[1] :=
    K=50;
   putOE[S_,True]:=Max[K-S,0];
   putOE[S_,False]:=0;
   callOE[S_,True]:=Max[S-K,0];
   callOE[S_,False]:=0;

In[2] := OV[50,300,.4,5/12,.1,callOE]
Out[2] := 6.11227

In[3] := OV[50,300,.4,5/12,.1,putOE]
Out[3] := 4.07174
```

In section 4 we discuss the accuracy and the convergence properties of the results.

The next module will evaluate any American option. The routine is based on the same method described in section 2, but is presented here in a function form, with some efficiency improvements. Although the valuation is again based on the expected value of the option, we must value the option at every lattice point in order to check for early exercise. As we shall discuss in Section 4, American options take much longer to value.

```
OV1[S_,n_,sigma_,T_,Rf_,OEF_]:=Module[ {p,q,up,down,r},
ClearAll[OpRecurse];
up=N[Exp[sigma Sqrt[T / n]]];
down=N[1/up];
r=N[Exp[ Rf T / n]];
p=(r-d)/(u-d)/r;
q=1/r-p;

OpRecurse[node_,level_]:=OpRecurse[node,level]=
         If [ level==n, OEF[S down^node up^(level-node),True],
         Max[{p,q}.{OpRecurse[node,level+1],OpRecurse[node+1,level+1]},
             OEF[S down^node up^(level-node),False] ] ];

OpRecurse[0,0] ];
```

Using the similar option exercise value functions, here we define both American calls and puts, and we can use *OV1* to value them.

```
In[1] :=
putAOE[s_,__]:=Max[50-s,0];      (* American put *)
callAOE[s_,__]:=Max[s-50,0];     (* American call *)
OV1[50,30,.4,5/12,.1,putAOE]
Out[1] := 4.26343
In[2] := OV1[50,30,.4,5/12,.1,callAOE]
Out[2] := 6.07425
In[3] := OV[50,30,.4,5/12,.1,callOE]
Out[3] := 6.07425
```

Notice that the American call is worth the same as an European call, a result in option theory which we noted in Section 2.

The next module provides a solution for path dependent options. An example of such an option is an Asian average strike call options; this option gives its owner the right to buy a share of stock for the average price (either geometric or arithmetic) during the past month.[8] The European version of such options can be valued using the Monte Carlo techniques that we will discuss in a later section; but the American Asian option must be valued using binary trees. This means that the number of nodes in the tree is exponential, and thus only small trees can be valued in reasonable time.

To value such option, we use a similar technique as before. However, the definitions differ slightly. We define EV as the a function of a list of stock prices. $EV = f[s_0, s_1, s_2, \ldots . s_t]$, where t is the time that has elapsed since the option was bought. For example, and American average option will have the following EV

$$EV = \max\left[\left(\frac{1}{t}\sum_{k=0}^{t} s_k\right) - s_t, 0\right] \quad for\ t = 1,2,3\ldots n \tag{16}$$

Because of the binary nature of the underlying tree, we can define the following option pricing formula for any binary options:

$$\begin{aligned} &OV(s_0, s_1, \ldots, s_n) = EV(s_0, s_1, \ldots, s_n) \\ &OV(s_0, s_1, \ldots, s_k) = Max\left[EV(s_0, s_1, \ldots, s_n), p \cdot OV(s_0, .., s_k, u \cdot s_k) + q \cdot OV(s_0, .., s_k, d \cdot s_k)\right] \quad for\ k = 1..n-1 \end{aligned} \tag{17}$$

The formula was simply translated to *Mathematica* to create the following *OV2* path dependent option pricing function:

[8] For simplicity, we assume that the average is done on a continuous basis, and not over a defined set of prices (as it is done in reality).

```
OV2[S0_,n_,sigma_,T_,Rf_,OEF_]:=
Module[ {p,u,d,r},

Op[prices_List,level_]:=Module[{Cu,Cd,C},
      If [level==n,OEF[prices,True],                    (* leaf level *)
      C={Op[Append[prices,prices[[-1]]*u],level+1],
         Op[Append[prices,prices[[-1]]*d],level+1]};
      Max[p.C,OEF[prices,False]]]];

   u=N[Exp[sigma Sqrt[T / n]]];
   d=N[1/u];
   r=N[Exp[ Rf T / n]];
   p={(r-d),(u-r)} /(u r - d r);                        (* vector {p,q} *)

Op[{S0},0]]
```

As an example, here is a valuation of an European average price option which pays the difference between the final stock price and its arithmetic average over the period:

```
In[1]:=
      callavg[prices_,True]:=Max[0,Mean[prices]-prices[[-1]]]
      callavg[__,False]:=0
In[2]:=
      Mean[l_]:=Apply[Plus,l]/Length[l]      (* define the Mean function *)
In[3]:=
      OV2[50,14,.4,5/12,.1,callavg]
Out[3]:=
      2.42334
```

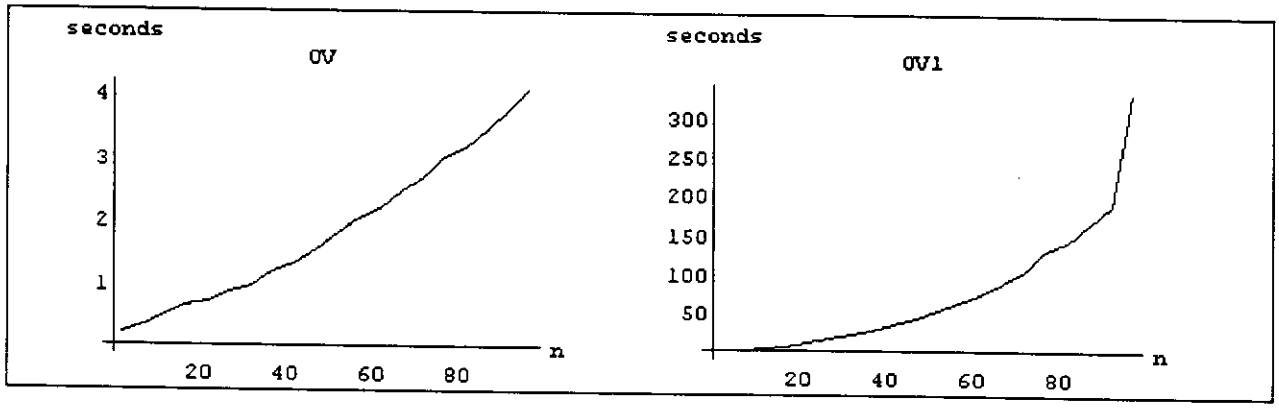
The module contains the functions that implement the recursive evaluation. The calculations are then performed to get the values for *u, d, r, p and q* and then the internal function is called with the root node of the tree. The result is a $o(2^n)$ calculation, which we could not do for more than 12 iterations on our computers, due to the time constraints. Notice that the algorithm has $o(n)$ memory requirements. Thus memory does not pose a limit here, but the computation time does. From our previous experience on the convergence of the non-path dependent options we can safely assume that the results obtained by low number of iterations are not accurate. However in the next section we will discuss a method for improving the accuracy of binomial approximations.

4. Timing and Convergence of the binary method

In this section we explore some of the numerical and timing problems of the previous algorithms.

Timing

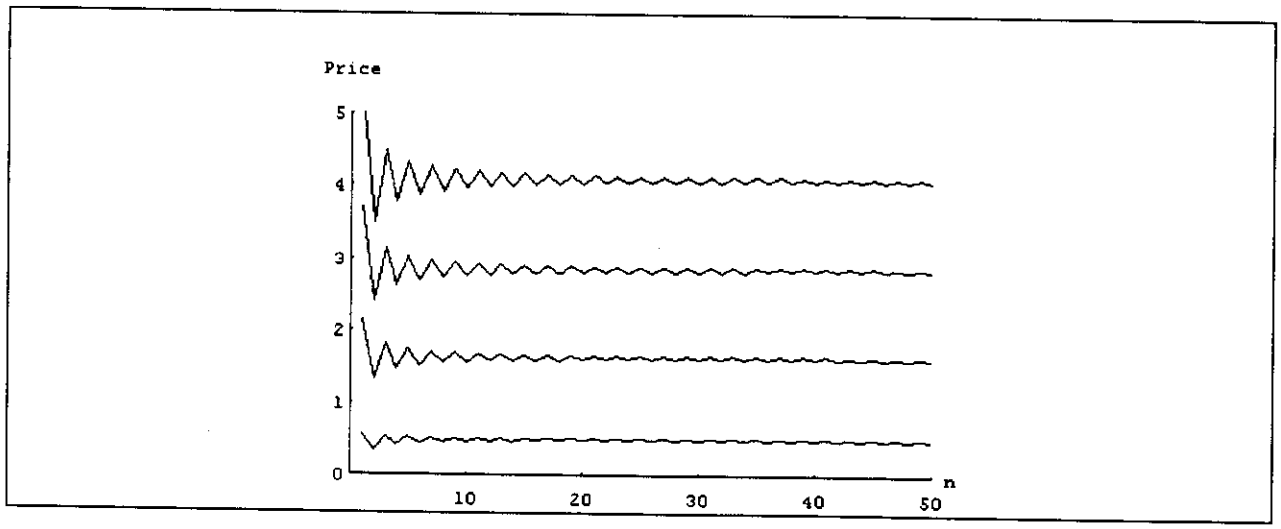
OV, the routine for European options, has a complexity of $o(n)$ while *OV1*, the program for American options, has a complexity of $o(n^2)$. This results in a much longer calculation time for the American options compared with the European options. We used *Mathematica* to plot the time it takes to evaluate each option:



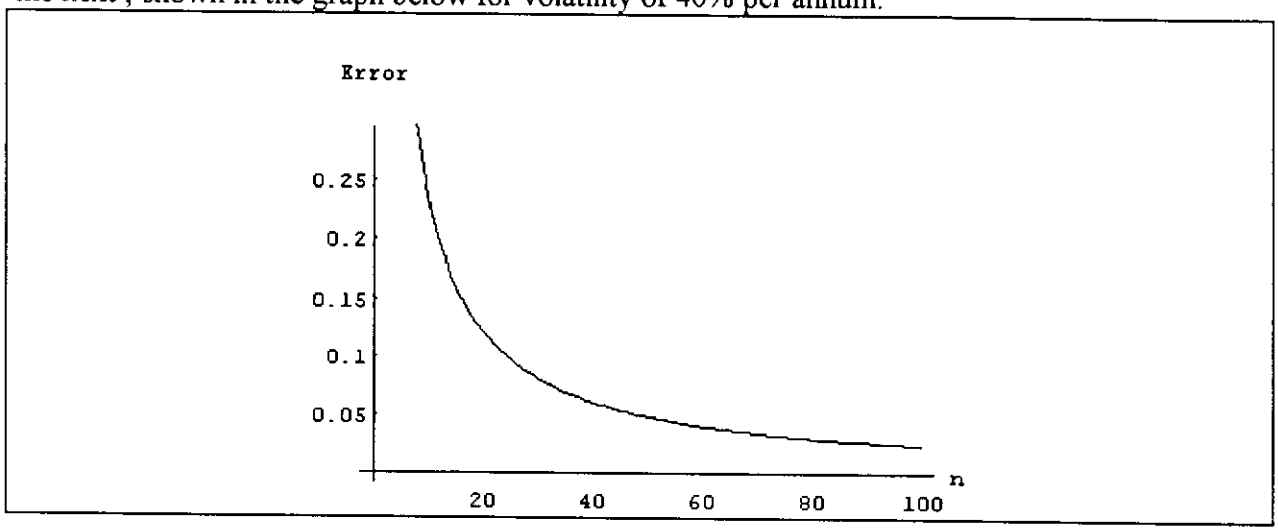
These times were achieved with *Mathematica 2.1* running under *Microsoft Windows* on a 386SX, 20 MHz, machine. Although one would be able to reduce those times using a faster CPU, the ratio between the two functions would not be changed. Because of the exponential nature of *OV2* we did not plot the timing; clearly, however, the user should not try trees bigger than about 15 levels.

Convergence

One interesting aspect of the binomial approximation is the speed of convergence to the final result. Here we used *OV* (due to its speed) to show how the results converge to the Black Scholes result. We used the same option example as before, but varied the volatility, which affects convergence the most. The following graphs shows the convergence for 10%, 20%, 30% and 40% volatility of the put option.



A further analysis can be done, by looking at the absolute change from each iteration to the next , shown in the graph below for volatility of 40% per annum.



Notice that even after 100 iterations, the result still changes by a few cents every iteration, which is rather disturbing. Applying those results to the American options, we can see that in order to be able to produce accurate results, a larger number of iterations must be performed, and the amount of time will be rather long, especially for more complex options.

In order to get a better approximation of the actual value, without going through the valuation of very large trees, we found that taking the geometric mean of two successive iteration results, provide a rather accurate estimation. However, we could not check that for every possible option (especially for path dependent ones), and this is by no means a general approximation. Another useful method, which can be implemented easily, is the control variate technique which is explain in the Monte Carlo section.

5. Valuation using the Monte Carlo Approach

The value at time zero of a European style option that pays off f_T at time T is :

(18) $$f = E\{f_T\, e^{-rT}\}$$

Where E denotes the expectation operator in a risk neutral world and r is the risk-free rate of interest. In general, f_T depends on the value of an underlying stochastic variable; usually this variable is the stock price or some simple function of the stock price such as the average over the last n days before maturity. If we make the simplifying assumption that the risk-free rate is known with certainty (as we did in the binomial models), we can rewrite equation (18) as,

(19) $$f = B_0 E\{f_T\} = e^{-rT} E\{f_T\},$$

where B_0 is the price of a zero-coupon discount bond that matures at time T.

The Monte Carlo technique approximates equation (19) by simulating the path of the stochastic variable in the risk-neutral world. By picking up enough random paths, we expect to achieve a large enough sample such that the mean of that sample will be close to the actual mean of the possible stock prices. In practice, approximately 5,000 runs are required to achieve option prices accurate to within a few cents.

The stock price process in the risk neutral world is :

(20) $$dS = S \cdot r\,dt + S \cdot \sigma dz$$

dz is a Wiener process and σ is the standard deviation of the stock's instantaneous return, m. In the risk neutral world, the stock's drift rate, m, has been replaced by the risk free interest rate, r.

The Monte Carlo approach has the benefit of allowing us not only to get an estimate of the option value, but to also get the standard deviation of the error of that estimate. Mathematica's list handling capability allows particularly simple and transparent programs to be written to implement the Monte Carlo approach.[9] The following Mathematica program implements the Monte Carlo valuation approach for a European option which depends only on the expiration date price of the stock.

```
<<Statistics`NormalDistribution`

monte[s_,n_,sigma_,T_,Rf_,OEF_]:= Module[ {m, sg, nor, tbl},
m=N[Log[s]+(Rf-1/2*sigma^2)*T];    (* Random mean *)
sg=N[sigma*Sqrt[T]];               (* Random Std. dev *)
nor:=Random[NormalDistribution[m,sg]];
tbl=Table[nor,{i,1,n}];
tbl=Map[OEF[#,True]&,Exp[Join[tbl,2*m-tbl]]]*Exp[-Rf*T];
{Mean[tbl],StandardErrorOfSampleMean[tbl]}]
```

Here is a simple application of the routine, to value the option we used in previous examples. Notice the result will be different from run to run, but at least some confidence interval can be established using the standard error ($0.40 in our case).

```
In[1]:=
    monte[50,300,.4,5/12,.1,callOE]
Out[1]:=
    {6.05954, 0.401488}
```

The parameters are similar to the binomial method, except that n must be much larger than the n required for the binomial approach. The result is a two numbered list where the first is the

[9] Unfortunately, Mathematica is not a statistical package, and thus lacks the speed which is required to generate larger samples.

estimated option price and the second is the standard error of that estimate. If n is large enough we can safely assume that the correct option price would lie within three standard errors from the estimate.

A few comments about this program are appropriate: in order to value a standard European option , we only require the distribution of terminal stock prices, not the complete path throughout time. If the stock price follows the process of equation (20) then it can be shown that the terminal stock price, in the risk neutral world, is lognormally distributed:

$$Ln(S_T / S) \sim N\left[(r - \frac{\sigma^2}{2})T\ , \sigma\sqrt{T}\ \right] \tag{21}$$

$N[\mu, \sigma]$ is the standard normal distribution with mean μ and standard deviation σ. It saves a considerable amount of computation time to simulate the terminal prices directly using equation 19 rather than the entire path using equation 20.

The Mathematica function `Random[]` allows us to generate random variables, distributed according to any distribution function. We used a method called *antithetic variable technique* in order to improve the accuracy of the result. The idea is to use the same sample of random numbers twice, each with different signs (relative to the mean), such that the total variance is reduced.[10]

While it is interesting to use the Monte Carlo approach to value standard European options, the prices of these options do have a closed form solution, or they can be accurately valued using numerical integration. The Monte Carlo approach is really useful for valuing options that do not have an analytic solution. In the following example we show how another type of average price Asian call may be valued. The pay-off from this option is $Max[0, \overline{S} - K]$ where $\overline{S}$ is the average price reached by the stock over the last 30 days before the option matures. The technique is to generate a random sample of price paths over the last 30 days before maturity, and to value the option over those paths. The following routine will evaluate an European Asian call function, but will have to be modified for other functions.

[10] Unfortunately those two samples are not perfectly negatively correlated because of the lognormal transformation, and the actual reduction in variance is only moderate

```
(* Function to return Accurate value of call option *)
Bsc[s_,k_,sigma_,T_,Rf_]:=Module[{d1,cnorm},
cnorm[x_]:=N[CDF[NormalDistribution[0,1],x]];
d1=(Log[s/k]+(Rf+sigma^2/2)T)/(sigma Sqrt[T]);
N[s*cnorm[d1]-k*Exp[-Rf*T]*cnorm[d1-sigma Sqrt[T]]]];

(* Asian option valuation function *)
Asian[s_,k_,sigma_,T1_,T_,Rf_,n_,navg_]:=
Module[{mdist,mpath,sdist,spath,path,endvals,bsestimate,callprice},
mdist=Log[s]+(Rf-sigma^2/2)*T1;      (* mean of start of path *)
sdist=sigma Sqrt[T1];                (* Stdev of start of path *)
mpath=1+Rf * (T-T1)/(navg-1);        (* mean of each path step *)
spath=sigma * Sqrt[(T-T1)/(navg-1)];(* Stdev of each path step *)
nor[mu_,sig_]:=Random[NormalDistribution[mu,sig]];
path=Table[NestList[#*nor[mpath,spath]&,
                    Exp[nor[mdist,sdist]],navg-1],{i,1,n}];
endvals=Map[#[[1]]&,path];
bsestimate=Mean[Map[N[Max[#-k,0]]&,endvals]]*Exp[-Rf*T1];
path=Map[N[Max[Mean[#]-k,0]]&,path]*Exp[-T*Rf];
callprice=Mean[path];
{callprice,StandardErrorOfSampleMean[path],
 callprice-(bsestimate-Bsc[s,k,sigma,T1,Rf])}]
```

The routine above accepts two additional parameters: *T1* is the time until the averaging period begins, and *T* is the maturity of the option. *navg* represent the number of averaging points in the period from *T1* until *T*.

The routine generates *n* random paths, each with *navg* numbers which are then averaged and the option is valued over those averages. In order to reduce the total error of the routine, we call another routine, Bsc, which calculates the exact value of simple call option. Then that result is compared to a simple call option which is valued over the same random numbers, in a similar way as the routine *Monte* valued options. Then a third number, the corrected estimate for the Asian option is returned. The formula for that number is $F_A = F_{MCA} + F_{BSE} - F_{MCE}$ where F_{MCA} is the initial Monte Carlo estimate for the price of the Asian option, F_{BSE} is the Black Scholes value for a standard European call and F_{MCE} is the Monte Carlo estimate for the price of the European call.

The routine returns a list of three numbers, where the first two are the option approximated value and the standard error of that estimate, and the third is the corrected value.

Here is a valuation of a call option with five months till maturity, and the strike price will be compared with the last month average price of the stock.

```
In[1]:=
   Asian[50,50,.4,4/12,5/12,.1,300,30]
Out[1]:=
   {5.41183, 0.472882, 5.68827}
```

Notice that such a call option is less valuable than a normal call option because the average tends to be less volatile compared with the actual stock price.

In general, we can notice that the Monte Carlo method can be used to give crude estimates of option prices. Because of the inability to value American type options, the need to develop a new program for each option type, and the relative speed (at least under *Mathematica*), we found this method to be inferior to the binomial approach. One of the main advantages of using the Monte Carlo method is the ability to price options using different stochastic assumptions, even observed probabilities, and to include multiple stochastic variables (such as interest rate and jump process) without the need to develop a formal model.

7. Conclusion

A number of recent papers discuss approximations to pricing formulas for difficult to price options.[11] In this paper we have shown that *Mathematica* may be used to price many of these options directly, using the parsimonious code which is its hallmark.

[11] See, for example, Kemna and Vorst (1990) or Ritchken, P., L. Sankarasubramanian, and A. M. Vijh (1993).

REFERENCES

Aitchison, J. and J. A. C. Brown (1966). *The Lognormal Distribution* (Cambridge: Cambridge University Press.

Black, F. and M. Scholes (1973). "The Pricing of Options and Corporate Liabilities." *Journal of Political Economy* 81 (May), pp. 659-673.

Cox, J., S. Ross, and M Rubinstein (1979). "Option Pricing: A Simplified Approach." *Journal of Financial Economics* 7 (October), pp. 229-264.

Cox, John and Mark Rubinstein (1985). "Option Markets." *Prentice Hall*

Hull, J. C. (1993). *Options, Futures, and other Derivative Securities*, Englewood Cliffs, NJ: Prentice-Hall.

Jarrow, R. A. and A. Rudd (1983). *Option Pricing*. Homewood, IL: Irwin.

Kemna, A. and T. Vorst (1990). "A Pricing Method for Options Based on Average Asset Values." *Journal of Banking and Finance* (March), pp. 113-129.

Merton, R. C. (1973). "Theory of Rational Option Pricing." *Bell Journal of Economics and Management Science* (Spring), pp. 141-183.

Miller, R. M. (1990) "Computer-Aided Financial Analysis: An Implementation of the Black-Scholes Model." *Mathematica Journal* 1 (Summer), pp. 75-79.

Omberg, E. (1987). "A Note on the Convergence of Binomial-Pricing and Compound-Option Models." *Journal of Finance* 42 (June), pp. 463-469.

Rendleman, R. and B. Bartter (1979). "Two State Option Pricing." *Journal of Finance* 34 (December), pp. 1093-1110.

Ritchken, P., L. Sankarasubramanian, and A. M. Vijh (1993). "The Valuation of Path Dependent Contracts on the Average." *Management Science*, forthcoming.