Barrier Options

- Their payoff depends on whether the underlying asset's price reaches a certain price level $H$.
- A knock-out option is an ordinary European option which ceases to exist if the barrier $H$ is reached by the price of its underlying asset.
- A call knock-out option is sometimes called a down-and-out option if $H < S$.
- A put knock-out option is sometimes called an up-and-out option when $H > S$.

\(^8\)A former student told me on March 26, 2004, she did not understand what I meant by barrier options until she started working in a bank.

---

Binomial Tree Algorithms

- Barrier options can be priced by binomial tree algorithms.
- Below is for the down-and-out option,

---

Barrier Options (concluded)

- A knock-in option comes into existence if a certain barrier is reached.
- A down-and-in option is a call knock-in option that comes into existence only when the barrier is reached and $H < S$.
- An up-and-in is a put knock-in option that comes into existence only when the barrier is reached and $H > S$.
- Formulas exist for all kinds of barrier options.

---

$S = 8$, $X = 6$, $H = 4$, $R = 1.25$, $u = 2$, and $d = 0.5$.

Backward-induction: $C = (0.5 \times C_u + 0.5 \times C_d) / 1.25$. 
Binomial Tree Algorithms (concluded)

- But convergence is erratic because $H$ is not at a price level on the tree (see plot on next page).
  - Typically, the barrier has to be adjusted to be at a price level.
- Solutions will be presented later.

Daily Monitoring

- Almost all barrier options monitor the barrier only for the daily closing prices.
- In that case, only nodes at the end of a day need to check for the barrier condition.
- We can even remove intraday nodes to create a multinomial tree.
  - A node is then followed by $d + 1$ nodes if each day is partitioned into $d$ periods,
- This saves time and space.

A Heptanomial Tree (6 Periods Per Day)
Foreign Currencies
- $S$ denotes the spot exchange rate in domestic/foreign terms.
- $\sigma$ denotes the volatility of the exchange rate.
- $r$ denotes the domestic interest rate.
- $\hat{r}$ denotes the foreign interest rate.
- A foreign currency is analogous to a stock paying a known dividend yield.
  - Foreign currencies pay a "continuous dividend yield" equal to $\hat{r}$ in the foreign currency.

Foreign Exchange Options
- Foreign exchange options are settled via delivery of the underlying currency.
- A primary use of foreign exchange (or forex) options is to hedge currency risk.
- Consider a U.S. company expecting to receive 100 million Japanese yen in March 2000.
- Those 100 million Japanese yen will be exchanged for U.S. dollars.

Foreign Exchange Options (continued)
- The contract size for the Japanese yen option is JPY6,250,000.
- The company purchases 100,000,000/6,250,000 = 16 puts on the Japanese yen with a strike price of $0.0088 and an exercise month in March 2000.
- This gives the company the right to sell 100,000,000 Japanese yen for 100,000,000 x .0088 = 880,000 U.S. dollars.

Foreign Exchange Options (concluded)
- The formulas derived for stock index options in Eqs. (26) on p. 258 apply with the dividend yield equal to $\hat{r}$:
  \[ C = S e^{\hat{r} r} N(x) \quad X e^{\hat{r} r} N(x + \sigma \sqrt{r}) \],
  \[ P = X e^{\hat{r} r} N( x + \sigma \sqrt{r} ) \quad S e^{\hat{r} r} N( x ) \],
- where
  \[ x \equiv \ln(S/X) + (r - \hat{r} + \sigma^2/2) r / \sigma \sqrt{r} . \]
Path-Dependent Derivatives

- Let $S_0, S_1, \ldots, S_n$ denote the prices of the underlying asset over the life of the option.
- $S_0$ is the known price at time zero.
- $S_n$ is the price at expiration.
- The standard European call has a terminal value depending only on the last price, $\max(S_n - X, 0)$.
- Its value thus depends only on the underlying asset’s terminal price regardless of how it gets there.

Path-Dependent Derivatives (continued)

- In contrast, some derivatives are path-dependent in that their terminal payoff depends “critically” on the path.
- The (arithmetic) average-rate call has a terminal value given by
  \[
  \max \left( \frac{1}{n + 1} \sum_{i=0}^{n} S_i - X, 0 \right).
  \]
- The average-rate put’s terminal value is given by
  \[
  \max \left( X - \frac{1}{n + 1} \sum_{i=0}^{n} S_i, 0 \right).
  \]

Path-Dependent Derivatives (continued)

- Average-rate options are also called Asian options.
- They are useful hedging tools for firms that will make a stream of purchases over a time period because the costs are likely to be linked to the average price.
- They are mostly European,
Path-Dependent Derivatives (concluded)

- A lookback call option on the minimum has a terminal payoff of $S_n \min_{0 \leq i \leq n} S_i$.
- A lookback put option on the maximum has a terminal payoff of $\max_{0 \leq i \leq n} S_i - S_n$.
- The fixed-strike lookback option provides a payoff of
  $\max(\max_{0 \leq i \leq n} S_i - X, 0)$ for the call and
  $\max(X - \min_{0 \leq i \leq n} S_i, 0)$ for the put.
- Lookback call and put options on the average are called
  average-strike options.

Average-Rate Options

- Average-rate options are notoriously hard to price.
- The binomial tree for the averages does not combine,
- A straightforward algorithm is to enumerate the $2^n$ price paths for an $n$-period binomial tree and then
  average the payoffs.
- But the exponential complexity makes this naive
  algorithm impractical,
- As a result, the Monte Carlo method and approximation
  algorithms are some of the alternatives left.

Pricing Some Path-Dependent Options

- Not all path-dependent derivatives are hard to price,
- When averaging is done geometrically, the option payoffs are
  $$\max \left( (S_0 S_1 \cdots S_n)^{1/(n+1)} - X, 0 \right),$$
  $$\max \left( X - (S_0 S_1 \cdots S_n)^{1/(n+1)}, 0 \right).$$
Pricing Some Path-Dependent Options (concluded)

- The limiting analytical solutions are the Black-Scholes formulas.
  - With the volatility set to $\sigma_n \equiv \sigma / \sqrt{3}$.
  - With the dividend yield set to $q_n \equiv (r + q + \sigma^2 / 6) / 2$.
- The formula is therefore

$$C = S e^{q_n \tau} N(x) - X e^{r \tau} N(x - \sigma_n \sqrt{\tau}),$$
$$P = X e^{r \tau} N \left( x + \sigma_n \sqrt{\tau} \right) - S e^{q_n \tau} N \left( x \right),$$

- where $x \equiv \frac{\ln(S/X) + (r - q + \sigma_n^2 / 2) \tau}{\sigma_n \sqrt{\tau}}$.

Approximation Algorithm for Asian Options

- Based on the BOPM.
- Consider a node at time $j$ with the underlying asset price equal to $S_0 u_j^{i+1} d^i$.
- Name such a node $N(j, i)$.
- The running sum $\sum_{m=0}^j S_m$ at this node has a maximum value of

$$S_0 \left( 1 + u + u^2 + \cdots + u^j + u^j \frac{i}{i+1} d \right) = S_0 \frac{1}{u} u_j^{i+1} + S_0 u_j^{i+1} d \frac{1}{d} d^i.$$

Approximation Algorithm for Asian Options (continued)

- Divide this value by $j + 1$ and call it $A_{\text{max}}(j, i)$.
- Similarly, the running sum has a minimum value of

$$S_0 \left( 1 + \frac{d}{1} + \frac{d}{1} + \cdots + \frac{d}{1} u + \cdots + d^i u \right)^{j} = S_0 \frac{1}{d} d^j + S_0 d^j u \frac{1}{u} u^i.$$
- Divide this value by $j + 1$ and call it $A_{\text{min}}(j, i)$.
- $A_{\text{min}}$ and $A_{\text{max}}$ are running averages.
Approximation Algorithm for Asian Options (continued)

- The possible running averages at $N(j, i)$ are far too many: $\binom{j}{i}$.
- But all lie between $A_{\min}(j, i)$ and $A_{\max}(j, i)$.
- Pick $k + 1$ equally spaced values in this range and treat them as the true and only running averages:

$$A_m(j, i) \equiv \left(\frac{k - m}{k}\right) A_{\min}(j, i) + \left(\frac{m}{k}\right) A_{\max}(j, i)$$

for $m = 0, 1, \ldots, k$.

---

Approximation Algorithm for Asian Options (continued)

- Backward induction calculates the option values at each node for the $k + 1$ running averages.
- Suppose the current node is $N(j + 1, i)$ and the running average is $a$.
- Assume the next node is $N(j + 1, i)$, after an up move.
- As the asset price there is $S_0u^{j+1}d^i$, we seek the option value corresponding to the running average

$$A_u \equiv \frac{(j + 1) a + S_0u^{j+1}d^i}{j + 2}.$$  

---

Approximation Algorithm for Asian Options (continued)

- But $A_u$ is not likely to be one of the $k + 1$ running averages at $N(j + 1, i)$!
- Find the running averages that bracket it, that is,

$$A_{\ell}(j + 1, i) \leq A_u \leq A_{\ell+1}(j + 1, i),$$

- Express $A_u$ as a linearly interpolated value of the two running averages,

$$A_u = x A_{\ell}(j + 1, i) + (1 - x) A_{\ell+1}(j + 1, i), \quad 0 \leq x \leq 1.$$
Approximation Algorithm for Asian Options (continued)

- Obtain the approximate option value given the running average $A_u$ via
  \[ C_u \equiv x C_{\ell}(j + 1, i) + (1 - x) C_{\ell+1}(j + 1, i). \]
  $C_\ell(t, s)$ denotes the option value at node $N(t, s)$ with running average $A_\ell(t, s)$.

- This interpolation introduces the second source of error.

Approximation Algorithm for Asian Options (continued)

- The same steps are repeated for the down node $N(j + 1, i + 1)$ to obtain another approximate option value $C_d$.

- Finally obtain the option value as
  \[ [pC_u + (1 - p)C_d] e^{-r\Delta t}. \]
- The running time is $O(kn^2)$.
- There are $O(n^2)$ nodes.
- Each node has $O(k)$ buckets.

Approximation Algorithm for Asian Options (concluded)

- Arithmetic average-rate options were assumed to be newly issued: There was no historical average to deal with.

- This problem can be easily dealt with (see text).

- How about the Greeks?\(^a\)

\(^a\)Thanks to a lively class discussion on March 31, 2004.
A Numerical Example

- Consider a European arithmetic average-rate call with strike price 50.
- Assume zero interest rate in order to dispense with discounting.
- The minimum running average at node A in the figure on p. 350 is 48.925.
- The maximum running average at node A in the same figure is 51.149.

A Numerical Example (continued)

- Each node picks $k = 3$ for 4 equally spaced running averages.
- The same calculations are done for node A's successor nodes B and C,
- Suppose node A is 2 periods from the root node.
- Consider the up move from node A with running average 49.666.

A Numerical Example (continued)

- Because the stock price at node B is 53.447, the new running average will be
  \[
  \frac{3 \times 49.666 + 53.447}{4} \approx 50.612.
  \]
- With 50.612 lying between 50.056 and 51.206 at node B, we solve
  \[
  50.612 = x \times 50.056 + (1-x) \times 51.206
  \]
  to obtain $x \approx 0.517$. 
A Numerical Example (continued)

- The option values corresponding to running averages 50.056 and 51.206 at node B are 0.056 and 1.206, respectively.
- Their contribution to the option value corresponding to running average 49.666 at node A is weighted linearly as
  \[ x \times 0.056 + (1 - x) \times 1.206 \approx 0.611. \]

A Numerical Example (concluded)

- The option values corresponding to running averages 47.903 and 48.979 at node C are both 0.0.
- Their contribution to the option value corresponding to running average 49.666 at node A is 0.0.
- Finally, the option value corresponding to running average 49.666 at node A equals
  \[ p \times 0.611 + (1 - p) \times 0.0 \approx 0.2956, \]
  where \( p = 0.483 \).
- The remaining three option values at node A can be computed similarly.

Remarks on Asian Option Pricing

- Asian option pricing is an active research area.
- The above algorithm overestimates the “true” value,\(^a\)
- To guarantee convergence, \( k \) needs to grow with \( n \).
- Analytical approximations for European Asian options exist.
- There is a convergent approximation algorithm that does away with interpolation with a provable running time of \( 2^\sqrt{n} \).\(^b\)

\(^a\)Dai, Huang, and Lyuu (2002).
\(^b\)Dai and Lyuu (2002, 2004),
Remarks on Asian Option Pricing (continued)

- There is an $O(kn^2)$-time algorithm with an error bound of $O(Xn/k)$ from the naive $O(2^n)$-time binomial tree algorithm in the case of European Asian options.\(^{a}\)
  - $k$ can be varied for trade-off between time and accuracy.
  - So if we pick $k = O(n^2)$, then the error is $O(1/n)$, and the running time is $O(n^4)$.
- In practice, log-linear interpolation works better,

\(^{a}\)Angworth, Motwani, and Oldham (2000).

Remarks on Asian Option Pricing (concluded)

- Another approximation algorithm reduces the error to $O(X\sqrt{n}/k)$.\(^{a}\)
  - It varies the number of buckets per node.
  - If we pick $k$ proportional to $n$, the error is $O(n^{-0.5})$.
  - So if we pick $k = O(n^{1.5})$, then the error is $O(1/n)$, and the running time is $O(n^{3.5})$.
- Under some “reasonable assumptions,” Hsu and Lyuu (2004) produce an $O(n^2)$-time algorithm with an error bound of $O(1/n)$.

\(^{a}\)Dai, Huang, and Lyuu (2002).